THE DEVELOPMENT OF AN EDUCATIONAL COMPUTER AIDED DESIGN SYSTEM FOR GARMENT MANUFACTURE

Volume 1

Submitted in accordance with the requirements for
the degree of
Doctor of Philosophy

by
Jin-sook Jo

The Department of Textile Industries
The University of Leeds
Leeds LS2 9JT

August 1989

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation and gratitude to Dr. S.C. Harlock for all his guidance, advice, encouragement and caring ever since I came to Leeds.

I am very grateful to Dr. G.A.V. Leaf for his help and understanding during the later period of the preparation of this thesis.

I am also very grateful to all the staff members of the Textile Industries Department for their academic and technical assistance.

I wish to thank Mrs. C. Scargill for her technical advice and help. I also wish to thank Ms. D. Barry for her comments and discussions.

Finally, I wish to express my special gratitude to Cheju National University in Korea, which has enabled me to complete this research course.

ABSTRACT

A computer aided design (CAD) system for the design of flat patterns for female clothing has been developed specifically for use in an Educational environment. This Pattern Making System (PMS) has been developed using a commercial computer aided design software system, CADDS 4X, originally marketed for Mechanical Engineering applications by the Computervision division of Prime Plc. The software operates on a Sun 3 computer Workstation under a Unix operating system environment.

The features of the PMS include software programs, written using CVMAC, the macroprogramming language of the host CAD system, for the creation of basic blocks for many styles of garments to suit any size of female body within a specified size chart or user defined anthropometric data. Also included are software programs for the design of auxiliary patterns such as collars, cuffs etc., together with programs to proportionally grade patterns to different sizes. These programs have been compiled with appropriate CADDS 4X graphics functions and structured into an icon menu driven shell to form the PMS.

The main emphasis in developing the PMS was to make it suitable for an Educational environment. Therefore an extensive set of demonstration programs and on-line help

files has been written and incorporated into the PMS to assist the user. These help files explain the details of the PMS and the CADDS environment, the functions of the programs and the graphics commands and give guidance in their use for the novice user. Furthermore, a series of explicit tutorials have been written so that inexperienced users can self-teach themselves in the use of the PMS for pattern design so that the PMS also forms a system for computer based learning.

The intention and hope is that, through the use of this tutorial based PMS, students will gain considerable insight into the use of a CAD system for pattern design which they can readily transfer to other commercial systems.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1. Computers in the Clothing Industry

After many decades of manufacturer-led mass production, the late 1970's and 1980's have seen attitudes change towards clothes. There has been an increasing demand for more choice and for clothes which are compatible with a consumer's individual life style. This consumer-led style change has resulted in more diverse design, faster fashion cycles and shorter production runs. In response to this trend, more emphasis has been placed on quick response, flexibility and efficiency of manufacturing to maintain competitiveness [1] [2] [3] [4] [5].

Furthermore, throughout the 1980's, the U.K. clothing industry has faced increasing shortages of skilled labour, pressure from competitive imports and rising energy costs. Aware of the potential benefits of computerisation such as speed, accuracy, consistency and flexibility, progressive manufacturers who have the financial status and volume of production to justify it have installed more and more computer systems to improve production efficiency. More recently, with the development of many smaller systems using personal computers and the falling price of software as well

as hardware, small or medium sized clothing manufacturers have started to install and up-grade computer systems [4] [6] [7] [8] [9].

Computer systems are available for use in many parts of the clothing industry. For management, computers are used for order planning, purchase control, payroll, invoice and shipping, inventory control, reports and statistics and marketing. For design and pattern making, Computer Aided Design (CAD) systems have evolved to fulfil functions including sketching, pattern designing, grading and marker making. Also Computer Aided Manufacturing (CAM) systems are used for cutting, sewing, finishing, material handling, production control and balancing and production monitoring [10] [11] [12] [13].

Most of the CAD systems developed for sketching and pattern making are based on a two-dimensional representation and manipulation of designs and patterns. However for better fit and design creativity, efforts are currently being directed towards developing advanced systems which can create a pattern piece from a three-dimensional form [14] [15].

1.2. The Need for an Educational CAD System

The CAD systems developed for use in industry place considerable emphasis on an understanding of the functions and capabilities of those systems, with operators requiring

extensive training and experience before they can use the system proficiently. Even though the systems are designed to be used by non-computer specialists, novices (particularly clothing/textile students) need the opportunity to experience CAD skills practically and to comprehend the basic concepts of CAD systems before entering industry. The systems currently available do not lend themselves readily to this. There is therefore a need for a CAD system to be developed for educational use which can serve as a tutorial-computer aided learning system at the same time as fulfilling the required design functions [16] [17].

This project has sought to fulfill this need by developing a flat pattern making system, which can be used in a training or an educational environment. A CAD system is basically a tool for designers. To be accepted by designers, a primary requirement is that it should be user-friendly so that they feel comfortable with the system and can use it efficiently. Therefore the primary objective of the research was to develop a pattern making system with appropriate tutorial and help functions which would be suitable for use in a teaching environment.

A second, but no less important objective was to provide the platform for a concurrent, separate research program in advanced pattern design which had particular emphasis on three-dimensional pattern design. This influenced the overall approach to tackling the problem and the creation of the CAD system.

The thesis consists of three parts. Part I is a description and explanation of how the system has been developed, Part II is the on-line help system and Part III comprises the tutorials which are in a separate volume.

CHAPTER 2

REVIEW OF COMMERCIAL COMPUTER AIDED DESIGN SYSTEMS

Throughout the clothing industry a great diversity of garments for different end uses are manufactured by many production organizations of varying size and complexity. In dealing with frequent changes in styles and fabrics together with the seasonal nature of the industry, clothing manufacturers face many problems. They have to plan, monitor and control design, development, pre-production, production and distribution to satisfy the demand, whilst simultaneously reducing standard costs and excess costs and increasing flexibility and improving response [2] [5] [15].

Anticipating that future competition will intensify, with the demand for ever-shorter lead times and greater variety in their product range, many companies have installed CAD systems to improve their efficiency. The benefits of CAD systems are increased personnel productivity and its consequent savings in labour, improved material utilization, reduction in overall throughput time, increased product diversity and flexibility, as well as providing the ability to supervise and verify every process at each stage [18] [19] [20].

Basically, CAD systems consist of an illustration system, a pattern input system, a pattern design system and a grading and marker making system. Systems are either modular or expandable, so that when a manufacturer starts implementing a system he can begin with a minimum amount of equipment and later the system can be added to or expanded as required [4] [10].

2.1. Illustration Systems

Illustration systems were originally intended to serve as visual communication between designer, marketing staff and customer in order to reduce the cost and effort of producing many sample garments which are often wasted. Therefore illustration systems, through realistic representation of designs, concentrate choice into a smaller selection for subsequent sampling and thereby reduce lead time. Another purpose of the illustration system is to help the designer to experiment with different trends, colours and fabric patterns which can be changed on the computer screen easily by using simulated pen, brush and wash techniques [21].

The current major suppliers of illustration systems are listed below.

Assyst GmbH

CDI (Computer Design, Inc.)

Cybrid Ltd.

Gerber Camsco, Inc.

Investronica

Lectra Systemes

Microdynamics Inc.

Pfaff GmbH

Most of their systems have similar facilities for freehand drawing and editing the garment design or fashion images using a colour monitor, light pen, data tablet and colour printer. Most of the systems allow a designer to use up to 250 colours at one time and select from a colour pallet of 16 million colours. Face and figure photographs can be input by a camera or a scanner which can be modified using simulated painting techniques. Similarly, fabric designs can be input by a camera or a scanner. The fabric designs can be scaled down to the proportionate size, to be married with photographed or sketched figures in the computer's memory. This facility can provide a realistic picture of how the fabric will appear when made up.

Some systems have advanced facilities to make more realistic images. Gerber's Creative Designer provides three-dimension simulation facilities, which allows the operator to use shading techniques to create for example, folds in a draped skirt, fullness for topcoats and gathering in blousons, giving a three-dimensional impression to the illustration. CDI has developed a three-dimension garment illustration system using NURBS (Non Uniform Rational B.Splines). NURBS is a new technique to illustrate a contour smoothly by using small lines, particularly in curves [21].

2.2. Pattern Input Systems

A pattern input system is a means to input graphic data from a flat pattern into a CAD system. Currently used pattern input systems use either a digitiser or a scanner [21]. Depending on the modules within a CAD system, the patterns at a certain stage of development have to be input to the CAD system. For example, if a system comprises only a grading and marker making system and a pattern input system, a manually produced production pattern has to be input. If a system has a pattern design system, a grading and marker making system and a pattern input system, then a manually made block pattern has to be input. However, if the pattern design system has the facility to create a block pattern using graphics functions, a pattern input system is not strictly necessary. However it is useful, from a commercial viewpoint, to input patterns that may have been created manually prior to the purchase of the CAD system.

The digitiser consists of a digitizer table, on which there is a freely moving cursor with communication buttons, number matrices, alphanumeric keyboard matrices and command buttons. In order to input the pattern shape a digitiser inputs a series of x and y coordinates of endpoints or notch points of the pattern together with instructions on how to connect each point such as a line or a curve, in order to regenerate the pattern within the computer system. Further information which can be input depends on the particular system. For example, on the Gerber AM-5 system,

grading points, alteration points, stripe and check matching points, notches and internal points may also be input.

Pattern scanners first scan pattern pieces and then input the information to the system and are speedy and accurate because of the elimination of point-by-point digitising. In the pattern scanning system produced by Lectra Systemes the pattern parts are laid in any order on a flat bed and scanned by video cameras which automatically enter the pattern shapes into the system. With the pattern scanning system produced by Cybrid Ltd. pattern parts are laid on the scanner's table surface up to 15 pattern parts at a time depending on their size. A taut wire grid frame serves to hold the patterns flat and enables the operator to arrange them in the correct grain direction. When the scanner works, each pattern part appears on the screen in turn as the scanning proceeds [21].

2.3. Pattern Design Systems

Pattern design systems are supplied by the major CAD system developers listed earlier (See 2.1. Illustration Systems). Because the aims or the processes of pattern making vary among the clothing manufacturers depending on the end product and the target consumer, the pattern design system suppliers have developed various facilities such as basic block pattern generation, secondary pattern generation, made-to-measure pattern making as well as pattern manipulation.

The "Assycad" developed by Assyst GmbH, has the facility to draft front, back and sleeve block patterns from model measurements, with computerised drafting instructions for each model and style type. Style patterns can be made from these engineered block patterns by the usual manipulation to produce the style features required, such as converting a set-in sleeve and armhole into raglan style [22].

The "PAR" system developed by Pfaff GmbH, has a similar facility. In this system the pattern maker enters his rules for basic pattern construction into a data base. By applying body measurements the computer generates the pattern. As a complete set of measurements for the company's size scales are in the data base, all other sizes are constructed automatically. New styles are created either by changing the construction description or if necessary, interactively on the computer screen. Measurements may be body measurements, finished goods or allowance measurements [21]. This facility saves a considerable amount of time because it eliminates the need to digitise manually created block patterns.

Most of the pattern manipulation functions of the different systems are similar. They allow the pattern maker to work on several pattern parts at the same time, marrying them together in order to engineer the complete style. Patterns can be checked by the computer, measuring from one point to another to ensure that no errors have been made in the length of the garment. Similarly, curves can also be measured and checked [23].

Additional facilities include: moving darts, notch points and lines, splitting seams, copying patterns or making a mirror image of the patterns and adding seam allowance.

The Pattern Generation system produced by Gerber Camsco Inc. automatically generates secondary patterns for linings, interlinings, fusibles and facings from the pattern data.

Investronica, Gerber Camsco Inc. and Lectra Systemes have developed made-to-measure pattern making systems with possibilities for remote entry data e.g. directly from the point of sale, handling asymmetrical alterations and complex amendments to deal with difficult body shapes [21].

2.4. Grading and Marker Making Systems

Most grading systems require a set of grading rules for the grading points. The grading rule can be newly input or recalled from a grading rule library. According to the grade rule, a system will grade patterns automatically. Recently, some systems have developed alternative ways of grading patterns to make grading simpler and more efficient. Using the Cybrid system, the pattern pieces can be stretched (shrunk) either along or across the body. The Assyst grading system has a grading method called "parallel grading" whereby one or more points are moved a parallel distance from an existing line. This is important in areas such as the shoulder seam where the offset is known, but not the x and y coordinates.

There are two ways of carrying out marker making. One is interactive and the other is automatic. Automatic marker making allows the system to automatically generate markers unattended by an operator and without the use of the graphic display terminal. In this mode, the system uses previously stored marker generation information to produce new markers as requested by the order input information. The system automatically tries different ways to make the marker, selecting the most efficient marker for storage and display. Data on marker width, length, marker utilization, and the marker number are available as in the case for interactively generated markers. For automatic marker making, the user may specify a minimum acceptable utilization. Automatically made markers failing to meet or exceed this criterion are not stored. Instead, a listing of such markers is retained so that an operator can, at a later time, interactively prepare, store, and output the markers.

Under certain conditions of material width, garment size, number of sizes and use of certain marker making techniques, it will be necessary to prepare the marker manually. Typically a system will present all the necessary patterns and display them on the graphics terminal. By using the data tablet and pen, the operator can place the pattern in the desired positions within the marker. During and at the conclusion of the marker making process, an alpha numeric display shows the operator data (e.g. marker length and efficiency) for the marker being made. All constraints for a given marker type e.g. rotational possibility of

pattern pieces from their reference line, buffering allowance around a piece or part of a piece will automatically be adhered to during this procedure on completion of a satisfactory marker, the operator can store this for future use e.g. plotting, cutting.

Most systems also have facilities for checking matching points or stripes. This provides a method of controlling marker construction to ensure patterns match with each other and/or to the fabric as required. The data representing the theoretical geometry of the fabric can be input with the order.

2.5. Current Research and Development of CAD systems

Current research and development related to the CAD systems for the clothing industry can be divided into two categories. One is concerned with increasing the efficiency, flexibility and reliability of the systems with the basic concept of the CAD system continuing to be based on the more conventional way of garment production. The other is three dimensional pattern design with simulation of the garment appearance to produce the garment patterns directly from the three dimensional design.

One example of trying to increase the efficient utilisation of the CAD system is to develop an educational system so that the user can use the CAD system more efficiently. A recent example of this is the Ormus-Fashion

system which is designed to give students the opportunity to experience a CAD system similar to those used in the industry. This system can be used for a variety of operations, such as fashion sketching, pattern making, grading and lay planning for costing. Also, for educational purposes, the system is relatively low-priced and compatible with many personal computers such as IBM or IBM compatibles.

Most of the CAD suppliers are making continuing efforts to improve their systems in various ways such as:-

* improving the graphic display,
* developing more flexible facilities for pattern manipulation,
* developing more easy-to-use and versatile grading systems,
* making the system more user-friendly [10] [11] [14] [15] [21].

In order to develop a three-dimensional design and pattern making system, the relevant basic research and developments are prerequisite. New types of data on body measurements are required to simulate appearance of the body figure in the computer. The data should have the three dimensional information, which can be transformed into x, y, and z coordinates. Recent research has focused on finding out the correct and practical method for getting the data of the figure [24]. Work at the Japanese Research Institute for Polymers and Textiles has been concerned with many aspects of the problem of using computers to describe clothes [25] [26] [27] [28] [29] [30]. Also, the relation

between a flat pattern and the three dimensional form which is formed by the flat pattern has been studied, and a CAD system for three dimensional design has already been developed [31] [32]. It is claimed that this system can display the appearance of a dress superimposed on a body by inputting the paper pattern for the dress, mechanical characteristics of the cloth and the shape of the body.

CHAPTER 3

THE PATTERN MAKING SYSTEM - AN OVERVIEW

3.1. Introduction

As mentioned in chapter 1, the purpose of this work was to develop a Computer Aided Design (CAD) system for garment and pattern design that was specially suited to educational needs. It was realised that such a broad objective was too ambitious to be completed in a single research project. Therefore the objectives were narrowed to a more attainable level with the expectation that refinements and enhancements would be made in subsequent projects.

From studying commercially available systems, the main criticism, from an educational standpoint, is the lack of guidance given to the operator in the use of the system. Apart from an initial training course and manuals supplied with the system there appeared to be little on-line (i.e. accessible at the computer terminal) information and assistance in how to use the system, which functions to use, how to adapt patterns using a computer etc. It is acknowledged that commercial systems are designed in this way for operatives who are experienced in pattern design who will use the system regularly and soon become familiar with its features. However, in an educational environment,

students are novices in pattern design, many are computer illiterate and, more importantly, will only have intermittent access to such a CAD system and are unlikely, therefore to become readily familiar with the system. Therefore they will require considerable assistance to learn how to use the system. Furthermore, diminishing academic resources emphasise the need to use resources as efficiently as possible and a move towards computer based learning can be beneficial in this respect, provided the systems are compatible with existing teaching methods.

Thus the objectives set for this work were to develop a CAD system for pattern design, adaptation and grading. The system should include basic functions to achieve these tasks, but of equal importance, it should offer as much assistance as possible to the user in the use of the system. It was proposed that the assistance should be in the form of on-line files and demonstration programs with tutorials for direct guidance. The proposed user of the system was expected to have a fundamental knowledge of pattern design but only a limited knowledge (if any) of computers and the user of a CAD system in particular. It was acknowledged that it would be difficult to incorporate as comprehensive a range of functions as desired, or as on a commercial system, for example. However it was considered important and sufficient to establish the minimal requirements as a basis for a system which could be further developed through successive projects.

In developing such a system, it was recognised that many graphics routines and functions would be required to create the pattern parts. Since the purpose was to develop an overall system, it was considered that to write appropriate graphics routines would be unproductive since many routines were available through commercial software programs. Therefore, rather than "re-invent the wheel", it was decided to build the system on the foundation of an existing computer aided design system. Many systems are available, each with different and distinctive features, however the system produced by Computervision Inc. (now Prime Plc.) was selected. The reasons for the selection were several, however the decisive factors were:

1. The system offered, at that time, the most comprehensive range of two and three dimensional graphics routines for use in Mechanical Engineering drafting which is similar to the techniques used in pattern design.

2. The system was Unix operating system based and was therefore portable to other hardware devices.

3. It offered the potential for expansion and linking into other proprietary software for future CAD/CAM (Computer aided manufacture) projects e.g. enhanced 3D imaging for garment illustration, kinematics, Robographics programs for off-line robot programming, flexible manufacturing systems cell simulation, all of which were consider to be important for future automation in garment manufacture and consequently for future educational requirements.

Inevitably, as well as offering considerable benefits in the form of a graphics software framework, it provided some constraints, notably the macroprogramming language within the system. This restricted some programming capabilities and overall system development.

At the outset, it was not possible to define detailed objectives for the system in terms of the exact structure of the system, exactly which functions should be used, etc. Furthermore the objectives were refined and features were added as the system was developed. Therefore, it was decided that the best approach to describe its development was to give an overview of the final system that was created followed by a description and explanation of the procedures used. The remainder of this chapter therefore provides the overview of the system. The following chapters describe the system development.

3.2. The Philosophy behind the Pattern Making System

The starting point in the creation of flat patterns for most garments is the basic block. This is a pattern whose shape and size is constructed according to anthropometric data and the specific type of garment to be created e.g. bodice, skirt etc. The pattern has minimal style features. Most computer aided design systems allow a block pattern to be created either by using appropriate graphics functions within the system or by digitising or scanning existing paper or card patterns into the computer memory as described

in Chapter 2. The first method is time consuming and tedious even using a computer and the second requires an existing pattern to have been produced manually. The philosophy adopted in this project was to try to expedite and simplify pattern creation and design as much as possible. So the approach chosen was to produce software programs that would automatically create a basic block for a given style of garment from anthropometric data generated and input by the user. Additional programs would be written to create auxiliary components such as collars, cuffs etc. to suit the block patterns having adapted them to the style of garment required.

It was reasoned that once the novice user became familiar with the principles of manual block design and adaptation, he/she (throughout the rest of the thesis reference to the users will assume them to be male) should find that such a system should considerably expedite the process of design and adaptation.

The Pattern Making System as it was called, and abbreviated to PMS for convenience, was thus evolved.

The PMS was created in four stages. Firstly, a suite of programs was written using the macroprogramming language to create a set of basic block patterns and later, the auxiliary programs were written in a similar way. Secondly, the structure of the PMS was developed by compiling a set of menu driven icons to access these programs and graphics functions within the host CAD system to adapt the patterns.

Thirdly, a further set of programs was written to add seam allowance and grade the patterns. Finally, help files, demonstration programs and tutorials were written to complete and enhance the system to make it as user friendly as possible and to support self-teaching.

A more detailed overview of the PMS is now presented. To avoid repetition, the reader is recommended to read the help files and tutorials in Chapter 8 and 9 respectively for a description of screen displays, presentations etc. The tutorials have been compiled into a separate volume (Chapter 9) following the conclusions. Although it may seem illogical and unnecessary, it was considered to be more suitable for reference by the potential user of the system to separate it from the main part of the thesis.

3.3. The Structure of the PMS

The development of the PMS was based on the CADDS system (Computervision Automated Design and Drafting System) marketed by Computervision, a division of Prime Plc. The CADDS system is an engineering graphics software system, which has many functions useful for pattern making. Also the system incorporates a programming language CVMAC, which is the macro language for the system. A part of CVMAC is the "Mechanical Design Application", which can be used for creating geometric entities [33]. Functions specifically required for making patterns have been programmed using CVMAC. Therefore the PMS can be considered as the

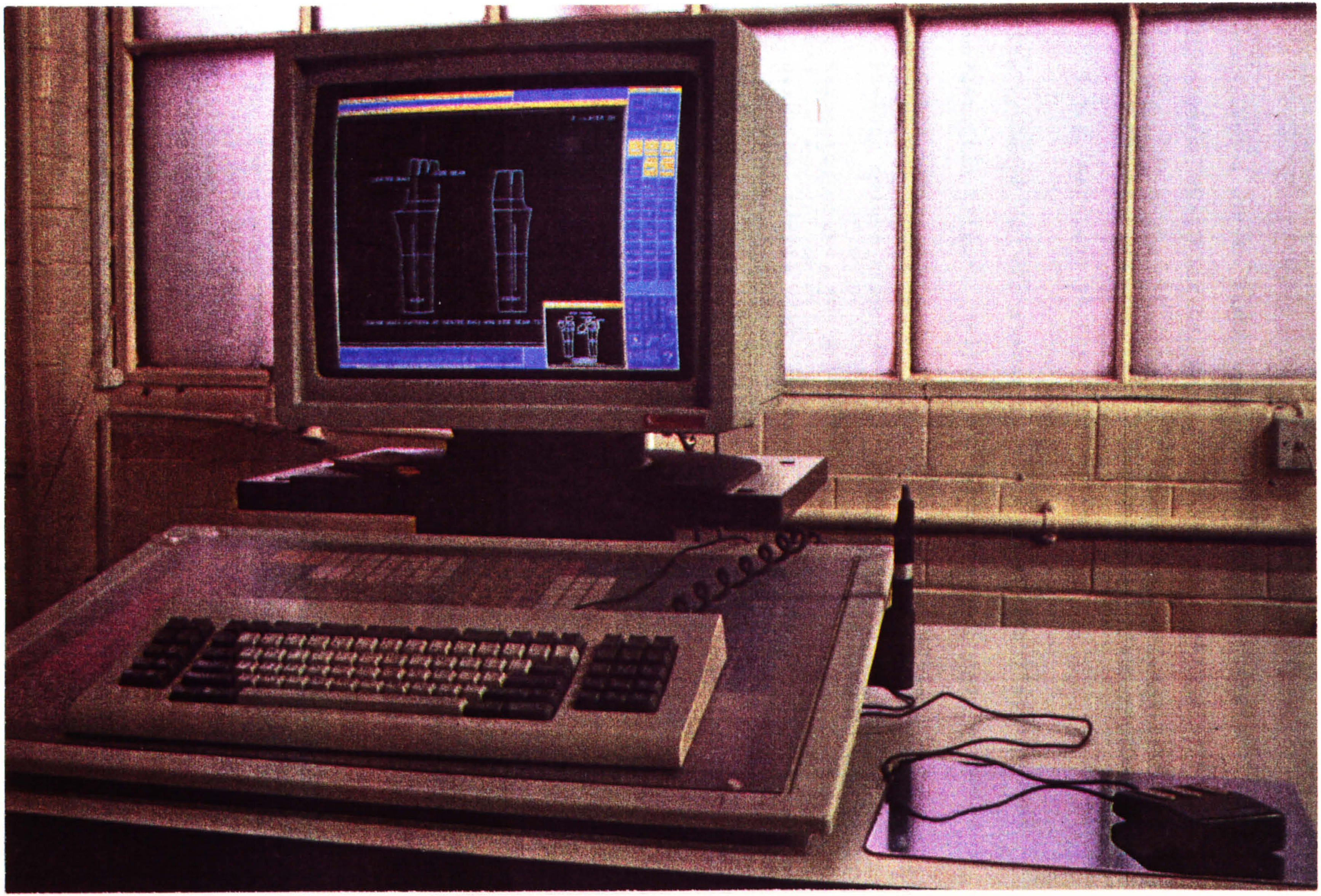


Figure 3-1: CADDS System Hardware

combination of the CADDS system and a suite of programs that have been developed to meet the special needs of pattern making.

3.3.1. Hardware of the CADDS System

The hardware of the system is based on a Sun 3 stand-alone workstation (Figure 3-1). The hardware has the following features [34]:

* 19-inch colour high resolution graphics display
* Keyboard, mouse, and mouse pad
* Deskside module with Motorola 68020 Central Processing Unit
* Two 170 Megabyte sealed disks (140MB formatted)
* C-size tablet and pen
* 1/4-inch tape drive
* Graphic Processing Unit to run CADDS software

3.3.2. Software of the CADDS System

The basic software that comprises the CADDS system are the Unix 4.2 operating system and the Suntools Window System, which allows the user to work on multiple tasks concurrently, so that the window system increases efficiency [34]. The window system divides the area available on the workstation screen into multiple rectangles (windows).

In addition there is the CADDS 4X Application Software, which provides many commands and functions that have been included in the PMS. These functions can create, divide, copy, move, mirror, rotate or delete entities in order to

design a pattern. The entity can be a point, a line, a curve (In the CADDS system a curve is defined by a B-spline.), a circle, an arc or a text. Also, there are many convenient functions, such as those for measuring the length of an entity, creating an offset of an entity or enlarging (zooming) the graphic display.

A description of every function used is given in Chapter 4, and detailed information and examples of when and how to use the functions are in Chapter 9 (Basic Tutorial for the PMS, Sessions 5 ~ 11).

3.3.3. CVMAC Programs Used in the PMS

Many programs have been written in the CVMAC programming language for the PMS to fulfill the objectives set for the system. The programs are listed in Table 1.

Table 1: CVMAC Programs of the PMS

* Bodice Block Patterns 1. Close Fitting Bodice Block, 2. Easy Fitting Bodice Block, 3. Dartless Bodice Block, 4. Overgarment Bodice Block, 5. Tailored Jacket Block, 6. Classic Shirt Block, 7. Jersey Wear Block, 8. Knitted Garment Block * Sleeve Block Patterns 1. One-piece Sleeve Block, 2. Two-piece Sleeve Block, 3. Easy Fitting Sleeve Block, 4. Shirt Sleeve Block * Skirt Patterns

1. Tailored Skirt Block, 2. Circular Skirt Pattern, 3. Pleated Skirt Pattern, 4. Gathered Skirt Pattern

* Trouser Block Patterns

1. Basic Trouser Block, 2. Easy Fitting Trouser Block, 3. Culotte Block

* One-piece Dress Block Patterns

1. Close Fitting One-piece Dress Block, 2. Easy Fitting One-piece Dress Block, 3. Dartless Easy Fitting One-piece Dress Block

* Two-piece Dress Block Patterns

1. Close Fitting Two-piece Dress Block, 2. Easy Fitting Two-piece Dress Block, 3. Dartless Easy Fitting Two-piece Dress Block

* Standing Collar Patterns

1. Straight Standing Collar, 2. Shirt Collar, 3. Convertible Collar, 4. Polo Collar

* Cuffs and Waistband Patterns

1. Shirt Cuff Pattern, 2. Frilled Cuff Pattern, 3. Waistband Pattern

* Inside Pockets

1. Flap Pocket, 2. Welt Pocket, 3. Slash Pocket

* Patch Pockets

1. Square Patch Pocket, 2. Chop-Corner Patch Pocket, 3. Round-Corner Patch Pocket, 4. Square-V Patch Pocket

* Buttonholes

1. Horizontal Worked Buttonholes, 2. Horizontal Bound

Buttonholes, 3. Vertical Worked Buttonholes, 4. Vertical Bound Buttonholes * Marking tools 1. Grain Line, 2. Notch Point * File Management program 1. Openfile * Grading Programs 1. Grade-Bodice, 2. Grade-Skirt, 3. Grade-Trouser, 4. Grade-Sleeve, 5. Grade-Collar, 6. Grade-Height, 7. Grade-Horizontal, 8. Grade-Vertical

Detailed information about each program is given in Chapter 5.

3.4. Management Structure of the PMS

In order to make the system user-friendly, most of the PMS functions were connected to an icon menu that was specially created to allow the user to select a function using the peripheral mouse or light pen facilities associated with the computer hardware.

The icon menu consists of several menu sheets, each of which has 21 icons in it. Three different types of icons have been used according to the functions to which they have been related. These are a single icon, a down arrow icon and a bent corner icon. The single icon is connected to a command, whereas the down arrow icon is connected a group of (up to seven) commands which have some common facilities. If

a user selects a down arrow icon, a list of options is displayed on the computer screen for the commands connected to the icon. The bent corner icon is used to move from a menu page to another page to select an icon on another menu page. Detailed instructions on how to use the icon menu have been written in Chapter 8. These are explained in the System Help File, 4. Icon Menus.

A User Menu for the PMS has also been created. It incorporates five menu pages. These are "Block", "Detail", "Adapt", "Grade" and "Appear". The name of each menu page relates to the different functions created within the system i.e. Block Pattern Generating (Block), Detail Pattern Generation (Detail), Pattern Adaptation (Adapt), Pattern Grading (Grade) and Appearance of the Pattern (Appear). Within each page of the user menu, icons are connected to the commands which are frequently used to perform the functions of the menu page. For example, icons in the menu page "Block" are connected to the commands which call programs to generate block patterns. Some icons are duplicated as the commands are needed on different menu pages, e.g. "Measure Length" appears on every menu page. The command(s) for each icon of the User Menu are explained in Chapter 6. Also, detailed information about how to use the icon menu of the PMS is given in Chapter 9 (Basic Tutorial for CADDS, Session 5).

3.5. Help Files

Three on-line help files have been created within the PMS.

These help files can be accessed by selecting one of three help icons that have been created. These icons are located at the top row of each menu page of the User Menu. The user help files are used to explain not only details of functions used the pattern making system but also those used in the CADDS system, since an understanding of these is a prerequisite to using the PMS. The CADDS system has some on-line documentation but lacks the succinctness required for easy use. The user is referred to this and other documentation as necessary in the help files. The three help files are:

* User Help File
* System Help File
* Page Help File

3.5.1. User Help File

The User Help File has been written to give the user a general brief guide to the system. The syntax has been written deliberately to try to be as user friendly as possible. The contents of this help file are as follows.

1. What Is the Pattern Making System?
2. What kind of work can this system do?
3. How to Work with This System
    - 3.1. CADDS Command Icon
    - 3.2. Programmed Icon
4. Useful CADDS Manuals
    - 4.1. CADDS User Guide
    - 4.2. CADDStation User Interface
    - 4.3. CADDS Core Reference (Vol. 1,2)

5. Help File for Pattern Making System

    5.1. User Help File

    5.2. System Help File

    5.3. Page Help File

6. Demonstration Program

7. Tutorial

The User Help File is described in the first part of Chapter 8.

3.5.2. System Help File

The System Help File describes the basic information and instructions which are needed to work with CADDS. This help file is divided into 10 subjects. When a user selects this help file, a numbered index menu will be displayed, requesting the subject for which more information is desired. Then the subject, according to the number has been selected will be shown. The 10 subjects and their sub-subjects are as follows.

1. Punctuation Marks, Control and Escape Sequences under CADDS

    1.1. Punctuation Marks

    1.2. Control and Escape Sequences

2. The Mouse and Mouse Pad

3. Windows

    3.1. Names and Functions of Windows under CADDS

    3.2. Hiding, Exposing, Closing and Reopening Windows

4. Icon Menus

    4.1. What are Icon Menus?

Also, at the end of each sub-subject, there are the guide lines explaining the relevant CADDS manuals that contain further information. The System Help File is described in the second part of Chapter 8.

3.5.3. Page Help File

The Page Help File has quite detailed information about each icon in the User Menu. On every User Menu page, there is an icon for the Page Help File. In order to enquire about an icon, the icon appropriate for the Page Help File on a menu page has to be selected. Then the Page Help File for that menu page will be opened, and a numbered index of icons on the page will be shown. For example, if the user wants to know about icon number 13 on the menu page "Adapt", he must select the icon for the Page Help File on the menu page "Adapt". Then he will receive the following index menu. The icons whose functions are in upper case have been connected to the CADDS command functions, whereas the icons whose functions are in lower case have been connected to the PMS programs.

Which icon do you want to know more about?

Icon No.	Function	Icon No.	Function
4	---> Tutorial	5	---> MEASURE LENGTH
6	---> MEASURE DISTANCE	7	---> CONSTRUCT GROUP
8	---> DISASSOCIATE GROUP	9	---> REGENERATE GRAPHICS
10	---> Grain Line	11	---> Notch Point
12	---> Highlight (DISCRIMINATE ENT)	13	---> Seam Allowance

```
14 ---> GENERATE OFFSET D              15 --->
16 ---> Translate, Rotate or Mirror    17 ---> Trim or Divide Entity
18 ---> Point                          19 ---> Line
20 ---> B-spline                       21 ---> Circle
-------------------------------------------------------------
    Enter the 'number' of an icon or 'Q' to quit please. --->
```

At the prompt the user enters the relevant number ("13" in this case), and the information about icon number 13 on the menu page "Adapt" will be shown.
The Page Help File is described in the third part of Chapter 8.

3.6. Tutorials

The tutorials for the PMS have been written specially for a novice user of the system. A novice user is expected to have a fundamental knowledge of pattern design and adaptation and some elementary experience in the use of micro computers.

3.6.1. Purpose of the Tutorials

The main purpose of the tutorials is to give a simple explanation on how to start to use the system, so that the user can become more familiar with the system and gain confidence. The information included in the tutorials is sufficient to become proficient at using the system as might be required by an undergraduate student. However, to improve their efficiency and gain more insight into the

system e.g. postgraduate student, upon completion of the tutorial the user is advised to find more information from the help files, CADDS manuals, on-line CADDS help files. Information on these references is included in the tutorials.

3.6.2. Structure of the Tutorials

The tutorial for the PMS comprises three tutorials, each of which is divided into several sessions.

The first tutorial is entitled "The Basic Tutorial for CADDS", which comprises 10 sessions. This tutorial teaches the user about basic CADDS commands to help the user understand and become familiar with the CADDS system. Understanding and practicing the CADDS system is very important because the PMS is heavily dependent upon he CADDS system.

The second tutorial is "The Basic Tutorial for Pattern Making System", which comprises 12 sessions. This tutorial teaches the user most of the facilities of PMS except for grading, which is covered in "The Advanced Tutorial".

The tutorials employ a "copy me" approach in teaching the user how to use the functions. Relevant explanations are given as appropriate together with default instructions, examples, exercises and practices. These tutorials are considered to be a very important aspect of this work. The user is encouraged to follow the tutorials in sequence since omitting any session will impair understanding and inhibit

progress in the use of the system. Also, at the end of every session, there is an assignment, which can be carried out easily if the user has understood the session.

3.6.3. Subjects in the Tutorials

The following unit covers the information given in the tutorials.

1. Basic Tutorial for CADDS

   Session 1: Login, Logout

   Session 2: Simple CADDS Commands

   Session 3: Getdata

   Session 4: Modifiers

   Session 5: Icon Menu

   Session 6: Getdata Modifiers

   Session 7: CADDS Display

   Session 8: Layers

   Session 9: File Handling

   Session 10: Plot Hard Copies

2. Basic Tutorial for Pattern Making System

   Session 1: User Menu for the Pattern Making System

   Session 2: Block pattern - Bodice

   Session 3: Block Pattern - Skirt, Trousers, Dress

   Session 4: Block Pattern - Sleeve

   Session 5: Adaptation - Separate Front and Back

   Session 6: Adaptation - Dart Manipulation

   Session 7: Adaptation - Convert Darts into Seams

   Session 8: Adaptation - Pleats, Flare and Gather

   Session 9: Adaptation - Kimono Sleeve

Session 10: Adaptation - Buttoned Front

Session 11: Adaptation - Flat Collar

Session 12: Additional programs in
the sub-menu page "Detail"

3. Advanced Tutorial

Session 1: Making Production Pattern

Session 2: Grading 1

Session 3: Grading 2

Details of the tutorials are given in Chapter 9 in Vol. 2.

CHAPTER 4

DESCRIPTION OF CADDS

4.1. Introduction

The CADDS functions included in the PMS can be divided into several groups. There are functions for:-

* Creating entities
* Measuring entities
* Translating, rotating or mirroring entities
* Dividing or triming entities
* Creating offset of entities
* Deleting entities
* Constructing a group of entities or disassociating the group
* Managing the display of the entities

The entity types which are used to design a pattern are;-

* point
* line
* B-spline (curve)
* circle
* text

All the block patterns consist of points, lines, curves, arcs and text. Straight seams or darts are drawn by lines. Curved seams and armholes are drawn using B-splines, which

are functions to draw complex curves in the CADDS system [35]. However some curves, whose shape is more similar to a circle rather than a B-spline such as a front neck line, may be drawn using a circle creation routine. The points are inserted when any points have to be marked on the pattern. Also, any text such as a pattern name or a size code can be written on the pattern as a user wishes.

The functions for creating entities are mainly used to generate block patterns, to draw new design lines when the block pattern is adapted and for grading the pattern. Functions for measuring entities are used to measure the length, distance or the angle of the entities to generate accurate pattern pieces. The functions for translating, rotating or mirroring entities and dividing or trimming entities are used to manipulate patterns to make a design pattern. The functions for creating offset of entities add seam allowance. The function for deleting entities removes the entities which have become unnecessary as a result of the pattern adaptation. The function for constructing a group of entities enables several entities to be treated as one entity and "Disassociate a Group" function breaks the group. The function for managing the display of the entities enlarges the drawing or discriminate some entities. These functions are used through the pattern making procedure.

Most of the CADDS functions can be used in two ways. One is by selecting the icon for the command interactively, and the other is through a CVMAC program. Even though the

same function is used, the CADDS command and the CVMAC program statement for the function have different syntaxes [33] [35]. This is because of the different process of data input. When the CADDS command is used, the data is input interactively following the command, but when the CVMAC program is run, the data has to be already included in the program statement. For example, following CADDS command and CVMAC program statement will produce the same result.

CADDS command) #n#INSERT POINT: MODEL loc X0Y0

CVMAC program) P1=POINT/X0Y0

However, a detailed explanation of the CVMAC programing language syntax [35] will not be included in the thesis. This is because the following explanation about CADDS functions should be sufficient to understand the facilities of the system.

These functions were selected after extensive experimentation. Details of this experimentation have not been included since the procedure was quite simplestic but extremely time consuming. Some of the functions required were obvious, other functions needed evaluating to assess their suitability. Whilst additional functions could also have been used, it was felt that these did not significantly enhance the perforance of the system, and in some cases could have caused confusion to the user.

4.2. Commands for Creating Entities

4.2.1. Point

INSERT POINT

Creates a point by selecting a location.

INSERT POINT ON

Inserts a point on a selected entity. The point is created by dropping a perpendicular from a selected location to an entity.

GENERATE POINT ON N <n>

Generates evenly distributed <n> points along an entity. This command can be used when more than two points are inserted.

GENERATE POINT ON DIST <x>

Generates two points on the entity, one at the start location of the entity and the other at a distance <x> from the first point.

4.2.2. Line

INSERT LINE

Creates a line between two locations.

INSERT LINE HOR (HORIZONTAL)

Creates a horizontal line. Two locations are required. The first location selected is the start of the line. The second location indicates the direction and distance along the horizontal axis.

INSERT LINE HOR LENGTH <x>

Creates a horizontal line of a specified length <x>. Two locations are required. The first location selected is the start of the line. The second location indicates the direction of the line for a distance of <x>.

INSERT LINE VER (VERTICAL)

Creates a vertical line. Two locations are required. The first location selected is the start of the line. The second location indicates the direction and distance along the vertical axis.

INSERT LINE VER LENGTH <x>

Creates a vertical line of a specified length <x>. Two locations are required. The first location selected is the start of the line. The second location indicates the direction of the line for a distance of <x>.

INSERT LINE PERP (PERPENDICULAR)

Creates a line perpendicular to an existing entity.

INSERT LINE PAR (PARALLEL)

Creates a line parallel to an existing line.

4.2.3. B-spline (Curve)

INSERT BSPLINE

Creates a B-spline; a smooth curve connecting a series of locations [35].

INSERT BSPLIN DEGREE <n>

Creates a B-spline according to a specified degree <n>, where <n> is an integer from 2 to 7. The default is degree 3 (cubic B-spline). In general, if different B-splines have the same locations, then the higher the degree, the smoother the curve.

INSERT BSPLIN DEGREE 3

Creates a 3 degree B-spline.

INSERT BSPLIN DEGREE 3 TANA

Creates a 3 degree B-spline with the starting vector, which is defined by two locations.

INSERT BSPLIN DEGREE 3 TANB

Creates a 3 degree B-spline with the ending vector, which is defined by two locations.

INSERT BSPLIN DEGREE <n> TANA

Creates a B-spline of degree <n> with the starting vector, which is defined by two locations.

INSERT BSPLIN DEGREE <dmvar> TANB

Creates a B-spline of degree <n> with the ending vector, which is defined by two locations.

4.2.4. Circle

INSERT CIRCLE

Inserts a circle which is defined by three locations.

INSERT FILLET CIRCLE -TRIM THRENT

Inserts a circle which is tangent to three entities.

INSERT CIRCLE DIAMETER <x>

Inserts a circle whose diameter is <x>. The selected location will be the centre of the circle.

INSERT CIRCLE RADIUS <x>

Inserts a circle whose radius is <x>. The selected location will be the centre of the circle.

INSERT FILLET CIRCLE -TRIM RADIUS <x>

Inserts a circle whose radius is <x>. The circle will be inserted to be tangent to the two entities.

INSERT CIRCLE TANTO

Creates a circle that is tangent to an existing entity. The first location will be the centre of the new circle. The second should be on the existing entity that the circle will be tangent to.

INSERT CIRCLE ARC Creates a circle from an existing arc.

4.2.5. Text

INSERT TEXT

Inserts standard text strings. Each text string contains alphanumeric characters, a location, and other characteristics to control its appearance and orientation, such as height, width, thickness, angle, slant, justification and various font type.

4.3. Commands for Measuring Entities

MEASURE LENGTH

Measures a length of a contour. A contour is a sequence of connected curves. The entities that can make up a contour include lines, arcs, and B-splines. Successive pairs of curves in the sequence (first and second, second and third, etc) must either match at their endpoints or intersect.

MEASURE ANGLE

Measures the angle between two intersecting lines.

MEASURE DISTANCE

Measures the following:

* The minimal distance and angle between two lines.
* The minimal distance between two locations or between two point entities.
* The minimal distance between a location and a point, between a location and a line, or between a point and a line.

4.4. Commands for Translating, Rotating or Mirroring Entities

4.4.1. Translate Entity

TRNSLATE ENTITY

Translates an entity or a group of entities. The vector of the translation will be defined by two selected locations.

TRANSLATE ENTITY COPY

Translates a copy of an entity or a group of entities.

4.4.2. Rotate Entity

ROTATE ENTITY

Rotates entities. Three locations will be required, the first location for the centre of the rotation, and the second and the third location for the angle of the rotation.

ROTATE ENTITY ANGLE <x>

Rotates entities. The selected location will be the centre of the rotation. The angle of the rotation will be <x>.

4.4.3. Mirror Entity

MIRROR ENTITY

Mirrors entities. This is done by reflecting the entity around an axis (mirror plane), which will be defined by two selected locations.

MIRROR ENTITY COPY

Mirrors a copy of entities.

MIRROR ENTITY COPY LINEPLANE

Mirrors a copy of entities. This is done by reflecting the entity around the selected line.

4.5. Commands for Dividing or Trimming Entities

4.5.1. Trim Entity

TRIM ENTITY

Shortens or lengthens an entity such as lines, arcs, circles and B-splines.

TRIM ENTITY ILENGTH <x>

Trims (extends or shortens) a line by a value of <x>. A positive <x> shortens the line while a negative <x> lengthens the line.

TRIM ENTITY MINTOF

Trims up to 100 entities by one intersecting entity.

TRIM ENTITY CORNER

Creates corners for curves and strings by simultaneously trimming them at their points of intersection.

4.5.2. Divide Entity

DIVIDE ENTITY MPROJ

Subdivides selected entities, creating new entities of the same type as those selected.

DIVIDE ENTITY INTERSECTION OF

Directs the system to divide the selected entities wherever they intersect a second entity.

DIVIDE ENTITY NUMBER OF DIVISIONS <n> Divides an entity into <n> subdivided entities.

4.6. Commands for Creating Offset of Entities

CONSTRUCT OFFSET D <x>

Creates an offset of an entity such as a point, line, arcs, circle, B-spline. An offset is a copy of an entity (or part of an entity) that is created at a specified distance <x> from the original.

GENERATE OFFSET D <x>

Creates an offset of lines and arcs by a constant distance, and clips (or extends) new lines and arcs at (to) intersection points. Up to 10,000 entities may be selected.

4.7. Command for Deleting Entities

DELETE ENTITY

Deletes entities from the data base.

4.8. Commands Related to Constructing a Group

CONSTRUCT GROUP

Creates groups of separate entities. Any number of entities can be combined into a group. The purpose of creating groups is to provide a means of quickly identifying entities within getdata.

DISASSOCIATE GROUP

Disassociate the association among entities set up by the command CONSTRUCT GROUP.

4.9. Commands for Managing Display

ZOOM DRAWING WINDOW

Zooms and scrolls the drawing within a window defined by two opposite corner locations.

ZOOM DRAWING ALL

Resets the drawing from a zoomed state to its original state.

DISCRIMINATE ENTITY ON

Temporarily highlights specified entities.

REGENERATE GRAPHICS

Regenerate and redisplay the entities.

CHAPTER 5

DEVELOPMENT OF PMS SOFTWARE

5.1. Introduction

The CVMAC programs of the PMS can be divided into three groups. The first comprises the block pattern generation programs, the second comprises the additional programs for generating auxiliary pattern components and the third, xomprises the grading programs. All the programs have user-friendly prompts and/or instructions whenever a user-input is expected for a data or for a selection from multiple choices.

A List of the programs are in Chapter 3, and the programs are detailed in Appendices A, B and C.

5.2. Block Pattern Generation Programs

A block pattern generation program is a framework which generates the graphic entities of a block pattern. A block pattern is a foundation pattern constructed to fit an average figure [36]. In the clothing industry the block pattern is the starting point for pattern making because most production patterns are made by adapting a block pattern according to the final garment design. Therefore the fit of a garment depends on the fit of the block pattern.

There are many ways of drafting a block pattern. This stems from the fact that block pattern drafting has evolved from the experiences of designers who have their own preferences for the fitting and design of patterns. There have been many books written on the subject of pattern construction [37] [38] [39]. Some of them have been concerned with methods for producing patterns for bulk manufacturing of stock sized garments, whilst others are aimed at the made to measure trade by explaining the various methods of cutting patterns to cater for figures other than "normal". For the PMS, the block pattern generating programs were written according to the instructions defined in Metric Pattern Cutting by W. Aldrich [36]. This method was preferred because the drafting instructions were found to be more suitable to computer programming than others that were considered [38] [39].

In total 25 pattern generating programs were written. These covered 8 different types of bodice block patterns, 4 types of sleeve block patterns, 4 types of skirt patterns, 3 types of trouser block patterns, 3 types of one-piece dress block patterns and 3 types of two-piece dress block patterns.

5.2.1. Structure of the Programs.

For compatibility, all the block pattern generating programs were produced to the same program structure. The structure can be divided into the following three stages.

1. Size Input

2. Points definition

3. Graphic Entity Creation

5.2.1.1. Size Input

The size input is processed by calling a size input procedure (sub-routine). There were 5 different size input procedures according to the different category of block patterns, "Bodice Size Procedure (BDSIZE)", "Sleeve Size Procedure (SLSIZE)", "Skirt Size Procedure (SKSIZE)", "Trouser Size Procedure (TRSIZE)" and "Dress Size Procedure (DRSIZE)".

Each procedure defines the body measurements (anthropometric data) that were needed to generate the block pattern. The following body measurements were defined respectively by each procedure.

* Bodice Size Procedure (BDSIZE)

Bust	Shoulder Length	Nape to Waist
Back Width	Waist to Hip	Dart
Armhole Depth	Chest	neck size

* Sleeve Size Procedure (SLSIZE)

Sleeve Length	Cuff Size	Elbow Length

* Skirt Size Procedure (SKSIZE)

Waist	Hip	Waist to Hip
Waist to Knee		

* Trouser Size Procedure (TRSIZE)

Waist	Body Rise	Hip
Waist to Floor	Waist to Hip	Trouser Width

Waist to Knee

* Dress Size Procedure (DRSIZE)

Bust	Shoulder Length	Nape to Waist
Back Width	Waist to Hip	Dart
Armhole Depth	Chest	Waist to Knee

The measurements can be derived either from a set of anthropometric body measurements or by entering individual body measurements. The user has the option to choose the method he prefers. In either case, the program refers to a standard set of body measurements of the program. The standard body measurements of the PMS is shown in Tables 2 and 3, which are based on the British Standard [36]. Table 2 is the standard body measurements for women of medium height, and Table 3 is the vertical measurements adjustment for tall or short women [36].

If a user has decided to use anthropometric body measurements, he needs to input the size code (an even number between 8 and 30) and the height (Short, Medium or Tall), then the program reads all the relevant measurements from the stored standard body measurements of the program.

If a user has decided to use the individual body measurements, he needs to input the nearest size code and the height. With the nearest size code and the height, the program reads the reference measurements from the stored standard body measurements. Then the program shows each reference measurement and allows the user to change it as he

Table 2: Standard Body Measurements (cm)

WOMEN OF MEDIUM HEIGHT 160cm-170cm (5ft 2½in-5ft 6½in)												
SIZE SYMBOL	8	10	12	14	16	18	20	22	24	26	28	30
BUST	80	84	88	92	97	102	107	112	117	122	127	132
WAIST	60	64	68	72	77	82	87	92	97	102	107	112
HIPS	85	89	93	97	102	107	112	117	122	127	132	137
BACK WIDTH	32.4	33.4	34.4	35.4	36.6	37.8	39	40.2	41.4	42.6	43.8	45
CHEST	30	31.2	32.4	33.6	35	36.5	38	39.5	41	42.5	44	45.5
SHOULDER LENGTH	11.75	12	12.25	12.5	12.8	13.1	13.4	13.7	14	14.3	14.6	14.9
NECK SIZE	35	36	37	38	39.2	40.4	41.6	42.8	44	45.2	46.4	47.6
DART	5.8	6.4	7	7.6	8.2	8.8	9.4	10	10.6	11.2	11.8	12.4
WRIST	15	15.5	16	16.5	17	17.5	18	18.5	19	19.5	20	20.5
NAPE TO WAIST	39	39.5	40	40.5	41	41.5	42	42.5	43	43.2	43.4	43.6
ARMHOLE DEPTH	20	20.5	21	21.5	22	22.5	23	23.5	24.2	24.9	25.6	26.3
WAIST TO KNEE	57.5	58	58.5	59	59.5	60	60.5	61	61.25	61.5	61.75	62
WAIST TO HIP	20	20.3	20.6	20.9	21.2	21.5	21.8	22.1	22.3	22.5	22.7	22.9
WAIST TO FLOOR	102	103	104	105	106	107	108	109	109.5	110	110.5	111
BODY RISE	26.6	27.3	28	28.7	29.4	30.1	30.8	31.5	32.5	33.5	34.5	35.5
SLEEVE LENGTH	57.2	57.8	58.4	59	59.5	60	60.5	61	61.2	61.4	61.6	61.8
Extra measurements (garments)												
CUFF SIZE SHIRTS	21	21	21.5	21.5	22	22.5	23	23.5	24	24.5	25	25.5
CUFF SIZE, TWO PIECE SLEEVE	13.25	13.5	13.75	14	14.25	14.5	14.75	15	15.25	15.5	15.75	16
TROUSER BOTTOM WIDTH	21	21.5	22	22.5	23	23.5	24	24.5	25.4	26.2	27	27.8

Table 3: Vertical Measurement Adjustment

	SHORT WOMEN 152cm-160cm (4ft 11½in-5ft 2½in)	TALL WOMEN 170cm-178cm (5ft 6½in-5ft 9½in)
NAPE TO WAIST	-2cm	+2cm
SLEEVE LENGTH	-2.5cm	+2.5cm
WAIST TO KNEE	-3cm	+3cm
WAIST TO FLOOR	-5cm	+5cm
BODY RISE	-1cm	+1cm

wants. This method has the advantage that, if he doesn't know any specific parts measurement, he can use the reference measurement as a default value.

The programs were written to accept input in both centimetres and inches. However all the data is converted into centimetres for subsequent processing. A more practical and detailed description of how to input data is given in Chapter 9 (The Basic Tutorial for Pattern Making System, Session 2).

5.2.1.2. Points Definition

Using the measurements, which are defined in the size input stage, the second part of the program defines the points in x, y coordinates needed to create the patterns. The location of the points determines the outline of the pattern. The points are then joined by lines, B-splines or circles by appropriate CADDS functions. Each program uses a different set of instructions to define the location of the points.

To create the pattern, a reference location is required to which the points can be related. This is generated by the user being requested to select one location on the graphic window to define the location of the pattern on the graphics window. This location is the location "1" in Figures 5-4 - 5-28. By adding or subtracting appropriate values from the x, y coordinates of this location, all necessary points were defined.

With reference to Programs in Appendix A, the points have the name "P(n)" to distinguish each point. The numbers in Figures 5-4 ~ 5-28 are the numbers of the point names. The location of "n" in the figure is the location of the point P(n). In the programs x(n), y(n) are the x, y coordinates of the point "P(n)". The mathematical expressions in the programs provide the frame work to define the x, y coordinates for every point. The mathematical expressions were derived from the relevant instructions taken from Aldrich [36].

For example, in Figure 5-4, the point P(4) is the centre back waist point and the point P(1) is the centre back neck point, the x, y coordinates of the point P(4) will be defined as follows.

x(4)=x(1)

y(4)=y(1)-(Nape to Waist)

To give another example, see Figure 5-4. P(1) is the centre back neck point, P(8) is the neck shoulder point and P(9) is the armhole shoulder point. Then the coordinates of P(8) and P(9) will be defined as follows.

x(8)=x(1)+(Neck size)/5-0.2

y(8)=y(1)+1.5

y(9)=y(1)-(Armhole Depth)/5+0.7

x(9)=x(8)+SQRT(((Shoulder Length)+1)**2-(y(8)-y(9))**2)

5.2.1.3. Graphic Entity Creation

The last part of the pattern generation program is the instructions for drawing graphic entities. The entity types which compose a pattern are as follows [33].

* Line
* B-spline
* Circle (Arc)
* Point
* Text

Using the Mechanical Design Application part of the CVMAC language, the graphic entities were generated between the points defined earlier. The Mechanical Design Application part of CVMAC is the part used specially for creating geometric entities [33]. For example, if a line L(1) needs to be drawn between the point P(1) and P(2), the line L(1) can be generated by the following CVMAC program.

```
L(1)=LINE/P(1), P(2)
```

If a B-spline "B(1)" needs to be drawn through the points P(1), P(2), P(3) and P(4), the B-spline B(1) can be generated by the following CVMAC program.

```
B(1)=BSPLIN/3,P(1),P(2),P(3),P(4)
```

In addition, generated entities can be copied, rotated, mirrored, translated or deleted using CVMAC statements [33]. For example (see Figure 5-10), the location numbers need only be expressed on the back pattern. This is because the front pattern is exactly the same as the back pattern except

for the neck line, so that the front pattern can be copied and mirrored from the back pattern. The following is an example of the program for mirror imaging.

```
MIRROR (COPY,cp(1)) 5,L(3),MX,x(10),0,0
MIRROR L(9),MX,x(10),0,0
```

The trouser patterns and the two-piece sleeve block pattern are initially generated overlapping each other, and later one of the overlapped pattern is mirrored. In Figures 5-13, 5-20 and 5-21 the initial overlapped pattern is shown in green. However when the patterns are generated, the green entities and the red numbers are prevented from appearing on the screen, so that only the black entities appear.

5.2.1.4. An Example of Block Pattern Generation

Figures 5-1 ~ 5-3 illustrate how the easy fitting trouser block pattern is generated as an example of the different stages of the block pattern generation.

After the size is input, the necessary point locations are defined as shown in Figure 5-1.

Then the graphic entities i.e. lines, B-splines and circles are drawn between the points as illustrated in Figure 5-2. As mentioned earlier, the front and back trouser patterns are initially generated overlapping each other. Therefore at the stage of Figures 5-1 and 5-2, the patterns are overlapped.

The last stage is making the pattern ready for use. To do so, a pattern has to be mirrored to avoid overlapping,

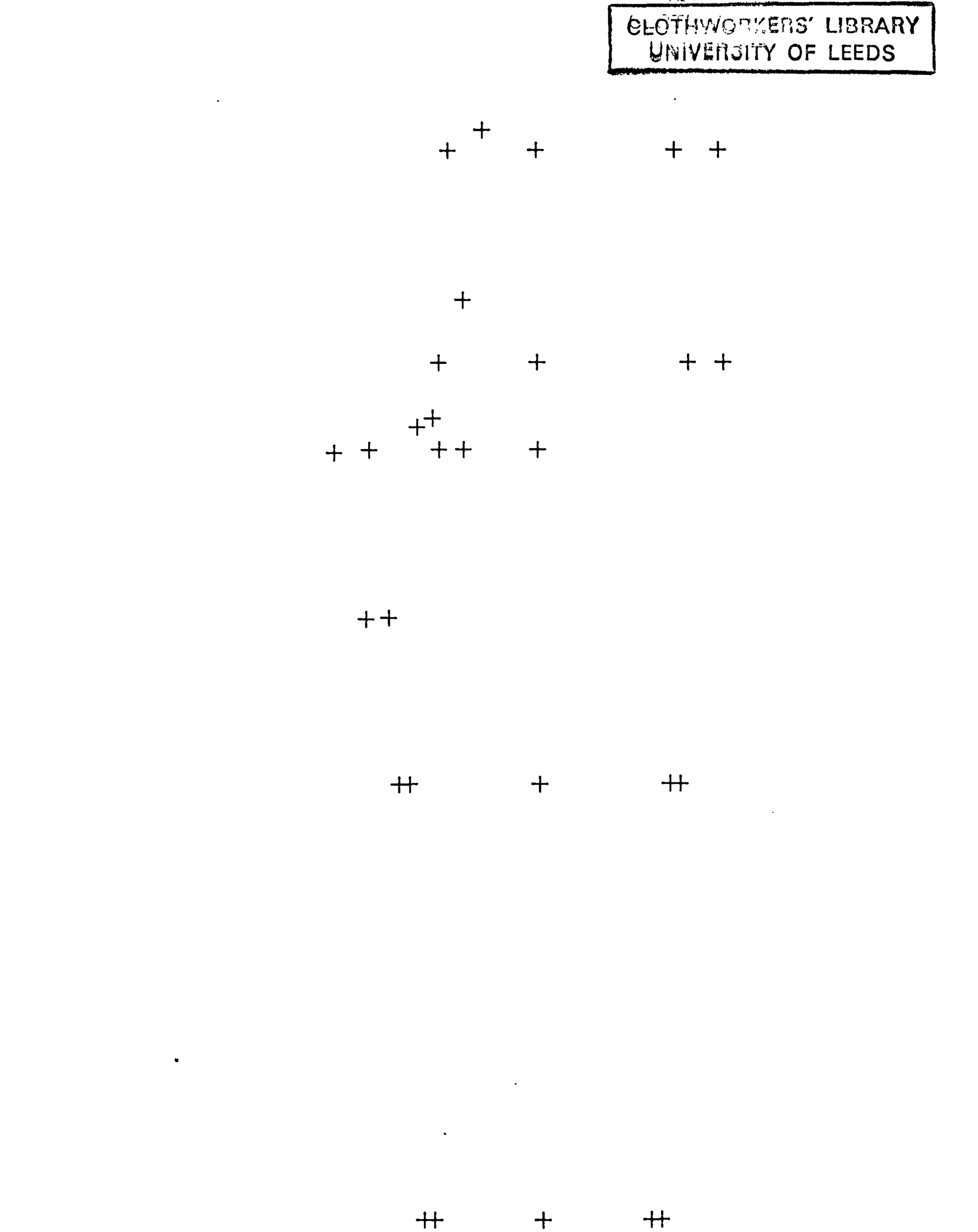


FIGURE 5-1: DEFINE LOCATIONS FOR BLOCK PATTERN
SCALE: 1/5

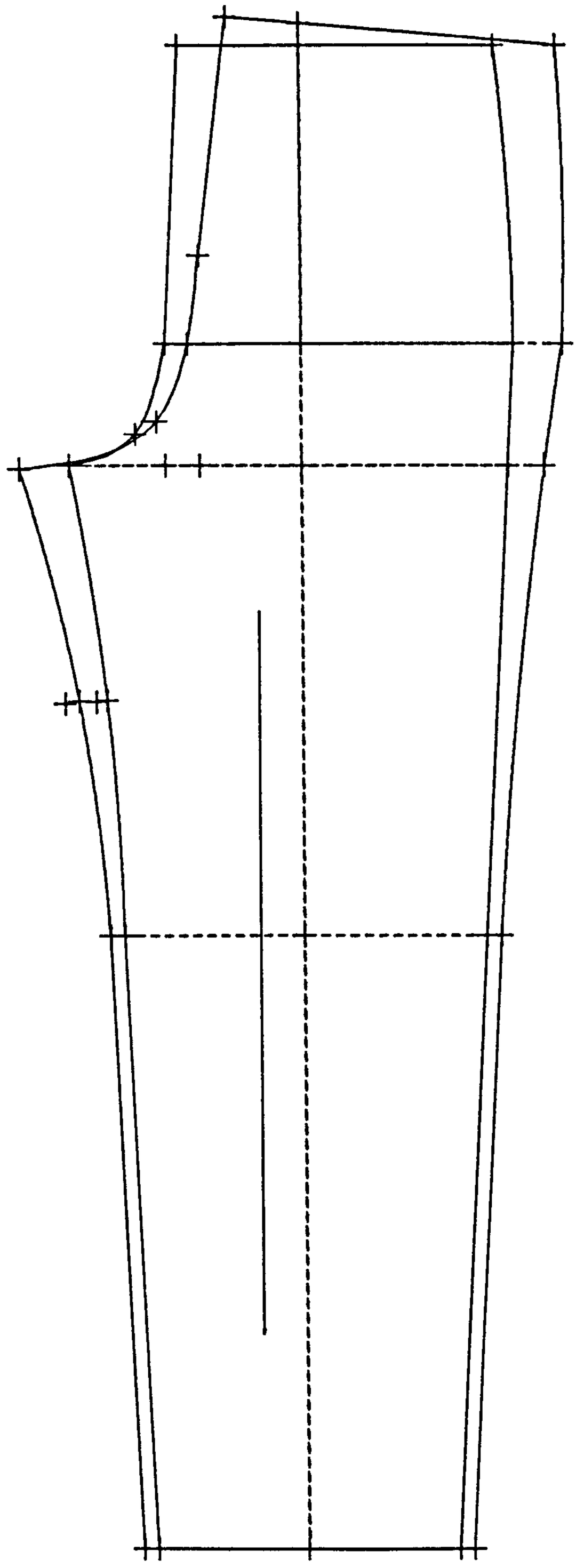


FIGURE 5-2: GRAPHIC ENTITY CREATION
SCALE: 1/5

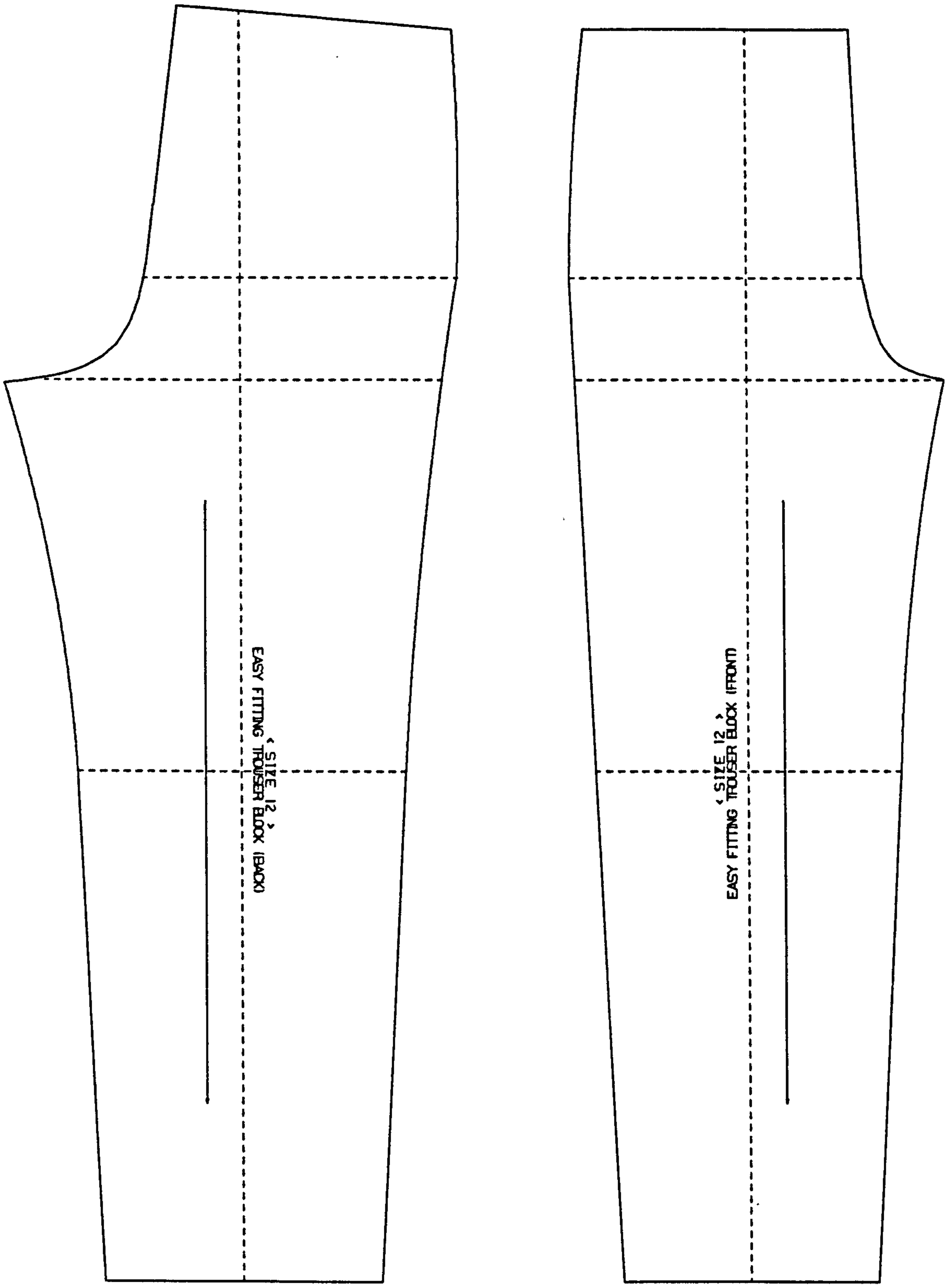


FIGURE 5-3: COMPLETED BLOCK PATTERN

SCALE: 1/5

unnecessary graphic entities are deleted and the text instructions are written. Figure 5-3 shows the last stage of the easy fitting trouser block pattern.

When the user generates a block pattern, he will not see any developmental stages except the finally completed patterns. This is to expedite the operation of the pattern generation, since, if the display of the graphic entities is postponed until the end of the program, the program runs faster.

Having explained the principles on which the blocks are created, it is considered sufficient merely to describe the other types of blocks for which programs were written. Only the mathematical expressions differ for each according to the shape required.

5.2.2. Bodice Block Generation Programs

5.2.2.1. The Close Fitting Bodice Block

The close fitting bodice block is quite a close fit to the figure. so this pattern can form the basis for a close fitting garment such as a close fit blouse, shirt or dress. See Figure 5-4.

5.2.2.2. The Easy Fitting Bodice Block

The easy fitting bodice block has less dart shaping and more ease than the close fitting one. So this pattern can form the basis for an easy fitting style garment such as a loose fitting blouse. See Figure 5-5.

5.2.2.3 The Dartless Easy Fitting Bodice Block

The dartless bodice block is very similar to the easy fitting one, the only difference is that this block does not have any darts. So this pattern can form the basis of easy fitting designs where dart shaping would destroy the design. However the block is not suitable for women with pronounced bust shaping. See Figure 5-6.

5.2.2.4. The Overgarment Block

The overgarment block can form the basis of a coat and an easy fitting jacket. The user can choose either full dart shaping or less bust shaping (half dart shaping) depending on the design. See Figure 5-7-1 and Figure 5-7-2.

5.2.2.5. The Tailored Jacket Block

This pattern can form the basis of a jacket with rever collars. The user can choose either full dart shaping or less bust shaping (half dart shaping) depending on the design of the garment. See Figure 5-8-1 and Figure 5-8-2.

5.2.2.6. The Classic Shirt Block

This pattern is a block pattern for a classic shirt. Hence it can form the basis of a variety of shirt patterns. Also the shirt sleeve and the shirt collar can be generated by other programs. See Figure 5-9.

5.2.2.7. Block Pattern For Jersey Wear

This pattern can be used as a base pattern for jersey wear such as tee shirts. See Figure 5-10.

5.2.2.8. Block Pattern For Knitwear

This pattern can be used as a base pattern for knitwear. See Figure 5-11.

5.2.3. Sleeve Block Generation Programs

Sleeve blocks were created in a similar way to the bodice block. However, in order to generate a sleeve block pattern, the armhole measurement which has to be measured from the bodice pattern is needed. So in the sleeve block generation programs, the user is given the opportunity to measure the front and the back armhole measurements from the bodice pattern after it has been adapted and enter them. More practical explanation about how to measure and enter the armhole is given in Chapter 9 (The Basic Tutorial for the PMS, Session 4).

5.2.3.1. One-Piece Sleeve Block

This pattern is a close fitting one-piece sleeve block. Where no special design requirements are specified, this one-piece sleeve block can form the basis of any sleeve. See Figure 5-12.

5.2.3.2. Two-Piece Sleeve Block

This pattern can form the basis of a two-piece sleeve such as a jacket sleeve or coat sleeve. See Figure 5-13.

5.2.3.3. Easy Fitting Sleeve Block

This pattern is the easy fitting one-piece sleeve block. The user can decide the sleevehead height (crown height) so that he can change the width of the sleeve. This sleeve pattern

can form the basis for a sleeve of an easy fitting garment such as sports wear, shirts or blouses. See Figure 5-14.

5.2.3.4. Shirt Sleeve Pattern

This pattern can form the basis of a shirt sleeve. The user can change the width of the cuff. The sleevehead height (crown height) is set at 1/4 of the armhole. See Figure 5-15.

5.2.4. Skirt Block Generation Programs

5.2.4.1. Tailored Skirt Block

This is a close fitting skirt block pattern which can form the basis for skirts of various designs. By introducing style lines, tucks, gathers, pleats or drapes to this skirt block pattern, many different style of skirt patterns, including a straight skirt, a paneled skirt, a straight skirt with some pleats such as box pleats or inverted pleats, a gored skirt and a flared skirt can be generated. See Figure 5-16.

5.2.4.2. Circular Skirt Pattern

This pattern is for a circular skirt. The circular skirt pattern can be generated for a full circular skirt (360 degree), or for a half circular skirt (180 degree). See Figure 5-17-1 and Figure 5-17-2.

5.2.4.3. Pleated Skirt Pattern

This pattern is for a skirt with allround pleats. The pleat shape can be determined by either pleat width or pleat number. A pleat width must be between 1.27 CM and 7.62 CM

(0.5 IN and 3 IN), and the pleat number must be a number between 10 and 60. See Figure 5-18.

5.2.4.4. Gathered Skirt Pattern

This pattern is for a gathered skirt. The user can generate either a slightly gathered skirt pattern which has a waist curve for better fitting or a very gathered skirt pattern which is a rectangle. See Figure 5-19-1 and Figure 5-19-2.

5.2.5. Trouser Block Generation Programs

5.2.5.1. Basic Trouser Block

This pattern is for a simple pair of trousers. By adapting this pattern, moving darts, giving more ease, introducing style lines or drape the user can generate trouser patterns of many different styles. See Figure 5-20.

5.2.5.2. Easy Fitting Trouser Block

This pattern has more ease than the basic trouser pattern. So this pattern can form the basis of baggy trousers and dungarees. See Figure 5-21.

5.2.5.3. Culottes Pattern

This pattern can form the basis of culottes of various designs. See Figure 5-22.

5.2.6. One-Piece Dress Block Generation Programs

5.2.6.1. Close Fitting One-Piece Dress Block

This pattern is made by extending the close fitting bodice block to the finished length of a dress, and the close fitting bodice block is quite a close fit to the figure.

Also the user has a choice about the waist shaping as to whether or not he wants waist darts and the side seam a close fit at the waist line. This pattern can form the basis of one-piece dress of various design if the dress is a close fit at the bodice part. See Figure 5-23-1 and Figure 5-23-2.

5.2.6.2. Easy Fitting One-Piece Dress Block

This pattern is made by extending the easy fitting bodice block to the finished length of a dress. The easy fitting bodice block has less dart shaping and more ease than the close fitting one. Again the user has a choice about the waist shaping as to whether or not he wants the side seam to be a close fit at the waist line. This pattern can form the basis of an easy fitting style one-piece dress such as a shift dress. See Figure 5-24-1 and Figure 5-24-2.

5.2.6.3. Dartless One-Piece Dress Block

This pattern is made by extending the dartless bodice block to the finished length of a dress. Again the user has a choice about the waist shaping as to whether or not he wants the side seam close fit at the waist line. This pattern can form the basis of an easy fitting style one-piece dress where dart shaping would destroy the design. However this block is not recommended for women with pronounced bust shaping. See Figure 5-25-1 and Figure 5-25-2.

5.2.7. Two-piece Dress Block Generation Programs

These blocks are basically made from two parts.

5.2.7.1. Close Fitting Two-Piece Dress Block

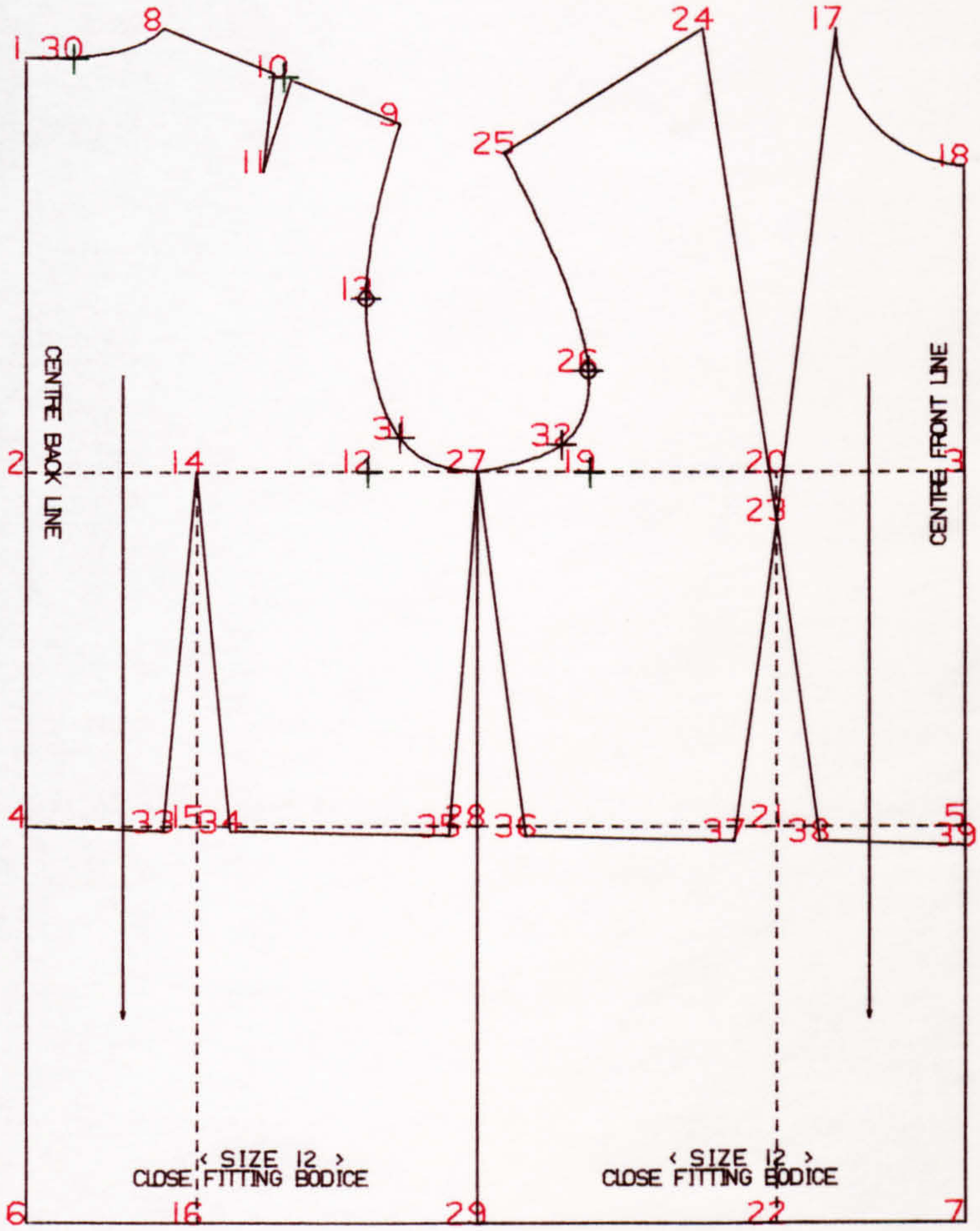


FIGURE 5-4: CLOSE FITTING BODICE BLOCK

SCALE: 1/5

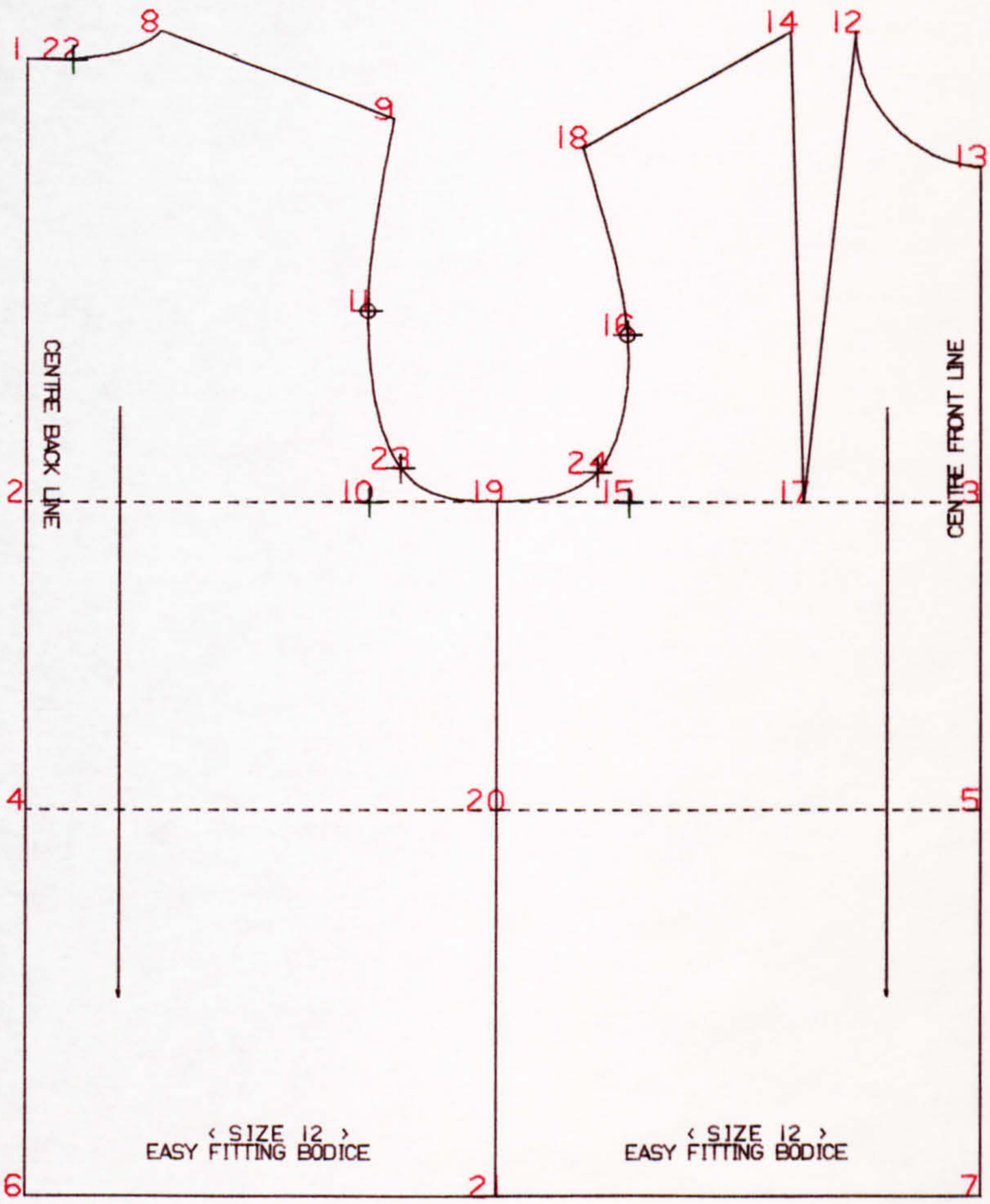


FIGURE 5-5: EASY FITTING BODICE BLOCK
SCALE: 1/5

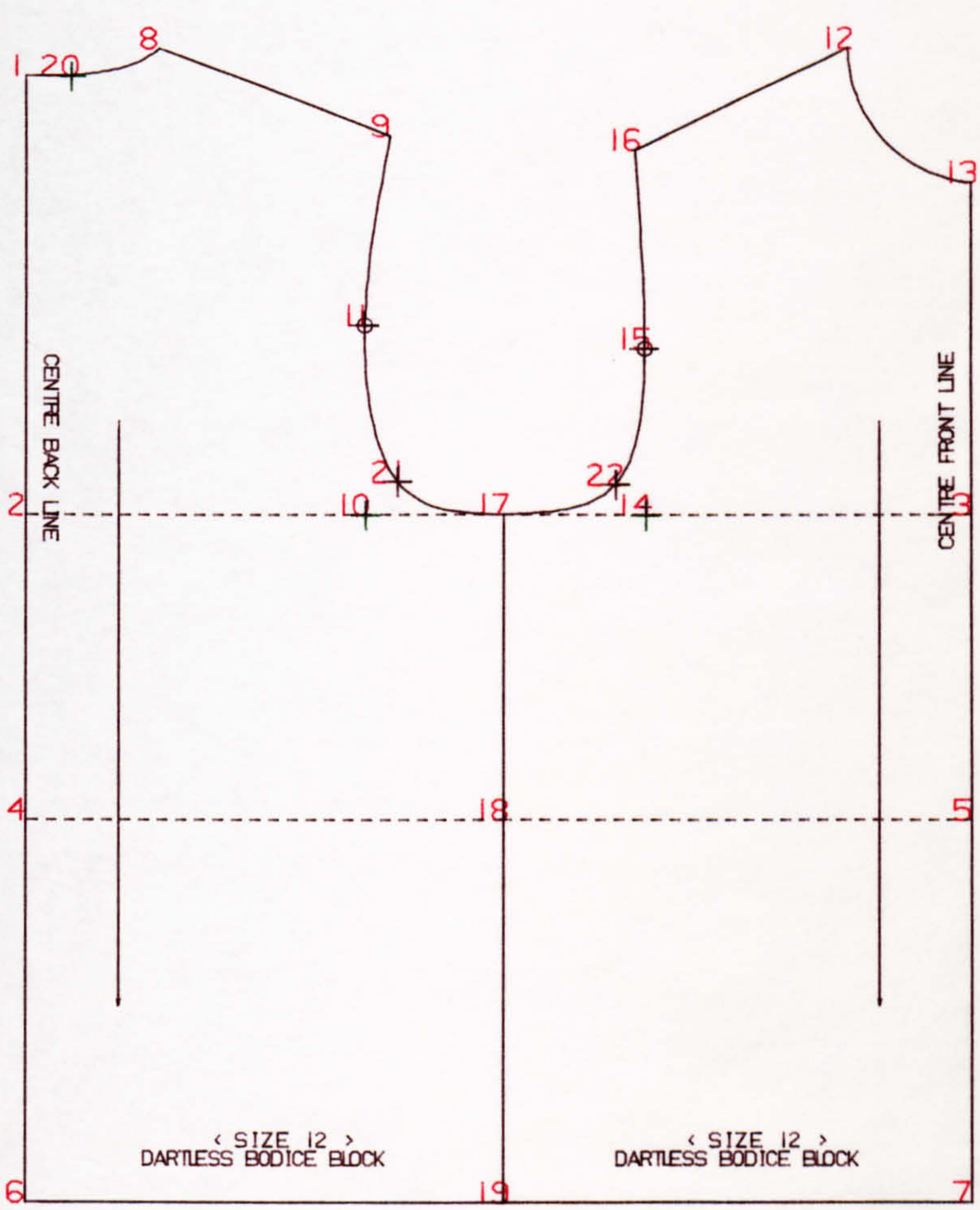


FIGURE 5-6: DARTLESS EASY FITTING BODICE BLOCK

SCALE : 1/5

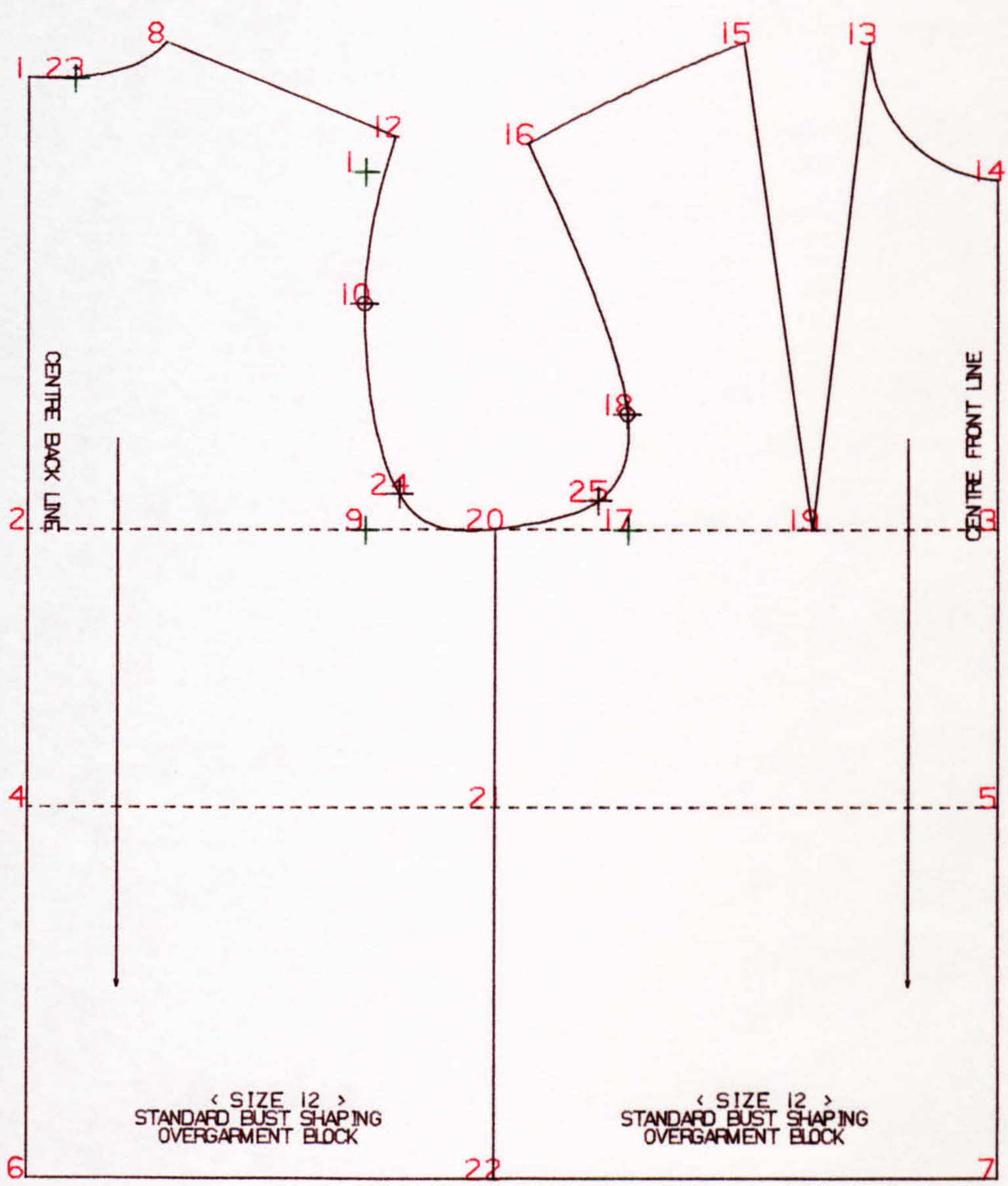


FIGURE 5-7-1: OVERGARMENT BLOCK, FULL DART

SCALE : 1/5

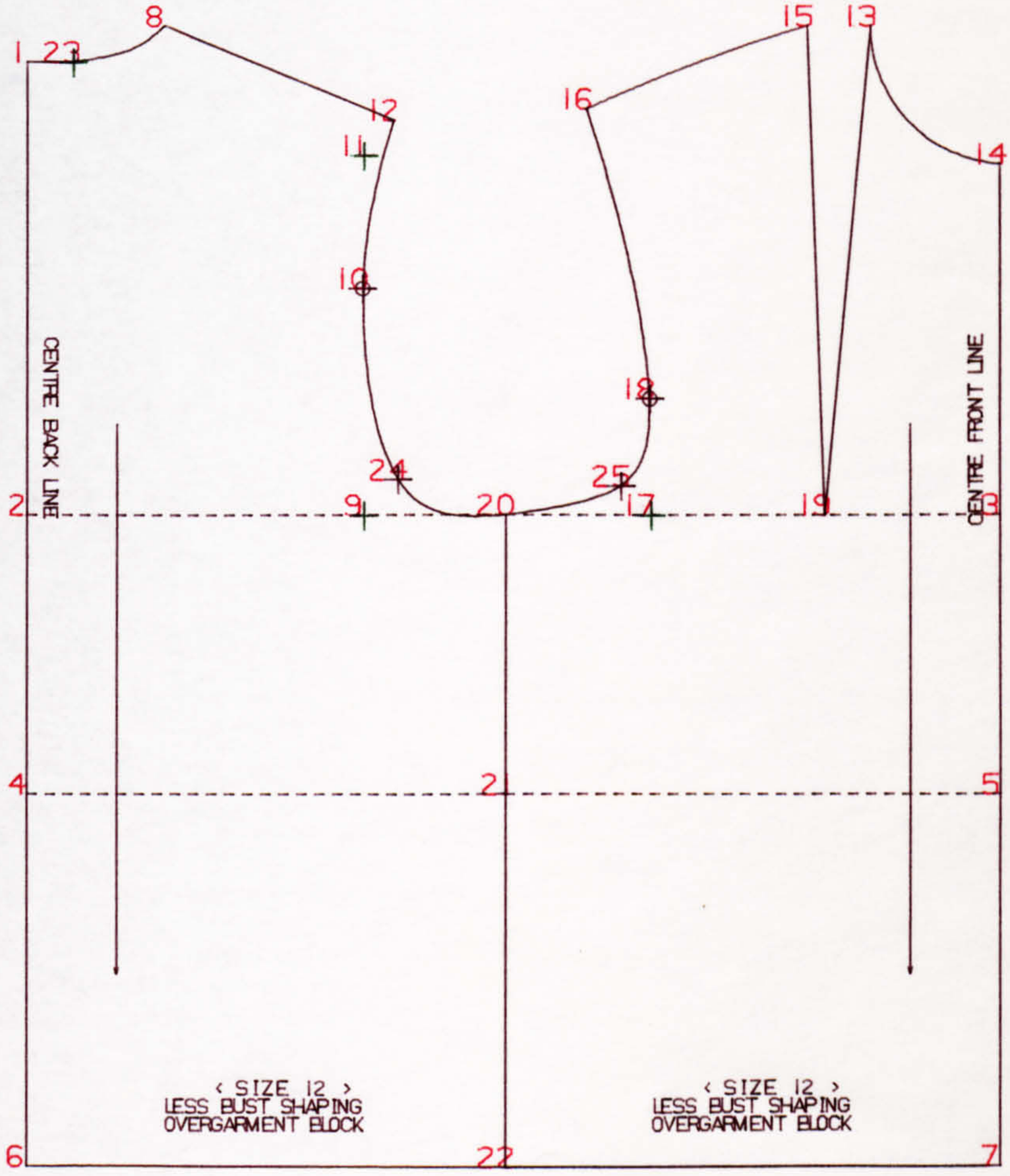


FIGURE 5-7-2: OVERGARMENT BLOCK, HALF DART

SCALE: 1/5

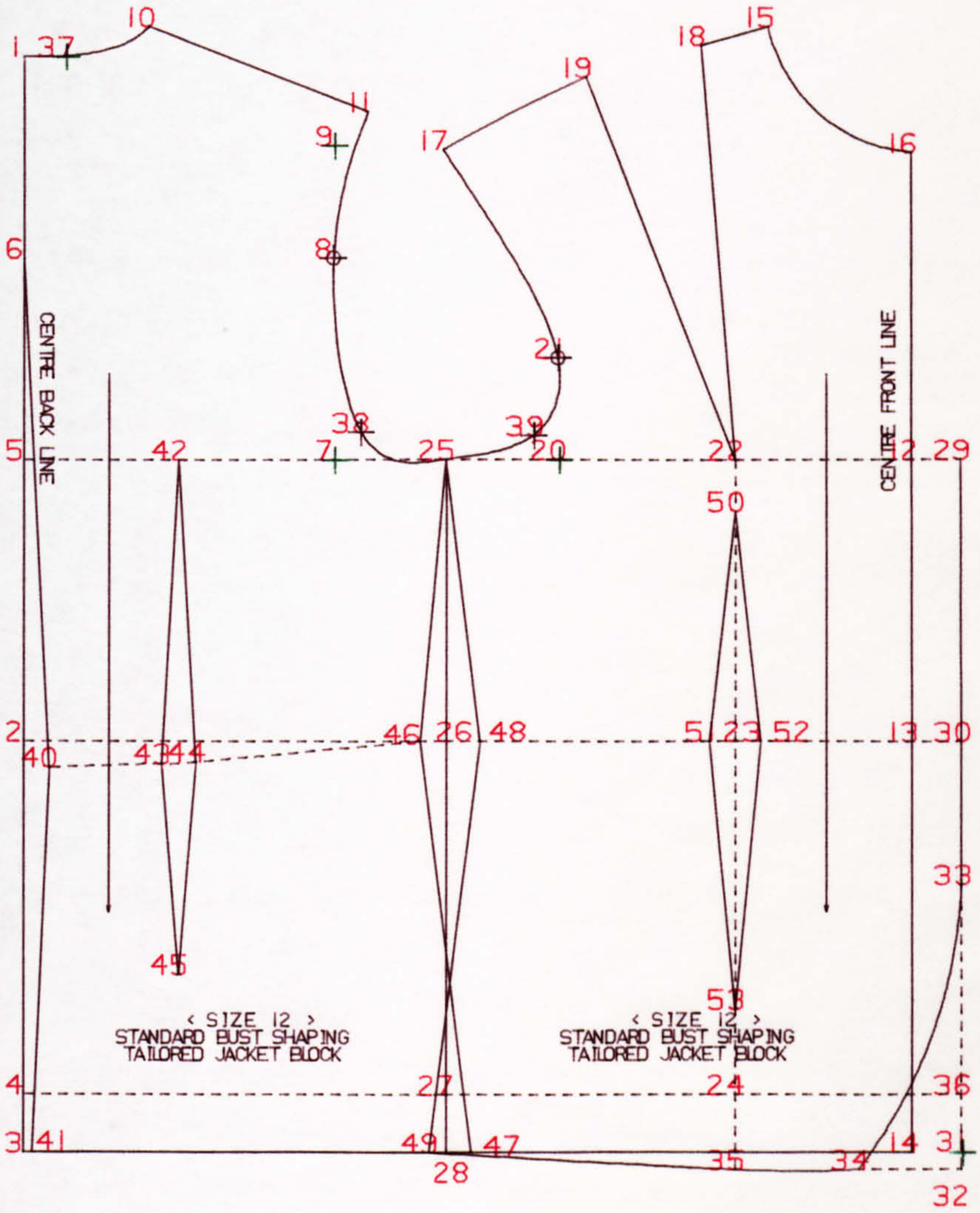


FIGURE 5-8-1: TAILORED JACKET BLOCK, FULL DART

SCALE: 1/5

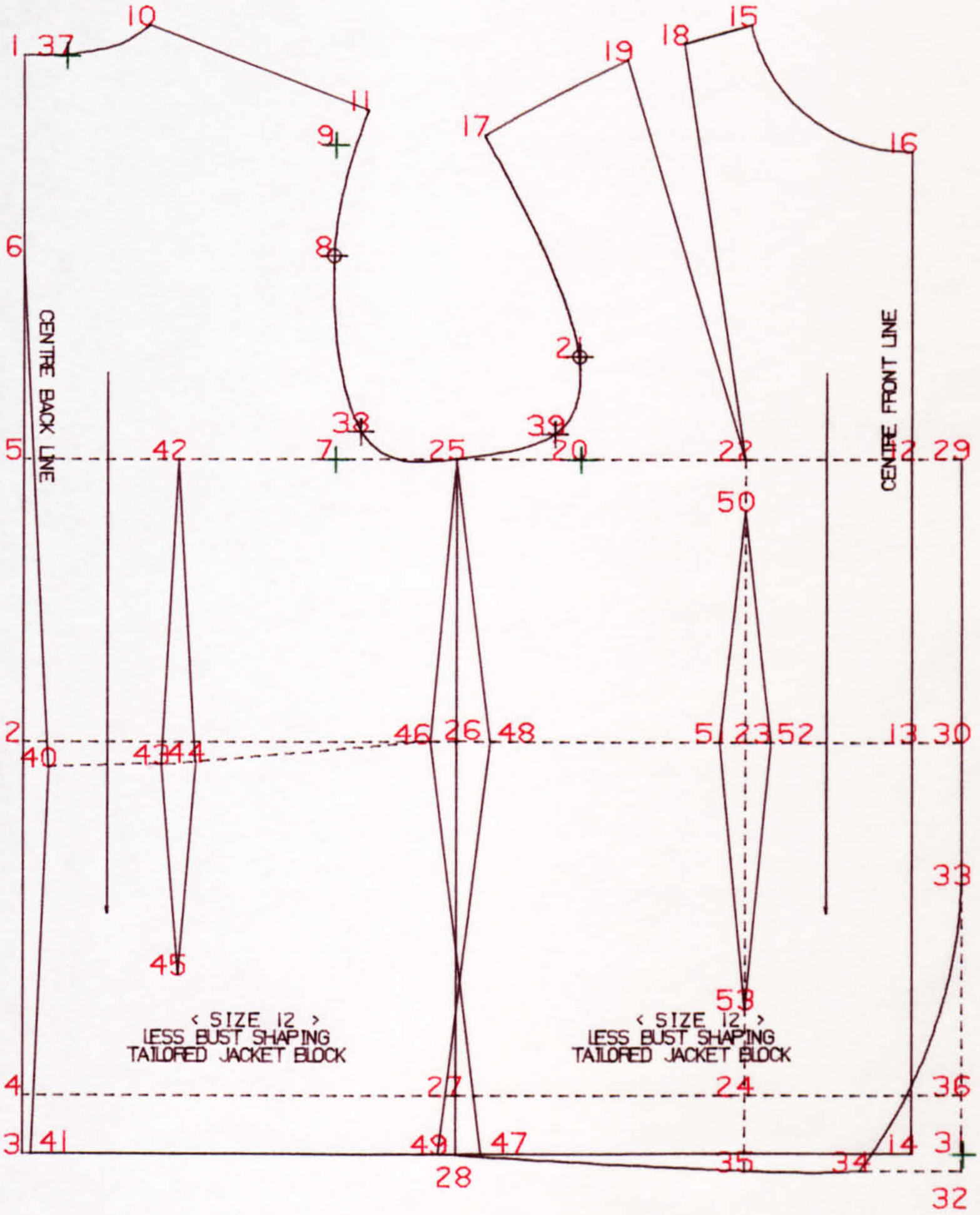


FIGURE 5-8-2: TAILORED JACKET BLOCK, HALF DART

SCALE: 1/5

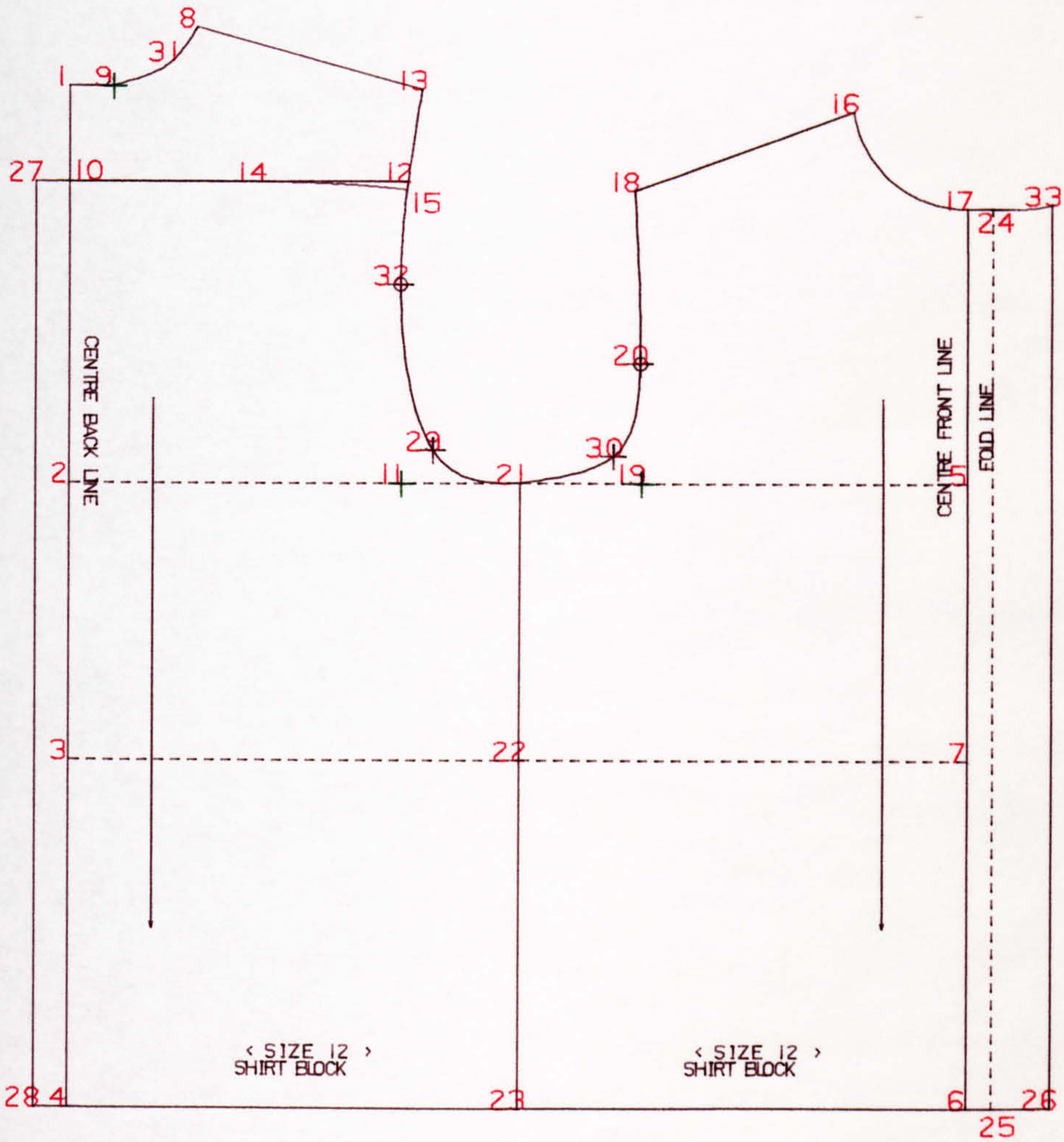


FIGURE 5-9: CLASSIC SHIRT BLOCK

SCALE: 1/5

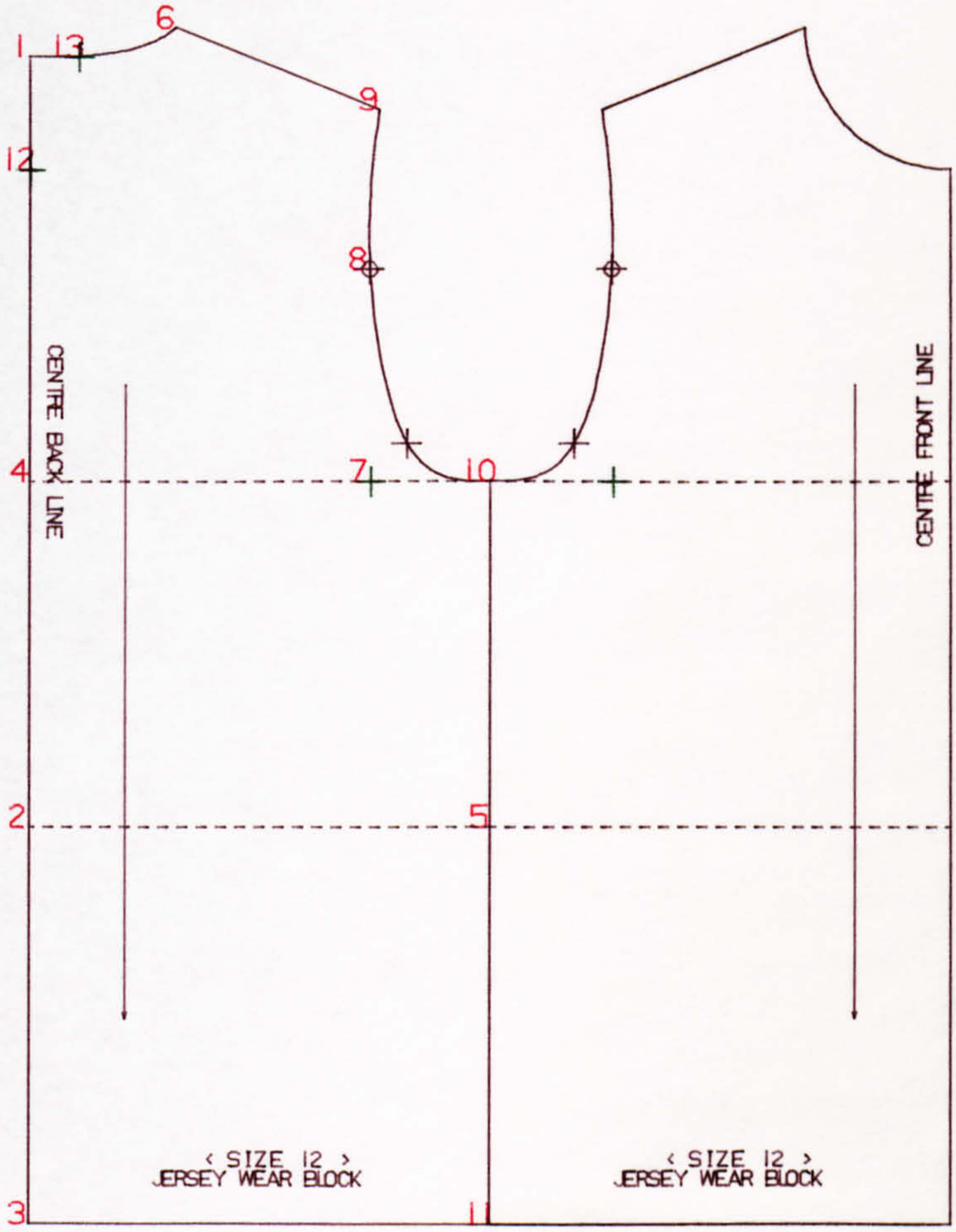


FIGURE 5-10: BLOCK PATTERN FOR JERSEY WEAR
SCALE: 1/5

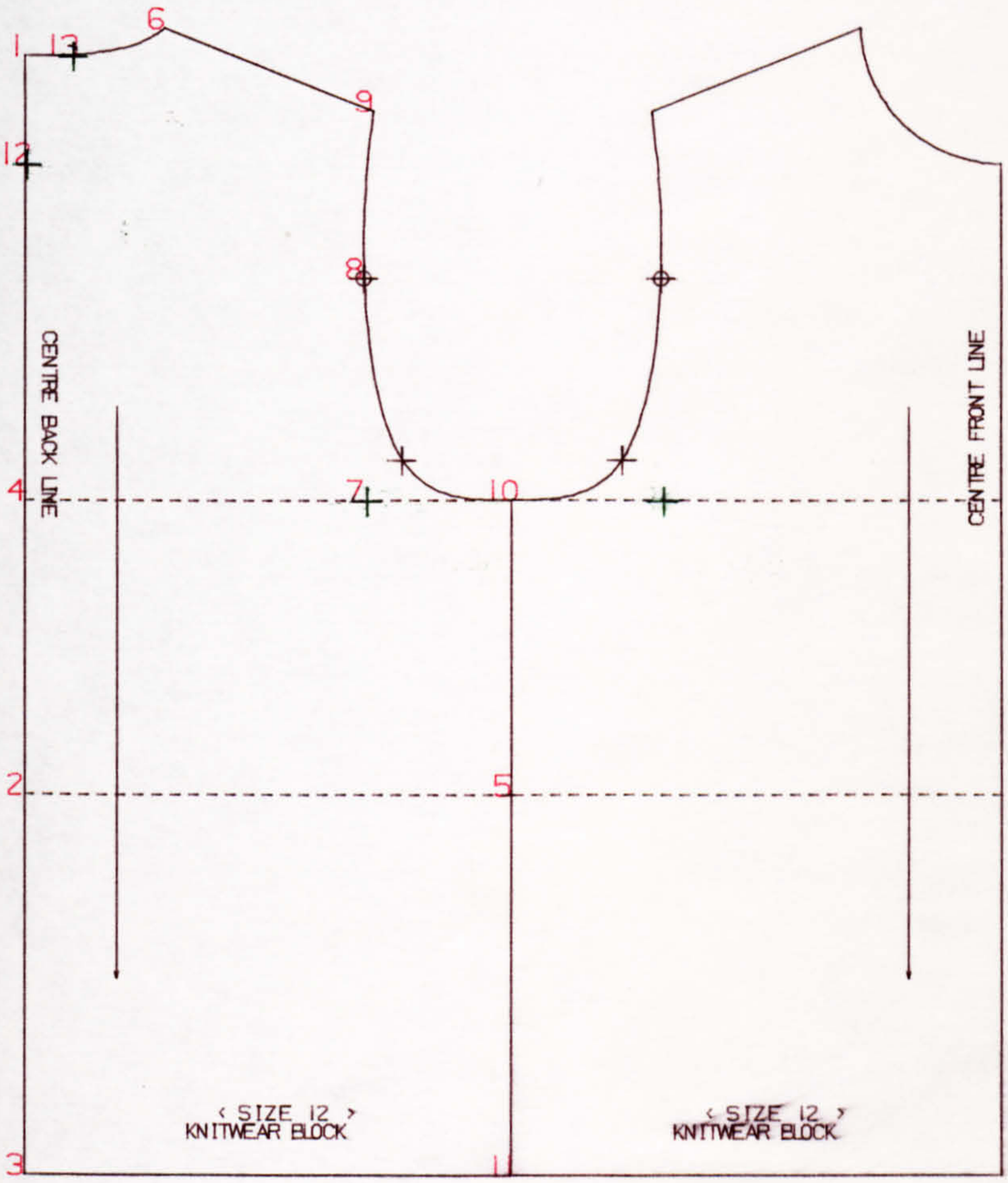


FIGURE 5-11 : BLOCK PATTERN FOR KNITWEAR

SCALE : 1/5

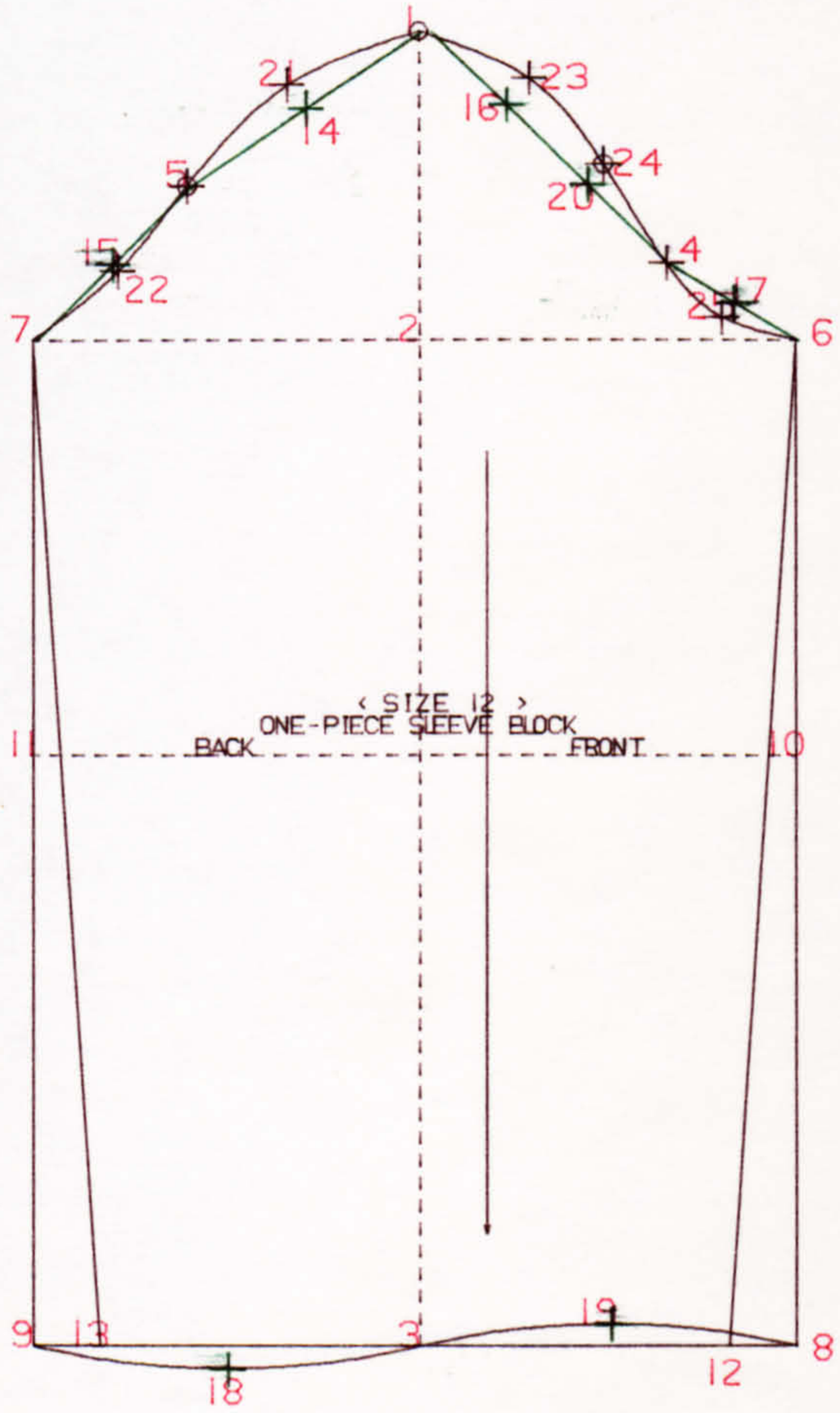


FIGURE 5-12: ONE-PIECE SLEEVE BLOCK

SCALE: 1/5

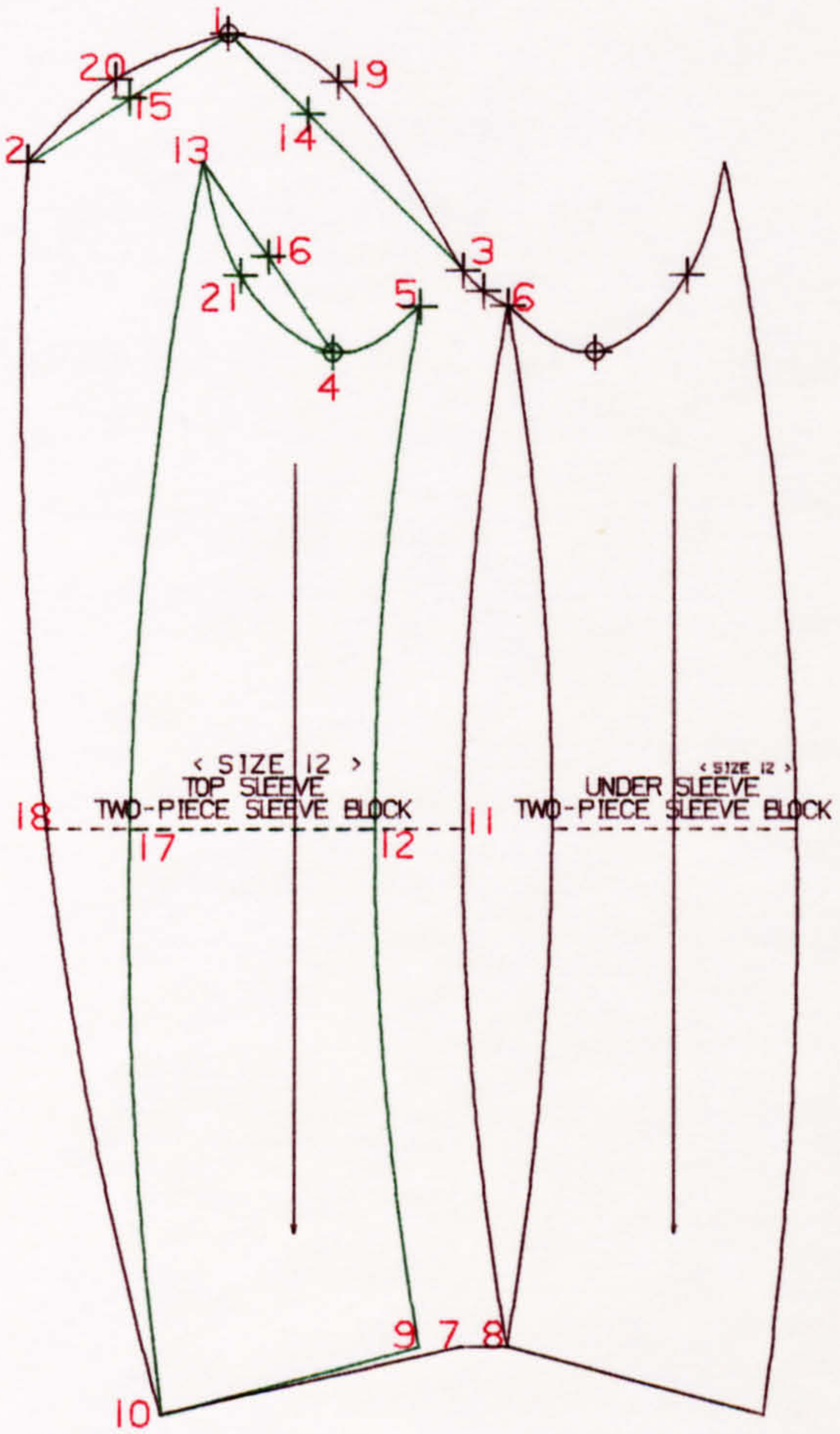


FIGURE 5-13: TWO-PIECE SLEEVE BLOCK

SCALE : 1/5

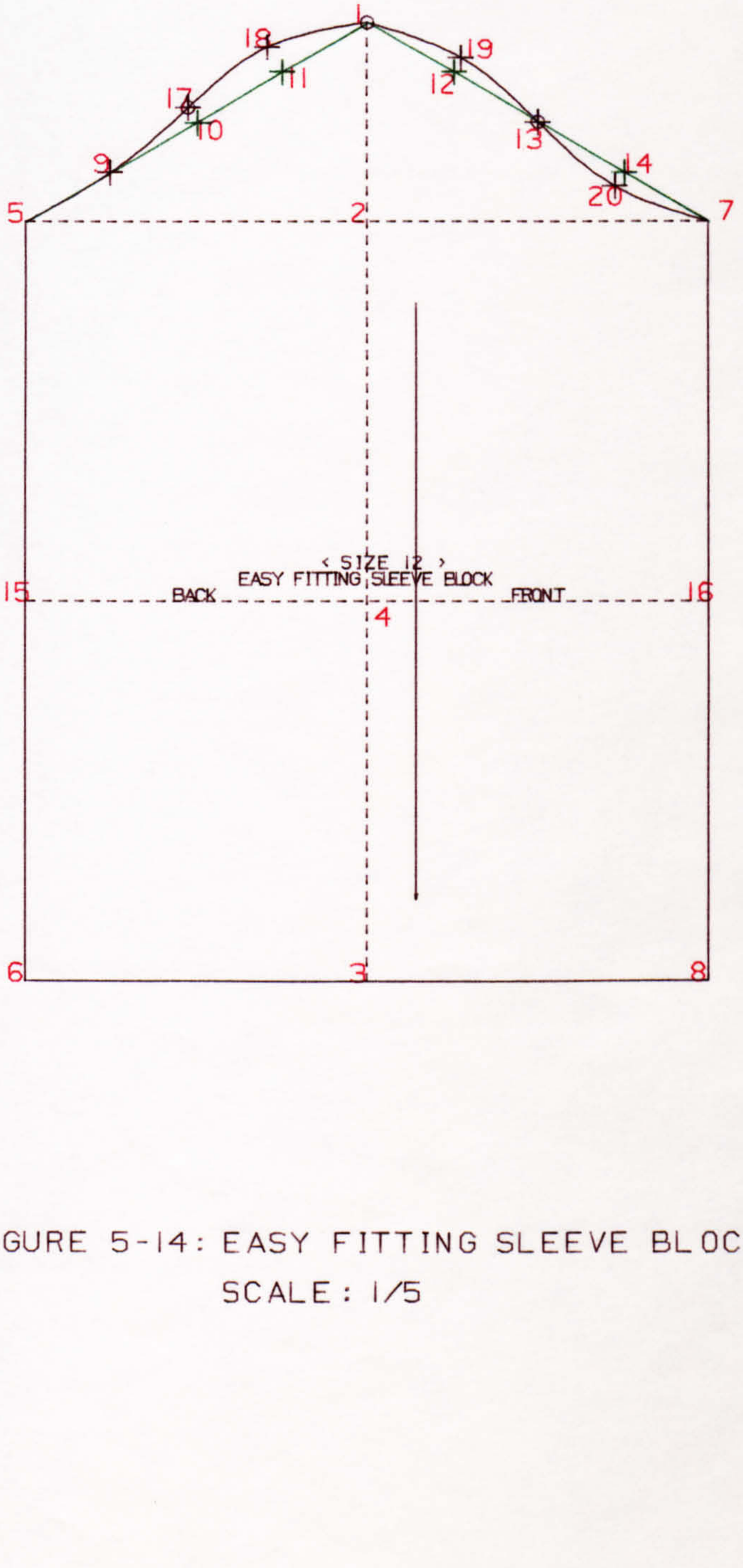


FIGURE 5-14: EASY FITTING SLEEVE BLOCK

SCALE: 1/5

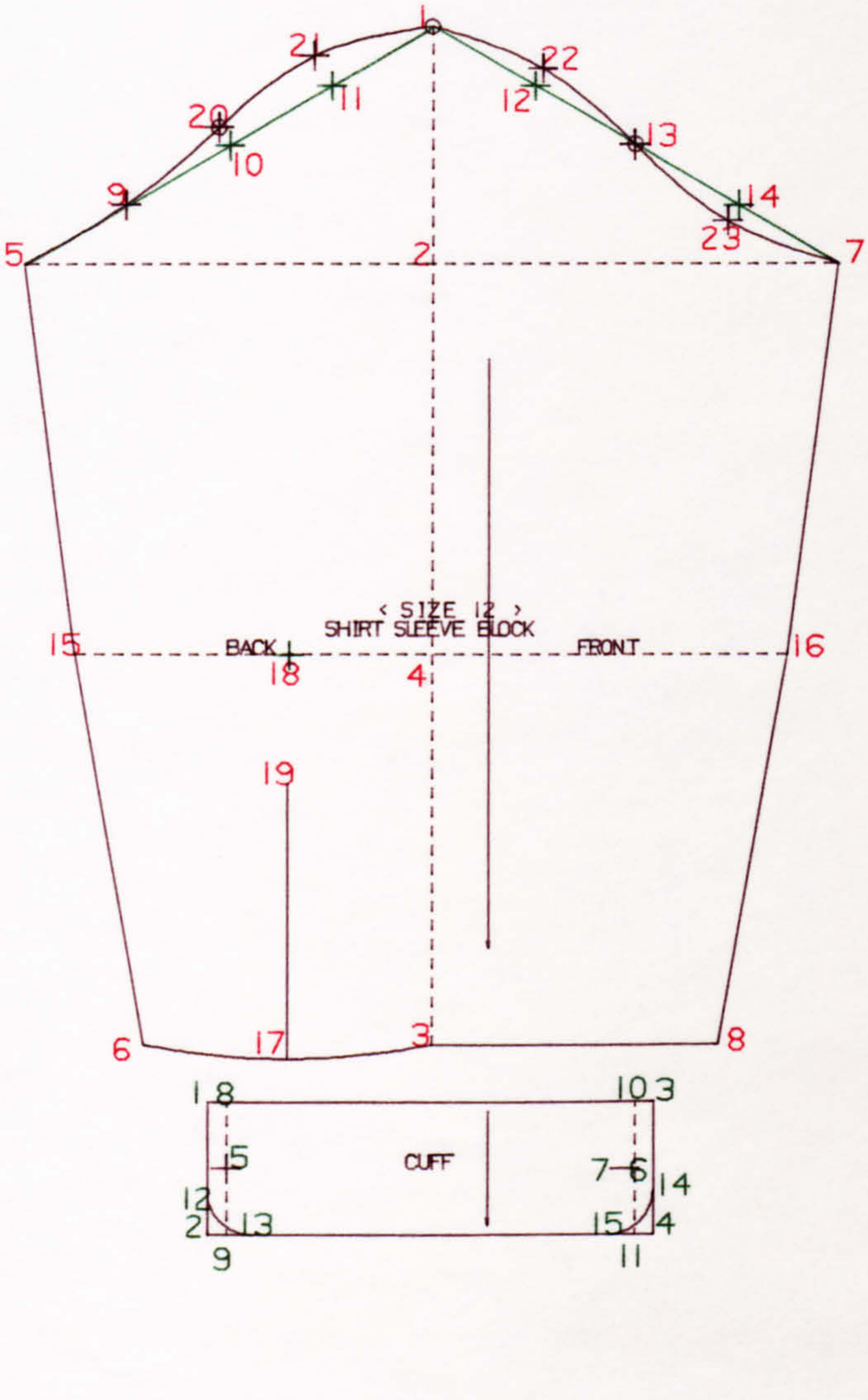


FIGURE 5-15: SHIRT SLEEVE PATTERN

SCALE : 1/5

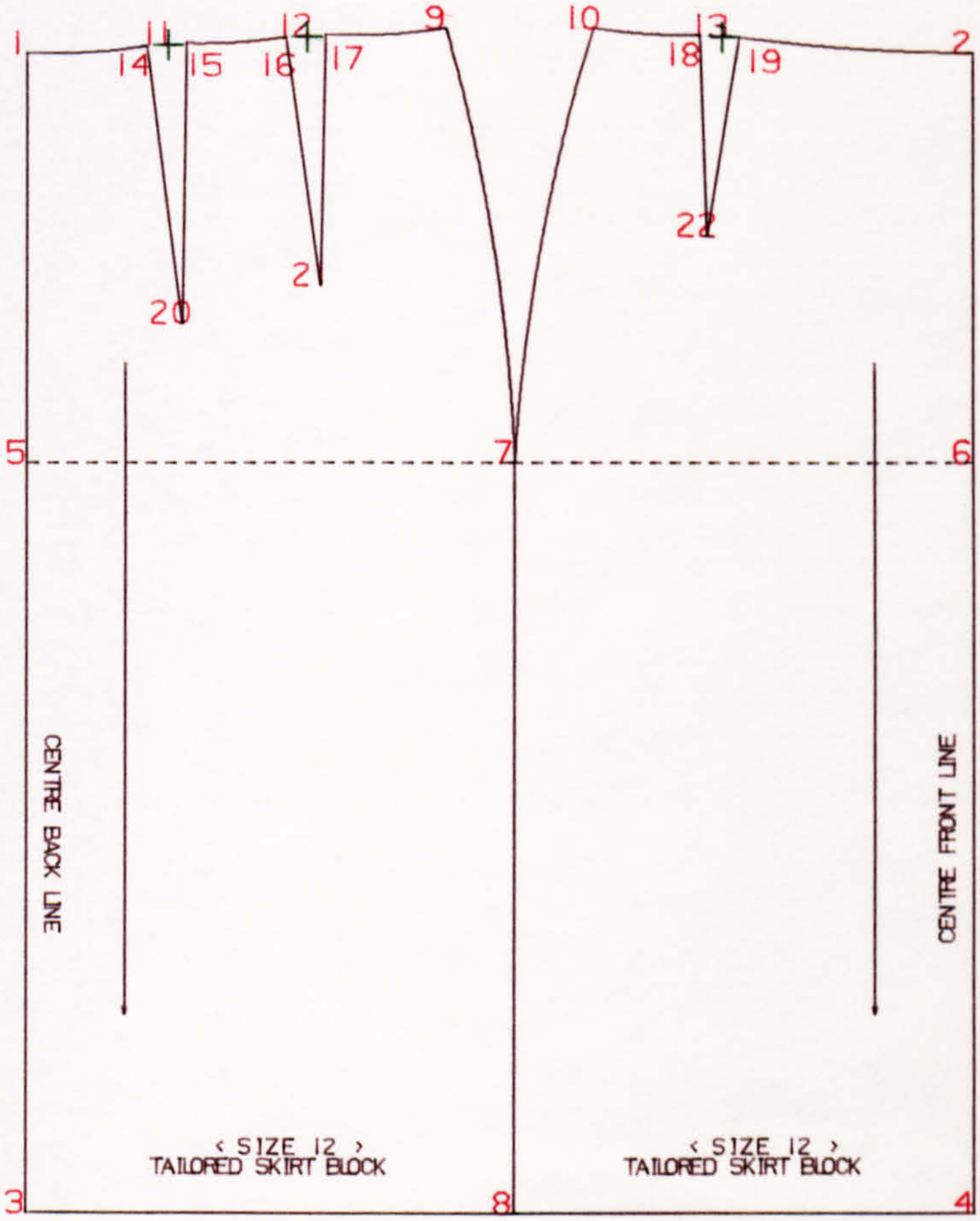


FIGURE 5-16: TAILORED SKIRT BLOCK
SCALE: 1/5

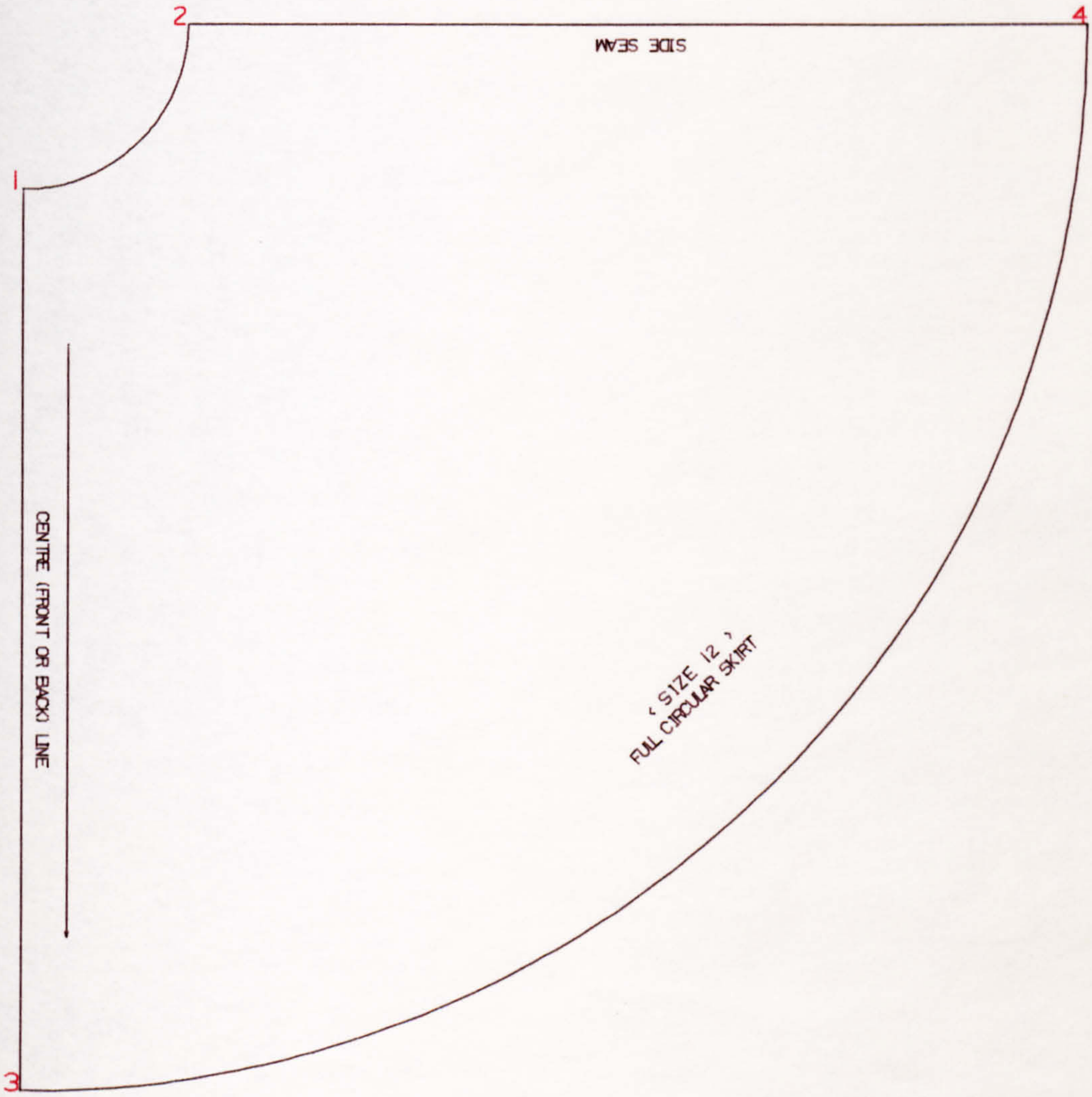


FIGURE 5-17-1: FULL CIRCULAR SKIRT PATTERN

SCALE: 1/5

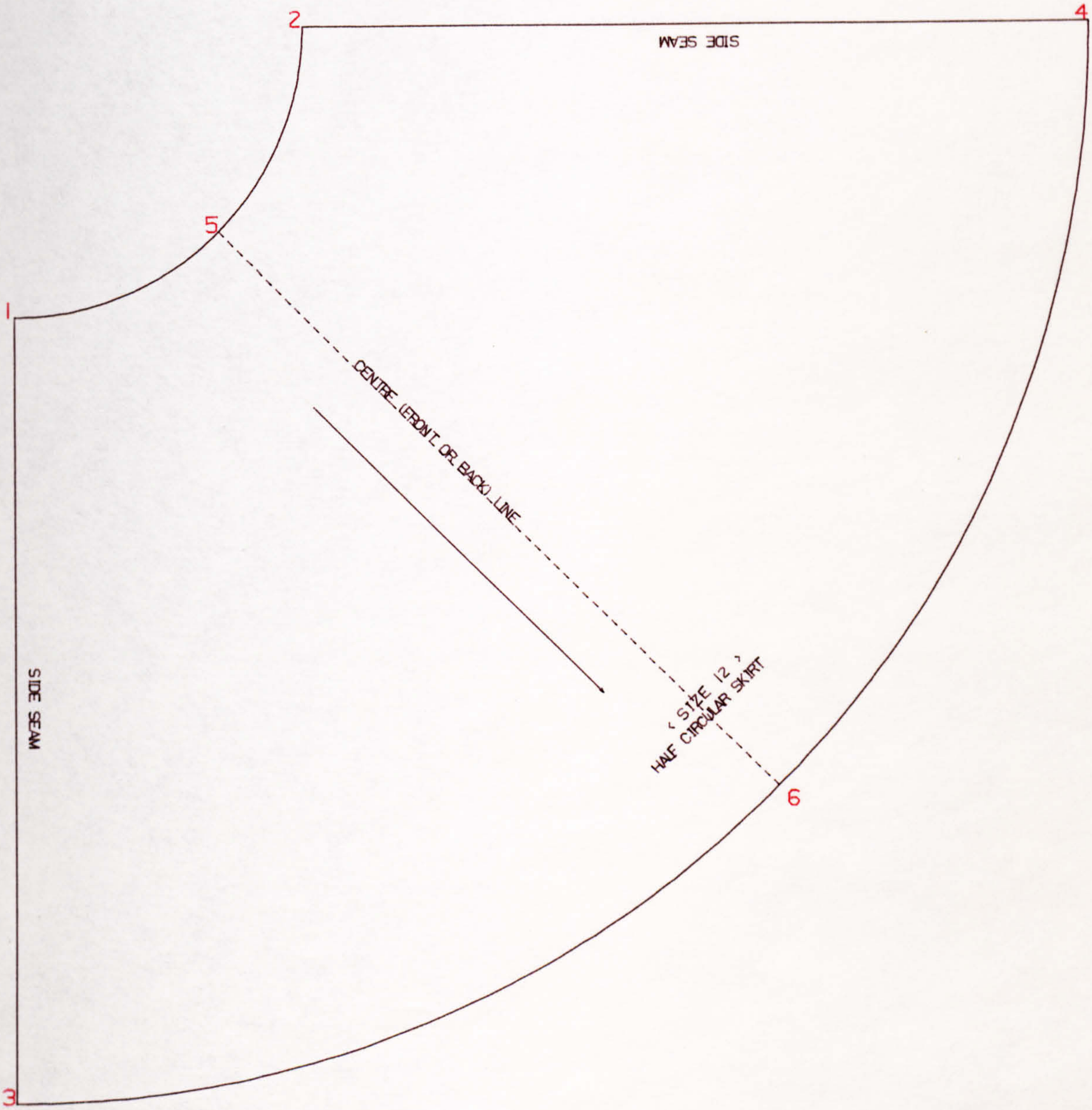


FIGURE 5-17-2: HALF CIRCULAR SKIRT PATTERN

SCALE: 1/5

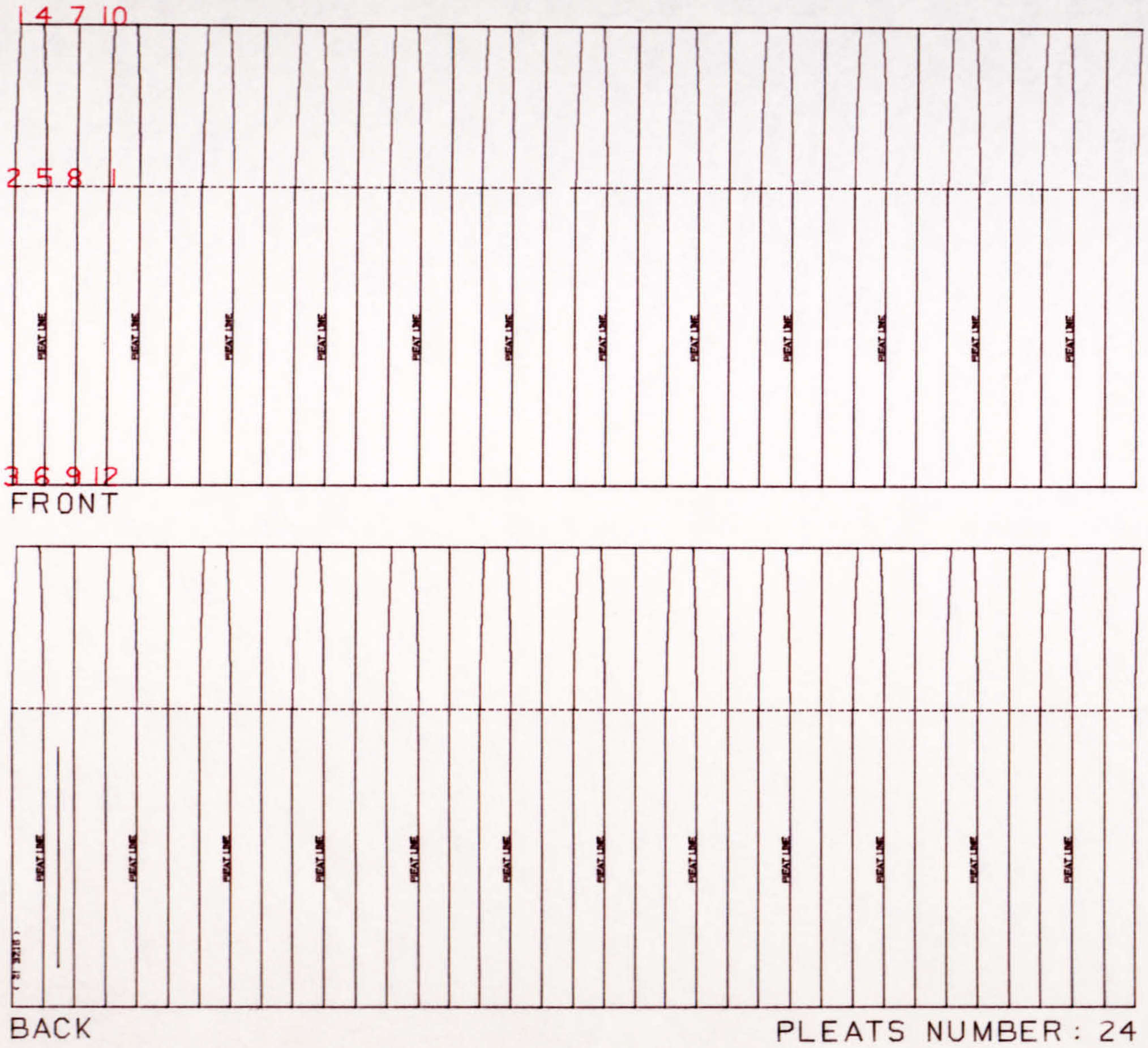


FIGURE 5-18 : PLEATED SKIRT PATTERN

SCALE : 1/10

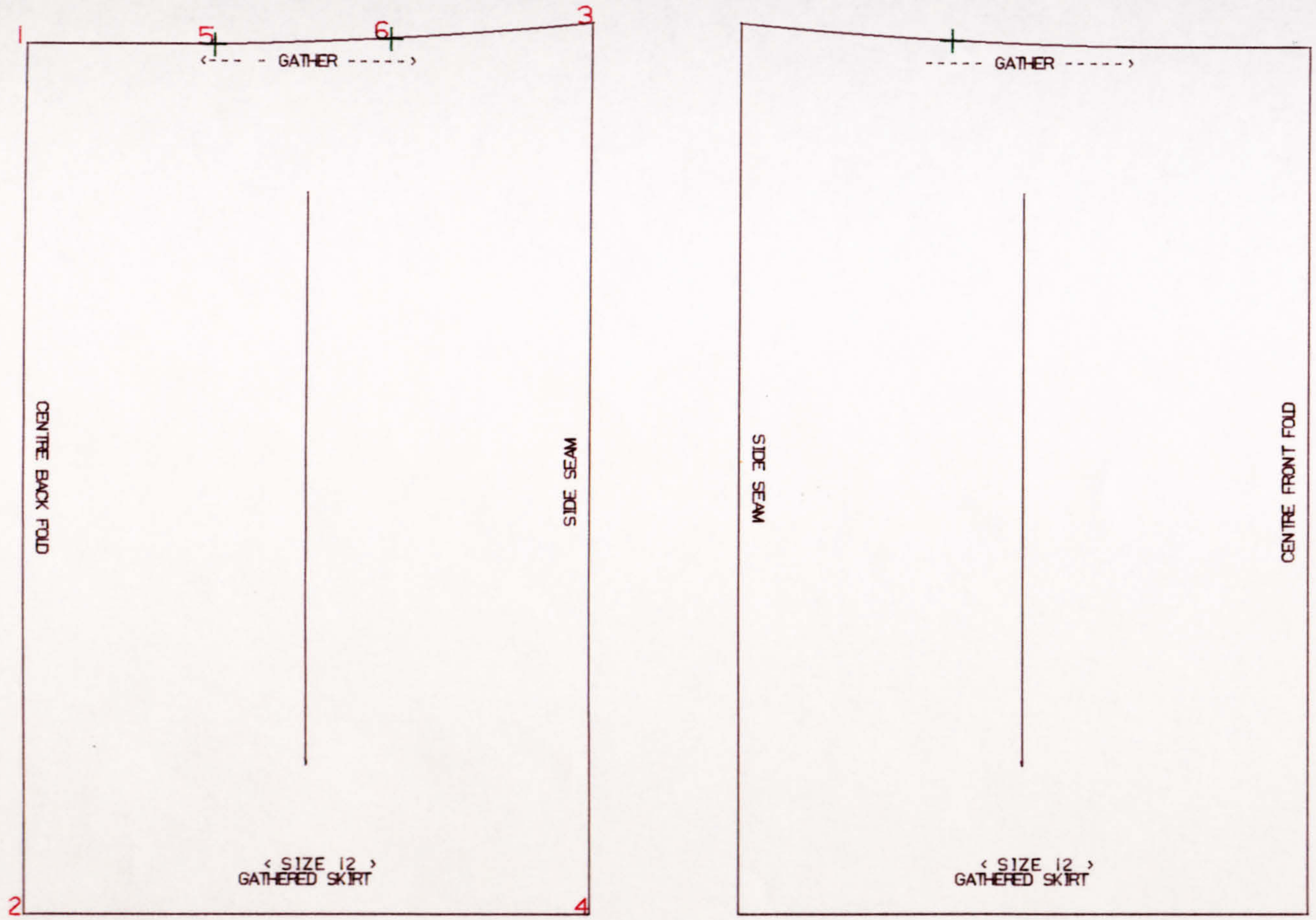


FIGURE 5-19-1: SLIGHTLY GATHERED SKIRT PATTERN

SCALE: 1/5

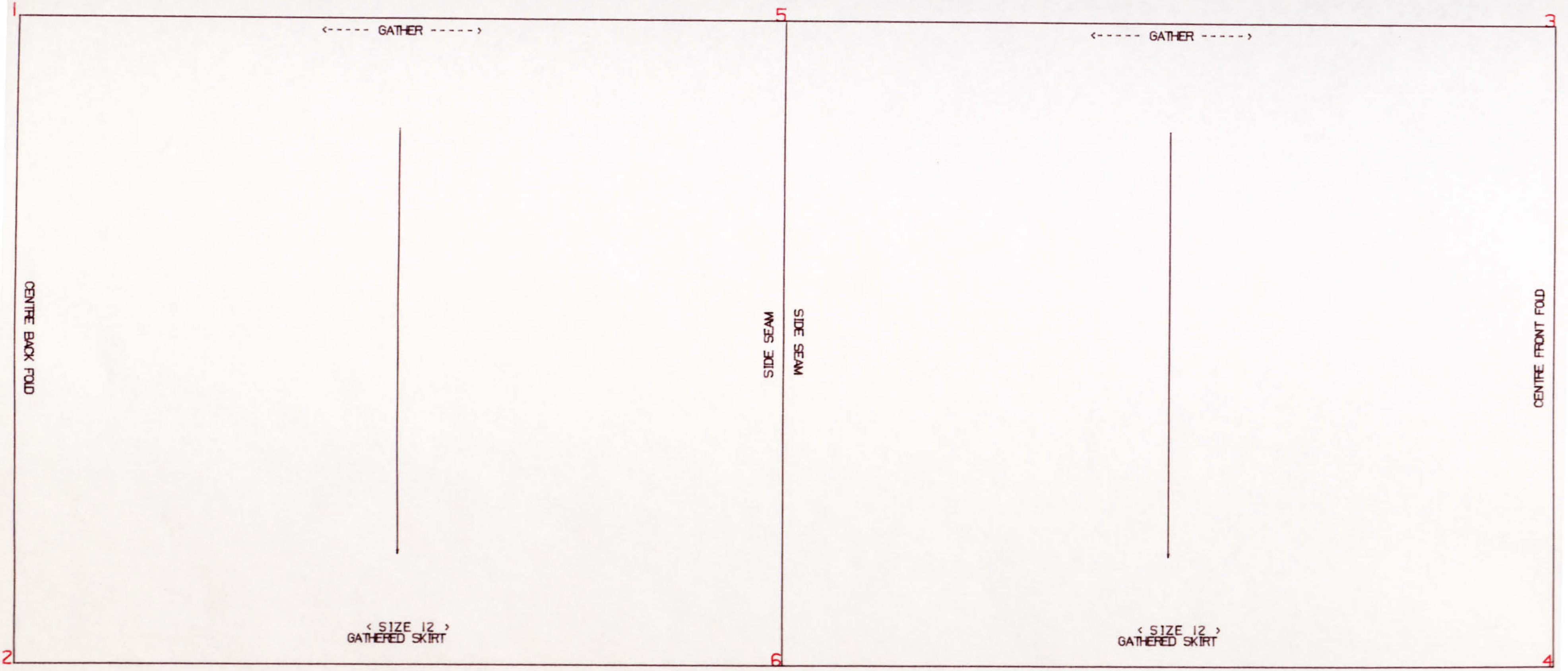


FIGURE 5-19-2: VERY GATHERED SKIRT PATTERN

SCALE: 1/5

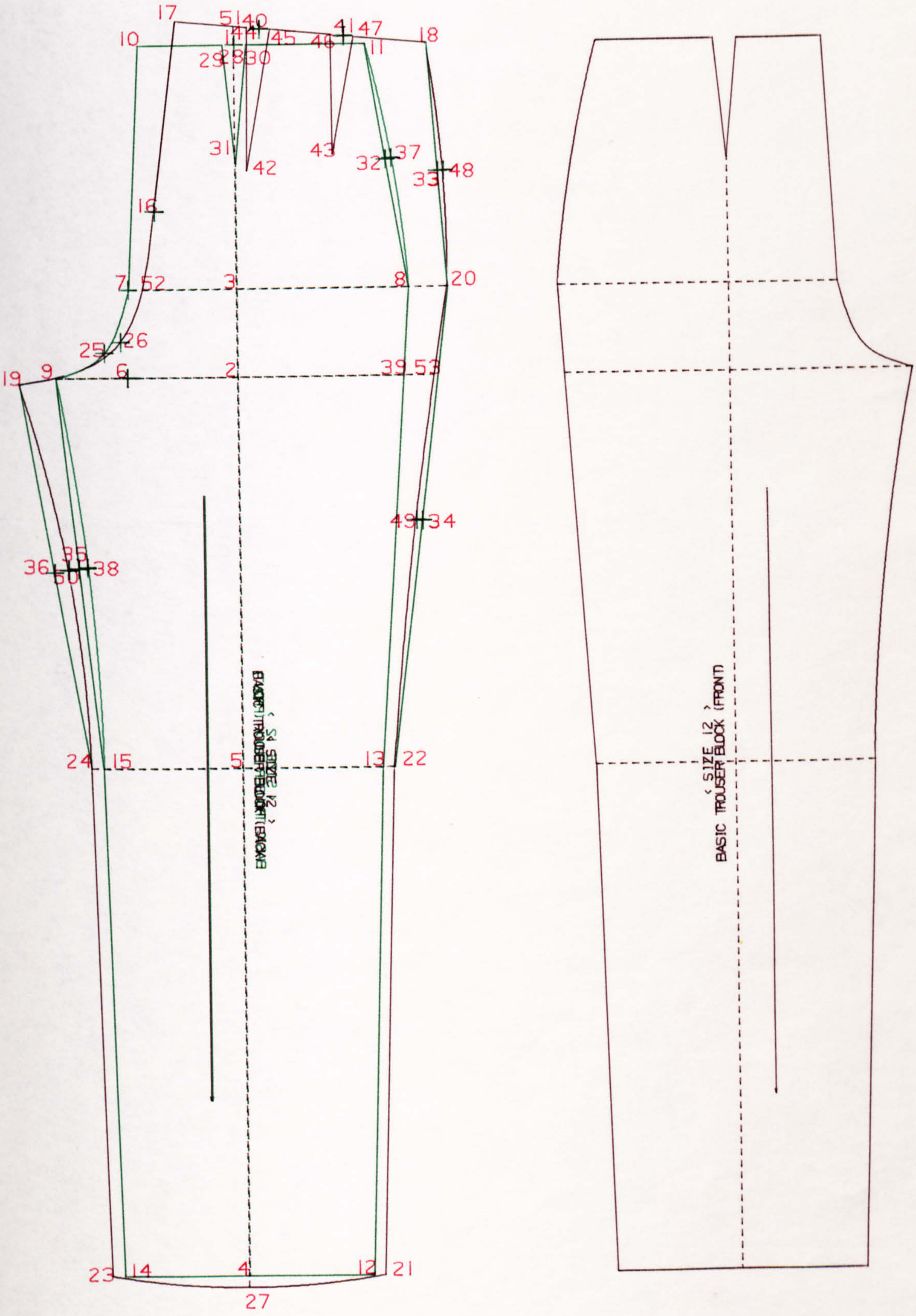


FIGURE 5-20: BASIC TROUSER BLOCK

SACLE: 1/5

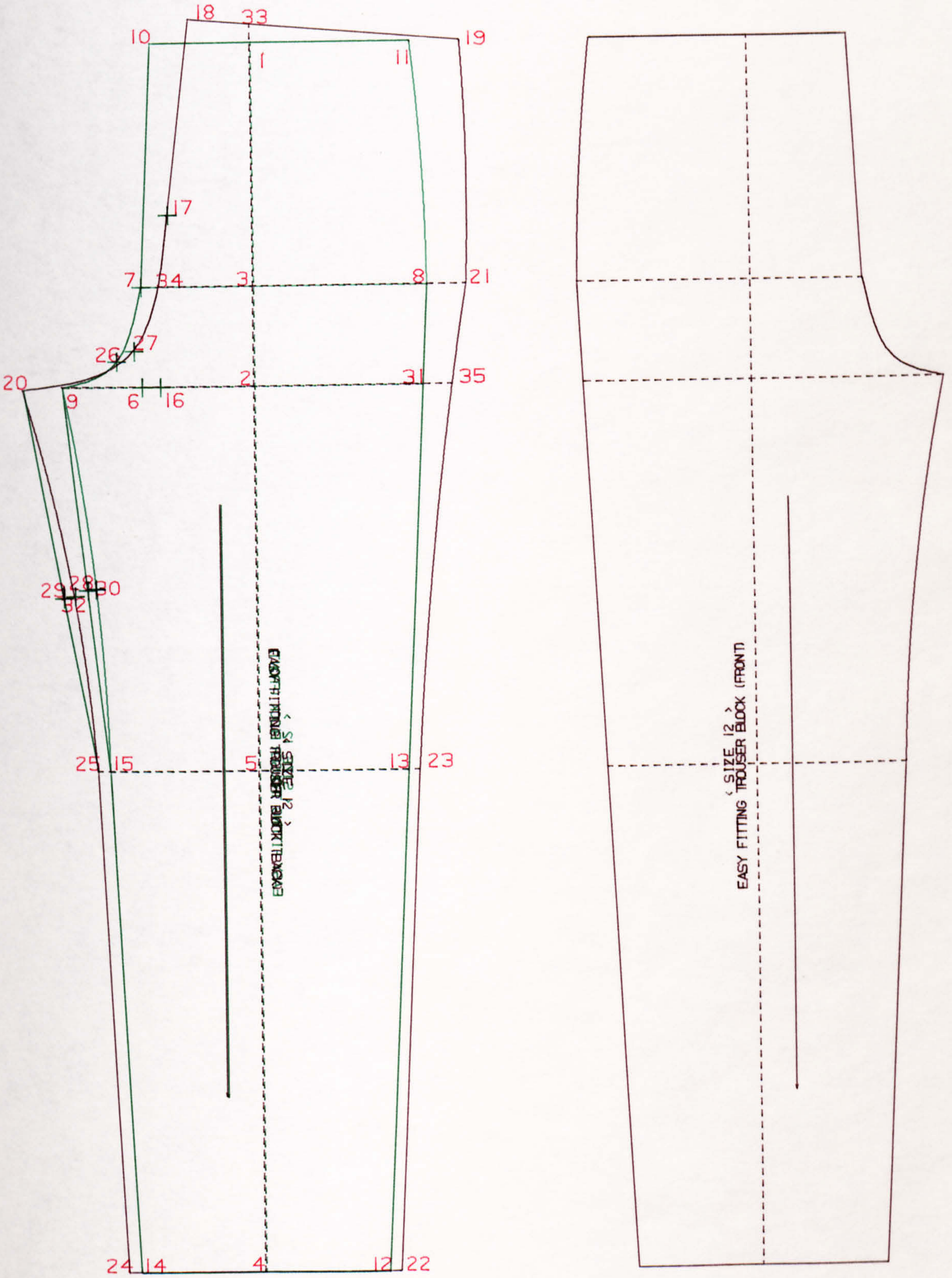


FIGURE 5-21: EASY FITTING TROUSER BLOCK

SCALE: 1/5

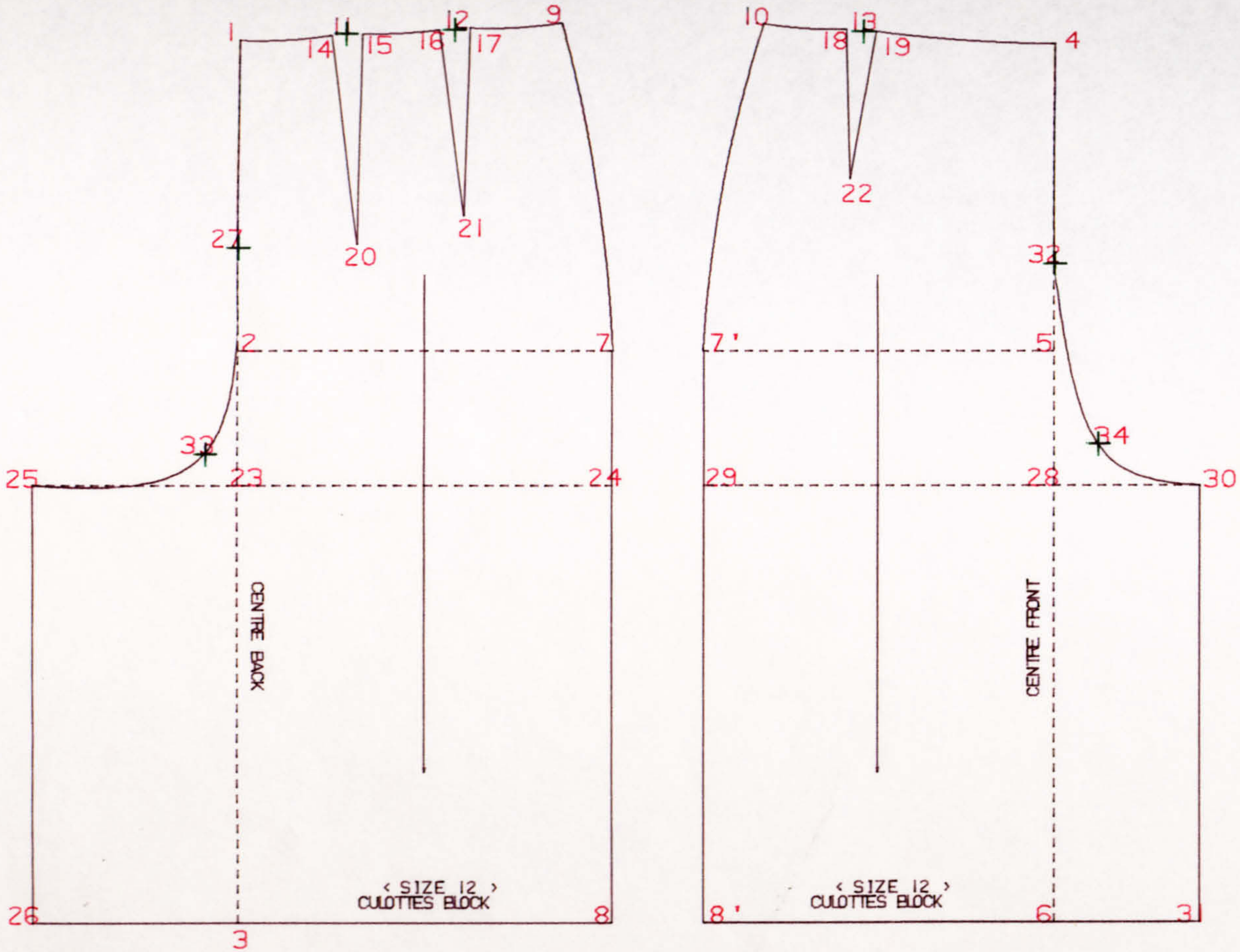


FIGURE 5-22: CULOTTES PATTERN

SCALE: 1/5

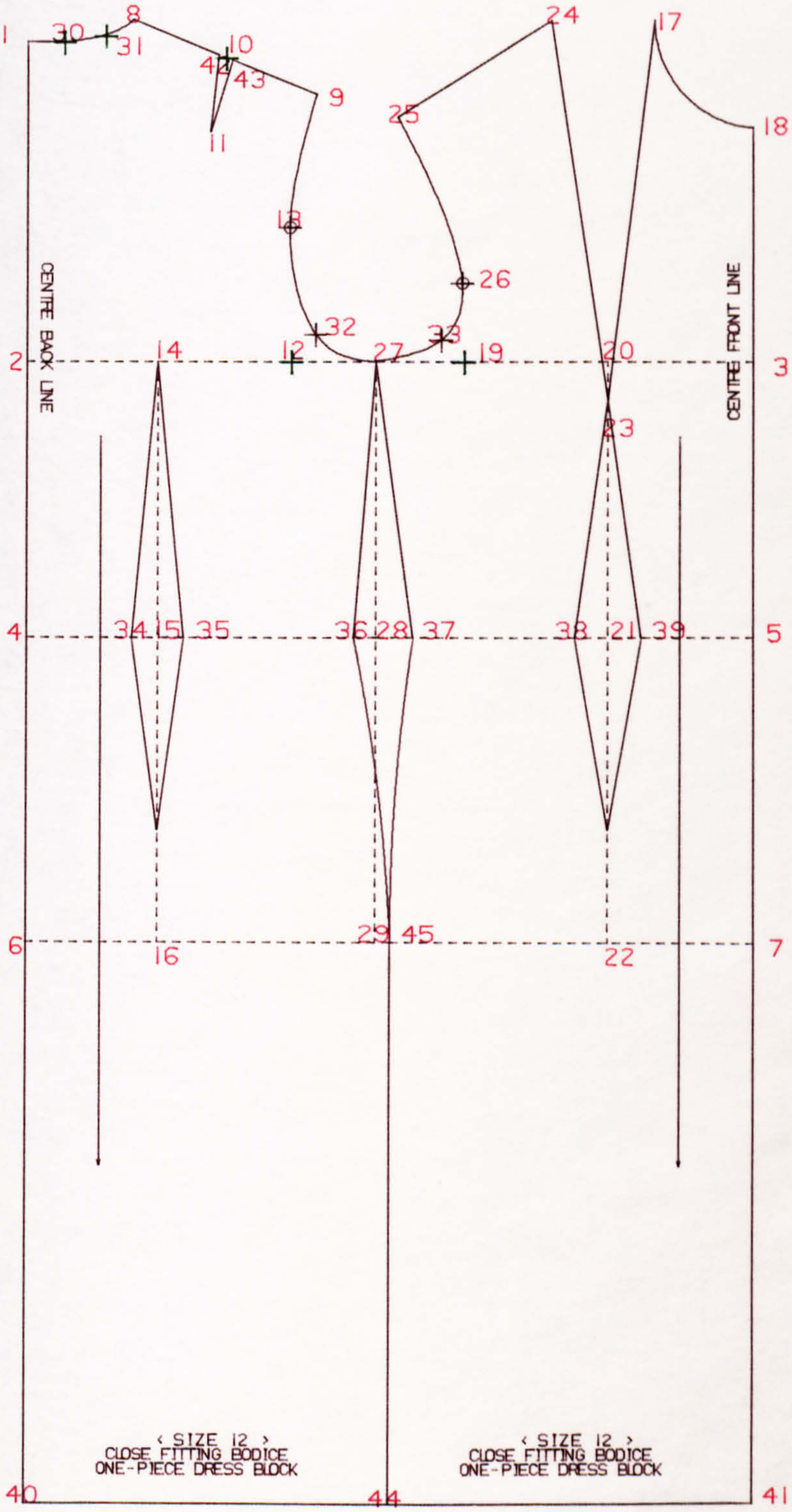


FIGURE 5-23-1: CLOSE FITTING ONE-PIECE DRESS BLOCK WITH WAIST SHAPING, SCALE: 1/5

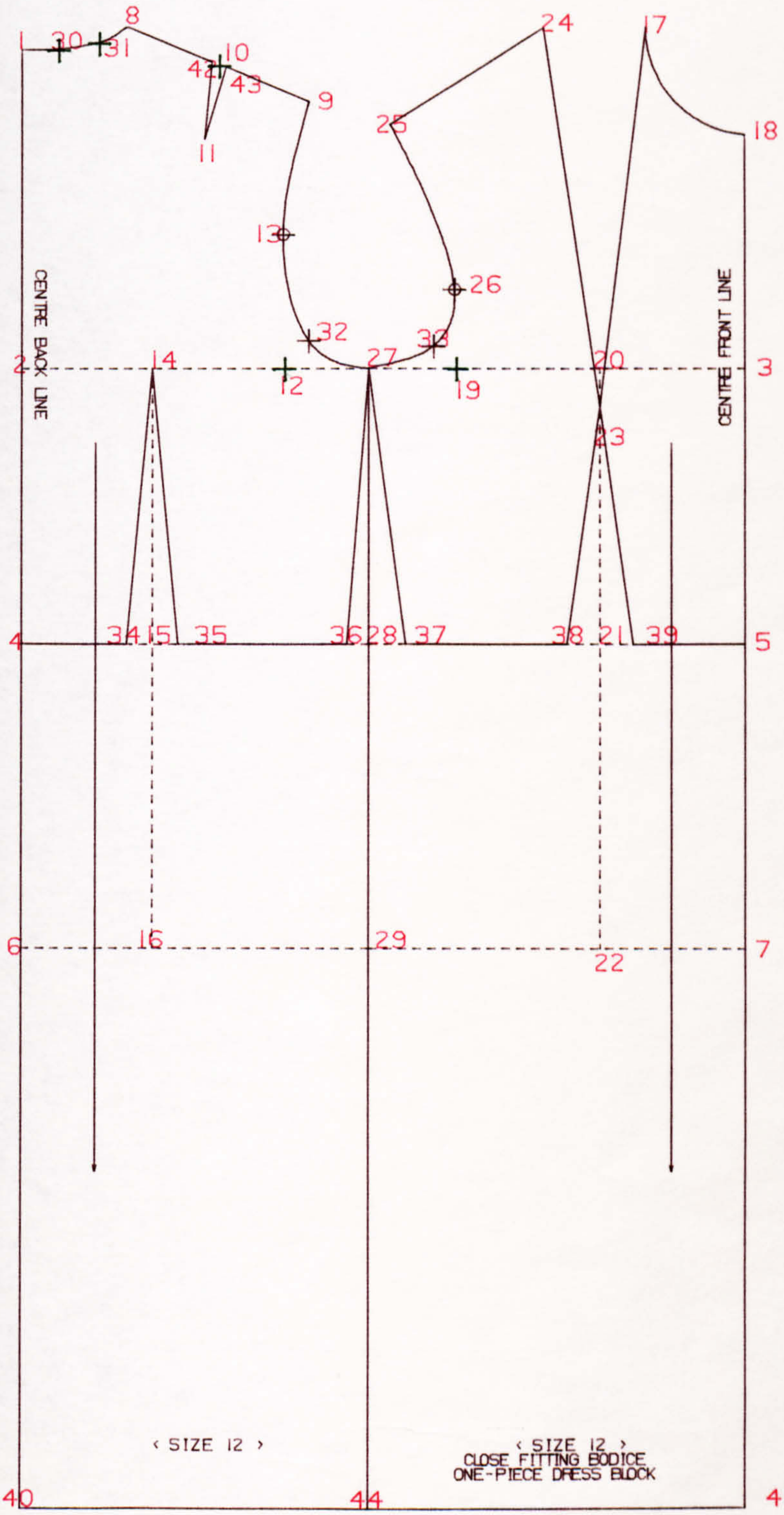


FIGURE 5-23-2: CLOSE FITTING ONE-PIECE DRESS BLOCK
WITHOUT WAIST SHAPING, SCALE: 1/5

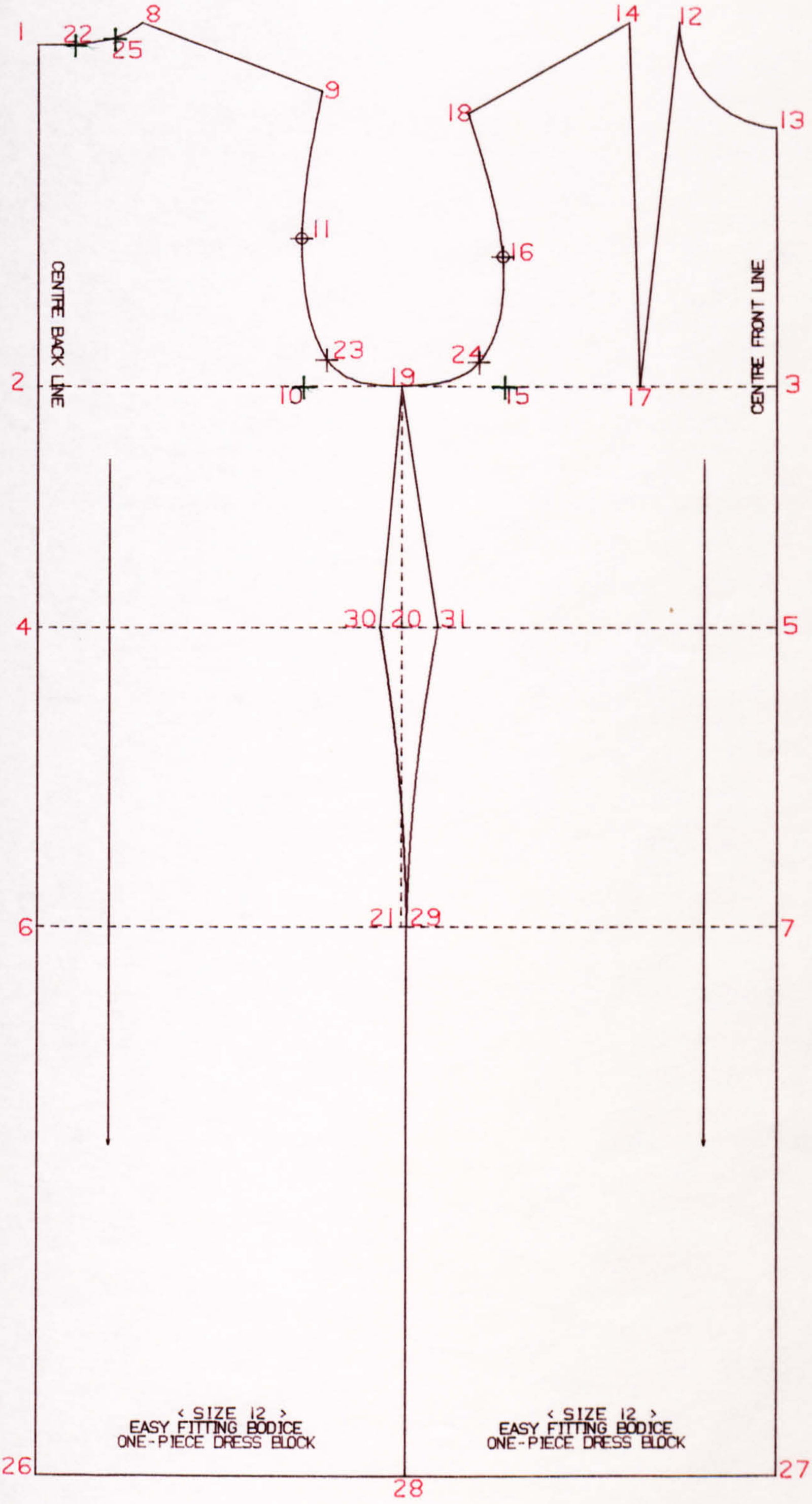


FIGURE 5-24-1: EASY FITTING ONE-PIECE DRESS BLOCK WITH WAIST SHAPING, SCALE: 1/5

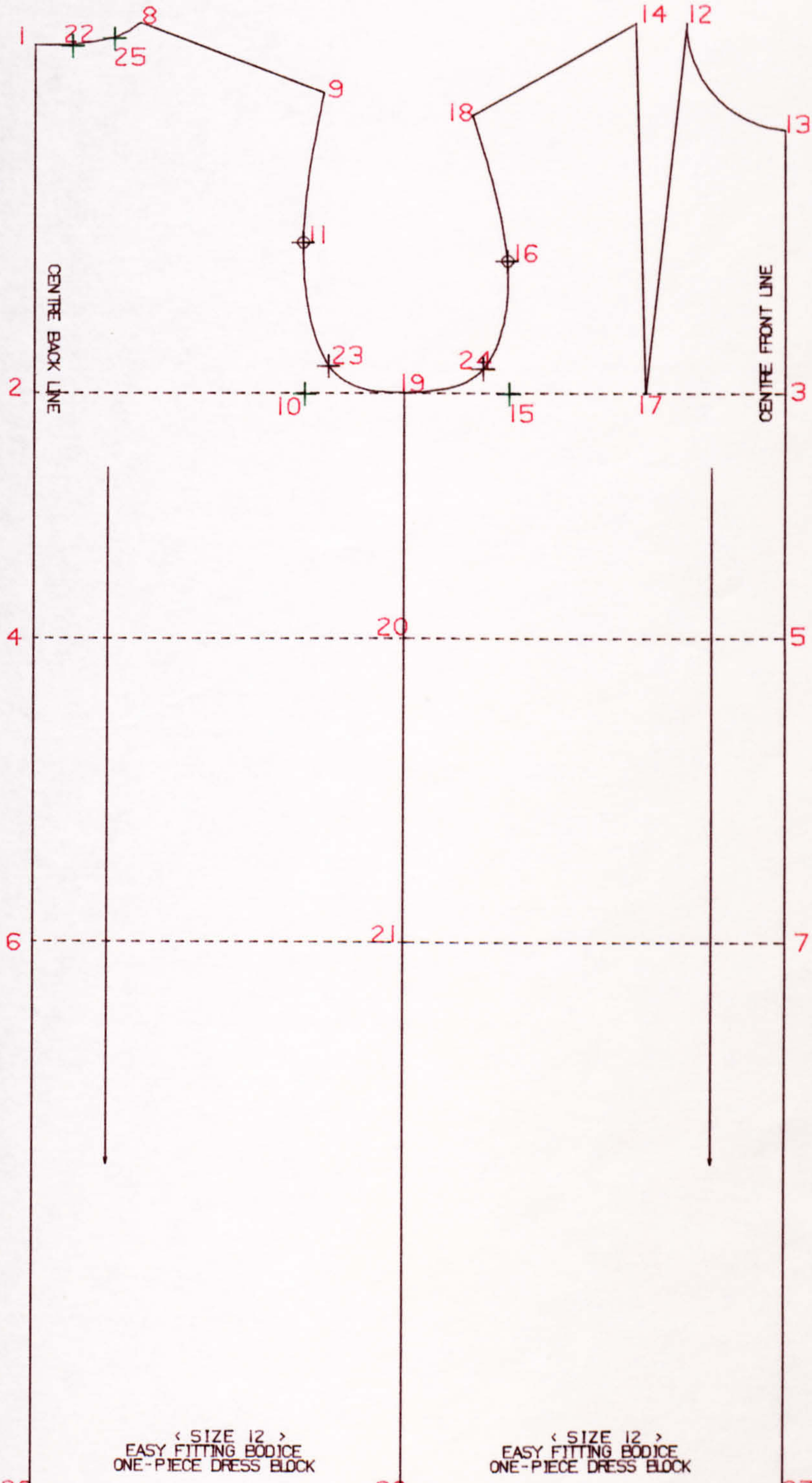


FIGURE 5-24-2: EASY FITTING ONE-PIECE DRESS BLOCK
WITHOUT WAIST SHAPING, SCALE: 1/5

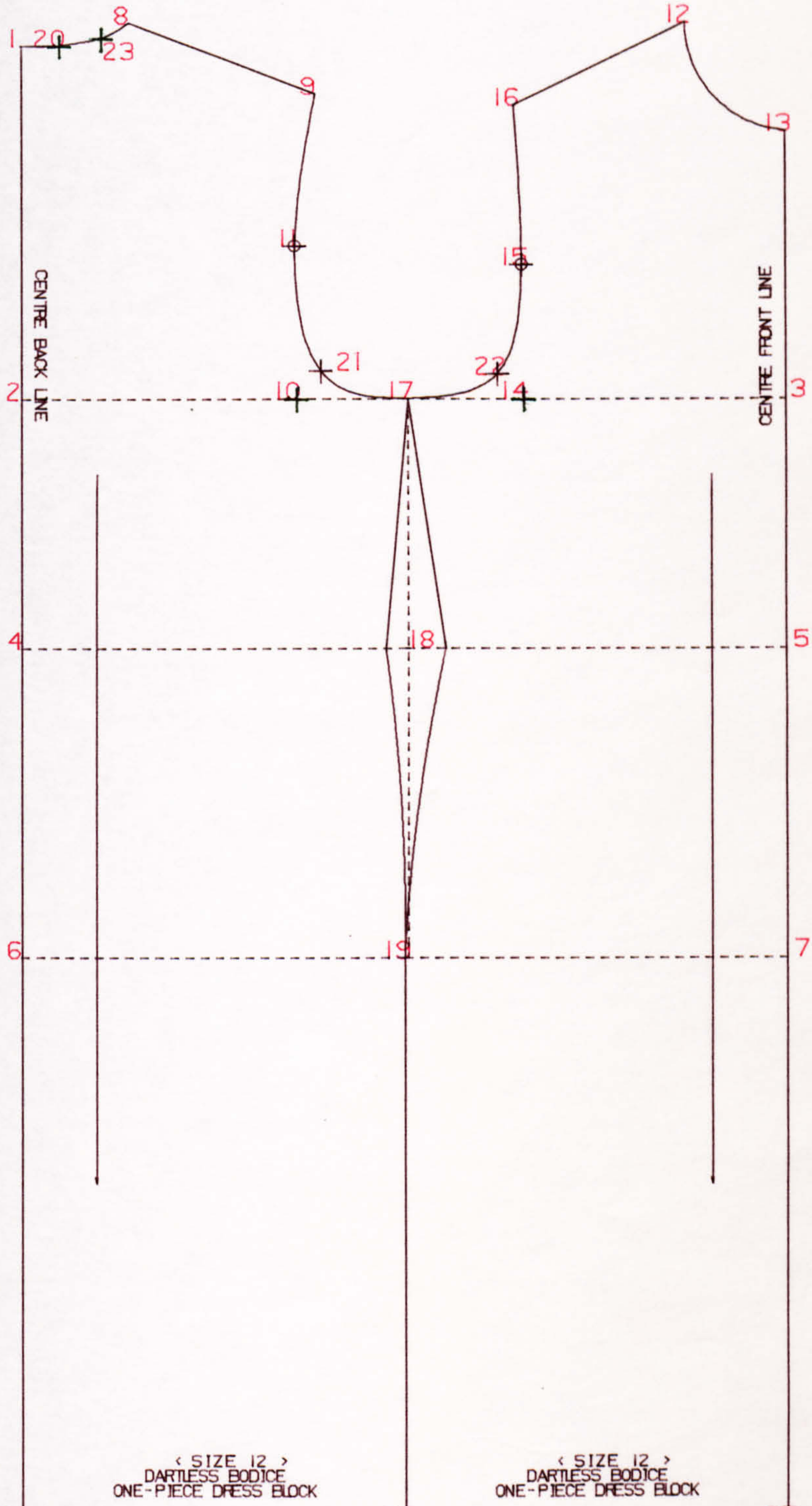


FIGURE 5-25-1: DARTLESS ONE-PIECE DRESS BLOCK
WITH WAIST SHAPING, SCALE: 1/5

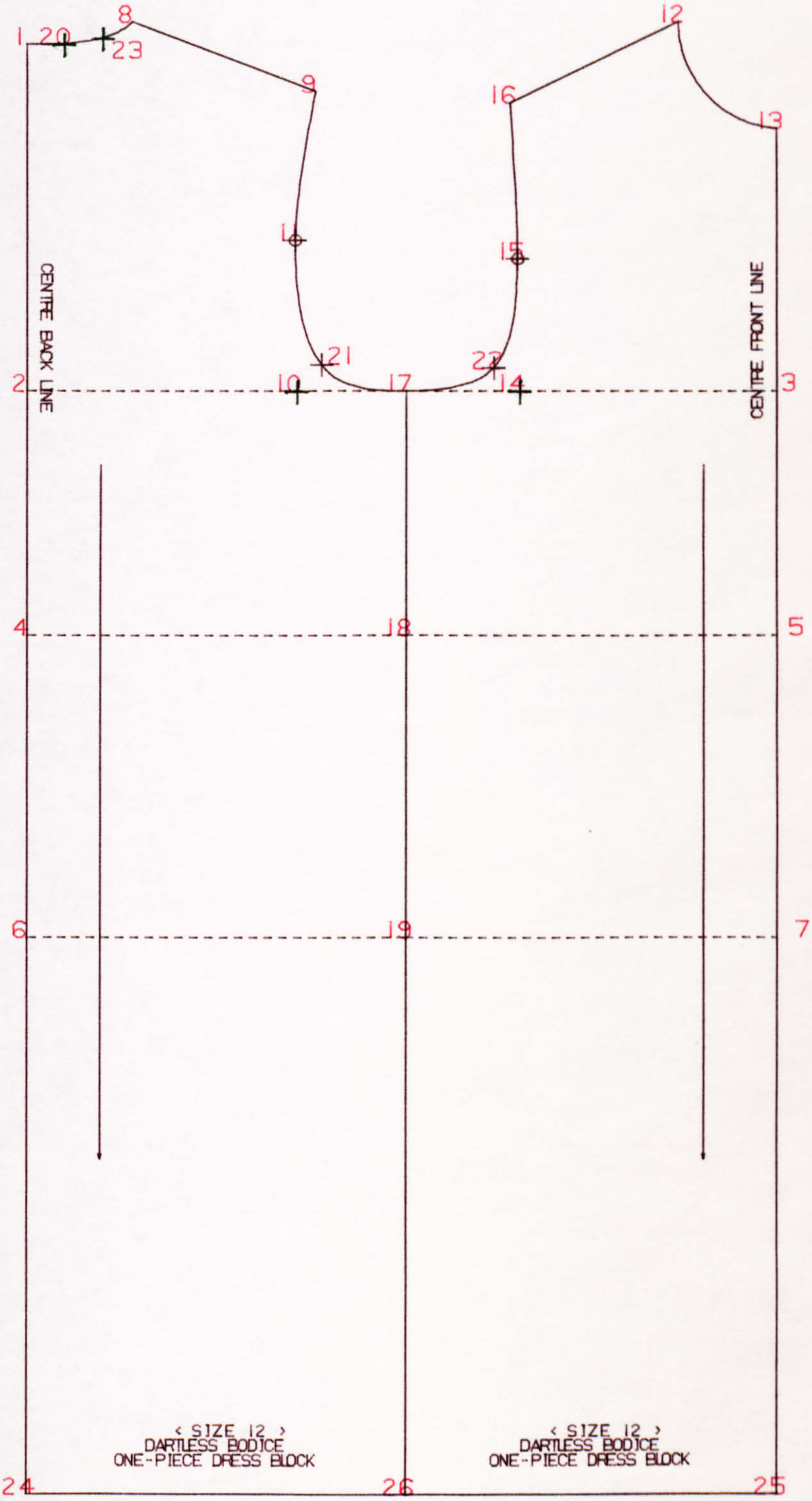


FIGURE 5-25-2: DARTLESS ONE-PIECE DRESS BLOCK WITHOUT WAIST SHAPING, SCALE: 1/5

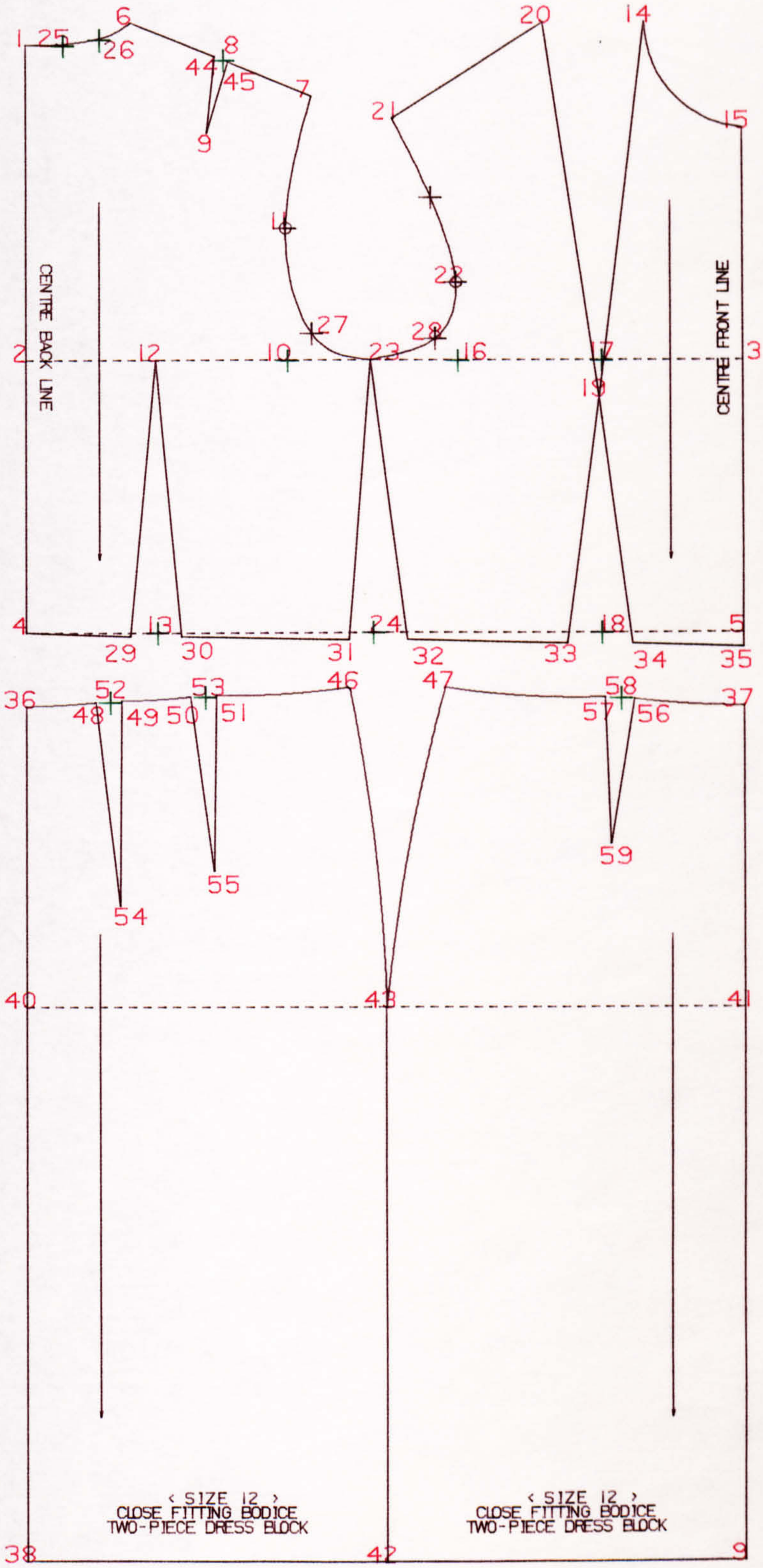


FIGURE 5-26: CLOSE FITTING TWO-PIECE DRESS BLOCK

SCALE: 1/5

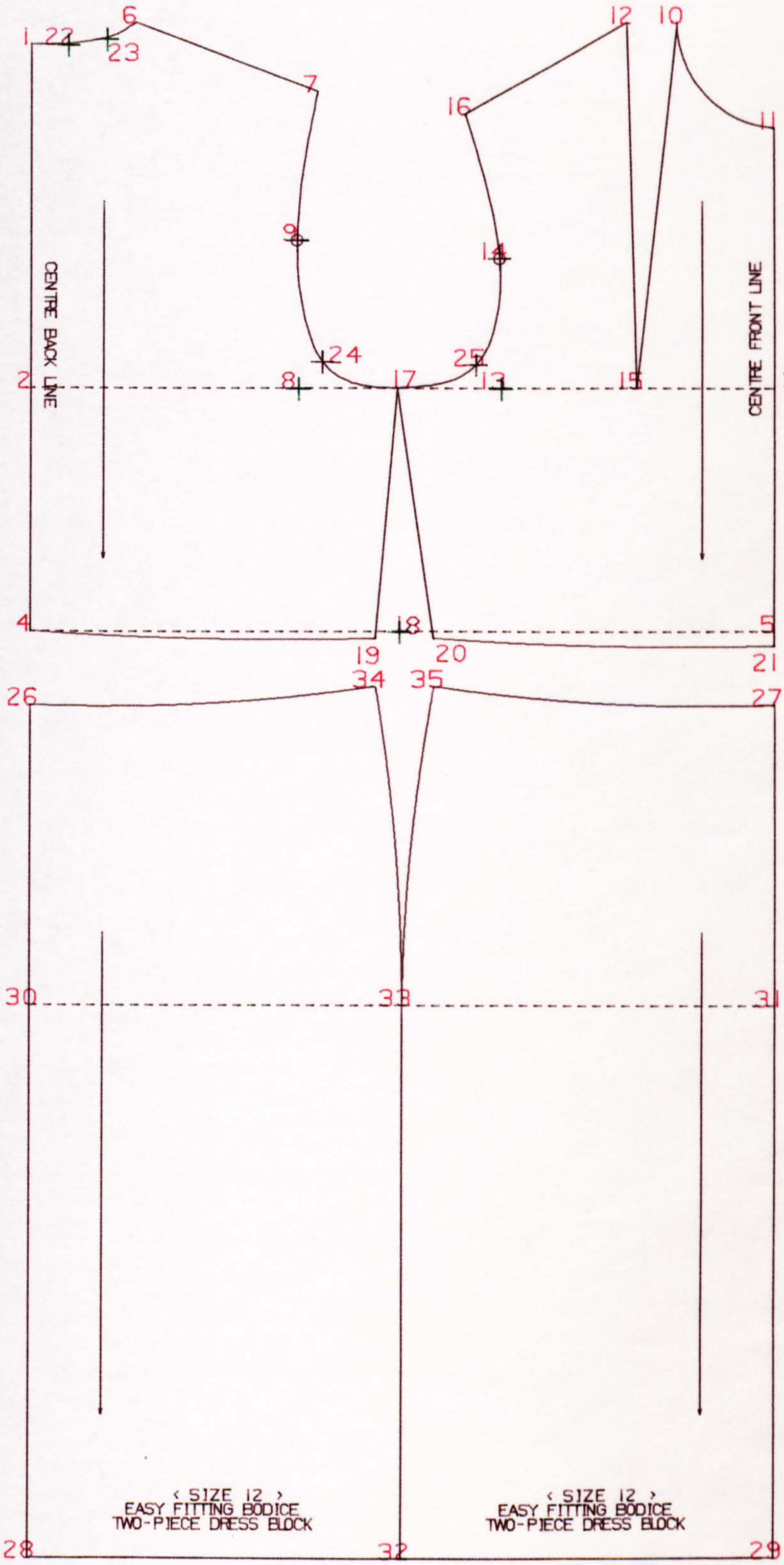


FIGURE 5-27 : EASY FITTING TWO-PIECE DRESS BLOCK

SCALE : 1/5

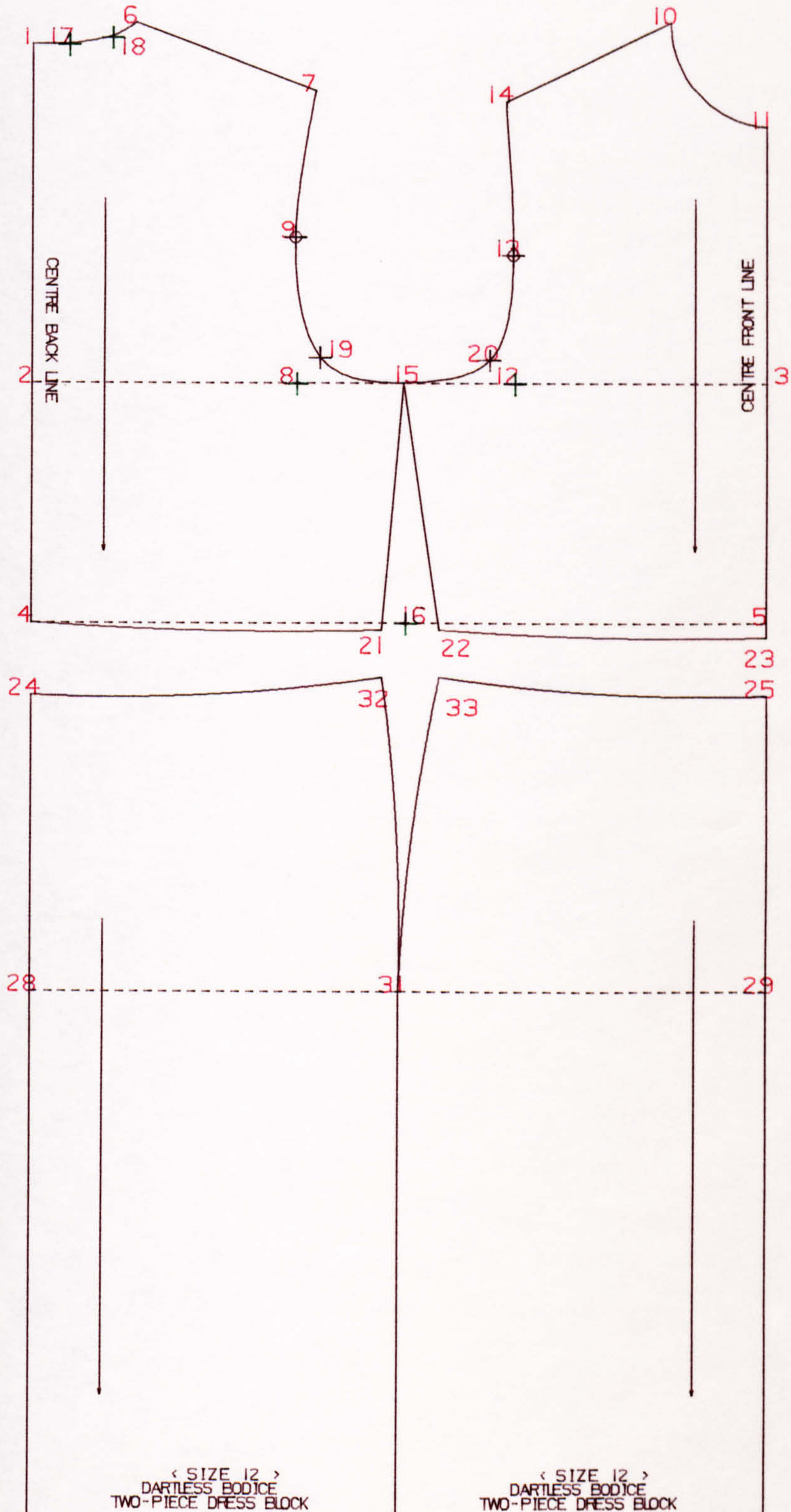


FIGURE 5-28 : DARTLESS TWO-PIECE DRESS BLOCK

SCALE : 1/5

This block is composed of the close fitting bodice block and the close fitting skirt block. It can form the basis of two-piece dresses of various designs if the dress is a close fit at the bodice and waist. See Figure 5-26.

5.2.7.2. Easy Fitting Two-Piece Dress Block

This block is composed of the easy fitting bodice block and the easy fitting skirt block. The bodice part has less dart shaping and more ease than the close fitting one, and at the waist line both bodice and skirt pattern are loose fit. This block can form the basis of any design of two-piece dress as long as the dress is an easy fit. See Figure 5-27.

5.2.7.3. Dartless Two-Piece Dress Block

This block is composed of the dartless bodice block and the easy fitting skirt block. The bodice part has as much ease as the easy fitting one without any dart shaping, and at the waist line both the bodice and skirt patterns are a loose fit. See Figure 5-28.

5.3. Additional Programs

Having created basic block patterns it was necessary to create patterns for auxiliary components for the garments, and special features such as button holes, grainline etc. The method for producing them was similar to that used for generating the block patterns. Hence, for brevity, no detailed description of their construction is given except to mention special features that were in corporated to

improve the drafting of them. The additional programs have been grouped into the following categories.

* Standing collar generation programs
* Cuffs and waistband generation programs
* Inside pocket generation programs
* Patch pocket generation programs
* Buttonhole marking programs
* Grainline and notch point marking programs

Each program required user inputted data to complete the draft the pattern. The data can be input in either centimetres or inches. Where possible default values were used for some variables, such as the collar width, so that where a default value exists, it will be used unless another value is input by the user to replace it. For example, in the program for the standing straight collar, the default value for the collar width is 5 cm (2 inches). If the user does not enter any value for the collar width, the collar width will be 5 cm. Where there is no default value, the user must input some value, if not, the value will be set to 0.

The method used for defining the points and drawing the graphic entities were similar to those used in the block pattern generation programs. The additional programs are listed in Appendix B.

5.3.1. Standing Collar Generation Programs

The standing collar is a collar which stands up around the neck or stands, rolls over then falls. In the PMS, there are four different styles standing collar generation programs. They are:-

* The Standing Straight Collar (Figure 5-29)
* The Shirt Collar (Figure 5-30)
* The Convertible Collar (Figure 5-31)
* The Polo Collar (Figure 5-32)

All these four programs require the half back and front neck measurements to be measured on the bodice pattern, to which the collar is intended to fit. Therefore, the collar pattern should be generated after the bodice pattern has been adapted. The following data is required from the user to generate the collar patterns.

* The Standing Straight Collar

  The half front and back neck measurement

  The standing collar width (default: 5 CM, 2 IN)

  The buttonstand width (default: 1.5 CM, 0.6 IN)

* The Shirt Collar

  The half front and back neck measurement

  The collar width (default: 5 CM, 2 IN)

  The collar stand depth (default: 3 CM, 1.2 IN)

  The buttonstand width (default: 1.5 CM, 0.6 IN)

* The Convertible Collar

  The half front and back neck measurement

  The collar width (default: 9 CM, 3.5 IN)

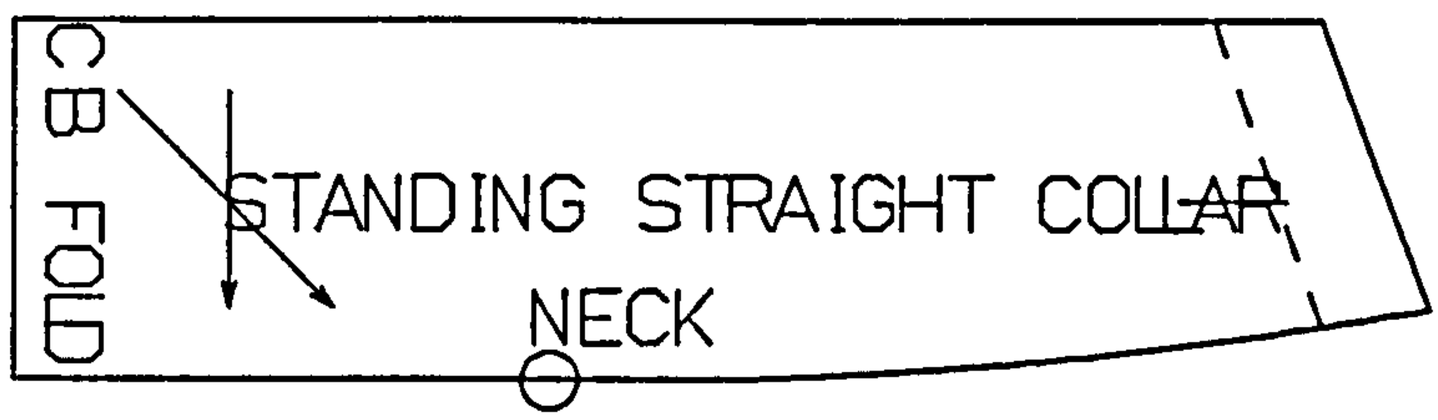
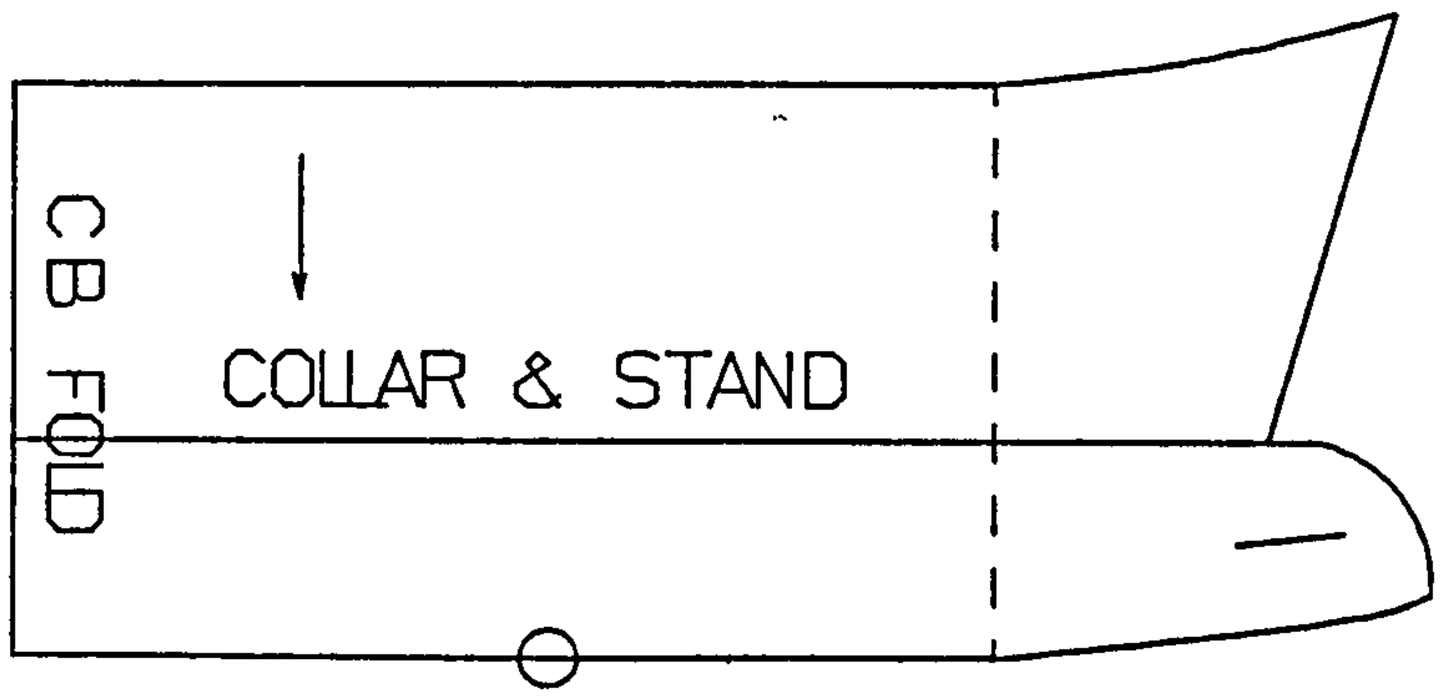
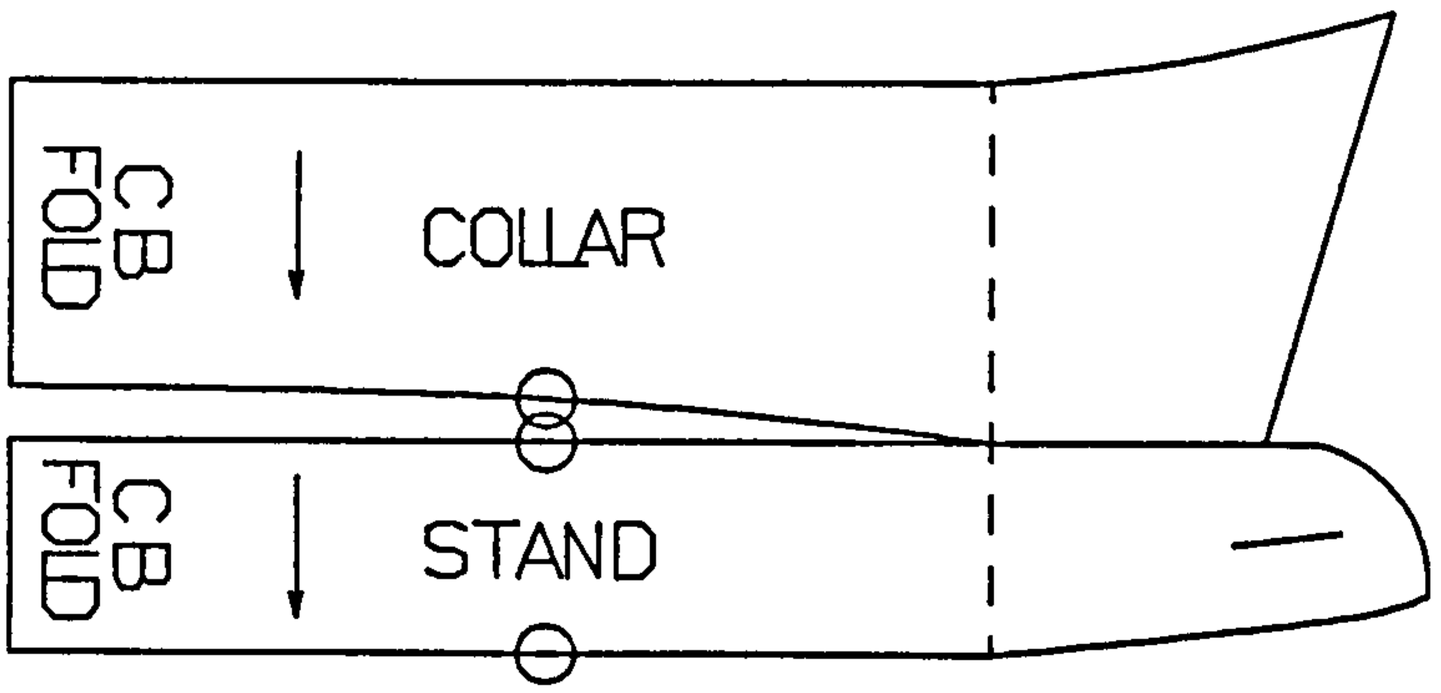


FIGURE 5-29: STANDING STRAIGHT COLLAR
SCALE: 1/2



COLLAR AND STAND IN ONE PIECE



COLLAR WITH SEPARATE STAND

FIGURE 5-30: SHIRT COLLAR
SCALE: 1/2

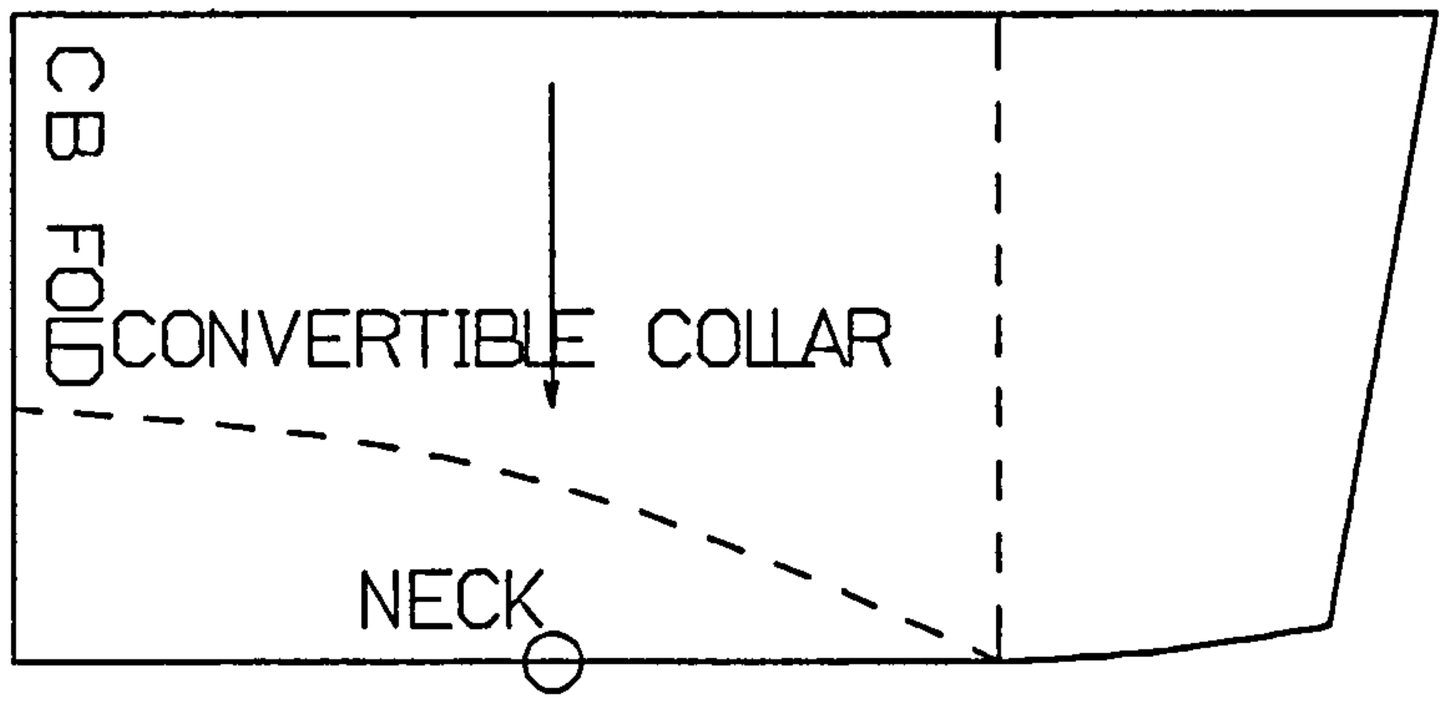


FIGURE 5-31: CONVERTIBLE COLLAR
SCALE: 1/2

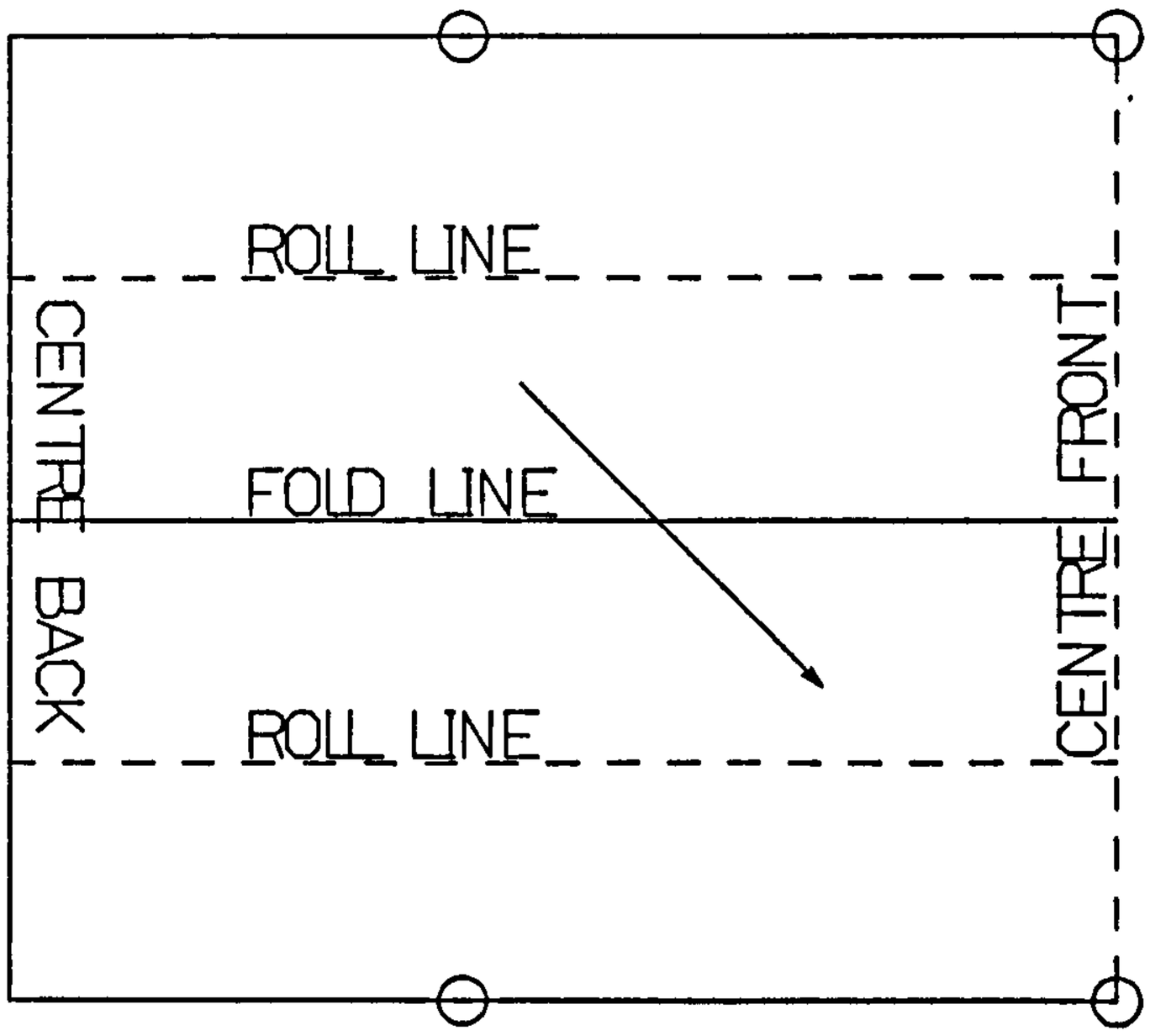


FIGURE 5-32: POLO COLLAR
SCALE: 1/2

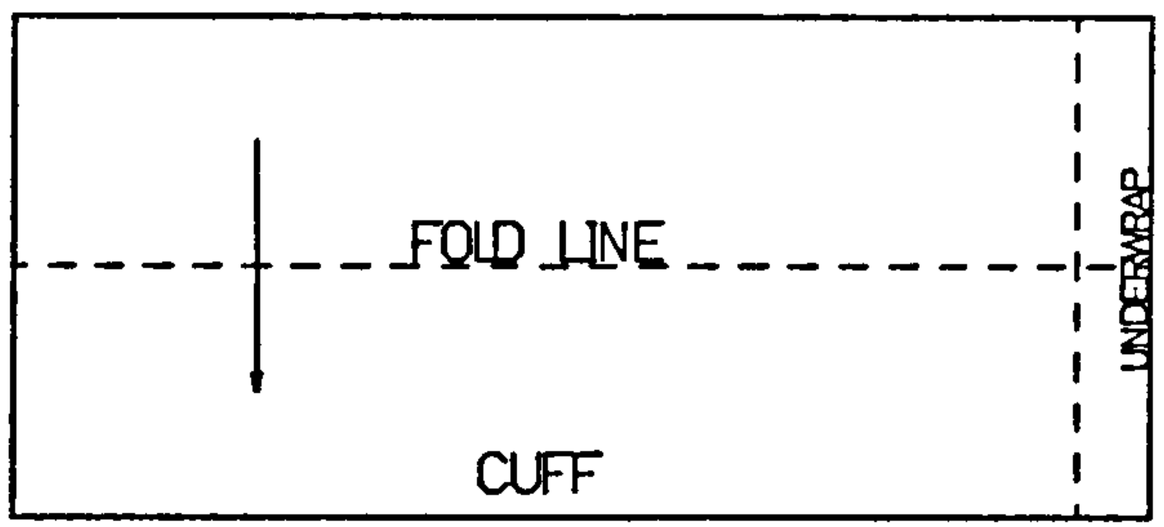


FIGURE 5-33: SHIRT CUFF
SCALE: 1/3

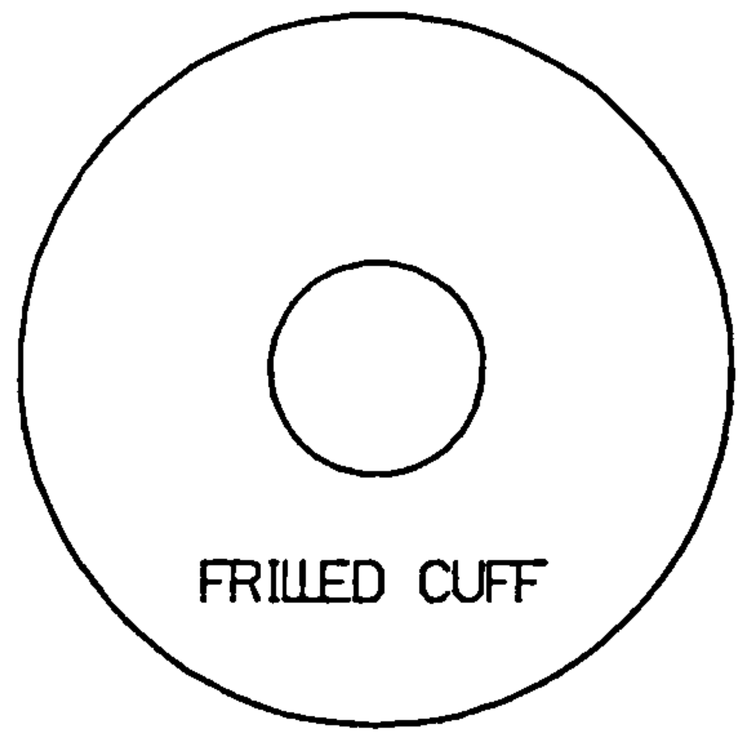


FIGURE 5-34: FRILLED CUFF
SCALE: 1/3

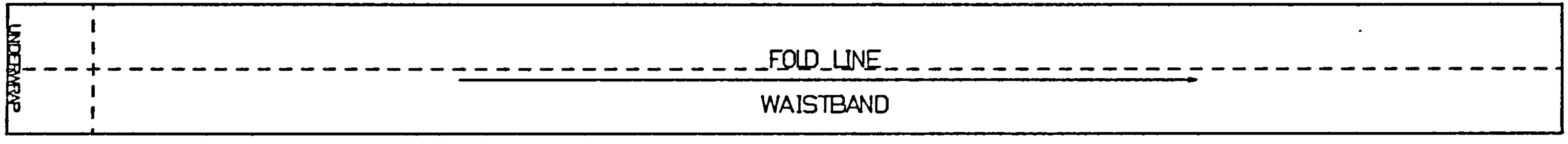


FIGURE 5-35: STRAIGHT WAISTBAND
SCALE: 1/3

* The Polo Collar

The half front and back neck measurement

The collar width (default: 4 CM, 1.6 IN)

5.3.2. Cuffs and Waistband Generation Programs

The required data for each program is as follows.

* The Shirt Cuff (Figure 5-33)

The cuff size

The cuffs depth (default: 5 CM, 2 IN)

The underwrap length (default: 1.5 CM, 0.6 IN)

* The Frilled Cuff (Figure 5-34)

The cuff size

The cuffs depth (default: 5 CM, 2 IN)

* The Waistband (Figure 5-35)

The waist length

The waistband depth (default: 3 CM, 1.2 IN)

The underwrap length (default: 4 CM, 1.7. IN)

5.3.3. Inside Pocket Generation Programs

In the PMS, there are three programs to generate three different types of inside pocket patterns, the flap pocket (Figure 5-36), the welt pocket (Figure 5-37) and the slashed pocket (Figure 5-38). The pocket place will be marked where the pocket will be made, and the patterns for both inside and outside of the pocket will be generated separately. The required data for each program is as follows.

* The Flap Pocket

    The flap depth

    Two locations for two upper ends of the flap pocket

    One location for the bottom line of the flap pocket

* The Welt Pocket

    The welt depth

    Two locations for two upper ends of the flap pocket

    One location for the bottom line of the flap pocket

* The Slashed Pocket

    The depth of one lip of the opening

    Two locations for two upper ends of the flap pocket

    One location for the bottom line of the flap pocket

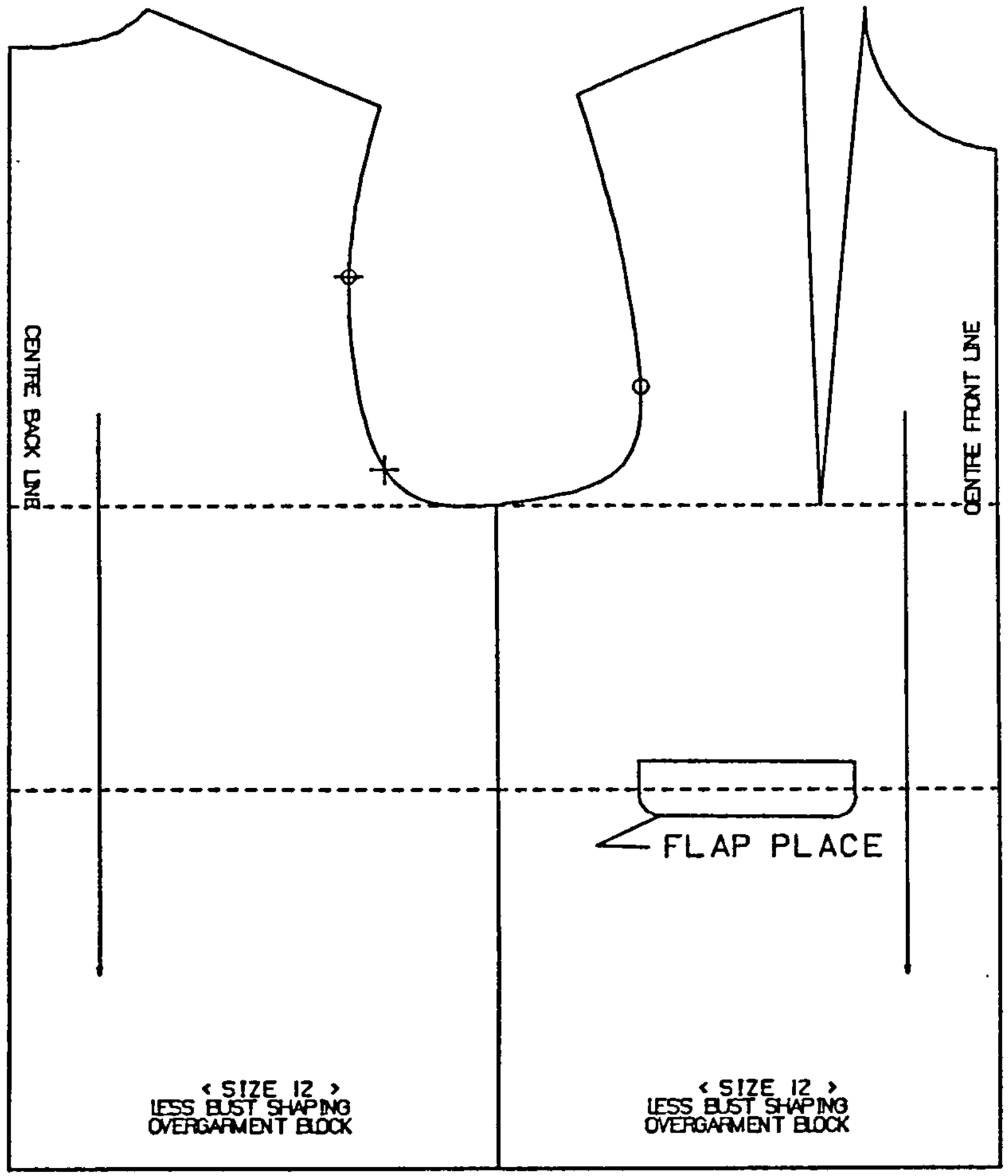
5.3.4. Patch Pocket Generation Programs

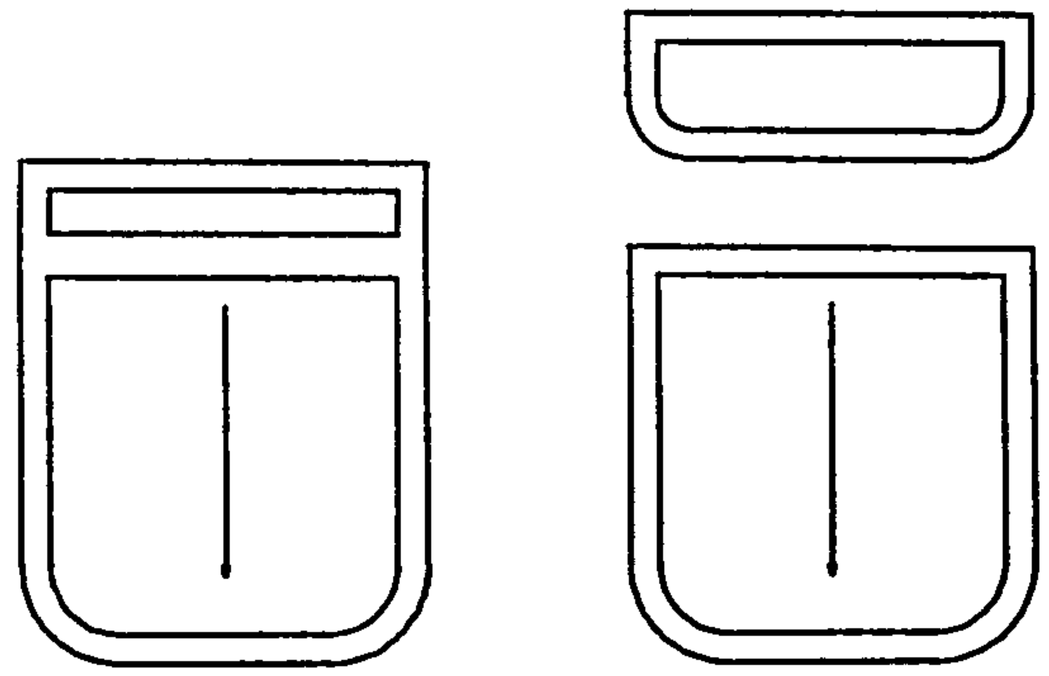
A patch pocket can be generated in four different shapes. They are:-

* The square patch pocket
* The chop-corner square patch pocket
* The round corner square patch pocket
* The square-V patch pocket

Also each of them can be generated as follows.

* Without flap
* With self-flap
* With separate flap





POCKET PATTERNS

FIGURE 5-36: FLAP POCKET
SCALE: 1/5

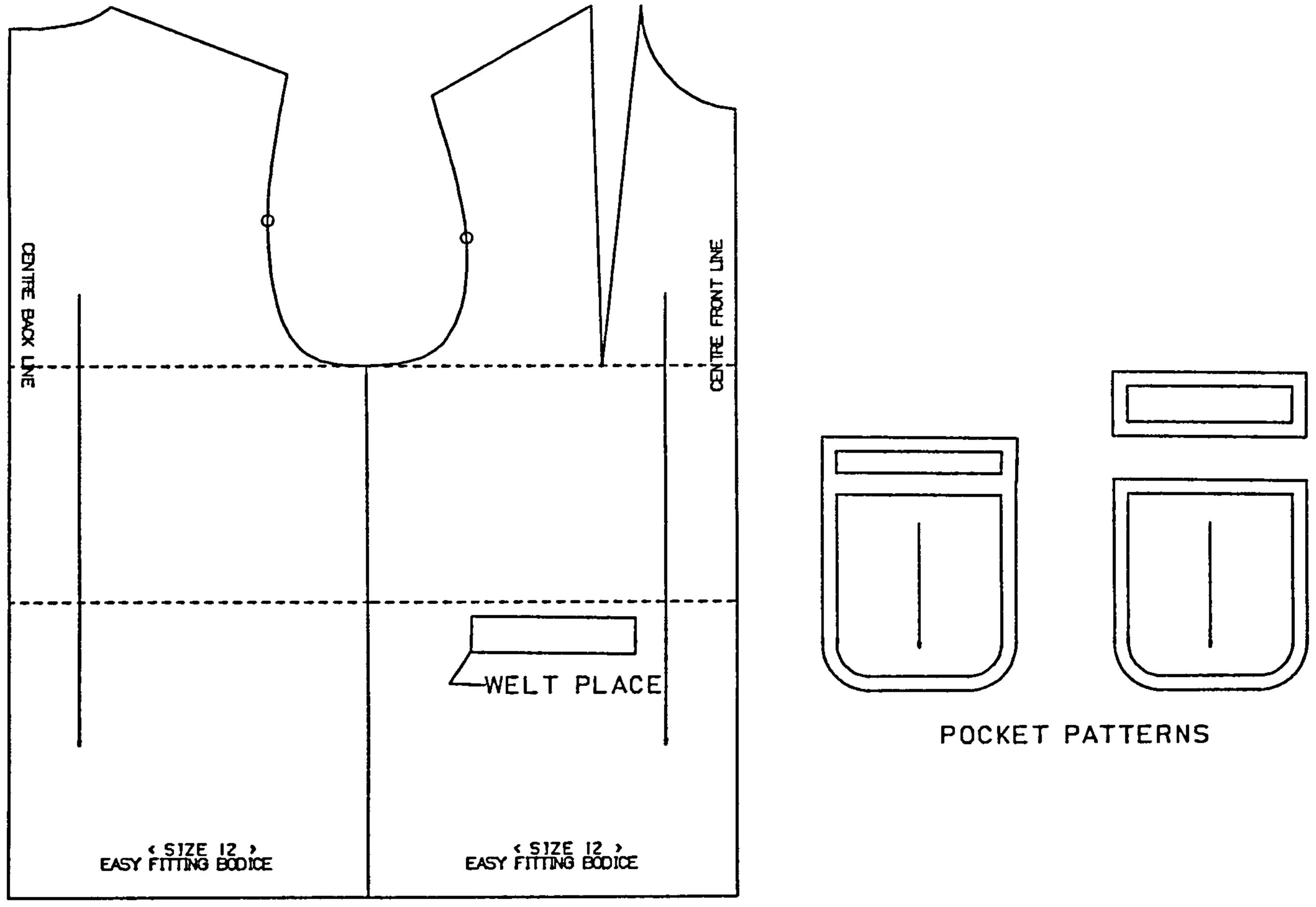
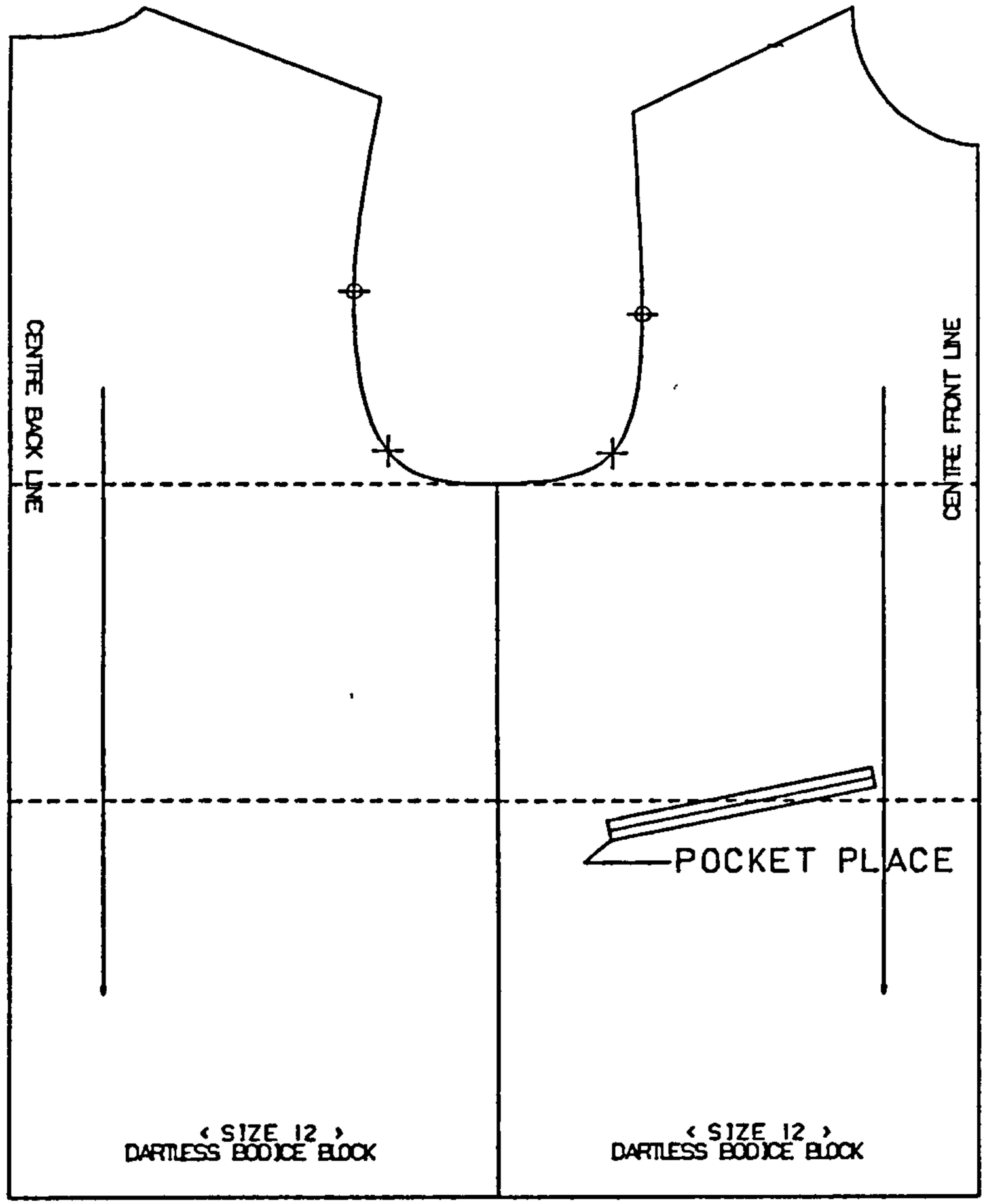
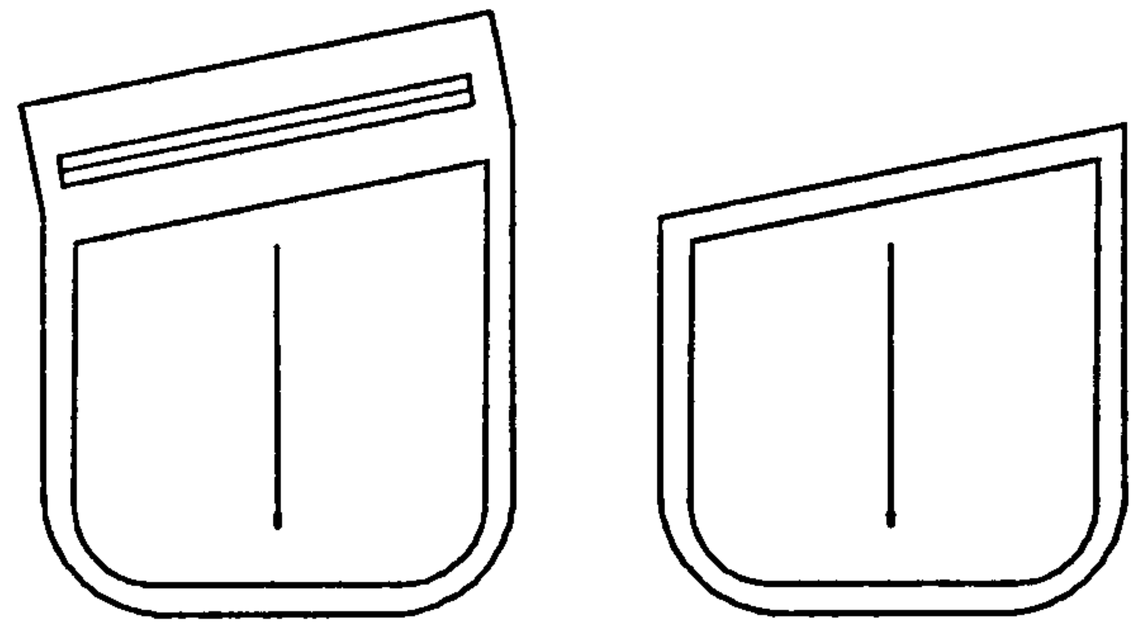


FIGURE 5-37: WELT POCKET
SCALE: 1/5





POCKET PATTERNS

FIGURE 5-38 : SLASHED POCKET
SCALE : 1/5

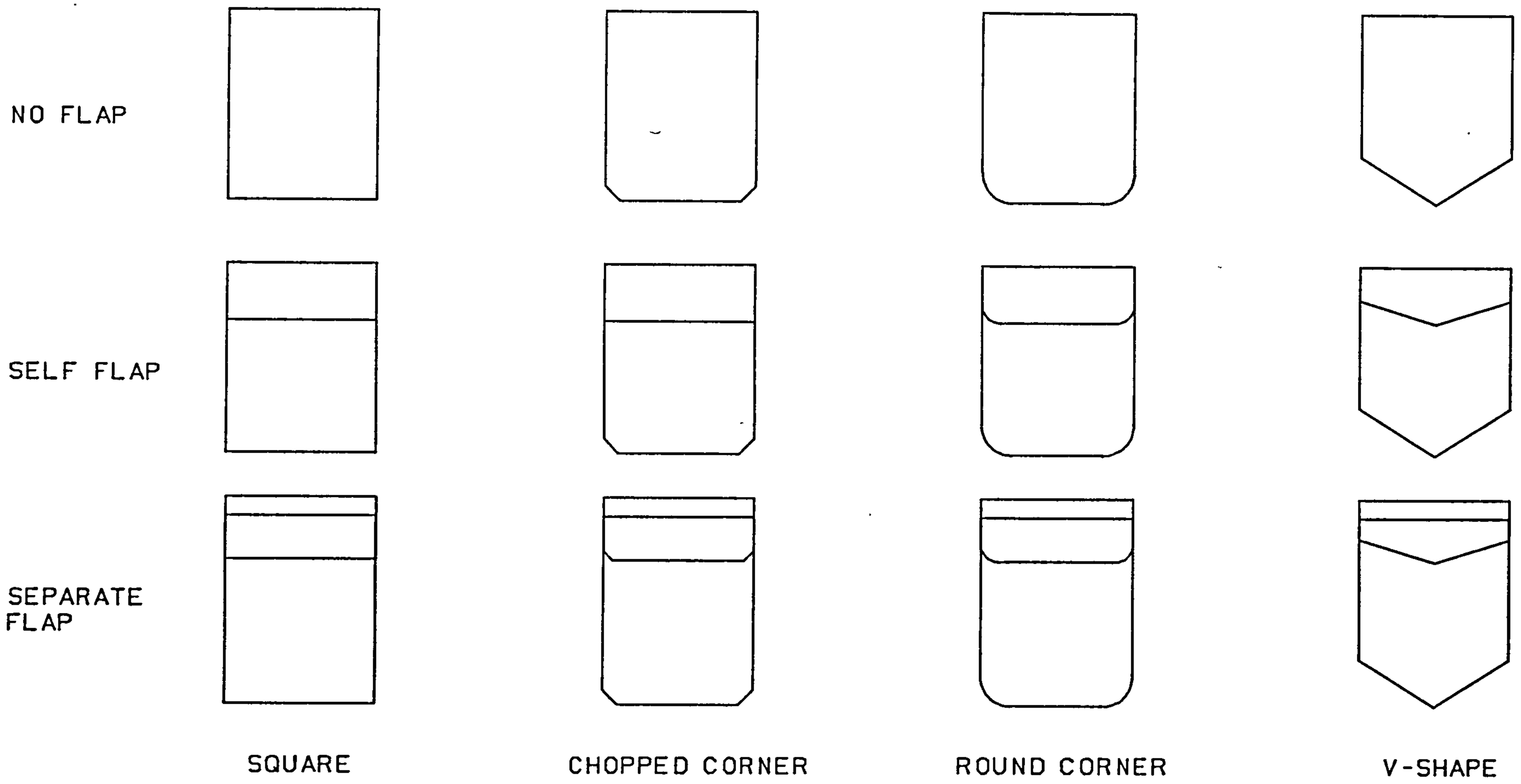


FIGURE 5-39-1: SHAPES OF PATCH POCKETS

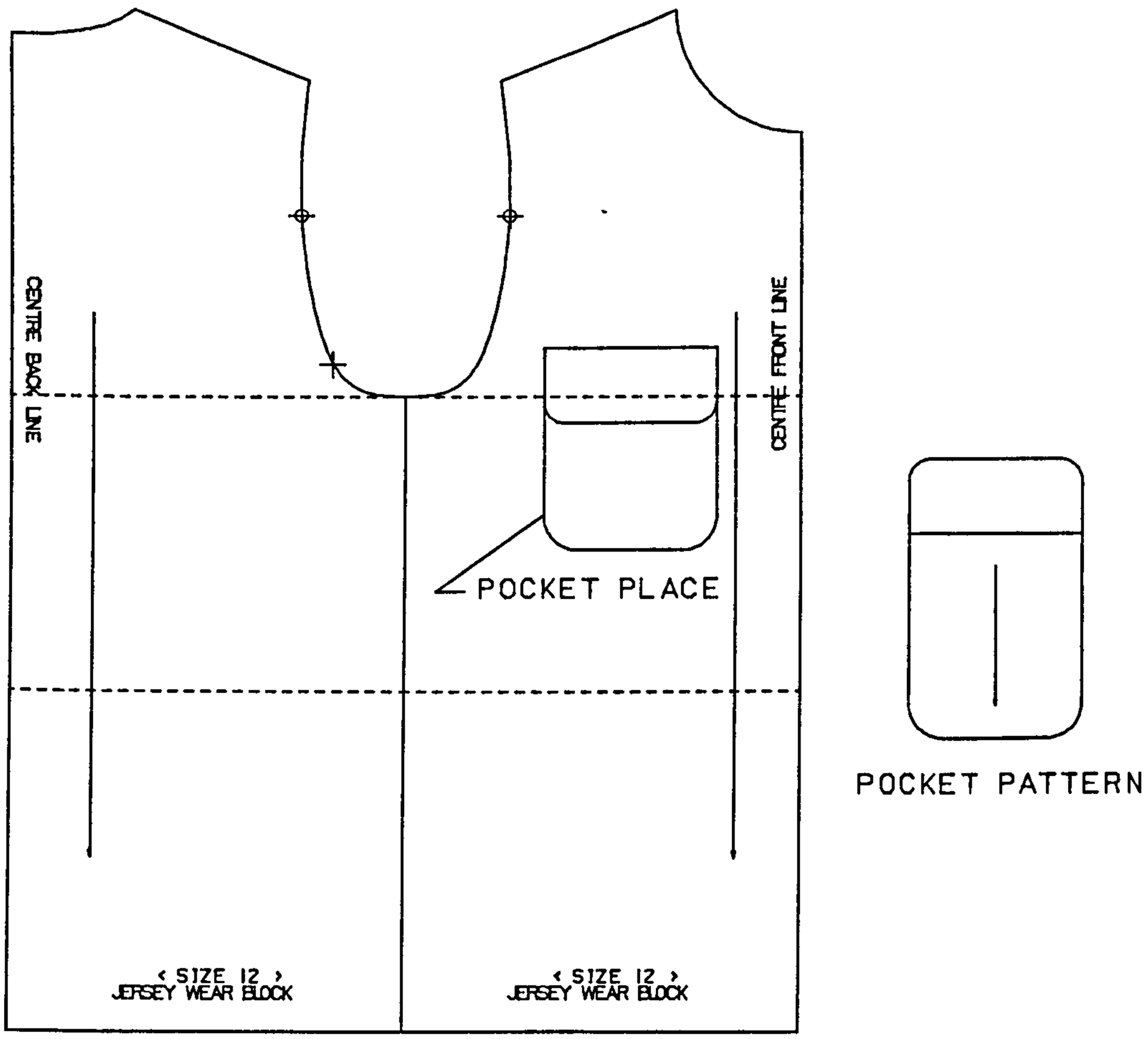


FIGURE 5-39-2: AN EXAMPLE OF THE PATCH POCKET
SCALE: 1/5

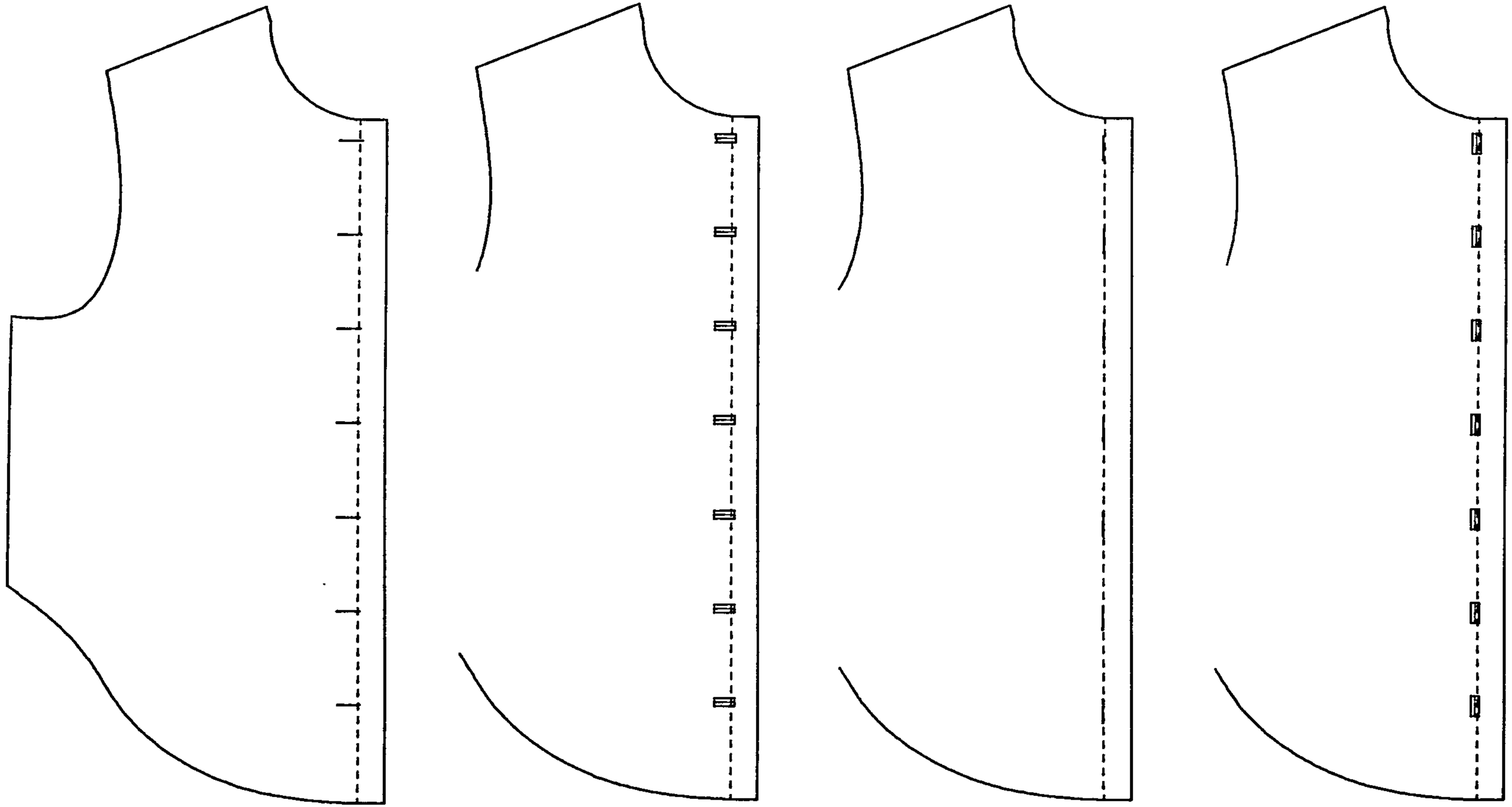


FIGURE 5-40: BUTTONHOLES
SCALE: 1/5

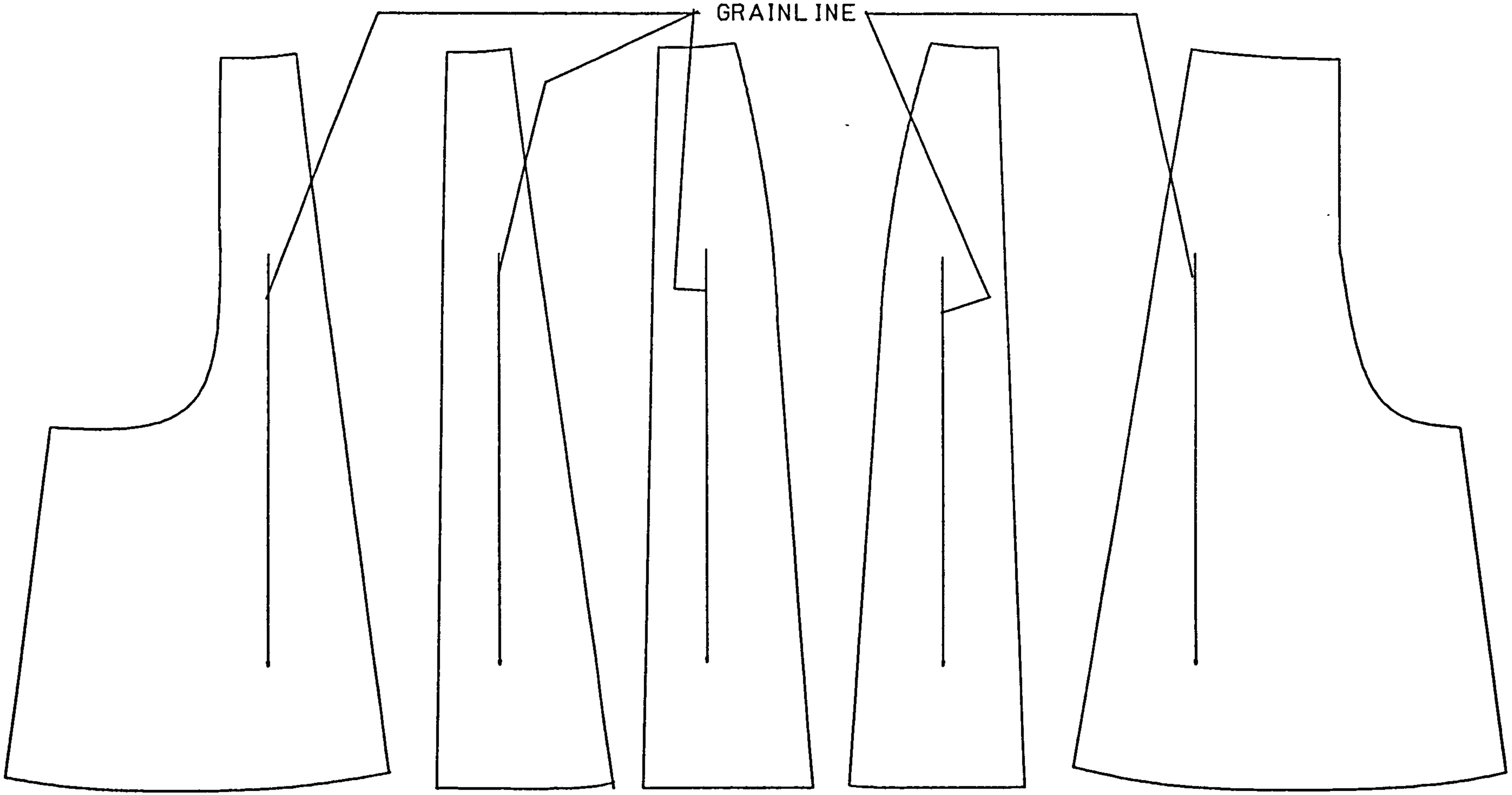
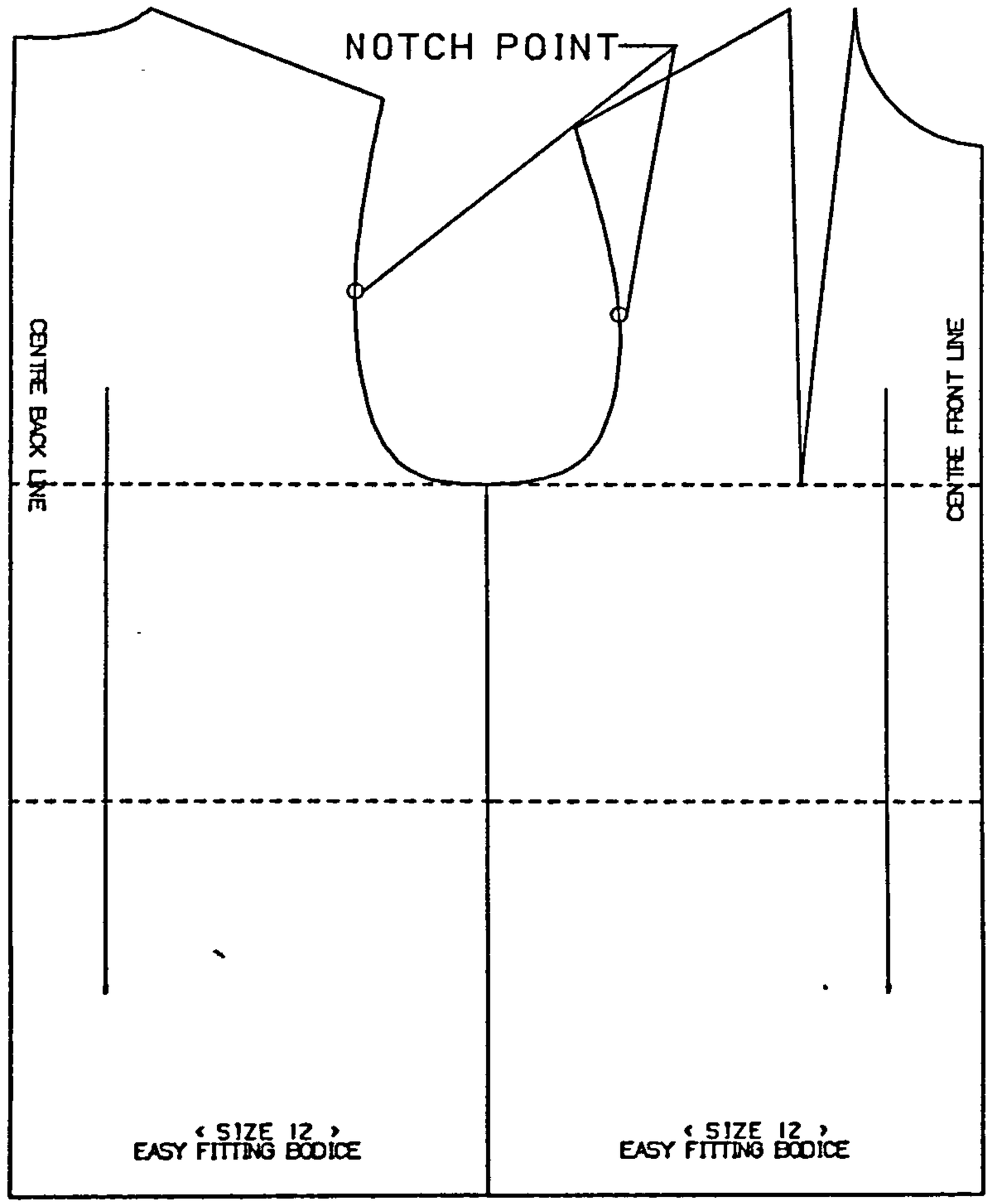


FIGURE 5-41: GRAINLINE
SCALE: 1/5



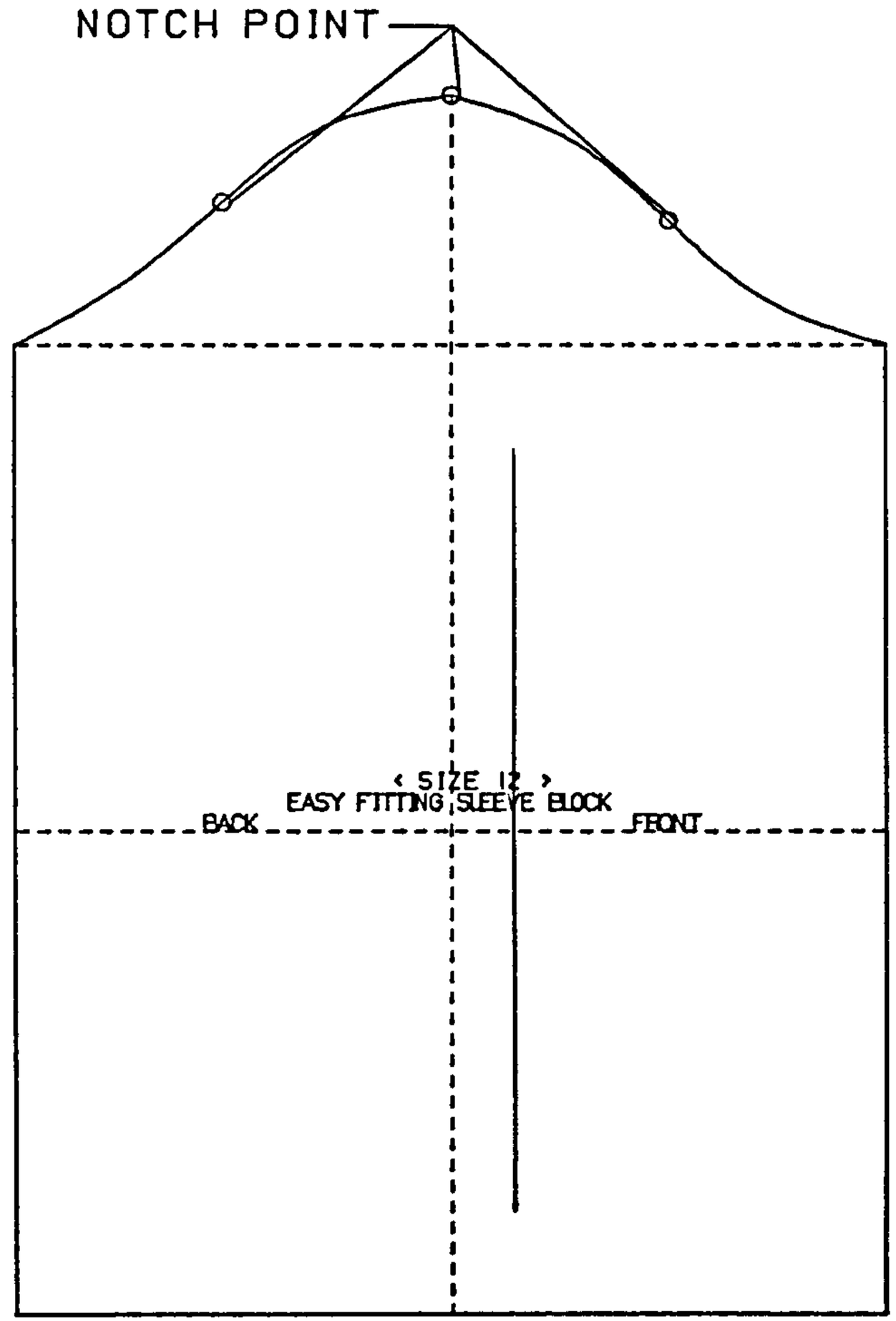


FIGURE 5-42: NOTCH POINTS
SCALE: 1/5

See Figures 5-39-1 and 5-39-2. Like the inside pocket generation programs, the pocket will be marked at the pocket place and the pocket pattern will be generated separately. The pocket size and the pocket location can be defined either by measurements or by a location input using the mouse.

5.3.5. Buttonhole Marking Programs

There are four buttonhole marking programs. They are for:-

* Horizontal worked buttonholes
* Horizontal bound buttonholes
* Vertical worked buttonholes
* Vertical bound buttonholes

The Buttonholes can be marked by the following user input.

* Number of buttonholes
* Diameter of the button
* Thickness of the button
* Two locations for the first and the last buttonholes.

See Figure 5-40. This program assumes even buttonhole placing. Presumably, if the spacing is uneven the user must mark each buttonhole separately.

5.3.6. Grainline and Notch Point Marking Programs

The user can mark the grain lines (Figure 5-41) and the notch points (Figure 5-42). The grain line can be inserted in any of following ways.

* Vertical grainline

* Horizontal grainline

* Bias grainline (an angle of 45 degrees)

* Bias grainline (an angle of 135 degrees)

* Any two locations

The notch point which is generated by the program is a circle of radius 0.4 cm, whose centre is at the location selected by the user.

5.3.7. Open File

Normally, within the CADDS system, the user must perform several selection procedures to create a graphics environment on the computer to use the PMS. For the novice user, this was considered confusing. Therefore a supplementary program entitled "Open File" was written to simplify the task of creating a graphics environment on the computer. When the user begins pattern making, he can simply open a graphic file using this program. Only the part name and the drawing name have to be entered. Commands which are automatically entered when this icon is selected are:

SELECT UNITS CM

ACTIVATE PART <part name>

SELECT TAG ON

SELECT DRAWING UNIT CM

SELECT DRAWING SIZE E

ACTIVATE DRAWING <draw name>

ECHO FRAME

```
ECHO TAG ON
DEFINE VIEW TOP CPLANE TOP SCALE 0.5 : X22Y17
```

SELECT UNITS CM sets the part unit to centimetres, and ACTIVATE PART <part name> creates a part.

SELECT TAG ON allows entities in the drawing to be referenced using a written tag, which is the name of the entity. SELECT DRAWING UNIT CM sets the drawing units to centimetres. SELECT DRAWING SIZE E sets the drawing size 86.36 x 111.76 CM (34 x 44 IN), and ACTIVATE DRAWING <draw name> creates a drawing of size 86.36 x 111.76 CM.

ECHO FRAME displays the frame of the drawing and a view, and ECHO TAG ON will display the tag if there is any.

DEFINE VIEW TOP CPLANE TOP SCALE 0.5 : X22Y17 defines the view TOP, whose origin (X0Y0) is at the middle of the screen and all the entities to be seen through this view will be displayed in half scale. Therefore entities can actually be drawn to a size of 172.72 x 223.52 CM.

5.4. Grading Programs

Pattern Grading is necessary to reproduce an adapted pattern for other sizes. The conventional grading method is to draft a basic size, assign grade points to the pattern and displace grade points a fixed amount according to a set of grade rules, which has to be defined by the grader. The grade rules are derived from anthropometric data in a size chart. A grade rule is a distance in both x and y direction by which a grading point has to move to reproduce other sizes [36] [39] [40].

There are many degrees of simplification of the grading system, and many slight variations in grading rules [38] [39] [40]. Therefore, the subject of grading is very complex, with no apparent standardization, where choice of method and grade rules is left to personal discretion [40]. Hence, to determine the correct grading rule for an adapted pattern (style grading) is not simple, and could cause differences between graders depending on their working knowledge of pattern designing.

For this reason, and to simplify the grading task and to enable data to be read directly from the size chart, a proportional grading system was developed. It is recognised that this has certain disadvantages. One of the disadvantages is that the user cannot control the grade rule for each grading point, because the grade rules are automatically defined by the grading ratio according the each grading program. So the very close fitting garment is not recommended to be graded by the proportional grading system, because the grader cannot control the pattern to keep the fit.

Using the CVMAC language within CADDS software environment, it proved difficult to develop an alternative method, without considerable effort. This would have detracted from the overall objective of the PMS so it was decided to develop a simple grading system with a view to improvements being made in future work. Therefore for easy fitting garments where fit is less important, the

proportional grading system was developed i.e. a patterns are graded in width and length direction in proportion to one key dimension on the pattern. The advantage of this is that the grading programs of the PMS did not require grade rules for every grading point, because pattern pieces will be proportionally increased in the x and (or) y direction by a certain ratio which was defined as the grade ratio.

Another advantage of proportional grading is that adapted pattern pieces are graded keeping the consistent proportion of the pattern shape. In conventional grading, when the patterns are adapted i.e. split, or given extra ease, the grader has to decide the grade rule for each pattern pieces considering the effect of the adaptation. Using proportional grading the pattern is graded by the "ratio" not by the x and y direction increment, so the adapted patterns are graded as simply as the block pattern, because the split gap or the extra ease is graded as well as the pattern pieces.

5.4.1. The Grading Programs of the PMS

Since the grade ratio would differ according to the patterns being graded e.g. bodice, skirt etc., separate grading programs were developed for each type of pattern. The five programs are:

* Bodice pattern grading program (Grade - Bodice)
* Skirt pattern grading program (Grade - Skirt)
* Trouser pattern grading program (Grade - Trouser)
* Sleeve pattern grading program (Grade - Sleeve)
* Collar pattern grading program (Grade - Collar)

In addition, the following three programs were produced to grade only in one direction according to either height or data entered by the user. The latter was written for use in grading cuffs and waistband, for example. These programs are:

* Program for grading pattern to different heights Short, Medium and Tall (Grade - Height)
* Program for grading pattern in the horizontal direction only (Grade - Horizon)
* Program for grading pattern in the vertical direction only (Grade - Vertical)

5.4.2. The Grading Ratio

The grading ratio for the proportional increase or decrease in x and y directions depend on the program. In the bodice pattern grading, the x direction ratio chosen was the bust measurement for each size to be graded relative to the bust measurement for the current size. The y direction ratio chosen was the nape to waist measurement for each size to be graded relative to the nape to waist measurement for the current size. The key dimensions chosen to define the x and y direction grading ratio for each grading program were as follows.

Program	x direction	y direction
Grade - Bodice	Bust	Nape to Waist
Grade - Skirt	Hip	Waist to Knee
Grade - Trouser	Hip	Waist to Floor
Grade - Sleeve	Armhole	Sleeve Length
Grade - Collar	Neck	Nape to Waist
Grade - Height	-	*
Grade - Horizon	**	-
Grade - Vertical	-	***

* One of the following measurements of each height of the current size. The key measurements depended on the pattern as follows.

Pattern	Key Measurement
Bodice	Nape to Waist
Skirt	Waist to Knee
Trouser	Waist to Floor
Sleeve	Sleeve Length

For example, if the trouser of size 12 height M is graded into height S and T, the y direction grading ratio will be defined by the Waist to Floor measurement for each height of the size 12.

** The x direction ratio which the user has to define.

*** The y direction ratio which the user has to define.

Therefore the grading ratio of the x and y direction was expressed as follows.

RXn = KeyXn / KeyXc

RYn = KeyYn / KeyYc

RXn : x direction grading ratio for the size n

RYn : y direction grading ratio for the size n

KeyXn: x direction key measurement for the size n

KeyYn: y direction key measurement for the size n

KeyXc: x direction key measurement for the current size

KeyYc: y direction key measurement for the current size

For example, if a skirt pattern of size 12 is graded into sizes 8, 10 and 14, the grading ratio will be as follows.

Rx8 = 85/93 Rx10 = 89/93 Rx14 = 97/93

Ry8 = 57.5/58.5 Ry10 = 58/58.5 Ry14 = 59/58.5

These are because the x direction key measurement for the skirt grading program is the hip measurement (size 8: 85cm, size 10: 89cm, size 12: 93cm, size 14: 97cm), and the y direction key measurement for the skirt grading program is the waist to knee measurement (size 8: 57.5cm, size 10: 58cm, size 12: 58.5cm, size 14: 59cm). The key measurements for each grading program were stored in the program, so that the program could define the grading ratio by the size codes, which are input by the user. Therefore the user only has to input the current size (basic size) and the sizes to be graded in terms of the size code (a size code should be an even number between 8 and 30).

Table 4: Comparision of Graded Dimensions to Standard Body Measurements (cm)

		SIZE CODE											
SIZE SYMBOL		8	10	12	14	16	18	20	22	24	26	28	30
BUST		80	84	88	92	97	102	107	112	117	122	127	132
HIPS		85	89	93	97	102	107	112	117	122	127	132	137
NAPE TO WAIST		39	39.5	40	40.5	41	41.5	42	42.5	43	43.2	43.4	43.6
WAIST TO KNEE		57.5	58	58.5	59	59.5	60	60.5	61	61.25	61.5	61.75	62
WAIST TO FLOOR		102	103	104	105	106	107	108	109	109.5	110	110.5	111
SLEEVE LENGTH		57.2	57.8	58.4	59	59.5	60	60.5	61	61.2	61.4	61.6	61.8
WAIST	1	60	64	68	72	77	82	89	92	97	102	107	112
	2	62.1	65.1		71	79	83		91	99	103		111
	3	2.1	1.1		-1	2	1		-1	2	1		-1
BACK WIDTH	1	32.4	33.4	34.4	35.4	36.6	37.8	39	40.2	41.4	42.6	43.8	45
	2	31.3	32.8		36.0	35.4	37.2		40.8	40.4	42.1		45.5
	3	-1.1	-0.6		0.6	-1.2	-0.6		0.6	-1	-0.5		0.5
CHEST	1	30	31.2	32.4	33.6	35	36.5	38	39.5	41	42.5	44	45.5
	2	29.5	30.9		33.9	34.4	36.2		39.8	40.5	42.3		45.7
	3	-0.5	-0.3		0.3	-0.6	-0.3		0.3	-0.5	-0.2		0.2
SHOULDER LENGTH	1	11.75	12	12.25	12.5	12.8	13.1	13.4	13.7	14	14.3	14.6	14.9
	2	11.14	11.69		12.81	12.15	12.8		14.0	13.45	14.03		15.17
	3	-0.61	-0.31		0.31	-0.65	-0.3		0.3	-0.55	-0.27		0.27
NECK SIZE	1	35	26	37	38	39.2	40.4	41.6	42.8	44	45.2	46.4	47.6
	2	33.6	35.3		38.7	37.7	39.7		43.5	42.7	44.6		48.2
	3	-1.4	-0.7		0.7	-1.5	-0.7		0.7	-1.3	-0.6		0.6
DART	1	5.8	6.4	7	7.6	8.2	8.8	9.4	10	10.6	11.2	11.8	12.4
	2	6.4	6.7		7.3	8.5	9.0		9.8	10.9	11.3		12.3
	3	0.6	0.3		-0.3	0.3	0.2		-0.2	0.3	0.1		-0.1
ARMHOLE DEPTH	1	20	20.5	21	21.5	22	22.5	23	23.5	24.2	24.9	25.6	26.3
	2	20.5	20.7		21.3	22.5	22.7		23.3	25.4	25.5		25.7
	3	0.5	0.2		-0.2	0.5	0.2		-0.2	1.2	0.6		-0.6
WAIST TO HIP	1	20	20.3	20.6	20.9	21.2	21.5	21.8	22.1	22.3	22.5	22.7	22.9
	2	20.1	20.4		20.8	21.4	21.6		22.0	22.5	22.6		22.8
	3	0.1	0.1		-0.1	0.2	0.1		-0.1	0.2	0.1		-0.1
BODY RISE	1	26.6	27.3	28	28.7	29.4	30.1	30.8	31.5	32.5	33.5	34.5	35.5
	2	27.5	27.7		28.3	30.2	30.5		31.1	34.2	34.3		34.7
	3	0.9	0.4		-0.4	0.8	0.4		-0.4	1.7	0.8		-0.8

(1) Standard Measurement. (2) Graded Measurement. (3) Difference.

Table 4 shows the comparison of the graded dimensions to the standard body measurements. In the table, the first six dimensions do not need to be compared, because these measurements are the key dimensions which define the grading ratio of the grading programs of the PMS. They are bust, hips, nape to waist, waist to knee, waist to floor and the sleeve length measurements. The rest of the measurements are compared using the following assumptions.

1. The patterns are graded to the sizes which belong to the same size group. The sizes are divided into three groups, 8 ~ 14, 16 ~ 22 and 24 ~ 30. Each size group has one basic size, they are size 12 for the size group 8 ~ 14, size 20 for the size group 16 ~ 22 and size 28 for the size group 24~30. The rest of the sizes are graded from the basic size pattern in the same size group. For example, patterns of sizes 8, 10 and 14 are graded from the size 12 pattern. Therefore, the dimensions of these basic sizes do not need to be compared in Table 4.
2. The "waist" and "waist to hip" dimensions are graded in the skirt grading program.
3. The "back width", "chest", "shoulder length", "neck size", "dart" and "armhole depth" dimensions are graded in the bodice grading program.
4. The "body rise" dimension is graded in the trouser grading program.

The difference between the two dimensions is the result of subtracting the standard measurement from the graded dimension. If the difference is a positive value, the graded

dimension is bigger by as much as the difference from the standard measurement, or if the difference is a negative value, the graded dimension is smaller by as much as the difference from the standard measurement.

There are differences in many dimensions. However, the graded dimensions are within the range of measurements for each size. For example, the graded chest measurement of size 8 is 29.5 cm, which is 0.5 cm smaller than the standard chest of the same size (30 cm), but this difference should be acceptable. Therefore, considering the simpler grading process of the PMS, these dimension difference could be tolerated unless the pattern is for very close fitting garment.

5.4.3. Necessary Graphics Data

In order to grade the pattern proportionally by the grading ratio, the user has to input the necessary graphic data which defines the pattern shape. These are the entity location and the entity type.

As in conventional grading, the proportional grading is performed by moving the end points (grading points) of each entity and drawing new lines or curves or points for the sizes to be graded. The difference in these two methods is in the way of moving the grading points. One moves the grading points according to a grading rule, and the other moves the grading points by a certain ratio in x and y directions. Therefore, the entity locations (locations of the grading points) have to be entered.

After moving the grading points, the system should know what type of entity should be drawn between the new locations. Unfortunately, the PMS grading programs do not automatically recognise the entity type from the basic size pattern, so the user has to input the entity type. The entity types which can be drawn in the PMS grading programs are a line, a B-spline (curve), a point and a notch point.

5.4.4. Zero Point

The user has to define a location for the zero point, which will be the constant point through the sizes to be graded. All other grading points (locations) will be moved according to the following rule.

$$Xn = X0 + (X - X0) * RXn$$

$$Yn = Y0 + (Y - Y0) * RYn$$

Where;

X0 : x coordinate of the zero point

Y0 : y coordinate of the zero point

X : x coordinate of a location of the current size

Y : y coordinate of a location of the current size

Xn : x coordinate of the location graded to size n

Yn : y coordinate of the location graded to size n

RXn: The grading ratio of x direction for the size n

RYn: The grading ratio of y direction for the size n

For example, if a trouser pattern is adapted as shown in Figure 5-43, the grading points will be moved as follows.

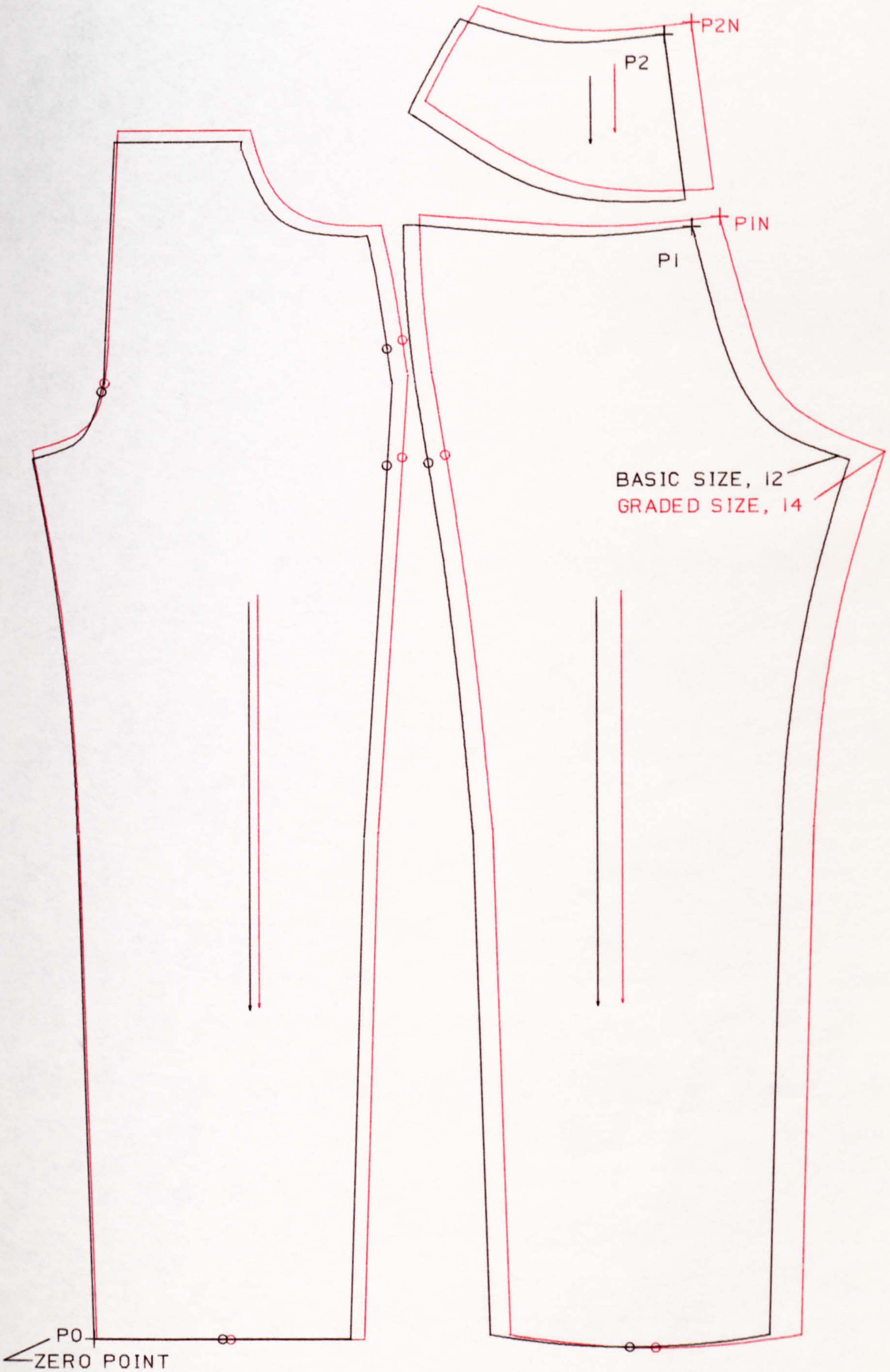


FIGURE 5-43: GRADING POINTS
SCALE: 1/5

$$X1n = X + (X1 - X0) * RXn$$

$$Y1n = Y + (Y1 - Y0) * RYn$$

$$X2n = X + (X2 - X0) * RXn$$

$$Y2n = Y + (Y2 - Y0) * RYn$$

Therefore,

$$X2n - X1n = (X0 + (X2 - X0) * RXn) - (X0 + (X1 - X0) * RXn)$$
$$= (X2 - X0) * RXn - (X1 - X0) * RXn$$
$$= (X2 - X1) * RXn$$
$$Y2n - Y1n = (Y0 + (Y2 - Y0) * RYn) - (Y0 + (Y1 - Y0) * RYn)$$
$$= (Y2 - Y0) * RYn - (Y1 - Y0) * RYn$$
$$= (Y2 - Y1) * RYn$$

In the end, the pattern pieces can be enlarged or diminished by the grade ratio in both the x and y direction, even though the patterns are split. In manual grading, the pattern grader has to decide the grade increments considering the proportion of the split piece, and mistakes can easily be made.

5.4.5. Special Features

In order to define the grading ratio of the sleeve grading programme, the user has to input the armhole measurement of sizes to be graded and the current size. The armhole measurement should be measured from the graded bodice patterns, to which the sleeve will fit. So the sleeve pattern should be graded after the bodice pattern is graded to the appropriate sizes. This applies also to the collar grading program, the only difference being that the collar

grading program requires the neck measurements instead of the armhole measurements. In both cases, the user is given the opportunity to measure the required measurements from the bodice pattern and input them.

Figures 5-44 ~ 5-49 are the examples of pattern grading using the PMS grading programs. For more detailed instructions about grading, see Chapter 9 (Advanced Tutorials, Sessions 2 and 3).

5.4.6. Horizontal and Vertical Grading Program

The horizontal grading program grades a pattern in a horizontal direction by a ratio which is defined by the user. This is in case the user may wishes to grade a pattern in a horizontal direction only, such as for cuffs, a waistband or a belt.

In this case the y direction grading ratio is always set to 1. The x direction grading ratio of each size to be graded is the ratio of a key measurement of each size relative to the key measurement of the current size. The user therefore has to decide the key measurement and input the measurement for each size. For example, if a waistband of size 12 is to be graded into the sizes 10 and 14, the key measurement can be the waist measurement, so the waist measurement of size 12, 10 and 14 has to be entered to define the grading ratio. See Figure 5-50.

The vertical grading program is exactly the same as the Horizontal Grading Program except that the grading

direction, and the grading ratios are defined in the vertical direction instead of the horizontal direction.

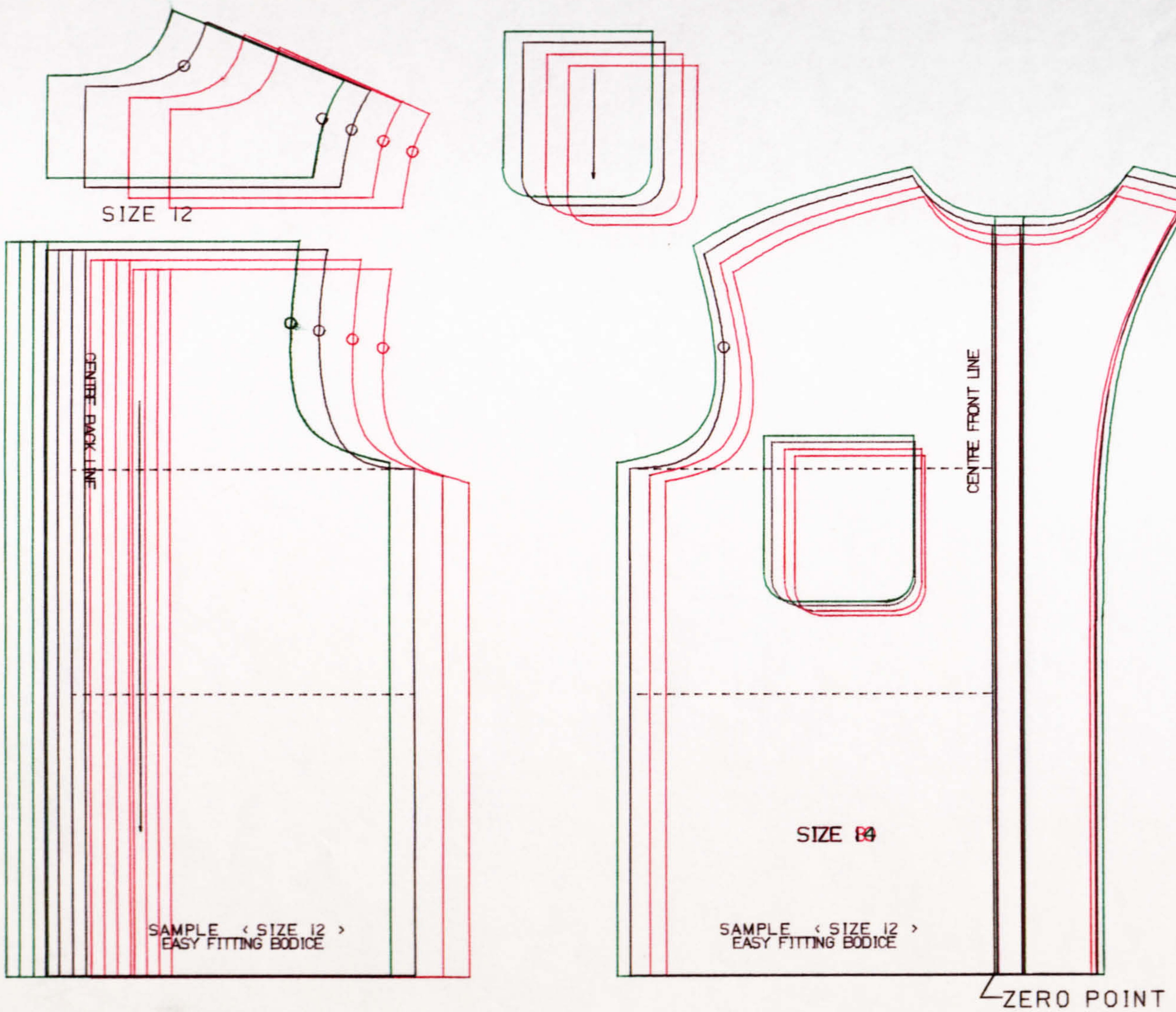


FIGURE 5-44: AN EXAMPLE OF BODICE PATTERN GRADING
SCALE: 1/5

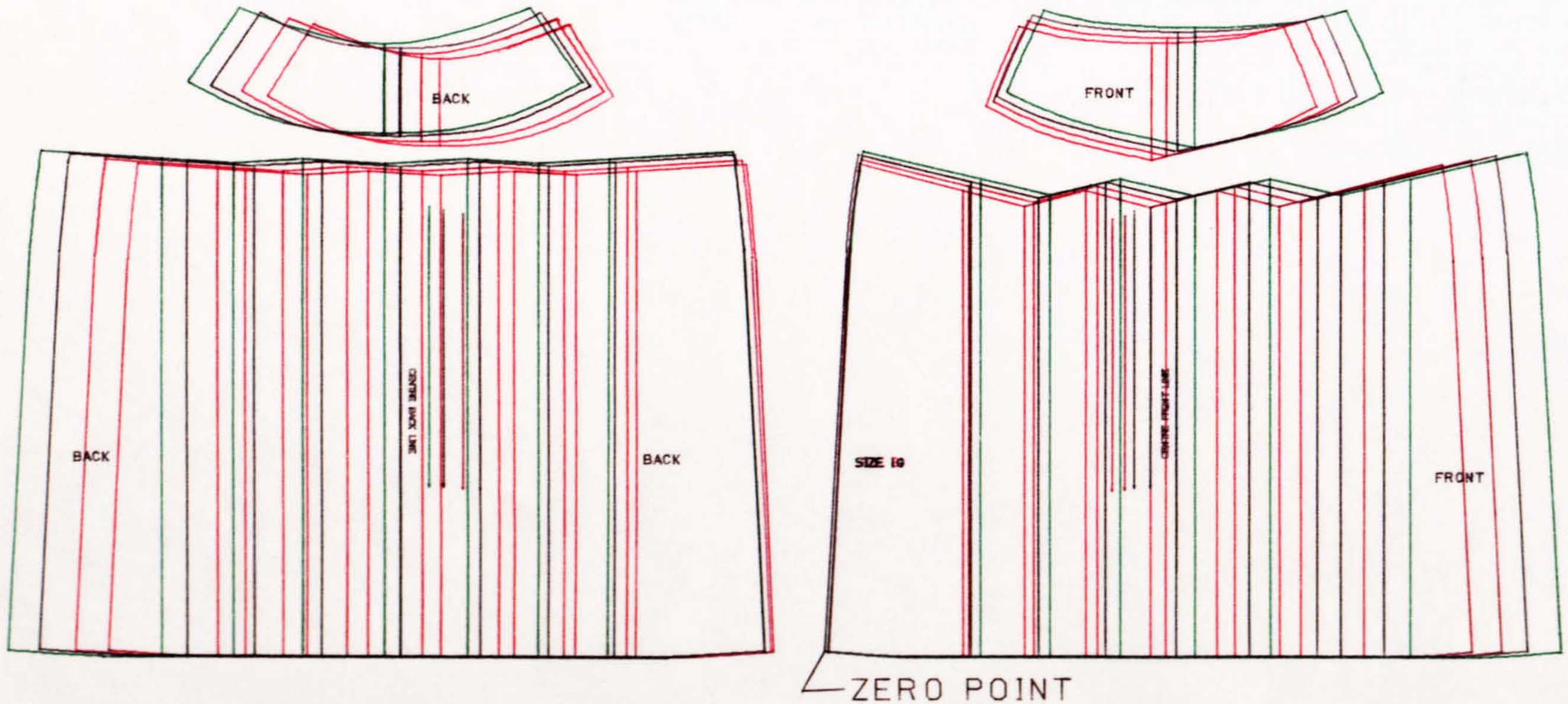


FIGURE 5-45: AN EXAMPLE OF SKIRT PATTERN GRADING
SCALE: 1/10

FIGURE 5-46: AN EXAMPLE OF TROUSER PATTERN GRADING
SCALE: 1/5

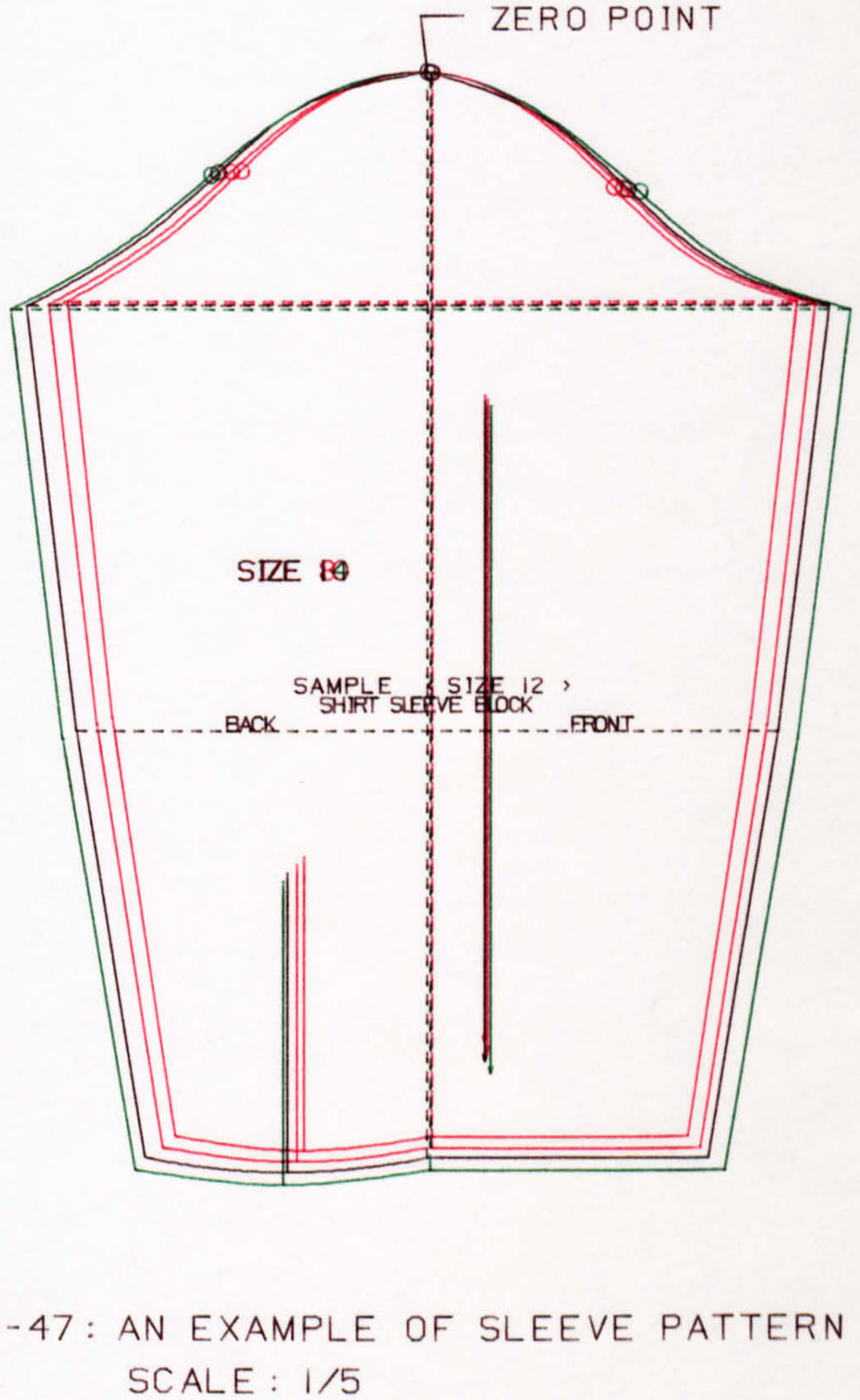


FIGURE 5-47: AN EXAMPLE OF SLEEVE PATTERN GRADING

SCALE: 1/5

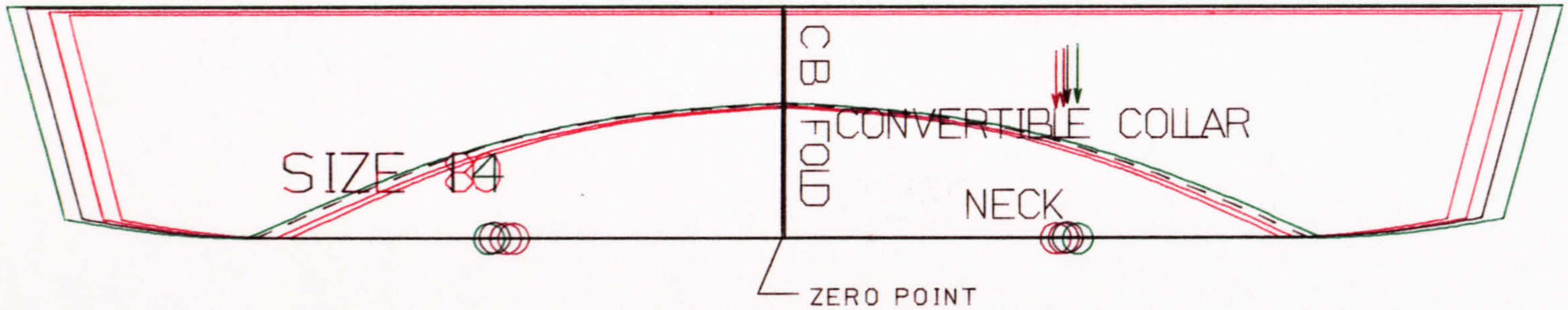


FIGURE 5-48 : AN EXAMPLE OF COLLAR PATTERN GRADING
SCALE : 1/2

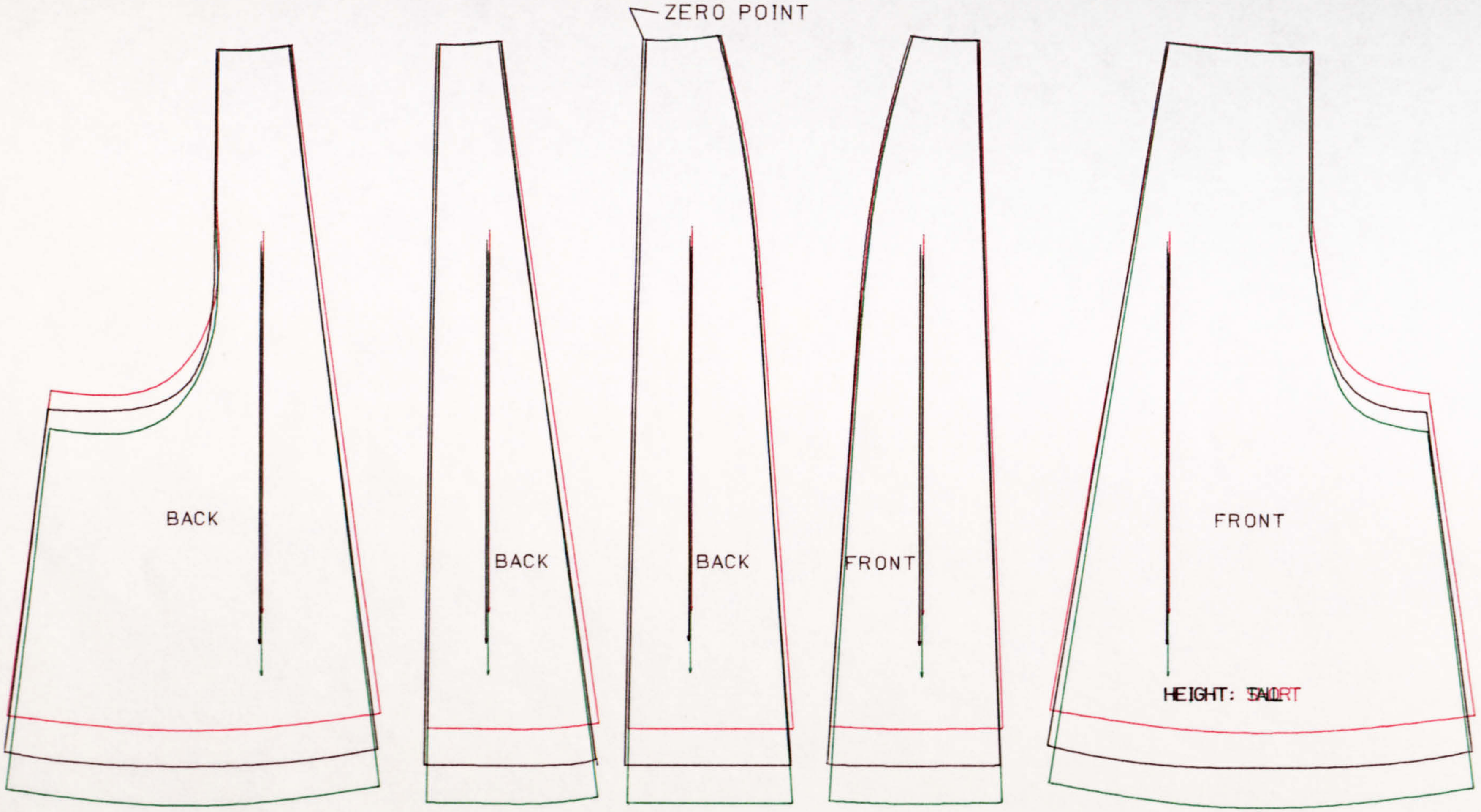


FIGURE 5-49: AN EXAMPLE OF GRADING TO DIFFERENT HEIGHT
SCALE: 1/5

FIGURE 5-50: AN EXAMPLE OF HORIZONTAL GRADING
SCALE: 1/2

CHAPTER 6

THE STRUCTURE OF THE PMS - THE ICON MENU

6.1. Introduction

Having developed the software programs to create block and auxiliary patterns and grade production patterns, it was necessary to build them into a structured Pattern Making System. In addition, appropriate graphics functions were needed to be incorporated into the system to adapt the patterns.

These basic elements then had to be compiled, indexed and presented to the user in a format that would be easily understandable and readily accessible. Finally on-line help files had to be written to provide assistance to the novice user by explaining the features and functions used in the system. This chapter describes these developments.

6.2. The Icon Menu

As shown in Figure 6-1, the host CADDS environment is presented to the user through a menu of icons. These icons are selected using a "mouse". Each icon accesses a CADDS function or a group of CADDS functions. It seemed logical to use a similar format to present the PMS to the user. At the same time, this format would create a structured

environment for the PMS. This necessitated the creation of icons that could access both the CVMAC software programs and appropriate graphic functions suitable for pattern adaptation. Consideration also had to be given as to how to organise the icons into a logical and readily accessible format for the user.

Within the CADDS software were several maintenance programs that allowed menus of user definable icons to be created. Using these maintenance programs, a command and an associated "name" could be defined for each icon of the PMS menu. Each descriptor name was displayed on a separate icon, such that when an icon was selected, the defined command for the icon was recognised as the user input. Therefore where an icon representing a CADDS graphic function was selected, a CADDS command was invoked. Alternatively where an icon representing a CVMAC program was selected, a command to run the program was initiated.

6.2.1. Menu Pages

Because all the commands could not be incorporated with one page of icons it was necessary to organise the commands into separate pages. Therefore the first decision that needed to be taken regarding the PMS menu was how to organise the menu pages. It was necessary to structure the icons and pages so that the user could easily identify the page required for the intended tasks and also to minimise the amount of page changing. Therefore, it was decided to divide the pattern making process into four stages, viz. block pattern

generation, auxiliary detailed pattern making, pattern adaptation and grading. The functions which would be frequently used needed to be grouped together on each menu page, so that the user could readily select them without having change the menu page too often. In addition, an extra page with functions necessary to improve the appearance of a pattern was needed. This was because these functions are necessary at any stage of pattern making, so that they could not be included in any particular menu page.

In the end, the PMS system consisted of five menu pages:

1. "Block" for block pattern generation
2. "Detail" for auxiliary detailed pattern generation
3. "Adapt" for pattern adaptation
4. "Grade" for pattern grading
5. "Appear" to enhance the appearance of the pattern

The five menu pages are shown in Figures 6-2 ~ 6-6 respectively.

6.2.2. Help Icons

The next step was to allocate functions to icons on each page, which could accommodate at most 57 functions. Before any functions were allocated, it was decided to assign the commands which open the help files to the first three icons of every menu page, so that the user can access the help files from any menu page without changing the menu page. On every page, the first icon was assigned to the User Help File; the second icon to the System Help Files and the third icon to the corresponding Page Help File. (The help files

are described in Chapter 8.) Also, the demonstration program for the PMS and a program which supplements the tutorial for block pattern generation were allocated to two icons on the menu page "Block". In addition, an icon of the menu page "Adapt" was assigned to the program which supplements the tutorial for pattern adaptation. (The tutorials are described in Chapter 9, Vol. 2) These icons are shown in red in Figures 6-2 ~ 6-6.

6.2.3. CVMAC Program Icons

After the help icons were allocated, the CVMAC programs were assigned on the relevant menu pages. The block pattern generation programs were located in the menu page "block", the auxiliary detailed pattern generation programs were located in the menu page "Detail", the grainline and the notch point marking programs were located in the menu page "Adapt" and the pattern grading programs were located in the menu page "Grade". The menu page "Appear" does not include any CVMAC programs, because all the functions of this page are performed by the CADDS commands. These icons are shown in green in Figures 6-2 ~ 6-6.

6.2.4. CADDS Function Icons

Finally the remaining icons were allocated to the CADDS functions which are frequently used to fulfill the task of the menu page. As explained earlier, many graphics functions are available in the CADDS environment, but from many experiments that were performed to use them for pattern

making, it was found that some were more appropriate and therefore were more frequently used than others at certain stages of the pattern making process. On this basis, subsets of the functions were selected for inclusion in each menu page. For example, to adapt a pattern, various functions to split, paste, add or move pattern pieces are necessary, so the menu page "Adapt" included various options of commands such as DIVIDE ENTITY, TRIM ENTITY, TRANSLATE ENTITY, MIRROR ENTITY AND ROTATE ENTITY.

Some commands were found to be used frequently at almost every stages. For example, the CADDS command MEASURE LENGTH was used at every stage of pattern making. Therefore, in order to reduce the necessity of changing menu page, this and other commands were duplicated onto every menu page.

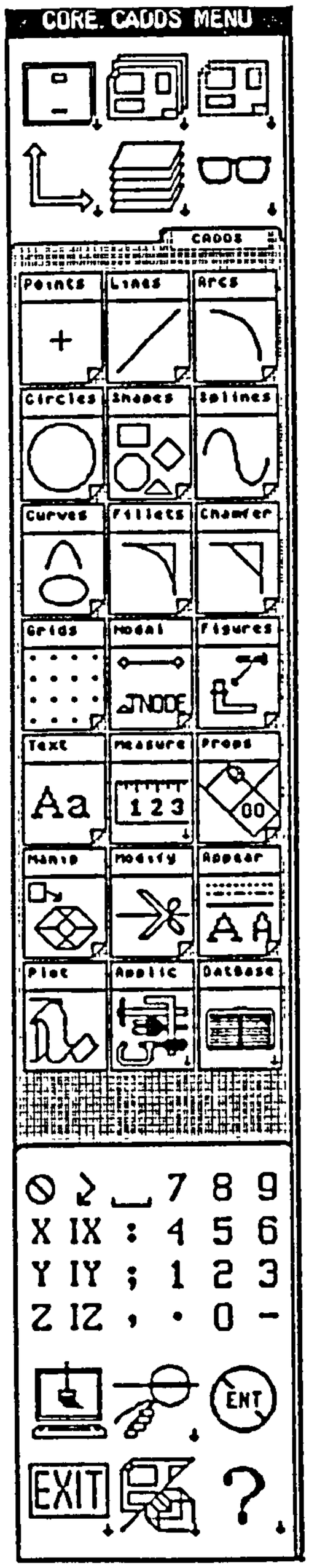
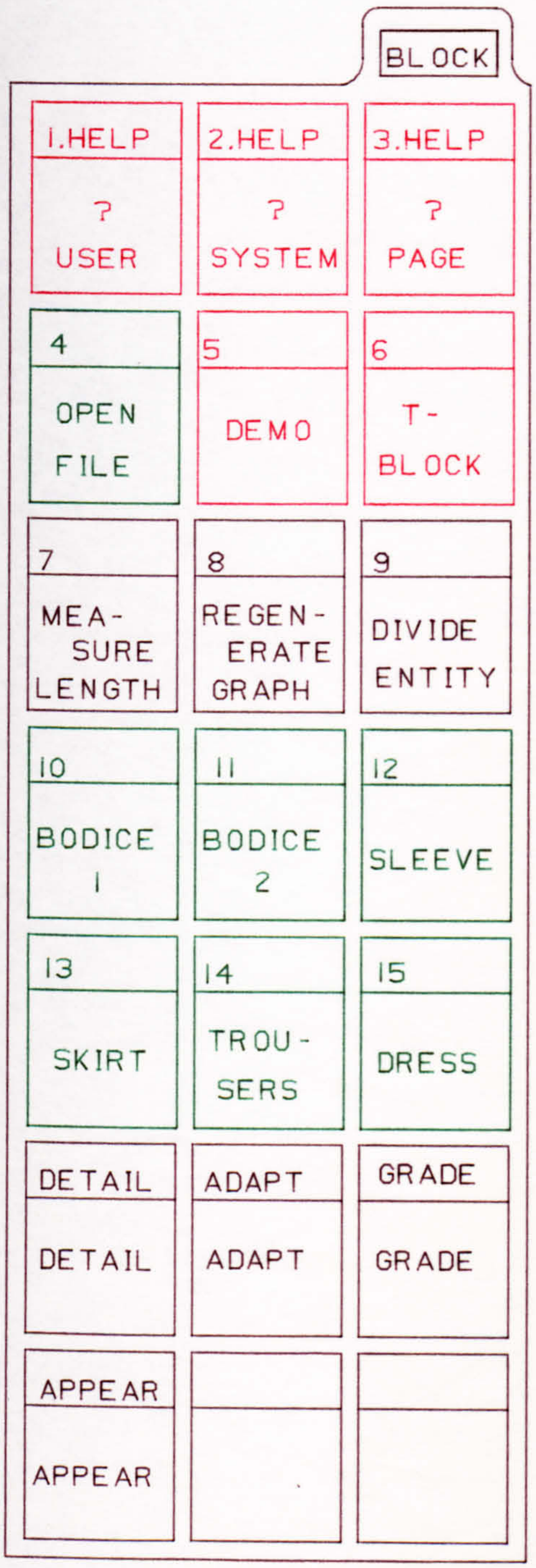


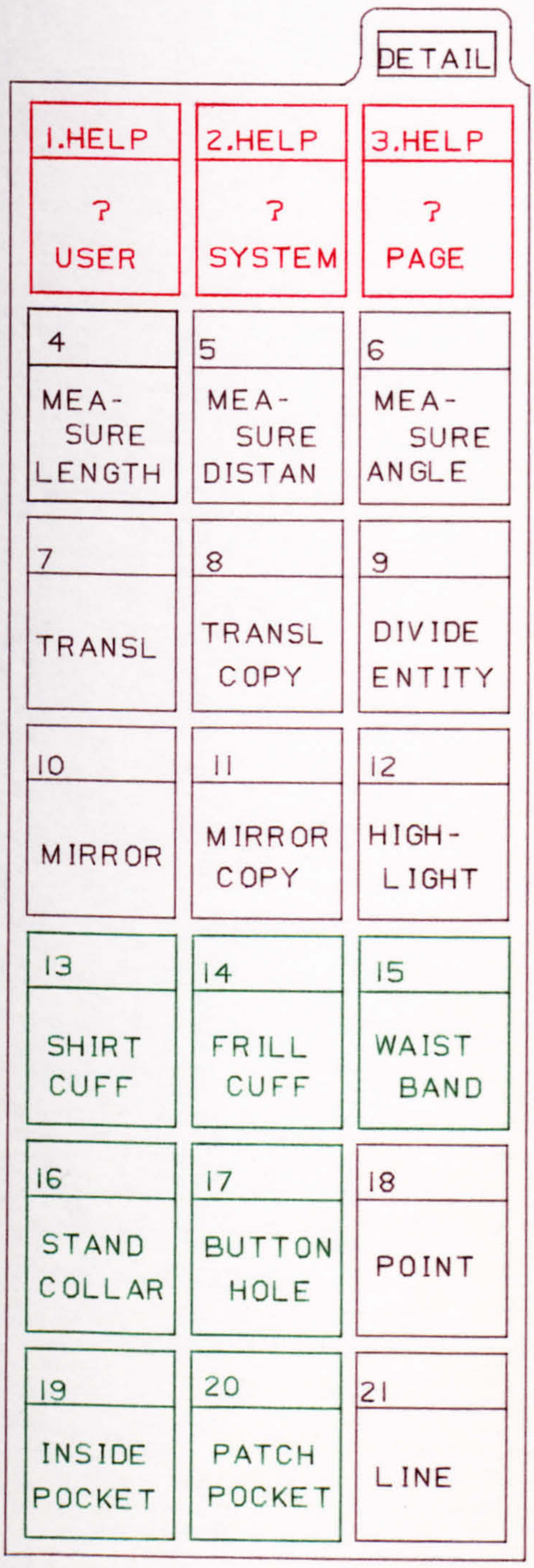
Figure 6-1: The Icon Menu



MENU PAGE - BLOCK

1. USER HELP FILE
2. SYSTEM HELP FILE
3. PAGE HELP FILE
4. OPEN FILE
5. DEMONSTRATION PROG.
6. TUTORIAL FOR BLOCK PATTERN
7. MEASURE LENGTH
8. REGENERATE GRAPHICS
9. DIVIDE ENTITY
10. BODICE BLOCK 1
11. BODICE BLOCK 2
12. SLEEVE BLOCK
13. SKIRT BLOCK
14. TROUSER BLOCK
15. DRESS BLOCK
16. PAGE CHANGE TO "DETAIL"
17. PAGE CHANGE TO "ADAPT"
18. PAGE CHANGE TO "GRADE"
19. PAGE CHANGE TO "APPEAR"
20.
21.

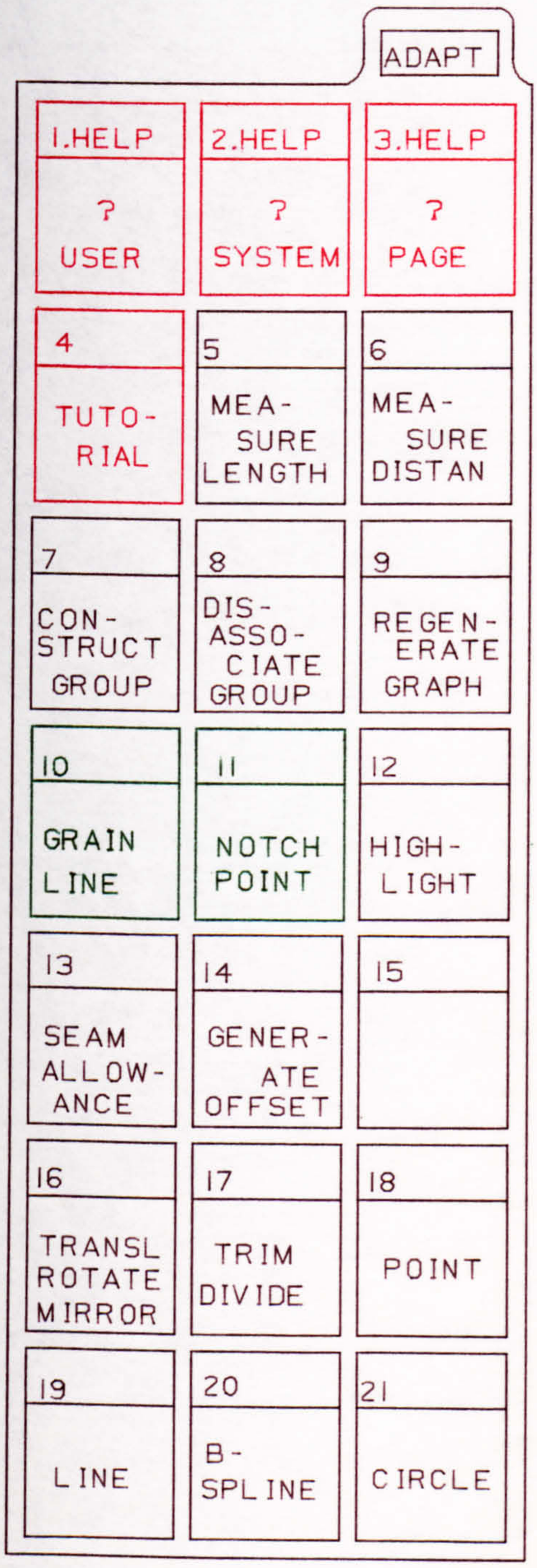
FIGURE 6-2: MENU PAGE "BLOCK"



MENU PAGE - DETAIL

1. USER HELP FILE
2. SYSTEM HELP FILE
3. PAGE HELP FILE
4. MEASURE LENGTH
5. MEASURE DISTANCE
6. MEASURE ANGLE
7. TRANSLATE ENTITY
8. TRANSLATE ENTITY COPY
9. DIVIDE ENTITY
10. MIRROR ENTITY
11. MIRROR ENTITY COPY
12. HIGHLIGHT ENTITY
13. SHIRT CUFF
14. FRILLED CUFF
15. WAISTBAND
16. STAND COLLARS
17. BUTTONHOLES
18. POINT
19. INSIDE POCKETS
20. PATCHED POCKETS
21. LINE

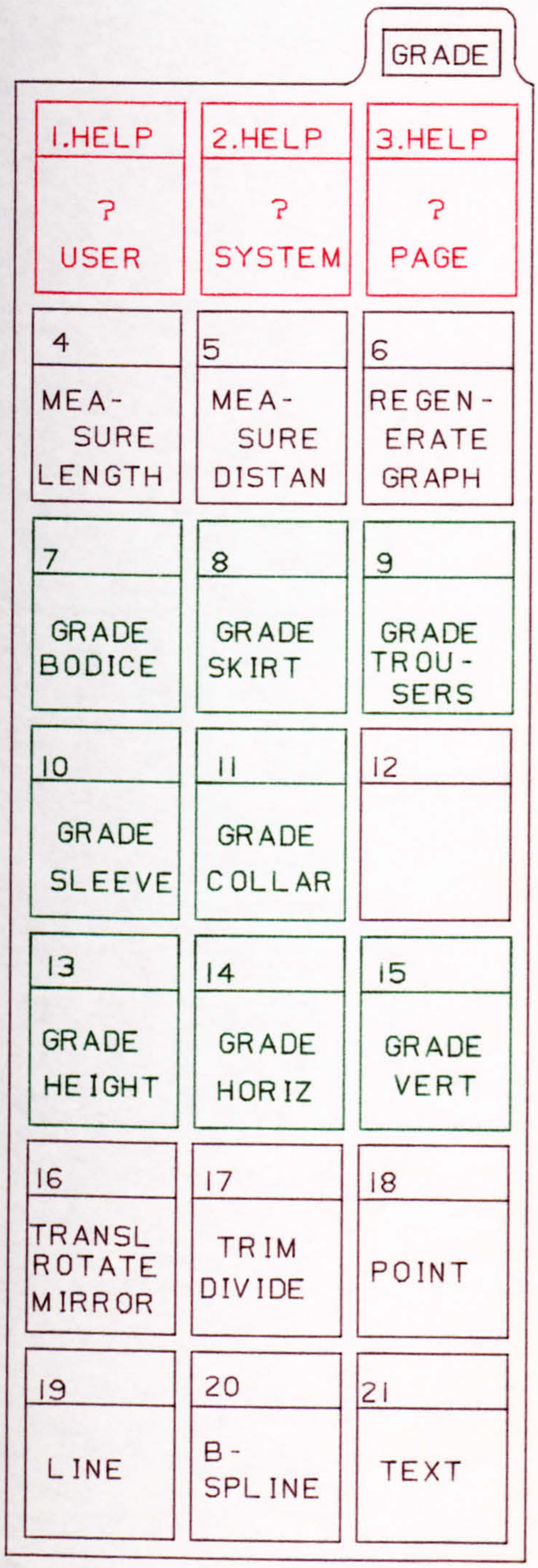
FIGURE 6-3: MENU PAGE "DETAIL"



MENU PAGE - ADAPT

1. USER HELP FILE
2. SYSTEM HELP FILE
3. PAGE HELP FILE
4. TUTORIAL FOR ADAPTATION
5. MEASURE LENGTH
6. MEASURE DISTANCE
7. CONSTRUCT GROUP
8. DISASSOCIATE GROUP
9. REGENERATE GRAPH
10. GRAIN LINE
11. NOTCH POINT
12. HIGHLIGHT ENTITY
13. SEAM ALLOWANCE
14. GENERATE OFFSET
15.
16. TRANSLATE, ROTATE AND MIRROR
17. TRIM AND DIVIDE ENTITY
18. POINT
19. LINE
20. B-SPLINE
21. CIRCLE

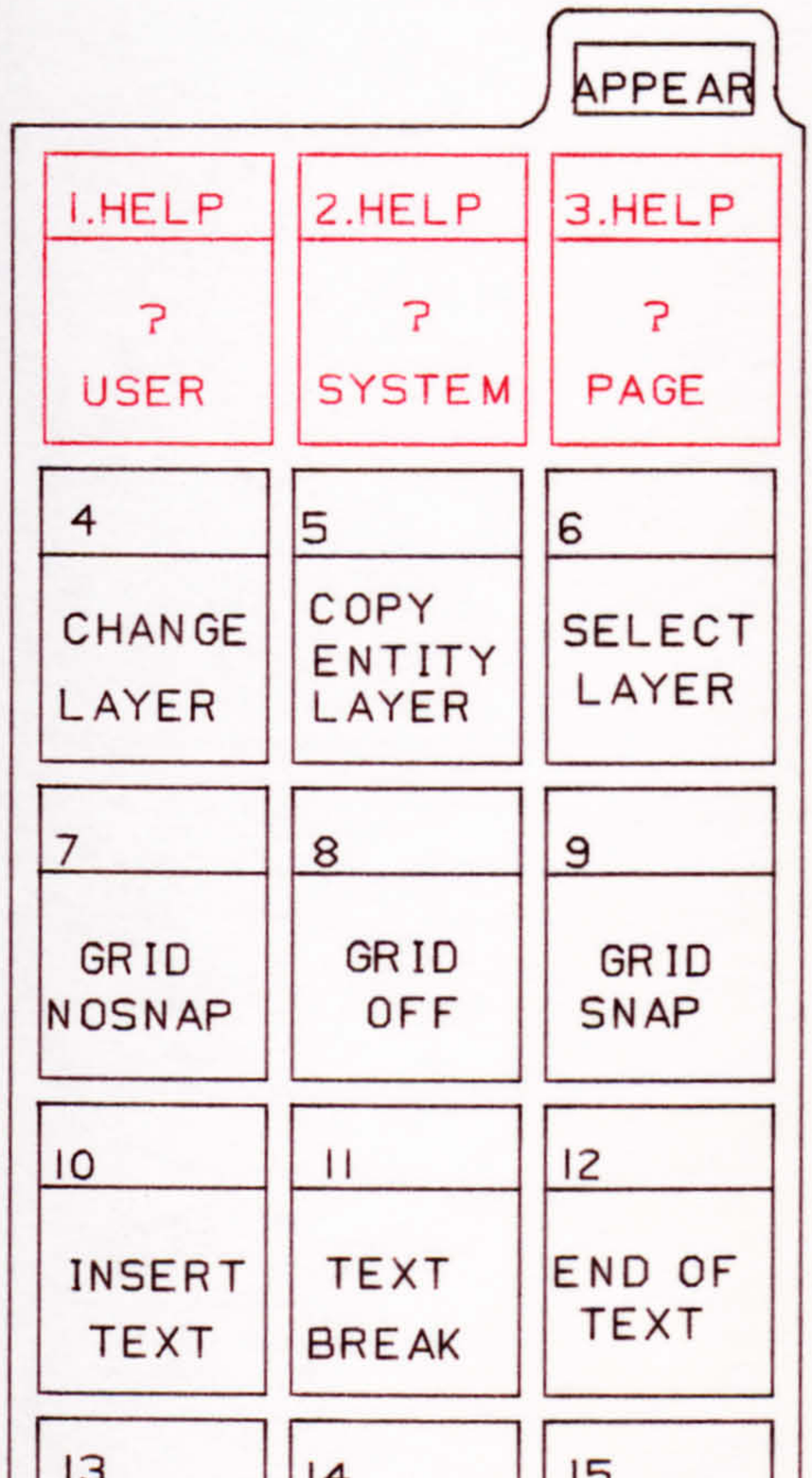
FIGURE 6-4: MENU PAGE "ADAPT"



MENU PAGE - GRADE

1. USER HELP FILE
2. SYSTEM HELP FILE
3. PAGE HELP FILE
4. MEASURE LENGTH
5. MEASURE DISTANCE
6. REGENERATE GRAPH
7. GRADE - BODICE
8. GRADE - SKIRT
9. GRADE - TROUSERS
10. GRADE - SLEEVE
11. GRADE - COLLAR
12.
13. GRADE - HEIGHT
14. GRADE - HORIZONTAL
15. GRADE - VERTICAL
16. TRANSLATE, ROTATE AND MIRROR
17. TRIM AND DIVIDE ENTITY
18. POINT
19. LINE
20. B-SPLINE
21. INSERT TEXT

FIGURE 6-5: MENU PAGE "GRADE"



MENU PAGE - APPEAR

1. USER HELP FILE
2. SYSTEM HELP FILE
3. PAGE HELP FILE
4. CHANGE LAYER
5. COPY ENTITY LAYER
6. SELECT LAYER
7. ECHO GRID NOSNAP
8. ECHO GRID OFF
9. ECHO GRID SNAP
10. INSERT TEXT
11. TEXT BREAK
12. END OF TEXT
13. COLON
14. REGENERATE GRAPH
15. HIGHLIGHT ENTITY
16. TEXT FONT
17. TEXT HEIGHT
18. TEXT ANGLE
19. TEXT SLANT
20. TEXT JUSTIFICATION
21. CHANGE FONT

FIGURE 6-6: MENU PAGE "APPEAR"

CHAPTER 7

SUMMARY, CONCLUSION AND SUGGESTIONS FOR FURTHER WORK

7.1. Summary

A computer aided design (CAD) system for the design of flat patterns for female clothing has been developed specifically for an Educational environment. This Pattern Making System (PMS) has been developed using commercial computer aided design software, Cadds 4X, originally marketed for Mechanical Engineering applications by the Computervision division of Prime Plc.

The features of the PMS include software programs, written using CVMAC, the macroprogramming language of the host CAD system, for the creation of basic blocks for many styles of garments to suit any size of female body within a specified size chart of user defined anthropometric data. Also included are software programs for the design of auxiliary patterns such as collars, cuffs etc., together with programs to proportionally grade patterns to different sizes. These programs have been compiled with appropriate Cadds 4X graphics functions, selected by experimentation, and structured into an icon menu driven shell to form the PMS.

The main emphasis in developing the PMS was to make it suitable for an Educational environment. Therefore, an extensive set of demonstration programs and on-line help files has been written and incorporated into the PMS to assist the user. These help files explain the details of the PMS and the CADDS environment, the functions of the programs and the graphics commands and give guidance in their usage to the novice user. Furthermore, a series of explicit tutorials have been written so that inexperienced users can self-teach themselves in the use of the PMS for pattern design so that the PMS also forms a system for computer based learning. Whilst the techniques used are obviously unique to the PMS, the expectation is that, through these tutorials, students will gain considerable insight into the use of a CAD system for pattern design which they can readily transfer to other commercial systems. In this context, the tutorials are regarded as an integral and essential part of the thesis.

As mentioned in Chapter 3, aspects of this system will provide the basis for further development in future projects. It is considered that such a dynamic approach well suits the question of the evaluation of the tutorials. Computer literature is often criticised as being too vague or else too esoteric. In an educational environment this need not be the case. Close liaison with the users means that feedback can be immediate. Advantage can be taken of the continuous stream of students whose reaction and opinions can be canvassed on the spot, and the nature of the

tutorials can progressively take into account their varying levels of computer literacy and aptitude. It is considered, therefore, that bearing in mind the importance of the tutorials within this system, it would be unwise to attempt an over-hasty programme of evaluation, and that the evaluation of the tutorials should be performed as part of course work in forthcoming teaching sessions.

7.2. Conclusions

Whilst there are a number of CAD systems commercially available for pattern design, with customized features in advance of the PMS, their usage throughout the Clothing Industry is currently limited. There are several factors to account for this. One has, until recently, been the relatively high cost of such systems with uncertainty over pay back. However, another contributing factor is the lack of trained personnel entering the industry with the experience and confidence to use these systems.

The development of the PMS represents an attempt to address this problem. It has concentrated on the educational difficulties involved in trying to teach the use of Computer Aided Design for the design of flat patterns for garment manufacture. The philosophy has been to try to eliminate some of the tedium of pattern design by simplifying the tasks wherever possible, for example, automatic basic block generation. Also by, automatically generating basic and auxiliary garment patterns, which at the outset of the work

was believed to be unique, student users are presented with a design basis with which they are familiar. This should, in turn, give them confidence and encourage them in the use of the system for further design adaptation. Having gained the confidence, assisted by the help files, demonstration programs and tutorials, users can then explore the additional extensive range of software available in Cadds 4X for even more design adaptation and experimentation.

The use of this Cadds 4X software represents the first application of this software for clothing manufacture. It is also the first time this mechanical engineering drafting software has been incorporated into a generic system for pattern design for garment manufacture. This in itself presented several problems. Limitations were experienced with the CVMAC programming language. For example certain 4X graphics functions could not be used in CVMAC. Transferring back and forth from the running programs to the Cadds 4X environment introduced task management difficulties that require additional keyboard activity by the user of the PMS. The format of the graphics and text windows can be irritating to the user. The overall presentation of the PMS is governed by the Cadds 4X environment and presentation. Nevertheless, in the light of the experience gained in using the system, the existing and potential advantages in using Cadds 4X, in terms of, for example three dimensional surface design and advanced graphics design and the inclusion of material properties into pattern design, outweigh these disadvantages.

For the Clothing Industry to utilise efficiently the many technological advances that are being made, further attention needs to be given to educational requirements to ensure that management and operatives have the understanding, experience and correct mental attitude towards the use of such technology. This can best be achieved by developing systems that incorporate the needs of both industry and education. The developments described in this thesis are an attempt to meet the educational needs. However more developments of this nature are required.

7.3. Suggestions for Further Work

It should be stressed that the PMS developed in this work is only a part of a more comprehensive computer aided design system planned for development. However, it is a vital part as it provides the foundations for the system and establishs the format for further developments. Clearly there are improvements that can be made. For example:

1. Currently block patterns are generated according to instructions by Aldrich [27]. Software programs may be written to facilitate the automatic drafting of basic blocks using alternative methods.

2. A simpler method for adding seam allowance is required, with for example the automatic generation according to seam construction.

3. A more versatile and faster grading system is required. The concept of automatically deriving the grade ratio from the size chart data should be applicable to determine grade rules automatically. This would, thereby, eliminate the need for grade rule generation separately.

Some suggestions for further development of the PMS are as follows:

1. The concept of automatic block generation may be extended to mens and childrens wear and clothing for other ethnic cultures.

2. The system needs to be interfaced to faster graphics type design systems to increase the flexibilities and design potential of the system.

3. The flat patterns designed by the system need to be convertible to surfaces as a precursor to three dimensional design of garments.

4. Material properties need to be incorporated into the system for the calculation of ease allowance, which is usually assessed subjectively, and to ensure the correct matching of parts to ensure acceptable garment appearance with for example minimal seam pucker.

CHAPTER 8
HELP FILES

8.1. User Help File

The User Help File gives some elementary explanation about Pattern Making and is a general guide on how to use this system.

1. What is the Pattern Making System?

The Pattern Making System is a group of programs which generates block patterns; makes detail patterns such as a collar or a pocket; adapts a block pattern; makes production patterns and grades them into different sizes.

The system is separated into small programs, each of which performs only one task, so you can arrange jobs in any order you wish. Also every program is connected to an icon in the user menu so that you can perform any task simply by selecting an icon using the mouse instead of invoking a complicated command.

2. What Kind of Work Can This System Do?

Programs are grouped and allocated into 5 pages of the user menu by way of a general pattern making procedure. Those 5 pages of the user menu are Block, Detail, Adapt, Grade and Appear.

The main functions of each page of the user menu are as follows.

Block --- Generates block patterns (bodice, sleeve, skirt, trouser and dress) in many different styles.

Detail -- Generates detail patterns (cuffs, waistband, collar, buttonhole place, pocket) in many different styles.

Adapt --- Adapts patterns by constructing new lines, cutting parts, rotating, mirroring, moving parts or join some parts. Also you can add seam allowance to make production patterns.

Grade --- Grading patterns into different sizes by built in grade rules. Also you can grade only in a lengthwise or widthwise direction.

Appear -- Changing line font, inserting text to give instructions to make patterns understandable.

Also there are many icons which call CADDS commands such as INSERT POINT, MEASURE LENGTH or ROTATE ENTITY to support the programs mentioned above.

3. How to Work with This System?

There are 2 types of icons, which work in very different ways. One type has CADDS command, and the other is connected with programs which have been written in CVMAC which is the CADDS programing language for this computer system.

When you select an icon using the mouse if you get a command line straight after the CADDS prompt #n#, that icon must be a CADDS command icon.

Ex) #n# INSERT LINE : MODEL loc ddd

#n# CONSTRUCT OFFSET D -> 1 : MODEL ent d MODEL loc d

But when you select an icon using the mouse if you see the command line "RUN CVMAC XXXXXX" after the CADDS prompt #n# and two blank lines and COMPUTERVISION CVMAC PROCESSOR VERSION 1.00 at the beginning of a program that icon must be a programed icon.

Ex) #n# RUN CVMAC BDCLIMAGE

COMPUTERVISION CVMAC PROCESSOR VERSION 1.00 date

3.1. CADDS Command Icon

These icons work in the way explained in the System Help File.

a. When a command line is completed, a colon (:) appears together with the getdata prompt such as "MODEL loc" or "MODEL ent" to let you know what kind of data you have to input using either the mouse or the keyboard.

b. If a command needs a user input modifier to complete the command, you receive a -> prompt. Enter modifiers from the keyboard. Then you will get a colon (:) and the getdata prompt as explained in a.

c. If you start your CADDS session using the icon "Open File", the operating mode is always in MODEL mode (actually you don't need to worry about the mode).

When you get the prompt "MODEL loc", input a location(s).

If you get the prompt "MODEL ent", input an entity(s) depending on the command.

d. When you want more information about a command, refer to CADDS Core Reference (Vol. 1,2).

e. For more information, read the System Help File.

3.2. Programed Icon

When you select one of these icons, the icon calls the execute file of a program written in CVMAC.

a. When you answer a question, you need to read the instructions in the program carefully and follow these instructions.

b. While you are working with the execute file, you cannot use any icon or input any CADDS command, because whatever you input is interpreted as a variable within the program.

c. Sometimes you might get instructions about #n# <VAR> sequence which does enable you to use icons or enter CADDS commands in the middle of the program. However, in this sequence the execute file is suspended and you

receive the prompt "#n# <VAR>" and your input is read and executed by CADDS processors. To finish this sequence and return to the program, you must press the Control key and X at the same time.

d. To stop the program at any time, press Esc key (Escape) and press Return. Then you receive the message TYPE "OK" TO ABORT. Type "ok" and press Return.

4. Useful CADDS Manuals

4.1. CADDS User Guide

The following sections comprise the CADDS User Guide:

a. Getting Started in CADDS (tutorial section for the beginner)

b. Understanding the Architecture of CADDS

c. Managing Your CADDS Session

d. Working in the Graphics Environment

e. Using CADDS Menus

4.2. CADDStation User Interface

This booklet explains CADDStation System's user interface, including how to use the mouse, how to work with menu sets, how to use getdata property sheets and popups.

4.3. CADDS Core Reference (Vol. 1,2)

This manual has information including description, syntax, modifiers and examples with diagrams about every CADDS command. When you enter a CADDS command and receive a

getdata prompt but you cannot decide what entity or what location to enter, consult this reference.

5. Help File for Pattern Making System

5.1. User Help File

The help file which you are reading now is the User Help File.

This help file has simple explanations about the Pattern Making System and is a general guide on how to use this system. You can always open this help file by selecting icon number 1 in any page of the user menu set.

5.2. System Help File

This help file describes basic information and instructions which are needed to work with CADDS (Computervision Automated Design and Drafting System), the software environment for Computer-aided design/Computer-aided manufacturing (CAD/CAM) graphics processing. You can open this help file by selecting icon number 2 in any page of the user menu set.

5.3. Page Help File

This help file provides more information about every icon (except help icon) on each page. You can open the Page Help File by selecting the icon number 3 in each page.

6. Demonstration Program

This program has been written to try to demonstrate how to use the Pattern Making System. The program comprises 5 parts which are as follows;

a. Block patterns: Displays block patterns which can be generated using this system.

b. Adapt: How to adapt a block pattern to make a design pattern.

c. Production Pattern: How to make production patterns.

d. Grade: Examples of patterns which are graded using this system.

e. An Example

You can run the demonstration program by selecting icon number 5 "Demo" in the root menu page "Block".

7. Tutorial

The tutorial is devised to give step by step guidance to the novice user of the Pattern Making System. The difference between the help files and the tutorial is that when you use the help file you have to search for the information you want. However, if you use the tutorial you do not need to be frustrated looking for any information because the tutorial is written as if the tutor is sitting with you and explaining the procedures and the necessary information. Therefore, if you are a novice user with some experience with other general purpose computers has never used CADDS, it would be better to follow the tutorial in sequence.

The tutorial course consists of the following parts.

Basic Tutorial for CADDS

Basic Tutorial for Pattern Making System

Advanced Tutorial

Also each tutorial is split into several sessions so that you can pause when you want.

8.2. System Help File

This help file describes basic information and instructions which are needed to work with CADDS (Computervision Automated Design and Drafting System), the software environment for Computer-aided design / Computeraided manufacturing (CAD/CAM) graphics processing.

Contents

1. Punctuation Marks and Control and Escape Sequences under CADDS

1.1. Punctuation Marks

CADDS uses a standard set of punctuation marks. A particular character will perform the same function throughout CADDS.

[SPACE BAR] (Blank)

Separates elements of the command (verb, noun, modifiers,values).

Ex: #n# ECHO LAYER ALL

, (Comma)

Separates sets of values(X2Y3,X5Y6) or items of a list.

Ex: #n# INSERT LINE: MODEL loc X2Y3,X5Y6

(Two locations X2Y3 and X5Y6 are separated by comma.)

(Period) .

Ends the command line, including getdata. CADDS executes the command and then reenters the same command without modifiers, but does not enter getdata.

Ex: #n# INSERT CIRCLE DIAM 2: MODEL loc d. DIAM 3: MODEL loc d

: (Colon)

* Outside getdata

Signals that the command line is complete and calls getdata.

Ex: #n# BLANK ENT: MODEL ent d

* In getdata

Tells CADDS to process the active command and then reenter that command, including modifiers, and reenter getdata. When ready, CADDS prompts for your input.

Ex: #n# CHANGE LAYER 5: MODEL ent d: MODEL ent dd

; (Semicolon)

Switches modes (from selecting entities to specifying coordinates) or signals the completion of a step in a procedure.

Ex: #n# TRANSLATE ENT: MODEL ent d; MODEL loc dd

[CR] (Return)

Ends command entry, tells the system to execute the command.

Ref: CADDS User Guide, 1-1 ~ 1-3.

1.2. Control and Escape Sequences

Most functions in CADDS are controlled by commands, but there are also functions controlled by special two- and three-key sequences.

[CTRL]BQ : Halts program execution and returns to the CADDS prompt.

[CTRL]E : Returns the process to its state before the last command.

[CTRL]X : Resumes processing of execute file halted with <VAR>.

[CTRL]N : Returns to the CADDS prompt.

[CTRL]U : Locks your terminal; unlocks by repeating [CTRL]U.

[CTRL]CC : Stops CADDS process and returns to the operating system. Do not use the control sequence [CTRL]CC without fully considering its effects. If a part

is active, abnormally ending the session will damage or lose the part.

[ESC] : Suspends execution of the command; execution resumes with a key stroke or RETURN.

[ESC]Q : Stops execution of the command and returns to CADDS.

Ref: CADDS User Guide, 1-3,1-4.

2. The Mouse and Mouse Pad

The mouse is an interactive input device used with a mouse pad as a cursor positioning device. The mouse has three buttons. The left button is for selecting an initial object. The right button is for activating a new menu depending on circumstances and cursor position. And the middle button is for querying an icon.

For more information, see '3. Windows', '4. Icon Menus' and '6. Getdata' in this help file please.

Ref: CADDStation User Interface, 7.

CADDStation User Guide, 3-25,4-6.

3. Windows

3.1. Names and Functions of Windows under CADDS

The CADDStation User Interface comprises the following areas and related tools.

3.1.1. CADDS 4X Graphics Window

This area of the screen, entitled CADDS 4X Graphics Window, contains displayed geometry. The size of the Window may be up to 1142 pixels wide and 877 pixels high.

3.1.2. CADDS 4X Drawing Window

This area shows you the relative position of your graphics window on the entire drawing, providing you with a context when you are working in a zoom or scroll state. It displays the entire drawing with a temporary zoom window on top of it, which represents the portion of the drawing you are working on.

3.1.3. CADDS 4X Status Window

This area contains a series of phrases that show the environmental status of the session. Status information includes the current part and drawing names, active mode and construction plane, as well as the command environment and selected base units.

3.1.4. CADDS 4X Text Window

This area contains 20 lines of command input and CADDS dialog.

Note: Shell Area

Below the graphics window there is an area where the background (a stippled surface) of the display is visible. Use this area to activate additional windows, store closed icons, or access UNIX operating system level commands.

Ref: CADDS User Guide, 1-44,1-45.

3.2. Hiding, Exposing, Closing and Reopening Windows

3.2.1. Hiding and Exposing Windows

There are two methods for hiding or exposing a window within CADDS. The first uses the popup found in the window frame. Move the cursor into the window's frame. As the cursor moves into the frame, it changes to a circle with a dot in the centre. Hold down the right button to popup the menu and slide the mouse until the desired option is highlighted, then release the right button.

In the second method, move the cursor into the window's frame, then click (press and release) the left button to move the window to the foreground. For example if the graphics window partially hides the text window, selecting the text window's frame causes it to then partially hide the graphics window.

3.2.2. Closing and Reopening Windows

To close a window, access the pulldown menu by moving the cursor into the window's frame. As the cursor moves into the window's frame, it changes to a circle with a dot in the centre. Hold down the right button to popup the menu and slide the mouse until the 'close' option is highlighted, then release the right button. The window closes to a single icon.

To reopen a closed icon (closed window), position the cursor over the closed icon of the window. Click the left button to select the closed icon. The full window is redisplayed.

Ref: CADDS User Guide, 1-46.

CADDStation User Guide, 3-1 ~ 3-7.

4. Icon Menus

4.1. What are Icon Menus?

Menus are tools that allow you to execute commands without actually typing a series of elaborate commands. The menu display is on the screen of your workstation, and you select an icon with your mouse. Icons are elements that identify actions CADDS can take or point to submenus. Icons can represent any command item from one letter or a punctuation mark to full commands, execute files or applications. Icon-driven menus contain more than one menu subset. An icon-driven menu usually contains a root menu and a number of additional hierarchically related menus. Only one level of the icon-driven menu appears on the screen at a time.

In this user menu set, the root menu is 'block' and there are four submenu pages, which are 'detail', 'adapt', 'grade' and 'appear'.

Ref: CADDS User Guide, 5-4 ~ 5-6.

CADDStation User Interface, 1 ~ 5.

4.2. Selecting an icon

Types of icons and their resulting actions are explained below.

Types of Icon	Action
Single Icon	Allows immediate execution of a single command.
Down Arrow Icon	Opens pulldown of options.
Bent Corner Icon	Provides access to sub-menu page of related icons.

4.2.1. Selecting a Single Icon

Move the mouse cursor onto a single icon until the icon is highlighted, then click the left button.

Note: The action your icon selection produces can comprise any amount of input - from a single character up to several CADDS commands strung together.

4.2.2. Selecting a Down Arrow Icon

Move the mouse cursor onto a down arrow icon until the icon is highlighted. Then keeping the left button depressed, highlight desired option by moving a cursor down the list of options. Release the left button to select option.

To make no selection, move the cursor off the option, then release the button.

Note: Your last selection becomes the default selection the next time you access a pulldown.

4.2.3. Selecting a Bent Corner Icon

Move the mouse cursor onto a bent corner icon until the icon is highlighted and then click the left button. The action

your icon selection produces is changing the menu page into a submenu page. Each menu page is marked by a tab icon similar to a file folder title. Select a tab icon to recall the associated menu page. To move from a submenu page to the root menu page, select a tab icon of root menu page.

4.2.4. Dynamic Menu Variable Input (dmvar)

Specific icons that issue commands requiring user input are programmed with dynamic menu variables. You might not know whether an icon has any dynamic menu variables, until you select the icon and get the prompt ->, which asks to enter the value for the dynamic menu variables.

When you select one of these commands, the system pauses after -> prompt so you can enter a proper value. You can enter your input from the keyboard. Enter a space to terminate your input and execute the command. If you want to find out exactly what kind of variable is required, refer to CADDS Core Reference (Vol 1,2).

Ex: #n# GENERATE POINT ON N -> 3 : MODEL ent d

In this command the number of points(3) has to be input after a -> prompt. Then the command line is completed and you can receive a getdata prompt (MODEL ent).

Ref: CADDStation User Interface, 6 ~ 15.

4.3. Querying an icon

For brief information, move a mouse cursor onto an icon until the icon is highlighted. Click the middle button. A

green help window appears in the centre of the screen to explain the icon's function. Click any button to unlock the cursor and resume working.

But if you want more information, see page help file. In the user menu set, the top right icon of any page is for a page help file, which has detailed information about icons in that menu page. Move the mouse cursor onto the page help icon until the icon is highlighted. Click the left button, the page help file will appears in CADDS Text Window. Follow instructions in the file.

Ref: CADDStation User Interface, 9.

4.4. Menutools Popup

Menutools popup lets you open and close a property sheet of getdata modifiers. For more information about property sheet and getdata, see '5. CADDS Commands' and '6. Getdata' in this help file please.

Move a cursor into a menu window or inside a property sheet. Press the right button to call a popup. Keeping the right button depressed, slide the cursor up or down until the your choice highlighted. Release the right button to issue your selection.

Ref: CADDStation User Interface, 32 ~ 33.

4.5. Constant Area in the Menu Window

4.5.1. Upper Constant Area

The six icons, visible during menu sessions, represent commands available throughout an application.

File Icon	Calls a pulldown of filing options.
Drawing Icon	Calls a pulldown of drawing related options.
View Icon	Calls a pulldown of view related options.
CPlane Icon	Calls a pulldown of Cplane related options. (CPlane: Construction Plane)
Layer Icon	Calls a pulldown of layer related options. See '9. Managing Layers' please.
Visibility Icon	Calls a pulldown of visibility and appearance related options.

4.5.2. Lower Constant Area

* Keypad Icons

Keypad icons supply numbers or punctuation (not a command). These icons work like a keyboard. Whenever you want to enter a number or punctuation shown below, select one of these icons instead of using the keyboard.

Reading left to right, these keypad icons do the following:

Stops operation.	Enters a RETURN.
Enters a space.	Enters the numeral 7,8,or 9.
Enters X.	Enters incremental X.
Enters a colon.	Enters the numeral 4,5,or 6.
Enters Y.	Enters incremental Y.
Enters a semicolon	Enters the numeral 1,2,or 3.
Enters Z.	Enters incremental Z.
Enters a comma.	Enters a period (decimal point).
Enters a 0 (zero).	Enters a minus sign (hyphen).

* Constant Operations Icons

Operations icons perform a single operation or contain pulldowns of options as follows:

Repaint Icon	Repaints the display.
Display Control Icon	Calls a pulldown of display control options. See '8. Managing the Display' please.
Delete Entity Icon	Removes entities from a part.
Exit Icon	Calls a pulldown of exiting options. See '7. Managing Files' please.
Mode Icon	Calls a pulldown of Model/Drawing mode options.
Question Mark Icon	Calls a pulldown of entity verification options.

Ref: CADDStation User Interface, 20 ~ 31.

5. CADDS Commands

5.1. Structure of CADDS Commands

CADDS operates interactively through commands. As a general rule, a CADDS command comprises the following:

verb noun [modifiers] : getdata

Ex: #n# INSERT CIRCLE DIAMETER 4.0 : MODEL loc X0Y0

There are processors in CADDS, which are responsible for reading and executing each of these elements. They are as follows:

* Verb/Noun Processor	Processes the verb/noun portion of the command.
* Modifier Processor	Processes modifiers.
* Command Processor	Processes the modifiers for commands not handled by the modifier processor.
* Getdata	Interactive data input for command execution. See '6. Getdata' please.

Ref: CADDS User Guide, 2-1 ~ 2-13.

5.2. Querying Commands

5.2.1. Listing Verb Options

Enter a question mark(?) to see which verbs are available with your CADDS.

Ex: #n# ?

CADDS prints a list of all verbs you can use.

5.2.2. Listing Noun Options

To find out which nouns are available with a particular verb, enter the verb followed by a question mark(?).

Ex: #n# INSERT ?

CADDS prints a list of available nouns with INSERT.

5.2.3. Asking for More Information

On-line documentation exists for every CADDS command, accessible by entering an exclamation mark(!) along with the verb and noun.

Ex: #n# INSERT ! POINT

CADDS displays the file describing 'INSERT POINT'.

Ref: CADDS User Guide, 1-15 ~ 1-18.

6. Getdata

Getdata is the part of CADDS that handles all graphics data required by CADDS commands. In the example below

#n# INSERT LINE HORIZONTAL : MODEL loc dd

INSERT is the verb, LINE is the noun, HORIZONTAL is the modifier, and 'MODEL loc' is the getdata prompt.

Communication is possible in both directions within getdata:

* Prompts tell you what getdata needs.
* Getdata modifiers you enter help specify your responses.

Note: The 'd' that appears throughout this help file or in any CADDS documents is the way CADDS shows it accepts your entity or location specification in getdata.

6.1. Getdata Prompt

Getdata displays messages called prompts. Each prompt asks you to supply a certain type of getdata in a certain mode.

6.1.1. Getdata Type

Each prompt asks you to supply one of three types of data:

* A location
* An entity
* A view

Depending on the command, a certain type of data is needed in a certain order. For example:

#n# DELETE ENTITY : MODEL ent dd

To delete entities, you have to define entities which you want to delete.

#n# SCROLL VIEW : view d ; DRAW loc d DRAW loc d

To scroll the view, first you have to select the view(s) to be scrolled. Then two locations have to be entered to define the way of scrolling, that is entities in that view are scrolled from the first location to the second location.

6.1.2. Getdata Mode

Getdata prompts include a reference to a mode as part of a prompt.

* Model mode: Refers to model space, the three-dimensional space in which the model resides.
* Draw Mode : Refers to drawing space, the two-dimensional space of the active drawing.

Under the model mode entities or locations are recognized by X,Y,Z values, but under the draw mode they are recognized by only X,Y values. Some command specially requires a certain

mode, but usually mode of the getdata prompt is same as the current mode, which you can find in the CADDS Status Window. If you have to change the mode, select Mode Icon at the lower constant Area of the menu window. If you started your current part (CADDS files) using the icon Open File, the current mode is model mode. As long as you work with Pattern Making System, model mode is sufficient.

Ref: CADDS User Guide, 2-14 ~ 2-16.

6.2. Getdata Modifiers

To give you the flexibility when responding to a getdata prompt, modifiers exist that help you specify, filter, or reference the needed data. Each time getdata displays a prompt, all previously entered modifiers are no longer active. You can backspace within getdata modifiers to correct an error.

Getdata handles three types of input:

* Location specification
* Entity selection
* View selection

More explanation about how to use getdata modifiers will be given in '6.3. Responding to Getdata Prompts'.

Ref: CADDS User Guide, 2-16, 2-17.

6.3. Responding to Getdata Prompts

In getdata, you are asked to identify locations, entities, or views as part of command execution. There are three ways to do this:

* Type the information on the keyboard.

* Make selections from the getdata popup menu.

* Make selections from the getdata property sheet.

For simplicity, this basic explanation of getdata responses will be based on the keyboard. As you become familiar with the getdata prompts and responses, you might want to try the getdata popup menu or property sheet.

Most responses are interpreted as coordinates. A coordinate consists of a name (X,Y,Z,R or A) and a value called the coordinate component value.

6.3.1. Specifying a Location

* What Is a Location ?

A location consists of an X-, Y-, and Z- coordination. No matter the method of input, getdata uses each location as a triad of coordinates. The way getdata reads your response depends on the needs of the command, the active mode, and the dimensions (two- or threedimensions) of the part.

* Explicit Coordinates

Explicit coordinates are exact geometric locations copied (mapped) directly to the data base. Getdata interprets explicit coordinate input in the context of the part, drawing, and view and interpretes the coordinate set based on the mode of drawing active when you enter getdata.

* Point-on-Entity Modifiers

To give getdata a location, selecting an entity at the desired location often is the easiest method for replying to a location prompt. After the getdata prompt, enter a point-on-entity modifier, and enter an entity(s).

Point-on-Entity Modifiers

Modifier	Explanation
END	Selects the endpoint of an entity to specify a location.
ORG	Selects an entity's origin to specify a location at it or relative to it.
NEAR	Selects the nearest or the normal point on the selected entity to specify a location.
INTO	Selects the location at the geometric intersection of the selected entities.

6.3.2. Entity Selection

There are many ways you can select entities in response to a getdata prompt. An entity is a single element in a design e.g. line, point etc..

* Entity Modifiers

An entity modifier tells getdata that the command applies uniformly to all entities of this type instead of requiring you to specify each entity individually.

Entity Modifiers

Modifier	Meaning	Modifier	Meaning

ARC	Arc	BSPL	B-spline
CIR	Circle	FIL	Fillet
GPOI	Grid Point	LIN	Line
POI	Point	TEXT	Text
XHA	Crosshatch		

* Specifying with Boundaries

When you wish to change something in a particular area, use one of the window modifiers listed below.

Window Modifiers

Modifier	Meaning	Explanation
PWIN	Polygon Window	Selects entities by specifying the vertices of a polygon enclosing the desired entities.
WIN	Window	Selects entities by specifying the diagonally opposite corner of a rectangle enclosing the desired entities.
VWIN	View Window	Selects entities by selecting a view that contains them.

* Selecting Entities by Name

In largely or densely organized parts, the selection of an entity by location, either by mechanical means or by coordinates, may not identify only one entity. Naming the entity you want makes the selection unambiguous for getdata.

To specify the entity by name, getdata needs to know that you are providing a name, not a modifier or some other data. The TAG modifier is used in these cases.

Entity Tag Modifier

Modifier	Meaning	Explanation
TAG	Tag name	References an entity by tag name instead of by location.

Ex: #n# DELETE ENTITY : MODEL ent TAG L1 L2

Lines whose tag name is L1 or L2 are deleted.

* Association Modifiers

Many entities can be affected by a single command if the entities are grouped. Using the command CONSTRUCT GROUP creates a relationship among a number of entities that need not have the same type. Thus, selecting the group in getdata allows you to affect entities you have associated based on your needs, not based on similar types or relations within a boundary.

Associative Modifiers

Modifier	Meaning	Explanation
GRO	Group	Specifies a group of entities by selecting one of the group.
CHN	Chain	Specifies a visually connected set of entities by selecting one of them.

6.3.3. View Selection

There are two ways to respond to a getdata prompt 'view':

* Specifies a view by name using a getdata modifier 'NAME'.

    Ex: #n# CHANGE VIEW SCALE 0.5 : view NAME TOP

* Selecting a location within the view using a mouse.

    Ex: #n# CHANGE VIEW SCALE 0.5 : view d

Ref: CADDS User Guide, 2-17 ~ 2-34.

6.4. Getdata Popup Menu

Once you are familiar with the getdata prompts and responses, you might want to use the getdata popup menu to respond to the prompts. It lists possible getdata modifiers and punctuation in menu form that you can select with the mouse.

1. Press the right button while the cursor is in the graphics window. The popup menu appears on the screen.
2. Keeping the right button depressed, slide the cursor down the list until the selection you need is highlighted. Notice that arrows appear to the right of some menu items. This indicates that you can access another page, or submenu of that menu item.
    a. If no arrow appears to the right of the selection, release the button. Your selection appears on the command line and the popup will disappear.
    b. If an arrow appears to the right of the selection, slide the cursor toward the arrow (keeping the right button depressed). One or more submenus fan out to

the right as you slide the cursor over the selection you need (it will be highlighted) and release the button. Your selection appears on the command line, and the popup menu will disappear.

3. If getdata requires a numeric value, you can enter it with either the keyboard or the number pad on the bottom of the menu.

4. To make no selection, slide the cursor outside the popup menu and release the button. The popup will disappear.

   Note: CADDS "remembers" the last getdata selection you made from the popup menu, so that your last choice becomes the default selection the next time you use it. If you made your last selection from a fanned submenu and you want to get back to the first page of the popup, press the right button of the mouse and slide the cursor left, back to the first page.

Ref: CADDS User Guide, 2-35,2-36.

CADDStation User Interface, 34,35.

6.5. Getdata Property Sheet

Like the getdata popup menu, the getdata property sheet provides another way to respond to getdata prompts. Unlike the getdata popup, it lists all possible getdata modifiers (but no punctuation) in one screen location, without the "fanning" submenus. Because a property sheet is a menu function, a menu must be active to use the getdata property sheet.

6.5.1. Opening the Getdata Property Sheet

Press the right button while the cursor is in the menu window. The Menutools popup menu appears on the screen. Keeping the right button depressed, slide the cursor to the "Open Getdata" menu item. When it is highlighted, release the button. The getdata property sheet will appear in the lower right corner of the screen.

6.5.2. Selecting a Getdata Modifier

Select the modifier you need from the property sheet by positioning the cursor over your selection. When it is highlighted click (press and release) the left button. Your selection will appear on the command line. You can enter coordinate values with either the keyboard or the number pad on the bottom of the menu.

6.5.3. Closing the Getdata Property Sheet

When you finish working with the property sheet, slide the cursor into the menu window and press the right button to get the Menutools popup menu. With the right button depressed, slide the cursor to the "Close Getdata" selection until it is highlighted. Release the button. The property sheet and the Menutools popup disappear.

Ref: CADDS User Guide, 2-36.

CADDStation User Interface, 32,33.

7. Managing Files

7.1. Opening Files

7.1.1. Part

CADDS uses a part, a collection of files, to store and display the results of your work. Each part has one or more drawings, each a separate file. The processed output of a drawing file is the screen display you see in CADDS. To create a new part or to reopen a old part, use the command ACTIVATE PART followed by the part name. See CADDS Core Reference, ACTIVATE PART.

7.1.2. Drawing

Each part file has one or more drawing files with which it is associated. Drawings are made at the size you specify or at the default size (C-size), which is 37.4 CM by 48.4 CM or 17 IN by 22 IN. CADDS Also gives the drawing file the name you supply. To Create a drawing, use the command ACTIVATE DRAWING followed by the drawing name. See CADDS Core Reference, ACTIVATE DRAWING.

7.1.3. View

When a part and drawing files are open, you need to open a view into your drawing so you can see it. You define a view by specifying its origin and clipping window edges. Views are defined relative to the drawing's origin. To define a view, use the command DEFINE VIEW. See CADDS Core Reference, DEFINE VIEW.

7.1.4. Echo Frame

Frames are graphics that indicate the boundaries of drawings and views. Use the command ECHO FRAME to see areas where you can work.

Note: The icon "Open file" in the user menu on the page "Block", has a group of commands, which are needed to work in graphic environment. Only part name and drawing name are required.

Ref: CADDS Core Reference.

CADDS User Guide, 1-6 ~ 1-14, Section 3.

7.2. Closing Files

At the end of a session, you need to give up access to your part. EXIT PART dismisses an active part, leaving the task free in CADDS to use nongraphic commands or to activate another part. Use the command EXIT PART to either store or delete your part. There are several options for EXIT PART; the most used are F(File, to store a part) and Q(Quit, to delete a part). See CADDS Core Reference, EXIT PART.

Ref: CADDS Core Reference.

CADDS User Guide, 1-42.

8. Managing the CADDS Display

8.1. ZOOM DRAW

This command temporarily enlarges or reduces a portion of the drawing, changing the display. Zooming a drawing creates a transient state of the drawing not filed when the drawing is deactivated.

8.1.1. ZOOM DRAW WINDOW

Zooms and scrolls the drawing within a window defined by two opposite corner locations you select, so the window area fills the screen. Two locations for opposite corner are required.

8.1.2. ZOOM DRAW ALL

Resets the drawing from a zoomed state to its original state.

8.1.3. ZOOM DRAW ALLDRAW

In addition to performing the ZOOM DRAW ALL command, the modifier ALLDRAW repaints the drawing in both the CADDS 4X Graphics WINDOW and CADDS 4X Drawing Windows.

Ref: CADDS Core Reference.

CADDS User Guide, 3-11 ~ 3-49.

8.2. ZOOM VIEW and RESET VIEW

This command temporarily enlarges or reduces the display seen through one or more view windows within the active drawing. This command does not permanently alter a drawing. Use the command RESET VIEW to return a view to its prezoomed state.

8.2.1. ZOOM VIEW WINDOW

Use two diagonally opposite corners you supply to define a window within the view which CADDS enlarges to fill the view.

8.2.2. ZOOM VIEW ALL

Zooms and scrolls a specified view so that the entire model fits within the view clipping window. This modifier ALL represents model graphics that may have disappeared from the screen during scrolling or reduces extremely large models to fit within the view boundary.

8.2.3. RESET VIEW

Returns a view to the state it was in before zooming.

Ref: CADDS Core Reference.

CADDS User Guide, 3-11 ~ 3-49.

8.3. SCROLL DRAW

This command scrolls the active drawing. Scrolling is a temporary movement of the drawing relative to the screen. This temporary state is not filed when the drawing is left. To return to the original drawing position, use ZOOM DRAW ALL.

Ref: CADDS Core Reference.

CADDS User Guide, 3-11 ~ 3-49.

8.4. SCROLL VIEW

This command temporarily repositions the view with respect to model space without changing the position of the view on the drawing.

Ref: CADDS Core Reference.

CADDS User Guide, 3-11 ~ 3-49.

8.5. DYNAMIC VIEW

This command enables interactive dynamics (zoom, scroll, or rotate) for one or two views according to speed values chosen by modifiers or by default. Once execution begins, CADDS enables the Image Control Unit (ICU) Tool, a collection of icons you use in manipulating the selected view(s).

The ICU Tool includes the following icons:

Reset	Ends command execution.
Zoom	Starts interactive zoom operation and displays related icons.
Scroll	Starts interactive scroll operation and displays related icons.
Rotate	Starts interactive rotate operation and displays related icons.
Original	Resets the selected view to its state when dynamics were enabled, working similar to the command RESET VIEW, except that it restores the view to the state it was in when the view was selected (not the RESET VIEW state).
Close	Changes the ICU tool display to its closed icon. Select the closed icon to reopen the tool.

Once you select an operation icon (e.g., zoom), CADDS displays the related icons. Selecting a related icon begins the operation in the manner indicated by the icon (e.g., Enlarge, Reduce, Depth, etc.). CADDS executes the operation (as defined by your use of modifiers before entering getdata) once for each selection of the icon you make. See CADDS Core Reference, DYNAMIC VIEW.

Ref: CADDS Core Reference.

CADDS User Guide, 3-11 ~ 3-49.

9. Managing Layers

Complex drawings in paper form often use tissue or acetate to construct a series of overlays, each containing a portion of the overall design. When viewed individually, each sheet or layer gives a partial picture of the design. Seen all at once, the composite of the information on all layers provides the complete, albeit congested depiction of the design. CADDS uses the concept of layers to allow you to group elements within your drawing by any association you wish. Also you can discriminate layer ranges by colour.

Echoing layers on or off allows you to tailor your display. Thus you see only the model graphics you wish to work with. Layering controls modification to the model in that you will only change what is displayed on the screen of your workstation.

9.1. SELECT LAYER

This command specifies the active construction layer for the current active drawing. The new layer can be selected by an integer in the range of 0-254, or by selecting an entity on the desired layer. See CADDS Core Reference, SELECT LAYER.

Ref: CADDS Core Reference.

CADDS User Guide, 4-12,4-13.

9.2. ECHO LAYER

This command enables you to choose the layers that are to be made visible (echoed) on the currently active drawing. A drawing must first be active. When first activated the default is set to layer 0. Use of ECHO LAYER does not change the active layer. Although at any given time only one layer can be the construction layer of an active drawing or view, any number of the allowable 255 (0-254) layers can be echoed by repainting the selected drawing or view with the layer to be echoed.

9.2.1. ECHO LAYER

Enter the desired layer number or numbers in a string with space in between (e.g., #n# ECHO LAYER 0 4 7 99) or in a 'FROM-TO' form (e.g., #n# ECHO LAYER 0 4-9 22 37-39). Not only will the layer requested echoed, but also the last layer activated.

9.2.2. ECHO LAYER ALL

All 255 layers are echoed or displayed.

9.2.3. ECHO LAYER INCL (INCLUDE)

The modifier INCLUDE will echo not only the layers listed after INCL modifier but also the currently visible layers.

9.2.4. ECHO LAYER EXCL (EXCLUDE)

All currently visible layers are echoed except for those listed after EXCL. A few unwanted layers can thus be unechoed instead of having to list every desired layer. See CADDS Core Reference, ECHO LAYER.

Ref: CADDS Core Reference.

CADDS User Guide, 4-12,4-13.

9.3. CHANGE LAYER

This command changes the layer associated with selected entities. All selected entities will be moved from their original layer to the specified layer. The layer on which an entity is located must be active or echoed (i.e., the entity must be visible) before that entity can be changed to another layer. See CADDS Core Reference, CHANGE LAYER.

Ref: CADDS Core Reference.

CADDS User Guide, 4-12,4-13.

9.4. COPY ENTITY

This command duplicates entities and copies them from one layer to one or more specified layers. Entities on several layers can be copied at the same time. See CADDS Core Reference, COPY ENTITY.

Ref: CADDS Core Reference.

CADDS User Guide, 4-12,4-13.

9.5. SELECT LDISCRIMINATION

This command creates a list of construction layer numbers whose entities are to be assigned to one of sixteen font types, one of two shades, or colours. This is done by picking a font type, a shade, or a colour, and then choosing a list of layers. Entities on those layers will exhibit the selected font type, colour, or shade. The purpose is to modify the line font, shading (intensity), or colour of the visible layers making one layer or group of layers readily distinguishable from other layers of groups. This command is

used within an active part. The selected discrimination is stored with the part when the part is filed and is in effect only for that part.

Use the command DISCRIMINATE LAYER to display layers in the new discrimination.

See CADDS Core Reference, SELECT LDISCRIMINATION, and DISCRIMINATE LAYER.

Ref: CADDS Core Reference.

CADDS User Guide, 4-12,4-13.

10. Plot Hard Copies

The display of your drawing on the screen is not very portable. Using a plotter to make a hard copy of the display gives you a version of your work that you can take with you.

Use the command PLOT BENSON. This command plots the active drawing on the Benson pen plotter, where the origin of the plot is at the right, underlying an x-axis that runs vertically upward and a y-axis that runs horizontally leftward.

* Modifiers

SCALE <x>

Applies the scale <x> to the drawing before plotting.

<x> is a positive decimal number, 1.0 by default.

Note: If your part and drawings are set by the icon "open file", your view scale is 0.5, unless you

define any view scale. So if you want to plot entities in actual size (scale 1.0), use modifier SCALE 2.

WINDOW

Select two locations on the drawing to be opposite corners of a plotting window. CADDS plots everything within the window. The default is the entire drawing.

ROTATE

Turns the drawing 90 degrees counterclockwise before plotting, making the x-axis run horizontally leftward and the y-axis vertically downward.

PEN1

PEN2

PEN3

Selects a pen numbered 1,2,3. Can be followed by a space and one or more numbers of layers for the pen (see the LAYERS modifier).

LAYERS <n,...,n>

Makes the pen of the preceding PEN<n> modifier plot the visible layers whose numbers are expressed in the given form.

* Note

After entering the command, getdata (if you use the modifier WINDOW, two opposite corner locations are required), and RETURN, you will get the message, "PLOT COMPLETE". Now, please follow the instructions.

1. Turn on the switch of the plotter. Turn off the switch "test" which is for self testing. Put pen(s) in right pen

holders and place pens at the right place depending on the modifiers, and press the "Auto" button.

2. Coming back to the computer, move the cursor to the shell window (outside of any CADDS window or Menu window). Press the right button to get the popup Suntools menu. Slide down the cursor to "New Shell" and release the right button. A new shell window appears. When the prompt "LUTD%" appears, type as follows:

```
LUTD% cd /usr/plotspool/vector
LUTD% ls
```

A file name (normally a group of numbers) appears.

```
LUTD% lpr -PBNSO -r -s -g filename
```

Ref: CADDS Core Reference, PLOT BENSON.

8.3. Page Help File

1. Help File for Page 'Block'

This Page Help File is to provide more information about every icon (except help icon) in the menu page 'Block'.

Block-4. Open File

This icon has a group of commands, which are needed to set up the graphic environment suitable to generate block patterns and adapt them using this user menu set. When you begin CADDS, you must select this icon to open files before you can commence your design.

Commands included are:

SELECT UNITS CM

ACTIVATE PART <part name>

SELECT TAG ON

SELECT DRAWING UNIT CM

SELECT DRAWING SIZE E

ACTIVATE DRAWING <draw name>

ECHO FRAME

ECHO TAG ON

DEFINE VIEW TOP CPLANE TOP SCALE 0.5 : X22Y17

Only a part name and a drawing name are required. Any name can be used for the part name or drawing name, as long as they are recognizable to you.

Block-5. Demonstration Program

The program represented by this icon is the demonstration program of the Pattern Making System. The demonstration program comprises five parts.

1. Block Patterns which can be generated by this system.
2. How to use this system to adapt block patterns to design patterns.
3. How to use this system to make production patterns.
4. Examples of pattern grading.
5. An example: Trousers

This shows the procedure included from generating the block pattern to produce graded patterns for a pair of trousers.

You can view either a part or the whole of the demonstration program.

Block-6. T-Block

The program represented by this icon is referenced T-Pl-B. This program has a group of commands which open the part file T-BLOCK and the drawing Dl, which stores some block patterns in order to show you examples.

If you are following the tutorial of the Pattern Making System, you will be given instructions as to how to use the part file T-BLOCK.

Block-7. MEASURE LENGTH

This command measures the length of a contour. A contour is a sequence of connected curves. The entities that can make up a contour include lines,arcs, conics, B-splines, Cpoles, strings and nodal lines. Successive pairs of curves in the sequence (first and second, second and third, etc) must either match at their endpoints or intersect.

A maximum of 300 entities can be included in a contour.

When you select this icon, you receive the 'MODEL ent' getdata prompt. Simply select the curves in the order that they form the contour, using the mouse. Move the mouse cursor onto the entity that you want to select, click (press and release) the left button. Repeat it until you have selected as many entities as you want to measure, then press RETURN to complete the command and execute it.

Block-8. REGENERATE GRAPHICS

This command will regenerate and redisplay the Temporary View File (TVF) of the active drawing. The TVF contains a set of graphic instructions for the Graphics Processing Unit (GPU). When a part is edited, unused spaces are left in the TVF. REGENERATE GRAPHICS will eliminate those spaces and redisplay the graphics.

Sometimes, after delete, translate, mirror or rotate entities, ghost images (faint dotted entities) remain. REGENERATE GRAPHICS will eliminate them.

Block-9. DIVIDE ENTITY MPROJ

This command subdivides selected entities, creating new entities of the same type as those selected. Valid entities are: arcs, B-splines, circles, conics (ellipse, parabolas, and hyperbolas), lines, and strings.

When you select this icon, you receive the 'MODEL ent' getdata prompt. Enter an entity to be divided, using the mouse. Move the mouse cursor onto the entity to be divided, click (press and release) the left button. Then press semicolon (;) to get a getdata prompt 'MODEL loc', which asks to enter a location to divide. Move the mouse cursor to the location, click the left mouse button. Press RETURN to complete the command and execute it.

Block-10. Bodice 1

This icon allows for the creation of a bodice block pattern. A block pattern is a foundation pattern constructed to fit an average figure. The designer uses a block pattern as a basis for making the pattern for a design. She/he may introduce style lines, tucks, gathers, pleats or drapes but still the basic fit of the pattern will conform to the block used.

Depends on the type of a garment, you can choose the type of a bodice block pattern. There are four options of bodice block patterns which are:

1. The Close Fitting Bodice Block

   The close fitting bodice block is quite a close fit to the figure. So this pattern can be the basis for a close fitting garment such as a close fit blouse, shirt or dress.

2. The Easy Fitting Bodice Block

The easy fitting bodice block has less dart shaping and more ease than the close fitting one. So this pattern can be the basis for an easy fitting style garment such as a loose fitting blouse.

3. The Dartless Bodice Block

The dartless bodice block is very similar to the easy fitting one, the only difference is that this block does not have any darts. So this pattern can form the basis of easy fitting designs where dart shaping would destroy the design. However the block is not suitable for women with pronounced bust shaping.

4. The Overgarment Block

The overgarment block can form the basis of a coat and a easy fitting jacket. You can choose either full dart shaping or less bust shaping (half dart shaping) depending on the design.

When you select one of the options you will need to answer several questions to produce the size you want. You can generate the block pattern for individual body measurements or by the size number of standard body measurements. See icon number 11 'Bodice 2' for more options of the bodice block pattern.

Block-11. Bodice 2

This icon has options for generating four different types of bodice block pattern. A block pattern is a foundation pattern constructed to fit an average figure. The designer

uses a block pattern as a basis for making the pattern for a design. She/he may introduce style lines, tucks, gathers, pleats or drapes but still the basic fit of the pattern will conform to the block used.

Four options of bodice block patterns are:

1. The Tailored Jacket Block

   This pattern can form the basis of a jacket with rever collars. You can choose either full dart shaping or less bust shaping (half dart shaping) depending on the design of the garment.

2. The Classic Shirt Block

   This pattern is a block pattern for a classic shirt. Hence it can form the basis of a variety of shirt patterns. Also you can generate shirt sleeves and shirt collars using other icons in this page.

3. Block Pattern For Jersey Wear

   This pattern can be used as a base pattern for jersey wear such as tee shirts.

4. Block Pattern For Knitwear

   This pattern can be used as a base pattern for knitwear.

When you select one of the options you will need to answer several questions to produce the size you want. You can generate the block pattern for individual body measurements or by the size number of standard body measurements.

See icon number 10 'Bodice 1' for more options of the bodice block pattern.

Block-12. Sleeve

This icon has options for generating four different types of sleeve block pattern. A block pattern is a foundation pattern constructed to fit an average figure. The type of sleeve you choose will depend on the style of garment required.

The four options of sleeve block patterns are:

1. The One-Piece Sleeve Block

   This pattern is a close fitting one-piece sleeve block. Without any special design requirement, this one-piece sleeve block can form the basis of any sleeve.

2. The Two-Piece Sleeve Block

   This pattern can form the basis of a two-piece sleeve such as a jacket sleeve or coat sleeve.

3. The Easy Fitting Sleeve Block

   This pattern is the easy fitting one-piece sleeve block. You can decide the sleevehead height (crown height) so that you can change the width of the sleeve. This sleeve pattern can form the basis for a sleeve of an easy fitting garment such as sports wear, shirts or blouses.

4. The Shirt Sleeve

   This pattern can form the basis of a shirt sleeve. You can change the width of the cuff. The sleevehead height (crown height) is 1/4 of the armhole.

When you select one of the options you will need to answer several questions to produce the size you want. You can generate the block pattern for individual body measurements or by the size number of standard body measurements.

You have to enter the length of the armhole of the bodice block while you run the program, because to generate the sleeve pattern the armhole length is needed. Follow the instructions with patience.

Block-13. Skirt

This icon has options for generating four different types of skirt pattern.

The four options for skirt patterns are:

1. The Tailored Skirt Block

   This is a close fitting skirt block pattern which can form the basis for skirts of various designs. Introducing style lines, tucks, gathers, pleats or drapes to this skirt block pattern, you can generate many different style of skirt patterns including a straight skirt, a paneled skirt, a straight skirt with some pleats such as box pleats or inverted pleats, a gored skirt and a flared skirt.

2. The Circular Skirt Pattern

   This pattern is for a circular skirt. The circular skirt pattern can be generated for a full circular skirt (360 degree), or for a half circular skirt (180 degree).

3. The Pleated Skirt Pattern

   This pattern is for a skirt with allround pleats. The pleat shape can be determined by either pleat width or pleat number. A pleat width must be between 1.27 CM and 7.62 CM (0.5 IN and 3 IN), and the pleat number must be a number between 10 and 60.

4. The Gathered Skirt Pattern

This pattern is for a gathered skirt. You can generate either a slightly gathered skirt pattern which has a waist curve for better fitting or a very gathered skirt pattern which is a rectangle.

When you select one of the options you will need to answer several questions to produce the size you want. You can generate the block pattern for individual body measurements or by the size number of standard body measurements.

Block-14. Trouser

This icon has options for generating three different types of trouser pattern.

The three options for trouser patterns are:

1. The Basic Trouser Block

This pattern is for a simple pair of trousers. By adapting this pattern, moving darts, giving more ease, introducing style lines or drape you can generate trouser patterns of many different styles.

2. The Easy Fitting Trouser Block

This pattern has more ease than the basic trouser pattern. So this pattern can form the basis of baggy trousers and dungarees.

3. The Culottes pattern

This pattern can form the basis of culottes of various designs.

When you select one of the options you are asked a few questions to determine the size you want. You can generate a

trouser pattern to fit your individual body measurements or for a specific size from standard body measurements.

Block-15. Dress

This icon has options for generating six different types of dress block pattern. The first three options are for generating a one-piece dress block and the other three options are for generating a two-piece dress block.

A block pattern is a foundation pattern constructed to fit an average figure. The designer uses a block pattern as the basis for making the pattern for a design. She/he may introduce style lines, tucks, gathers, pleats or drapes but still the basic fit of the pattern will conform to the block used. Depending on the type of a garment, you can choose the type of a dress block pattern.

When you select one of the options you are asked a few questions to determine the size you want. You can generate a dress pattern to fit your individual body measurements or for a specific size from standard body measurements.

Six options of dress block patterns are:

1. The Close Fitting One-Piece Dress Block

This pattern is made by extending the close fitting bodice block to the finished length of a dress, and the close fitting bodice block is quite a close fit to the figure. Also you have a choice about the waist shaping whether you want waist darts and the side seam close fit at the waist line or not.

This pattern can form the basis of an one-piece dress of various design if the dress is close fit at the bodice part.

2. The Easy Fitting One-Piece Dress Block

This pattern is made by extending the easy fitting bodice block to the finished length of a dress, and the easy fitting bodice block has less dart shaping and more ease than the close fitting one. Also you have a choice about the waist shaping whether you want the side seam to be a close fit at the waist line or not.

This pattern can form the basis of an easy fitting style one-piece dress such as a shift dress.

3. The Dartless One-Piece Dress Block

This pattern is made by extending the dartless bodice block to the finished length of a dress. The dartless bodice block is very similar to the easy fitting one, the only difference is that this block does not have a dart. Also you have a choice about the waist shaping whether you want the side seam close fit at the waist line or not.

This pattern can form the basis of an easy fitting style one-piece dress where dart shaping would destroy the design. However this block is not suitable for women with pronounced bust shaping.

4. The Close Fitting Two-Piece Dress Block

This block is made of two parts, which are the close fitting bodice block and the close fitting skirt block.

This block can form the basis of a two-piece dress of various designs if the dress is close fit at the bodice and waist.

5. The Easy Fitting Two-Piece Dress Block

This block is made of two parts, which are the easy fitting bodice block and the easy fitting skirt block. The bodice part has less dart shaping and more ease than the close fitting one, and at the waist line both bodice and skirt pattern are loose fit.

This block can form the basis of a two-piece dress of various design if the dress is easy fit.

6. The Dartless Two-Piece Dress Block

This block is made of two parts, which are the dartless bodice block and the easy fitting skirt block. The bodice part has as much ease as the easy fitting one without any dart shaping, and at the waist line both bodice and skirt pattern are loose fit.

This block can form the basis of a loose fitting two-piece dress where dart shaping would destroy the design. But this block is not suitable for women with pronounced bust shaping.

Block-16. Changing Page to 'Detail'

This icon changes a submenu page to 'Detail'.

The icons (commands) on the submenu page 'Detail' are:

1. User Helpfile	2. System Helpfile
3. Page Helpfile	4. MEASURE LENGTH
5. MEASURE DISTANCE	6. MEASURE ANGLE
7. TRANSLATE ENTITY	8. TRANSLATE ENTITY COPY

9. DIVIDE ENTITY MPROJ	10. MIRROR ENTITY
11. MIRROR ENTITY COPY	12. Highlight (DISCRIMINATE ENT)
13. Shirt Cuff	14. Frilled Cuff
15. Waistband	16. Standing Collar
17. Buttonhole(s)	18. Point
19. Inside Pocket	20. Patched Pocket
21. Line	

When you select this page changing icon, no command appears in the text window, only the menu page changes and new icons appear in the menu window. Then you can select each icon on the new submenu page.

Block-17. Changing Page to 'Adapt'

This icon changes a submenu page to 'Adapt'.

The icons (commands) on the submenu page 'Adapt' are:

1. User Helpfile	2. System Helpfile
3. Page Helpfile	4. Tutorial
5. MEASURE LENGTH	6. MEASURE DISTANCE
7. CONSTRUCT GROUP	8. DISASSOCIATE GROUPS
9. REGENERATE GRAPHICS	10. Grain Line
11. Notch Point	12. Highlight (DISCRIMINATE ENT)
13. Seam Allowance	14. GENERATE OFFSET D
15.	16. Translate, Rotate or Mirror
17. Trim or Divide Ent	18. Point
19. Line	20. B-spline
21. Circle	

When you select this page changing icon, no command appears in the text window, only the menu page changes and new icons

appear in the menu window. Then you can select each icon on the new submenu page.

Block-18. Changing Page to 'Grade'

This icon changes a submenu page to 'Grade'.

The icons (commands) on the submenu page 'Grade' are:

1. User Helpfile	2. System Helpfile
3. Page Helpfile	4. MEASURE LENGTH
5. MEASURE DISTANCE	6. REGENERATE GRAPHICS
7. Grade Bodice Pattern	8. Grade Skirt Pattern
9. Grade Trouser Pattern	10. Grade Sleeve Pattern
11. Grade Collar Pattern	12.
13. Grade Height	14. Grade Horizontal Direction
15. Grade Vertical	16. Translate, Rotate or Mirror
17. Trim or Divide Entity	18. Point
19. Line	20. B-spline

21. INSERT TEXT

When you select this page changing icon, no command appears in the text window, only the menu page changes and new icons appear in the menu window. Then you can select each icon on the new submenu page.

Block-19. Changing Page to 'Appear'

This icon changes a submenu page to 'Appear'.

The icons (commands) on the submenu page 'Appear' are:

1. User Helpfile	2. System Helpfile
3. Page Helpfile	4. CHANGE LAYER
5. COPY ENTITY TO	6. SELECT LAYER
7. ECHO GRID ON NOSNAP	8. ECHO GRID OFF

9. ECHO GRID ON SNAP
10. INSERT TEXT $$
11. $ (Breaks a text)
12. $$ (Ends a text string)
13. : (execute)
14. REGENERATE GRAPHICS
15. Highlight
16. Text Font
17. Text Height
18. Text Angle in Degrees
19. Character Slant
20. Text Justification
21. CHANGE APPEARANCE FONT FROM ANY TO

When you select this page changing icon, no command appears in the text window, only the menu page changes and new icons appear in the menu window. Then you can select each icon on the new submenu page.

Block-20. Changing Page to 'Page e'

This icon changes a submenu page to 'Page e'. No command or program is connected to any icon on this submenu page.

Block-21. Changing Page to 'Page f'

This icon changes a submenu page to 'Page f'. No command or program is connected to any icon on this submenu page.

2. Help File for Page 'Detail'

This Page Help File is to provide more information about every icon (except help icon) in the menu page 'Detail'.

Detail-4. MEASURE LENGTH

See "Block-7. MEASURE LENGTH" in the Page Help File "Block".

Detail-5. MEASURE DISTANCE

This command measures the following:

* The minimum distance and angle between two lines.
* The minimum distance between two locations or between two point entities.
* The minimum distance between a location and a point, between a location and a line, or between a point and a line.
* B-spline and Cpoles.

The default system prompt is 'MODEL ent'. With this prompt you can select only lines, Nlines, points, Gpoints and Cnodes. However, you can override this prompt with getdata modifiers to do simple location specification on any entity that has an end, origin, or vertex.

Select entities or a location, using the mouse. Move the mouse cursor onto the entity or location which you want to select, click (press and release) the left button. Repeat it until you have finished your input, and press RETURN to complete the command and execute it.

Detail-6. MEASURE ANGLE

This command measures the angle between two intersecting lines. The system prints the acute angle as well as the obtuse angle formed by the two lines. The angle is given in three ways:

* Radians
* Degrees
* Degrees, minutes and seconds

When you select this icon, you receive a 'MODEL ent' getdata prompt. Move the mouse cursor onto a line, click (press and release) the left mouse button, and repeat for the other line.

Detail-7. TRANSLATE ENTITY

This command translates (moves) an entity or group of entities. You can translate entities by selecting them or by specifying a window using the getdata modifier WIN or PWIN.

When you select this icon, you receive a 'MODEL ent' getdata prompt. If there are not too many entities to be translated, move the cursor onto an entity, click the left mouse button. Repeat it until all the entities to be translated are entered. Then press semicolon (;) to get a getdata prompt 'MODEL loc', which requests two locations to be defined for the start and end of the translation.

When you want to translate many entities in a particular area, use the getdata modifier WIN (Window) or PWIN (Polygon Window). Using WIN you can select entities by specifying the diagonally opposite corner of a rectangle enclosing entities to be translated. If entities are not in a rectangle, using PWIN you can select entities by specifying the vertices of a polygon enclosing entities to be translated. Then press semicolon (;) to get a getdata prompt 'MODEL loc', which asks for two locations to define the start and end of the translation.

Detail-8. TRANSLATE ENTITY COPY

This command translates (moves) a COPY of an entity or group of entities. You can translate entities by selecting them or by specifying a window using the getdata modifier WIN or PWIN.

When you select this icon, you receive 'MODEL ent' getdata prompt. If there are not too many entities to be translated, move the cursor onto an entity, click the left mouse button. Repeat it until all the entities to be translated are entered. Then press semicolon (;) to get a getdata prompt 'MODEL loc', which requests two locations to be defined for the start and end of the translation.

When you want to translate many entities in a particular area, use the getdata modifier WIN (Window) or PWIN (Polygon Window). Using WIN you can select entities by specifying the diagonally opposite corner of a rectangle enclosing entities to be translated. If entities are not in a rectangle, using PWIN you can select entities by specifying the vertices of a polygon enclosing entities to be translated. Then press semicolon (;) to get a getdata prompt 'MODEL loc', which asks for two locations to define the start and end of the translation.

Detail-9. DIVIDE ENTITY MPROJ

See "Block-9. DIVIDE ENTITY MPROJ" in the Page Help File "Block".

Detail-10. MIRROR ENTITY

This command mirrors entities. This is done by reflecting the entity(s) around an axis (mirror plane). The mirror plane can be created by selecting two locations.
When you select this icon, you receive a 'MODEL ent' getdata prompt. If there are not too many entities to be mirrored, move the cursor onto an entity, click the left mouse button. Repeat it until all the entities to be mirrored are entered. Then press semicolon (;) to get a getdata prompt 'MODEL loc', which requests two locations to define the mirror plane.
When you want to mirror many entities in a particular area, use the getdata modifier WIN (Window) or PWIN (Polygon Window). Using WIN you can select entities by specifying the diagonally opposite corner of a rectangle enclosing entities to be mirrored. If entities are not in a rectangle, using PWIN you can select entities by specifying the vertices of a polygon enclosing entities to be mirrored. Then press semicolon (;) to get a getdata prompt 'MODEL loc', which asks for two locations to define the mirror plane.

Detail-11. MIRROR ENTITY COPY

This command mirrors and COPIES entities. This is done by reflecting the entity(s) around an axis (mirror plane). The mirror plane can be created by selecting two locations.
When you select this icon, you receive a 'MODEL ent' getdata prompt. If there are not too many entities to be mirrored, move the cursor onto an entity, click the left mouse button. Repeat it until all the entities to be mirrored are entered.

Then press semicolon (;) to get a getdata prompt 'MODEL loc', which requests two locations to define the mirror plane.

When you want to mirror many entities in a particular area, use the getdata modifier WIN (Window) or PWIN (Polygon Window). Using WIN you can select entities by specifying the diagonally opposite corner of a rectangle enclosing entities to be mirrored. If entities are not in a rectangle, using PWIN you can select entities by specifying the vertices of a polygon enclosing entities to be mirrored. Then press semicolon (;) to get a getdata prompt 'MODEL loc', which asks for two locations to define the mirror plane.

Detail-12. Highlight (DISCRIMINATE ENTITY ON)

This command temporarily highlights specified entities.

When you select this icon, you receive a -> prompt and the system pauses after -> prompt so you can enter a modifier to specify the discrimination. Available modifiers are HIGHLIGHT, RED, BLUE or GREEN. You can enter it from the keyboard. Enter a space to terminate your input.

When you receive a getdata prompt 'MODEL ent', move the mouse cursor onto an entity to be discriminated, click (press and release) the left mouse button. Repeat it until every entity to be discriminated is entered.

Detail-13. Shirt Cuff

This icon generates a pattern for a shirt cuff in the measurements you specify. The cuff length, the cuff depth

and the underwrap length are needed to generate it. The underwrap length is the amount of overraping for fastening. When you are asked to enter a shirt cuff length, refer to the table of "Shirt cuff sizes for women". Enter the cuff length. There is no default value for the cuff length, so if you do not enter the cuff length and press RETURN nothing can be generated because the cuff length is set to 0.

However for the cuff depth and the underwrap length, there are default values. The default value of the cuff depth is 5 CM (2 IN), and the default value of the underwrap is 1.5 CM (0.6 IN). When you are asked to enter one of these values, enter the value for that part, but if you press RETURN without entering the value, the default value is set to that variable.

Detail-14. Frilled Cuff

This icon generates a pattern for a frilled cuff in the measurements you specify. The cuff length and the cuff depth are needed to generate it.

Because a frilled cuff has to be fit to the sleeve pattern, you have to measure the length of the cuff at the sleeve pattern. When you get the instruction to measure length, follow the instruction with patience. Then enter the cuff length. There is no default value for the cuff length, so if you don't enter the cuff length and press RETURN nothing can be generated because the cuff length is set to 0.

The default value of the cuff depth is 5 CM (2 IN). When you are asked to enter the cuff depth, enter it as you want, but

if you press RETURN without entering the cuff depth, the default value is set to it.

Detail-15. Waistband

This icon generates a pattern for a waistband by the measurements you specify. The waist size, the waistband depth and the underwrap length are needed to generate it. When you are asked to enter a waist size, refer to the table of "Waist sizes for women". Enter the waist size (not the size number). There is no default value for the waistband, so if you don't enter the waist size and press RETURN nothing can be generated because the waist size is set to 0. However for the waistband depth and the underwrap length, there are default values. The default value of the waistband depth is 3 CM (1.2 IN), and the default value of the underwrap is 4 CM (1.7 IN). When you are asked to enter one of these values, enter the value for that part, but if you press RETURN without entering the value, the default value is set to that variable.

Detail-16. Standing Collar

This icon has options for generating four different designs of standing collar pattern.

The four options of collar patterns are:

1. The Standing Straight Collar

   The half neck measurement, the collar width and the buttonstand width are needed to generate the pattern of the standing collar. When you select this option, you

receive instructions to measure the neck at the bodice pattern and enter that measurement. Follow instructions with patience. There is no default value for the neck measurement.

The default value for the collar width is 5 CM (2 IN), and the default value for the buttonstand is 1.5 CM (0.6 IN). When you are asked to enter one of these values, enter the value for that part, but if you press RETURN without entering the value, the default value is set to that variable.

2. The Shirt Collar

The shirt collar is made of two parts, collar and stand. You can generate the collar and the stand either in one piece or in two pieces.

When you select this option, you get instructions to measure the neck at the bodice pattern and enter that measurement. Follow instructions with patience. There is no default value for the neck measurement.

The half neck measurement, the collar width, the stand depth and the buttonstand width are needed to generate the pattern for the shirt collar.

The default value for the collar width is 5 CM (2 IN), the default value for the stand depth is 3 CM (1.2 IN), and the default value for the buttonstand width is 1.5 CM (0.6 IN). When you are asked to enter one of these values, enter the value for that part, but if you press RETURN without entering the value, the default value is set to that variable.

3. The Convertible Collar

The half neck measurement and the collar width are needed to generate the pattern for the convertible collar.

When you select this option, you receive instructions to measure the neck at the bodice pattern and enter that measurement. Follow instructions. The default value for the collar width (including folded part) is 9 CM (3.5 IN). When you are asked to enter the collar width, enter it as you want, but if you press RETURN without entering it, the default value is set to collar width.

4. The Polo Collar

The half neck measurement and the collar width are needed to generate the pattern for the polo collar.

When you select this option, you get instructions to measure the neck at the bodice pattern and enter that measurement. Follow the instructions. The default value for the collar width is 4 CM (1.6 IN). When you are asked to enter the collar width, enter it as you want, but if you press RETURN without entering it, the default value is set to collar width.

Detail-17. Buttonhole(s)

This icon has 4 options for marking buttonholes on any pattern where buttonholes are needed. The 5th option is to place buttons on the button placement line.

Four options for marking buttonholes are:

1. Horizontal Worked Buttonholes
2. Horizontal Bound Buttonholes

3. Vertical Worked Buttonholes

4. Vertical Bound Buttonholes

All the above four options require the number of buttonholes, the button diameter and the button thickness, which are needed to work out the buttonhole size. Also you have to enter the first and the last buttonhole place on the buttonhole placement line.

The length of the worked buttonhole is the sum of the button diameter and the button thickness plus 0.3 CM, and the length of the bound buttonhole is the sum of the button diameter and the button thickness.

Detail-18. Point

This icon has 5 options for inserting or generating point(s).

1. Location (INSERT POINT : MODEL loc d)

   Inserts point at the location selected.

2. On Entity (INSERT POINT ON : MODEL ent d MODEL loc d)

   Inserts a point on a selected entity. The point is created by dropping a perpendicular from a reference location to the entity.

3. Number (GENERATE POINT N -> n : MODEL ent d)

   The number of points to be inserted along the contour is the value of 'n'. 'n' points are evenly distributed along the curve.

4. Distance, Entity (GENERATE POINT DIST -> x : MODEL ent d)

   Specifies the distance between the first two points. The command places as many specified points on the curve at distance 'x' as can fit.

5. Origin (GENERATE POINT ORG : MODEL ent d)

Inserts points at the segment endpoints of B-splines, strings and nodal lines. The use of ORG excludes the use of other modifiers.

Detail-19. Inside Pocket

This icon has 3 options to mark the opening of a inside pocket on any pattern where an inside pocket will be made, and at the same time generate pattern pieces which are needed to make the inside pocket.

The three different types of inside pockets are:

1. The Flap Pocket

The flap depth, 2 locations to define 2 upper ends of the flap and a location to define the bottom of the pocket are required.

2. The Welt Pocket

The welt depth, 2 locations to define 2 upper ends of the welt and a location to define the bottom of the pocket are required.

3. The Slashed Pocket

The depth of one lip of the opening (in CM), 2 locations to define 2 upper ends of the slashed pocket opening and a location to define the bottom of the pocket are required.

In the end you receive the instruction "Select the position for the left lower corner of pocket pattern please." Select the location where you want to generate actual pattern pieces to make the pocket.

Detail-20. Patch Pocket

This icon has 4 options to mark the place of a patch pocket on any pattern where the patch pocket will be attached, and at the same time generate pattern pieces which are needed to make the patch pocket.

The three different shape of patched pockets are:

1. Square
2. Chop-Corner
3. Round-Corner
4. Square-V

In addition every option has three choices of flap, they are no flap or self-flap or separate flap.

You can define the location of the patch pocket either by the width and depth of the pocket and the location of one corner or by diagonal corners of the pocket.

In the case of making a flap, there are two ways to define the flap shape. one is by flap depth and the other is by digitizing the lowest position of the flap.

Detail-21. Line

This icon has 7 options to insert line(s).

1. 2 Locations (INSERT LINE : MODEL loc dd)

   Lines are inserted between 2 locations selected. Lines are not inserted into the part until a colon (:) or RETURN is entered.

2. Horizontal (INSERT LINE HOR : MODEL loc dd)

   Creates a horizontal line. The first location selected is the start of the line. The second location indicates the direction and distance along the horizontal axis.

3. Horizontal, Length (INSERT LINE HOR LNG x : MODEL loc dd)

Creates a horizontal line of a specified length 'x'. The first location selected is the start of the line. The second location indicates the direction of the line for a distance of 'x'.

4. Vertical (INSERT LINE VER : MODEL loc dd)

Creates a vertical line. The first location selected is the start of the line. The second location indicates the direction and distance along the vertical axis.

5. Vertical, Length (INSERT LINE VER LNG x : MODEL loc dd)

Creates a vertical line of a specified length 'x'. The first location selected is the start of the line. The second location indicates the direction of the line for a distance of 'x'.

6. Perpendicular (INSERT LINE PERP : MODEL ent d MODEL loc dd)

Creates a line perpendicular to an existing entity. The first location selection identifies the line used as the perpendicular reference. The second location selected is the start of the line, and the third location indicates the direction and distance of the new line.

7. Parallel (INSERT LINE PAR x : MODEL ent d MODEL loc dd)

Creates a line parallel to an existing line. The optional value 'x' causes the line to be created at x units distance from the existing line. The first location selection identifies the line used as the parallel reference. The second location selected is the start of the line, and the third location indicates the direction and distance of the new line.

3. Help File for Page 'Adapt'

This Page Help File is to provide more information about every icon (except help icon) in the menu page 'adapt'.

Adapt-4. Tutorial

The program for this icon is T-P1-D1, which opens the part file T-ADAPT and demonstrates to you how to adapt block patterns to make design patterns.
There are two identical patterns, one is for demonstrating and the other is for your practice. If you follow the tutorial for the Pattern Making System, you will be given detailed instructions regarding practice.

Adapt-5. MEASURE LENGTH

See "Block-7. MEASURE LENGTH" in the Page Help File "Block".

Adapt-6. MEASURE DISTANCE

See "Detail-5. MEASURE DISTANCE" in the Page Help File "Detail".

Adapt-7. CONSTRUCT GROUP

This command creates groups of separate entities. Any number of entities can be combined into a group. The purpose of creating groups is to provide a means of quickly identifying

entities within getdata. All members within a group may be manipulated as a single entity by using the getdata modifier GRO (group).

Entities belonging to a group cannot be individually deleted until the group is dissociated using DISSOCIATE GROUP command.

When you select this icon, you receive the 'MODEL ent' getdata prompt. If there are not too many entities in a group, move the cursor onto an entity, click the left mouse button. Repeat it until all wanted entities are entered. Press RETURN to execute the command.

When you want to construct a group with many entities in a particular area, use the getdata modifier WIN (Window) or PWIN (Polygon Window). Using WIN you can select entities by specifying the diagonally opposite corner of a rectangle enclosing entities to make a group. If the entities are not in a rectangle, using PWIN you can select entities by specifying the vertices of a polygon enclosing entities to make a group.

Adapt-8. DISASSOCIATE GROUP

This command is used to dissociate the association among entities set up by the command CONSTRUCT GROUP.

Only one member in a group has to be selected in getdata to dissociate that group. If an entity belongs to more than one group, the command dissociates only the first group it finds that is identified with that entity. More than one group may be dissociated at a time by selecting one entity for each group.

Entities associated with groups may not be deleted until the group is dissociated.

Adapt-9. REGENERATE GRAPHICS

See "Block-8. REGENERATE GRAPHICS" in the Page Help File "Block".

Adapt-10. Grain Line

This icon inserts grain lines. The grain line indicates the lengthwise direction of a fabric, which is the direction of warp yarns. This gives an axis for alignment of the pattern along or across the lay in marker making.

When you generate block patterns using this system, block patterns already have grain lines. But when you adapt block patterns to make style patterns, you might cut pattern pieces into many pieces or rotate them or put together two pattern pieces which have different grain lines. So you have to insert grain lines into new pattern pieces, or change grain lines depending on the design.

You can insert a vertical, horizontal or bias grainline (45 degrees or 135 degrees counterclockwise from the horizontal line). Also a grain line can be inserted between any 2 locations you enter.

Adapt-11. Notch Point

This icon inserts notch points (small circle shape). Mostly they are used as a matching point between different pattern pieces.

When you generate block patterns using this system, sometimes block patterns already have notch points. But when you adapt block patterns to make style patterns, you might cut pattern pieces into many pieces or rotate them or put together many patterns into one piece. Then you need to insert new notch points because matching points are changed.

When you select locations for notch points, you may use getdata modifiers "POI", "END" or "ORG". For example, if you want to insert a notch point at the same location of a point you can use the modifier "POI", or if you want to insert a notch point at the end of a curve you can use the modifier "END". See System helpfile, 6. Getdata. When you want to use a getdata modifier, enter that modifier following the prompt "Enter the locations for notch points up to 30 :" and select locations using the mouse.

Adapt-12. Highlight (DISCRIMINATE ENTITY ON)

See "Detail-12. Highlight (DISCRIMINATE ENTITY ON)" in the Page Help File "Detail".

Adapt-13. Seam Allowance

The command for this icon is CONSTRUCT OFFSET D. This command creates an offset of an entity such as a point, line, arcs, circle, conic, string, B-spline. An offset is a copy of an entity (or part of an entity) that is created at a specified distance from the original.

When you select this icon, you receive a -> prompt after a modifier D and the system pauses after -> prompt so you can

enter a distance of the offset from the original entity. In order to give a seam allowance, you have to enter the amount of the seam allowance in cm following a -> prompt.

When the getdata prompt 'MODEL ent' appears, enter the original entity. Then the getdata prompt 'MODEL loc' appears to ask a location which indicates the direction (left or right) of the offset, enter a location using the mouse.

When constructing an offset using lines and arcs, also refer to the GENERATE OFFSET command.

Adapt-14. GENERATE OFFSET D

This command creates an offset of lines and arcs by a constant distance, and clips (or extends) new lines and arcs at (to) intersection points. Up to 10,000 entities may be selected.

When you select this icon, you receive a -> prompt after a modifier D and the system pauses after -> prompt so you can enter a distance for the offset from the original entity. There is no default value for a distance. You can enter a distance for the offset from the keyboard. When a getdata prompt 'MODEL ent' appears, enter the original entities. Then press semicolon (;) to finish selecting entities and to get another getdata prompt 'MODEL loc' which asks a location for the direction (left or right) of the offset. Enter a location.

This command is similar to the CONSTRUCT OFFSET command. While the CONSTRUCT OFFSET command has a more limited

functionality, it can operate on a greater variety of entities including surfaces.

Adapt-15. Icon No. 15

This icon has no function because no command or program is connected with the icon.

Adapt-16. Translate, Rotate or Mirror

This icon has 7 options for moving (translate, rotate or mirror) entities.

1. Translate (TRANSLATE ENTITY : MODEL ent ddd ; MODEL loc dd)

   This command translates an entity or a group of entities. After the getdata prompt 'MODEL ent', you can select entities using the mouse. If many entities have to be selected, use the getdata modifier WIN or PWIN.

   When you have selected every entity to be translated, press the semicolon (;) to get a getdata prompt 'MODEL loc', which asks for two locations to be entered to define the start and end of the translation. Enter two locations.

2. Translate, Copy (TRANSLATE ENTITY COPY: MODEL ent ddd; MODEL loc dd)

   This command works in exactly the same way as TRANSLATE ENTITY, the only difference is that this command translates a COPY of the entity.

3. Rotate (ROTATE ENTITY : MODEL ent ddd ; MODEL loc ddd)

This command rotates entities. Up to 15,000 entities or grouped entities can be rotated with one command.

After the getdata prompt 'MODEL ent', you can select entities using the mouse. If many entities have to be selected, use the getdata modifier WIN or PWIN.

When you have selected every entity to be rotated, press the semicolon (;) to get a getdata prompt 'MODEL loc', which asks to enter three locations to define a rotate axis. Enter three locations; the first selection is the centre of the rotation axis, the second and third define the start and end of the rotation.

4. Rotate, Angle (ROTATE ENT ANGLE -> x : MODEL ent ddd ; MODEL loc d)

   This command rotates entities by the given angle around the z-axis counterclockwise.

   Entities to be rotated can be selected in the same way of using the command ROTATE ENTITY. Then you receive a getdata prompt 'MODEL loc', select one location for the centre of the rotation.

5. Mirror (MIRROR ENTITY : MODEL ent ddd; MODEL loc dd)

   This command mirrors entities. This is done by reflecting the entity around an axis (mirror plane).

   After getdata prompt 'MODEL ent', you can select entities using the mouse. If many entities have to be selected, use the getdata modifier WIN or PWIN. When you select every entity to be mirrored, press semicolon (;) to get a getdata prompt 'MODEL loc', which asks for two locations to define the mirror plane. Enter two locations which are at the ends of the mirror plane.

6. Mirror, Copy (MIRROR ENTITY COPY : MODEL ent ddd; MODEL loc dd)

   This command works exactly the same as MIRROR ENTITY, the only difference is that this command mirrors a COPY of the entity.

7. Mirror, Copy, Line

(MIRROR ENTITY COPY LINEPLANE : MODEL ent ddd; MODEL ent d)

This command works exactly the same as the MIRROR ENTITY COPY, the only difference is the way of defining the mirror plane. After you select entities to be mirrored, when you press semicolon (;), a getdata prompt 'MODEL ent' appears to ask to enter a line, which will be mirror plane. Enter the line.

Adapt-17. Trim or Divide Entity

This icon has 7 options to trim or divide entities.

1. Trim (TRIM ENTITY : MODEL ent d MODEL loc d)

   This command shortens or lengthens an entity such as lines, arcs, circles, conics, splines, B-splines ans strings.

   When the getdata prompt 'MODEL ent' appears, enter the entity to be trimmed, and when the next getdata prompt 'MODEL loc' appears, enter the location to trim.

2. Trim, Ilength (TRIM ENTITY ILENGTH -> x : MODEL ent d)

   This command trims (extends or shortens) a line by a value of <x>. A positive <x> shortens the line while a negative <x> lengthens the line. For example, if you

start with a line 10 units long and specify ILENGTH 2, the resulting line will be 8 units long. When you select the entity (line), select it at the location near to the end where you want to trim it.

3. Trim, Mintof

(TRIM ENTITY MINTOF: Select ents to be trimmed ddd; Trimming ents d)

This command trims up to 100 entities by one intersecting entity. The system prompts 'Select ents to be trimmed' requesting you to select up to 100 entities to be trimmed. After selection, enter a semicolon (;) and the system prompts 'Trimming ents' requesting you to select one intersecting entity to be used as the trimming entity. Enter the trimming entity.

4. Trim, Corner

(TRIM ENTITY CORNER : MODEL ent d By MODEL ent d)

This command creates corners for curves and strings by simultaneously trimming them at their points of intersection. The getdata prompts 'MODEL ent' requesting you to select an entity to be trimed to create a corner. Enter a entity. Then the prompt 'By MODEL ent' requests you to select a second entity to create a corner. Enter the second entity.

5. Divide, Mproj (DIVIDE ENTITY MPROJ : MODEL ent d MODEL loc d)

This command subdivides selected entities, creating new entities of the same type as those selected.

When getdata prompts 'MODEL ent', enter the entity to be divided. Then a prompt 'MODEL loc' appears, enter locations to divide. The system project the points of locations on the entity to be divided.

6. Divide,Intof(DIVIDE ENTITY INTERSECTION OF: MODEL ent d MODEL ent d)

This command directs the system to divide the selected entities wherever they intersect a second entity. The first entity selected is the one divided. The second selection identifies the entities that should intersect the first entities selected, the intersection(s) determining where the first is to be divided.

7. Divide, Ndiv (DIVIDE ENTITY NUMBER OF DIVISIONS -> n : MODEL ent d)

This command divides an entity into 'n' subdivided entities. When you select this icon, you receive a -> prompt and the system pauses after -> prompt so you can enter a number of division. Enter it by the keyboard. Then enter an entity to be divided.

Adapt-18. Point

See "Detail-18. Point" in the Page Help File "Detail".

Adapt-19. Line

See "Detail-21. Line" in the Page Help File "Detail".

Adapt-20. B-spline

This icon has 7 options to insert B-spline(s), which is a smooth curve connecting a series of locations. A B-spline is inserted by specifying three or more locations.

The differences between the 7 options of this icon are modifiers. Modifiers define the shape of the B-spline in different ways. Modifiers and their functions are:

* DEG n (DEGREE n)

  Specifies the degree of the B-spline, where 'n' is an integer from 2 to 7. The default is degree 3 (cubic B-spline). In general (except degree 2 and 3), the higher the degree, the smoother the curve. Two special cases exist:

  DEGREE 2 : Quadratic B-spline, ensures first degree derivative continuity (continuous tangency).

  DEGREE 3 : Cubic B-spline, ensures second degree derivative continuity (continuous tangency and continuous rate of change in slope).

* TANA

  Indicates a starting vector for the B-spline. When you select the option (command) having the modifier TANA, a getdata prompt 'MODEL loc' is displayed. Select two locations. The direction indicated between the first and second locations defines the beginning slope of the B-spline. The longer the distance between the two locations, the more the B-spline is pulled in that direction at its start.

* TANB

Indicates a ending vector for the B-spline. This modifier works similarly to the TANA modifier. The direction between the first and second locations defines the ending slope of the B-spline. The longer the distance between the two locations, the more the B-spline is pulled in that direction at its end.

The 7 different options are:

1. B-spline (INSERT BSPLINE : MODEL loc ddddddd)

   The simplest way of inserting a B-spline. When you get the getdata prompt 'MODEL loc', enter at least 3 locations.

2. Degree ? (INSERT BSPLINE DEGREE n : MODEL loc ddddddd)

   The degree has to be entered after the prompt ->. A degree is an integer from 2 to 7. When you get the getdata prompt 'MODEL loc' enter at least 3 locations.

3. Degree 3 (INSERT BSPLINE DEGREE 3 : MODEL loc ddddddd)

   The degree of a B-spline is 3. When you get the getdata prompt 'MODEL loc', enter at least 3 locations.

4. Degree 3 TANA

   INSERT BSPLINE DEGREE 3 TANA: MODEL loc dd MODEL loc ddddddd)

   The degree of a B-spline is 3. The B-spline starts by the vector defined by the first and second location selection. When you enter two locations you get another getdata prompt 'MODEL loc'. Enter at least 3 locations.

5. Degree 3 TANB

   (INSERT BSPLINE DEGREE 3 TANB: MODEL loc dd MODEL loc ddddddd)

The degree of a B-spline is 3. The B-spline ends at the vector defined by the first and second location selected. When you enter two locations you get another getdata prompt 'MODEL loc'. Enter at least 3 locations.

6. Degree ? TANA

(INSERT BSPLINE DEGREE n TANA: MODEL loc dd MODEL loc ddddddd)

The degree has to be entered after the prompt ->. Also the B-spline starts from the vector defined by the first and second location selected. When you enter two locations you get another getdata prompt 'MODEL loc'. Enter at least 3 locations.

7. Degree ? TANB

(INSERT BSPLINE DEGREE n TANB: MODEL loc dd MODEL loc ddddddd)

The degree has to be entered after the prompt ->. Also the B-spline ends at the vector defined by the first and second location selected. When you enter two locations you get another getdata prompt 'MODEL loc'. Enter at least 3 locations.

Adapt-21. Circle

This icon has 7 options to insert circle(s).

1. 3 Location (INSERT CIRCLE: MODEL loc ddd)

When you get the getdata prompt 'MODEL loc', select three locations defining the circumference of the circle.

2. 3 Entities (INSERT FILLET CIRCLE -TRIM THRENT: MODEL ent ddd)

When you get the getdata prompt 'MODEL ent', select three entities to define the circle. The circle is inserted to be tangent to three entities.

3. Diameter, Location (INSERT CIRCLE DIAMETER x : MODEL loc d)

The diameter of the circle has to be entered after the prompt ->. Enter a value for 'x'. When you get the getdata prompt 'MODEL loc' specify one location, the center of the circle.

4. Radius, Location (INSERT CIRCLE RADIUS x : MODEL loc d)

The radius of the circle has to be entered after the prompt ->. Enter a value for 'x'. When you get the getdata prompt 'MODEL loc' specify one location, the center of the circle.

5. 2 Entities, Radius

(INSERT FILLET CIRCLE -TRIM RADIUS x : MODEL ent dd)

The radius of the circle has to be entered after the prompt ->. Enter a value for 'x'. When you get the getdata prompt 'MODEL ent' specify two entities to which the circle will be tangent.

6. Tangent to (INSERT CIRCLE TANTO : MODEL loc d MODEL ent d)

This option creates a circle that is tangent to an existing entity. When you get the getdata prompt 'MODEL loc' select the location for the centre of the new circle. Then the next getdata prompt 'MODEL loc'

appears, select the entity that the circle will be tangent to.

7. Part -> Whole (INSERT CIRCLE ARC : MODEL ent d)

This option creates a circle from an existing arc. When you get the getdata prompt 'MODEL ent' select the arc.

4. Help File for Page 'Grade'

This Page Help File is to provide more information about every icon (except help icon) in the menu page 'grade'.

Grade-4. MEASURE LENGTH

See "Block-7. MEASURE LENGTH" in the Page Help File "Block".

Grade-5. MEASURE DISTANCE

See "Detail-5. MEASURE DISTANCE" in the Page Help File "Detail".

Grade-6. REGENERATE GRAPHICS

See "Block-8. REGENERATE GRAPHICS" in the Page Help File "Block".

Grade-7. Grade Bodice Pattern

This icon is for grading pattern pieces which make the bodice part of a garment.

First, the current size of the pattern is required. Enter a value which must be an even number between 8 and 30. Then enter up to 9 sizes which you want to grade. Enter values which must be even numbers between 8 and 30.

In this program the grade rule for the x-direction is a proportional increase or decrease of the x-direction distance from the zero point to each grading point, where the ratio is the bust measurement of each size to be graded to the bust measurement of the current size. Also the grade rule for the y-direction is a proportional increase or decrease of the y-direction distance from the zero point to each grading point, where the ratio is the nape to waist measurement of each size to be graded to the nape to waist measurement of the current size.

When you get the message "Wait a moment please." wait for a few seconds while layers are set to different sizes to distinguish one size from another. When you are asked to enter a location for the zero point, select the zero point using the mouse. Then you are asked to enter a location for the text. Select the location where you want to write the pattern size to prevent being confused.

Now you are going to grade lines, curves or points. At this stage the program knows the grade rule and zero point, so what you have to input are grading points and entity type. Available entity types are lines, B-spline (curve), points or notch point (small circle), you can select one of them. Depending on the entity type, there are some instructions, follow them carefully. When you get the prompt "Enter

locations to be graded :", enter grading points. Repeat this procedure until all entities are graded. Graded patterns appear in different colours.

Grade-8. Grade Skirt Pattern

This icon is for grading skirt pattern pieces. First, the current size of the pattern is required. Enter a value which must be an even number between 8 and 30. Then enter up to 9 sizes which you want to grade. Enter values which must be even numbers between 8 and 30.

In this program the grade rule for the x-direction is a proportional increase or decrease of the x-direction distance from the zero point to each grading point, where the ratio is the hip measurement of each size to be graded to the hip measurement of the current size. Also the grade rule for the y-direction is a proportional increase or decrease of the y-direction distance from the zero point to each grading point, where the ratio is the waist to knee measurement of each sizes to be graded to the waist to knee measurement of the current size.

When you get the message "Wait a moment please." wait for a few seconds while layers are set to different sizes to distinguish one size from another. When you are asked to enter a location for the zero point, select the zero point using the mouse. Then you are asked to enter a location for the text. Select the location where you want to write the pattern size to prevent being confused.

Now you are going to grade lines, curves or points. At this stage the program knows the grade rule and zero point, so what you have to input are grading points and entity type. Available entity types are lines, B-spline (curve), points or notch point (small circle), you can select one of them. Depending on the entity type, there are some instructions, follow them carefully.

When you get the prompt "Enter locations to be graded :", enter grading points. Repeat this procedure until all entities are graded. Graded patterns appear in different colours.

Grade-9. Grade Trouser Pattern

This icon is for grading trouser pattern pieces. First, the current size of the pattern is required. Enter a value which must be an even number between 8 and 30. Then enter up to 9 sizes which you want to grade. Enter values which must be even numbers between 8 and 30.

In this program the grade rule for the x-direction is a proportional increase or decrease of the x-direction distance from the zero point to each grading point, where the ratio is the hip measurement of each size to be graded to the hip measurement of the current size. Also the grade rule for the y-direction is a proportional increase or decrease of the y-direction distance from the zero point to each grading point, where the ratio is the waist to floor measurement of each size to be graded to the waist to floor measurement of the current size.

When you get the message "Wait a moment please." wait for a few seconds while layers are set to different sizes to distinguish one size from another. When you are asked to enter a location for the zero point, select the zero point using the mouse. Then you are asked to enter a location for the text. Select the location where you want to write the pattern size to prevent being confused.

Now you are going to grade lines, curves or points. At this stage the program knows the grade rule and zero point, so what you have to input are grading points and entity type. Available entity types are lines, B-spline (curve), points or notch point (small circle), you can select one of them. Depending on the entity type, there are some instructions, follow them carefully.

When you get the prompt "Enter locations to be graded :", enter grading points. Repeat this procedure until all entities are graded. Graded patterns appear in different colours.

Grade-10. Grade Sleeve Pattern

This icon is for grading sleeve pattern pieces. First, the current size of the pattern is required. Enter a value which must be an even number between 8 and 30. Then enter up to 9 sizes which you want to grade. Enter values which must be even numbers between 8 and 30.

When you get the message "Wait a moment please." wait for a few seconds while layers are set to different sizes to distinguish one size from another. In this program the

grade rule for the x-direction is a proportional increase or decrease of the x-direction distance from the zero point to each grading point, where the ratio is the armhole measurement at the bodice of each size to be graded to the same part measurement of the current size. So you have to grade the bodice before grading the sleeve patterns. There are instructions on how to measure and input these measurements, follow them carefully.

The grade rule for the y-direction is a proportional increase or decrease of the y-direction distance from the zero point to each grading point, where the ratio is the sleeve length of each size to be graded to the sleeve length of the current size.

When you are asked to enter a location for the zero point, select the zero point using the mouse. Then you are asked to enter a location for the text. Select the location where you want to write the pattern size pattern size to prevent being confused.

At this stage the program knows the grade rule and zero point, so what you have to input are grading points and entity type. Available entity types are lines, B-spline (curve), points or notch point, you can select one of them. Depending on the entity type, there are some instructions, follow them carefully.

When you get the prompt "Enter locations to be graded :", enter grading points. Repeat this procedure until all entities are graded. Graded pattern appears in different colours.

Grade-11. Grade Collar Pattern

This icon is for grading collar pattern pieces. First, the current size of the pattern is required. Enter a value which must be an even number between 8 and 30. Then enter up to 9 sizes which you want to grade. Enter values which must be even numbers between 8 and 30.

When you get the message "Wait a moment please." wait for a few seconds while layers are set to different sizes to distinguish one size from another.

In this program the grade rule for the x-direction is a proportional increase or decrease of the x-direction distance from the zero point to each grading point, where the ratio is the neck measurement (half) at the bodice of each size to be graded to the same part measurement of the current size. So you have to grade the bodice pattern before grading the collar patterns. There are instructions how to measure and these measurements, follow them carefully.

The grade rule for the y-direction is a proportional increase or decrease of the y-direction distance from the zero point to each grading point, where the ratio is the ratio of the nape to waist measurement of each size to be graded to the nape to waist measurement of the current size.

When you are asked to enter a location for the zero point, select the zero point using the mouse. Then you are asked to enter a location for the text. Select the location where you want to write the pattern size to prevent being confused.

At this stage the program knows the grade rule and zero point, so what you have to input are entity type and grading points. Available entity types are lines, B-spline (curve),

points or notch point, you can select one of them. Depending on the entity type, there are some instructions, follow them carefully.

When you get the prompt "Enter locations to be graded :", enter grading points. Repeat this procedure until all entities are graded. Graded pattern appears in different colours.

Grade-12. Icon No. 12

This icon has no function because no command or program is connected with the icon.

Grade-13. Grade Height

This icon is for grading pattern pieces only lengthwise (y-direction), so that you can grade patterns to short, medium and tall height having the same size number.

First you have to enter the pattern type, because each type of pattern is graded by a different grade rule. Available pattern types are bodice, skirt, trousers or sleeves. Enter the pattern type. Then the size of the pattern is required. Enter a value which must be an even number between 8 and 30. Now, you have to enter the current pattern height (one of short, medium, or tall), so that the system can grade patterns to the other two heights. For example, if the current height is medium, patterns will be graded to short and tall height, or if the current height is tall, patterns will be graded to short and medium.

When you get the message "Wait a moment please." wait for a few seconds while layers are set to different sizes to distinguish each size from others.

The bodice patterns are graded by the difference of the nape to waist measurement of each height (short, medium and tall), which is 2 CM. The skirt patterns are graded by the difference of the waist to knee measurement of each height, which is 3 CM. The trouser patterns are graded by the difference of the waist to floor measurement of each height, which is 5 CM. And the sleeve patterns are graded by the difference of the sleeve length of each height, which is 2.5 CM.

If the trouser patterns are graded, the grade rule is a proportional increase or decrease of the y-direction distance from the zero point to each grading point, where the ratio is the waist to floor measurement plus (in case of grading from medium to tall) or minus (in case of grading from medium to short) 5 CM to the waist to floor measurement. Patterns of other types are graded in the same way only different measurements such as the nape to waist measurement, the waist to knee measurement or the sleeve length are used instead of the waist to floor measurement.

When you are asked to enter a location for the zero point, select the zero point using the mouse. Then you are asked to enter a location for the text. Select the location where you want to write the pattern size to prevent being confused.

At this stage, the program knows the grade rule and zero point, so what you have to input are entity type and grading

points. Available entity types are lines, B-spline (curve), points or notch point, you can select one of them. Depending on the entity type, there are some instructions, follow them carefully.

When you get the prompt "Enter locations to be graded :", enter grading points. Repeat this procedure until all entities are graded. Graded pattern appears in different colours.

Grade-14. Grade Horizontal Direction

This icon is for grading pattern pieces only widthwise (x-direction). First, the current size of the pattern is required. Enter a value which must be an even number between 8 and 30. Then enter up to 9 sizes which you want to grade. Enter values which must be even numbers between 8 and 30.

When you get the message "Wait a moment please." wait for a few seconds while layers are set to different sizes to distinguish one size from another.

In this program the grade rule for the x-direction is a proportional increase or decrease of the x-direction distance from the zero point to the each grading point, where the ratio is some horizontal length (or distance) of each size to be graded to the same part measurement of the current size. For example, if you grade cuff pattern you might want to grade it only horizontal way. Then the grading ratio is the cuff length of each size to be graded to the cuff length of the current size. Or if you grade collar pattern in horizontal way only, the grading ratio is the

neck measurement at the bodice of each size to be graded to the same measurement of the current size.

There is a question whether you want to measure some length to define the grade rule or not. For example if you grade the cuff pattern, you might input cuff length from the size chart so you don't need to measure, but if you grade the collar pattern, you might want to measure the neck measurement of each size. Decide and either enter "M" to measure or press "RETURN". If you enter "M", you will get instructions, follow them carefully. Enter measurements to define grading ratio. When you are asked to enter a location for the zero point, select the zero point using the mouse. Then you are asked to enter a location for the text. Select the location where you want to write the pattern size to prevent being confused.

At this stage the program knows the grade rule and zero point, so what you have to input are entity type and grading points. Available entity types are lines, B-spline (curve), points or notch point, you can select one of them. Depending on the entity type, there are some instructions, follow them carefully.

When you get the prompt "Enter locations to be graded :", enter the grading points. Repeat this procedure until all entities are graded. Graded pattern appears in different colours.

Grade-15. Grade Vertical Direction

This icon is for grading pattern pieces only lengthwise(y-direction). First, the current size of the pattern is required. Enter a value which must be an even number between 8 and 30. Then enter up to 9 sizes which you want to grade. Enter values which must be even numbers between 8 and 30.

When you get the message "Wait a moment please." wait for a few seconds while layers are set to different sizes to distinguish one size from another.

In this program the grade rule for the y-direction is a proportional increase or decrease of the y-direction distance from the zero point to the each grading point, where the ratio is some vertical length (or distance of each size to be graded to the same part measurement of the current size. For example, if you grade a front strap you might want to grade it only vertically. Then the grading ratio is the front opening length of each size to be graded to the front opening length of the current size.

There is a question whether you want to measure some length to define the grade rule or not. For example, when you grade the front strap, if you already know the front opening length of each size you don't need to measure them, but if you don't know them you need to measure them now. Decide and either enter "M" to measure or press "RETURN". If you enter "M", you will get instructions, follow them carefully. Enter measurements to define grading ratio.

When you are asked to enter a location for the zero point, select the zero point using the mouse. Then you are asked to enter a location for the text. Select the location where you want to write the pattern size to prevent being confused.

At this stage, the program knows the grade rule and zero point, so what you have to input are entity type and grading points. Available entity types are lines, B-spline (curve), points or notch point, you can select one of them. Depending on the entity type, there are some instructions, follow them carefully.

When you get the prompt "Enter locations to be graded :", enter grading points. Repeat this procedure until all entities are graded. Graded pattern appears in different colours.

Grade-16. Translate, Rotate or Mirror

See "Adapt-16. Translate, Rotate or Mirror" in the Page Help File "Adapt".

Grade-17. Trim or Divide Entity

See "Adapt-17. Trim or Divide Entity" in the Page Help File "Adapt".

Grade-18. Point

See "Detail-18. Point" in the Page Help File "Detail".

Grade-19. Line

See "Detail-21. Line" in the Page Help File "Detail".

Grade-20. B-spline

See "Adapt-20. B-spline" in the Page Help File "Adapt".

Grade-21. INSERT TEXT

Being different from other pulldown icons, this icon has 7 options, which are divided into parts (verb noun, modifiers) of command INSERT TEXT. All text is inserted by uppercase. You have to start with option 1 and 2, and end with option 7, Option 3~6 are optional, so you can select any combination of modifiers.

The 7 options are:

1. Insert Text (INSERT TEXT $$)

   This is a the verb noun part of the command to insert standard text strings. Each text string contains alphanumeric characters and spaces. You have to start with this option because CADDS command starts with verb noun (INSERT TEXT).

   When the system pauses after $$, type the text string as you want.

2. End of Text ($$)

   When you finish typing text string, select this option to make it clear that the text string you want to insert is between $$ (following INSERT TEXT) and $$.

3. Height (HEIGHT x)

   This modifier is optional. Without this modifier, the text is inserted at 0.4 CM (0.156 IN) height.

   When you select this modifier, after HEIGHT you get the prompt -> and the system pauses, enter the value (with the selected unit) for the height.

4. Centre Location (CJT)

   This modifier is optional. Without this modifier, the text is justified at the left of the text. If you

select this modifier, the text is justified at the centre of the text, which means the location you enter after the prompt 'MODEL loc' is the centre of the text. The modifier CJT and RJT cannot be used together.

5. Right Location (RJT)

This modifier is optional. Without this modifier, the text is justified at the left of the text. If you select this modifier, the text is justified at the right of the text, which means the location you enter after the prompt 'MODEL loc' is the right of the text. The modifier CJT and RJT cannot be used together.

6. Angle of Line (ANGLE x)

This modifier is optional. Without this modifier, the text is inserted horizontal.

When you select this modifier, after ANGLE you get the prompt -> and system pauses, enter the value for the angle of the line in degrees counterclockwise from the horizontal.

7. Execute (:)

You have to select this option when you want to execute the command. Because colon (:) completes the command line and call the the getdata prompt 'MODEL loc' so you can select the place to insert the text string(s).

The location you enter can be the left, centre or right of the text depending on the modifier CJT or RJT.

Ex 1) #n# INSERT TEXT $$FRONT PIECE$$ HEIGHT 1.5 : MODEL loc
d

Ex 2) #n# INSERT TEXT $$DAVID$$ HEIGHT 0.7 ANGLE 90 : MODEL loc d

EX 3) #n# INSERT TEXT $$THANK YOU!$$ CJT ANGLE 270 : MODEL loc d

4. Help File for Page 'Appear'

This Page Help File is to provide more information about every icon (except help icon) in the menu page 'appear'.

Appear-4. CHANGE LAYER

This command changes the layer associated with selected entities. All selected entities will be moved from their original layer to the specified layer. The layer on which an entity is located must be active or echoed (i.e, the entity must be visible) before that entity can be changed to another layer.

When you select this icon, you receive a -> prompt and the system pauses after -> prompt so you can enter a layer number to which selected entities moves. A layer number can be selected by an integer in the range of 0-254. You can enter it from the keyboard. Enter a space to terminate your input. Then you get the getdata prompt 'MODEL ent'. Select as many entities as you want. Press RETURN to execute the command.

Ex: #n# CHANGE LAYER -> 9 : MODEL ent ddd

This example changes selected entities to the layer 9.

Appear-5. COPY ENTITY TO

This command duplicates entities and copies them from one layer to one or more specified layers. Entities on several layer can be copied at the same time. The layer on which an entity is located must be active or echoed (i.e, the entity must be visible) before that entity can be changed to another layer.

When you select this icon, you receive a -> prompt and the system pauses after -> prompt so you can enter a layer number to which selected entities are copied. A layer number can be selected by an integer in the range of 0-254. You can enter it from the keyboard. Enter a space to terminate your input. Then you get the getdata prompt 'MODEL ent'. Select as many entities as you want. Press RETURN to execute the command.

Ex: #n# COPY ENTITY TO 3 : MODEL ent ddd

This example copies selected entities to the layer 3.

Appear-6. SELECT LAYER

This command specifies the active construction layer for the currently active drawing. The new layer can be selected by an integer in the range of 0 - 254. Entities that are subsequently selected or generated will reside on the selected layer.

A layer number must be specified in the command. When you select this icon, you receive a -> prompt and the system pauses after -> prompt so you can enter a layer number. You can enter it from the keyboard. Enter a space to terminate your input and execute the command.

Ex: #n# SELECT LAYER -> 5

This example selects layer 5 as the new active construction layer.

Appear-7. ECHO GRID ON NOSNAP

This command displays the grid preventing location selections from snapping to the nearest grid point. In other words, the grid is simply an aid for selecting locations, and has no effect to your getdata input.
The system will display grids at every 1 cm in x- and y-direction. A grid is a rectangular array of grid points useful in controlling the placement of entities in the active drawing.

Appear-8. ECHO GRID OFF

This command ends the display of the grid. A grid is a rectangular array of grid points useful in controlling the placement of entities in the active drawing.

Appear-9. ECHO GRID ON SNAP

This command displays the grid making location selections snapped (moved) to the nearest grid point. In other words, when you select a location, your getdata input is not the location you select but the nearest grid position. When you click the left mouse button, the small green cross appears on the nearest grid point and that is the getdata input.
The system will display grids at every 1 cm in x- and y-direction.

A grid is a rectangular array of grid points useful in controlling the placement of entities in the active drawing.

Appear-10. INSERT TEXT $$

Being different from other icons, this icon doesn't have a completed command, which means to insert text strings you have to select a few more icons (some are compulsory, others are optional). When you want to insert text strings, this icon (INSERT TEXT $$) has to be selected first, because CADDS commands starts with verb noun and 'INSERT TEXT' is the verb noun part of the command.

'$$' delimits a text string on both sides. So when you get the '$$' following INSERT TEXT, type the text strings which contains alphanumeric characters and spaces. All text strings are inserted in uppercase.

If you want to break a text line to place each part of the text at a different location, select '$' (icon number 11) and type another part of text. You can break text string as many times as you want using '$' before the delimiter '$$' is entered.

If you don't want to break the text string, when you finish typing in the text select '$$' (icon number 12) to define the end of the text.

Appear-11. $ (Breaks a text string)

This icon is optional in cases where you want to insert multiple strings simultaneously.

If you want to break a text line to place each part of the text at a different location, divide the text using '$'. You can break a text string as many times as you want using '$' before the delimiter '$$' entered. The order of text strings in the command corresponds to the order of insertion.

When you finish typing in the text (including '$'), select '$$' (icon number 12) to define the end of the text.

Note: This icon can be selected only after INSERT TEXT $$ (icon number 10) and before '$$' (icon number 12).

Appear-12. $$ (Ends a text string)

This icon is compulsory. You have to select this icon to define the end of the text string. So this icon can be selected following either 'INSERT TEXT $$' (icon number 10), or $ (icon number 11).

Now, you may select modifiers which specify the appearance of the text.

These optional modifiers are:

Icon No. 16 ---> text font (FONT n)

Icon No. 17 ---> text height (HEIGHT x)

Icon No. 18 ---> text angle (ANGLE x)

Icon No. 19 ---> character slant (SLANT x)

Icon No. 20 ---> text justification

(L, M, R, or bottom, middle)

You can select as many modifiers as you want. When you finish selecting modifiers, select the colon (:) (icon number 13) to complete the command line and execute it.

If you don't want to select any modifiers, select the colon (:) (icon number 13) to complete the command line and execute it.

Appear-13. : (execute)

This icon is compulsory. The colon (:) signals that the command line is complete and calls getdata.

When you finish selecting modifiers if any, you have to select a colon. If you don't want to select any modifiers, after selecting $$ (icon number 12) you have to select a colon. Then you receive the getdata prompt 'MODEL loc' which asks for locations where text strings are inserted. If you didn't break the text string using '$', at every location you enter the whole text string will be repeatedly inserted. If you break the text string using '$', at each location you enter each broken part of the text will be inserted. The order of insertion corresponds to the order of text strings in the command.

Ex: #n# INSERT TEXT $$CENTRE LINE$$ ANGLE 90 : MODEL loc d

In this example 'CENTRE LINE' is inserted vertically.

Ex: #n# INSERT TEXT $$HERE$THERE$$: MODEL loc dd

In this example 'HERE' is inserted at the first location you entered and 'THERE' is inserted at the second location you entered.

Appear-14. REGENERATE GRAPHICS

See "Block-8. REGENERATE GRAPHICS" in the Page Help File "Block".

Appear-15. Highlight (DISCRIMINATE ENTITY ON)

See "Detail-12. Highlight (DISCRIMINATE ENTITY ON)" in the Page Help File "Detail".

Appear-16. Text Font

This icon is one of the optional modifiers of the command INSERT TEXT. You can select this modifier after specifying text strings and before entering a colon (icon number 13). If you don't select any option from this icon (if you don't specify a text font), the text will be inserted by the standard (default) font.

The text font modifier has 7 options:

1. STANDARD (default)
2. LEROY
3. STICK (fastest to display)
4. MICROFILM STANDARD
5. LIGHTLINE GOTHIC
6. NEWS GOTHIC
7. CENTURY SCHOOLBOOK

When you select an option of this modifier, select a colon (icon number 13) to complete the command and execute it unless you want to select more modifiers. If you want to select another modifier, do it before entering a colon.

Appear-17. Text Height

This icon is one of the optional modifiers of the command INSERT TEXT. You can select this modifier after specifying text strings and before entering a colon (icon number 13).

If you don't select any option from this icon (if you don't specify a text height, the text will be inserted by the height of 0.4 CM (0.156 IN).

The text height modifier has 7 options:

1. other ---> A height should be entered after the prompt ->.
2. 0.1560
3. 0.1875
4. 0.2500
5. 0.3750
6. 0.5000
7. 1.0000

When you select an option of this modifier, select a colon (icon number 13) to complete the command and execute it unless you want to select more modifiers. If you want to select another modifier, do it before entering a colon.

Appear-18. Text Angle in Degrees

This icon is one of the optional modifiers of the command INSERT TEXT. You can select this modifier after specifying text strings and before entering a colon. If you don't select any option from this icon (if you don't specify a text angle), the text will be inserted horizontal. The angle is in degrees counterclockwise from the horizontal.

The text angle modifier has 7 options:

1. other ---> An angle should be entered after the prompt ->.
2. 0 (default)
3. 15

4. 30

5. 45

6. 60

7. 75

When you select an option of this modifier, select a colon (icon number 13) to complete the command and execute it unless you want to select more modifiers. If you want to select another modifier, do it before entering a colon.

Appear-19. Character Slant

This icon is one of the optional modifiers of the command INSERT TEXT. You can select this modifier after specifying text strings and before entering a colon. If you don't select any option from this icon (if you don't specify character slant), characters will be inserted vertical. The slant is defined in degrees clockwise from the vertical. Slant anglation is possible between -45 and +45 degrees.

The character slant modifier has 6 options:

1. other ---> Slant angle should be entered after the prompt ->.

2. 0 (default)

3. 5

4. 10

5. 15 (Italics)

6. 20

When you select an option of this modifier, select a colon (icon number 13) to complete the command and execute it unless you want to select more modifiers. If you want to select another modifier, do it before entering a colon.

Appear-20. Text Justification

This icon has 2 groups of optional modifiers of the command INSERT TEXT. You can select a modifier from each group after specifying text strings and before entering a colon.

Modifiers of one group are LJT (left justification), CJT (center justification) and RJT (right justification), which specify left, center, and right justification of the text, respectively. The modifier LJT, CJT and RJT cannot be used together. If you don't select any modifier from this group, the text will be justified at the left of the text (default, LJT).

Modifiers of the other group are BJT (bottom justification) and MJT (middle justification). BJT justifies the text on the text string origin and base angle. MJT justifies the text downward by half of the height of the text string relative to the origin and base angle of the string. BJT and MJT can not be used together. If you don't select any modifier from this group, the text will be justified on the text string origin and base angle (BJT).

Appear-21. CHANGE APPEARANCE FONT

This command permits the changing of the font of the graphics of selected entities.

This icon has 7 options for new fonts:

1. SOLID -------- Entities changes from any font to solid.

   CHANGE APPEARANCE FONT FROM ANY TO SOLID : MODEL ent ddd

2. DASH --------- Entities changes from any font to dash.

   CHANGE APPEARANCE FONT FROM ANY TO DASH : MODEL ent ddd

3. 1-DASH ------- Entities changes from any font to 1-dash.

CHANGE APPEARANCE FONT FROM ANY TO 1-DASH : MODEL ent ddd

4. 2-DASH ------- Entities changes from any font to 2-dash.

   CHANGE APPEARANCE FONT FROM ANY TO PHANTOM : MODEL ent ddd

5. ARROW -------- Entities changes from any font to arrow.

   CHANGE APPEARANCE FONT FROM ANY TO ARROW : MODEL ent ddd

6. THICK SOLID -- Entities changes from any font to thick solid.

   CHANGE APPEARANCE FONT FROM ANY TO A-LINE : MODEL ent ddd

7. THICK DASH --- Entities changes from any font to thick dash.

   CHANGE APPEARANCE FONT FROM ANY TO J-LINE : MODEL ent ddd

REFERENCES

1. G. Segal. "Information Processing Executes Quick Response", Bobbin, November, 1986, pp. 72-76.

2. E.O. Weller. "Apparel Distribution in the Future", Bobbin, January, 1986, pp. 93-101.

3. A.I. Colgate. "Seminole on the Path of Quick Response", Bobbin, January, 1988, p. 56.

4. G. Segal. "De-mystifying CAD/CAM Systems", Bobbin, February, 1986, pp. 41-47.

5. P.G. Gillease. "On the Fast Track", Bobbin, January, 1988, pp. 44-54.

6. K. Kosh. "Computer Systems Automate Design Function", Bobbin, February, 1987, pp. 51-64.

7. M. Disher. "CAD Becomes a Necessity", Manufacturing Clothier, March, 1987, pp. 28-29.

8. G. Segal. "Integration, EDI Take Spotlight", Bobbin, November, 1989, pp. 78-87.

9. M. Disher. "Cost-effective Adaptable Developments", Manufacturing Clothier, March, 1987, pp. 24-25.

10. "Computers in Clothing Conference", Manufacturing Clothier, April, 1987, pp. 31-35.

11. M. Disher. "Computers in Clothing', Manufacturing Clothier, May, 1987, pp. 13-19.

12. G. Segal. "The ABC's od EDI, UPC, MRP, Etc.", Bobbin, December 1988, pp. 52-58.

13. K. Kosh. "No Miss with MIS", Bobbin, February, 1988, pp. 34-43.

14. A.I.Tray. "Computer Design's Third Dimension", Bobbin, February, 1987, pp. 58-59.

15. K. Kosh. "How CAD Fits in the Design Process", Bobbin, February, 1989, pp. 67-70.

16. P.B. Miller, J. DeJonge. "University Explores Computer Aided Design by Microcomputer" (Education Update), Bobbin, February, 1987, pp. 127-128.

17. P.B. Miller, A. Racine. "Less Means More in CAD", Bobbin, September, 1988, pp. 222-225.

18. A.I. Colgate. "Heritage in Real Time", Bobbin, February, 1988, pp. 46-47.

19. P.B. Miller. "Trim Fit for Design", Bobbin, March, 1989, pp. 58-64.

20. A.I. Colgate. "Seminole on the Path of Quick Response", Bobbin, January, 1988, p. 56.

21. M. Disher. "CAD Updates", Manufacturing Clothier, October, 1988, pp. 15-20.

22. "CAD/CAM Connections", Manufacturing Clothier, June, 1988, pp. 27-29.

23. "CAD/CAM Put to the Test", Manufacturing Clothier, December, 1988, pp. 33-35.

24. P. R.M. Jones and G. M. West. "Automated Anthropometry: Its Role in Garment Manufacture" (Abstract), SERC/ACME Conference on Automation of Garment Manufacture, 1986.

25. S. Nishikawa and N. Suda. "Technique of Measurements for Three-Dimensional Shape of Flared-Skirts and Examination on the Shape of Node Formed by Draping", Bulletin of Research Institute for Polymer and Textiles, No. 142, 1984, pp. 5-24.

26. Y. Besso and A. Shibuya. "A Representation Method of Three-Dimensional Shapes from Moire Photographs", Bulletin of Research Institute for Polymer and Textiles, No. 142, 1984, pp. 25-28.

27. H. Okabe and H. Akami. "The Estimation of the Three-Dimensional Shapes of Garments", Bulletin of Research Institute for Polymer and Textiles, No. 142, 1984, pp. 63-71.

28. H. Imaoka, H. Okabe, S. Nishikawa, A. Shibuya and H. Akami. "A Method of Estimating the Three-Dimensional Shapes of Garments by Use of Triangular Finite Elements", Bulletin of Research Institute for Polymer and Textiles, No. 142, 1984, pp. 72-80.

29. Y. Besso and H. Akami. "A Representation of the Computational Shapes of Garments", Bulletin of Research Institute for Polymer and Textiles, No. 142, 1984, pp. 81-85.

30. H. Okabe, H. Imaoka and Y. Besso. "On the Implement of the New Pattern-Making-System - the System for the Estimation of Three Dimensional Shapes of Garments from Their Patterns", Bulletin of Research Institute for Polymer and Textiles, No. 142, 1984, pp. 87-96.

31. H. Okabe, H. Imaoka and H. Akami. "Paper Patterns of Dress for 3-Dimensional CAD/CAM and Their Automatical Division into Finite Elements", Sen-i Gakkaishi (Journal of The Society of Fiber Science and Technology, Japan), Vol. 42, No. 4, 1986, pp. 231-239.

32. H. Okabe, H. Imaoka, T. Tomiha, M. Terao, M. Yamada, A. Shibuya, H. Akami and N. Aisaka. "Transformation from Paper Pattern to Spatial Structure of Dress by Computer-Simulation of Sewing and Dressing", Sen-i Gakkaishi (Journal of The Society of Fiber Science and Technology, Japan), Vol. 44, No. 3, 1988, pp. 129-136.

33. CADDStation Software CVMAC Manual, Computervision Corporation, 1986.

34. CADDStation Systems Workstation User Guide, Computervision Corporation, 1986.

35. CADDStation Systems Core Reference (Vol. 1 & Vol. 2), Computervision Corporation, 1986.

36. Winifred Aldrich. Metric Pattern Cutting, Unwin Hyman Ltd, London, 1985.

37. O. Rahman. Computer Aided Design in Pattern Making, Ph.D Thesis, The University of Leeds, 1986.

38. P. Kunick. Sizing, Pattern Construction and Grading for Women's and Children's Garments, Philip Kunick Ltd., London, 1967.

39. E. Kopp, V. Rolfo, B. Zelin and L. Gross. Designing Apparel Through the Flat Pattern, Fairchild Publications, NewYork, 1981.

40. P. Taylor and M. Shoben. Grading for the Fashion Industry, Grading for the Fashion Industry, Hutchinson, London, 1984.

APPENDIX A

BLOCK PATTERN GENERATION PROGRAMS

```
+-----------------------------------------------------------+
|                                                           |
|              BDCLLK (Linkfile: BDCL, BDSIZE)              |
|                                                           |
+-----------------------------------------------------------+

MAIN BDCL
IMAGE BDCLIMAGE
LINKM BDSIZE
ENDLINK
```

```
+-----------------------------------------------------------+
|                                                           |
|              BDEALK (Linkfile: BDEA, BDSIZE)              |
|                                                           |
+-----------------------------------------------------------+

MAIN BDEA
IMAGE BDEAIMAGE
LINKM BDSIZE
ENDLINK
```

```
+-----------------------------------------------------------+
|                                                           |
|              BDDLLK (Linkfile: BDDL, BDSIZE)              |
|                                                           |
+-----------------------------------------------------------+

MAIN BDDL
IMAGE BDDLIMAGE
LINKM BDSIZE
ENDLINK
```

```
+-----------------------------------------------------------+
|                                                           |
|              BDOVLK (Linkfile: BDOV, BDSIZE)              |
|                                                           |
+-----------------------------------------------------------+

MAIN BDOV
IMAGE BDOVIMAGE
LINKM BDSIZE
ENDLINK
```

```
+-----------------------------------------------------------------+
|                                                                 |
|                BDTALK (Linkfile: BDTA, BDSIZE)                  |
|                                                                 |
+-----------------------------------------------------------------+
```

```
MAIN BDTA
IMAGE BDTAIMAGE
LINKM BDSIZE
ENDLINK
```

```
+-----------------------------------------------------------------+
|                                                                 |
|                BDSHLK (Linkfile: BDSH, BDSIZE)                  |
|                                                                 |
+-----------------------------------------------------------------+
```

```
MAIN BDSH
IMAGE BDSHIMAGE
LINKM BDSIZE
ENDLINK
```

```
+-----------------------------------------------------------------+
|                                                                 |
|                BDJELK (Linkfile: BDJE, BDSIZE)                  |
|                                                                 |
+-----------------------------------------------------------------+
```

```
MAIN BDJE
IMAGE BDJEIMAGE
LINKM BDSIZE
ENDLINK
```

```
+-----------------------------------------------------------------+
|                                                                 |
|                BDKNLK (Linkfile: BDKN, BDSIZE)                  |
|                                                                 |
+-----------------------------------------------------------------+
```

```
MAIN BDKN
IMAGE BDKNIMAGE
LINKM BDSIZE
ENDLINK
```

The bodice size (BDSIZE) procedure

```
<#
<#
PROC BDSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,ARMDPT,<#
>CHEST,NECK,&n,&data)
DECLARE REAL bust(12),shln(12),natowa(12),bwd(12),<#
>watohi(12),dart(12),armdpt(12),chest(12),neck(12)
DATA bust/80,84,88,92,97,102,107,112,117,122,127,132/
DATA shln/11.75,12,12.25,12.5,12.8,13.1,13.4,13.7,<#
>14,14.3,14.6,14.9/
DATA natowa/39,39.5;40,40.5,41,41.5,42,42.5,43,<#
>43.2,43.4,43.6/
DATA bwd/32.4,33.4,34.4,35.4,36.6,37.8,39,40.2,<#
>41.4,42.6,43.8,45/
DATA watohi/20,20.3,20.6,20.9,21.2,21.5,21.8,22.1,<#
>22.3,22.5,22.7,22.9/
DATA dart/5.8,6.4,7,7.6,8.2,8.8,9.4,10,10.6,<#
>11.2,11.8,12.4/
DATA armdpt/20,20.5,21,21.5,22,22.5,23,23.5,24.2,<#
>24.9,25.6,26.3/
DATA chest/30,31.2,32.4,33.6,35,36.5,38,39.5,41,<#
>42.5,44,45.5/
DATA neck/35,36,37,38,39.2,40.4,41.6,42.8,44,45.2,<#
>46.4,47.6/
ERTRAP 999
<#
<#
#10 CONTINUE
PRINT {''}
&data=''
PRINT      Which measurements do you want to use?
PRINT  ----------------------------------------
PRINT  The body measurements of this program --- 1
PRINT  Your individual body measurements ------- 2
PRINT  ----------------------------------------
READ(        Enter the number please. ---> )&data
WHEN &data='1'.OR.&data='2'
          CONTINUE
  ELSE
          PRINT Wrong answer, enter again please.
          GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
PRINT {''}
&unit=''
READ( Which units do you want to use? <#
>    Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'.OR.&unit='IN'.OR.<#
>&unit='in'
          CONTINUE
  ELSE
          PRINT Wrong answer, enter again please.
```

```
          GOTO 20
ENDWHEN
<#
<#
#30 CONTINUE
PRINT {''}
&n=''
WHEN &unit='CM'.OR.&unit='cm'
          GOSUB 1000
  ELSE
          GOSUB 2000
ENDWHEN
PRINT  The size must be an even number between 8 and 30.
WHEN &data='1'
READ(                Enter the size please. ---> )&n
     ELSE
READ(          Enter the nearest size please. ---> )&n
ENDWHEN
WHEN &n='8'.OR.&n='10'.OR.&n='12'.OR.&n='14'<#
>.OR.&n='16'.OR.&n='18'.OR.&n='20'.OR.&n='22'.OR.<#
>&n='24'.OR.&n='26'.OR.&n='28'.OR.&n='30'
          n=&n
          i=(n-6)/2
          BUST=bust(i)
          SHLN=shln(i)
          NATOWA=natowa(i)
          BWD=bwd(i)
          WATOHI=watohi(i)
          DART=dart(i)
          ARMDPT=armdpt(i)
          CHEST=chest(i)
          NECK=neck(i)
  ELSE
          PRINT Wrong value entered, enter again please.
          GOTO 30
ENDWHEN
<#
<#
#40 CONTINUE
PRINT {''}
&height=''
WHEN &unit='CM'.OR.&unit='cm'
    PRINT                  Which height do you want?
    PRINT    ----------------------------------------
    PRINT    The short women ( 152cm - 160cm ) -------- S
    PRINT    The medium women ( 160cm - 170cm ) -------- M
    PRINT    The tall women ( 170cm - 178cm ) ---------- T
    PRINT    ----------------------------------------
   ELSE
    PRINT                Which height do you want?
    PRINT    ----------------------------------------
    PRINT    The short women (5ft - 5ft 3in) ---------- S
    PRINT    The medium women (5ft 3in - 5ft 7in) ------ M
    PRINT    The tall women (5ft 7in - 5ft 10in) ------- T
    PRINT    ----------------------------------------
ENDWHEN
WHEN &data='1'
READ( Enter the height please.  S or M or T ---> )&height
```

```
ELSE
READ( Enter the nearest height please.<#
>  S or M or T ---> )&height
ENDWHEN
WHEN &height='S'.OR.&height='s'
          NATOWA=NATOWA-2
          ARMDPT=ARMDPT-0.8
  OR &height='M'.OR.&height='m'
          CONTINUE
  OR &height='T'.OR.&height='t'
          NATOWA=NATOWA+2
          ARMDPT=ARMDPT+0.8
  ELSE
          PRINT Wrong value entered, enter again please.
          GOTO 40
ENDWHEN
<#
<#
IF (&unit='CM'.OR.&unit='cm') GOTO 50
BUST=BUST/2.54
SHLN=SHLN/2.54
NATOWA=NATOWA/2.54
BWD=BWD/2.54
WATOHI=WATOHI/2.54
DART=DART/2.54
ARMDPT=ARMDPT/2.54
CHEST=CHEST/2.54
NECK=NECK/2.54
<#
<#
#50 CONTINUE
&BUST=BUST
&SHLN=SHLN
&NATOWA=NATOWA
&BWD=BWD
&WATOHI=WATOHI
&DART=DART
&ARMDPT=ARMDPT
&CHEST=CHEST
&NECK=NECK
<#
<#
WHEN &data='1'
          PRINT {''}
          &change=''
PRINT       The data you entered is<#
> ------------------------------------
PRINT          size; {&n}          height; {&height} <#
>             unit; {&unit}
PRINT          ------------------------------<#
>-----------------------------------
PRINT          bust={&BUST(,4)}   <#
>    shoulder length={&SHLN(,4)}<#
>       nape to waist={&NATOWA(,4)}
PRINT          back width={&BWD(,4)} <#
>     waist to hip={&WATOHI(,4)}<#
>       dart={&DART(,4)}
PRINT          armhole depth={&ARMDPT(,4)}   <#
```

```
<      chest={&CHEST(,4)}<#
>       neck size={&NECK(,4)}
PRINT      ------------------------------<#
>-------------------------------
READ(                    Do you want to change? <#
>     Y/N ---> )&change
         WHEN &change='Y'.OR.&change='y'
                 GOTO 10
           OR &change='N'.OR.&change='n'
                 CONTINUE
           ELSE
              PRINT Wrong value entered, enter again please.
              GOTO 50
         ENDWHEN
  OR &data='2'
         PRINT {''}
PRINT    ------------------------------------<#
>-------------------------------------
WHEN &unit='CM'.OR.&unit='cm'
PRINT Enter your individual measurements please. (unit: cm)
ELSE
PRINT Enter your individual measurements please. (unit: in)
ENDWHEN
PRINT    If you don't want to change the measurement<#
> of the part, press RETURN.
PRINT    ------------------------------------<#
>-------------------------------------
PRINT      bust (size;{&n}, height;{&height}) =<#
> {&BUST(,4)}
READ(            your bust ---> )BUST
PRINT      shoulder length (size;{&n},<#
> height;{&height}) = {&SHLN(,4)}
READ(            your shoulder length ---> )SHLN
PRINT      nape to waist (size;{&n},<#
> height;{&height}) = {&NATOWA(,4)}
READ(            your nape to waist ---> )NATOWA
PRINT      back width (size;{&n},<#
> height;{&height}) = {&BWD(,4)}
READ(            your back width ---> )BWD
PRINT      waist to hip (size;{&n},<#
> height;{&height}) = {&WATOHI(,4)}
READ(            your waist to hip ---> )WATOHI
PRINT      dart (size;{&n}, height;{&height})<#
> = {&DART(,4)}
READ(            your dart ---> )DART
PRINT      armhole depth (size;{&n},<#
> height;{&height}) = {&ARMDPT(,4)}
READ(            your armhole depth ---> )ARMDPT
PRINT      chest (size;{&n}, height;{&height}) =<#
> {&CHEST(,4)}
READ(            your chest ---> )CHEST
PRINT      neck size (size;{&n}, height;{&height})<#
> = {&NECK(,4)}
READ(            your neck size ---> )NECK
<#
<#
#60 CONTINUE
&BUST=BUST
```

```
&SHLN=SHLN
&NATOWA=NATOWA
&BWD=BWD
&WATOHI=WATOHI
&DART=DART
&ARMDPT=ARMDPT
&CHEST=CHEST
&NECK=NECK
<#
<#
&change=''
PRINT {''}
PRINT       The data you entered is<#
> ------------------------------------
PRINT          nearest size; {&n}    <#
>  nearest height; {&height}     unit; {&unit}
PRINT          ------------------------------<#
>-------------------------------
PRINT          bust={&BUST(,4)}    <#
>shoulder length={&SHLN(,4)}    nape to waist={&NATOWA(,4)}
PRINT          back width={&BWD(,4)}    <#
>waist to hip={&WATOHI(,4)}    dart={&DART(,4)}
PRINT          armhole depth={&ARMDPT(,4)}       <#
>chest={&CHEST(,4)}     neck size={&NECK(,4)}
PRINT       ------------------------------------<#
>-------------------------------
READ(      Do you want to change?      Y/N ---> )&change
        WHEN &change='Y'.OR.&change='y'
                GOTO 10
          OR &change='N'.OR.&change='n'
                CONTINUE
          ELSE
           PRINT Wrong value entered, enter again please.
           GOTO 60
        ENDWHEN
ENDWHEN
<#
<#
IF (&unit='CM'.OR.&unit='cm') GOTO 999
BUST=BUST*2.54
SHLN=SHLN*2.54
NATOWA=NATOWA*2.54
BWD=BWD*2.54
WATOHI=WATOHI*2.54
DART=DART*2.54
ARMDPT=ARMDPT*2.54
CHEST=CHEST*2.54
NECK=NECK*2.54
<#
<#
#999 RETURN
<#
<#
#1000 CONTINUE
PRINT                        <#
> WOMEN OF MEDIUM HEIGHT 160CM-170CM    (UNIT: CM)
PRINT      ------------------------------<#
>-----------------------------------------
```

```
PRINT                    8    10    12    14    16    18   <#
> 20    22    24    26    28    30
PRINT      ------------------------------<#
>-------------------------------------
PRINT      BUST        80    84    88    92    97   <#
>102  107   112   117   122   127   132
PRINT      NA-TO-WA   39    39.5 40    40.5 41   <#
> 41.5 42    42.5 43    43.2 43.4 43.6
PRINT      B-WIDTH    32.4 33.4 34.4 35.4 36.6 <#
>37.8 39    40.2 41.4 42.6 43.8 45
PRINT      CHEST       30    31.2 32.4 33.6 35    <#
>36.5 38    39.5 41    42.5 44    45.5
PRINT      ------------------------------<#
>-------------------------------
PRINT                    NA-TO-WA: nape to waist <#
>          B-WIDTH: back width
PRINT {''}
RTNSUB
<#
<#
#2000 CONTINUE
PRINT                <#
> WOMEN OF MEDIUM HEIGHT 5FT 3IN - 5FT 7IN (UNIT: IN)
PRINT      ------------------------------<#
>-------------------------------
PRINT                    8    10    12    14    16    18   <#
> 20    22    24    26    28    30
PRINT      ------------------------------<#
>-------------------------------
PRINT      BUST        31.5 33.1 34.6 36.2 38.2 <#
>40.2 42.2 44.1 46.1 48    50    52
PRINT      NA-TO-WA   15.4 15.6 15.7 15.9 16.1 <#
>16.3 16.5 16.7 16.9 17    17.1 17.2
PRINT      B-WIDTH    12.8 13.1 13.5 13.9 14.4 <#
>14.9 15.4 15.8 16.3 16.8 17.2 17.7
PRINT      CHEST       11.8 12.3 12.8 13.2 13.8 <#
>14.4 15    15.6 16.1 16.7 17.3 17.9
PRINT      ------------------------------<#
>-------------------------------
PRINT                    NA-TO-WA: nape to waist <#
>          B-WIDTH: back width
PRINT {''}
RTNSUB
```

```
+-----------------------------------------------------------+
|                                                           |
|        The close fitting bodice block pattern (BDCL)      |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL x(39),y(39)
DECLARE LOCATION P(39)
DECLARE ENTITY p(5),PNT(3),L(31),T(6),B(3),C(3)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP BDSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,<#
>ARMDPT,CHEST,NECK,&n,&data)
<#
<#
PRINT {''}
PRINT  Select the position for the centre back neck<#
> of the bodice block please.
PRINT  ------------------------------<#
>----------------------------------------
PRINT     Move the mouse on the pad to place the cursor<#
> on the graphic window
PRINT     at the position you require. Press the left<#
> button.
PRINT  ------------------------------<#
>----------------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(2)=:x(4)=:x(6)=:X
y(1)=y(30)=:Y
y(2)=y(14)=:y(12)=:y(27)=:y(19)=:y(20)=:y(3)=:<#
>y(1)-ARMDPT-0.5
x(3)=x(18)=:x(5)=:x(39)=:x(7)=:x(1)+BUST/2+5
y(4)=y(15)=:y(28)=:y(21)=:y(5)=:y(1)-NATOWA
y(6)=y(16)=:y(29)=:y(22)=:y(7)=:y(4)-WATOHI
x(8)=x(1)+NECK/5-0.2
y(8)=y(1)+1.5
y(9)=y(1)-ARMDPT/5+0.7
x(9)=x(8)+SQRT((SHLN+1)**2-(y(8)-y(9))**2)
x(10)=(x(8)+x(9))/2
y(10)=(y(8)+y(9))/2
x(11)=x(10)-1
y(11)=y(10)-5
x(12)=x(13)=:x(2)+BWD/2+0.5
y(13)=(y(9)+y(12))/2
x(14)=x(15)=:x(16)=:(x(2)+x(12))/2
x(17)=x(3)-NECK/5+0.7
WHEN BUST<95
        y(17)=y(24)=:y(1)+1.5
```

```
OR BUST.GE.95.AND.BUST<100
      y(17)=y(24)=:y(1)+1.8
OR BUST.GE.100.AND.BUST<105
      y(17)=y(24)=:y(1)+2.1
OR BUST.GE.105.AND.BUST<110
      y(17)=y(24)=:y(1)+2.4
OR BUST.GE.110.AND.BUST<115
      y(17)=y(24)=:y(1)+2.7
OR BUST.GE.115.AND.BUST<120
      y(17)=y(24)=:y(1)+3
  OR BUST.GE.120.AND.BUST<125
        y(17)=y(24)=:y(1)+3.3
  OR BUST.GE.125.AND.BUST<130
        y(17)=y(24)=:y(1)+3.6
  ELSE
        y(17)=y(24)=:y(1)+3.9
ENDWHEN
y(18)=y(17)-NECK/5+0.2
x(19)=x(26)=:x(3)-(CHEST+DART)/2
x(20)=x(23)=:x(21)=:x(22)=:(x(3)+x(19))/2
y(23)=y(3)-2.5
x(24)=x(17)-DART
y(25)=y(9)-1.5
x(25)=x(24)-SQRT(SHLN**2-(y(24)-y(25))**2)
y(26)=y(3)+(y(18)-y(3))/3
x(27)=x(28)=:x(29)=:(x(12)+x(19))/2
x(30)=x(1)+(x(8)-x(1))/3
WHEN BUST<95
        backcurve=2.5
        frontcurve=2
  OR BUST.GE.95.AND.BUST.LT.115
        backcurve=3
        frontcurve=2.5
  ELSE
        backcurve=3.5
        frontcurve=3
ENDWHEN
x(31)=x(12)+backcurve/SQRT(2)
y(31)=y(12)+backcurve/SQRT(2)
x(32)=x(19)-frontcurve/SQRT(2)
y(32)=y(19)+frontcurve/SQRT(2)
WHEN BUST<95
        waistdrop=1
  OR BUST.GE.95.AND.BUST<115
        waistdrop=1.5
  ELSE
        waistdrop=2
ENDWHEN
x(33)=x(15)-1.75
y(33)=y(34)=:y(4)-waistdrop/4
x(34)=x(15)+1.75
x(35)=x(28)-1.5
y(35)=y(36)=:y(4)-waistdrop/2
x(36)=x(28)+2.5
x(37)=x(21)-2.25
y(37)=y(38)=:y(4)-waistdrop*3/4
x(38)=x(21)+2.25
y(39)=y(4)-waistdrop
```

```
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>39
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
$E=LINE/P(8),P(10)
PNT(1)=POINT/P(10),DISTAN,-0.5,$E
DELETE $E
$E=LINE/P(9),P(10)
PNT(2)=POINT/P(10),DISTAN,-0.5,$E
DELETE $E
PNT(3)=POINT/x(8)-(x(8)-x(1))/4,y(1)+(y(8)-y(1))/3,0
p(1)=POINT/x(13),y(13),0
p(2)=POINT/x(31),y(31),0
p(3)=POINT/x(32),y(32),0
p(4)=POINT/x(26),y(26),0
<#
<#
L(1)=LINE/P(1),P(6)
L(2)=LINE/P(6),P(29)
L(3)=LINE/P(27),P(29)
L(4)=LINE/P(8),PNT(1)
L(5)=LINE/PNT(1),P(11)
L(6)=LINE/PNT(2),P(11)
L(7)=LINE/PNT(2),P(9)
L(8)=LINE/P(18),P(7)
L(9)=LINE/P(7),P(29)
L(10)=LINE/P(17),P(23)
L(11)=LINE/P(24),P(23)
L(12)=LINE/P(24),P(25)
L(13)=LINE/P(4),P(33)
L(14)=LINE/P(14),P(33)
L(15)=LINE/P(14),P(34)
L(16)=LINE/P(34),P(35)
L(17)=LINE/P(27),P(35)
L(18)=LINE/P(27),P(36)
L(19)=LINE/P(36),P(37)
L(20)=LINE/P(23),P(37)
L(21)=LINE/P(23),P(38)
L(22)=LINE/P(38),P(39)
L(23)=LINE/P(2),P(27),FONT,'DASH'
L(24)=LINE/P(4),P(28),FONT,'DASH'
L(25)=LINE/P(14),P(16),FONT,'DASH'
L(26)=LINE/P(3),P(27),FONT,'DASH'
L(27)=LINE/P(5),P(28),FONT,'DASH'
L(28)=LINE/P(20),P(22),FONT,'DASH'
L(29)=LINE/P(1),P(30)
<#
<#
B(1)=BSPLIN/3,P(27),P(31),P(13),P(9)    <# b-armhole
$L1=LINE/P(25),P(26)
$L2=LINE/(x(25)+x(26))/2,(y(25)+y(26))/2,0,PERP,$L1,<#
>LENGTH,0.5
p(5)=POINT/(x(25)+x(26))/2,(y(25)+y(26))/2,0,DISTAN,<#
>0.5,$L2
```

```
DELETE $L1,$L2
B(2)=BSPLIN/3,P(27),P(32),P(26),p(5),P(25)   <# f-armhole
B(3)=BSPLIN/3,P(30),PNT(3),P(8)              <# b-neck
C(1)=CIRCLE/RADIUS,NECK/5-0.2,RIGHT,P(17),P(18),<#
>PGO,P(17),PEND,P(18)
<#
<#
C(2)=CIRCLE/CENTER,P(13),RADIUS,0.4
C(3)=CIRCLE/CENTER,P(26),RADIUS,0.4
L(30)=LINE/x(1)+5,y(2)+5,0,x(1)+5,y(4)-10,0,FONT,'ARROW'
L(31)=LINE/x(3)-5,y(2)+5,0,x(3)-5,y(4)-10,0,FONT,'ARROW'
<#
<#
&T1='close fitting bodice'
T(1)=TEXT/&T1,(x(6)+x(29))/2,y(6)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(7)+x(29))/2,y(6)+2,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
T(3)=TEXT/&name+' < SIZE '+&n+' >',(x(6)+x(29))/2,<#
>y(6)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name+' < SIZE '+&n+' >',(x(7)+x(29))/2,<#
>y(6)+3,0,TJST,'CJT',THGT,0.75
  OR &data='2'
IF (&name='') GOTO 200
T(3)=TEXT/&name,(x(6)+x(29))/2,y(6)+3,0,TJST,'CJT',<#
>THGT,0.75
T(4)=TEXT/&name,(x(7)+x(29))/2,y(6)+3,0,TJST,'CJT',<#
>THGT,0.75
#200 CONTINUE
ENDWHEN
T(5)=TEXT/'centre back line',x(1)+1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,<#
>TANG,270
T(6)=TEXT/'centre front line',x(3)-1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,<#
>TANG,90
<#
<#
DELETE 3,PNT(1)
<#
<#
DISP ALL
#999 END
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|          The easy fitting bodice block pattern (BDEA)                 |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE REAL x(25),y(25)
DECLARE LOCATION P(25)
DECLARE ENTITY L(16),T(6),B(3),C(3),p(4)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP BDSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,<#
>ARMDPT,CHEST,NECK,&n,&data)
<#
<#
PRINT {''}
PRINT  Select the position for the centre back neck<#
> of the bodice block please.
PRINT  ------------------------------------<#
>------------------------------------
PRINT     Move the mouse on the pad to place the cursor<#
> on the graphic window
PRINT     at the position you require. Press the left<#
> button.
PRINT  ------------------------------------<#
>------------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(2)=:x(4)=:x(6)=:X
y(1)=y(22)=:Y
y(2)=y(10)=:y(19)=:y(15)=:y(17)=:y(3)=:y(1)-ARMDPT-2.5
x(3)=x(13)=:x(5)=:x(7)=:x(1)+BUST/2+7
y(4)=y(20)=:y(5)=:y(1)-NATOWA
y(6)=y(21)=:y(7)=:y(4)-WATOHI
x(8)=x(1)+NECK/5-0.2
y(8)=y(1)+1.5
y(9)=y(1)-ARMDPT/5+1
x(9)=x(8)+SQRT((SHLN+1)**2-(y(8)-y(9))**2)
x(10)=x(11)=:x(1)+BWD/2+1
y(11)=(y(10)+y(9))/2
x(12)=x(3)-NECK/5+0.7
WHEN BUST<95
          y(12)=y(14)=:y(1)+1.5
  OR BUST.GE.95.AND.BUST<100
          y(12)=y(14)=:y(1)+1.8
  OR BUST.GE.100.AND.BUST<105
          y(12)=y(14)=:y(1)+2.1
  OR BUST.GE.105.AND.BUST<110
          y(12)=y(14)=:y(1)+2.4
```

```
OR BUST.GE.110.AND.BUST<115
      y(12)=y(14)=:y(1)+2.7
OR BUST.GE.115.AND.BUST<120
      y(12)=y(14)=:y(1)+3
OR BUST.GE.120.AND.BUST<125
      y(12)=y(14)=:y(1)+3.3
OR BUST.GE.125.AND.BUST<130
      y(12)=y(14)=:y(1)+3.6
ELSE
      y(12)=y(14)=:y(1)+3.9
ENDWHEN
y(13)=y(12)-NECK/5+0.2
x(14)=x(12)-DART/2
x(15)=x(16)=:x(3)-(CHEST/2+1+DART/4)
y(16)=(y(3)+y(13))/2
x(17)=(x(3)+x(15))/2
y(18)=y(9)-1.5
x(18)=x(14)-SQRT((SHLN+0.5)**2-(y(14)-y(18))**2)
x(19)=x(20)=:x(21)=:(x(10)+x(15))/2
x(22)=x(1)+(x(8)-x(1))/3
WHEN BUST<95
        backcurve=2.5
        frontcurve=2.25
  OR BUST.GE.95.AND.BUST.LT.115
        backcurve=3
        frontcurve=2.75
  ELSE
        backcurve=3.5
        frontcurve=3.25
ENDWHEN
x(23)=x(10)+backcurve/SQRT(2)
y(23)=y(10)+backcurve/SQRT(2)
x(24)=x(15)-frontcurve/SQRT(2)
y(24)=y(15)+frontcurve/SQRT(2)
x(25)=x(8)-(x(8)-x(1))/4
y(25)=y(1)+(y(8)-y(1))/3
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>25
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
p(1)=POINT/x(11),y(11),0
p(2)=POINT/x(23),y(23),0
p(3)=POINT/x(24),y(24),0
p(4)=POINT/x(16),y(16),0
<#
<#
L(1)=LINE/P(1),P(6)
L(2)=LINE/P(6),P(21)
L(3)=LINE/P(19),P(21)
L(4)=LINE/P(8),P(9)
L(5)=LINE/P(13),P(7)
L(6)=LINE/P(7),P(21)
L(7)=LINE/P(12),P(17)
L(8)=LINE/P(14),P(17)
```

```
L(9)=LINE/P(14),P(18)
L(10)=LINE/P(2),P(19),FONT,'DASH'
L(11)=LINE/P(4),P(20),FONT,'DASH'
L(12)=LINE/P(3),P(19),FONT,'DASH'
L(13)=LINE/P(5),P(20),FONT,'DASH'
L(14)=LINE/P(1),P(22)
<#
<#
B(1)=BSPLIN/3,P(19),P(23),P(11),P(9)      <# b-armhole
B(2)=BSPLIN/3,P(19),P(24),P(16),P(18)     <# f-armhole
B(3)=BSPLIN/3,P(22),P(25),P(8)            <# b-neck
C(1)=CIRCLE/RADIUS,NECK/5-0.2,RIGHT,P(12),P(13),<#
>PGO,P(12),PEND,P(13)
<#
<#
C(2)=CIRCLE/CENTER,P(11),RADIUS,0.4
C(3)=CIRCLE/CENTER,P(16),RADIUS,0.4
L(15)=LINE/x(1)+5,y(2)+5,0,x(1)+5,y(4)-10,0,FONT,'ARROW'
L(16)=LINE/x(3)-5,y(2)+5,0,x(3)-5,y(4)-10,0,FONT,'ARROW'
<#
<#
&T1='easy fitting bodice'
T(1)=TEXT/&T1,(x(6)+x(21))/2,y(6)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(7)+x(21))/2,y(6)+2,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
T(3)=TEXT/&name+' < SIZE '+&n+' >',(x(6)+x(21))/2,<#
>y(6)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name+' < SIZE '+&n+' >',(x(7)+x(21))/2,<#
>y(6)+3,0,TJST,'CJT',THGT,0.75
  OR &data='2'
IF (&name='') GOTO 200
T(3)=TEXT/&name,(x(6)+x(21))/2,y(6)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name,(x(7)+x(21))/2,y(6)+3,0,TJST,'CJT',THGT,0.75
#200 CONTINUE
ENDWHEN
T(5)=TEXT/'centre back line',x(1)+1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(6)=TEXT/'centre front line',x(3)-1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
DISP ALL
#999 END
```

```
+-----------------------------------------------------+
|                                                     |
|   The dartless easy fitting bodice block pattern (BDDL)   |
|                                                     |
+-----------------------------------------------------+
<#
<#
DECLARE REAL x(23),y(23)
DECLARE LOCATION P(23)
DECLARE ENTITY L(14),T(6),B(3),C(3),p(4)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP BDSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,ARMDPT,<#
>CHEST,NECK,&n,&data)
<#
<#
PRINT {''}
PRINT  Select the position for the centre back neck<#
> of the bodice block please.
PRINT  ------------------------------------<#
>------------------------------------
PRINT     Move the mouse on the pad to place the cursor<#
> on the graphic window
PRINT     at the position you require. Press the left<#
> button.
PRINT  ------------------------------------<#
>------------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(2)=:x(4)=:x(6)=:X
y(1)=y(20)=:Y
y(2)=y(10)=:y(17)=:y(14)=:y(3)=:y(1)-ARMDPT-2.5
x(3)=x(13)=:x(5)=:x(7)=:x(1)+BUST/2+7
y(4)=y(18)=:y(5)=:y(1)-NATOWA
y(6)=y(19)=:y(7)=:y(4)-WATOHI
x(8)=x(1)+NECK/5-0.2
y(8)=y(1)+1.5
y(9)=y(1)-ARMDPT/5+1
x(9)=x(8)+SQRT((SHLN+1)**2-(y(8)-y(9))**2)
x(10)=x(11)=:x(1)+BWD/2+1
y(11)=(y(10)+y(9))/2
x(12)=x(3)-NECK/5+0.7
WHEN  BUST<95
          y(12)=y(1)+1.5
   OR  BUST.GE.95.AND.BUST<100
          y(12)=y(1)+1.8
   OR  BUST.GE.100.AND.BUST<105
          y(12)=y(1)+2.1
   OR  BUST.GE.105.AND.BUST<110
          y(12)=y(1)+2.4
```

```
OR BUST.GE.110.AND.BUST<115
      y(12)=y(1)+2.7
OR BUST.GE.110.AND.BUST<120
      y(12)=y(1)+3
OR BUST.GE.120.AND.BUST<125
      y(12)=y(1)+3.3
OR BUST.GE.125.AND.BUST<130
      y(12)=y(1)+3.6
ELSE
      y(12)=y(1)+3.9
ENDWHEN
y(13)=y(12)-NECK/5+0.2
x(14)=x(15)=:x(3)-(CHEST/2+1.5)
y(15)=(y(3)+y(13))/2
y(16)=y(9)-0.75
x(16)=x(12)-SQRT((SHLN+0.5)**2-(y(12)-y(16))**2)
x(17)=x(18)=:x(19)=:(x(10)+x(14))/2
x(20)=x(1)+(x(8)-x(1))/3
WHEN BUST<95
        backcurve=2.5
        frontcurve=2.25
  OR BUST.GE.95.AND.BUST.LT.115
        backcurve=3
        frontcurve=2.75
  ELSE
        backcurve=3.5
        frontcurve=3.25
ENDWHEN
x(21)=x(10)+backcurve/SQRT(2)
y(21)=y(10)+backcurve/SQRT(2)
x(22)=x(14)-frontcurve/SQRT(2)
y(22)=y(14)+frontcurve/SQRT(2)
x(23)=x(8)-(x(8)-x(1))/4
y(23)=y(1)+(y(8)-y(1))/3
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>23
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
p(1)=POINT/x(11),y(11),0
p(2)=POINT/x(21),y(21),0
p(3)=POINT/x(22),y(22),0
p(4)=POINT/x(15),y(15),0
<#
<#
L(1)=LINE/P(1),P(6)
L(2)=LINE/P(6),P(19)
L(3)=LINE/P(17),P(19)
L(4)=LINE/P(8),P(9)
L(5)=LINE/P(13),P(7)
L(6)=LINE/P(7),P(19)
L(7)=LINE/P(12),P(16)
L(8)=LINE/P(2),P(17),FONT,'DASH'
L(9)=LINE/P(4),P(18),FONT,'DASH'
L(10)=LINE/P(3),P(17),FONT,'DASH'
```

```
L(11)=LINE/P(5),P(18),FONT,'DASH'
L(12)=LINE/P(1),P(20)
<#
<#
B(1)=BSPLIN/3,P(17),P(21),P(11),P(9)     <# b-armhole
B(2)=BSPLIN/3,P(17),P(22),P(15),P(16)    <# f-armhole
B(3)=BSPLIN/3,P(20),P(23),P(8)           <# b-neck
C(1)=CIRCLE/RADIUS,NECK/5-0.2,RIGHT,P(12),P(13),<#
>PGO,P(12),PEND,P(13)
<#
<#
C(2)=CIRCLE/CENTER,P(11),RADIUS,0.4
C(3)=CIRCLE/CENTER,P(15),RADIUS,0.4
L(13)=LINE/x(1)+5,y(2)+5,0,x(1)+5,y(4)-10,0,FONT,'ARROW'
L(14)=LINE/x(3)-5,y(2)+5,0,x(3)-5,y(4)-10,0,FONT,'ARROW'
<#
<#
&T1='dartless bodice block'
T(1)=TEXT/&T1,(x(6)+x(19))/2,y(6)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(7)+x(19))/2,y(6)+2,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
T(3)=TEXT/&name+' < SIZE '+&n+' >',(x(6)+x(19))/2,<#
>y(6)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name+' < SIZE '+&n+' >',(x(7)+x(19))/2,<#
>y(6)+3,0,TJST,'CJT',THGT,0.75
  OR &data='2'
IF (&name='') GOTO 200
T(3)=TEXT/&name,(x(6)+x(19))/2,y(6)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name,(x(7)+x(19))/2,y(6)+3,0,TJST,'CJT',THGT,0.75
#200 CONTINUE
ENDWHEN
T(5)=TEXT/'centre back line',x(1)+1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(6)=TEXT/'centre front line',x(3)-1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
DISP ALL
#999 END
```

```
+-----------------------------------------------------------+
|                                                           |
|          The overgarment bodice block pattern (BDOV)      |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL x(26),y(26)
DECLARE LOCATION P(26)
DECLARE ENTITY L(15),B(3),C(4),T(8),p(5)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP BDSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,<#
>ARMDPT,CHEST,NECK,&n,&data)
<#
<#
#10 CONTINUE
PRINT {''}
&bust=''
PRINT              Which bust shaping do you want?
PRINT     ----------------------------------------
PRINT     The standard bust shaping ---------------- 1
PRINT     The less bust shaping (half dart size) ---- 2
PRINT     ----------------------------------------
READ(              Enter the number please. ---> )&bust
WHEN &bust='1'.OR.&bust='2'
          CONTINUE
   ELSE
          PRINT Wrong answer, enter again please.
          GOTO 10
ENDWHEN
<#
<#
PRINT {''}
PRINT  Select the position for the centre back neck<#
> of the bodice block please.
PRINT  ------------------------------<#
>------------------------------
PRINT      Move the mouse on the pad to place the cursor<#
> on the graphic window
PRINT      at the position you require. Press the left<#
> button.
PRINT  ------------------------------<#
>------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(2)=:x(4)=:x(6)=:X
y(1)=y(23)=:Y
y(2)=y(9)=:y(20)=:y(17)=:y(19)=:y(3)=:Y-ARMDPT-4
x(3)=x(14)=:x(5)=:x(7)=:x(1)+BUST/2+10
```

```
y(4)=y(21)=:y(5)=:y(1)-NATOWA-0.5
y(6)=y(22)=:y(7)=:y(4)-WATOHI
x(8)=x(1)+NECK/5+0.3
y(8)=y(1)+2
x(9)=x(10)=:x(11)=:x(2)+BWD/2+1.5
y(10)=(y(1)+y(2))/2
y(11)=y(1)-ARMDPT/4
y(12)=y(11)+2
x(12)=x(8)+SQRT((SHLN+1.5)**2-(y(8)-y(12))**2)
x(13)=x(3)-NECK/5+0.2
WHEN BUST<95
        y(13)=y(15)=:y(1)+2
OR BUST.GE.95.AND.BUST<100
        y(13)=y(15)=:y(1)+2.3
OR BUST.GE.100.AND.BUST<105
        y(13)=y(15)=:y(1)+2.6
OR BUST.GE.105.AND.BUST<110
        y(13)=y(15)=:y(1)+2.9
OR BUST.GE.110.AND.BUST<115
        y(13)=y(15)=:y(1)+3.2
OR BUST.GE.115.AND.BUST<120
        y(13)=y(15)=:y(1)+3.5
OR BUST.GE.120.AND.BUST<125
        y(13)=y(15)=:y(1)+3.8
OR BUST.GE.125.AND.BUST<130
        y(13)=y(15)=:y(1)+4.1
ELSE
        y(13)=y(15)=:y(1)+4.4
ENDWHEN
y(14)=y(13)-NECK/5-0.3
WHEN &bust='1'
        x(15)=x(13)-DART
  ELSE
        x(15)=x(13)-DART/2
ENDWHEN
shln=SHLN+0.5
x(16)=x(15)-shln*(x(15)-x(11))/SQRT((x(15)-x(11))**2<#
>+(y(15)-y(11))**2)
WHEN &bust='1'
y(16)=y(15)-shln*(y(15)-y(11))/SQRT((x(15)-x(11))**2<#
>+(y(15)-y(11))**2)-1.5
x(17)=x(18)=:x(3)-CHEST/2-DART/2-1
  ELSE
y(16)=y(15)-shln*(y(15)-y(11))/SQRT((x(15)-x(11))**2<#
>+(y(15)-y(11))**2)-1
x(17)=x(18)=:x(3)-CHEST/2-DART/4-1.5
ENDWHEN
y(18)=y(17)+(y(14)-y(3))/3
x(19)=(x(3)+x(17))/2
x(20)=x(21)=:x(22)=:(x(9)+x(17))/2
x(23)=x(1)+(x(8)-x(1))/3
WHEN BUST<95
        backcurve=2.75
        frontcurve=2.25
  OR BUST.GE.95.AND.BUST.LT.115
        backcurve=3.25
        frontcurve=2.75
  ELSE
```

```
        backcurve=3.75
        frontcurve=3.25
ENDWHEN
x(24)=x(9)+backcurve/SQRT(2)
y(24)=y(9)+backcurve/SQRT(2)
x(25)=x(17)-frontcurve/SQRT(2)
y(25)=y(17)+frontcurve/SQRT(2)
x(26)=x(8)-(x(8)-x(1))/4
y(26)=y(1)+(y(8)-y(1))/3
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>26
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
p(1)=POINT/x(10),y(10),0
p(2)=POINT/x(24),y(24),0
p(3)=POINT/x(18),y(18),0
p(4)=POINT/x(25),y(25),0
<#
<#
L(1)=LINE/P(1),P(6)
L(2)=LINE/P(6),P(22)
L(3)=LINE/P(20),P(22)
L(4)=LINE/P(8),P(12)
L(5)=LINE/P(14),P(7)
L(6)=LINE/P(7),P(22)
L(7)=LINE/P(13),P(19)
L(8)=LINE/P(15),P(19)
L(9)=LINE/P(2),P(20),FONT,'DASH'
L(10)=LINE/P(4),P(21),FONT,'DASH'
L(11)=LINE/P(3),P(20),FONT,'DASH'
L(12)=LINE/P(5),P(21),FONT,'DASH'
L(13)=LINE/P(1),P(23)
<#
<#
B(1)=BSPLIN/3,P(20),P(24),P(10),P(12)          <# b-armhole
$L1=LINE/P(16),P(18)
$L2=LINE/(x(16)+x(18))/2,(y(16)+y(18))/2,0,<#
>PERP,$L1,LENGTH,0.5
p(5)=POINT/(x(16)+x(18))/2,(y(16)+y(18))/2,0,<#
>DISTAN,0.5,$L2
DELETE $L1,$L2
B(2)=BSPLIN/3,P(20),P(25),P(18),p(5),P(16)   <# f-armhole
B(3)=BSPLIN/3,P(23),P(26),P(8)                <# b-neck
C(1)=CIRCLE/RADIUS,NECK/5+0.3,RIGHT,P(13),P(14),<#
>PGO,P(13),PEND,P(14)
C(2)=CIRCLE/RADIUS,BUST,RIGHT,P(15),P(16),<#
>PGO,P(15),PEND,P(16)                         <# f-shoulder
<#
<#
C(3)=CIRCLE/CENTER,P(10),RADIUS,0.4
C(4)=CIRCLE/CENTER,P(18),RADIUS,0.4
L(14)=LINE/x(1)+5,y(2)+5,0,x(1)+5,y(4)-10,0,FONT,'ARROW'
L(15)=LINE/x(3)-5,y(2)+5,0,x(3)-5,y(4)-10,0,FONT,'ARROW'
<#
```

```
<#
&T1='overgarment block'
T(1)=TEXT/&T1,(x(6)+x(22))/2,y(6)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(7)+x(22))/2,y(6)+2,0,TJST,'CJT',THGT,0.75
WHEN &bust='1'
          &T2='standard bust shaping'
  ELSE
          &T2='less bust shaping'
ENDWHEN
T(3)=TEXT/&T2,(x(6)+x(22))/2,y(6)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&T2,(x(7)+x(22))/2,y(6)+3,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
T(5)=TEXT/&name+' < SIZE '+&n+' >',(x(6)+x(22))/2,y(6)+4,<#
>0,TJST,'CJT',THGT,0.75
T(6)=TEXT/&name+' < SIZE '+&n+' >',(x(7)+x(22))/2,y(6)+4,<#
>0,TJST,'CJT',THGT,0.75
  ELSE
IF (&name='') GOTO 200
T(5)=TEXT/&name,(x(6)+x(22))/2,y(6)+4,0,TJST,'CJT',THGT,0.75
T(6)=TEXT/&name,(x(7)+x(22))/2,y(6)+4,0,TJST,'CJT',THGT,0.75
#200 CONTINUE
ENDWHEN
T(7)=TEXT/'centre back line',x(1)+1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(8)=TEXT/'centre front line',x(3)-1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
DISP ALL
#999 END
```

+--+
| |
| The tailored jacket bodice block pattern (BDTA) |
| |
+--+

```
<#
<#
DECLARE REAL x(54),y(54)
DECLARE LOCATION P(54)
DECLARE ENTITY L(37),T(8),B(5),C(5),p(5),cp(1)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP BDSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,<#
>ARMDPT,CHEST,NECK,&n,&data)
<#
<#
#10 CONTINUE
PRINT {''}
&bust=''
PRINT              Which bust shaping do you want?
PRINT    ----------------------------------------
PRINT    The standard bust shaping ---------------- 1
PRINT    The less bust shaping (half dart size) ---- 2
PRINT    ----------------------------------------
READ(              Enter the number please. ---> )&bust
WHEN &bust='1'.OR.&bust='2'
         CONTINUE
  ELSE
         PRINT Wrong answer, enter again please.
         GOTO 10
ENDWHEN
<#
<#
PRINT {''}
PRINT  Select the position for the centre back neck <#
>of the bodice block please.
PRINT  ----------------------------------<#
>--------------------------------
PRINT     Move the mouse on the pad to place the cursor<#
> on the graphic window
PRINT     at the position you require. Press the left<#
> button.
PRINT  ----------------------------------<#
>--------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(6)=:x(5)=:x(2)=:x(4)=:x(3)=:X
y(1)=y(37)=:Y
y(2)=y(26)=:y(23)=:y(13)=:y(30)=:y(46)=:y(48)=:<#
>y(51)=:y(52)=:y(1)-NATOWA
```

```
y(3)=y(28)=:y(14)=:y(31)=:y(41)=:y(47)=:y(49)=:<#
>y(1)-NATOWA*8/5
y(4)=y(27)=:y(24)=:y(36)=:y(2)-WATOHI
y(5)=y(7)=:y(25)=:y(20)=:y(22)=:y(12)=:y(29)=:<#
>y(42)=:y(1)-ARMDPT-2.5
y(6)=y(8)=:(y(1)+y(5))/2
x(7)=x(8)=:x(9)=:x(1)+BWD/2+1
y(9)=y(1)-ARMDPT/4
x(10)=x(1)+NECK/5
y(10)=y(1)+1.75
y(11)=y(9)+2
x(11)=x(10)+SQRT((SHLN+1.5)**2-(y(10)-y(11))**2)
x(12)=x(16)=:x(13)=:x(14)=:x(1)+BUST/2+8
WHEN &bust='1'
        x(15)=x(12)-NECK/5-1
  ELSE
        x(15)=x(12)-NECK/5-2
ENDWHEN
WHEN BUST<95
        y(15)=y(10)
  OR BUST.GE.95.AND.BUST<100
        y(15)=y(10)+0.3
  OR BUST.GE.100.AND.BUST<105
        y(15)=y(10)+0.6
  OR BUST.GE.105.AND.BUST<110
        y(15)=y(10)+0.9
  OR BUST.GE.110.AND.BUST<115
        y(15)=y(10)+1.2
  OR BUST.GE.115.AND.BUST<120
        y(15)=y(10)+1.5
  OR BUST.GE.120.AND.BUST<125
        y(15)=y(10)+1.8
  OR BUST.GE.125.AND.BUST<130
        y(15)=y(10)+2.1
  ELSE
        y(15)=y(10)+2.4
ENDWHEN
y(16)=y(15)-NECK/5
WHEN &bust='1'
        dart=DART
  ELSE
        dart=DART/2
ENDWHEN
shln=SHLN+dart+0.5
x(17)=x(15)-shln*(x(15)-x(9))/SQRT((x(15)-x(9))**2<#
>+(y(15)-y(9))**2)
y(17)=y(15)-shln*(y(15)-y(9))/SQRT((x(15)-x(9))**2<#
>+(y(15)-y(9))**2)-2
x(18)=x(15)-(SHLN/3)*(x(15)-x(9))/<#
>SQRT((x(15)-x(9))**2+(y(15)-y(9))**2)
y(18)=y(15)-(SHLN/3)*(y(15)-y(9))/<#
>SQRT((x(15)-x(9))**2+(y(15)-y(9))**2)
x(19)=x(18)-dart*(x(18)-x(9))/SQRT((x(18)-x(9))**2<#
>+(y(18)-y(9))**2)
y(19)=y(18)-dart*(y(18)-y(9))/SQRT((x(18)-x(9))**2<#
>+(y(18)-y(9))**2)
WHEN &bust='1'
        x(20)=x(21)=:x(12)-CHEST/2-dart/2-1
```

```
  ELSE
        x(20)=x(21)=:x(12)-CHEST/2-dart/2-1.5
ENDWHEN
y(21)=y(20)+(y(16)-y(12))/3
x(22)=x(23)=:x(24)=:x(35)=:x(50)=:x(53)=:(x(12)+x(20))/2
x(25)=x(26)=:x(27)=:x(28)=:(x(7)+x(20))/2
x(29)=x(30)=:x(33)=:x(36)=:x(31)=:x(32)=:x(12)+BUST/30
y(32)=y(34)=:y(35)=:y(3)-1
y(33)=y(32)+(y(30)-y(32))*2/3
x(34)=x(32)-(x(32)-x(28))/5
x(37)=x(1)+(x(10)-x(1))/3
WHEN BUST<95
        backcurve=2.25
        frontcurve=2
  OR BUST.GE.95.AND.BUST.LT.115
        backcurve=2.75
        frontcurve=2.5
  ELSE
        backcurve=3.25
        frontcurve=3
ENDWHEN
x(38)=x(7)+backcurve/SQRT(2)
y(38)=y(7)+backcurve/SQRT(2)
x(39)=x(20)-frontcurve/SQRT(2)
y(39)=y(20)+frontcurve/SQRT(2)
x(40)=x(1)+1.5
y(40)=y(2)-1.5
x(41)=x(1)+0.5
x(42)=x(45)=:(x(5)+x(7))/2
x(43)=x(42)-1
y(43)=y(2)-1.3
x(44)=x(42)+1
y(44)=y(2)-1.2
y(45)=y(4)+7
x(46)=x(25)-1.5
x(47)=x(25)+1.5
x(48)=x(25)+2
x(49)=x(25)-1
y(50)=y(22)-3
x(51)=x(22)-1.5
x(52)=x(22)+1.5
y(53)=y(4)+5
x(54)=x(10)-(x(10)-x(1))/4
y(54)=y(1)+(y(10)-y(1))/3
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>54
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
p(1)=POINT/x(8),y(8),0
p(2)=POINT/x(38),y(38),0
p(3)=POINT/x(21),y(21),0
p(4)=POINT/x(39),y(39),0
<#
<#
```

```
L(1)=LINE/P(1),P(3)
L(2)=LINE/P(3),P(28)
L(3)=LINE/P(10),P(11)
L(4)=LINE/P(1),P(37)
L(5)=LINE/P(4),P(27),FONT,'DASH'
L(6)=LINE/P(5),P(25),FONT,'DASH'
L(7)=LINE/P(2),P(26),FONT,'DASH'
L(8)=LINE/P(6),P(40)
L(9)=LINE/P(40),P(41)
L(10)=LINE/P(41),P(47)
L(11)=LINE/P(46),P(47)          <# b-waist shaping
L(12)=LINE/P(25),P(46)          <# b-waist shaping
L(13)=LINE/P(42),P(43)          <# b-waist dart
L(14)=LINE/P(43),P(45)          <# b-waist dart
L(15)=LINE/P(42),P(44)          <# b-waist dart
L(16)=LINE/P(44),P(45)          <# b-waist dart
L(17)=LINE/P(25),P(28)
L(18)=LINE/P(16),P(14)
L(19)=LINE/P(28),P(14)
L(20)=LINE/P(15),P(18)
L(21)=LINE/P(18),P(22)
L(22)=LINE/P(19),P(22)
L(23)=LINE/P(29),P(33)
L(24)=LINE/P(29),P(25),FONT,'DASH'
L(25)=LINE/P(30),P(26),FONT,'DASH'
L(26)=LINE/P(36),P(27),FONT,'DASH'
L(27)=LINE/P(22),P(35),FONT,'DASH'
L(28)=LINE/P(33),P(32),FONT,'DASH'
L(29)=LINE/P(32),P(34),FONT,'DASH'
L(30)=LINE/P(25),P(48)          <# f-waist shaping
L(31)=LINE/P(48),P(49)          <# f-waist shaping
L(32)=LINE/P(50),P(51)          <# f-waist dart
L(33)=LINE/P(51),P(53)          <# f-waist dart
L(34)=LINE/P(50),P(52)          <# f-waist dart
L(35)=LINE/P(52),P(53)          <# f-waist dart
<#
<#
B(1)=BSPLIN/3,P(25),P(38),P(8),P(11)    <# b-armhole
B(2)=BSPLIN/3,P(37),P(54),P(10)         <# b-neck
$L1=LINE/P(17),P(21)
$L2=LINE/(x(17)+x(21))/2,(y(17)+y(21))/2,0,<#
>PERP,$L1,LENGTH,0.5
p(5)=POINT/(x(17)+x(21))/2,(y(17)+y(21))/2,0,<#
>DISTAN,0.5,$L2
B(3)=BSPLIN/3,P(25),P(39),P(21),p(5),P(17) <# f-armhole
DELETE $L1,$L2
B(4)=BSPLIN/3,P(34),P(35),P(49)          <# f-hemline
B(5)=BSPLIN/3,P(40),P(43),P(44),P(46),FONT,'DASH'
WHEN &dart='1'
        C(1)=CIRCLE/RADIUS,NECK/5+1,RIGHT,P(15),<#
>P(16),PGO,P(15),PEND,P(16)             <# f-neck
  ELSE
        C(1)=CIRCLE/RADIUS,NECK/5+2,RIGHT,P(15),<#
>P(16),PGO,P(15),PEND,P(16)             <# f-neck
ENDWHEN
C(2)=CIRCLE/RADIUS,BUST,RIGHT,P(19),P(17),<#
>PGO,P(19),PEND,P(17)                   <# f-shoulder
C(3)=CIRCLE/RADIUS,BUST/3,LEFT,P(34),P(33),<#
```

```
>PGO,P(34),PEND,P(33)                          <# f-edge
<#
<#
C(4)=CIRCLE/CENTER,P(21),RADIUS,0.4
C(5)=CIRCLE/CENTER,P(8),RADIUS,0.4
L(36)=LINE/x(1)+5,y(5)+5,0,x(1)+5,y(2)-10,0,FONT,'ARROW'
L(37)=LINE/x(13)-5,y(5)+5,0,x(13)-5,y(2)-10,0,FONT,'ARROW'
<#
<#
&T1='tailored jacket block'
T(1)=TEXT/&T1,(x(4)+x(27))/2,y(4)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(27)+x(12))/2,y(4)+2,0,TJST,'CJT',THGT,0.75
WHEN &bust='1'
        &T2='standard bust shaping'
  ELSE
        &T2='less bust shaping'
ENDWHEN
T(3)=TEXT/&T2,(x(4)+x(27))/2,y(4)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&T2,(x(27)+x(12))/2,y(4)+3,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
T(5)=TEXT/&name+' < SIZE '+&n+' >',(x(4)+x(27))/2,<#
>y(4)+4,0,TJST,'CJT',THGT,0.75
T(6)=TEXT/&name+' < SIZE '+&n+' >',(x(27)+x(12))/2,<#
>y(4)+4,0,TJST,'CJT',THGT,0.75
  OR &data='2'
IF (&name='') GOTO 200
T(5)=TEXT/&name,(x(4)+x(27))/2,y(4)+4,0,<#
>TJST,'CJT',THGT,0.75
T(6)=TEXT/&name,(x(27)+x(12))/2,y(4)+4,0,<#
>TJST,'CJT',THGT,0.75
#200 CONTINUE
ENDWHEN
T(7)=TEXT/'centre back line',x(1)+1,(y(1)+y(2))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(8)=TEXT/'centre front line',x(12)-1,(y(1)+y(2))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
DISP ALL
#999 END
```

```
+-----------------------------------------------------------+
|                                                           |
|                The classic shirt block (BDSH)             |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL x(32),y(32)
DECLARE LOCATION P(32)
DECLARE ENTITY PNT(3),L(21),B(3),C(5),T(7),cp(3),p(4)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP BDSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,ARMDPT<#
>,CHEST,NECK,&n,&data)
<#
<#
PRINT {''}
PRINT  Select the position for the centre back neck<#
> of the bodice block please.
PRINT  ------------------------------------<#
>------------------------------------
PRINT      Move the mouse on the pad to place the cursor<#
> on the graphic window
PRINT      at the position you require. Press the left<#
> button.
PRINT  ------------------------------------<#
>------------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(10)=:x(2)=:x(3)=:x(4)=:X
y(1)=y(9)=:Y
y(2)=y(11)=:y(21)=:y(19)=:y(5)=:y(1)-ARMDPT-2.5
y(3)=y(22)=:y(7)=:y(1)-NATOWA
y(4)=y(28)=:y(23)=:y(6)=:y(25)=:y(26)=:y(3)-WATOHI
x(5)=x(17)=:x(7)=:x(6)=:x(1)+BUST/2+9.5
x(8)=x(1)+NECK/5+0.2
y(8)=y(1)+3.5
x(9)=x(1)+(x(8)-x(1))/3
y(10)=y(27)=:y(14)=:y(12)=:y(1)-(ARMDPT+2.5)/5-1
x(11)=x(12)=:x(15)=:x(32)=:x(1)+BWD/2+2.5
x(13)=x(12)+1.25
y(13)=y(8)-ARMDPT/5+0.5
x(14)=(x(10)+x(12))/2+1.5
y(15)=y(12)-0.5
x(16)=x(5)-NECK/5+0.6
y(16)=y(8)-5
y(17)=y(24)=:y(16)-NECK/5+1.6
y(18)=y(16)-ARMDPT/5-0.5
x(18)=x(16)-SQRT((x(13)-x(8))**2+(y(8)-y(13))**2-<#
>(y(16)-y(18))**2)
```

```
x(19)=x(20)=:x(5)-x(11)+x(1)+0.2
y(20)=(y(5)+y(17))/2-1
x(21)=x(22)=:x(23)=:(x(11)+x(19))/2
x(24)=x(25)=:x(17)+1.5
x(26)=x(25)+3.5
x(27)=x(28)=:x(10)-2
WHEN BUST<95
        backcurve=2.75
        frontcurve=2.25
  OR BUST.GE.95.AND.BUST.LT.115
        backcurve=3
        frontcurve=2.5
  ELSE
        backcurve=3.25
        frontcurve=2.75
ENDWHEN
x(29)=x(11)+backcurve/SQRT(2)
y(29)=y(11)+backcurve/SQRT(2)
x(30)=x(19)-frontcurve/SQRT(2)
y(30)=y(19)+frontcurve/SQRT(2)
x(31)=x(8)-(x(8)-x(1))/4
y(31)=y(1)+(y(8)-y(1))/3
y(32)=(y(1)+y(2))/2
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>32
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
p(1)=POINT/P(32)
p(2)=POINT/P(29)
p(3)=POINT/P(30)
p(4)=POINT/P(20)
<#
<#
L(1)=LINE/P(1),P(10)
L(2)=LINE/P(8),P(13)
L(3)=LINE/P(27),P(28)
L(4)=LINE/P(27),P(14)
L(5)=LINE/P(28),P(23)
L(6)=LINE/P(21),P(23)
L(7)=LINE/P(10),P(4)
L(8)=LINE/P(2),P(21),FONT,'DASH'
L(9)=LINE/P(3),P(22),FONT,'DASH'
L(10)=LINE/P(16),P(18)
L(11)=LINE/P(17),P(6)
L(12)=LINE/P(24),P(25),FONT,'DASH'
L(13)=LINE/P(23),P(26)
L(14)=LINE/P(5),P(21),FONT,'DASH'
L(15)=LINE/P(7),P(22),FONT,'DASH'
L(16)=LINE/P(17),P(24)
L(17)=LINE/P(1),P(9)
<#
<#
B(1)=BSPLIN/3,P(21),P(29),P(32),P(13)     <# b-armhole
B(2)=BSPLIN/3,P(21),P(30),P(20),P(18)     <# f-armhole
```

```
B(3)=BSPLIN/3,P(9),P(31),P(8)                    <# b-neck
C(1)=CIRCLE/RADIUS,NECK/5-0.6,RIGHT,P(16),P(17),<#
>PGO,P(16),PEND,P(17)
$E=LINE/P(12),HORIZ,LENGTH,3
PNT(1)=POINT/INTOF,$E,XSMALL,B(1)
L(20)=LINE/P(10),PNT(1)
DELETE $E,PNT(1)
$E=LINE/P(15),HORIZ,LENGTH,3
PNT(2)=POINT/INTOF,$E,XSMALL,B(1)
C(2)=CIRCLE/RADIUS,BUST,LEFT,PNT(2),P(14),<#
>PGO,PNT(2),PEND,P(14)                           <# b-yoke
DELETE $E,PNT(2)
$P=POINT/P(17)
MIRROR (COPY,cp(1)) C(1),L(16),$P,x(24),y(24),0,x(25),y(25),0
$E=LINE/P(26),VERT,P(16)
PNT(3)=POINT/INTOF,$E,FIRST,cp(1)
L(21)=LINE/PNT(3),P(26)
DELETE $E,cp(1)
C(3)=CIRCLE/RADIUS,NECK/5-0.6,LEFT,cp(3),PNT(3),<#
>PGO,cp(3),PEND,PNT(3)
DELETE $P,cp(3),PNT(3)
<#
<#
<#
C(4)=CIRCLE/CENTER,P(32),RADIUS,0.4
C(5)=CIRCLE/CENTER,P(20),RADIUS,0.4
L(18)=LINE/x(1)+5,y(2)+5,0,x(1)+5,y(3)-10,0,FONT,'ARROW'
L(19)=LINE/x(6)-5,y(2)+5,0,x(6)-5,y(3)-10,0,FONT,'ARROW'
<#
<#
&T1='shirt block'
T(1)=TEXT/&T1,(x(4)+x(23))/2,y(4)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(6)+x(23))/2,y(4)+2,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
T(3)=TEXT/&name+' < SIZE '+&n+' >',(x(4)+x(23))/2,<#
>y(4)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name+' < SIZE '+&n+' >',(x(6)+x(23))/2,<#
>y(4)+3,0,TJST,'CJT',THGT,0.75
ELSE
IF (&name='') GOTO 200
T(3)=TEXT/&name,(x(4)+x(23))/2,y(4)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name,(x(6)+x(23))/2,y(4)+3,0,TJST,'CJT',THGT,0.75
#200 CONTINUE
ENDWHEN
T(5)=TEXT/'centre back line',x(1)+1,(y(1)+y(3))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(6)=TEXT/'centre front line',x(5)-1,(y(1)+y(3))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
T(7)=TEXT/'fold line',x(24),(y(1)+y(3))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
!REGENERATE GRAPHICS
EXECDF
DISP ALL
#999 END
```

```
+-----------------------------------------------------------+
|                                                           |
|     The bodice block pattern for jersey garment (BDJE)    |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL x(15),y(15)
DECLARE LOCATION P(15)
DECLARE ENTITY cp(9),p(2),L(9),B(2),C(2),T(6)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP BDSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,ARMDPT<#
>,CHEST,NECK,&n,&data)
<#
<#
PRINT {''}
PRINT  Select the position for the centre back neck<#
> of the bodice block please.
PRINT  ------------------------------------<#
>----------------------------------
PRINT     Move the mouse on the pad to place the cursor <#
>on the graphic window
PRINT     at the position you require. Press the left <#
>button.
PRINT  ------------------------------------<#
>----------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(12)=:x(4)=:x(2)=:x(3)=:X
y(1)=y(13)=:Y
y(2)=y(5)=:y(1)-NATOWA
y(3)=y(11)=:y(2)-WATOHI
y(4)=y(7)=:y(10)=:y(1)-ARMDPT-1
x(5)=x(10)=:x(11)=:x(1)+BUST/4+2
x(6)=x(1)+NECK/5+0.25
y(6)=y(1)+1.5
x(7)=x(8)=:x(1)+BWD/2+0.5
y(8)=(y(1)+y(4))/2
y(9)=y(1)-(y(1)-y(8))/4
x(9)=x(8)+0.5
y(12)=y(1)-NECK/5+1.5
x(13)=x(1)+(x(6)-x(1))/3
x(14)=x(6)-(x(6)-x(1))/4
y(14)=y(1)+(y(6)-y(1))/3
WHEN BUST<95
          backcurve=2.75
   OR BUST.GE.95.AND.BUST.LT.115
          backcurve=3.25
   ELSE
```

```
          backcurve=3.75
ENDWHEN
x(15)=x(7)+backcurve/SQRT(2)
y(15)=y(7)+backcurve/SQRT(2)
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>15
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
p(1)=POINT/P(8)
p(2)=POINT/P(15)
<#
<#
L(1)=LINE/P(1),P(3)
L(2)=LINE/P(10),P(11)
L(3)=LINE/P(3),P(11)
L(4)=LINE/P(6),P(9)
L(5)=LINE/P(4),P(10),FONT,'DASH'
L(6)=LINE/P(2),P(5),FONT,'DASH'
L(8)=LINE/P(1),P(13)
L(9)=LINE/P(12),P(3)
<#
<#
B(1)=BSPLIN/3,P(10),P(15),P(8),P(9)          <# armhole
B(2)=BSPLIN/3,P(13),P(14),P(6)                <# b-neck
C(1)=CIRCLE/RADIUS,NECK/5+0.25,LEFT,P(12),P(6),<#
>PGO,P(12),PEND,P(6)
<#
<#
C(2)=CIRCLE/CENTER,P(8),RADIUS,0.4
L(7)=LINE/x(1)+5,y(4)+5,0,x(1)+5,y(2)-10,0,FONT,'ARROW'
<#
<#
MIRROR (COPY,cp(1)) 5,L(3),MX,x(10),0,0
MIRROR (COPY,cp(6)) B(1),MX,x(10),0,0
MIRROR (COPY,cp(7)) C(2),MX,x(10),0,0
MIRROR (COPY,cp(8)) 2,p(1),MX,x(10),0,0
MIRROR L(9),MX,x(10),0,0
MIRROR C(1),MX,x(10),0,0
<#
<#
&T1='jersey wear block'
T(1)=TEXT/&T1,(x(3)+x(11))/2,y(3)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(3)+(x(11)-x(3))*3/2),y(3)+2,0,<#
>TJST,'CJT',THGT,0.75
WHEN &data='1'
T(3)=TEXT/&name+' < SIZE '+&n+' >',(x(3)+x(11))/2,<#
>y(3)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name+' < SIZE '+&n+' >',(x(3)+<#
>(x(11)-x(3))*3/2),y(3)+3,0,<#
>TJST,'CJT',THGT,0.75
  ELSE
IF (&name='') GOTO 200
T(3)=TEXT/&name,(x(3)+x(11))/2,y(3)+3,0,<#
>TJST,'CJT',THGT,0.75
```

```
T(4)=TEXT/&name,x(3)+(x(11)-x(3))*3/2,y(3)+3,0,<#
>TJST,'CJT',THGT,0.75
#200 CONTINUE
ENDWHEN
T(5)=TEXT/'centre back line',x(1)+1,(y(1)+y(2))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(6)=TEXT/'centre front line',x(1)+(x(5)-x(1))*2-1,<#
>(y(1)+y(2))/2,0,TJST,'CJT',THGT,0.75,TANG,90
<#
<#
!REGENERATE GRAPHICS
EXECDF
DISP ALL
#999 END
```

```
+-----------------------------------------------------------+
|                                                           |
|    The bodice block pattern for knitted garment (BDKN)    |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL x(15),y(15)
DECLARE LOCATION P(15)
DECLARE ENTITY cp(9),p(2),L(9),B(2),C(2),T(6)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP BDSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,<#
>ARMDPT,CHEST,NECK,&n,&data)
<#
<#
PRINT {''}
PRINT  Select the position for the centre back neck<#
> of the bodice block please.
PRINT  ------------------------------------<#
>------------------------------------
PRINT     Move the mouse on the pad to place the cursor<#
> on the graphic window
PRINT     at the position you require. Press the left <#
>button.
PRINT  ------------------------------------<#
>------------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(12)=:x(4)=:x(2)=:x(3)=:X
y(1)=y(13)=:Y
y(2)=y(5)=:y(1)-NATOWA
y(3)=y(11)=:y(2)-WATOHI
y(4)=y(7)=:y(10)=:y(1)-ARMDPT-3
x(5)=x(10)=:x(11)=:x(1)+BUST/4+4.5
x(6)=x(1)+NECK/5+0.25
y(6)=y(1)+1.5
x(7)=x(8)=:x(1)+BWD/2+1.25
y(8)=(y(1)+y(4))/2
y(9)=y(1)-(y(1)-y(8))/4
x(9)=x(8)+0.5
y(12)=y(1)-NECK/5+1.5
x(13)=x(1)+(x(6)-x(1))/3
x(14)=x(6)-(x(6)-x(1))/4
y(14)=y(1)+(y(6)-y(1))/3
WHEN BUST<95
          backcurve=3
  OR BUST.GE.95.AND.BUST.LT.115
          backcurve=3.5
  ELSE
```

```
        backcurve=4
ENDWHEN
x(15)=x(7)+backcurve/SQRT(2)
y(15)=y(7)+backcurve/SQRT(2)
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>15
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
p(1)=POINT/P(8)
p(2)=POINT/P(15)
<#
<#
L(1)=LINE/P(1),P(3)
L(2)=LINE/P(10),P(11)
L(3)=LINE/P(3),P(11)
L(4)=LINE/P(6),P(9)
L(5)=LINE/P(4),P(10),FONT,'DASH'
L(6)=LINE/P(2),P(5),FONT,'DASH'
L(8)=LINE/P(1),P(13)
L(9)=LINE/P(12),P(3)
<#
<#
B(1)=BSPLIN/3,P(10),P(15),P(8),P(9)          <# armhole
B(2)=BSPLIN/3,P(13),P(14),P(6)               <# b-neck
C(1)=CIRCLE/RADIUS,NECK/5+0.25,LEFT,P(12),P(6),<#
>PGO,P(12),PEND,P(6)
<#
<#
C(2)=CIRCLE/CENTER,P(8),RADIUS,0.4
L(7)=LINE/x(1)+5,y(4)+5,0,x(1)+5,y(2)-10,0,FONT,'ARROW'
<#
<#
MIRROR (COPY,cp(1)) 5,L(3),MX,x(10),0,0
MIRROR (COPY,cp(6)) B(1),MX,x(10),0,0
MIRROR (COPY,cp(7)) C(2),MX,x(10),0,0
MIRROR (COPY,cp(8)) 2,p(1),MX,x(10),0,0
MIRROR L(9),MX,x(10),0,0
MIRROR C(1),MX,x(10),0,0
<#
<#
&T1='knitwear block'
T(1)=TEXT/&T1,(x(3)+x(11))/2,y(3)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(3)+(x(11)-x(3))*3/2),y(3)+2,0,<#
>TJST,'CJT',THGT,0.75
WHEN &data='1'
T(3)=TEXT/&name+' < SIZE '+&n+' >',(x(3)+x(11))/2,<#
>y(3)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name+' < SIZE '+&n+' >',<#
>(x(3)+(x(11)-x(3))*3/2),y(3)+3,0,TJST,'CJT',THGT,0.75
  ELSE
IF (&name='') GOTO 200
T(3)=TEXT/&name,(x(3)+x(11))/2,y(3)+3,0,<#
>TJST,'CJT',THGT,0.75
T(4)=TEXT/&name,x(3)+(x(11)-x(3))*3/2,y(3)+3,0,<#
```

```
>TJST,'CJT',THGT,0.75
#200 CONTINUE
ENDWHEN
T(5)=TEXT/'centre back line',x(1)+1,(y(1)+y(2))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(6)=TEXT/'centre front line',x(1)+(x(5)-x(1))*2-1,<#
>(y(1)+y(2))/2,0,TJST,'CJT',THGT,0.75,TANG,90
<#
<#
!REGENERATE GRAPHICS
EXECDF
DISP ALL
#999 END
```

SLONELK (Linkfile: SLONE, SLSIZE)

```
MAIN SLONE
IMAGE SLONEIMAGE
LINKM SLSIZE
ENDLINK
```

SLTWOLK (Linkfile: SLTWO, SLSIZE)

```
MAIN SLTWO
IMAGE SLTWOIMAGE
LINKM SLSIZE
ENDLINK
```

SLEALK (Linkfile : SLEA, SLSIZE)

```
MAIN SLEA
IMAGE SLEAIMAGE
LINKM SLSIZE
ENDLINK
```

SLSHLK (linkfile: SLSH, SLSIZE)

```
MAIN SLSH
IMAGE SLSHIMAGE
LINKM SLSIZE
ENDLINK
```

```
+-----------------------------------------------------------+
|                                                           |
|              The sleeve size (SLSIZE) procedure           |
|                                                           |
+-----------------------------------------------------------+
<#
<#
PROC SLSIZE(SLLNGT,CUFFTWO,CUFFSH,WRIST,ELBOW,<#
>&n,&unit,&data,&type)
DECLARE REAL sllngt(12),cufftwo(12),cuffsh(12),<#
>wrist(12),elbow(12)
DATA sllngt/57.2,57.8,58.4,59,59.5,60,60.5,61,<#
>61.2,61.4,61.6,61.8/
DATA cufftwo/13.25,13.5,13.75,14,14.25,14.5,<#
>14.75,15,15.25,15.5,15.75,16/
DATA cuffsh/21,21,21.5,21.5,22,22.5,23,23.5,<#
>24,24.5,25,25.5/
DATA wrist/15,15.5,16,16.5,17,17.5,18,18.5,<#
>19,19.5,20,20.5/
DATA elbow/18.5,18.5,18.5,18.5,18.5,18.5,18.5,<#
>18.5,18.3,17.8,17.3,16.8/
ERTRAP 999
<#
<#
#10 CONTINUE
PRINT {''}
&data=''
PRINT        Which measurements do you want to use?
PRINT     ----------------------------------------
PRINT     The body measurements of this program --- 1
PRINT     Your individual body measurements ------- 2
PRINT     ----------------------------------------
READ(              Enter the number please. ---> )&data
WHEN &data='1'.OR.&data='2'
         CONTINUE
  ELSE
         PRINT Wrong answer, enter again please.
         GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
PRINT {''}
&unit=''
READ( Which units do you want to use?  <#
>  Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'.OR.&unit='IN'.OR.&unit='in'
         CONTINUE
  ELSE
         PRINT Wrong answer, enter again please.
         GOTO 20
ENDWHEN
<#
<#
#30 CONTINUE
PRINT {''}
&n=''
WHEN &unit='CM'.OR.&unit='cm'
```

```
PRINT                  WOMEN OF MEDIUM HEIGHT 160CM-170CM<#
>   (UNIT: CM)
PRINT ------------------------------------------------<#
>--------------------------
PRINT                      8    10    12    14    16    18    20   <#
> 22    24    26    28    30
PRINT ------------------------------------------------<#
>--------------------------
PRINT SLEEVE LENGTH 57.2 57.8 58.4 59    59.5 60    60.5<#
> 61    61.2 61.4 61.6 61.8
PRINT WRIST           15    15.5 16    16.5 17    17.5 18   <#
> 18.5 19    19.5 20    20.5
PRINT ------------------------------------------------<#
>--------------------------
PRINT {''}
  ELSE
PRINT                  WOMEN OF MEDIUM HEIGHT 5FT 3IN -<#
> 5FT 7IN (UNIT: IN)
PRINT ------------------------------------------------<#
>--------------------------
PRINT                      8    10    12    14    16    18    20   <#
> 22    24    26    28    30
PRINT ------------------------------------------------<#
>--------------------------
PRINT SLEEVE LENGTH 22.5 22.8 23    23.2 23.4 23.6 23.8 <#
>24    24.1 24.2 24.3 24.3
PRINT WRIST           5.9   6.1   6.3   6.5   6.7   6.9   7.1 <#
> 7.3   7.5   7.7   7.9   8.1
PRINT ------------------------------------------------<#
>--------------------------
PRINT {''}
ENDWHEN
PRINT  The size must be an even number between 8 and 30.
WHEN &data='1'
READ(            Enter the size please. ---> )&n
ELSE
READ(         Enter the nearest size please. ---> )&n
ENDWHEN
WHEN &n='8'.OR.&n='10'.OR.&n='12'.OR.&n='14'.OR.&n='16'<#
>.OR.&n='18'.OR.&n='20'.OR.&n='22'.OR.&n='24'.OR.&n='26'<#
>.OR.&n='28'.OR.&n='30'
          n=&n
          i=(n-6)/2
          SLLNGT=sllngt(i)
          WRIST=wrist(i)
          CUFFTWO=cufftwo(i)
          CUFFSH=cuffsh(i)
          ELBOW=elbow(i)
   ELSE
          PRINT Wrong value entered, enter again please.
          GOTO 30
ENDWHEN
<#
<#
#40 CONTINUE
PRINT {''}
&height=''
WHEN &unit='CM'.OR.&unit='cm'
```

```
   PRINT              Which height do you want?
   PRINT     ------------------------------------------
   PRINT     The short women ( 152cm - 160cm ) --------- S
   PRINT     The medium women ( 160cm - 170cm ) -------- M
   PRINT     The tall women ( 170cm - 178cm ) ---------- T
   PRINT     ------------------------------------------
ELSE
   PRINT              Which height do you want?
   PRINT     ------------------------------------------
   PRINT     The short women (5ft - 5ft 3in) ----------- S
   PRINT     The medium women (5ft 3in - 5ft 7in) ------ M
   PRINT     The tall women (5ft 7in - 5ft 10in) ------- T
   PRINT     ------------------------------------------
ENDWHEN
WHEN &data='1'
READ(  Enter the height please.  S or M or T ---> )&height
ELSE
READ( Enter the nearest height please.  <#
>S or M or T ---> )&height
ENDWHEN
WHEN &height='S'.OR.&height='s'
          SLLNGT=SLLNGT-2.5
  OR &height='M'.OR.&height='m'
          CONTINUE
  OR &height='T'.OR.&height='t'
          SLLNGT=SLLNGT+2.5
  ELSE
          PRINT Wrong value entered, enter again please.
          GOTO 40
ENDWHEN
<#
<#
IF (&unit='CM'.OR.&unit='cm') GOTO 50
SLLNGT=SLLNGT/2.54
WRIST=WRIST/2.54
CUFFTWO=CUFFTWO/2.54
CUFFSH=CUFFSH/2.54
<#
<#
#50 CONTINUE
&SLLNGT=SLLNGT
&WRIST=WRIST
&CUFFTWO=CUFFTWO
&CUFFSH=CUFFSH
<#
<#
WHEN &data='1'
          PRINT {''}
          &change=''
PRINT       The data you entered is <#
>-----------------------------------
PRINT          size; {&n}          height; {&height}<#
>              unit; {&unit}
PRINT          ------------------------------------<#
>--------------------------
WHEN &type='one'.OR.&type='easy'
PRINT          sleeve length={&SLLNGT(,4)}
OR &type='two'
```

```
PRINT          sleeve length={&SLLNGT(,4)}    <#
> cuff size (two-piece sleeve)=<#
>{&CUFFTWO(,4)}
OR &type='shirtsleeve'
PRINT          sleeve length={&SLLNGT(,4)}    <#
> cuff size (shirt sleeve)=<#
>{&CUFFSH(,4)}
ENDWHEN
PRINT      ------------------------------------<#
>-------------------------
READ(          Do you want to change?      Y/N ---> )&change
          WHEN &change='Y'.OR.&change='y'
                  GOTO 10
            OR &change='N'.OR.&change='n'
                  CONTINUE
            ELSE
             PRINT Wrong value entered, enter again please.
                  GOTO 50
          ENDWHEN
  OR &data='2'
          PRINT {''}
PRINT     ------------------------------------<#
>-------------------------
WHEN &unit='CM'.OR.&unit='cm'
PRINT             Enter your individual measurements<#
> please.    (unit: cm)
ELSE
PRINT             Enter your individual measurements<#
> please.    (unit: in)
ENDWHEN
PRINT    If you don't want to change the measurement<#
> of the part, press RETURN.
PRINT    ------------------------------------<#
>-------------------------
PRINT      sleeve length (size;{&n},<#
> height;{&height}) = {&SLLNGT(,4)}
READ(              your sleeve length ---> )SLLNGT
WHEN &type='two'
PRINT      cuff size (two-piece sleeve)<#
> (size;{&n}, height;{&height}) = {&CUFFTWO(,4)}
READ(              your cuff size <two-piece sleeve> <#
>---> )CUFFTWO
OR &type='shirtsleeve'
PRINT      cuff size (shirt sleeve) (size;{&n},<#
> height;{&height}) ={&CUFFSH(,4)}
READ(              your cuff size <shirt sleeve> ---> )CUFFSH
ENDWHEN
<#
<#
#60 CONTINUE
&SLLNGT=SLLNGT
&WRIST=WRIST
&CUFFTWO=CUFFTWO
&CUFFSH=CUFFSH
<#
<#
&change=''
PRINT {''}
```

```
PRINT      The data you entered is <#
>-----------------------------------
PRINT          nearest size; {&n}     <#
> nearest height; {&height}    unit; {&unit}
PRINT          ------------------------------<#
>-------------------------
WHEN &type='one'.OR.&type='easy'
PRINT          sleeve length={&SLLNGT(,4)}
OR &type='two'
PRINT          sleeve length={&SLLNGT(,4)}    <#
> cuff size (two-piece sleeve)={&CUFFTWO(,4)}
OR &type='shirtsleeve'
PRINT           sleeve length={&SLLNGT(,4)}     <#
>cuff size (shirt sleeve)={&CUFFSH(,4)}
ENDWHEN
PRINT       ------------------------------------<#
>-------------------------
READ(     Do you want to change?      Y/N ---> )&change
         WHEN &change='Y'.OR.&change='y'
                GOTO 10
           OR &change='N'.OR.&change='n'
                CONTINUE
           ELSE
            PRINT Wrong value entered, enter again please.
                GOTO 60
         ENDWHEN
ENDWHEN
<#
<#
IF (&unit='CM'.OR.&unit='cm') GOTO 999
SLLNGT=SLLNGT*2.54
WRIST=WRIST*2.54
CUFFTWO=CUFFTWO*2.54
CUFFSH=CUFFSH*2.54
#999 RETURN
```

```
+-----------------------------------------------------------------+
|                                                                 |
|          The one-piece sleeve block pattern (SLONE)             |
|                                                                 |
+-----------------------------------------------------------------+
<#
<#
DECLARE REAL x(20),y(20)
DECLARE LOCATION P(20)
DECLARE ENTITY PNT(5),sp(6),L(9),C(5),B(2),T(4),S1(2)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
&type='one'
CALLP SLSIZE(SLLNGT,CUFFTWO,CUFFSH,WRIST,ELBOW,<#
>&n,&unit,&data,&type)
<#
<#
PRINT {''}
PRINT  ------------------------------------<#
>----------------------------
PRINT  * Enter the back and front armholes please.
PRINT         Move the mouse on the pad to place <#
>the cursor on the graphic
PRINT         window at the armhole of the bodice<#
> to which this sleeve will be
PRINT         fit. Press the left button.
PRINT  * Enter the underarm point please.
PRINT         Move the mouse on the pad to place<#
> the cursor on the graphic
PRINT         window at the underarm point of the<#
> bodice. Press the left button.
PRINT  ------------------------------------<#
>----------------------------
DIGMSK BSPL
DIGI (MENT,NOWAIT,'BSPL',' Enter the back armhole :') S1(1)
PRINT {''}
DIGI (MENT,NOWAIT,'BSPL',' Enter the front armhole :') S1(2)
PRINT {''}
DIGMSK ALL
DIGI (MLOC,NOWAIT,'END',' Enter the underarm point :')<#
> xl,yl,zl
PRINT {''}
<#
<#
PRINT {''}
PRINT          ------------------------------------<#
>-------------------------
PRINT              You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ------------------------------<#
>-----------------------------
PRINT          * Measure the length of the back and<#
> front armhole using icon
```

```
PRINT              (MEASURE LENGTH).
PRINT              The unit of the length will be "cm".
PRINT            * After Measuring, please press keys<#
> "Control" and "x" at the
PRINT              same time  to restart the program.
PRINT           ------------------------------------<#
>------------------------
!<VAR>
EXECDF
READ( Enter the length of the armhole please.<#
> <unit; cm> ---> )armhole
<#
<#
DISP OFF
n=&n
WHEN n<=14
          sleevehead=armhole/3-0.5
  OR n>14.AND.n<=22
          sleevehead=armhole/3-0.3
  OR n>22
          sleevehead=armhole/3
ENDWHEN
sp(1)=POINT/x1,y1+sleevehead/2,0
$E=LINE/sp(1),HORIZ,LENGTH,-30
sp(2)=POINT/INTOF,$E,XSMALL,S1(1),TAG,'P1'
DELETE $E,sp(1)
sp(3)=POINT/x1,y1+sleevehead/4,0
$E=LINE/sp(3),HORIZ,LENGTH,30
sp(4)=POINT/INTOF,$E,XLARGE,S1(2),TAG,'P2'
DELETE $E,sp(3)
DISP ALL
<#
<#
PRINT {''}
PRINT           ------------------------------<#
>----------------------------
PRINT                 You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT           ------------------------------<#
>----------------------------
PRINT           * Divide each armhole by points "P1"<#
> and "p2" using icon
PRINT             (DIVIDE ENTITY), to measure the<#
> length of divided parts.
PRINT           * Measure the length using icon<#
> (MEASURE LENGTH)
PRINT                    from the back shoulder point<#
> to the point "P1",
PRINT                    from the point "P1" to the<#
> back underarm point,
PRINT                    from the front shoulder point<#
> to the point "P2"
PRINT                    and from the point "P2" to <#
>the front underarm point.
PRINT             The unit of the length will be "cm".
PRINT           * After Measuring, please press keys<#
> "Control" and "x" at the
PRINT             same time to restart the program.
```

```
PRINT       ------------------------------------<#
>----------------------------
!<VAR>
EXECDF
PRINT Enter the length of the each part please.
READ( From the back shoulder point to the point<#
> "P1". <unit: cm> ---> )BAU
READ( From the point "P1" to the back underarm<#
> point. <unit: cm> ---> )BAD
READ( From the front shoulder point to the point<#
> "P2". <unit: cm> ---> )FAU
READ( From the point "P2" to the front underarm<#
> point. <unit: cm> ---> )FAD
DELETE sp(2),sp(4)
<#
<#
PRINT {''}
PRINT    Select the position for the shoulder point<#
> of the sleeve block please.
PRINT    ------------------------------------<#
>----------------------------
PRINT    * Move the mouse on the pad to place <#
>the cursor on the graphic window
PRINT      at the position you require.<#
> Press the left button.
PRINT    ------------------------------------<#
>----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
DISP OFF
WHEN n<=14
         FSU=FAU+1
         BSU=BAU+1
  OR n>14.AND.n<=22
         FSU=FAU+1.25
         BSU=BAU+1.25
  OR n>22
         FSU=FAU+1.5
         BSU=BAU+1.5
ENDWHEN
FSD=FAD-0.3
BSD=BAD-0.3
x(1)=x(2)=:x(3)=:X
y(1)=Y
y(2)=y(6)=:y(7)=:y(1)-sleevehead
y(3)=y(8)=:y(9)=:y(13)=:y(12)=:y(1)-SLLNGT
y(4)=y(1)-(y(1)-y(2))*3/4
x(4)=x(1)+SQRT(FSU**2-(y(1)-y(4))**2)
y(5)=(y(1)+y(2))/2
x(5)=x(1)-SQRT(BSU**2-(y(1)-y(5))**2)
x(6)=x(8)=:x(10)=:x(4)+SQRT(FSD**2-(y(4)-y(2))**2)
x(7)=x(11)=:x(9)=:x(5)-SQRT(BSD**2-(y(5)-y(7))**2)
y(10)=y(11)=:y(2)-ELBOW
WHEN SLLNGT<59.3
         x(12)=x(8)-3
```

```
        x(13)=x(9)+3
  OR SLLNGT.GE.59.3.AND.SLLNGT<61.1
        x(12)=x(8)-4
        x(13)=x(9)+4
  ELSE
        x(12)=x(8)-5
        x(13)=x(9)+5
ENDWHEN
x(14)=(x(1)+x(5))/2
y(14)=(y(1)+y(5))/2
x(15)=(x(5)+x(7))/2
y(15)=(y(5)+y(7))/2
x(16)=x(1)+(x(4)-x(1))/3
y(16)=y(1)-(y(1)-y(4))/3
x(17)=(x(4)+x(6))/2
y(17)=(y(4)+y(6))/2
x(18)=(x(9)+x(3))/2
y(18)=y(3)-1
x(19)=(x(3)+x(8))/2
y(19)=y(3)+1
x(20)=x(4)-(x(4)-x(1))/3
y(20)=y(4)+(y(1)-y(4))/3
<#
<#
REPEAT 100: i=1,1,i>20
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
n=&n
WHEN n<=14
        cv1=1.2
        cv2=0.5
        cv3=1.8
        cv4=1.3
        cv5=0.8
  OR n>14.AND.n<=22
        cv1=1.35
        cv2=0.75
        cv3=2
        cv4=1.5
        cv5=1
  OR n>22
        cv1=1.5
        cv2=1
        cv3=2.2
        cv4=1.7
        cv5=1.2
ENDWHEN
$E=LINE/P(5),P(1)
$EP=LINE/P(14),PERP,$E,LENGTH,cv1
PNT(1)=POINT/P(14),DISTAN,cv1,$EP
DELETE $E,$EP
$E=LINE/P(5),P(7)
$EP=LINE/P(15),PERP,$E,LENGTH,cv2
PNT(2)=POINT/P(15),DISTAN,cv2,$EP
DELETE $E,$EP
$E=LINE/P(1),P(4)
```

```
$EP1=LINE/P(16),PERP,$E,LENGTH,cv3
PNT(3)=POINT/P(16),DISTAN,cv3,$EP1
$EP2=LINE/P(20),PERP,$E,LENGTH,cv4
PNT(4)=POINT/P(20),DISTAN,cv4,$EP2
DELETE $E,$EP1,$EP2
$E=LINE/P(6),P(4)
$EP=LINE/P(17),PERP,$E,LENGTH,cv5
PNT(5)=POINT/P(17),DISTAN,cv5,$EP
DELETE $E,$EP
sp(5)=POINT/P(5)
sp(6)=POINT/P(4)
L(1)=LINE/P(1),P(3),FONT,'DASH'
L(2)=LINE/P(6),P(7),FONT,'DASH'
L(3)=LINE/P(10),P(11),FONT,'DASH'
L(4)=LINE/P(6),P(8)
L(5)=LINE/P(7),P(9)
L(6)=LINE/P(6),P(12)
L(7)=LINE/P(7),P(13)
L(8)=LINE/P(8),P(9)
L(9)=LINE/x(1)+3,y(2)-5,0,x(1)+3,y(3)+5,0,FONT,'ARROW'
C(1)=CIRCLE/P(9),P(18),P(3),PGO,P(3),PEND,P(9)
C(2)=CIRCLE/P(8),P(19),P(3),PGO,P(3),PEND,P(8)
C(3)=CIRCLE/CENTER,P(1),RADIUS,0.4
C(4)=CIRCLE/CENTER,P(5),RADIUS,0.4
C(5)=CIRCLE/CENTER,PNT(4),RADIUS,0.4
B(1)=BSPLIN/3,P(1),PNT(1),P(5),PNT(2),P(7)
B(2)=BSPLIN/3,P(1),PNT(3),PNT(4),P(4),PNT(5),P(6)
<#
<#
&T1='one-piece sleeve block'
T(1)=TEXT/&T1,x(1),y(10)+1,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
        T(2)=TEXT/&name+' < SIZE '+&n+' >',x(1),<#
>y(10)+2,0,TJST,'CJT',THGT,0.75
  OR &data='2'
        IF (&name='') GOTO 200
        T(2)=TEXT/&name,x(1),y(10)+2,0,TJST,'CJT',THGT,0.75
        #200 CONTINUE
ENDWHEN
T(3)=TEXT/'back',(x(3)+x(9))/2,y(10),0,TJST,'CJT',THGT,0.75
T(4)=TEXT/'front',(x(3)+x(8))/2,y(10),0,TJST,'CJT',THGT,0.75
<#
<#
DISP ALL
#999 END
```

```
+-----------------------------------------------------------+
|                                                           |
|          The two-piece sleeve block pattern (SLTWO)       |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL x(18),y(18)
DECLARE ENTITY P(21),L(7),B(4),C(6),cp(1),T(6),sp(5),S2(2)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
&type='two'
CALLP SLSIZE(SLLNGT,CUFFTWO,CUFFSH,WRIST,ELBOW,<#
>&n,&unit,&data,&type)
<#
<#
PRINT {''}
PRINT  ----------------------------------<#
>--------------------------------
PRINT  * Enter the back and front armholes please.
PRINT          Move the mouse on the pad to place<#
> the cursor on the graphic
PRINT          window at the armhole of the bodice <#
>to which this sleeve will be
PRINT          fit. Press the left button.
PRINT  * Enter the underarm point and the front <#
>armhole pitch point please.
PRINT          Move the mouse on the pad to place<#
> the cursor on the graphic
PRINT          window at the underarm point of the<#
> bodice. Press the left button.
PRINT          Do the same way for the front<#
> armhole pitch point.
PRINT  ----------------------------------<#
>--------------------------------
DIGMSK BSPL
DIGI (MENT,NOWAIT,'BSPL',' Enter the back armhole :') S2(1)
PRINT {''}
DIGMSK BSPL
DIGI (MENT,NOWAIT,'BSPL',' Enter the front armhole :') S2(2)
PRINT {''}
DIGMSK ALL
DIGI (MLOC,NOWAIT,'END',' Enter the underarm point :')<#
> x1,y1,z1
PRINT {''}
DIGMSK POI
DIGI (MENT,NOWAIT,'POI',' Enter the front armhole pitch<#
> point :') sp(1),x2,y2,z2
DIGMSK ALL
PRINT {''}
<#
<#
PRINT {''}
PRINT          ----------------------------------<#
```

```
>------------------------
PRINT                You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ------------------------------------<#
>------------------------
PRINT            * Measure the length of the front and <#
>back armhole using icon
PRINT              (MEASURE LENGTH).
PRINT              The unit of the length will be "cm".
PRINT            * After Measuring, please press keys<#
> "Control" and "x" at the
PRINT              same time  to restart the program.
PRINT          ------------------------------------<#
>------------------------
!<VAR>
EXECDF
READ( Enter the length of the armhole please.<#
> <unit: cm> ---> )armhole
<#
<#
DISP OFF
sp(2)=POINT/x1,y1+armhole*2/9-1,0
$E=LINE/sp(2),HORIZ,LENGTH,-30
sp(3)=POINT/INTOF,$E,XSMALL,S2(1),TAG,'P1'
$L1=LINE/sp(3),x1,y1,0,TAG,'L1'
DELETE sp(2),$E
sp(4)=POINT/x1,y1+armhole/12,0
$E=LINE/sp(4),HORIZ,LENGTH,30
sp(5)=POINT/INTOF,$E,XLARGE,S2(2),TAG,'P2'
DELETE sp(4),$E
DISP ALL
<#
<#
PRINT {''}
PRINT          ------------------------------------<#
>------------------------
PRINT              You can use the icon tool, <#
>while the prompt is "<VAR>".
PRINT          ------------------------------------<#
>------------------------
PRINT            * Divide each armhole by points "P1"<#
> and "p2" using icon
PRINT              (DIVIDE ENTITY), to measure the<#
> length of divided parts.
PRINT            * Measure the length using icon<#
> (MEASURE LENGTH)
PRINT                    from the back shoulder point <#
>to the point "P1" and
PRINT                    from the front shoulder point<#
> to the point "P2".
PRINT              The unit of the length will be "cm".
PRINT            * Measure the length of "L1" using <#
>icon (MEASURE LENGTH).
PRINT            * After Measuring, please press keys<#
> "Control" and "x" at
PRINT              the same time to restart the program.
PRINT          ------------------------------------<#
>------------------------
```

```
!<VAR>
EXECDF
PRINT Enter the length of the each part please.
READ( From the back shoulder point to the point<#
> "P1". <unit: cm> ---> )BAU
READ( From the front shoulder point to the point<#
> "P2". <unit: cm> ---> )FAU
READ( The length of the line "L1". <unit: cm> ---> )L1
DELETE $L1,sp(3),sp(5)
<#
<#
PRINT {''}
PRINT    Select the position for the shoulder point<#
> of the sleeve block please.
PRINT    ------------------------------------<#
>----------------------------
PRINT    * Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT       at the position you require. <#
>Press the left button.
PRINT    ------------------------------------<#
>------------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
DISP OFF
n=&n
WHEN n<=14
          BSU=BAU+0.8
          FSU=FAU+1
          c=2.3
  OR n>14.AND.n<=22
          BSU=BAU+1
          FSU=FAU+1.25
          c=2.5
  OR n>22
          BSU=BAU+1.2
          FSU=FAU+1.5
          c=2.7
ENDWHEN
x(1)=X
y(1)=Y
y(2)=y(13)=:y(1)-armhole/9-1
x(2)=x(1)-SQRT(BSU**2-(y(1)-y(2))**2)
y(3)=y(1)-armhole/4
x(3)=x(11)=:x(7)=:x(1)+SQRT(FSU**2-(y(1)-y(3))**2)
x(4)=x(3)-x2+x1
y(4)=y(1)-armhole/3
y(5)=y(6)=:y(4)+2
x(5)=x(9)=:x(3)-2
x(6)=x(8)=:x(3)+2
y(7)=y(8)=:y(9)=:y(1)-SLLNGT
y(10)=y(7)-3
x(10)=x(7)-SQRT(CUFFTWO**2-9)
y(11)=y(12)=:y(17)=:y(18)=:(y(6)+y(8))/2
x(12)=x(5)-2
```

```
x(13)=x(4)-SQRT(L1**2-(y(13)-y(4))**2)
x(14)=x(1)+(x(3)-x(1))/3
y(14)=y(1)-(y(1)-y(3))/3
x(15)=(x(1)+x(2))/2
y(15)=(y(1)+y(2))/2
x(16)=(x(4)+x(13))/2
y(16)=(y(4)+y(13))/2
x(17)=x(13)-(x(13)-x(10))*(y(13)-y(11))/(y(13)-y(10))-c
x(18)=x(2)+(x(10)-x(2))*(y(2)-y(11))/(y(2)-y(10))-c
<#
<#
REPEAT 100: i=1,1,i>18
P(i)=POINT/x(i),y(i),0
#100 CONTINUE
<#
<#
$E=LINE/P(1),P(3)
$EP=LINE/P(14),PERP,$E,LENGTH,2
P(19)=POINT/P(14),DISTAN,2,$EP
DELETE $E,$EP
$E=LINE/P(2),P(1)
$EP=LINE/P(15),PERP,$E,LENGTH,1
P(20)=POINT/P(15),DISTAN,1,$EP
DELETE $E,$EP
$E=LINE/P(4),P(13)
$EP=LINE/P(16),PERP,$E,LENGTH,1.5
P(21)=POINT/P(16),DISTAN,1.5,$EP
DELETE $E,$EP
B(1)=BSPLIN/3,P(1),P(20),P(2)
$L1=LINE/P(6),P(3)
$L2=LINE/(x(3)+x(6))/2,(y(3)+y(6))/2,0,PERP,$L1,LENGTH,0.15
$P=POINT/(x(3)+x(6))/2,(y(3)+y(6))/2,0,DISTAN,0.15,$L2
DELETE $L1,$L2
B(2)=BSPLIN/3,P(1),P(19),P(3)
B(3)=BSPLIN/3,P(3),$P,P(6)
<#
<#
MIRROR (COPY,cp(1)) P(10),MX,x(3),0,0
MIRROR P(4),P(5),P(9),P(12),MX,x(3),0,0
MIRROR P(13),P(17),P(21),MX,x(3),0,0
<#
<#
B(4)=BSPLIN/3,P(13),P(21),P(4),P(5)
L(1)=LINE/P(11),P(18),FONT,'DASH'
L(2)=LINE/P(12),P(17),FONT,'DASH'
L(3)=LINE/P(10),P(7)
L(4)=LINE/P(7),P(8)
L(5)=LINE/cp(1),P(9)
L(6)=LINE/x(1)+3,y(4)-5,0,x(1)+3,y(8)+5,0,FONT,'ARROW'
L(7)=LINE/x(3)*2-(x(12)+x(17))/2,y(4)-5,0,<#
>x(3)*2-(x(12)+x(17))/2,y(8)+5,0,FONT,'ARROW'
C(1)=CIRCLE/P(2),P(18),P(10),PGO,P(10),PEND,P(2)
C(2)=CIRCLE/P(6),P(11),P(8),PGO,P(8),PEND,P(6)
C(3)=CIRCLE/P(13),P(17),cp(1),PGO,cp(1),PEND,P(13)
C(4)=CIRCLE/P(5),P(12),P(9),PGO,P(9),PEND,P(5)
C(5)=CIRCLE/CENTER,P(1),RADIUS,0.4
C(6)=CIRCLE/CENTER,P(4),RADIUS,0.4
DELETE 13,P(6)
```

```
DELETE cp(1)
<#
<#
&T1='two-piece sleeve block'
T(1)=TEXT/&T1,(x(11)+x(18))/2,y(11)+0.5,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,x(3)*2-(x(12)+x(17))/2,y(11)+0.5,0,<#
>TJST,'CJT',THGT,0.75
T(3)=TEXT/'top sleeve',(x(11)+x(18))/2,y(11)+1.5,0,<#
>TJST,'CJT',THGT,0.75
T(4)=TEXT/'under sleeve',x(3)*2-(x(12)+x(17))/2,<#
>y(11)+1.5,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
        T(5)=TEXT/&name+' < SIZE '+&n+' >',<#
>(x(11)+x(18))/2,y(11)+2.5,0,TJST,'CJT',THGT,0.75
        T(6)=TEXT/&name+' < SIZE '+&n+' >',<#
>x(3)*2-(x(12)+x(17))/2,y(11)+2.5,0,TJST,'CJT',THGT,0.75
  OR &data='2'
        IF (&name='') GOTO 200
        T(5)=TEXT/&name,(x(11)+x(18))/2,y(11)+2.5,0,<#
>TJST,'CJT',THGT,0.75
        T(6)=TEXT/&name,x(3)*2-(x(12)+x(17))/2,<#
>y(11)+2.5,0,TJST,'CJT',THGT,0.75
        #200 CONTINUE
ENDWHEN
<#
<#
!REGENERATE GRAPHICS
EXECDF
DISP ALL
#999 END
```

```
+-----------------------------------------------------------+
|                                                           |
|        The easy fitting sleeve block pattern (SLEA)       |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL x(16),y(16)
DECLARE LOCATION P(16)
DECLARE ENTITY PNT(4),p(2),L(7),B(2),T(4),C(3)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
&type='easy'
CALLP SLSIZE(SLLNGT,CUFFTWO,CUFFSH,WRIST,ELBOW,<#
>&n,&unit,&data,&type)
<#
<#
PRINT {''}
PRINT          ------------------------------------<#
>------------------------
PRINT                  You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ------------------------------------<#
>------------------------
PRINT          * Measure the length of the back and front<#
> armhole using icon
PRINT            (MEASURE LENGTH).
PRINT            The unit of the length will be "cm".
PRINT          * After Measuring, please press keys<#
> "Control" and "x" at the
PRINT            same time  to restart the program.
PRINT          ------------------------------------<#
>------------------------
!<VAR>
EXECDF
READ( Enter the length of the armhole please.<#
> <unit: cm> ---> )armhole
<#
<#
PRINT {''}
sleevehead=armhole/4
IF (&unit='in'.OR.&unit='IN') sleevehead=sleevehead/2.54
&sleevehead=sleevehead
&sh=&sleevehead(,4)
PRINT The default value of the sleevehead height<#
> is {&sh} {&unit}, which is 1/4 of armhole.
WHEN &unit='cm'.OR.&unit='CM'
READ( Enter the sleevehead height <crown height><#
> please. <unit: cm> ---> )sleevehead
  OR &unit='in'.OR.&unit='IN'
READ( Enter the sleevehead height <crown height> <#
>please. <unit: in> ---> )sleevehead
sleevehead=sleevehead*2.54
```

```
ENDWHEN
<#
<#
PRINT {''}
PRINT    Select the position for the shoulder point <#
>of the sleeve block please.
PRINT    ------------------------------------<#
>----------------------------
PRINT    * Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT      at the position you require.<#
> Press the left button.
PRINT    ------------------------------------<#
>----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
DISP OFF
x(1)=x(2)=:x(3)=:x(4)=:X
y(1)=Y
y(2)=y(5)=:y(7)=:y(1)-sleevehead
y(3)=y(6)=:y(8)=:y(1)-SLLNGT
y(4)=y(15)=:y(16)=:(y(2)+y(3))/2
x(5)=x(15)=:x(6)=:x(1)-SQRT((armhole/2)**2-(y(1)-y(5))**2)
x(7)=x(16)=:x(8)=:x(1)+SQRT((armhole/2)**2-(y(1)-y(7))**2)
x(9)=x(5)+(x(1)-x(5))/4
y(9)=y(5)+(y(1)-y(5))/4
x(10)=(x(5)+x(1))/2
y(10)=(y(5)+y(1))/2
x(11)=x(1)-(x(1)-x(5))/4
y(11)=y(1)-(y(1)-y(5))/4
x(12)=x(1)+(x(7)-x(1))/4
y(12)=y(1)-(y(1)-y(7))/4
x(13)=(x(1)+x(7))/2
y(13)=(y(1)+y(7))/2
x(14)=x(7)-(x(7)-x(1))/4
y(14)=y(7)+(y(1)-y(7))/4
<#
<#
REPEAT 100: i=1,1,i>16
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
p(1)=POINT/P(9)
p(2)=POINT/P(13)
<#
<#
WHEN sleevehead>=armhole/5
          cv1=1
          cv2=1.75
  OR sleevehead<armhole/5.AND.sleevehead>=armhole/7
          cv1=0.75
          cv2=1.25
  OR sleevehead<armhole/7
          cv1=0.5
```

```
        cv2=0.85
ENDWHEN
$E=LINE/P(5),P(1)
$EP1=LINE/P(10),PERP,$E,LENGTH,cv1
PNT(1)=POINT/P(10),DISTAN,cv1,$EP1
$EP2=LINE/P(11),PERP,$E,LENGTH,cv2
PNT(2)=POINT/P(11),DISTAN,cv2,$EP2
DELETE $E,$EP1,$EP2
$E=LINE/P(1),P(7)
$EP=LINE/P(12),PERP,$E,LENGTH,cv1
PNT(3)=POINT/P(12),DISTAN,cv1,$EP
DELETE $E,$EP
$E=LINE/P(7),P(1)
$EP=LINE/P(14),PERP,$E,LENGTH,cv1
PNT(4)=POINT/P(14),DISTAN,cv1,$EP
DELETE $E,$EP1,$EP2
<#
<#
L(1)=LINE/P(1),P(3),FONT,'DASH'
L(2)=LINE/P(5),P(7),FONT,'DASH'
L(3)=LINE/P(15),P(16),FONT,'DASH'
L(4)=LINE/P(5),P(6)
L(5)=LINE/P(7),P(8)
L(6)=LINE/P(6),P(8)
L(7)=LINE/x(1)+3,y(2)-5,0,x(1)+3,y(3)+5,0,FONT,'ARROW'
C(1)=CIRCLE/CENTER,P(1),RADIUS,0.4
C(2)=CIRCLE/CENTER,PNT(1),RADIUS,0.4
C(3)=CIRCLE/CENTER,P(13),RADIUS,0.4
B(1)=BSPLIN/3,P(1),PNT(2),PNT(1),P(9),P(5)
B(2)=BSPLIN/3,P(1),PNT(3),P(13),PNT(4),P(7)
<#
<#
&T1='easy fitting sleeve block'
T(1)=TEXT/&T1,x(1),y(4)+1,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
          T(2)=TEXT/&name+' < SIZE '+&n+' >',<#
>x(1),y(4)+2,0,TJST,'CJT',THGT,0.75
   OR &data='2'
          IF (&name='') GOTO 200
        T(2)=TEXT/&name,x(1),y(4)+2,0,TJST,'CJT',THGT,0.75
          #200 CONTINUE
ENDWHEN
T(3)=TEXT/'back',(x(4)+x(15))/2,y(4),0,TJST,'CJT',THGT,0.75
T(4)=TEXT/'front',(x(4)+x(16))/2,y(4),0,TJST,'CJT',THGT,0.75
<#
<#
DISP ALL
#999 END
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|                The shirt sleeve block pattern (SLSH)                  |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE REAL x(19),y(19),xc(15),yc(15)
DECLARE LOCATION P(19),PC(15)
DECLARE ENTITY PNT(5),p(2),L(10),B(2),C(6),T(5),LC(8)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
&type='shirtsleeve'
CALLP SLSIZE(SLLNGT,CUFFTWO,CUFFSH,WRIST,ELBOW,<#
>&n,&unit,&data,&type)
<#
<#
PRINT {''}
PRINT          ------------------------------<#
>--------------------------
PRINT                  You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ------------------------------<#
>--------------------------
PRINT          * Measure the length of the back and <#
>front armhole using icon
PRINT            (MEASURE LENGTH).
PRINT            The unit of the length will be "cm".
PRINT          * After Measuring, please press keys<#
> "Control" and "x" at the
PRINT            same time  to restart the program.
PRINT          ------------------------------<#
>--------------------------
!<VAR>
EXECDF
READ( Enter the length of the armhole please. <#
><unit: cm> ---> )armhole
<#
<#
PRINT {''}
cuffdept=7
IF (&unit='in'.OR.&unit='IN') cuffdepth=cuffdepth/2.54
WHEN &unit='cm'.OR.&unit='CM'
PRINT The default value of the cuffdepth is 7 cm.
READ( Enter the cuff depth please.<#
> <unit: cm> ---> )cuffdepth
  OR &unit='in'.OR.&unit='IN'
PRINT The default value of the cuffdepth is 2.76 in.
READ( Enter the cuff depth please.<#
> <unit: in> ---> )cuffdepth
        cuffdepth=cuffdepth*2.54
ENDWHEN
<#
```

```
<#
PRINT {''}
PRINT    Select the position for the shoulder point<#
> of the sleeve block please.
PRINT    ------------------------------------<#
>-----------------------------
PRINT    * Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT      at the position you require.<#
> Press the left button.
PRINT    ------------------------------------<#
>-----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
DISP OFF
x(1)=x(2)=:x(3)=:x(4)=:X
y(1)=Y
y(2)=y(5)=:y(7)=:y(1)-armhole/4
y(3)=y(6)=:y(8)=:y(1)-SLLNGT+cuffdepth-2
y(4)=y(15)=:y(18)=:y(16)=:(y(2)+y(3))/2
x(5)=x(1)-SQRT((armhole/2)**2-(y(1)-y(5))**2)
x(6)=x(5)+(x(3)-x(5))/4+1
x(7)=x(1)+SQRT((armhole/2)**2-(y(1)-y(7))**2)
x(8)=x(7)-(x(7)-x(3))/4-1
x(9)=x(5)+(x(1)-x(5))/4
y(9)=y(5)+(y(1)-y(5))/4
x(10)=(x(5)+x(1))/2
y(10)=(y(5)+y(1))/2
x(11)=x(1)-(x(1)-x(5))/4
y(11)=y(1)-(y(1)-y(5))/4
x(12)=x(1)+(x(7)-x(1))/4
y(12)=y(1)-(y(1)-y(7))/4
x(13)=(x(1)+x(7))/2
y(13)=(y(1)+y(7))/2
x(14)=x(7)-(x(7)-x(1))/4
y(14)=y(7)+(y(1)-y(7))/4
x(15)=x(5)+(x(3)-x(5))/8
x(16)=x(7)-(x(7)-x(3))/8
x(17)=x(18)=:x(19)=:(x(3)+x(6))/2
y(17)=y(3)-0.75
y(19)=y(4)-(y(4)-y(3))/3
<#
<#
REPEAT 100: i=1,1,i>19
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
p(1)=POINT/x(9),y(9),0
p(2)=POINT/x(13),y(13),0
$E=LINE/P(1),P(5)
$EP1=LINE/P(10),PERP,$E,LENGTH,-1
PNT(1)=POINT/P(10),DISTAN,1,$EP1
$EP2=LINE/P(11),PERP,$E,LENGTH,-1.75
PNT(2)=POINT/P(11),DISTAN,1.75,$EP2
```

```
DELETE $E,$EP1,$EP2
$E=LINE/P(1),P(7)
$EP1=LINE/P(12),PERP,$E,LENGTH,1
PNT(3)=POINT/P(12),DISTAN,1,$EP1
$EP2=LINE/P(14),PERP,$E,LENGTH,-1
PNT(4)=POINT/P(14),DISTAN,1,$EP2
DELETE $E,$EP1,$EP2
L(1)=LINE/P(1),P(3),FONT,'DASH'
L(2)=LINE/P(5),P(7),FONT,'DASH'
L(3)=LINE/P(15),P(16),FONT,'DASH'
L(4)=LINE/P(5),P(15)
L(5)=LINE/P(15),P(6)
L(6)=LINE/P(7),P(16)
L(7)=LINE/P(16),P(8)
L(8)=LINE/P(3),P(8)
L(9)=LINE/P(19),P(17)
L(10)=LINE/x(1)+3,y(2)-5,0,x(1)+3,y(3)+5,0,FONT,'ARROW'
C(1)=CIRCLE/P(6),P(17),P(3),PGO,P(3),PEND,P(6)
C(2)=CIRCLE/CENTER,P(1),RADIUS,0.4
C(3)=CIRCLE/CENTER,PNT(1),RADIUS,0.4
C(4)=CIRCLE/CENTER,P(13),RADIUS,0.4
B(1)=BSPLIN/3,P(1),PNT(2),PNT(1),P(9),P(5)
B(2)=BSPLIN/3,P(1),PNT(3),P(13),PNT(4),P(7)
<#
<#
&T1='shirt sleeve block'
T(1)=TEXT/&T1,x(1),y(4)+1,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
        T(2)=TEXT/&name+' < SIZE '+&n+' >',x(1),<#
>y(4)+2,0,TJST,'CJT',THGT,0.75
  OR &data='2'
        IF (&name='') GOTO 200
        T(2)=TEXT/&name,x(1),y(4)+2,0,TJST,'CJT',THGT,0.75
        #200 CONTINUE
ENDWHEN
T(3)=TEXT/'back',(x(4)+x(15))/2,y(4),0,TJST,'CJT',THGT,0.75
T(4)=TEXT/'front',(x(4)+x(16))/2,y(4),0,TJST,'CJT',THGT,0.75
<#
<#     * cuffs *
<#
xc(1)=xc(2)=:xc(12)=:x(1)-CUFFSH/2-1
yc(1)=yc(3)=:yc(8)=:yc(10)=:y(3)-3
yc(2)=yc(4)=:yc(9)=:yc(11)=:yc(13)=:yc(15)=:yc(1)-cuffdepth
xc(3)=xc(4)=:xc(14)=:x(1)+CUFFSH/2+1
xc(5)=xc(1)+1
yc(5)=yc(6)=:yc(7)=:(yc(1)+yc(2))/2
xc(6)=xc(3)-0.75
xc(7)=xc(6)-1.5
xc(8)=xc(9)=:xc(1)+1
xc(10)=xc(11)=:xc(3)-1
yc(12)=yc(14)=:yc(2)+cuffdepth/3
xc(13)=xc(1)+cuffdepth/3
xc(15)=xc(4)-cuffdepth/3
<#
<#
REPEAT 300: i=1,1,i>15
PC(i)=VECT(xc(i),yc(i),0)
#300 CONTINUE
```

```
<#
<#
PNT(5)=POINT/xc(5),yc(5),0
LC(1)=LINE/PC(1),PC(2)
LC(2)=LINE/PC(1),PC(3)
LC(3)=LINE/PC(2),PC(4)
LC(4)=LINE/PC(3),PC(4)
LC(5)=LINE/PC(8),PC(9),FONT,'DASH'
LC(6)=LINE/PC(10),PC(11),FONT,'DASH'
LC(7)=LINE/PC(7),PC(6)
LC(8)=LINE/x(1)+3,yc(1)-0.5,0,x(1)+3,yc(2)+0.5,0,FONT,'ARROW'
C(5)=CIRCLE/RADIUS,cuffdepth/3,RIGHT,PC(12),PC(13),<#
>PGO,PC(12),PEND,PC(13)
C(6)=CIRCLE/RADIUS,cuffdepth/3,LEFT,PC(15),PC(14),<#
>PGO,PC(15),PEND,PC(14)
T(5)=TEXT/'cuff',x(1),yc(5),0,TJST,'CJT',THGT,0.75
<#
<#
DISP ALL
#999 END
```

SKTALK (Linkfile: SKTA, SKSIZE)

```
MAIN SKTA
IMAGE SKTAIMAGE
LINKM SKSIZE
ENDLINK
```

SKCILK (Linkfile: SKCI, SKSIZE)

```
MAIN SKCI
IMAGE SKCIIMAGE
LINKM SKSIZE
ENDLINK
```

SKPLLK (Linkfile: SKPL, SKSIZE)

```
MAIN SKPL
IMAGE SKPLIMAGE
LINKM SKSIZE
ENDLINK
```

SKGALK (Linkfile: SKGA, SKSIZE)

```
MAIN SKGA
IMAGE SKGAIMAGE
LINKM SKSIZE
ENDLINK
```

```
+-----------------------------------------------------------+
|                                                           |
|              The skirt size (SKSIZE) procedure            |
|                                                           |
+-----------------------------------------------------------+
<#
<#
PROC SKSIZE(WAIST,HIP,WATOHI,WATOKN,&n,&data,&unit)
DECLARE REAL waist(12),hip(12),watohi(12),watokn(12)
DATA waist/60,64,68,72,77,82,87,92,97,102,107,112/
DATA hip/85,89,93,97,102,107,112,117,122,127,132,137/
DATA watohi/20,20.3,20.6,20.9,21.2,21.5,21.8,22.1,<#
>22.3,22.5,22.7,22.9/
DATA watokn/57.5,58,58.5,59,59.5,60,60.5,61,<#
>61.25,61.5,61.75,62/
ERTRAP 999
<#
<#
#10 CONTINUE
PRINT {''}
&data=''
PRINT          Which measurements do you want to use?
PRINT      ----------------------------------------
PRINT      The body measurements of this program --- 1
PRINT      Your individual body measurements ------- 2
PRINT      ----------------------------------------
READ(              Enter the number please. ---> )&data
WHEN &data='1'.OR.&data='2'
          CONTINUE
  ELSE
          PRINT Wrong answer, enter again please.
          GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
PRINT {''}
&unit=''
READ( Which units do you want to use? <#
>    Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'.OR.&unit='IN'.OR.&unit='in'
          CONTINUE
  ELSE
          PRINT Wrong answer, enter again please.
          GOTO 20
ENDWHEN
<#
<#
#30 CONTINUE
PRINT {''}
&n=''
WHEN &unit='CM'.OR.&unit='cm'
          GOSUB 1000
  ELSE
          GOSUB 2000
ENDWHEN
PRINT    The size must be an even number between 8 and 30.
WHEN &data='1'
```

```
READ(              Enter the size please. ---> )&n
ELSE
READ(          Enter the nearest size please. ---> )&n
ENDWHEN
WHEN &n='8'.OR.&n='10'.OR.&n='12'.OR.&n='14'.OR.&n='16'<#
>.OR.&n='18'.OR.&n='20'.OR.&n='22'.OR.&n='24'.OR.&n='26'<#
>.OR.&n='28'.OR.&n='30'
         n=&n
         i=(n-6)/2
         WAIST=waist(i)
         HIP=hip(i)
         WATOHI=watohi(i)
         WATOKN=watokn(i)
   ELSE
         PRINT Wrong value entered, enter again please.
         GOTO 30
ENDWHEN
<#
<#
#40 CONTINUE
&height=''
PRINT {''}
WHEN &unit='CM'.OR.&unit='cm'
     PRINT                 Which height do you want?
     PRINT  ---------------------------------------
     PRINT  The short women ( 152cm - 160cm ) --------- S
     PRINT  The medium women ( 160cm - 170cm ) -------- M
     PRINT  The tall women ( 170cm - 178cm ) ---------- T
     PRINT  ---------------------------------------
   ELSE
     PRINT                 Which height do you want?
     PRINT  ---------------------------------------
     PRINT  The short women (5ft - 5ft 3in) ----------- S
     PRINT  The medium women (5ft 3in - 5ft 7in) ------ M
     PRINT  The tall women (5ft 7in - 5ft 10in) ------- T
     PRINT  ---------------------------------------
ENDWHEN
WHEN &data='1'
READ(   Enter the height please.  S or M or T ---> )&height
ELSE
READ( Enter the nearest height please. <#
> S or M or T ---> )&height
ENDWHEN
WHEN &height='S'.OR.&height='s'
         WATOKN=WATOKN-3
   OR &height='M'.OR.&height='m'
         CONTINUE
   OR &height='T'.OR.&height='t'
         WATOKN=WATOKN+3
   ELSE
         PRINT Wrong value entered, enter again please.
         GOTO 40
ENDWHEN
<#
<#
IF (&unit='CM'.OR.&unit='cm') GOTO 50
WAIST=WAIST/2.54
HIP=HIP/2.54
```

```
WATOHI=WATOHI/2.54
WATOKN=WATOKN/2.54
<#
<#
#50 CONTINUE
&WAIST=WAIST
&HIP=HIP
&WATOHI=WATOHI
&WATOKN=WATOKN
<#
<#
WHEN &data='1'
        PRINT {''}
        &change=''
PRINT       The data you entered is <#
>-----------------------------------
PRINT          size; {&n}          height; {&height}<#
>              unit; {&unit}
PRINT          ---------------------------------<#
>-----------------------------------
PRINT          waist={&WAIST(,4)}                    <#
> hip={&HIP(,4)}
PRINT          waist to hip={&WATOHI(,4)}     <#
>         waist to knee={&WATOKN(,4)}
PRINT       ---------------------------------<#
>-----------------------------------
READ(          Do you want to change?      Y/N ---> )&change
        WHEN &change='Y'.OR.&change='y'
                GOTO 10
             OR &change='N'.OR.&change='n'
                   CONTINUE
             ELSE
               PRINT Wrong value entered, enter again please.
                   GOTO 50
          ENDWHEN
  OR &data='2'
        PRINT {''}
PRINT   ------------------------------------------<#
>-----------------------------
WHEN &unit='CM'.OR.&unit='cm'
PRINT             Enter your individual measurements<#
> please.    (unit: cm)
ELSE
PRINT             Enter your individual measurements<#
> please.    (unit: in)
ENDWHEN
PRINT    If you don't want to change the measurement<#
> of the part, press RETURN.
PRINT   ------------------------------------------<#
>-----------------------------
PRINT waist (size;{&n}, height;{&height}) = {&WAIST(,4)}
READ(       your waist ---> )WAIST
PRINT hip (size;{&n}, height;{&height}) = {&HIP(,4)}
READ(       your hip ---> )HIP
PRINT waist to hip (size;{&n}, height;{&height}) =<#
> {&WATOHI(,4)}
READ(       your waist to hip ---> )WATOHI
PRINT waist to knee (size;{&n}, height;{&height}) =<#
```

```
> {&WATOKN(,4)}
READ(       your waist to knee < or skirt length ><#
> ---> )WATOKN
<#
<#
#60 CONTINUE
&WAIST=WAIST
&HIP=HIP
&WATOHI=WATOHI
&WATOKN=WATOKN
<#
<#
&change=''
PRINT {''}
PRINT       The data you entered is<#
> ------------------------------------
PRINT          nearest size; {&n}    <#
>  nearest height; {&height}     unit; {&unit}
PRINT          ------------------------------------<#
>----------------------------
PRINT          waist={&WAIST(,4)}       <#
>  waist to hip={&WATOHI(,4)}
PRINT          hip={&HIP(,4)}     <#
>  waist to knee (skirt length)={&WATOKN(,4)}
PRINT       ------------------------------------<#
>----------------------------
READ(       Do you want to change?      Y/N ---> )&change
         WHEN &change='Y'.OR.&change='y'
                GOTO 10
              OR &change='N'.OR.&change='n'
                CONTINUE
              ELSE
               PRINT Wrong value entered, enter again please.
                GOTO 60
           ENDWHEN
ENDWHEN
<#
<#
IF (&unit='CM'.OR.&unit='cm') GOTO 999
WAIST=WAIST*2.54
HIP=HIP*2.54
WATOHI=WATOHI*2.54
WATOKN=WATOKN*2.54
<#
<#
#999 RETURN
<#
<#
#1000 CONTINUE
PRINT                      WOMEN OF MEDIUM HEIGHT 160CM-170CM<#
>    (UNIT: CM)
PRINT      ------------------------------------------------<#
>----------------------------
PRINT                     8    10    12    14    16    18    20   <#
> 22    24    26    28    30
PRINT      ------------------------------------------------<#
>----------------------------
PRINT      WAIST      60    64    68    72    77    82    87   <#
```

```
> 92   97   102   107   112
PRINT       HIPS        85    89    93    97   102   107   112 <#
> 117   122   127   132   137
PRINT       WA-TO-HI   20    20.3 20.6 20.9 21.2 21.5<#
> 21.8 22.1 22.3 22.5 22.7 22.9
PRINT       WA-TO-KN   57.5 58    58.5 59    59.5 60   <#
> 60.5 61    61.3 61.5 61.8 62
PRINT       ----------------------------------------<#
>-------------------------
PRINT                    WA-TO-HI: waist to hip    <#
>    WA-TO-KN: waist to knee
PRINT {''}
RTNSUB
<#
<#
#2000 CONTINUE
PRINT                      WOMEN OF MEDIUM HEIGHT 5FT 3IN -<#
> 5FT 7IN (UNIT: IN)
PRINT       ----------------------------------------<#
>-------------------------
PRINT                       8    10    12    14    16    18    20   <#
> 22    24    26    28    30
PRINT       ----------------------------------------<#
>-------------------------
PRINT       WAIST       23.6 25.2 26.8 28.3 30.3 32.3<#
> 34.3 36.2 38.2 40.2 42.1 44.1
PRINT       HIPS        33.5 35    36.6 38.2 40.2 42.1<#
> 44.1 46.1 48    50    52    53.9
PRINT       WA-TO-HI    7.9   8     8.1   8.2   8.3   8.5 <#
> 8.6   8.7   8.8   8.9   8.9   9
PRINT       WA-TO-KN   22.6 22.8 23    23.2 23.4 23.6 <#
>23.8 24    24.1 24.2 24.3 24.4
PRINT       ----------------------------------------<#
>-------------------------
PRINT                    WA-TO-HI: waist to hip     <#
>   WA-TO-KN: waist to knee
PRINT {''}
RTNSUB
```

```
+-----------------------------------------------------------------+
|                                                                 |
|          The tailored skirt block pattern (SKTA)                |
|                                                                 |
+-----------------------------------------------------------------+
<#
<#
DECLARE REAL x(13),y(13)
DECLARE ENTITY L(15),P(22),T(6),C(7)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP SKSIZE(WAIST,HIP,WATOHI,WATOKN,&n,&data,&unit)
<#
<#
PRINT {''}
PRINT  Select the position for the centre back<#
> waist of the skirt block please.
PRINT  ----------------------------------<#
>---------------------------------
PRINT     Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT     at the position you require.<#
> Press the left button.
PRINT  ----------------------------------<#
>---------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(5)=:x(3)=:X
y(1)=y(2)=:Y
x(2)=x(6)=:x(4)=:x(1)+HIP/2+1.5
y(3)=y(8)=:y(4)=:y(1)-WATOKN
y(5)=y(7)=:y(6)=:y(1)-WATOHI
x(7)=x(8)=:x(1)+HIP/4+1.5
x(9)=x(1)+WAIST/4+4.25
y(9)=y(10)=:y(1)+1.25
x(10)=x(2)-WAIST/4-2.25
x(11)=x(1)+(x(9)-x(1))/3
x(12)=x(1)+(x(9)-x(1))*2/3
y(11)=y(1)+1.25/3
y(12)=y(1)+1.25*2/3
x(13)=x(2)-(x(2)-x(10))*2/3
y(13)=y(2)+1.25*2/3
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>13
P(i)=POINT/x(i),y(i),0
#100 CONTINUE
<#
<#
```

```
$E=LINE/P(9),P(1)
$EP=LINE/P(11),PERP,$E,LENGTH,14
P(20)=POINT/P(11),DISTAN,14,$EP
P(14)=POINT/P(11),DISTAN,1,$E
P(15)=POINT/P(11),DISTAN,-1,$E
DELETE $EP
$EP=LINE/P(12),PERP,$E,LENGTH,12.5
P(21)=POINT/P(12),DISTAN,12.5,$EP
P(16)=POINT/P(12),DISTAN,1,$E
P(17)=POINT/P(12),DISTAN,-1,$E
DELETE $E,$EP
$E=LINE/P(2),P(10)
$EP=LINE/P(13),PERP,$E,LENGTH,10
P(22)=POINT/P(13),DISTAN,10,$EP
P(18)=POINT/P(13),DISTAN,1,$E
P(19)=POINT/P(13),DISTAN,-1,$E
DELETE $E,$EP
<#
<#
L(1)=LINE/P(1),P(3)
L(2)=LINE/P(3),P(8)
L(3)=LINE/P(7),P(8)
L(4)=LINE/P(5),P(7),FONT,'DASH'
L(5)=LINE/P(14),P(20)
L(6)=LINE/P(15),P(20)
L(7)=LINE/P(16),P(21)
L(8)=LINE/P(17),P(21)
L(9)=LINE/P(2),P(4)
L(10)=LINE/P(4),P(8)
L(11)=LINE/P(6),P(7),FONT,'DASH'
L(12)=LINE/P(19),P(22)
L(13)=LINE/P(18),P(22)
L(14)=LINE/x(1)+5,y(5)+5,0,x(1)+5,y(3)+10,0,FONT,'ARROW'
L(15)=LINE/x(2)-5,y(5)+5,0,x(2)-5,y(3)+10,0,FONT,'ARROW'
C(1)=CIRCLE/RADIUS,HIP/3,LEFT,P(1),P(14),<#
>PGO,P(1),PEND,P(14)
C(2)=CIRCLE/RADIUS,HIP/3,LEFT,P(15),P(16),<#
>PGO,P(15),PEND,P(16)
C(3)=CIRCLE/RADIUS,HIP/3,LEFT,P(17),P(9),<#
>PGO,P(17),PEND,P(9)
C(4)=CIRCLE/RADIUS,HIP,LEFT,P(9),P(7),PGO,P(7),PEND,P(9)
C(5)=CIRCLE/RADIUS,HIP,RIGHT,P(19),P(2),PGO,P(19),PEND,P(2)
C(6)=CIRCLE/RADIUS,HIP/3,RIGHT,P(10),P(18),<#
>PGO,P(10),PEND,P(18)
C(7)=CIRCLE/RADIUS,HIP,RIGHT,P(10),P(7),PGO,P(10),PEND,P(7)
<#
<#
&T1='tailored skirt block'
T(1)=TEXT/&T1,(x(3)+x(8))/2,y(3)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(4)+x(8))/2,y(3)+2,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
T(3)=TEXT/&name+' < SIZE '+&n+' >',(x(3)+x(8))/2,y(3)+3<#
>,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name+' < SIZE '+&n+' >',(x(4)+x(8))/2,y(3)+3<#
>,0,TJST,'CJT',THGT,0.75
  OR &data='2'
IF (&name='') GOTO 200
T(3)=TEXT/&name,(x(3)+x(8))/2,y(3)+3,0,TJST,'CJT',THGT,0.75
```

```
T(4)=TEXT/&name,(x(4)+x(8))/2,y(3)+3,0,TJST,'CJT',THGT,0.75
#200 CONTINUE
ENDWHEN
T(5)=TEXT/'centre back line',x(1)+1,(y(5)+y(3))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(6)=TEXT/'centre front line',x(2)-1,(y(5)+y(3))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
DELETE 22,P(1)
DISP ALL
#999 END
```

```
+-----------------------------------------------------------+
|                                                           |
|              The circular skirt pattern (SKCI)            |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL x(6),y(6)
DECLARE LOCATION P(6)
DECLARE ENTITY L(4),T(5),C(2)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP SKSIZE(WAIST,HIP,WATOHI,WATOKN,&n,&data,&unit)
<#
<#
#10 CONTINUE
PRINT {''}
&angle=''
PRINT          Which circular skirt do you want?
PRINT    ------------------------------------
PRINT    The full circular skirt ------------------- 1
PRINT    The half circular skirt ------------------- 2
PRINT    ------------------------------------------
READ(          Enter the number please. ---> )&angle
WHEN &angle='1'
         &T1='full circular skirt'
  OR &angle='2'
         &T1='half circular skirt'
  ELSE
         PRINT Wrong answer, enter again please.
         GOTO 10
ENDWHEN
<#
<#
PRINT {''}
PRINT      Select the position for the centre<#
> waist of the skirt block please.
PRINT  ------------------------------------<#
>--------------------------
PRINT      Move the mouse on the pad to place the cursor<#
> on the graphic window
PRINT      at the position you require.<#
> Press the left button.
PRINT  ------------------------------------<#
>--------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
WHEN &angle='1'
         wradius=WAIST/6.28
  OR &angle='2'
```

```
        wradius=WAIST/3.14
ENDWHEN
x(1)=x(3)=:X
y(1)=Y
x(2)=x(1)+wradius
y(2)=y(4)=:y(1)+wradius
y(3)=y(1)-WATOKN
x(4)=x(2)+WATOKN
x(5)=x(1)+wradius/SQRT(2)
y(5)=y(1)+wradius-wradius/SQRT(2)
x(6)=x(1)+(WATOKN+wradius)/SQRT(2)
y(6)=y(1)+wradius-(WATOKN+wradius)/SQRT(2)
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>6
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
L(1)=LINE/P(1),P(3)
L(2)=LINE/P(2),P(4)
C(1)=CIRCLE/RADIUS,wradius,LEFT,P(1),P(2),<#
>PGO,P(1),PEND,P(2)
C(2)=CIRCLE/RADIUS,wradius+WATOKN,LEFT,P(3),P(4),<#
>PGO,P(3),PEND,P(4)
WHEN &angle='1'
L(3)=LINE/x(1)+3,y(1)-10,0,x(1)+3,y(3)+10,0,FONT,'ARROW'
  OR &angle='2'
L(3)=LINE/x(5)+10,y(5)-10,0,x(6)-10,y(6)+10,0,FONT,'ARROW'
        TRANSL L(3),INDIRV,-3,-3,0
        L(4)=LINE/P(5),P(6),FONT,'DASH'
ENDWHEN
<#
<#
T(1)=TEXT/&T1,x(6)-5,y(6)+5,0,TJST,'CJT',THGT,0.75,TANG,45
WHEN &data='1'
T(2)=TEXT/&name+' < SIZE '+&n+' >',x(6)-6,y(6)+6,0,<#
>TJST,'CJT',THGT,0.75,TANG,45
  OR &data='2'
IF (&name='') GOTO 200
T(2)=TEXT/&name,x(6)-6,y(6)+6,0,TJST,'CJT',<#
>THGT,0.75,TANG,45
#200 CONTINUE
ENDWHEN
T(3)=TEXT/'side seam',(x(2)+x(4))/2,y(2)-1,0,<#
>TJST,'CJT',THGT,0.75,TANG,180
WHEN &angle='1'
T(4)=TEXT/'centre (front or back) line',x(1)+1,<#
>(y(1)+y(3))/2,0,TJST,'CJT',THGT,0.75,TANG,270
  OR &angle='2'
T(4)=TEXT/'side seam',x(1)+1,(y(1)+y(3))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(5)=TEXT/'centre (front or back) line',<#
>x(5)+10,y(5)-10,0,TJST,'LJT',THGT,0.75,TANG,315
ENDWHEN
<#
<#
```

```
DISP ALL
#999 END
```

+----------+
| |
| The pleated skirt pattern (SKPL) |
| |
+----------+

```
<#
<#
DECLARE REAL x(12),y(12)
DECLARE LOCATION P(12)
DECLARE ENTITY L(12),C(3),T(2),HTCH(2),cp1(1000),<#
>cpc(200),cpt(100),cph(200)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP SKSIZE(WAIST,HIP,WATOHI,WATOKN,&n,&data,&unit)
<#
<#
#10 CONTINUE
PRINT {''}
&pleats=''
PRINT        Which way do you want to decide the pleats width?
PRINT ----------------------------------<#
>------------------------
WHEN &unit='CM'.OR.&unit='cm'
PRINT The width of pleats first (1.27 ~ 7.62 cm)<#
> ---------------- 1
  OR &unit='IN'.OR.&unit='in'
PRINT The width of pleats first (0.5 ~ 3 inch) <#
>------------------- 1
ENDWHEN
PRINT The number of pleats first (10 ~ 60)<#
> ---------------------- 2
PRINT ----------------------------------------<#
>----------------------
READ(              Enter the number please. ---> )&pleats
WHEN &pleats='1'
         PRINT {''}
         #11 CONTINUE
         WHEN &unit='CM'.OR.&unit='cm'
PRINT The pleats width must be in between 1.27 cm and 7.62 cm.
READ( Enter the pleats width please. <unit: cm> ---> ) pwd
           OR &unit='IN'.OR.&unit='in'
PRINT The pleats width must be in between 0.5 inch and 3 inch.
READ( Enter the pleats width please. <unit: in> ---> ) pwd
                  pwd=pwd*2.54
         ENDWHEN
         IF (pwd<1.27.OR.pwd>7.62) GOTO 11
  OR &pleats='2'
         PRINT {''}
         #12 CONTINUE
PRINT The pleats number must be a number in between 10 and 60.
READ( Enter the pleats number please. ---> )np
         IF (np<10.OR.np>60) GOTO 12
  ELSE
```

```
        PRINT Wrong answer, enter again please.
        GOTO 10
ENDWHEN
<#
<#
PRINT {''}
PRINT     Select the position for the centre back<#
> waist of the skirt please.
PRINT  ------------------------------------<#
>--------------------------
PRINT     Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT     at the position you require.<#
> Press the left button.
PRINT  ------------------------------------<#
>--------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
WHEN &pleats='1'
        number=(HIP+2)/pwd
        &number=number
        WHEN &number(2)='.'.OR.&number(2)=''
                nl=&number(1)
                WHEN number-nl>=0.5
                        np=nl+1
          ELSE
                        np=nl
      ENDWHEN
OR &number(3)='.'.OR.&number(3)=''
        nl=&number(,2)
        WHEN number-nl>=0.5
                        np=nl+1
          ELSE
                        np=nl
        ENDWHEN
ELSE
        nl=&number(,3)
        WHEN number-nl>=0.5
                        np=nl+1
          ELSE
                        np=nl
                      ENDWHEN
          ENDWHEN
          wwd=(WAIST+1)/np
  OR &pleats='2'
          pwd=(HIP+2)/np
          wwd=(WAIST+1)/np
ENDWHEN
<#
<#
x(1)=X
y(1)=y(4)=:y(7)=:y(10)=:Y
x(2)=x(3)=:x(1)-(pwd-wwd)/2
y(2)=y(5)=:y(8)=:y(11)=:y(1)-WATOHI
y(3)=y(6)=:y(9)=:y(12)=:y(1)-WATOKN
```

```
x(4)=x(1)+wwd
x(5)=x(6)=:x(2)+pwd
x(7)=x(8)=:x(9)=:x(5)+pwd
x(10)=x(1)+pwd*3
x(11)=x(12)=:x(8)+pwd
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>12
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
L(1)=LINE/P(2),P(3)
L(2)=LINE/P(1),P(4)
L(3)=LINE/P(3),P(6)
L(4)=LINE/P(5),P(6)
L(5)=LINE/P(4),P(7)
L(6)=LINE/P(6),P(9)
L(7)=LINE/P(7),P(9)
L(8)=LINE/P(7),P(10)
L(9)=LINE/P(9),P(12)
L(10)=LINE/P(11),P(12)
L(11)=LINE/P(2),P(11),FONT,'DASH'
L(12)=LINE/(x(5)+x(8))/2,y(2)-5,0,(x(5)+x(8))/2,<#
>y(3)+5,0,FONT,'ARROW'
C(1)=CIRCLE/RADIUS,HIP*5,RIGHT,P(1),P(2),PGO,P(1),PEND,P(2)
C(2)=CIRCLE/RADIUS,HIP*5,LEFT,P(5),P(4),PGO,P(5),PEND,P(4)
C(3)=CIRCLE/RADIUS,HIP*5,RIGHT,P(10),P(11),<#
>PGO,P(10),PEND,P(11)
<#
<#
WHEN &data='1'
T(1)=TEXT/&name+' < SIZE '+&n+' >',x(2)+0.15,y(3)+2,0,<#
>TJST,'RJT',THGT,0.75,TANG,270
  OR &data='2'
IF (&name='') GOTO 200
T(1)=TEXT/&name,x(2)+0.15,y(3)+2,0,<#
>TJST,'RJT',THGT,0.75,TANG,270
#200 CONTINUE
ENDWHEN
T(2)=TEXT/'pleat line',x(5)-0.1,(y(2)+y(3))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
@V=VECT(pwd*3,0,0)
TRANSL (COPY,np-1,cpl(1)) 10,L(2),INDIRV,@V
TRANSL (COPY,np-1,cpc(1)) 2,C(2),INDIRV,@V
TRANSL (COPY,np-1,cpt(1)) T(2),INDIRV,@V
<#
<#
DISP ALL
#999 END
```

```
+-----------------------------------------------------------+
|                                                           |
|              The gathered skirt pattern (SKGA)            |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL x(6),y(6)
DECLARE LOCATION P(6)
DECLARE ENTITY L(7),T(10),$C,cp(6)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP SKSIZE(WAIST,HIP,WATOHI,WATOKN,&n,&data,&unit)
<#
<#
#10 CONTINUE
PRINT {''}
&waist=''
PRINT          Which gathered skirt do you want?
PRINT ------------------------------------------
PRINT Slightly gathered skirt (with waist curve) ----- 1
PRINT Very gathered skirt (rectangle) ---------------- 2
PRINT ------------------------------------------------
READ(          Enter the number please. ---> )&waist
WHEN &waist='1'.OR.&waist='2'
        CONTINUE
  ELSE
        PRINT Wrong answer, enter again please.
        GOTO 10
ENDWHEN
<#
<#
PRINT {''}
PRINT      Select the position for the centre back<#
> waist of the skirt please.
PRINT  ------------------------------------<#
>----------------------------
PRINT      Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT      at the position you require. <#
>Press the left button.
PRINT  ------------------------------------<#
>----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
WHEN &waist='1'
        GOSUB 1000
  OR &waist='2'
        GOSUB 2000
ENDWHEN
```

```
#999 END
<#
<#
#1000 CONTINUE
x(1)=x(2)=:X
y(1)=y(5)=:Y
y(2)=y(4)=:Y-WATOKN
x(3)=x(4)=:x(1)+HIP/4+15
y(3)=y(1)+1.5
x(5)=x(1)+(x(3)-x(1))/3
x(6)=(x(5)+x(3))/2
y(6)=y(1)+1.5/3
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>6
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
L(1)=LINE/P(1),P(2)
L(2)=LINE/P(2),P(4)
L(3)=LINE/P(3),P(4)
L(4)=LINE/P(1),P(5)
L(5)=LINE/(x(1)+x(3))/2,y(1)-10,0,(x(1)+x(3))/2,<#
>y(2)+10,0,FONT,'ARROW'
$C=CIRCLE/P(5),P(6),P(3),PGO,P(3),PEND,P(5)
<#
<#
MIRROR (COPY,cp(1)) 5,L(1),MX,x(3)+5,y(3),0
MIRROR (COPY,cp(6)) $C,MX,x(3)+5,y(3),0
<#
<#
&T1='gathered skirt'
T(1)=TEXT/&T1,(x(2)+x(4))/2,y(2)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,x(4)*3/2-x(2)/2+10,y(2)+2,0,<#
>TJST,'CJT',THGT,0.75
WHEN &data='1'
T(3)=TEXT/&name+' < SIZE '+&n+' >',(x(2)+x(4))/2,<#
>y(2)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name+' < SIZE '+&n+' >',x(4)*3/2-x(2)/2+10,<#
>y(2)+3,0,TJST,'CJT',<#
>THGT,0.75
  OR &data='2'
IF (&name='') GOTO 200
T(3)=TEXT/&name,(x(2)+x(4))/2,y(2)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name,x(4)*3/2-x(2)/2+10,y(2)+3,0,<#
>TJST,'CJT',THGT,0.75
#200 CONTINUE
ENDWHEN
T(5)=TEXT/'centre back fold',x(1)+1,(y(1)+y(2))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(6)=TEXT/'centre front fold',x(4)*2-x(1)+9,<#
>(y(1)+y(2))/2,0,TJST,'CJT',THGT,0.75,TANG,90
T(7)=TEXT/'<----- gather ----->',(x(2)+x(4))/2,y(1)-1.5,<#
>0,TJST,'CJT',THGT,0.75
T(8)=TEXT/'<----- gather ----->',x(4)*3/2-x(2)/2+10,<#
>y(1)-1.5,0,TJST,'CJT',THGT,0.75
```

```
T(9)=TEXT/'side seam',x(3)-1,(y(1)+y(2))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
T(10)=TEXT/'side seam',x(3)+11,(y(1)+y(2))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
<#
<#
!REGENERATE GRAPHICS
EXECDF
DISP ALL
RTNSUB
<#
<#
#2000 CONTINUE
x(1)=x(2)=:X
y(1)=y(3)=:y(5)=:Y
y(2)=y(4)=:y(6)=:Y-WATOKN
x(3)=x(4)=:X+HIP*1.5
x(5)=x(6)=:(x(1)+x(3))/2
<#
<#
DISP OFF
REPEAT 300: i=1,1,i>6
P(i)=VECT(x(i),y(i),0)
#300 CONTINUE
<#
<#
L(1)=LINE/P(1),P(2)
L(2)=LINE/P(2),P(4)
L(3)=LINE/P(3),P(4)
L(4)=LINE/P(1),P(3)
L(5)=LINE/P(5),P(6)
L(6)=LINE/(x(1)+x(5))/2,y(1)-10,0,(x(1)+x(5))/2,<#
>y(2)+10,0,FONT,'ARROW'
L(7)=LINE/(x(3)+x(5))/2,y(1)-10,0,(x(3)+x(5))/2,<#
>y(2)+10,0,FONT,'ARROW'
<#
<#
&T1='gathered skirt'
T(1)=TEXT/&T1,(x(2)+x(6))/2,y(2)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(4)+x(6))/2,y(2)+2,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
T(3)=TEXT/&name+' < SIZE '+&n+' >',(x(2)+x(6))/2,<#
>y(2)+3,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&name+' < SIZE '+&n+' >',(x(4)+x(6))/2,<#
>y(2)+3,0,TJST,'CJT',THGT,0.75
  OR &data='2'
IF (&name='') GOTO 400
T(3)=TEXT/&name,(x(2)+x(6))/2,y(2)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name,(x(4)+x(6))/2,y(2)+3,0,TJST,'CJT',THGT,0.75
#400 CONTINUE
ENDWHEN
T(5)=TEXT/'centre back fold',x(1)+1,(y(1)+y(2))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(6)=TEXT/'centre front fold',x(3)-1,(y(1)+y(2))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
T(7)=TEXT/'<----- gather ----->',(x(1)+x(5))/2,y(1)-1.5,<#
>0,TJST,'CJT',THGT,0.75
T(8)=TEXT/'<----- gather ----->',(x(5)+x(3))/2,y(1)-1.5,<#
```

```
>0,TJST,'CJT',THGT,0.75
T(9)=TEXT/'side seam',x(5)-1,(y(1)+y(2))/2,0,<#
>TJST,'CJT',THGT;0.75,TANG,90
T(10)=TEXT/'side seam',x(5)+1,(y(1)+y(2))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
<#
<#
DISP ALL
RTNSUB
```

```
+-----------------------------------------------------------+
|                                                           |
|              TRBALK (Linkfile: TRBA, TRSIZE)              |
|                                                           |
+-----------------------------------------------------------+

MAIN TRBA
IMAGE TRBAIMAGE
LINKM TRSIZE
ENDLINK
```

```
+-----------------------------------------------------------+
|                                                           |
|              TREALK (Linkfile: TREA, TRSIZE)              |
|                                                           |
+-----------------------------------------------------------+

MAIN TREA
IMAGE TREAIMAGE
LINKM TRSIZE
ENDLINK
```

```
+-----------------------------------------------------------+
|                                                           |
|              TRCULK (Linkfile: TRCU, TRSIZE)              |
|                                                           |
+-----------------------------------------------------------+

MAIN TRCU
IMAGE TRCUIMAGE
LINKM TRSIZE
ENDLINK
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|                  The trouser size (TRSIZE) procedure                  |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
PROC TRSIZE(WAIST,BDRISE,HIP,WATOFL,WATOHI,TWID,JWID,<#
>WATOKN,&n,&data,&type,&unit)
DECLARE REAL waist(12),bdrise(12),hip(12),watofl(12),<#
>watohi(12),twid(12),jwid(12),watokn(12)
DATA waist/60,64,68,72,77,82,87,92,97,102,107,112/
DATA bdrise/26.6,27.3,28,28.7,29.4,30.1,30.8,31.5,<#
>32.5,33.5,34.5,35.5/
DATA hip/85,89,93,97,102,107,112,117,122,127,132,137/
DATA watofl/102,103,104,105,106,107,108,109,109.5,<#
>110,110.5,111/
DATA watohi/20,20.3,20.6,20.9,21.2,21.5,21.8,22.1,<#
>22.3,22.5,22.7,22.9/
DATA twid/21,21.5,22,22.5,23,23.5,24,24.5,25.4,<#
>26.2,27,27.8/
DATA jwid/18.5,18.5,19,19,19.5,20,20.5,21,21.5,<#
>22,22.5,23/
DATA watokn/57.5,58,58.5,59,59.5,60,60.5,61,<#
>61.25,61.5,61.75,62/
ERTRAP 999
<#
<#
#10 CONTINUE
PRINT {''}
&data=''
PRINT          Which measurements do you want to use?
PRINT      ----------------------------------------
PRINT      The body measurements of this program --- 1
PRINT      Your individual body measurements ------- 2
PRINT      ----------------------------------------
READ(              Enter the number please. ---> )&data
WHEN &data='1'.OR.&data='2'
          CONTINUE
  ELSE
          PRINT Wrong answer, enter again please.
          GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
PRINT {''}
&unit=''
READ( Which units do you want to use?  <#
>  Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'.OR.&unit='IN'.OR.&unit='in'
          CONTINUE
  ELSE
          PRINT Wrong answer, enter again please.
          GOTO 20
ENDWHEN
<#
<#
```

```
#30 CONTINUE
PRINT {''}
&n=''
WHEN &unit='CM'.OR.&unit='cm'
        GOSUB 1000
  ELSE
        GOSUB 2000
ENDWHEN
PRINT      The size must be an even number between 8 and 30.
WHEN &data='1'
READ(                Enter the size please. ---> )&n
ELSE
READ(                Enter the nearest size please. ---> )&n
ENDWHEN
WHEN &n='8'.OR.&n='10'.OR.&n='12'.OR.&n='14'.OR.&n='16'<#
>.OR.&n='18'.OR.&n='20'.OR.&n='22'.OR.&n='24'.OR.&n='26'<#
>.OR.&n='28'.OR.&n='30'
         n=&n
         i=(n-6)/2
         WAIST=waist(i)
         BDRISE=bdrise(i)
         HIP=hip(i)
         WATOFL=watofl(i)
         WATOHI=watohi(i)
         TWID=twid(i)
         JWID=jwid(i)
         WATOKN=watokn(i)
   ELSE
         PRINT Wrong value entered, enter again please.
         GOTO 30
ENDWHEN
<#
<#
#40 CONTINUE
&height=''
PRINT {''}
WHEN &unit='CM'.OR.&unit='cm'
     PRINT                    Which height do you want?
     PRINT  ------------------------------------------
     PRINT  The short women ( 152cm - 160cm ) --------- S
     PRINT  The medium women ( 160cm - 170cm ) -------- M
     PRINT  The tall women ( 170cm - 178cm ) ---------- T
     PRINT  ------------------------------------------
   ELSE
     PRINT                    Which height do you want?
     PRINT  ------------------------------------------
     PRINT  The short women (5ft - 5ft 3in) ----------- S
     PRINT  The medium women (5ft 3in - 5ft 7in) ------ M
     PRINT  The tall women (5ft 7in - 5ft 10in) ------- T
     PRINT  ------------------------------------------
ENDWHEN
WHEN &data='1'
READ(   Enter the height please.  S or M or T ---> )&height
ELSE
READ( Enter the nearest height please. <#
> S or M or T ---> )&height
ENDWHEN
WHEN &height='S'.OR.&height='s'
```

```
        WATOFL=WATOFL-5
        BDRISE=BDRISE-1
        WATOKN=WATOKN-3
OR &height='M'.OR.&height='m'
        CONTINUE
OR &height='T'.OR.&height='t'
        WATOFL=WATOFL+5
        BDRISE=BDRISE+1
        WATOKN=WATOKN+3
ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 40
ENDWHEN
<#
<#
IF (&unit='CM'.OR.&unit='cm') GOTO 50
WAIST=WAIST/2.54
BDRISE=BDRISE/2.54
HIP=HIP/2.54
WATOFL=WATOFL/2.54
WATOHI=WATOHI/2.54
TWID=TWID/2.54
JWID=JWID/2.54
WATOKN=WATOKN/2.54
<#
<#
#50 CONTINUE
&WAIST=WAIST
&BDRISE=BDRISE
&HIP=HIP
&WATOFL=WATOFL
&WATOHI=WATOHI
&TWID=TWID
&JWID=JWID
&WATOKN=WATOKN
<#
<#
WHEN &data='l'
        PRINT {''}
        &change=''
PRINT       The data you entered is <#
>-----------------------------------
PRINT          size; {&n}          height; {&height}<#
>              unit; {&unit}
PRINT          ------------------------------------<#
>-------------------------
PRINT          waist={&WAIST(,4)}              <#
>body rise={&BDRISE(,4)}        hip={&HIP(,4)}
WHEN &type='basic'.OR.&type='ease'
PRINT          waist to floor={&WATOFL(,4)}          <#
>waist to hip={&WATOHI(,4)}
PRINT          trouser bottom width={&TWID(,4)}
OR &type='close'
PRINT          waist to floor={&WATOFL(,4)}          <#
>waist to hip={&WATOHI(,4)}
PRINT          jeans bottom width={&JWID(,4)}
OR &type='culotte'
PRINT          waist to hip={&WATOHI(,4)}       <#
```

```
> waist to knee={&WATOKN(,4)}
ENDWHEN
PRINT     ----------------------------------<#
>-------------------------
READ(          Do you want to change?      Y/N ---> )&change
        WHEN &change='Y'.OR.&change='y'
                GOTO 10
          OR &change='N'.OR.&change='n'
                CONTINUE
          ELSE
            PRINT Wrong value entered, enter again please.
                GOTO 50
        ENDWHEN
  OR &data='2'
        PRINT {''}
PRINT   ----------------------------------<#
>-------------------------
WHEN &unit='CM'.OR.&unit='cm'
PRINT          Enter your individual measurements<#
> please.   (unit: cm)
ELSE
PRINT          Enter your individual measurements<#
> please.   (unit: in)
ENDWHEN
PRINT   If you don't want to change the measurement<#
> of the part, press RETURN.
PRINT   ----------------------------------<#
>-------------------------
PRINT waist (size;{&n}, height;{&height}) = {&WAIST(,4)}
READ(      your waist ---> )WAIST
PRINT body rise (size;{&n}, height;{&height}) =<#
> {&BDRISE(,4)}
READ(      your body rise ---> )BDRISE
PRINT hip (size;{&n}, height;{&height}) = {&HIP(,4)}
READ(      your hip ---> )HIP
WHEN &type='basic'.OR.&type='ease'
PRINT waist to floor (size;{&n}, height;{&height})<#
> = {&WATOFL(,4)}
READ(      your waist to floor < or trouser length ><#
> ---> )WATOFL
PRINT waist to hip (size;{&n}, height;{&height})<#
> = {&WATOHI(,4)}
READ(      your waist to hip ---> )WATOHI
PRINT trouser bottom width (size;{&n},<#
> height;{&height}) = {&TWID(,4)}
READ(      your trouser bottom width ---> )TWID
OR &type='close'
PRINT waist to floor (size;{&n}, height;{&height})<#
> = {&WATOFL(,4)}
READ(      your waist to floor < or trouser length ><#
> ---> )WATOFL
PRINT waist to hip (size;{&n}, height;{&height})<#
> = {&WATOHI(,4)}
READ(      your waist to hip ---> )WATOHI
PRINT jeans bottom width (size;{&n},<#
> height;{&height}) = {&JWID(,4)}
READ(      your jeans bottom width ---> )JWID
OR &type='culotte'
```

```
PRINT waist to hip (size;{&n}, height;{&height})<#
> = {&WATOHI(,4)}
READ(      your waist to hip ---> )WATOHI
PRINT waist to knee (size;{&n}, height;{&height})<#
> = {&WATOKN(,4)}
READ(      your waist to knee < or culottes length ><#
> ---> )WATOKN
ENDWHEN
<#
<#
#60 CONTINUE
&WAIST=WAIST
&BDRISE=BDRISE
&HIP=HIP
&WATOFL=WATOFL
&WATOHI=WATOHI
&TWID=TWID
&JWID=JWID
&WATOKN=WATOKN
<#
<#
PRINT {''}
&change=''
PRINT      The data you entered is<#
> ------------------------------------
PRINT         nearest size; {&n}      <#
>nearest height; {&height}    unit; {&unit}
PRINT         ------------------------------<#
>-------------------------------
PRINT         waist={&WAIST(,4)}          <#
>body rise={&BDRISE(,4)}         hip={&HIP(,4)}
WHEN &type='basic'.OR.&type='ease'
PRINT         waist to floor (trouser length)={&WATOFL(,4)}<#
>   waist to hip={&WATOHI(,4)}
PRINT         trouser bottom width={&TWID(,4)}
OR &type='close'
PRINT         waist to floor (trouser length)={&WATOFL(,4)}<#
>   waist to hip={&WATOHI(,4)}
PRINT         jeans bottom width={&JWID(,4)}
OR &type='culotte'
PRINT         waist to hip={&WATOHI(,4)}<#
>    waist to knee (culottes length)={&WATOKN(,4)}
ENDWHEN
PRINT      ------------------------------<#
>-------------------------------
READ(            Do you want to change?     Y/N ---> )&change
        WHEN &change='Y'.OR.&change='y'
             GOTO 10
          OR &change='N'.OR.&change='n'
             CONTINUE
          ELSE
            PRINT Wrong value entered, enter again please.
             GOTO 60
        ENDWHEN
ENDWHEN
<#
<#
IF (&unit='CM'.OR.&unit='cm') GOTO 999
```

```
WAIST=WAIST*2.54
BDRISE=BDRISE*2.54
HIP=HIP*2.54
WATOFL=WATOFL*2.54
WATOHI=WATOHI*2.54
TWID=TWID*2.54
JWID=JWID*2.54
WATOKN=WATOKN*2.54
<#
<#
#999 RETURN
<#
<#
#1000 CONTINUE
PRINT                    WOMEN OF MEDIUM HEIGHT 160CM-170CM<#
>   (UNIT: CM)
PRINT     ------------------------------------------<#
>-------------------------
PRINT                  8    10    12    14    16    18    20   <#
> 22    24    26    28    30
PRINT     ------------------------------------------<#
>-------------------------
PRINT      WAIST      60    64    68    72    77    82    87 <#
>  92    97   102   107   112
PRINT      HIPS       85    89    93    97   102   107   112 <#
> 117   122   127   132   137
PRINT      WA-TO-HI   20    20.3 20.6 20.9 21.2 21.5 21.8<#
> 22.1 22.3 22.5 22.7 22.9
PRINT      WA-TO-FL 102   103   104   105   106   107   108  <#
>109   109   110   110   111
PRINT      BODY RISE 26.6 27.3 28    28.7 29.4 30.1 30.8<#
> 31.5 32.5 33.5 34.5 35.5
PRINT     ------------------------------------------<#
>-------------------------
PRINT                  WA-TO-HI: waist to hip        <#
> WA-TO-FL: waist to floor
PRINT {''}
RTNSUB
<#
<#
#2000 CONTINUE
PRINT                  WOMEN OF MEDIUM HEIGHT 5FT 3IN -<#
> 5FT 7IN (UNIT: IN)
PRINT     ------------------------------------------<#
>-------------------------
PRINT                  8    10    12    14    16    18    20   <#
> 22    24    26    28    30
PRINT     ------------------------------------------<#
>-------------------------
PRINT      WAIST      23.6 25.2 26.8 28.3 30.3 32.3 34.3<#
> 36.2 38.2 40.2 42.1 44.1
PRINT      HIPS       33.5 35   36.6 38.2 40.2 42.1 44.1<#
> 46.1 48   50   52   53.9
PRINT      WA-TO-HI   7.9  8    8.1  8.2  8.3  8.5  8.6<#
>  8.7  8.8  8.9  8.9  9
PRINT      WA-TO-FL  40.2 40.6 40.9 41.3 41.7 42.1 42.5<#
> 42.9 42.9 43.3 43.3 43.7
PRINT      BODY RISE 10.5 10.7 11   11.3 11.6 11.9 12.1<#
```

```
> 12.4 12.8 13.2 13.6 14                                    -
PRINT     ----------------------------------------------<#
>-------------------------
PRINT                  WA-TO-HI: waist to hip          WA-TO-FL: waist to f
PRINT {''}
RTNSUB
```

```
+-----------------------------------------------------------+
|                                                           |
|          The basic trouser block pattern (TRBA)           |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL x(54),y(53)
DECLARE ENTITY P(53),L(28),C(5),B(3),T(4),cp(1)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
&type='basic'
CALLP TRSIZE(WAIST,BDRISE,HIP,WATOFL,WATOHI,TWID,JWID,<#
>WATOKN,&n,&data,&type,&unit)
<#
<#
PRINT {''}
PRINT    Select the position for the waist point <#
>of trouser block please.
PRINT  ------------------------------------<#
>----------------------------
PRINT     Move the mouse on the pad to place the <#
>cursor on the graphic window
PRINT     at the position you require. Press the <#
>left button.
PRINT  ------------------------------------<#
>----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(3)=:x(2)=:x(5)=:x(4)=:x(27)=:x(28)=:x(29)=:X
y(1)=y(10)=:y(11)=:y(18)=:y(28)=:y(30)=:y(31)=:Y
y(2)=y(9)=:y(6)=:y(1)-BDRISE
y(3)=y(7)=:y(8)=:y(20)=:y(1)-WATOHI
y(4)=y(23)=:y(14)=:y(12)=:y(21)=:y(1)-WATOFL
y(5)=y(24)=:y(15)=:y(13)=:y(22)=:y(2)-(y(2)-y(4))/2+5
x(6)=x(7)=:x(1)-HIP/12-1.5
x(8)=x(6)+HIP/4+0.5
x(9)=x(6)-HIP/16-0.5
x(10)=x(6)+1
x(11)=x(10)+WAIST/4+2.25
x(12)=x(4)+TWID/2-0.5
WHEN HIP<100
        x(13)=x(4)+TWID/2+0.8
        x(15)=x(4)-TWID/2-0.8
  OR HIP.GE.100.AND.HIP<120
        x(13)=x(4)+TWID/2+1
        x(15)=x(4)-TWID/2-1
  ELSE
        x(13)=x(4)+TWID/2+1.2
        x(15)=x(4)-TWID/2-1.2
```

```
ENDWHEN
x(14)=x(4)-TWID/2+0.5
x(16)=x(6)+(x(2)-x(6))/4
y(16)=(y(1)+y(2))/2
x(17)=x(16)+2
y(17)=y(1)+2
x(18)=SQRT((WAIST/4+4.25)**2-(y(17)-y(18))**2)+x(17)
x(19)=x(9)-(x(6)-x(9))/2
y(19)=y(2)-0.5
x(20)=x(16)+HIP/4+1.5
x(21)=x(12)+1
x(22)=x(13)+1
x(23)=x(14)-1
x(24)=x(15)-1
WHEN HIP<100
        backcurve=4.25
        frontcurve=3
  OR HIP.GE.100.AND.HIP<120
        backcurve=4.5
        frontcurve=3.25
  ELSE
        backcurve=4.75
        frontcurve=3.5
ENDWHEN
x(25)=x(6)-frontcurve/SQRT(2)
y(25)=y(6)+frontcurve/SQRT(2)
x(26)=x(16)-backcurve/SQRT(2)
y(26)=y(2)+backcurve/SQRT(2)
y(27)=y(4)-1
y(29)=y(1)-10
x(30)=x(1)-1
x(31)=x(1)+1
x(32)=(x(8)+x(11))/2
y(32)=(y(8)+y(11))/2
x(33)=(x(18)+x(20))/2
y(33)=(y(18)+y(20))/2
x(34)=(x(20)+x(22))/2
y(34)=(y(20)+y(22))/2
x(35)=(x(9)+x(15))/2
y(35)=(y(9)+y(15))/2
x(36)=(x(19)+x(24))/2
y(36)=(y(19)+y(24))/2
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>36
P(i)=POINT/x(i),y(i),0
#100 CONTINUE
<#
<#
L(1)=LINE/P(29),P(4),FONT,'DASH'
L(2)=LINE/P(10),P(30)
L(3)=LINE/P(30),P(29)
L(4)=LINE/P(31),P(29)
L(5)=LINE/P(31),P(11)
$E=LINE/P(11),P(8)
$EP=LINE/P(32),PERP,$E,LENGTH,0.5
P(37)=POINT/P(32),DISTAN,0.5,$EP
```

```
DELETE $E,$EP
C(1)=CIRCLE/P(11),P(37),P(8),PGO,P(8),PEND,P(11)
L(6)=LINE/P(8),P(13)
L(7)=LINE/P(13),P(12)
L(8)=LINE/P(10),P(7)
B(1)=BSPLIN/3,P(7),P(25),P(9)
$E=LINE/P(9),P(15)
$EP=LINE/P(35),PERP,$E,LENGTH,0.75
P(38)=POINT/P(35),DISTAN,0.75,$EP
DELETE $E,$EP
C(2)=CIRCLE/P(9),P(38),P(15),PGO,P(15),PEND,P(9)
L(9)=LINE/P(15),P(14)
L(10)=LINE/P(14),P(12)
L(11)=LINE/P(7),P(8),FONT,'DASH'
$E=LINE/P(2),HORIZ,LENGTH,60
P(39)=POINT/INTOF,$E,L(6)
DELETE $E
L(12)=LINE/P(9),P(39),FONT,'DASH'
L(13)=LINE/P(15),P(13),FONT,'DASH'
<#
<#
$E=LINE/P(18),P(17)
x(40)=x(17)+(x(18)-x(17))/3
y(40)=y(18)+(y(17)-y(18))*2/3
x(41)=x(17)+(x(18)-x(17))*2/3
y(41)=y(18)+(y(17)-y(18))/3
P(40)=POINT/x(40),y(40),0
P(41)=POINT/x(41),y(41),0
$EP1=LINE/P(40),PERP,$E,LENGTH,12
P(42)=POINT/P(40),DISTAN,12,$EP1
$EP2=LINE/P(41),PERP,$E,LENGTH,10
P(43)=POINT/P(41),DISTAN,10,$EP2
P(44)=POINT/P(40),DISTAN,1,$E
P(45)=POINT/P(40),DISTAN,-1,$E
P(46)=POINT/P(41),DISTAN,1,$E
P(47)=POINT/P(41),DISTAN,-1,$E
DELETE $E,$EP1,$EP2
L(14)=LINE/P(17),P(44)
L(15)=LINE/P(44),P(42)
L(16)=LINE/P(45),P(42)
L(17)=LINE/P(45),P(46)
L(18)=LINE/P(46),P(43)
L(19)=LINE/P(47),P(43)
L(20)=LINE/P(47),P(18)
$E=LINE/P(18),P(20)
$EP=LINE/P(33),PERP,$E,LENGTH,0.5
P(48)=POINT/P(33),DISTAN,0.5,$EP
DELETE $E,$EP
C(3)=CIRCLE/P(18),P(48),P(20),PGO,P(20),PEND,P(18)
$E=LINE/P(22),P(20)
$EP=LINE/P(34),PERP,$E,LENGTH,0.5
P(49)=POINT/P(34),DISTAN,0.5,$EP
DELETE $E,$EP
B(2)=BSPLIN/3,P(20),P(49),P(22)
L(21)=LINE/P(22),P(21)
C(4)=CIRCLE/P(23),P(27),P(21),PGO,P(21),PEND,P(23)
L(22)=LINE/P(24),P(23)
$E=LINE/P(19),P(24)
```

```
$EP=LINE/P(36),PERP,$E,LENGTH,1.25
P(50)=POINT/P(36),DISTAN,1.25,$EP
DELETE $E,$EP
C(5)=CIRCLE/P(24),P(50),P(19),PGO,P(19),PEND,P(24)
B(3)=BSPLIN/3,P(16),P(26),P(19)
L(23)=LINE/P(16),P(17)
$E=LINE/P(28),VERT,LENGTH,2
P(51)=POINT/INTOF,$E,L(14)
DELETE $E
L(24)=LINE/P(51),P(27),FONT,'DASH'
L(25)=LINE/P(24),P(22),FONT,'DASH'
$E=LINE/P(7),HORIZ,LENGTH,20
P(52)=POINT/INTOF,$E,XSMALL,B(3)
DELETE $E
L(26)=LINE/P(52),P(20),FONT,'DASH'
$E=LINE/P(39),HORIZ,LENGTH,9
P(53)=POINT/INTOF,$E,XSMALL,B(2)
DELETE $E
L(27)=LINE/P(9),P(53),FONT,'DASH'
L(28)=LINE/x(1)-3,y(2)-10,0,x(1)-3,y(4)+15,0,FONT,'ARROW'
DELETE 53,P(1)
<#
<#
MIRROR 13,L,MX,x(20)+3,0,0
MIRROR C(1),C(2),MX,x(20)+3,0,0
MIRROR B(1),MX,x(20)+3,0,0
MIRROR (COPY,cp(1)) L(28),MX,x(20)+3,0,0
<#
<#
x(54)=(x(20)+3)*2-x(1)
&T1='basic trouser block (back)'
T(1)=TEXT/&T1,x(1)+1,y(5),0,TJST,'CJT',THGT,0.75,TANG,270
&T2='basic trouser block (front)'
T(2)=TEXT/&T2,x(54)-1,y(5),0,TJST,'CJT',THGT,0.75,TANG,90
WHEN &data='1'
T(3)=TEXT/&name+' < SIZE '+&n+' >',x(1)+2,y(5),0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(4)=TEXT/&name+' < SIZE '+&n+' >',x(54)-2,y(5),0,<#
>TJST,'CJT',THGT,0.75,TANG,90
  OR &data='2'
IF (&name='') GOTO 200
T(3)=TEXT/&name,x(1)+2,y(5),0,TJST,'CJT',THGT,0.75,TANG,270
T(4)=TEXT/&name,x(54)-2,y(5),0,TJST,'CJT',THGT,0.75,TANG,90
#200 CONTINUE
ENDWHEN
<#
<#
!REGENERATE GRAPHICS
EXECDF
DISP ALL
#999 END
```

```
+-----------------------------------------------------------+
|                                                           |
|        The easy fitting trouser block pattern (TREA)      |
|                                                           |
+-----------------------------------------------------------+
<#
<#
<#
DECLARE REAL x(30),y(29)
DECLARE ENTITY P(35),L(20),B(2),C(5),T(4),cp(1)
ERTRAP 999
<#
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
&type='ease'
CALLP TRSIZE(WAIST,BDRISE,HIP,WATOFL,WATOHI,TWID,JWID,<#
>WATOKN,&n,&data,&type,&unit)
<#
<#
<#
PRINT {''}
PRINT    Select the position for the waist point<#
> of the trouser block please.
PRINT  ------------------------------------<#
>------------------------------
PRINT     Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT     at the position you require. Press the <#
>left button.
PRINT  ------------------------------------<#
>------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
<#
x(1)=x(2)=:x(3)=:x(4)=:x(5)=:X
y(1)=y(10)=:y(11)=:y(19)=:Y
y(2)=y(9)=:y(6)=:y(16)=:y(1)-BDRISE-1
y(3)=y(7)=:y(8)=:y(21)=:y(1)-WATOHI
y(4)=y(24)=:y(14)=:y(12)=:y(22)=:y(1)-WATOFL
y(5)=y(25)=:y(15)=:y(13)=:y(23)=:y(2)-(y(2)-y(4))/2+5
x(6)=x(7)=:x(1)-HIP/12-1.8
x(8)=x(6)+HIP/4+1
x(9)=x(6)-HIP/16-1
x(10)=x(6)+1
x(11)=x(10)+WAIST/4+5
x(12)=x(4)+TWID/2-0.5
x(13)=x(12)+(y(13)-y(12))*(x(8)-x(12))/(y(8)-y(12))
x(14)=x(4)-TWID/2+0.5
x(15)=x(5)-(x(13)-x(5))
x(16)=x(17)=:x(6)+(x(1)-x(6))/4
y(17)=(y(1)+y(2))/2
x(18)=x(16)+2
y(18)=y(1)+2
x(19)=SQRT((WAIST/4+6)**2-(y(18)-y(19))**2)+x(18)
x(20)=x(9)-(x(6)-x(9))/2
```

```
y(20)=y(2)-0.25
x(21)=x(16)+HIP/4+2
x(22)=x(12)+1
x(23)=x(13)+1
x(24)=x(14)-1
x(25)=x(15)-1
WHEN HIP<100
        backcurve=4.25
        frontcurve=3
  OR HIP.GE.100.AND.HIP<120
        backcurve=4.5
        frontcurve=3.25
  ELSE
        backcurve=4.75
        frontcurve=3.5
ENDWHEN
x(26)=x(6)-frontcurve/SQRT(2)
y(26)=y(6)+frontcurve/SQRT(2)
x(27)=x(16)-backcurve/SQRT(2)
y(27)=y(16)+backcurve/SQRT(2)
x(28)=(x(9)+x(15))/2
y(28)=(y(9)+y(15))/2
x(29)=(x(20)+x(25))/2
y(29)=(y(20)+y(25))/2
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>29
P(i)=POINT/x(i),y(i),0
#100 CONTINUE
<#
<#
L(1)=LINE/P(1),P(4),FONT,'DASH'
L(2)=LINE/P(10),P(11)
C(1)=CIRCLE/RADIUS,HIP*2,LEFT,P(8),P(11),<#
>PGO,P(8),PEND,P(11)
L(3)=LINE/P(8),P(12)
L(4)=LINE/P(12),P(14)
L(5)=LINE/P(15),P(14)
$E=LINE/P(9),P(15)
$EP=LINE/P(28),PERP,$E,LENGTH,0.75
P(30)=POINT/P(28),DISTAN,0.75,$EP
DELETE $E,$EP
C(2)=CIRCLE/P(15),P(30),P(9),PGO,P(9),PEND,P(15)
B(1)=BSPLIN/3,P(9),P(26),P(7)
L(6)=LINE/P(10),P(7)
L(7)=LINE/P(7),P(8),FONT,'DASH'
$E=LINE/P(2),HORIZ,LENGTH,60
P(31)=POINT/INTOF,$E,L(3)
DELETE $E
L(8)=LINE/P(9),P(31),FONT,'DASH'
L(9)=LINE/P(15),P(13),FONT,'DASH'
<#
<#
L(10)=LINE/P(18),P(19)
C(3)=CIRCLE/RADIUS,HIP*2,LEFT,P(21),P(19),<#
>PGO,P(21),PEND,P(19)
C(4)=CIRCLE/RADIUS,HIP*4,RIGHT,P(21),P(23),<#
```

```
>PGO,P(21),PEND,P(23)
L(11)=LINE/P(23),P(22)
L(12)=LINE/P(24),P(22)
L(13)=LINE/P(25),P(24)
$E=LINE/P(20),P(25)
$EP=LINE/P(29),PERP,$E,LENGTH,1
P(32)=POINT/P(29),DISTAN,1,$EP
DELETE $E,$EP
C(5)=CIRCLE/P(25),P(32),P(20),PGO,P(20),PEND,P(25)
B(2)=BSPLIN/3,P(20),P(27),P(17)
L(14)=LINE/P(18),P(17)
$E=LINE/P(1),VERT,LENGTH,2
P(33)=POINT/INTOF,$E,L(10)
DELETE $E
L(15)=LINE/P(33),P(3),FONT,'DASH'
L(16)=LINE/P(3),P(4),FONT,'DASH'
L(17)=LINE/P(25),P(23),FONT,'DASH'
$E=LINE/P(7),HORIZ,LENGTH,20
P(34)=POINT/INTOF,$E,XSMALL,B(2)
DELETE $E
L(18)=LINE/P(34),P(21),FONT,'DASH'
$E=LINE/P(31),HORIZ,LENGTH,9
P(35)=POINT/INTOF,$E,XSMALL,C(4)
DELETE $E
L(19)=LINE/P(9),P(35),FONT,'DASH'
L(20)=LINE/x(1)-3,y(2)-10,0,x(1)-3,y(4)+15,0,FONT,'ARROW'
DELETE 35,P(1)
<#
<#
MIRROR 9,L,MX,x(21)+3,0,0
MIRROR 2,C,MX,x(21)+3,0,0
MIRROR B(1),MX,x(21)+3,0,0
MIRROR (COPY,cp(1)) L(20),MX,x(21)+3,0,0
<#
<#
x(30)=(x(21)+3)*2-x(1)
&T1='easy fitting trouser block (back)'
T(1)=TEXT/&T1,x(1)+1,y(5),0,TJST,'CJT',THGT,0.75,TANG,270
&T2='easy fitting trouser block (front)'
T(2)=TEXT/&T2,x(30)-1,y(5),0,TJST,'CJT',THGT,0.75,TANG,90
WHEN &data='1'
T(3)=TEXT/&name+' < SIZE '+&n+' >',x(1)+2,y(5),0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(4)=TEXT/&name+' < SIZE '+&n+' >',x(30)-2,y(5),0,<#
>TJST,'CJT',THGT,0.75,TANG,90
  OR &data='2'
IF (&name='') GOTO 200
T(3)=TEXT/&name,x(1)+2,y(5),0,TJST,'CJT',THGT,0.75,TANG,270
T(4)=TEXT/&name,x(30)-2,y(5),0,TJST,'CJT',THGT,0.75,TANG,90
#200 CONTINUE
ENDWHEN
<#
<#
!REGENERATE GRAPHICS
EXECDF
DISP ALL
#999 END
```

```
+-----------------------------------------------------------------+
|                                                                 |
|                  The culottes pattern (TRCU)                    |
|                                                                 |
+-----------------------------------------------------------------+
<#
<#
DECLARE REAL x(34),y(34)
DECLARE ENTITY P(34),L(22),B(2),C(7),T(6)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
&type='culotte'
CALLP TRSIZE(WAIST,BDRISE,HIP,WATOFL,WATOHI,TWID,<#
>JWID,WATOKN,&n,&data,&type,&unit)
<#
<#
PRINT {''}
PRINT    Select the position for the waist point of<#
> the trouser block please.
PRINT  ------------------------------------------<#
>--------------------------
PRINT     Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT     at the position you require. <#
>Press the left button.
PRINT  ------------------------------------<#
>----------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(5)=:x(3)=:X
y(1)=y(2)=:Y
x(2)=x(6)=:x(4)=:x(1)+HIP/2+1.5
y(3)=y(8)=:y(4)=:y(1)-WATOKN
y(5)=y(7)=:y(6)=:y(1)-WATOHI
x(7)=x(8)=:x(1)+HIP/4+1.5
x(9)=x(1)+WAIST/4+4.25
y(9)=y(10)=:y(1)+1.25
x(10)=x(2)-WAIST/4-2.25
x(11)=x(1)+(x(9)-x(1))/3
x(12)=x(1)+(x(9)-x(1))*2/3
y(11)=y(1)+1.25/3
y(12)=y(1)+1.25*2/3
x(13)=x(2)-(x(2)-x(10))*2/3
y(13)=y(2)+1.25*2/3
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>13
P(i)=POINT/x(i),y(i),0
#100 CONTINUE
```

```
<#
<#
$E=LINE/P(9),P(1)
$EP=LINE/P(11),PERP,$E,LENGTH,14
P(20)=POINT/P(11),DISTAN,14,$EP
P(14)=POINT/P(11),DISTAN,1,$E
P(15)=POINT/P(11),DISTAN,-1,$E
DELETE $EP
$EP=LINE/P(12),PERP,$E,LENGTH,12.5
P(21)=POINT/P(12),DISTAN,12.5,$EP
P(16)=POINT/P(12),DISTAN,1,$E
P(17)=POINT/P(12),DISTAN,-1,$E
DELETE $E,$EP
$E=LINE/P(2),P(10)
$EP=LINE/P(13),PERP,$E,LENGTH,10
P(22)=POINT/P(13),DISTAN,10,$EP
P(18)=POINT/P(13),DISTAN,1,$E
P(19)=POINT/P(13),DISTAN,-1,$E
DELETE $E,$EP
x(23)=x(27)=:x(1)
y(23)=y(24)=:y(25)=:y(29)=:y(28)=:y(30)=:y(1)-BDRISE-1.5
x(24)=x(29)=:x(7)
x(25)=x(26)=:x(1)-HIP/8-2
y(26)=y(31)=:y(3)
y(27)=(y(1)+y(23))/2+1
x(28)=x(32)=:x(2)
x(30)=x(31)=:x(2)+HIP/8-2
y(32)=(y(2)+y(28))/2
x(33)=x(23)-3/SQRT(2)
y(33)=y(23)+3/SQRT(2)
x(34)=x(28)+4/SQRT(2)
y(34)=y(28)+4/SQRT(2)
<#
<#
REPEAT 200: i=23,1,i>34
P(i)=POINT/x(i),y(i),0
#200 CONTINUE
<#
<#
L(1)=LINE/P(1),P(27)
B(1)=BSPLIN/3,P(27),P(33),P(25)
L(2)=LINE/P(25),P(26)
L(3)=LINE/P(26),P(8)
L(4)=LINE/P(7),P(8)
L(5)=LINE/P(1),P(3),FONT,'DASH'
L(6)=LINE/P(5),P(7),FONT,'DASH'
L(7)=LINE/P(14),P(20)
L(8)=LINE/P(15),P(20)
L(9)=LINE/P(16),P(21)
L(10)=LINE/P(17),P(21)
L(11)=LINE/P(24),P(25),FONT,'DASH'
L(12)=LINE/(x(5)+x(7))/2,y(5)+5,0,(x(5)+x(7))/2,<#
>y(3)+10,0,FONT,'ARROW'
C(1)=CIRCLE/RADIUS,HIP/3,LEFT,P(1),P(14),<#
>PGO,P(1),PEND,P(14)
C(2)=CIRCLE/RADIUS,HIP/3,LEFT,P(15),P(16),<#
>PGO,P(15),PEND,P(16)
C(3)=CIRCLE/RADIUS,HIP/3,LEFT,P(17),P(9),<#
```

```
>PGO,P(17),PEND,P(9)
C(4)=CIRCLE/RADIUS,HIP,LEFT,P(9),P(7),<#
>PGO,P(7),PEND,P(9)
<#
<#
L(13)=LINE/P(2),P(32)
B(2)=BSPLIN/3,P(32),P(34),P(30)
L(14)=LINE/P(30),P(31)
L(15)=LINE/P(31),P(8)
L(16)=LINE/P(7),P(8)
L(17)=LINE/P(19),P(22)
L(18)=LINE/P(18),P(22)
L(19)=LINE/P(2),P(4),FONT,'DASH'
L(20)=LINE/P(29),P(30),FONT,'DASH'
L(21)=LINE/P(7),P(6),FONT,'DASH'
L(22)=LINE/(x(7)+x(6))/2,y(5)+5,0,(x(7)+x(6))/2,<#
>y(3)+10,0,FONT,'ARROW'
C(5)=CIRCLE/RADIUS,HIP,RIGHT,P(19),P(2),<#
>PGO,P(19),PEND,P(2)
C(6)=CIRCLE/RADIUS,HIP/3,RIGHT,P(10),P(18),<#
>PGO,P(10),PEND,P(18)
C(7)=CIRCLE/RADIUS,HIP,RIGHT,P(10),P(7),<#
>PGO,P(10),PEND,P(7)
DELETE 34,P(1)
<#
<#
@V=VECT(6,0,0)
TRANSL 10,L(13),INDIRV,@V
TRANSL B(2),INDIRV,@V
TRANSL 3,C(5),INDIRV,@V
<#
<#
&T1='culottes block'
T(1)=TEXT/&T1,(x(3)+x(8))/2,y(3)+1,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(4)+x(8))/2+6,y(3)+1,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
T(3)=TEXT/&name+' < SIZE '+&n+' >',(x(3)+x(8))/2,y(3)+2,<#
>0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name+' < SIZE '+&n+' >',(x(4)+x(8))/2+6,y(3)+2,<#
>0,TJST,'CJT',THGT,0.75
  OR &data='2'
IF (&name='') GOTO 300
T(3)=TEXT/&name,(x(3)+x(8))/2,y(3)+2,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&name,(x(4)+x(8))/2+6,y(3)+2,0,<#
>TJST,'CJT',THGT,0.75
#300 CONTINUE
ENDWHEN
T(5)=TEXT/'centre back',x(1)+1,(y(5)+y(3))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(6)=TEXT/'centre front',x(4)+5,(y(5)+y(3))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
!REGENERATE GRAPHICS
EXECDF
DISP ALL
#999 END
```

DR1CLLK (Link file: DR1CL, DRSIZE)

```
MAIN DR1CL
IMAGE DR1CLIMAGE
LINKM DRSIZE
ENDLINK
```

DR1EALK (Link file: DR1EA, DRSIZE)

```
MAIN DR1EA
IMAGE DR1EAIMAGE
LINKM DRSIZE
ENDLINK
```

DR1DLLK (Link file: DR1DL, DRSIZE)

```
MAIN DR1DL
IMAGE DR1DLIMAGE
LINKM DRSIZE
ENDLINK
```

DR2CLLK (Link file: DR2CL, DRSIZE)

```
MAIN DR2CL
IMAGE DR2CLIMAGE
LINKM DRSIZE
ENDLINK
```

DR2EALK (Link file: DR2EA, DRSIZE)

```
MAIN DR2EA
IMAGE DR2EAIMAGE
LINKM DRSIZE
ENDLINK
```

DR2DLLK (Link file: DR2DL, DRSIZE)

```
MAIN DR2DL
IMAGE DR2DLIMAGE
LINKM DRSIZE
ENDLINK
```

```
+-----------------------------------------------------------+
|                                                           |
|              The dress size (DRSIZE) procedure            |
|                                                           |
+-----------------------------------------------------------+
<#
<#
PROC DRSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,ARMDPT<#
>,CHEST,NECK,WATOKN,&n,&data)
DECLARE REAL bust(12),shln(12),natowa(12),bwd(12),<#
>watohi(12),dart(12),armdpt(12),chest(12),neck(12),<#
>watokn(12)
DATA bust/80,84,88,92,97,102,107,112,117,122,127,132/
DATA shln/11.75,12,12.25,12.5,12.8,13.1,13.4,13.7,<#
>14,14.3,14.6,14.9/
DATA natowa/39,39.5,40,40.5,41,41.5,42,42.5,43,<#
>43.2,43.4,43.6/
DATA bwd/32.4,33.4,34.4,35.4,36.6,37.8,39,40.2,<#
>41.4,42.6,43.8,45/
DATA watohi/20,20.3,20.6,20.9,21.2,21.5,21.8,22.1,<#
>22.3,22.5,22.7,22.9/
DATA dart/5.8,6.4,7,7.6,8.2,8.8,9.4,10,10.6,11.2,<#
>11.8,12.4/
DATA armdpt/20,20.5,21,21.5,22,22.5,23,23.5,24.2,<#
>24.9,25.6,26.3/
DATA chest/30,31.2,32.4,33.6,35,36.5,38,39.5,41,<#
>42.5,44,45.5/
DATA neck/35,36,37,38,39.2,40.4,41.6,42.8,44,<#
>45.2,46.4,47.6/
DATA watokn/57.5,58,58.5,59,59.5,60,60.5,61,61.25,<#
>61.5,61.75,62/
ERTRAP 999
<#
<#
#10 CONTINUE
PRINT {''}
&data=''
PRINT       Which measurements do you want to use?
PRINT     ----------------------------------------
PRINT     The body measurements of this program --- 1
PRINT     Your individual body measurements ------- 2
PRINT     ----------------------------------------
READ(          Enter the number please. ---> )&data
WHEN &data='1'.OR.&data='2'
          CONTINUE
  ELSE
          PRINT Wrong answer, enter again please.
          GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
PRINT {''}
&unit=''
READ( Which units do you want to use?  <#
>  Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'.OR.&unit='IN'.OR.&unit='in'
          CONTINUE
```

```
  ELSE
        PRINT Wrong answer, enter again please.
        GOTO 20
ENDWHEN
<#
<#
#30 CONTINUE
PRINT {''}
&n=''
WHEN &unit='CM'.OR.&unit='cm'
        GOSUB 1000
  ELSE
        GOSUB 2000
ENDWHEN
PRINT   The size must be an even number between 8 and 30.
WHEN &data='1'
READ(        Enter the size please. ---> )&n
ELSE
READ(     Enter the nearest size please. ---> )&n
ENDWHEN
WHEN &n='8'.OR.&n='10'.OR.&n='12'.OR.&n='14'.OR.&n='16'<#
>.OR.&n='18'.OR.&n='20'.OR.&n='22'.OR.&n='24'.OR.&n='26'<#
>.OR.&n='28'.OR.&n='30'
        n=&n
        i=(n-6)/2
        BUST=bust(i)
        SHLN=shln(i)
        NATOWA=natowa(i)
        BWD=bwd(i)
        WATOHI=watohi(i)
        DART=dart(i)
        ARMDPT=armdpt(i)
        CHEST=chest(i)
        NECK=neck(i)
        WATOKN=watokn(i)
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 30
ENDWHEN
<#
<#
#40 CONTINUE
PRINT {''}
&height=''
WHEN &unit='CM'.OR.&unit='cm'
    PRINT                Which height do you want?
    PRINT    ----------------------------------------
    PRINT    The short women ( 152cm - 160cm ) --------- S
    PRINT    The medium women ( 160cm - 170cm ) -------- M
    PRINT    The tall women ( 170cm - 178cm ) ---------- T
    PRINT    ----------------------------------------
  ELSE
    PRINT                Which height do you want?
    PRINT    ----------------------------------------
    PRINT    The short women (5ft - 5ft 3in) ----------- S
    PRINT    The medium women (5ft 3in - 5ft 7in) ------ M
    PRINT    The tall women (5ft 7in - 5ft 10in) ------- T
    PRINT    ----------------------------------------
```

```
ENDWHEN
WHEN &data='1'
READ(  Enter the height please.  S or M or T ---> )&height
ELSE
READ(Enter the nearest height please. <#
> S or M or T ---> )&height
ENDWHEN
WHEN &height='S'.OR.&height='s'
         NATOWA=NATOWA-2
         ARMDPT=ARMDPT-0.8
         WATOKN=WATOKN-3
  OR &height='M'.OR.&height='m'
         CONTINUE
  OR &height='T'.OR.&height='t'
         NATOWA=NATOWA+2
         ARMDPT=ARMDPT+0.8
         WATOKN=WATOKN+3
  ELSE
         PRINT Wrong value entered, enter again please.
         GOTO 40
ENDWHEN
<#
<#
IF (&unit='CM'.OR.&unit='cm') GOTO 50
BUST=BUST/2.54
SHLN=SHLN/2.54
NATOWA=NATOWA/2.54
BWD=BWD/2.54
WATOHI=WATOHI/2.54
DART=DART/2.54
ARMDPT=ARMDPT/2.54
CHEST=CHEST/2.54
NECK=NECK/2.54
WATOKN=WATOKN/2.54
<#
<#
#50 CONTINUE
&BUST=BUST
&SHLN=SHLN
&NATOWA=NATOWA
&BWD=BWD
&WATOHI=WATOHI
&DART=DART
&ARMDPT=ARMDPT
&CHEST=CHEST
&NECK=NECK
&WATOKN=WATOKN
<#
<#
WHEN &data='1'
         PRINT {''}
         &change=''
PRINT       The data you entered is <#
>-----------------------------------
PRINT          size; {&n}          height; {&height}    <#
>          unit; {&unit}
PRINT          ------------------------------------<#
>-------------------------
```

```
PRINT          bust={&BUST(,4)}      shoulder length={&SHLN(,4)}<#
>      nape to waist={&NATOWA(,4)}
PRINT          back width={&BWD(,4)}      <#
>waist to hip={&WATOHI(,4)}      dart={&DART(,4)}
PRINT          armhole depth={&ARMDPT(,4)}        <#
>chest={&CHEST(,4)}       neck size={&NECK(,4)}
PRINT          waist to knee={&WATOKN(,4)}
PRINT       ------------------------------------<#
>-------------------------
READ(          Do you want to change?      Y/N ---> )&change
         WHEN &change='Y'.OR.&change='y'
                 GOTO 10
            OR &change='N'.OR.&change='n'
                 CONTINUE
           ELSE
            PRINT Wrong value entered, enter again please.
                 GOTO 50
         ENDWHEN
   OR &data='2'
          PRINT {''}
PRINT    ------------------------------------<#
>-------------------------------
WHEN &unit='CM'.OR.&unit='cm'
PRINT             Enter your individual measurements<#
> please.    (unit: cm)
ELSE
PRINT             Enter your individual measurements<#
> please.    (unit: in)
ENDWHEN
PRINT    If you don't want to change the measurement<#
> of the part, press RETURN.
PRINT    ------------------------------------<#
>-------------------------------
PRINT bust (size;{&n}, height;{&height}) = {&BUST(,4)}
READ(       your bust ---> )BUST
PRINT shoulder length (size;{&n}, height;{&height})<#
> = {&SHLN(,4)}
READ(       your shoulder length ---> )SHLN
PRINT nape to waist (size;{&n}, height;{&height})<#
> = {&NATOWA(,4)}
READ(       your nape to waist ---> )NATOWA
PRINT back width (size;{&n}, height;{&height}) = <#
>{&BWD(,4)}
READ(       your back width ---> )BWD
PRINT waist to hip (size;{&n}, height;{&height}) =<#
> {&WATOHI(,4)}
READ(       your waist to hip ---> )WATOHI
PRINT dart (size;{&n}, height;{&height}) = {&DART(,4)}
READ(       your dart ---> )DART
PRINT armhole depth (size;{&n}, height;{&height})<#
> = {&ARMDPT(,4)}
READ(       your armhole depth ---> )ARMDPT
PRINT chest (size;{&n}, height;{&height}) = {&CHEST(,4)}
READ(       your chest ---> )CHEST
PRINT neck size (size;{&n}, height;{&height}) = <#
>{&NECK(,4)}
READ(       your neck size ---> )NECK
PRINT waist to knee (size;{&n}, height;{&height})<#
```

```
> = {&WATOKN(,4)}
READ(      your waist to knee < or skirt length > <#
>---> )WATOKN
<#
<#
#60 CONTINUE
&BUST=BUST
&SHLN=SHLN
&NATOWA=NATOWA
&BWD=BWD
&WATOHI=WATOHI
&DART=DART
&ARMDPT=ARMDPT
&CHEST=CHEST
&NECK=NECK
&WATOKN=WATOKN
<#
<#
&change=''
PRINT {''}
PRINT       The data you entered is <#
>-----------------------------------
PRINT          nearest size; {&n}      <#
>nearest height; {&height}    unit; {&unit}
PRINT          ---------------------------------<#
>-----------------------------
PRINT          bust={&BUST(,4)}   <#
>shoulder length={&SHLN(,4)}   nape to waist={&NATOWA(,4)}
PRINT          back width={&BWD(,4)}   <#
>waist to hip={&WATOHI(,4)}   dart={&DART(,4)}
PRINT          armhole depth={&ARMDPT(,4)}      <#
>chest={&CHEST(,4)}    neck size={&NECK(,4)}
PRINT          waist to knee (or skirt length)={&WATOKN(,4)}
PRINT       ---------------------------------<#
>-----------------------------
READ(          Do you want to change?     Y/N ---> )&change
          WHEN &change='Y'.OR.&change='y'
                  GOTO 10
            OR &change='N'.OR.&change='n'
                  CONTINUE
            ELSE
              PRINT Wrong value entered, enter again please.
                  GOTO 60
          ENDWHEN
ENDWHEN
<#
<#
IF (&unit='CM'.OR.&unit='cm') GOTO 999
BUST=BUST*2.54
SHLN=SHLN*2.54
NATOWA=NATOWA*2.54
BWD=BWD*2.54
WATOHI=WATOHI*2.54
DART=DART*2.54
ARMDPT=ARMDPT*2.54
CHEST=CHEST*2.54
NECK=NECK*2.54
WATOKN=WATOKN*2.54
```

```
<#
<#
#999 RETURN
<#
<#
#1000 CONTINUE
PRINT                    WOMEN OF MEDIUM HEIGHT 160CM-170CM<#
>   (UNIT: CM)
PRINT     ----------------------------------------<#
>-------------------------
PRINT                    8    10    12    14    16    18    20  <#
> 22    24    26    28    30
PRINT     ----------------------------------------<#
>-------------------------
PRINT      BUST        80    84    88    92    97   102   107   <#
>112   117   122   127   132
PRINT      NA-TO-WA    39    39.5 40    40.5 41    41.5 42   <#
> 42.5 43    43.2 43.4 43.6
PRINT      B-WIDTH     32.4 33.4 34.4 35.4 36.6 37.8 39   <#
> 40.2 41.4 42.6 43.8 45
PRINT      CHEST       30    31.2 32.4 33.6 35    36.5 38   <#
> 39.5 41    42.5 44    45.5
PRINT      WA-TO-KN    57.5 58    58.5 59    59.5 60    60.5 <#
>61    61.3 61.5 61.8 62
PRINT     ----------------------------------------<#
>-------------------------
PRINT      NA-TO-WA: nape to waist    B-WIDTH: back width<#
>   WA-TO-KN: waist to knee
PRINT {''}
RTNSUB
<#
<#
#2000 CONTINUE
PRINT                     WOMEN OF MEDIUM HEIGHT 5FT 3IN -<#
> 5FT 7IN (UNIT: IN)
PRINT     ----------------------------------------<#
>-------------------------
PRINT                     8    10    12    14    16    18    20    <#
>22    24    26    28    30
PRINT     ----------------------------------------<#
>-------------------------
PRINT      BUST        31.5 33.1 34.6 36.2 38.2 40.2 42.2<#
> 44.1 46.1 48    50    52
PRINT      NA-TO-WA    15.4 15.6 15.7 15.9 16.1 16.3 16.5<#
> 16.7 16.9 17    17.1 17.2
PRINT      B-WIDTH     12.8 13.1 13.5 13.9 14.4 14.9 15.4 <#
>15.8 16.3 16.8 17.2 17.7
PRINT      CHEST       11.8 12.3 12.8 13.2 13.8 14.4 15   <#
> 15.6 16.1 16.7 17.3 17.9
PRINT      WA-TO-KN    22.6 22.8 23    23.2 23.4 23.6 23.8 <#
>24    24.1 24.2 24.3 24.4
PRINT     ----------------------------------------<#
>-------------------------
PRINT      NA-TO-WA: nape to waist    B-WIDTH: back width<#
>   WA-TO-KN: waist to knee
PRINT {''}
RTNSUB
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|              The close fitting one-piece dress (DR1CL)                |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE REAL x(41),y(41)
DECLARE LOCATION P(41)
DECLARE ENTITY p(5),PNT(6),L(34),B(3),C(5),T(8)
ERTRAP 999
<#
<#
#10 CONTINUE
PRINT {''}
&waist=''
PRINT                Which waist shaping do you want?
PRINT     ----------------------------------------
PRINT     The close fitting waist ----------------- 1
PRINT     The loose fitting waist ----------------- 2
PRINT     ----------------------------------------
READ(              Enter the number please. ---> )&waist
WHEN &waist='1'.OR.&waist='2'
          CONTINUE
  ELSE
          PRINT Wrong answer, enter again please.
          GOTO 10
ENDWHEN
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP DRSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,ARMDPT<#
>,CHEST,NECK,WATOKN,&n,&data)
<#
<#
PRINT {''}
PRINT   Select the position for the centre back neck<#
> of the dress block please.
PRINT  ----------------------------------------<#
>--------------------------
PRINT     Move the mouse on the pad to place the cursor<#
> on the graphic window
PRINT     at the position you require.<#
> Press the left button.
PRINT  ----------------------------------------<#
>--------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(2)=:x(4)=:x(6)=:x(40)=:X
y(1)=y(30)=:Y
y(2)=y(14)=:y(12)=:y(27)=:y(19)=:y(20)=:y(3)=:<#
>y(1)-ARMDPT-0.5
```

```
x(3)=x(18)=:x(5)=:x(39)=:x(7)=:x(41)=:x(1)+BUST/2+5
y(4)=y(15)=:y(28)=:y(21)=:y(5)=:y(1)-NATOWA
y(6)=y(16)=:y(29)=:y(22)=:y(7)=:y(4)-WATOHI
x(8)=x(1)+NECK/5-0.2
y(8)=y(1)+1.5
y(9)=y(1)-ARMDPT/5+0.7
x(9)=x(8)+SQRT((SHLN+1)**2-(y(8)-y(9))**2)
x(10)=(x(8)+x(9))/2
y(10)=(y(8)+y(9))/2
x(11)=x(10)-1
y(11)=y(10)-5
x(12)=x(13)=:x(2)+BWD/2+0.5
y(13)=(y(9)+y(12))/2
x(14)=x(15)=:x(16)=:(x(2)+x(12))/2
x(17)=x(3)-NECK/5+0.7
WHEN BUST<95
        y(17)=y(24)=:y(1)+1.5
  OR BUST.GE.95.AND.BUST<100
        y(17)=y(24)=:y(1)+1.8
  OR BUST.GE.100.AND.BUST<105
        y(17)=y(24)=:y(1)+2.1
  OR BUST.GE.105.AND.BUST<110
        y(17)=y(24)=:y(1)+2.4
  OR BUST.GE.110.AND.BUST<115
        y(17)=y(24)=:y(1)+2.7
  OR BUST.GE.115.AND.BUST<120
        y(17)=y(24)=:y(1)+3
  OR BUST.GE.120.AND.BUST<125
        y(17)=y(24)=:y(1)+3.3
  OR BUST.GE.125.AND.BUST<130
        y(17)=y(24)=:y(1)+3.6
  ELSE
        y(17)=y(24)=:y(1)+3.9
ENDWHEN
y(18)=y(17)-NECK/5+0.2
x(19)=x(26)=:x(3)-(CHEST+DART)/2
x(20)=x(23)=:x(21)=:x(22)=:(x(3)+x(19))/2
y(23)=y(3)-2.5
x(24)=x(17)-DART
y(25)=y(9)-1.5
x(25)=x(24)-SQRT(SHLN**2-(y(24)-y(25))**2)
y(26)=y(3)+(y(18)-y(3))/3
x(27)=x(28)=:x(29)=:(x(12)+x(19))/2
x(30)=x(1)+(x(8)-x(1))/3
x(31)=x(8)-(x(8)-x(1))/4
y(31)=y(1)+(y(8)-y(1))/3
WHEN BUST<95
        backcurve=2.5
        frontcurve=2
  OR BUST.GE.95.AND.BUST.LT.115
        backcurve=3
        frontcurve=2.5
  ELSE
        backcurve=3.5
        frontcurve=3
ENDWHEN
x(32)=x(12)+backcurve/SQRT(2)
y(32)=y(12)+backcurve/SQRT(2)
```

```
x(33)=x(19)-frontcurve/SQRT(2)
y(33)=y(19)+frontcurve/SQRT(2)
x(34)=x(14)-1.75
y(34)=y(35)=:y(36)=:y(37)=:y(38)=:y(39)=:y(4)
x(35)=x(14)+1.75
x(36)=x(27)-1.5
x(37)=x(27)+2.5
x(38)=x(23)-2.25
x(39)=x(23)+2.25
y(40)=y(41)=:y(4)-WATOKN
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>41
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
$E=LINE/P(8),P(10)
PNT(1)=POINT/P(10),DISTAN,-0.5,$E
DELETE $E
$E=LINE/P(9),P(10)
PNT(2)=POINT/P(10),DISTAN,-0.5,$E
DELETE $E
WHEN &waist='2'
          PNT(3)=POINT/x(29),y(40),0
  OR &waist='1'
          PNT(3)=POINT/(x(40)+x(41))/2,y(40),0
          PNT(4)=POINT/(x(40)+x(41))/2,y(6),0
          PNT(5)=POINT/x(15),y(4)-13,0
          PNT(6)=POINT/x(21),y(4)-13,0
ENDWHEN
<#
<#
p(1)=POINT/x(13),y(13),0
p(2)=POINT/x(32),y(32),0
p(3)=POINT/x(26),y(26),0
p(4)=POINT/x(33),y(33),0
<#
<#
L(1)=LINE/P(1),P(40)
L(2)=LINE/P(4),P(28),FONT,'DASH'
L(3)=LINE/P(2),P(27),FONT,'DASH'
L(4)=LINE/P(40),PNT(3)
L(5)=LINE/P(8),PNT(1)
L(6)=LINE/PNT(1),P(11)
L(7)=LINE/PNT(2),P(11)
L(8)=LINE/PNT(2),P(9)
L(9)=LINE/P(18),P(41)
L(10)=LINE/P(41),PNT(3)
L(11)=LINE/P(5),P(28),FONT,'DASH'
L(12)=LINE/P(3),P(27),FONT,'DASH'
L(13)=LINE/P(17),P(23)
L(14)=LINE/P(24),P(23)
L(15)=LINE/P(24),P(25)
L(16)=LINE/P(1),P(30)
L(17)=LINE/P(14),P(34)
L(18)=LINE/P(14),P(35)
```

```
L(19)=LINE/P(27),P(36)
L(20)=LINE/P(27),P(37)
L(21)=LINE/P(23),P(38)
L(22)=LINE/P(23),P(39)
L(23)=LINE/P(14),P(16),FONT,'DASH'
L(24)=LINE/P(20),P(22),FONT,'DASH'
L(25)=LINE/x(1)+5,y(2)-5,0,x(1)+5,y(6)-15,0,FONT,'ARROW'
L(26)=LINE/x(3)-5,y(2)-5,0,x(3)-5,y(6)-15,0,FONT,'ARROW'
B(1)=BSPLIN/3,P(27),P(32),P(13),P(9)        <# b-armhole
$L1=LINE/P(25),P(26)
$L2=LINE/(x(25)+x(26))/2,(y(25)+y(26))/2,0,<#
>PERP,$L1,LENGTH,0.5
p(5)=POINT/(x(25)+x(26))/2,(y(25)+y(26))/2,0,DISTAN,0.5,$L2
DELETE $L1,$L2
B(2)=BSPLIN/3,P(27),P(33),P(26),p(5),P(25)    <# f-armhole
B(3)=BSPLIN/3,P(30),P(31),P(8)                <# b-neck
C(1)=CIRCLE/CENTER,P(13),RADIUS,0.4
C(2)=CIRCLE/CENTER,P(26),RADIUS,0.4
C(3)=CIRCLE/RADIUS,NECK/5-0.2,RIGHT,P(17),P(18),<#
>PGO,P(17),PEND,P(18)
WHEN &waist='2'
        L(27)=LINE/P(4),P(34)
        L(28)=LINE/P(35),P(36)
        L(29)=LINE/P(37),P(38)
        L(30)=LINE/P(39),P(5)
        L(31)=LINE/P(27),PNT(3)
        L(32)=LINE/P(6),P(29),FONT,'DASH'
        L(33)=LINE/P(7),P(29),FONT,'DASH'
        DELETE 3,PNT(1)
  OR &waist='1'
        L(27)=LINE/P(34),PNT(5)
        L(28)=LINE/P(35),PNT(5)
        L(29)=LINE/P(38),PNT(6)
        L(30)=LINE/P(39),PNT(6)
        L(31)=LINE/P(27),P(29),FONT,'DASH'
        L(32)=LINE/PNT(4),PNT(3)
        L(33)=LINE/P(6),PNT(4),FONT,'DASH'
        L(34)=LINE/P(7),PNT(4),FONT,'DASH'
        C(4)=CIRCLE/RADIUS,BUST*1.5,LEFT,P(36),<#
>PNT(4),PGO,PNT(4),PEND,P(36)
        C(5)=CIRCLE/RADIUS,BUST*1.5,RIGHT,P(37),<#
>PNT(4),PGO,P(37),PEND,PNT(4)
        DELETE 6,PNT(1)
ENDWHEN
<#
<#
&T1='one-piece dress block'
T(1)=TEXT/&T1,(x(6)+x(29))/2,y(40)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(7)+x(29))/2,y(40)+2,0,TJST,'CJT',THGT,0.75
&T2='close fitting bodice'
T(3)=TEXT/&T2,(x(6)+x(29))/2,y(40)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&T2,(x(7)+x(29))/2,y(40)+3,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
        T(5)=TEXT/&name+' < SIZE '+&n+' >',(x(6)+x(29))/2<#
>,y(40)+4,0,TJST,'CJT',THGT,0.75
        T(6)=TEXT/&name+' < SIZE '+&n+' >',(x(7)+x(29))/2<#
>,y(40)+4,0,TJST,'CJT',THGT,0.75
  OR &data='2'
```

```
        IF (&name='') GOTO 200
T(5)=TEXT/&name,(x(6)+x(29))/2,y(40)+4,0,TJST,'CJT',THGT,0.75
T(6)=TEXT/&name,(x(7)+x(29))/2,y(40)+4,0,TJST,'CJT',THGT,0.75
        #200 CONTINUE
ENDWHEN
T(7)=TEXT/'centre back line',x(1)+1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(8)=TEXT/'centre front line',x(3)-1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
DISP ALL
#999 END
```

```
+-----------------------------------------------------------+
|                                                           |
|          The easy fitting one-piece dress (DR1EA)         |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL x(27),y(27)
DECLARE LOCATION P(27)
DECLARE ENTITY p(4),PNT(4),L(21),B(3),C(5),T(8)
ERTRAP 999
<#
<#
#10 CONTINUE
PRINT {''}
&waist=''
PRINT              Which waist shaping do you want?
PRINT     ----------------------------------------
PRINT     The close fitting waist ----------------- 1
PRINT     The loose fitting waist ----------------- 2
PRINT     ----------------------------------------
READ(            Enter the number please. ---> )&waist
WHEN &waist='1'.OR.&waist='2'
        CONTINUE
  ELSE
        PRINT Wrong answer, enter again please.
        GOTO 10
ENDWHEN
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP DRSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,ARMDPT<#
>,CHEST,NECK,WATOKN,&n,&data)
<#
<#
PRINT {''}
PRINT   Select the position for the centre back neck<#
> of the dress block please.
PRINT  ----------------------------------------<#
>--------------------------
PRINT     Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT     at the position you require.<#
> Press the left button.
PRINT  ----------------------------------<#
>----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(2)=:x(4)=:x(6)=:x(26)=:X
y(1)=y(22)=:Y
y(2)=y(10)=:y(19)=:y(15)=:y(17)=:y(3)=:y(1)-ARMDPT-2.5
x(3)=x(13)=:x(5)=:x(7)=:x(27)=:x(1)+BUST/2+7
```

```
y(4)=y(20)=:y(5)=:y(1)-NATOWA
y(6)=y(21)=:y(7)=:y(4)-WATOHI
x(8)=x(1)+NECK/5-0.2
y(8)=y(1)+1.5
y(9)=y(1)-ARMDPT/5+1
x(9)=x(8)+SQRT((SHLN+1)**2-(y(8)-y(9))**2)
x(10)=x(11)=:x(1)+BWD/2+1
y(11)=(y(10)+y(9))/2
x(12)=x(3)-NECK/5+0.7
WHEN BUST<95
          y(12)=y(14)=:y(1)+1.5
  OR BUST.GE.95.AND.BUST<100
          y(12)=y(14)=:y(1)+1.8
  OR BUST.GE.100.AND.BUST<105
          y(12)=y(14)=:y(1)+2.1
  OR BUST.GE.105.AND.BUST<110
          y(12)=y(14)=:y(1)+2.4
  OR BUST.GE.110.AND.BUST<115
          y(12)=y(14)=:y(1)+2.7
  OR BUST.GE.115.AND.BUST<120
          y(12)=y(14)=:y(1)+3
  OR BUST.GE.120.AND.BUST<125
          y(12)=y(14)=:y(1)+3.3
  OR BUST.GE.125.AND.BUST<130
          y(12)=y(14)=:y(1)+3.6
  ELSE
          y(12)=y(14)=:y(1)+3.9
ENDWHEN
y(13)=y(12)-NECK/5+0.2
x(14)=x(12)-DART/2
x(15)=x(16)=:x(3)-(CHEST/2+1+DART/4)
y(16)=(y(3)+y(13))/2
x(17)=(x(3)+x(15))/2
y(18)=y(9)-1.5
x(18)=x(14)-SQRT((SHLN+0.5)**2-(y(14)-y(18))**2)
x(19)=x(20)=:x(21)=:(x(10)+x(15))/2
x(22)=x(1)+(x(8)-x(1))/3
WHEN BUST<95
          backcurve=2.5
          frontcurve=2.25
  OR BUST.GE.95.AND.BUST.LT.115
          backcurve=3
          frontcurve=2.75
  ELSE
          backcurve=3.5
          frontcurve=3.25
ENDWHEN
x(23)=x(10)+backcurve/SQRT(2)
y(23)=y(10)+backcurve/SQRT(2)
x(24)=x(15)-frontcurve/SQRT(2)
y(24)=y(15)+frontcurve/SQRT(2)
x(25)=x(8)-(x(8)-x(1))/4
y(25)=y(1)+(y(8)-y(1))/3
y(26)=y(27)=:y(4)-WATOKN
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>27
```

```
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
WHEN &waist='2'
          PNT(1)=POINT/x(21),y(26),0
  OR &waist='1'
          PNT(1)=POINT/(x(26)+x(27))/2,y(26),0
          PNT(2)=POINT/(x(26)+x(27))/2,y(6),0
          PNT(3)=POINT/x(21)-1.5,y(4),0
          PNT(4)=POINT/x(21)+2.5,y(4),0
ENDWHEN
<#
<#
p(1)=POINT/x(11),y(11),0
p(2)=POINT/x(23),y(23),0
p(3)=POINT/x(16),y(16),0
p(4)=POINT/x(24),y(24),0
<#
<#
L(1)=LINE/P(1),P(26)
L(2)=LINE/P(26),PNT(1)
L(3)=LINE/P(8),P(9)
L(4)=LINE/P(2),P(19),FONT,'DASH'
L(5)=LINE/P(4),P(20),FONT,'DASH'
L(6)=LINE/P(13),P(27)
L(7)=LINE/P(27),PNT(1)
L(8)=LINE/P(12),P(17)
L(9)=LINE/P(14),P(17)
L(10)=LINE/P(14),P(18)
L(11)=LINE/P(3),P(19),FONT,'DASH'
L(12)=LINE/P(5),P(20),FONT,'DASH'
L(13)=LINE/P(1),P(22)
L(14)=LINE/x(1)+5,y(2)-5,0,x(1)+5,y(6)-15,0,FONT,'ARROW'
L(15)=LINE/x(3)-5,y(2)-5,0,x(3)-5,y(6)-15,0,FONT,'ARROW'
B(1)=BSPLIN/3,P(19),P(23),P(11),P(9)          <# b-armhole
B(2)=BSPLIN/3,P(19),P(24),P(16),P(18)         <# f-armhole
B(3)=BSPLIN/3,P(22),P(25),P(8)                <# b-neck
C(1)=CIRCLE/CENTER,P(11),RADIUS,0.4
C(2)=CIRCLE/CENTER,P(16),RADIUS,0.4
C(3)=CIRCLE/RADIUS,NECK/5-0.2,RIGHT,P(12),P(13),<#
>PGO,P(12),PEND,P(13)
<#
<#
WHEN &waist='2'
          L(16)=LINE/P(19),PNT(1)
          L(17)=LINE/P(6),P(21),FONT,'DASH'
          L(18)=LINE/P(7),P(21),FONT,'DASH'
          DELETE PNT(1)
  OR &waist='1'
          L(16)=LINE/P(19),PNT(3)
          L(17)=LINE/P(19),PNT(4)
          L(18)=LINE/PNT(2),PNT(1)
          L(19)=LINE/P(19),P(21),FONT,'DASH'
          L(20)=LINE/P(6),PNT(2),FONT,'DASH'
          L(21)=LINE/P(7),PNT(2),FONT,'DASH'
          C(4)=CIRCLE/RADIUS,BUST*1.5,LEFT,PNT(3),PNT(2),<#
>PGO,PNT(2),PEND,PNT(3)
```

```
        C(5)=CIRCLE/RADIUS,BUST*1.5,RIGHT,PNT(4),PNT(2),<#
>PGO,PNT(4),PEND,PNT(2)
        DELETE 4,PNT(1)
ENDWHEN
<#
<#
&T1='one-piece dress block'
T(1)=TEXT/&T1,(x(6)+x(21))/2,y(26)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(7)+x(21))/2,y(26)+2,0,TJST,'CJT',THGT,0.75
&T2='easy fitting bodice'
T(3)=TEXT/&T2,(x(6)+x(21))/2,y(26)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&T2,(x(7)+x(21))/2,y(26)+3,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
T(5)=TEXT/&name+' < SIZE '+&n+' >',(x(6)+x(21))/2,<#
>y(26)+4,0,TJST,'CJT',THGT,0.75
T(6)=TEXT/&name+' < SIZE '+&n+' >',(x(7)+x(21))/2,<#
>y(26)+4,0,TJST,'CJT',THGT,0.75
  OR &data='2'
        IF (&name='') GOTO 200
T(5)=TEXT/&name,(x(6)+x(21))/2,y(26)+4,0,<#
>TJST,'CJT',THGT,0.75
T(6)=TEXT/&name,(x(7)+x(21))/2,y(26)+4,0,<#
>TJST,'CJT',THGT,0.75
        #200 CONTINUE
ENDWHEN
T(7)=TEXT/'centre back line',x(1)+1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(8)=TEXT/'centre front line',x(3)-1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
DISP ALL
#999 END
```

```
+-----------------------------------------------------------+
|                                                           |
|     The dartless easy fitting one-piece dress (DR1DL)     |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL x(25),y(25)
DECLARE LOCATION P(25)
DECLARE ENTITY p(4),PNT(4),L(19),B(3),C(5),T(8)
ERTRAP 999
<#
<#
#10 CONTINUE
PRINT {''}
&waist=''
PRINT            Which waist shaping do you want?
PRINT     ----------------------------------------
PRINT     The close fitting waist ---------------- 1
PRINT     The loose fitting waist ---------------- 2
PRINT     ----------------------------------------
READ(          Enter the number please. ---> )&waist
WHEN &waist='1'.OR.&waist='2'
        CONTINUE
  ELSE
        PRINT Wrong answer, enter again please.
        GOTO 10
ENDWHEN
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP DRSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,ARMDPT,<#
>CHEST,NECK,WATOKN,&n,&data)
<#
<#
PRINT {''}
PRINT   Select the position for the centre back neck<#
> of the dress block please.
PRINT  ----------------------------------------<#
>----------------------------
PRINT     Move the mouse on the pad to place the <#
>cursor on the graphic window
PRINT     at the position you require. <#
>Press the left button.
PRINT  ----------------------------------------<#
>----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(2)=:x(4)=:x(6)=:x(24)=:X
y(1)=y(20)=:Y
y(2)=y(10)=:y(17)=:y(14)=:y(3)=:y(1)-ARMDPT-2.5
x(3)=x(13)=:x(5)=:x(7)=:x(25)=:x(1)+BUST/2+7
```

```
y(4)=y(18)=:y(5)=:y(1)-NATOWA
y(6)=y(19)=:y(7)=:y(4)-WATOHI
x(8)=x(1)+NECK/5-0.2
y(8)=y(1)+1.5
y(9)=y(1)-ARMDPT/5+1
x(9)=x(8)+SQRT((SHLN+1)**2-(y(8)-y(9))**2)
x(10)=x(11)=:x(1)+BWD/2+1
y(11)=(y(10)+y(9))/2
x(12)=x(3)-NECK/5+0.7
WHEN BUST<95
          y(12)=y(1)+1.5
  OR BUST.GE.95.AND.BUST<100
          y(12)=y(1)+1.8
  OR BUST.GE.100.AND.BUST<105
          y(12)=y(1)+2.1
  OR BUST.GE.105.AND.BUST<110
          y(12)=y(1)+2.4
  OR BUST.GE.110.AND.BUST<115
          y(12)=y(1)+2.7
  OR BUST.GE.110.AND.BUST<120
          y(12)=y(1)+3
  OR BUST.GE.120.AND.BUST<125
          y(12)=y(1)+3.3
  OR BUST.GE.125.AND.BUST<130
          y(12)=y(1)+3.6
  ELSE
          y(12)=y(1)+3.9
ENDWHEN
y(13)=y(12)-NECK/5+0.2
x(14)=x(15)=:x(3)-(CHEST/2+1.5)
y(15)=(y(3)+y(13))/2
y(16)=y(9)-0.75
x(16)=x(12)-SQRT((SHLN+0.5)**2-(y(12)-y(16))**2)
x(17)=x(18)=:x(19)=:(x(10)+x(14))/2
x(20)=x(1)+(x(8)-x(1))/3
WHEN BUST<95
          backcurve=2.5
          frontcurve=2.25
  OR BUST.GE.95.AND.BUST.LT.115
          backcurve=3
          frontcurve=2.75
  ELSE
          backcurve=3.5
          frontcurve=3.25
ENDWHEN
x(21)=x(10)+backcurve/SQRT(2)
y(21)=y(10)+backcurve/SQRT(2)
x(22)=x(14)-frontcurve/SQRT(2)
y(22)=y(14)+frontcurve/SQRT(2)
x(23)=x(8)-(x(8)-x(1))/4
y(23)=y(1)+(y(8)-y(1))/3
y(24)=y(25)=:y(4)-WATOKN
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>25
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
```

```
<#
<#
WHEN &waist='2'
          PNT(1)=POINT/x(19),y(24),0
  OR &waist='1'
          PNT(1)=POINT/(x(24)+x(25))/2,y(24),0
          PNT(2)=POINT/(x(24)+x(25))/2,y(6),0
          PNT(3)=POINT/x(17)-1.5,y(4),0
          PNT(4)=POINT/x(17)+2.5,y(4),0
ENDWHEN
<#
<#
p(1)=POINT/x(11),y(11),0
p(2)=POINT/x(21),y(21),0
p(3)=POINT/x(15),y(15),0
p(4)=POINT/x(22),y(22),0
<#
<#
L(1)=LINE/P(1),P(24)
L(2)=LINE/P(24),PNT(1)
L(3)=LINE/P(8),P(9)
L(4)=LINE/P(2),P(17),FONT,'DASH'
L(5)=LINE/P(4),P(18),FONT,'DASH'
L(6)=LINE/P(13),P(25)
L(7)=LINE/P(25),PNT(1)
L(8)=LINE/P(12),P(16)
L(9)=LINE/P(3),P(17),FONT,'DASH'
L(10)=LINE/P(5),P(18),FONT,'DASH'
L(11)=LINE/P(1),P(20)
L(12)=LINE/x(1)+5,y(2)-5,0,x(1)+5,y(6)-15,0,FONT,'ARROW'
L(13)=LINE/x(3)-5,y(2)-5,0,x(3)-5,y(6)-15,0,FONT,'ARROW'
B(1)=BSPLIN/3,P(17),P(21),P(11),P(9)          <# b-armhole
B(2)=BSPLIN/3,P(17),P(22),P(15),P(16)         <# f-armhole
B(3)=BSPLIN/3,P(20),P(23),P(8)                <# b-neck
C(1)=CIRCLE/CENTER,P(11),RADIUS,0.4
C(2)=CIRCLE/CENTER,P(15),RADIUS,0.4
C(3)=CIRCLE/RADIUS,NECK/5-0.2,RIGHT,P(12),P(13),<#
>PGO,P(12),PEND,P(13)
<#
<#
WHEN &waist='2'
          L(14)=LINE/P(17),PNT(1)
          L(15)=LINE/P(6),P(19),FONT,'DASH'
          L(16)=LINE/P(7),P(19),FONT,'DASH'
          DELETE PNT(1)
  OR &waist='1'
          L(14)=LINE/P(17),PNT(3)
          L(15)=LINE/P(17),PNT(4)
          L(16)=LINE/PNT(2),PNT(1)
          L(17)=LINE/P(17),P(19),FONT,'DASH'
          L(18)=LINE/P(6),PNT(2),FONT,'DASH'
          L(19)=LINE/P(7),PNT(2),FONT,'DASH'
          C(4)=CIRCLE/RADIUS,BUST*1.5,LEFT,PNT(3),PNT(2),<#
>PGO,PNT(2),PEND,PNT(3)
          C(5)=CIRCLE/RADIUS,BUST*1.5,RIGHT,PNT(4),PNT(2)<#
>,PGO,PNT(4),PEND,PNT(2)
          DELETE 4,PNT(1)
ENDWHEN
```

```
<#
<#
&T1='one-piece dress block'
T(1)=TEXT/&T1,(x(6)+x(19))/2,y(24)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(7)+x(19))/2,y(24)+2,0,TJST,'CJT',THGT,0.75
&T2='dartless bodice'
T(3)=TEXT/&T2,(x(6)+x(19))/2,y(24)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&T2,(x(7)+x(19))/2,y(24)+3,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
T(5)=TEXT/&name+' < SIZE '+&n+' >',(x(6)+x(19))/2,<#
>y(24)+4,0,TJST,'CJT',THGT,0.75
T(6)=TEXT/&name+' < SIZE '+&n+' >',(x(7)+x(19))/2,<#
>y(24)+4,0,TJST,'CJT',THGT,0.75
  OR &data='2'
        IF (&name='') GOTO 200
T(5)=TEXT/&name,(x(6)+x(19))/2,y(24)+4,0,TJST,'CJT',THGT,0.75
T(6)=TEXT/&name,(x(7)+x(19))/2,y(24)+4,0,TJST,'CJT',THGT,0.75
        #200 CONTINUE
ENDWHEN
T(7)=TEXT/'centre back line',x(1)+1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(8)=TEXT/'centre front line',x(3)-1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
DISP ALL
#999 END
```

```
+-----------------------------------------------------------------+
|                                                                 |
|          The close fitting two-piece dress (DR2CL)              |
|                                                                 |
+-----------------------------------------------------------------+
<#
<#
DECLARE REAL x(43),y(43)
DECLARE LOCATION P(43)
DECLARE ENTITY p(5),PNT(16),L(41),B(3),C(10),T(8)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP DRSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,ARMDPT<#
>,CHEST,NECK,WATOKN,&n,&data)
<#
<#
PRINT {''}
PRINT   Select the position for the centre back<#
> neck of the dress block please.
PRINT  ----------------------------------------<#
>--------------------------
PRINT     Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT     at the position you require.<#
> Press the left button.
PRINT  ----------------------------------------<#
>--------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(2)=:x(4)=:x(36)=:x(40)=:x(38)=:X
y(1)=y(25)=:Y
y(2)=y(12)=:y(10)=:y(23)=:y(16)=:y(17)=:y(3)=:<#
>y(1)-ARMDPT-0.5
x(3)=x(15)=:x(5)=:x(35)=:x(37)=:x(41)=:x(39)=:<#
>x(1)+BUST/2+5
y(4)=y(13)=:y(24)=:y(18)=:y(5)=:y(1)-NATOWA
x(6)=x(1)+NECK/5-0.2
y(6)=y(1)+1.5
y(7)=y(1)-ARMDPT/5+0.7
x(7)=x(6)+SQRT((SHLN+1)**2-(y(6)-y(7))**2)
x(8)=(x(6)+x(7))/2
y(8)=(y(6)+y(7))/2
x(9)=x(8)-1
y(9)=y(8)-5
x(10)=x(11)=:x(2)+BWD/2+0.5
y(11)=(y(7)+y(10))/2
x(12)=x(13)=:(x(2)+x(10))/2
x(14)=x(3)-NECK/5+0.7
WHEN BUST<95
        y(20)=y(14)=:y(1)+1.5
```

```
OR BUST.GE.95.AND.BUST<100
      y(20)=y(14)=:y(1)+1.8
OR BUST.GE.100.AND.BUST<105
      y(20)=y(14)=:y(1)+2.1
OR BUST.GE.105.AND.BUST<110
      y(20)=y(14)=:y(1)+2.4
OR BUST.GE.110.AND.BUST<115
      y(20)=y(14)=:y(1)+2.7
OR BUST.GE.115.AND.BUST<120
      y(20)=y(14)=:y(1)+3
OR BUST.GE.120.AND.BUST<125
      y(20)=y(14)=:y(1)+3.3
  OR BUST.GE.125.AND.BUST<130
        y(20)=y(14)=:y(1)+3.6
  ELSE
        y(20)=y(14)=:y(1)+3.9
ENDWHEN
y(15)=y(14)-NECK/5+0.2
x(16)=x(22)=:x(3)-(CHEST+DART)/2
x(17)=x(19)=:x(18)=:(x(3)+x(16))/2
y(19)=y(17)-2.5
x(20)=x(14)-DART
y(21)=y(7)-1.5
x(21)=x(20)-SQRT(SHLN**2-(y(20)-y(21))**2)
y(22)=y(3)+(y(15)-y(3))/3
x(23)=x(24)=:(x(10)+x(16))/2
x(25)=x(1)+(x(6)-x(1))/3
x(26)=x(6)-(x(6)-x(1))/4
y(26)=y(1)+(y(6)-y(1))/3
WHEN BUST<95
        backcurve=2.5
        frontcurve=2
  OR BUST.GE.95.AND.BUST.LT.115
        backcurve=3
        frontcurve=2.5
  ELSE
        backcurve=3.5
        frontcurve=3
ENDWHEN
x(27)=x(10)+backcurve/SQRT(2)
y(27)=y(10)+backcurve/SQRT(2)
x(28)=x(16)-frontcurve/SQRT(2)
y(28)=y(16)+frontcurve/SQRT(2)
x(29)=x(12)-1.75
y(29)=y(30)=:y(4)-0.25
x(30)=x(12)+1.75
x(31)=x(23)-1.5
y(31)=y(32)=:y(4)-0.5
x(32)=x(23)+2.5
x(33)=x(18)-2.25
y(33)=y(34)=:y(4)-0.75
x(34)=x(18)+2.25
y(35)=y(5)-1
y(36)=y(37)=:y(4)-5
y(38)=y(39)=:y(42)=:y(36)-WATOKN
y(40)=y(41)=:y(43)=:y(36)-WATOHI
x(42)=x(43)=:(x(38)+x(39))/2
<#
```

```
<#
DISP OFF
REPEAT 100: i=1,1,i>43
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
$E=LINE/P(6),P(8)
PNT(1)=POINT/P(8),DISTAN,-0.5,$E
DELETE $E
$E=LINE/P(7),P(8)
PNT(2)=POINT/P(8),DISTAN,-0.5,$E
DELETE $E
PNT(3)=POINT/x(31),y(36)+1.25,0
PNT(4)=POINT/x(32)+2.5,y(36)+1.25,0
$E=LINE/P(36),PNT(3)
PNT(5)=POINT/P(36),DISTAN,(x(29)-x(4))*2/3,$E
PNT(6)=POINT/PNT(5),DISTAN,1.75,$E
PNT(7)=POINT/PNT(6),DISTAN,(x(29)-x(4))*2/3,$E
PNT(8)=POINT/PNT(7),DISTAN,1.75,$E
PNT(9)=POINT/PNT(5),DISTAN,0.875,$E
PNT(10)=POINT/PNT(7),DISTAN,0.875,$E
$EP1=LINE/PNT(9),PERP,$E,LENGTH,-14
PNT(11)=POINT/PNT(9),DISTAN,14,$EP1
$EP2=LINE/PNT(10),PERP,$E,LENGTH,-12
PNT(12)=POINT/PNT(10),DISTAN,12,$EP2
DELETE $E,$EP1,$EP2
$E=LINE/P(37),PNT(4)
PNT(13)=POINT/P(37),DISTAN,x(35)-x(34),$E
PNT(14)=POINT/PNT(13),DISTAN,2,$E
PNT(15)=POINT/PNT(13),DISTAN,1,$E
$EP=LINE/PNT(15),PERP,$E,LENGTH,10
PNT(16)=POINT/PNT(15),DISTAN,10,$EP
DELETE $E,$EP
<#
<#
<#
p(1)=POINT/x(11),y(11),0
p(2)=POINT/x(27),y(27),0
p(3)=POINT/x(22),y(22),0
p(4)=POINT/x(28),y(28),0
<#
<#
<#
L(1)=LINE/P(1),P(4)
L(2)=LINE/P(4),P(29)
L(3)=LINE/P(12),P(29)
L(4)=LINE/P(12),P(30)
L(5)=LINE/P(30),P(31)
L(6)=LINE/P(23),P(31)
L(7)=LINE/P(1),P(25)
L(8)=LINE/P(6),PNT(1)
L(9)=LINE/PNT(1),P(9)
L(10)=LINE/PNT(2),P(9)
L(11)=LINE/PNT(2),P(7)
L(12)=LINE/P(2),P(23),FONT,'DASH'
L(13)=LINE/P(4),P(24),FONT,'DASH'
<#
<#
<#
L(14)=LINE/P(15),P(35)
```

```
L(15)=LINE/P(34),P(35)
L(16)=LINE/P(19),P(34)
L(17)=LINE/P(19),P(33)
L(18)=LINE/P(32),P(33)
L(19)=LINE/P(23),P(32)
L(20)=LINE/P(14),P(19)
L(21)=LINE/P(20),P(19)
L(22)=LINE/P(20),P(21)
L(23)=LINE/P(3),P(23),FONT,'DASH'
L(24)=LINE/P(5),P(24),FONT,'DASH'
L(25)=LINE/x(1)+5,y(15)-5,0,x(1)+5,y(4)+5,0,FONT,'ARROW'
L(26)=LINE/x(3)-5,y(15)-5,0,x(3)-5,y(4)+5,0,FONT,'ARROW'
B(1)=BSPLIN/3,P(23),P(27),P(11),P(7)        <# b-armhole
$L1=LINE/P(21),P(22)
$L2=LINE/(x(21)+x(22))/2,(y(21)+y(22))/2,0,<#
>PERP,$L1,LENGTH,0.5
p(5)=POINT/(x(21)+x(22))/2,(y(21)+y(22))/2,0,<#
>DISTAN,0.5,$L2
DELETE $L1,$L2
B(2)=BSPLIN/3,P(23),P(28),P(22),p(5),P(21) <# f-armhole
B(3)=BSPLIN/3,P(25),P(26),P(6)              <# b-neck
C(1)=CIRCLE/CENTER,P(11),RADIUS,0.4
C(2)=CIRCLE/CENTER,P(22),RADIUS,0.4
C(3)=CIRCLE/RADIUS,NECK/5-0.2,RIGHT,P(14),P(15),<#
>PGO,P(14),PEND,P(15)
<#
<#
L(27)=LINE/P(36),P(38)
L(28)=LINE/P(38),P(42)
L(29)=LINE/P(43),P(42)
L(30)=LINE/P(40),P(43),FONT,'DASH'
L(31)=LINE/P(37),P(39)
L(32)=LINE/P(39),P(42)
L(33)=LINE/P(41),P(43),FONT,'DASH'
C(4)=CIRCLE/RADIUS,BUST*1.5,LEFT,P(43),PNT(3),<#
>PGO,P(43),PEND,PNT(3)
C(5)=CIRCLE/RADIUS,BUST*1.5,RIGHT,PNT(4),P(43),<#
>PGO,PNT(4),PEND,P(43)
L(34)=LINE/PNT(5),PNT(11)
L(35)=LINE/PNT(6),PNT(11)
L(36)=LINE/PNT(7),PNT(12)
L(37)=LINE/PNT(8),PNT(12)
L(38)=LINE/PNT(14),PNT(16)
L(39)=LINE/PNT(13),PNT(16)
L(40)=LINE/x(40)+5,y(40)+5,0,x(40)+5,y(38)+10,0,<#
>FONT,'ARROW'
L(41)=LINE/x(41)-5,y(40)+5,0,x(41)-5,y(38)+10,0,<#
>FONT,'ARROW'
C(6)=CIRCLE/RADIUS,BUST*2/3,LEFT,P(36),PNT(5),<#
>PGO,P(36),PEND,PNT(5)
C(7)=CIRCLE/RADIUS,BUST*2/3,LEFT,PNT(6),PNT(7),<#
>PGO,PNT(6),PEND,PNT(7)
C(8)=CIRCLE/RADIUS,BUST*2/3,LEFT,PNT(8),PNT(3),<#
>PGO,PNT(8),PEND,PNT(3)
C(9)=CIRCLE/RADIUS,BUST*2/3,RIGHT,PNT(4),PNT(14),<#
>PGO,PNT(4),PEND,PNT(14)
C(10)=CIRCLE/RADIUS,BUST*2/3,RIGHT,PNT(13),P(37),<#
>PGO,PNT(13),PEND,P(37)
```

```
DELETE 16,PNT(1)
<#
<#
&T1='two-piece dress block'
T(1)=TEXT/&T1,(x(38)+x(42))/2,y(38)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(39)+x(42))/2,y(38)+2,0,TJST,'CJT',THGT,0.75
&T2='close fitting bodice'
T(3)=TEXT/&T2,(x(38)+x(42))/2,y(38)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&T2,(x(39)+x(42))/2,y(38)+3,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
        T(5)=TEXT/&name+' < SIZE '+&n+' >',(x(38)+x(42))/2<#
>,y(38)+4,0,TJST,'CJT',THGT,0.75
        T(6)=TEXT/&name+' < SIZE '+&n+' >',(x(39)+x(42))/2<#
>,y(38)+4,0,TJST,'CJT',THGT,0.75
  OR &data='2'
        IF (&name='') GOTO 200
T(5)=TEXT/&name,(x(38)+x(42))/2,y(38)+4,0,TJST,'CJT',THGT,0.75
T(6)=TEXT/&name,(x(39)+x(42))/2,y(38)+4,0,TJST,'CJT',THGT,0.75
        #200 CONTINUE
ENDWHEN
T(7)=TEXT/'centre back line',x(1)+1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(8)=TEXT/'centre front line',x(3)-1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
DISP ALL
#999 END
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|              The easy fitting two-piece dress (DR2EA)                 |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE REAL x(35),y(35)
DECLARE LOCATION P(35)
DECLARE ENTITY p(4),L(24),B(3),C(9),T(8)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP DRSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,ARMDPT,<#
>CHEST,NECK,WATOKN,&n,&data)
<#
<#
PRINT {''}
PRINT   Select the position for the centre back neck<#
> of the dress block please.
PRINT  ------------------------------------<#
>----------------------------
PRINT     Move the mouse on the pad to place the cursor<#
> on the graphic window
PRINT     at the position you require.<#
> Press the left button.
PRINT  ------------------------------------<#
>----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(2)=:x(4)=:x(26)=:x(30)=:x(28)=:X
y(1)=y(22)=:Y
y(2)=y(8)=:y(17)=:y(13)=:y(15)=:y(3)=:y(1)-ARMDPT-2.5
x(3)=x(11)=:x(5)=:x(21)=:x(27)=:x(31)=:x(29)=:x(1)+BUST/2+7
y(4)=y(18)=:y(5)=:y(1)-NATOWA
x(6)=x(1)+NECK/5-0.2
y(6)=y(1)+1.5
y(7)=y(1)-ARMDPT/5+1
x(7)=x(6)+SQRT((SHLN+1)**2-(y(6)-y(7))**2)
x(8)=x(9)=:x(1)+BWD/2+1
y(9)=(y(8)+y(7))/2
x(10)=x(3)-NECK/5+0.7
WHEN BUST<95
          y(10)=y(12)=:y(1)+1.5
   OR BUST.GE.95.AND.BUST<100
          y(10)=y(12)=:y(1)+1.8
   OR BUST.GE.100.AND.BUST<105
          y(10)=y(12)=:y(1)+2.1
   OR BUST.GE.105.AND.BUST<110
          y(10)=y(12)=:y(1)+2.4
   OR BUST.GE.110.AND.BUST<115
```

```
        y(10)=y(12)=:y(1)+2.7
  OR BUST.GE.115.AND.BUST<120
        y(10)=y(12)=:y(1)+3
  OR BUST.GE.120.AND.BUST<125
        y(10)=y(12)=:y(1)+3.3
  OR BUST.GE.125.AND.BUST<130
        y(10)=y(12)=:y(1)+3.6
  ELSE
        y(10)=y(12)=:y(1)+3.9
ENDWHEN
y(11)=y(10)-NECK/5+0.2
x(12)=x(10)-DART/2
x(13)=x(14)=:x(3)-(CHEST/2+1+DART/4)
y(14)=(y(3)+y(11))/2
x(15)=(x(3)+x(13))/2
y(16)=y(7)-1.5
x(16)=x(12)-SQRT((SHLN+0.5)**2-(y(12)-y(16))**2)
x(17)=x(18)=:(x(8)+x(13))/2
x(19)=x(34)=:x(17)-1.5
y(19)=y(20)=:y(4)-0.5
x(20)=x(35)=:x(17)+2.5
y(21)=y(5)-1
x(22)=x(1)+(x(6)-x(1))/3
x(23)=x(6)-(x(6)-x(1))/4
y(23)=y(1)+(y(6)-y(1))/3
WHEN BUST<95
        backcurve=2.5
        frontcurve=2.25
  OR BUST.GE.95.AND.BUST.LT.115
        backcurve=3
        frontcurve=2.75
  ELSE
        backcurve=3.5
        frontcurve=3.25
ENDWHEN
x(24)=x(8)+backcurve/SQRT(2)
y(24)=y(8)+backcurve/SQRT(2)
x(25)=x(13)-frontcurve/SQRT(2)
y(25)=y(13)+frontcurve/SQRT(2)
y(26)=y(27)=:y(4)-5
y(28)=y(29)=:y(32)=:y(26)-WATOKN
y(30)=y(33)=:y(31)=:y(26)-WATOHI
x(32)=x(33)=:(x(28)+x(29))/2
y(34)=y(35)=:y(26)+1.25
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>35
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
p(1)=POINT/x(9),y(9),0
p(2)=POINT/x(24),y(24),0
p(3)=POINT/x(14),y(14),0
p(4)=POINT/x(25),y(25),0
<#
<#
```

```
L(1)=LINE/P(1),P(4)
L(2)=LINE/P(6),P(7)
L(3)=LINE/P(17),P(19)
L(4)=LINE/P(1),P(22)
L(5)=LINE/P(2),P(17),FONT,'DASH'
L(6)=LINE/P(4),P(18),FONT,'DASH'
<#
<#
<#
L(7)=LINE/P(11),P(21)
L(8)=LINE/P(10),P(15)
L(9)=LINE/P(12),P(15)
L(10)=LINE/P(12),P(16)
L(11)=LINE/P(17),P(20)
L(12)=LINE/P(3),P(17),FONT,'DASH'
L(13)=LINE/P(5),P(18),FONT,'DASH'
L(14)=LINE/x(1)+5,y(11)-5,0,x(1)+5,y(4)+5,0,FONT,'ARROW'
L(15)=LINE/x(3)-5,y(11)-5,0,x(3)-5,y(4)+5,0,FONT,'ARROW'
B(1)=BSPLIN/3,P(17),P(24),P(9),P(7)          <# b-armhole
B(2)=BSPLIN/3,P(17),P(25),P(14),P(16)        <# f-armhole
B(3)=BSPLIN/3,P(22),P(23),P(6)               <# b-neck
C(1)=CIRCLE/RADIUS,NECK/5-0.2,RIGHT,P(10),P(11),PGO,P(10),PEND,P(1
C(2)=CIRCLE/RADIUS,BUST*3,RIGHT,P(4),P(19),PGO,P(4),PEND,P(19)
C(3)=CIRCLE/RADIUS,BUST*3,RIGHT,P(20),P(21),PGO,P(20),PEND,P(21)
C(4)=CIRCLE/CENTER,P(9),RADIUS,0.4
C(5)=CIRCLE/CENTER,P(14),RADIUS,0.4
<#
<#
<#
L(16)=LINE/P(26),P(28)
L(17)=LINE/P(28),P(32)
L(18)=LINE/P(33),P(32)
L(19)=LINE/P(30),P(33),FONT,'DASH'
<#
<#
<#
L(20)=LINE/P(27),P(29)
L(21)=LINE/P(29),P(32)
L(22)=LINE/P(31),P(33),FONT,'DASH'
L(23)=LINE/x(30)+5,y(30)+5,0,x(30)+5,y(28)+10,0,FONT,'ARROW'
L(24)=LINE/x(31)-5,y(30)+5,0,x(31)-5,y(28)+10,0,FONT,'ARROW'
C(6)=CIRCLE/RADIUS,BUST*1.5,LEFT,P(26),P(34),<#
>PGO,P(26),PEND,P(34)
C(7)=CIRCLE/RADIUS,BUST*1.5,RIGHT,P(35),P(27),<#
>PGO,P(35),PEND,P(27)
C(8)=CIRCLE/RADIUS,BUST*1.5,LEFT,P(33),P(34),<#
>PGO,P(33),PEND,P(34)
C(9)=CIRCLE/RADIUS,BUST*1.5,RIGHT,P(35),P(33),<#
>PGO,P(35),PEND,P(33)
<#
<#
<#
&T1='two-piece dress block'
T(1)=TEXT/&T1,(x(28)+x(32))/2,y(28)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(29)+x(32))/2,y(28)+2,0,TJST,'CJT',THGT,0.75
&T2='easy fitting bodice'
T(3)=TEXT/&T2,(x(28)+x(32))/2,y(28)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&T2,(x(29)+x(32))/2,y(28)+3,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
          T(5)=TEXT/&name+' < SIZE '+&n+' >',(x(28)+x(32))/2<#
>,y(28)+4,0,TJST,'CJT',THGT,0.75
          T(6)=TEXT/&name+' < SIZE '+&n+' >',(x(29)+x(32))/2<#
```

```
>,y(28)+4,0,TJST,'CJT',THGT,0.75
  OR &data='2'
        IF (&name='') GOTO 200
T(5)=TEXT/&name,(x(28)+x(32))/2,y(28)+4,0,TJST,'CJT',THGT,0.75
T(6)=TEXT/&name,(x(29)+x(32))/2,y(28)+4,0,TJST,'CJT',THGT,0.75
        #200 CONTINUE
ENDWHEN
T(7)=TEXT/'centre back line',x(1)+1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(8)=TEXT/'centre front line',x(3)-1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
DISP ALL
#999 END
```

```
+-----------------------------------------------------------+
|                                                           |
|     The dartless easy fitting two-piece dress (DR2DL)     |
|                                                           |
+-----------------------------------------------------------+
<#
<#
<#
DECLARE REAL x(33),y(33)
DECLARE LOCATION P(33)
DECLARE ENTITY p(4),L(22),B(3),C(9),T(8)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the name of patterns please. -----> )&name
<#
<#
CALLP DRSIZE(BUST,SHLN,NATOWA,BWD,WATOHI,DART,ARMDPT,<#
>CHEST,NECK,WATOKN,&n,&data)
<#
<#
PRINT {''}
PRINT   Select the position for the centre back neck<#
> of the dress block please.
PRINT  ------------------------------------<#
>----------------------------
PRINT     Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT     at the position you require.<#
> Press the left button.
PRINT  ------------------------------------<#
>----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
x(1)=x(2)=:x(4)=:x(24)=:x(28)=:x(26)=:X
y(1)=y(17)=:Y
y(2)=y(8)=:y(15)=:y(12)=:y(3)=:y(1)-ARMDPT-2.5
x(3)=x(11)=:x(5)=:x(23)=:x(25)=:x(29)=:x(27)=:<#
>x(1)+BUST/2+7
y(4)=y(16)=:y(5)=:y(1)-NATOWA
x(6)=x(1)+NECK/5-0.2
y(6)=y(1)+1.5
y(7)=y(1)-ARMDPT/5+1
x(7)=x(6)+SQRT((SHLN+1)**2-(y(6)-y(7))**2)
x(8)=x(9)=:x(1)+BWD/2+1
y(9)=(y(8)+y(7))/2
x(10)=x(3)-NECK/5+0.7
WHEN BUST<95
          y(10)=y(1)+1.5
   OR BUST.GE.95.AND.BUST<100
          y(10)=y(1)+1.8
   OR BUST.GE.100.AND.BUST<105
          y(10)=y(1)+2.1
   OR BUST.GE.105.AND.BUST<110
          y(10)=y(1)+2.4
```

```
  OR BUST.GE.110.AND.BUST<115
        y(10)=y(1)+2.7
  OR BUST.GE.110.AND.BUST<120
        y(10)=y(1)+3
  OR BUST.GE.120.AND.BUST<125
        y(10)=y(1)+3.3
  OR BUST.GE.125.AND.BUST<130
        y(10)=y(1)+3.6
  ELSE
        y(10)=y(1)+3.9
ENDWHEN
y(11)=y(10)-NECK/5+0.2
x(12)=x(13)=:x(3)-(CHEST/2+1.5)
y(13)=(y(3)+y(11))/2
y(14)=y(7)-0.75
x(14)=x(10)-SQRT((SHLN+0.5)**2-(y(10)-y(14))**2)
x(15)=x(16)=:(x(8)+x(12))/2
x(17)=x(1)+(x(6)-x(1))/3
x(18)=x(6)-(x(6)-x(1))/4
y(18)=y(1)+(y(6)-y(1))/3
WHEN BUST<95
        backcurve=2.5
        frontcurve=2.25
  OR BUST.GE.95.AND.BUST.LT.115
        backcurve=3
        frontcurve=2.75
  ELSE
        backcurve=3.5
        frontcurve=3.25
ENDWHEN
x(19)=x(8)+backcurve/SQRT(2)
y(19)=y(8)+backcurve/SQRT(2)
x(20)=x(12)-frontcurve/SQRT(2)
y(20)=y(12)+frontcurve/SQRT(2)
x(21)=x(32)=:x(16)-1.5
x(22)=x(33)=:x(16)+2.5
y(21)=y(22)=:y(4)-0.5
y(23)=y(4)-1
y(24)=y(25)=:y(4)-5
y(26)=y(27)=:y(30)=:y(24)-WATOKN
y(28)=y(29)=:y(31)=:y(24)-WATOHI
x(30)=x(31)=:(x(26)+x(27))/2
y(32)=y(33)=:y(24)+1.25
<#
<#
DISP OFF
REPEAT 100: i=1,1,i>33
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
p(1)=POINT/x(9),y(9),0
p(2)=POINT/x(19),y(19),0
p(3)=POINT/x(13),y(13),0
p(4)=POINT/x(20),y(20),0
<#
<#
L(1)=LINE/P(1),P(4)
```

```
L(2)=LINE/P(6),P(7)
L(3)=LINE/P(1),P(17)
L(4)=LINE/P(15),P(21)
L(5)=LINE/P(2),P(15),FONT,'DASH'
L(6)=LINE/P(4),P(16),FONT,'DASH'
<#
<#
<#
L(7)=LINE/P(11),P(23)
L(8)=LINE/P(10),P(14)
L(9)=LINE/P(15),P(22)
L(10)=LINE/P(3),P(15),FONT,'DASH'
L(11)=LINE/P(5),P(16),FONT,'DASH'
L(12)=LINE/x(1)+5,y(11)-5,0,x(1)+5,y(4)+5,0,FONT,'ARROW'
L(13)=LINE/x(3)-5,y(11)-5,0,x(3)-5,y(4)+5,0,FONT,'ARROW'
B(1)=BSPLIN/3,P(15),P(19),P(9),P(7)          <# b-armhole
B(2)=BSPLIN/3,P(15),P(20),P(13),P(14)        <# f-armhole
B(3)=BSPLIN/3,P(17),P(18),P(6)               <# b-neck
C(1)=CIRCLE/RADIUS,NECK/5-0.2,RIGHT,P(10),P(11),<#
>PGO,P(10),PEND,P(11)
C(2)=CIRCLE/RADIUS,BUST*3,RIGHT,P(4),P(21),<#
>PGO,P(4),PEND,P(21)
C(3)=CIRCLE/RADIUS,BUST*3,RIGHT,P(22),P(23),<#
>PGO,P(22),PEND,P(23)
C(4)=CIRCLE/CENTER,P(9),RADIUS,0.4
C(5)=CIRCLE/CENTER,P(13),RADIUS,0.4
<#
<#
<#
L(14)=LINE/P(24),P(26)
L(15)=LINE/P(26),P(30)
L(16)=LINE/P(31),P(30)
L(17)=LINE/P(28),P(31),FONT,'DASH'
<#
<#
<#
L(18)=LINE/P(25),P(27)
L(19)=LINE/P(27),P(30)
L(20)=LINE/P(29),P(31),FONT,'DASH'
L(21)=LINE/x(28)+5,y(28)+5,0,x(28)+5,y(26)+10,0,FONT,'ARROW'
L(22)=LINE/x(29)-5,y(28)+5,0,x(29)-5,y(26)+10,0,FONT,'ARROW'
C(6)=CIRCLE/RADIUS,BUST*1.5,LEFT,P(24),P(32),<#
>PGO,P(24),PEND,P(32)
C(7)=CIRCLE/RADIUS,BUST*1.5,RIGHT,P(33),P(25),<#
>PGO,P(33),PEND,P(25)
C(8)=CIRCLE/RADIUS,BUST*1.5,LEFT,P(31),P(32),<#
>PGO,P(31),PEND,P(32)
C(9)=CIRCLE/RADIUS,BUST*1.5,RIGHT,P(33),P(31),<#
>PGO,P(33),PEND,P(31)
<#
<#
<#
&T1='two-piece dress block'
T(1)=TEXT/&T1,(x(26)+x(30))/2,y(26)+2,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/&T1,(x(27)+x(30))/2,y(26)+2,0,TJST,'CJT',THGT,0.75
&T2='dartless bodice'
T(3)=TEXT/&T2,(x(26)+x(30))/2,y(26)+3,0,TJST,'CJT',THGT,0.75
T(4)=TEXT/&T2,(x(27)+x(30))/2,y(26)+3,0,TJST,'CJT',THGT,0.75
WHEN &data='1'
        T(5)=TEXT/&name+' < SIZE '+&n+' >',(x(26)+x(30))/2<#
>,y(26)+4,0,TJST,'CJT',THGT,0.75
        T(6)=TEXT/&name+' < SIZE '+&n+' >',(x(27)+x(30))/2<#
```

```
>,y(26)+4,0,TJST,'CJT',THGT,0.75
  OR &data='2'
        IF (&name='') GOTO 200
T(5)=TEXT/&name,(x(26)+x(30))/2,y(26)+4,0,TJST,'CJT',THGT,0.75
T(6)=TEXT/&name,(x(27)+x(30))/2,y(26)+4,0,TJST,'CJT',THGT,0.75
        #200 CONTINUE
ENDWHEN
T(7)=TEXT/'centre back line',x(1)+1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(8)=TEXT/'centre front line',x(3)-1,(y(1)+y(4))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,90
<#
<#
DISP ALL
#999 END
```

APPENDIX B

ADDITIONAL PROGRAMS

```
+-----------------------------------------------------------------------+
|                                                                       |
|                                Openfile                               |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE TEXT pname,dname,curpart,curdraw
<#
<#
PRINT Enter the part name and drawing name please.
REPEAT
        READ( Part name -----> )pname
UNTIL pname <> ''
REPEAT
        READ( Drawing name -----> )dname
UNTIL dname <> ''
<#
<#
ERTRAP 999
PRTNAM curpart,curdraw
WHEN curpart <> pname
        REPEAT
        READ( Would you like to file the current part?  <#
>Y/N -----> )&answer
        UNTIL &answer='Y'.OR.&answer='y'.OR.&answer='N'<#
>.OR.&answer='n'
        WHEN &answer='Y'.OR.&answer='y'
                !EXIT PART F
          OR &answer='N'.OR.&answer='n'
                !EXIT PART Q
        ENDWHEN
        PRINT {''}
        PRINT Wait a moment please, the part is being created.
        !SELECT UNITS CM
        !ACT PART {pname}
        !SELECT TAG ON
        !SELECT DRAWING UNIT CM
        !SELECT DRAW SIZE E
        !ACT DRA {dname}
        !ECHO FRAME
        !ECHO TAG ON
        !DEFINE VIEW TOP CPLANE TOP SCALE 0.5 :X22Y17
        EXECDF
OR curdraw=''.OR.curdraw<>dname
        !ACT DRA {dname}
        EXECDF
ENDWHEN
END
<#
<#
#999 CONTINUE
ERNUM errcode,errline
WHEN errcode=48
        PRINT {''}
        PRINT Wait a moment please, the part is being created.
        !SELECT UNITS CM
        !ACT PART {pname}
```

```
        !SELECT TAG ON
        !SELECT DRAWING UNIT CM
        !SELECT DRAW SIZE E
        !ACT DRA {dname}
        !ECHO FRAME
        !ECHO TAG ON
        !DEFINE VIEW TOP CPLANE TOP SCALE 0.5 :X22Y17
        UNTRAP
        EXECDF
ENDWHEN
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|                        Stand collar (CLSTAND)                         |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE REAL x(11),y(11)
DECLARE LOCATION P(11)
DECLARE ENTITY L(8),T(3)
ERTRAP 999
<#
<#
PRINT {''}
PRINT          ----------------------------------<#
>-------------------------------
PRINT                You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ----------------------------------<#
>-------------------------------
PRINT          * Measure the length of a half back neck<#
> line and a half front
PRINT            neck line using the icon (MEASURE LENGTH).
PRINT            The unit of the length will be "cm".
PRINT          * After Measuring, please press keys<#
> "Control" and "x" at the
PRINT            same time  to restart the program.
PRINT          ----------------------------------<#
>-------------------------------
!<VAR>
EXECDF
READ( Enter the length of the half back neckline<#
> please. <unit: cm> ---> )bneck
READ( Enter the length of the half front neckline<#
> please. <unit: cm> ---> )fneck
<#
<#
#10 CONTINUE
PRINT {''}
READ( Which units do you want to use?    <#
> Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'
          cwidth=5
          buttonstand=1.5
       PRINT default value: collar width = 5cm
       READ( Enter the width of the standing collar please. <#
><unit: cm> ---> )cwidth
       PRINT default value: buttonstand = 1.5cm
       READ( Enter the width of the buttonstand please. <#
><unit: cm> ---> )buttonstand
  OR &unit='IN'.OR.&unit='in'
          cwidth=2
          buttonstand=0.6
       PRINT default value: collar width = 2in
       READ( Enter the width of the standing collar please. <#
><unit: in>  ---> )cwidth
       PRINT default value: buttonstand = 0.6in
       READ( Enter the width of the buttonstand please. <#
```

```
><unit: in> ---> )buttonstand
          cwidth=cwidth*2.54
          buttonstand=buttonstand*2.54
   ELSE
       PRINT Wrong answer, enter again please.
       GOTO 10
ENDWHEN
<#
<#
PRINT {''}
PRINT      Select the position for the left lower <#
>corner of the collar please.
PRINT   ------------------------------------<#
>-----------------------------
PRINT   * Move the mouse on the pad to place the <#
>cursor on the graphic window
PRINT      at the position you require. Press the <#
>left button.
PRINT   ------------------------------------<#
>-----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
DISP OFF
neck=bneck+fneck
x(1)=x(2)=:X
y(1)=y(6)=:y(7)=:y(11)=:Y
y(2)=y(5)=:y(4)=:y(1)+cwidth
x(3)=x(1)+neck+buttonstand
y(3)=y(1)+1
x(4)=x(3)-1.5
x(5)=x(4)-buttonstand
x(6)=(x(1)+x(3))/2
x(7)=x(3)-buttonstand
x(8)=(x(5)+x(7))/2+0.25
y(8)=y(9)=:(y(1)+y(2))/2
x(9)=x(8)-1.5
x(10)=(x(6)+x(3))/2
y(10)=y(1)+0.3
x(11)=x(1)+bneck
<#
<#
REPEAT 100: i=1,1,i>11
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
L(1)=LINE/P(1),P(2)
L(2)=LINE/P(2),P(4)
L(3)=LINE/P(4),P(3)
L(4)=LINE/P(1),P(6)
L(5)=LINE/P(8),P(9)
$B=BSPLIN/3,P(6),P(10),P(3)
$E=LINE/P(7),VERT,LENGTH,1
$P=POINT/INTOF,$E,YSMALL,$B
L(6)=LINE/P(5),$P,FONT,'DASH'
```

```
DELETE $E,$P
$C=CIRCLE/CENTER,P(11),RADIUS,0.4
L(7)=LINE/x(1)+3,y(2)-1,0,x(1)+3,y(1)+1,0,FONT,'ARROW'
L(8)=LINE/x(1)+1.5,y(8)+1.5,0,x(1)+4.5,y(8)-1.5,0,<#
>FONT,'ARROW'
T(1)=TEXT/'CB fold',x(1)+0.5,(y(1)+y(2))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(2)=TEXT/'neck',x(6),y(1)+0.5,0,TJST,'RJT',THGT,0.75
T(3)=TEXT/'standing straight collar',x(1)+3,<#
>(y(1)+y(2))/2-0.4,0,<#
>TJST,'LJT',THGT,0.75
DISP ALL
#999 END
```

+--+
| |
| Shirt collar (CLSHIRT) |
| |
+--+

```
<#
<#
DECLARE REAL x(18),y(18)
DECLARE LOCATION P(18)
DECLARE ENTITY L(11),B(3),T(6),C(4)
ERTRAP 999
<#
<#
#10 CONTINUE
PRINT {''}
&shcollar=''
PRINT              Which shirt collar do you want?
PRINT     ----------------------------------------
PRINT     Collar and stand in one piece ------------ 1
PRINT     Collar with separate stand ---------------- 2
PRINT     ----------------------------------------
READ(          Enter the number please. ---> )&shcollar
WHEN &shcollar='1'.OR.&shcollar='2'
          CONTINUE
  ELSE
          PRINT Wrong answer, enter again please.
          GOTO 10
ENDWHEN
<#
<#
PRINT {''}
PRINT          ------------------------------<#
>------------------------------
PRINT               You can use the icon tool, while<#
> the prompt is "<VAR>".
PRINT          ------------------------------------<#
>------------------------------
PRINT          * Measure the length of a half back neck<#
> line and a half front
PRINT            neck line using the icon (MEASURE LENGTH).
PRINT            The unit of the length will be "cm".
PRINT          * After Measuring, please press keys<#
> "Control" and "x" at the
PRINT            same time  to restart the program.
PRINT          ------------------------------------<#
>------------------------------
!<VAR>
EXECDF
READ( Enter the length of the half back neckline please.<#
> <unit: cm> ---> )bneck
READ( Enter the length of the half front neckline<#
> please. <unit: cm> ---> )fneck
<#
<#
#20 CONTINUE
PRINT {''}
READ( Which units do you want to use?  <#
>  Enter "IN" or "CM" ---> )&unit
```

```
WHEN &unit='CM'.OR.&unit='cm'
        cstand=3
        cwidth=5
        buttonstand=1.5
   PRINT default value: collar width = 5cm
   READ( Enter the width of the shirt collar please. <#
><unit: cm> ---> )cwidth
   PRINT default value: collar stand depth = 3cm
   READ( Enter the depth of the shirt collar stand please. <#
><unit: cm> ---> )cstand
   PRINT default value: buttonstand = 1.5cm
   READ( Enter the width of the buttonstand please. <#
><unit: cm> ---> )buttonstand
  OR &unit='IN'.OR.&unit='in'
        cstand=1.2
        cwidth=2
        buttonstand=0.6
   PRINT default value: collar width = 2in
   READ( Enter the width fo the shirt collar please. <#
><unit: in> ---> )cwidth
   PRINT default value: collar stand depth = 1.2in
   READ( Enter the depth of the shirt collar stand please. <#
><unit: in> ---> )cstand
   PRINT default value: buttonstand = 0.6in
   READ( Enter the width of the buttonstand please. <#
><unit:in> ---> )buttonstand
        cwidth=cwidth*2.54
        cstand=cstand*2.54
        buttonstand=buttonstand*2.54
  ELSE
   PRINT Wrong answer, enter again please.
        GOTO 20
ENDWHEN
<#
<#
PRINT {''}
PRINT     Select the position for the left <#
>lower corner of the collar please.
PRINT   ------------------------------------<#
>--------------------------
PRINT   * Move the mouse on the pad to place the cursor<#
> on the graphic window
PRINT     at the position you require. Press the <#
>left button.
PRINT   ------------------------------------<#
>--------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
DISP OFF
neck=bneck+fneck
x(1)=x(2)=:x(5)=:x(15)=:X
y(1)=y(11)=:y(3)=:Y
y(2)=y(9)=:y(1)+cstand+cwidth
x(3)=x(9)=:x(14)=:x(1)+neck*3/4
x(4)=x(6)=:x(1)+neck
```

```
y(4)=y(1)+0.5
y(5)=y(12)=:y(14)=:y(6)=:y(7)=:y(1)+cstand
x(7)=x(6)-0.75
x(8)=x(4)+buttonstand
y(8)=y(4)+0.4
x(10)=x(6)+1
y(10)=y(9)+1
x(11)=x(12)=:x(13)=:x(1)+bneck
y(13)=y(12)+0.6
y(15)=y(5)+0.75
x(16)=x(4)+0.3
y(16)=(y(4)+y(6))/2
x(17)=x(16)-1.5
y(17)=y(16)-1.5*(y(3)-y(4))/(x(3)-x(4))
x(18)=(x(9)+x(10))/2
y(18)=y(9)+(y(10)-y(9))/3
<#
<#
REPEAT 100: i=1,1,i>18
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
L(1)=LINE/P(1),P(2)
L(2)=LINE/P(1),P(3)
L(3)=LINE/P(5),P(6)
L(4)=LINE/P(2),P(9)
L(5)=LINE/P(10),P(7)
L(6)=LINE/P(9),P(3),FONT,'DASH'
L(7)=LINE/P(16),P(17)
L(8)=LINE/x(1)+4,y(2)-1,0,x(1)+4,y(2)-3,0,FONT,'ARROW'
B(1)=BSPLIN/3,P(9),P(18),P(10)
B(2)=BSPLIN/3,P(3),P(4),P(8)
C(1)=CIRCLE/RADIUS,(y(6)-y(4))*4/5,LEFT,P(8),P(6),<#
>PGO,P(8),PEND,P(6)
C(2)=CIRCLE/CENTER,P(11),RADIUS,0.4
WHEN &shcollar='1'
        T(1)=TEXT/'CB fold',x(1)+0.5,(y(1)+y(2))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
        T(2)=TEXT/'collar & stand',x(1)+3,y(5)+0.5,0,<#
>TJST,'LJT',THGT,0.75
  OR &shcollar='2'
        DELETE L(1)
        L(10)=LINE/P(2),P(15)
        L(10)=LINE/P(1),P(5)
        L(11)=LINE/x(1)+4,y(5)-0.5,0,x(1)+4,y(1)+0.5,0,<#
>FONT,'ARROW'
        B(3)=BSPLIN/3,P(15),P(13),P(14)
        C(3)=CIRCLE/CENTER,P(12),RADIUS,0.4
        C(4)=CIRCLE/CENTER,P(13),RADIUS,0.4
        T(1)=TEXT/'CB',x(1)+1.5,(y(2)+y(15))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
        T(2)=TEXT/'fold',x(1)+0.5,(y(2)+y(15))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
        T(3)=TEXT/'CB',x(1)+1.5,(y(1)+y(5))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
        T(4)=TEXT/'fold',x(1)+0.5,(y(1)+y(5))/2,0,<#
>TJST,'CJT',THGT,0.75,TANG,270
```

```
        T(5)=TEXT/'collar',x(11),(y(2)+y(15))/2-0.4,0,<#
>TJST,'CJT',THGT,0.75
        T(6)=TEXT/'stand',x(11),(y(1)+y(5))/2-0.4,0,<#
>TJST,'CJT',THGT,0.75
ENDWHEN
DISP ALL
#999 END
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|                    Convertible collar (CLCONVER)                      |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE REAL x(10), y(10)
DECLARE LOCATION P(10)
DECLARE ENTITY L(6),B(2),T(3)
ERTRAP 999
<#
<#
PRINT {''}
PRINT          ------------------------------<#
>------------------------------
PRINT                You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ------------------------------<#
>------------------------------
PRINT          * Measure the length of a half back neck<#
> line and a half front
PRINT            neck line using the icon (MEASURE LENGTH).
PRINT            The unit of the length will be "cm".
PRINT          * After Measuring, please press keys<#
> "Control" and "x" at the
PRINT            same time  to restart the program.
PRINT          ------------------------------<#
>------------------------------
!<VAR>
EXECDF
READ( Enter the length of the half back neckline<#
> please. <unit: cm> ---> )bneck
READ( Enter the length of the half front neckline<#
> please. <unit: cm> ---> )fneck
<#
<#
#10 CONTINUE
PRINT {''}
READ( Which units do you want to use?<#
>     Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'
         cwidth=9
PRINT default value: collar width = 9cm (including folded part)
READ( Enter the width of the convertible collar please. <#
><unit:cm> ---> )cwidth
  OR &unit='IN'.OR.&unit='in'
    cwidth=3.5
PRINT default value: collar width = 3.5in <#
>(including folded part)
READ( Enter the width of the convertible collar please. <#
><unit:in> ---> )cwidth
         cwidth=cwidth*2.54
  ELSE
         PRINT Wrong answer, enter again please.
         GOTO 10
ENDWHEN
<#
```

```
<#
PRINT {''}
PRINT      Select the position for the left<#
> lower corner of the collar please.
PRINT   ------------------------------------<#
>-----------------------------------
PRINT    * Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT      at the position you require. Press the<#
> left button.
PRINT   ------------------------------------------<#
>---------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
DISP OFF
neck=bneck+fneck
x(1)=x(2)=:x(8)=:X
y(1)=y(3)=:y(10)=:Y
y(2)=y(4)=:y(6)=:y(1)+cwidth
x(3)=x(4)=:x(1)+neck*3/4
x(5)=x(1)+neck
y(5)=y(1)+0.5
x(6)=x(5)+1.5
x(7)=(x(3)+x(5))/2
y(7)=y(1)+0.15
y(8)=y(1)+3.5
x(9)=(x(3)+x(8))/2
y(9)=y(1)+2.625
x(10)=x(1)+bneck
<#
<#
REPEAT 100: i=1,1,i>10
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
L(1)=LINE/P(1),P(2)
L(2)=LINE/P(2),P(6)
L(3)=LINE/P(6),P(5)
L(4)=LINE/P(1),P(3)
L(5)=LINE/P(4),P(3),FONT,'DASH'
L(6)=LINE/x(10),y(2)-1,0,x(10),y(8),0,FONT,'ARROW'
$C=CIRCLE/CENTER,P(10),RADIUS,0.4
B(1)=BSPLIN/3,P(8),P(9),P(3),FONT,'DASH'
B(2)=BSPLIN/3,P(3),P(7),P(5)
T(1)=TEXT/'CB fold',x(1)+0.5,y(2)-0.5,0,TJST,'LJT',<#
>THGT,0.75,TANG,270
T(2)=TEXT/'neck',x(10),y(1)+0.5,0,TJST,'RJT',THGT,0.75
T(3)=TEXT/'convertible collar',(x(1)+x(3))/2,<#
>(y(1)+y(2))/2-0.4,0,<#
>TJST,'CJT',THGT,0.75
DISP ALL
#999 END
```

+--+
| |
| Polo collar (CLPOLO) |
| |
+--+

```
<#
<#
DECLARE REAL x(16),y(16)
DECLARE LOCATION P(16)
DECLARE ENTITY L(9),T(7),C(6)
ERTRAP 999
<#
<#
PRINT {''}
PRINT          ------------------------------<#
>--------------------------
PRINT              You can use the icon tool, while<#
> the prompt is "<VAR>".
PRINT          ------------------------------<#
>--------------------------
PRINT          * Measure the length of a half back neck<#
> line and a half front
PRINT            neck line using the icon (MEASURE LENGTH).
PRINT            The unit of the length will be "cm".
PRINT          * After Measuring, please press keys<#
> "Control" and "x" at the
PRINT            same time  to restart the program.
PRINT          ------------------------------<#
>--------------------------
!<VAR>
EXECDF
READ( Enter the length of the half back neckline<#
> please. <unit: cm> ---> )bneck
READ( Enter the length of the half front neckline<#
> please. <unit: cm> ---> )fneck
<#
<#
#10 CONTINUE
PRINT {''}
READ( Which units do you want to use?<#
>     Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'
          cwidth=4
          PRINT default value: collar width = 4cm (depth of fall)
          READ( Enter the width of the polo collar<#
> please. <unit: cm> ---> )cwidth
   OR &unit='IN'.OR.&unit='in'
          cwidth=1.6
          PRINT default value: collar width = 1.6in<#
> (depth of fall)
          READ( Enter the width of the polo collar<#
> please. <unit: in> ---> )cwidth
          cwidth=cwidth*2.54
   ELSE
          PRINT Wrong answer, enter again please.
          GOTO 10
ENDWHEN
<#
```

```
<#
PRINT {''}
PRINT      Select the position for the left lower<#
> corner of the collar please.
PRINT   ------------------------------------<#
>-----------------------------
PRINT   * Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT      at the position you require. Press the <#
>left button.
PRINT   ------------------------------------<#
>-----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
DISP OFF
neck=bneck+fneck
x(1)=x(2)=:x(9)=:x(7)=:x(11)=:X
y(1)=y(13)=:y(5)=:y(14)=:y(3)=:Y
y(2)=y(6)=:y(4)=:y(15)=:y(16)=:y(1)+cwidth*4
x(3)=x(4)=:x(10)=:x(8)=:x(12)=:x(1)+neck*2
x(5)=x(6)=:x(1)+neck
y(7)=y(8)=:y(1)+cwidth*2
y(9)=y(10)=:y(7)+cwidth
y(11)=y(12)=:y(1)+cwidth
x(13)=x(15)=:x(1)+bneck
x(14)=x(16)=:x(3)-bneck
<#
<#
REPEAT 100: i=1,1,i>16
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
<#
<#
L(1)=LINE/P(1),P(2)
L(2)=LINE/P(2),P(4)
L(3)=LINE/P(1),P(3)
L(4)=LINE/P(4),P(3)
L(5)=LINE/P(7),P(8)
L(6)=LINE/P(5),P(6),FONT,'DASH'
L(7)=LINE/P(9),P(10),FONT,'DASH'
L(8)=LINE/P(11),P(12),FONT,'DASH'
L(9)=LINE/x(16)-2.5,y(8)+2.5,0,x(16)+2.5,y(8)-2.5,<#
>0,FONT,'ARROW'
C(1)=CIRCLE/CENTER,P(5),RADIUS,0.4
C(2)=CIRCLE/CENTER,P(6),RADIUS,0.4
C(3)=CIRCLE/CENTER,P(13),RADIUS,0.4
C(4)=CIRCLE/CENTER,P(14),RADIUS,0.4
C(5)=CIRCLE/CENTER,P(15),RADIUS,0.4
C(6)=CIRCLE/CENTER,P(16),RADIUS,0.4
T(1)=TEXT/'polo collar',(x(3)+x(5))/2,y(7)+0.5,0,<#
>TJST,'CJT',THGT,0.75
T(2)=TEXT/'fold line',x(1)+4,y(7)+0.1,0,<#
>TJST,'LJT',THGT,0.75
T(3)=TEXT/'roll line',x(1)+4,y(9)+0.1,0,<#
>TJST,'LJT',THGT,0.75
```

```
T(4)=TEXT/'roll line',x(1)+4,y(11)+0.1,0,<#
>TJST,'LJT',THGT,0.75
T(5)=TEXT/'centre front',x(5)-0.2,y(7),0,<#
>TJST,'CJT',THGT,0.75,TANG,90
T(6)=TEXT/'centre back',x(1)+0.5,y(7),0,<#
>TJST,'CJT',THGT,0.75,TANG,270
T(7)=TEXT/'centre back',x(3)-0.5,y(7),0,<#
>TJST,'CJT',THGT,0.75,TANG,90
DISP ALL
#999 END
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|                          Shirt cuffs (CUFFSH)                         |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE REAL x(8),y(8)
DECLARE LOCATION P(8)
DECLARE ENTITY L(7),T(3)
ERTRAP 999
<#
<#
#10 CONTINUE
READ( Which units do you want to use? <#
>   Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'.OR.&unit='IN'.OR.&unit='in'
        CONTINUE
  ELSE
        PRINT Wrong answer, enter again please.
        GOTO 10
ENDWHEN
PRINT {''}
WHEN &unit='CM'.OR.&unit='cm'
PRINT                    Shirt cuff size of women <unit: cm>
ELSE
PRINT                    Shirt cuff size of women <unit: in>
ENDWHEN
PRINT ----------------------------------<#
>------------------------------
PRINT                8    10    12    14    16    18    <#
>20    22    24    26    28    30
PRINT ----------------------------------<#
>------------------------------
WHEN &unit='CM'.OR.&unit='cm'
PRINT cuff(cm)  21    21    21.5 21.5 22    22.5 <#
>23    23.5 24    24.5 25    25.5
ELSE
PRINT cuff(in)    8.3   8.3   8.5   8.5   8.7   8.9   <#
>9.1   9.3   9.4   9.6   9.8 10
ENDWHEN
PRINT ----------------------------------<#
>------------------------------
WHEN &unit='CM'.OR.&unit='cm'
        cuffdept=5
        underwrap=1.5
   READ( Enter the cuff size please. <unit: cm> ---> )cuff
   PRINT {''}
   PRINT default value: cuff depth = 5cm
   READ( Enter the depth of the cuffs please.<#
> <unit: cm> ---> )cuffdepth
   PRINT default value: underwrap length = 1.5cm
   READ( Enter the length of the underwrap please. <#
><unit: cm> ---> )underwrap
  ELSE
        cuffdept=2
        underwrap=0.6
   READ( Enter the cuff size please. <unit: in> ---> )cuff
```

```
   PRINT {''}
   PRINT default value: cuff depth = 2in
   READ( Enter the depth of the cuffs please.<#
> <unit: in> ---> )cuffdepth
   PRINT default value: underwrap length = 0.6in
   READ( Enter the length of the underwrap please. <#
><unit: in> ---> )underwrap
         cuff=cuff*2.54
         cuffdepth=cuffdepth*2.54
         underwrap=underwrap*2.54
ENDWHEN
PRINT {''}
PRINT  Select the position for the left lower corner<#
> of the cuff pattern please.
PRINT    ------------------------------------<#
>----------------------------
PRINT    * Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT      at the position you require. Press <#
>the left button.
PRINT    ------------------------------------<#
>----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
DISP OFF
x(1)=x(2)=:x(7)=:X
y(1)=y(3)=:y(5)=:Y
y(2)=y(4)=:y(6)=:y(1)+cuffdepth*2
x(3)=x(4)=:x(8)=:x(1)+cuff+underwrap
x(5)=x(6)=:x(3)-underwrap
y(7)=y(8)=:y(1)+cuffdepth
REPEAT 100: i=1,1,i>8
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
L(1)=LINE/P(1),P(2)
L(2)=LINE/P(2),P(4)
L(3)=LINE/P(4),P(3)
L(4)=LINE/P(1),P(3)
L(5)=LINE/P(5),P(6),FONT,'DASH'
L(6)=LINE/P(7),P(8),FONT,'DASH'
L(7)=LINE/x(1)+5,y(7)+2.5,0,x(1)+5,y(7)-2.5,0,FONT,'ARROW'
T(1)=TEXT/'fold line',(x(1)+x(5))/2,y(7)+0.1,0,<#
>TJST,'CJT',THGT,0.75
T(2)=TEXT/'cuff',(x(1)+x(5))/2,y(1)+0.5,0,<#
>TJST,'CJT',THGT,0.75
T(3)=TEXT/'underwrap',x(3)-0.1,y(8),0,<#
>TJST,'CJT',THGT,0.5,TANG,90
DISP ALL
#999 END
```

+--+
| |
| Frilled cuffs (CUFFFR) |
| |
+--+

```
<#
<#
DECLARE ENTITY C(2)
ERTRAP 999
<#
<#
PRINT {''}
PRINT          ------------------------------------<#
>--------------------------
PRINT                You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ------------------------------------<#
>--------------------------
PRINT          * Measure the length of the cuff at<#
> the sleeve pattern using
PRINT            the icon (MEASURE LENGTH).
PRINT            The unit of the length will be "cm".
PRINT          * After Measuring, please press keys <#
>"Control" and "x" at the
PRINT            same time  to restart the program.
PRINT          ------------------------------------<#
>--------------------------
!<VAR>
EXECDF
READ( Enter the length of the cuff please.<#
> <unit; cm> ---> )cuff
<#
<#
#10 CONTINUE
PRINT {''}
READ( Which units do you want to use? <#
>    Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'
         cuffdept=5
    PRINT default value: frilled cuff depth = 5cm
    READ( Enter the depth of the frilled cuff please. <#
><unit: cm> ---> )cuffdepth
  OR &unit='IN'.OR.&unit='in'
         cuffdept=2
    PRINT default value: frilled cuff depth = 2in
    READ( Enter the depth of the frilled cuff please. <#
><unit: in> ---> )cuffdepth
         cuffdepth=cuffdepth*2.54
  ELSE
    PRINT Wrong answer, enter again please.
    GOTO 10
ENDWHEN
<#
<#
PRINT {''}
PRINT  Select the position for the left lower corner<#
> of the cuff pattern please.
PRINT    ------------------------------------<#
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|                        Frilled cuffs (CUFFFR)                         |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE ENTITY C(2)
ERTRAP 999
<#
<#
PRINT {''}
PRINT          ------------------------------------<#
>--------------------------
PRINT                You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ------------------------------------<#
>--------------------------
PRINT          * Measure the length of the cuff at<#
> the sleeve pattern using
PRINT            the icon (MEASURE LENGTH).
PRINT            The unit of the length will be "cm".
PRINT          * After Measuring, please press keys <#
>"Control" and "x" at the
PRINT            same time  to restart the program.
PRINT          ------------------------------------<#
>--------------------------
!<VAR>
EXECDF
READ( Enter the length of the cuff please.<#
> <unit; cm> ---> )cuff
<#
<#
#10 CONTINUE
PRINT {''}
READ( Which units do you want to use? <#
>    Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'
          cuffdept=5
    PRINT default value: frilled cuff depth = 5cm
    READ( Enter the depth of the frilled cuff please. <#
><unit: cm> ---> )cuffdepth
  OR &unit='IN'.OR.&unit='in'
          cuffdept=2
    PRINT default value: frilled cuff depth = 2in
    READ( Enter the depth of the frilled cuff please. <#
><unit: in> ---> )cuffdepth
          cuffdepth=cuffdepth*2.54
  ELSE
    PRINT Wrong answer, enter again please.
    GOTO 10
ENDWHEN
<#
<#
PRINT {''}
PRINT  Select the position for the left lower corner<#
> of the cuff pattern please.
PRINT    ------------------------------------<#
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|                        Waistband (WAISTBD)                            |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE REAL x(14),y(14)
DECLARE LOCATION P(14)
DECLARE ENTITY L(7),T(3)
ERTRAP 999
<#
<#
#10 CONTINUE
READ( Which units do you want to use?    <#
> Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'.OR.&unit='IN'.OR.&unit='in'
        CONTINUE
  ELSE
        PRINT Wrong answer, enter again please.
        GOTO 10
ENDWHEN
PRINT {''}
WHEN &unit='CM'.OR.&unit='cm'
PRINT                          Waist size of women <unit: cm>
ELSE
PRINT                          Waist size of women <unit: in>
ENDWHEN
PRINT    ----------------------------------<#
>----------------------------------
PRINT                  8    10    12    14    16    18 <#
>  20    22    24    26    28    30
PRINT    ----------------------------------<#
>----------------------------------
WHEN &unit='CM'.OR.&unit='cm'
PRINT    waist(cm) 60    64    68    72    77    82 <#
>  87    92    97   102   107   112
ELSE
PRINT    waist(in) 23.6 25.2 26.8 28.3 30.3 32.3<#
> 34.3 36.2 38.2 40.2 42.1 44.1
ENDWHEN
PRINT    ----------------------------------<#
>----------------------------------
WHEN &unit='CM'.OR.&unit='cm'
        underwrap=4
        depth=3
 READ( Enter the waist length please. <unit: cm> --->)waist
   PRINT {''}
   PRINT default value: waistband depth = 3cm
   READ( Enter the depth of the waistband please.<#
> <unit: cm> ---> )depth
   PRINT default value: underwrap length = 4cm
   READ( Enter the length of the underwrap please. <#
><unit: cm> ---> )underwrap
ELSE
        underwrap=1.7
        depth=1.2
   READ( Enter the waist length please. <unit: in> --->)waist
```

```
   PRINT {''}
   PRINT default value: waistband depth = 1.2in
   READ( Enter the depth of the waistband please.<#
> <unit:in> ---> )depth
   PRINT default value: underwrap length = 1.7in
   READ( Enter the length of the underwrap please. <#
><unit: in> ---> )underwrap
        waist=waist*2.54
        depth=depth*2.54
        underwrap=underwrap*2.54
ENDWHEN
PRINT {''}
PRINT   Select the position for the left lower corner <#
>of the waistband please.
PRINT   ----------------------------------------<#
>------------------------
PRINT   * Move the mouse on the pad to place the cursor<#
> on the graphic window
PRINT     at the position you require. Press the<#
> left button.
PRINT   ----------------------------------------<#
>------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
PRINT Working, just a moment please.
<#
<#
DISP OFF
x(1)=x(2)=:x(7)=:X
y(1)=y(3)=:y(5)=:y(9)=:y(11)=:y(13)=:Y
y(2)=y(6)=:y(10)=:y(12)=:y(14)=:y(4)=:y(1)+depth*2
x(3)=x(4)=:x(8)=:x(1)+waist+underwrap
x(5)=x(6)=:x(1)+underwrap
y(7)=y(8)=:y(1)+depth
x(9)=x(10)=:x(5)+waist/4
x(11)=x(12)=:x(5)+waist/2
x(13)=x(14)=:x(5)+waist*3/4
REPEAT 100: i=1,1,i>14
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
L(1)=LINE/P(1),P(2)
L(2)=LINE/P(2),P(4)
L(3)=LINE/P(4),P(3)
L(4)=LINE/P(1),P(3)
L(5)=LINE/P(5),P(6),FONT,'DASH'
L(6)=LINE/P(7),P(8),FONT,'DASH'
L(7)=LINE/x(9),y(7)-0.5,0,x(13),y(7)-0.5,0,FONT,'ARROW'
T(1)=TEXT/'fold line',x(11),y(7)+0.1,0,TJST,'CJT',THGT,0.75
T(2)=TEXT/'waistband',x(11),y(1)+1,0,TJST,'CJT',THGT,0.75
T(3)=TEXT/'underwrap',x(1)+0.1,y(7),0,<#
>TJST,'CJT',THGT,0.5,TANG,270
DISP ALL
#999 END
```

Flap pocket (PCFLAP)

```
<#
<#
DECLARE REAL x(26),y(26)
DECLARE ENTITY P(28),L(23),C(8),1(10),cp1(12),cp2(6)
ERTRAP 999
PRINT {''}
PRINT Enter 2 locations which will be 2 UPPER ENDS<#
> of the flap pocket please.
DIGI (MLOC,NOWAIT)xa,ya,za
PRINT {''}
DIGI (MLOC,NOWAIT)xb,yb,zb
#10 CONTINUE
PRINT {''}
READ( Which units do you want to use? <#
>   Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'
READ( Enter the depth of the flap please.  unit; cm --->)flap
  OR &unit='IN'.OR.&unit='in'
READ( Enter the depth of the flap please.  unit; in --->)flap
     flap=flap*2.54
  ELSE
PRINT Wrong answer, enter again please.
GOTO 10
ENDWHEN
PRINT Enter the location which will be the BOTTOM<#
> of the pocket please.
DIGI (MLOC,NOWAIT)x2,y2,z2
PRINT {''}
PRINT   Select the position for the left lower corner<#
> of pocket pattern please.
PRINT   ----------------------------------<#
>----------------------------
PRINT   * Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT     at the position you require. Press the <#
>left button.
PRINT   ----------------------------------<#
>----------------------------
DIGI (MLOC,NOWAIT)x3,y3,z3
PRINT {''}
PRINT Working, just a moment please.
DISP OFF
WHEN xa<=xb
          x(1)=xa
          x(2)=xb
          y(1)=ya
          y(2)=yb
  ELSE
          x(1)=xb
          x(2)=xa
          y(1)=yb
          y(2)=ya
ENDWHEN
```

```
P(1)=POINT/x(1),y(1),0
P(2)=POINT/x(2),y(2),0
pwidth=SQRT((x(2)-x(1))**2+(y(2)-y(1))**2)
L(1)=LINE/P(1),P(2)
1(1)=LINE/P(1),PERP,L(1),LENGTH,-flap+pwidth/10
P(3)=POINT/P(1),DISTAN,flap-pwidth/10,1(1)
1(2)=LINE/P(2),PERP,L(1),LENGTH,-flap+pwidth/10
P(4)=POINT/P(2),DISTAN,flap-pwidth/10,1(2)
$E=POINT/P(1),DISTAN,pwidth/10,L(1)
1(3)=LINE/$E,PERP,L(1),LENGTH,-flap+1
P(5)=POINT/$E,DISTAN,flap,1(3)
P(11)=POINT/$E,DISTAN,flap+1,1(3)
DELETE $E
$E=POINT/P(2),DISTAN,-pwidth/10,L(1)
1(4)=LINE/$E,PERP,L(1),LENGTH,-flap+1
P(6)=POINT/$E,DISTAN,flap,1(4)
P(12)=POINT/$E,DISTAN,flap+1,1(4)
DELETE $E
$E=POINT/P(1),DISTAN,-1,1(1)
1(5)=LINE/$E,ATANGL,180,L(1),LENGTH,1
P(7)=POINT/$E,DISTAN,1,1(5)
1(6)=LINE/$E,PARLEL,L(1),LENGTH,pwidth+1
P(8)=POINT/$E,DISTAN,pwidth+1,1(6)
DELETE $E
1(7)=LINE/P(7),PERP,1(5),LENGTH,flap-pwidth/10+1
P(9)=POINT/P(7),DISTAN,flap-pwidth/10+1,1(7)
1(8)=LINE/P(8),PERP,1(6),LENGTH,-flap+pwidth/10-1
P(10)=POINT/P(8),DISTAN,flap-pwidth/10+1,1(8)
L(2)=LINE/P(2),P(4)
L(3)=LINE/P(5),P(6)
L(4)=LINE/P(1),P(3)
C(1)=CIRCLE/RADIUS,pwidth/10,RIGHT,P(3),P(5),<#
>PGO,P(3),PEND,P(5)
C(2)=CIRCLE/RADIUS,pwidth/10,LEFT,P(6),P(4),<#
>PGO,P(6),PEND,P(4)
P(13)=POINT/P(1),DISTAN,-1.5,1(1)
P(14)=POINT/P(2),DISTAN,-1.5,1(2)
P(15)=POINT/P(1),DISTAN,-3,1(1)
P(16)=POINT/P(2),DISTAN,-3,1(2)
x(23)=x(1)
x(24)=x(2)
x(17)=x(19)=:x(1)-1
y(17)=y(1)+1-(y(2)-y(1))/(x(2)-x(1))
x(18)=x(20)=:x(2)+1
y(18)=y(2)+1+(y(2)-y(1))/(x(2)-x(1))
y(19)=y(20)=:y(23)=:y(24)=:y2+pwidth/5
x(21)=x(25)=:x(1)+pwidth/5
x(22)=x(26)=:x(2)-pwidth/5
y(21)=y(22)=:y2-1
y(25)=y(26)=:y2
REPEAT 100: i=17,1,i>26
P(i)=POINT/x(i),y(i),0
#100 CONTINUE
1(9)=LINE/P(17),PARLEL,1(1),LENGTH,-3
P(27)=POINT/P(17),DISTAN,3,1(9)
1(10)=LINE/P(18),PARLEL,1(2),LENGTH,-3
P(28)=POINT/P(18),DISTAN,3,1(10)
DELETE 10,1(1)
```

```
L(5)=LINE/P(7),P(8)
L(6)=LINE/P(8),P(10)
L(7)=LINE/P(11),P(12)
L(8)=LINE/P(7),P(9)
C(3)=CIRCLE/RADIUS,pwidth/10+1,RIGHT,P(9),P(11),<#
>PGO,P(9),PEND,P(11)
C(4)=CIRCLE/RADIUS,pwidth/10+1,LEFT,P(12),P(10),<#
>PGO,P(12),PEND,P(10)
$L=LINE/(x(17)+x(18))/2,(y(17)+y(18))/2-2,0,<#
>(x(17)+x(18))/2,y(25)+2,0,FONT,'ARROW'
@V=VECT(x3-x(19),y3-y(21),0)
TRANSL 28,P(1),INDIRV,@V
TRANSL $L,INDIRV,@V
L(9)=LINE/P(1),P(2)
L(10)=LINE/P(2),P(24)
L(11)=LINE/P(25),P(26)
L(12)=LINE/P(1),P(23)
L(13)=LINE/P(13),P(14)
L(14)=LINE/P(14),P(16)
L(15)=LINE/P(15),P(16)
L(16)=LINE/P(15),P(13)
L(17)=LINE/P(17),P(19)
L(18)=LINE/P(21),P(22)
L(19)=LINE/P(18),P(20)
L(20)=LINE/P(18),P(28)
L(21)=LINE/P(27),P(28)
L(22)=LINE/P(27),P(17)
L(23)=LINE/P(17),P(18)
C(5)=CIRCLE/RADIUS,pwidth/5,RIGHT,P(23),P(25),<#
>PGO,P(23),PEND,P(25)
C(6)=CIRCLE/RADIUS,pwidth/5,LEFT,P(26),P(24),<#
>PGO,P(26),PEND,P(24)
C(7)=CIRCLE/RADIUS,pwidth/5+1,RIGHT,P(19),P(21),<#
>PGO,P(19),PEND,P(21)
C(8)=CIRCLE/RADIUS,pwidth/5+1,LEFT,P(22),P(20),<#
>PGO,P(22),PEND,P(20)
@v1=VECT((x(18)-x(17))*1.5,0,0)
@v2=VECT(x3-x(19)+(x(18)-x(17))*1.5,y3-y(21)+flap+5,0)
TRANSL(COPY,1,cp1) 4,L(9),INDIRV,@v1
TRANSL(COPY,1,cp1(5)) 3,L(17),INDIRV,@v1
TRANSL(COPY,1,cp1(8)) 4,C(5),INDIRV,@v1
TRANSL(COPY,1,cp1(12)) $L,INDIRV,@v1
TRANSL L(23),INDIRV,@v1
TRANSL(COPY,1,cp2) 4,L(1),INDIRV,@v2
TRANSL(COPY,1,cp2(5)) 2,C(1),INDIRV,@v2
TRANSL 4,L(5),INDIRV,@v2
TRANSL 2,C(3),INDIRV,@v2
DELETE 28,P(1)
DISP ALL
#999 END
```

Welt pocket (PCWELT)

```
<#
<#
DECLARE REAL x(24),y(24)
DECLARE ENTITY P(24),L(23),C(4),1(8),cp1(12),cp2(4)
ERTRAP 999
PRINT {''}
PRINT Enter 2 locations which will be 2 UPPER ENDS<#
> of the welt pocket please.
DIGI (MLOC,NOWAIT)xa,ya,za
PRINT {''}
DIGI (MLOC,NOWAIT)xb,yb,zb
#10 CONTINUE
PRINT {''}
READ( Which units do you want to use?  <#
>  Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'
READ( Enter the depth of the welt please.  unit; cm --->)welt
  OR &unit='IN'.OR.&unit='in'
READ( Enter the depth of the welt please.  unit; in --->)welt
     welt=welt*2.54
  ELSE
        PRINT Wrong answer, enter again please.
        GOTO 10
ENDWHEN
PRINT Enter the location which will be the BOTTOM of<#
> the pocket please.
DIGI (MLOC,NOWAIT)x2,y2,z2
PRINT {''}
PRINT   Select the position for the left lower<#
> corner of pocket pattern please.
PRINT   ----------------------------------<#
>----------------------------
PRINT   * Move the mouse on the pad to place<#
> the cursor on the graphic window
PRINT     at the position you require. Press <#
>the left button.
PRINT   ----------------------------------<#
>----------------------------
DIGI (MLOC,NOWAIT)x3,y3,z3
PRINT {''}
PRINT Working, just a moment please.
DISP OFF
WHEN xa<=xb
        x(1)=xa
        x(2)=xb
        y(1)=ya
        y(2)=yb
  ELSE
        x(1)=xb
        x(2)=xa
        y(1)=yb
        y(2)=ya
ENDWHEN
```

```
P(1)=POINT/x(1),y(1),0
P(2)=POINT/x(2),y(2),0
pwidth=SQRT((x(2)-x(1))**2+(y(2)-y(1))**2)
L(1)=LINE/P(1),P(2)
1(1)=LINE/P(1),PERP,L(1),LENGTH,-welt
P(3)=POINT/P(1),DISTAN,welt,1(1)
1(2)=LINE/P(2),PERP,L(1),LENGTH,-welt
P(4)=POINT/P(2),DISTAN,welt,1(2)
$E=POINT/P(1),DISTAN,-1,1(1)
1(3)=LINE/$E,ATANGL,180,L(1),LENGTH,1
P(5)=POINT/$E,DISTAN,1,1(3)
1(4)=LINE/$E,PARLEL,L(1),LENGTH,pwidth+1
P(6)=POINT/$E,DISTAN,pwidth+1,1(4)
DELETE $E
1(5)=LINE/P(5),PERP,1(3),LENGTH,welt+2
P(7)=POINT/P(5),DISTAN,welt+2,1(5)
1(6)=LINE/P(6),PERP,1(4),LENGTH,-welt-2
P(8)=POINT/P(6),DISTAN,welt+2,1(6)
L(2)=LINE/P(2),P(4)
L(3)=LINE/P(3),P(4)
L(4)=LINE/P(1),P(3)
P(9)=POINT/P(1),DISTAN,-1.5,1(1)
P(10)=POINT/P(2),DISTAN,-1.5,1(2)
P(11)=POINT/P(1),DISTAN,-3,1(1)
P(12)=POINT/P(2),DISTAN,-3,1(2)
x(19)=x(1)
x(20)=x(2)
x(13)=x(15)=:x(1)-1
y(13)=y(1)+1-(y(2)-y(1))/(x(2)-x(1))
x(14)=x(16)=:x(2)+1
y(14)=y(2)+1+(y(2)-y(1))/(x(2)-x(1))
y(15)=y(19)=:y(20)=:y(16)=:y2+pwidth/5
x(17)=x(21)=:x(1)+pwidth/5
x(18)=x(22)=:x(2)-pwidth/5
y(17)=y(18)=:y2-1
y(21)=y(22)=:y2
REPEAT 100: i=13,1,i>22
P(i)=POINT/x(i),y(i),0
#100 CONTINUE
1(7)=LINE/P(13),PARLEL,1(1),LENGTH,-3
P(23)=POINT/P(13),DISTAN,3,1(7)
1(8)=LINE/P(14),PARLEL,1(2),LENGTH,-3
P(24)=POINT/P(14),DISTAN,3,1(8)
DELETE 8,1(1)
L(5)=LINE/P(5),P(6)
L(6)=LINE/P(6),P(8)
L(7)=LINE/P(7),P(8)
L(8)=LINE/P(5),P(7)
$L=LINE/(x(1)+x(2))/2,(y(1)+y(2))/2-2,0,(x(1)+x(2))/2,<#
>y2+2,0,FONT,'ARROW'
@V=VECT(x3-x(15),y3-y(17),0)
TRANSL 24,P(1),INDIRV,@V
TRANSL $L, INDIRV,@V
L(9)=LINE/P(1),P(2)
L(10)=LINE/P(2),P(20)
L(11)=LINE/P(21),P(22)
L(12)=LINE/P(1),P(19)
L(13)=LINE/P(9),P(10)
```

```
L(14)=LINE/P(10),P(12)
L(15)=LINE/P(11),P(12)
L(16)=LINE/P(9),P(11)
L(17)=LINE/P(13),P(15)
L(18)=LINE/P(17),P(18)
L(19)=LINE/P(14),P(16)
L(20)=LINE/P(14),P(24)
L(21)=LINE/P(23),P(24)
L(22)=LINE/P(23),P(13)
L(23)=LINE/P(13),P(14)
C(1)=CIRCLE/RADIUS,pwidth/5,RIGHT,P(19),P(21),<#
>PGO,P(19),PEND,P(21)
C(2)=CIRCLE/RADIUS,pwidth/5,LEFT,P(22),P(20),<#
>PGO,P(22),PEND,P(20)
C(3)=CIRCLE/RADIUS,pwidth/5+1,RIGHT,P(15),P(17),<#
>PGO,P(15),PEND,P(17)
C(4)=CIRCLE/RADIUS,pwidth/5+1,LEFT,P(18),P(16),<#
>PGO,P(18),PEND,P(16)
@v1=VECT((x(16)-x(15))*1.5,0,0)
@v2=VECT(x3-x(15)+(x(16)-x(15))*1.5,y3-y(17)+welt+5,0)
TRANSL(COPY,1,cp1) 4,L(9),INDIRV,@v1
TRANSL(COPY,1,cp1(5)) 3,L(17),INDIRV,@v1
TRANSL(COPY,1,cp1(8)) 4,C(1),INDIRV,@v1
TRANSL(COPY,1,cp1(12)) $L,INDIRV,@v1
TRANSL L(23),INDIRV,@v1
TRANSL(COPY,1,cp2) 4,L(1),INDIRV,@v2
TRANSL 4,L(5),INDIRV,@v2
DELETE 24,P(1)
DISP ALL
#999 END
```

Slashed pocket (PCSLASH)

```
<#
<#
DECLARE REAL x(16),y(16)
DECLARE ENTITY P(24),C(4),L(20),cp1(1),cp2(12),1(4)
PRINT {''}
PRINT Enter 2 locations which will be 2 ENDS<#
> of the slash pocket OPENING please.
DIGI (MLOC,NOWAIT)xa,ya,za
PRINT {''}
DIGI (MLOC,NOWAIT)xb,yb,zb
PRINT {''}
PRINT Enter the location which will be the BOTTOM<#
> of the pocket please.
DIGI (MLOC,NOWAIT)x2,y2,z2
PRINT {''}
READ( Enter the depth of one lip of the opening<#
> please.  <unit; cm>  ---> )lip
PRINT {''}
PRINT   Select the position for the left lower<#.
> corner of pocket pattern please.
PRINT   ----------------------------------<#
>----------------------------------
PRINT   * Move the mouse on the pad to place <#
>the cursor on the graphic window
PRINT     at the position you require. Press the<#
> left button.
PRINT   ----------------------------------<#
>----------------------------------
DIGI (MLOC,NOWAIT)x3,y3,z3
PRINT {''}
PRINT Working, just a moment please.
DISP OFF
WHEN xa>=xb
      x(1)=xa
      x(2)=xb
      y(1)=ya
      y(2)=yb
  ELSE
      x(1)=xb
      x(2)=xa
      y(1)=yb
      y(2)=ya
ENDWHEN
P(1)=POINT/x(1),y(1),0
P(2)=POINT/x(2),y(2),0
L(1)=LINE/P(1),P(2)
1(1)=LINE/P(1),PERP,L(1),LENGTH,-lip*7-1
1(2)=LINE/P(1),PERP,L(1),LENGTH,lip
1(3)=LINE/P(2),PERP,L(1),LENGTH,lip
1(4)=LINE/P(2),PERP,L(1),LENGTH,-lip*7-1
P(3)=POINT/P(1),DISTAN,lip,1(1)
P(4)=POINT/P(2),DISTAN,lip,1(4)
P(5)=POINT/P(1),DISTAN,lip,1(2)
```

```
P(6)=POINT/P(2),DISTAN,lip,l(3)
L(2)=LINE/P(3),P(5)
L(3)=LINE/P(5),P(6)
L(4)=LINE/P(6),P(4)
L(5)=LINE/P(4),P(3)
x(7)=x(1)
y(7)=y(8)=:y2+(x(1)-x(2))/5
x(8)=x(2)
x(9)=x(1)-(x(1)-x(2))/5
x(10)=x(2)+(x(1)-x(2))/5
y(9)=y(10)=:y2
REPEAT 100: i=7,1,i>10
P(i)=POINT/x(i),y(i),0
#100 CONTINUE
L(6)=LINE/P(5),P(7)
L(7)=LINE/P(6),P(8)
L(8)=LINE/P(10),P(9)
C(1)=CIRCLE/RADIUS,(x(1)-x(2))/5,RIGHT,P(8),P(10),<#
>PGO,P(8),PEND,P(10)
C(2)=CIRCLE/RADIUS,(x(1)-x(2))/5,LEFT,P(9),P(7),<#
>PGO,P(9),PEND,P(7)
x(11)=x(1)+1
y(11)=y(1)+1-lip+(y(1)-y(2))/(x(1)-x(2))
x(12)=x(2)-1
y(12)=y(2)+1-lip-(y(1)-y(2))/(x(1)-x(2))
x(13)=x(7)+1
x(14)=x(8)-1
y(13)=y(14)=:y(7)
x(15)=x(9)
x(16)=x(10)
y(15)=y(16)=:y(9)-1
REPEAT 200: i=11,1,i>16
P(i)=POINT/x(i),y(i),0
#200 CONTINUE
$E3=LINE/P(11),ATANGL,180,L(2),LENGTH,lip*8
P(17)=POINT/P(11),DISTAN,lip*8,$E3
$E4=LINE/P(12),PARLEL,L(4),LENGTH,lip*8
P(18)=POINT/P(12),DISTAN,lip*8,$E4
P(19)=POINT/P(1),DISTAN,lip*5,l(1)
P(20)=POINT/P(1),DISTAN,lip*4,l(1)
P(21)=POINT/P(1),DISTAN,lip*3,l(1)
P(22)=POINT/P(2),DISTAN,lip*5,l(4)
P(23)=POINT/P(2),DISTAN,lip*4,l(4)
P(24)=POINT/P(2),DISTAN,lip*3,l(4)
$L=LINE/(x(1)+x(2))/2,(y(1)+y(2))/2-2,0,<#
>(x(1)+x(2))/2,y(9)+2,0,FONT,'ARROW'
@V=VECT(x3-x(14),y3-y(15),0)
TRANSL(COPY,1,cp1) L(3),INDIRV,@V
TRANSL 3,L(6),INDIRV,@V
TRANSL 2,C(1),INDIRV,@V
TRANSL 14,P(11),INDIRV,@V
TRANSL $L,INDIRV,@V
L(9)=LINE/P(11),P(13)
L(10)=LINE/P(15),P(16)
L(11)=LINE/P(12),P(14)
L(12)=LINE/P(12),P(18)
L(13)=LINE/P(18),P(17)
L(14)=LINE/P(17),P(11)
```

```
L(15)=LINE/P(19),P(22)
L(16)=LINE/P(20),P(23)
L(17)=LINE/P(21),P(24)
L(18)=LINE/P(19),P(21)
L(19)=LINE/P(24),P(22)
L(20)=LINE/P(11),P(12)
C(3)=CIRCLE/RADIUS,(x(1)-x(2))/5+1.5,RIGHT,<#
>P(14),P(16),PGO,P(14),PEND,P(16)
C(4)=CIRCLE/RADIUS,(x(1)-x(2))/5+1.5,LEFT,<#
>P(15),P(13),PGO,P(15),PEND,P(13)
@v=VECT((x(1)-x(2))*1.5,0,0)
TRANSL(COPY,1,cp2) cp1,INDIRV,@v
TRANSL(COPY,1,cp2(2)) 3,L(6),INDIRV,@v
TRANSL(COPY,1,cp2(5)) 2,C(1),INDIRV,@v
TRANSL(COPY,1,cp2(7))3,L(9),INDIRV,@v
TRANSL(COPY,1,cp2(10)) 2,C(3),INDIRV,@v
TRANSL(COPY,1,cp2(12)) $L,INDIRV,@v
TRANSL L(20),INDIRV,@v
DELETE 24,P(1)
DELETE 4,1(1)
DELETE $E1,$E2,$E3,$E4
DISP ALL
```

PCPA1LK (Linkfile: PCPA1, PCPASIZE)

```
MAIN PCPA1
IMAGE PCPA1IMAGE
LINKM PCPASIZE
ENDLINK
```

PCPA2LK (Linkfile: PCPA2, PCPASIZE)

```
MAIN PCPA2
IMAGE PCPA2IMAGE
LINKM PCPASIZE
ENDLINK
```

PCPA3LK (Linkfile: PCPA3, PCPASIZE)

```
MAIN PCPA3
IMAGE PCPA3IMAGE
LINKM PCPASIZE
ENDLINK
```

PCPA4LK (Linkfile: PCPA4, PCPASIZE)

```
MAIN PCPA4
IMAGE PCPA4IMAGE
LINKM PCPASIZE
ENDLINK
```

```
+-----------------------------------------------------------+
|                                                           |
|          Patch pocket size (PCPASIZE) procedure           |
|                                                           |
+-----------------------------------------------------------+
<#
<#
PROC PCPASIZE(&flap,&loc,&corner,&fd,pwidth,pdepth,fdepth,<#
>pxl,pyl,pxl,pyl,pxs,pys,px3,py3)
DECLARE REAL px(3),py(3),pz(3)
ERTRAP 999
<#
<#
#10 CONTINUE
PRINT {''}
&flap=''
PRINT            Which patch pocket do you want?
PRINT    ------------------------------------------
PRINT    Patch pocket with no flap ----------------- 1
PRINT    Patch pocket with self-flap --------------- 2
PRINT    Patch pocket with separate flap ----------- 3
PRINT    ------------------------------------------
READ(          Enter the number please. ---> )&flap
WHEN &flap='1'.OR.&flap='2'.OR.&flap='3'
         CONTINUE
  ELSE
         PRINT Wrong answer, enter again please.
         GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
PRINT {''}
&loc=''
PRINT Which way do you want to decide size and <#
>location of the pocket?
PRINT          ---------------------------------------
PRINT          The width & depth of the pocket
PRINT             and the location of one corner ------- 1
PRINT          Diagonal corners of the pocket ---------- 2
PRINT          ---------------------------------------
READ(               Enter the number please. ---> )&loc
WHEN &loc='1'
    #30 CONTINUE
    PRINT {''}
    &corner=''
    PRINT         Which corner do you want to digitize?
    PRINT    ---------------------------------------
    PRINT    Upper left corner of the pocket ---------- 1
    PRINT    Upper right corner of the pocket --------- 2
    PRINT    Lower left corner of the pocket ---------- 3
    PRINT    Lower right corner of the pocket --------- 4
    PRINT    ---------------------------------------
    READ(          Enter the number please. ---> )&corner
    WHEN &corner='1'.OR.&corner='2'.OR.&corner='3'.OR.<#
>&corner='4'
        CONTINUE
      ELSE
```

```
      PRINT Wrong answer, enter again please.
      GOTO 30
   ENDWHEN
OR &loc='2'
    WHEN &flap='2'.OR.&flap='3'
      #40 CONTINUE
      PRINT {''}
      &fd=''
      PRINT  Which way do you want to decide the flap depth?
      PRINT   ------------------------------------
      PRINT   Entering the depth of flap ---------------- 1
      PRINT   Digitize the lowest position of the flap -- 2
      PRINT   ------------------------------------
      READ(          Enter the number please. ---> )&fd
      WHEN &fd='1'.OR.&fd='2'
            CONTINUE
         ELSE
            PRINT Wrong answer, enter again please.
            GOTO 40
      ENDWHEN
    ENDWHEN
  ELSE
    PRINT Wrong answer, enter again please.
    GOTO 20
ENDWHEN
IF(&loc='2'.AND.&fd='2') GOTO 60
IF(&loc='3'.AND.&fd='2') GOTO 60
IF(&flap='1'.AND.&loc='2') GOTO 60
#50 CONTINUE
PRINT {''}
READ( Which units do you want to use? <#
>    Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'.OR.&unit='IN'.OR.&unit='in'
    CONTINUE
  ELSE
    PRINT Wrong answer, enter again please.
    GOTO 50
ENDWHEN
WHEN &loc='1'
 READ( Enter the width of the pocket please.  ---> )pwidth
 READ( Enter the depth of the pocket please.  ---> )pdepth
 IF(&flap='2'.OR.&flap='3') READ( Enter the depth of the<#
> flap please.  ---> )fdepth
  OR &loc='2'
    WHEN &flap='2'.OR.&flap='3'
        IF (&fd='1') READ( Enter the depth of the<#
> flap please.  ---> )fdepth
    ENDWHEN
ENDWHEN
WHEN &unit='IN'.OR.&unit='in'
    pwidth=pwidth*2.54
    pdepth=pdepth*2.54
    fdepth=fdepth*2.54
ENDWHEN
#60 CONTINUE
PRINT {''}
WHEN &flap='1'
    WHEN &loc='1'
```

```
        DIGI (MLOC,NOWAIT,'',' Digitize the corner<#
> position of the pocket please. :')px(1),py(1),pz(1)
        PRINT {''}
      OR &loc='2'
        DIGI (MLOC,NOWAIT,2,n,'',' Digitize 2 diagonal<#
> corner position of the pocket please. :')px(1),py(1),pz(1)
        PRINT {''}
                  WHEN px(1)>=px(2)
                       pxl=px(1)
                       pxs=px(2)
                  ELSE
                       pxl=px(2)
                       pxs=px(1)
                  ENDWHEN
                  WHEN py(1)>=py(2)
                       pyl=py(1)
                       pys=py(2)
                  ELSE
                       pyl=py(2)
                       pys=py(1)
                  ENDWHEN
    ENDWHEN
   OR &flap='2'.OR.&flap='3'
       WHEN &loc='1'
            DIGI (MLOC,NOWAIT,'',' Digitize the corner<#
> position of the pocket please. :')px(1),py(1),pz(1)
            PRINT {''}
          OR &loc='2'
            DIGI (MLOC,NOWAIT,2,n,'',' Digitize 2 diagonal<#
> corner position of the pocket please. :')px(1),py(1),pz(1)
            PRINT {''}
                  WHEN px(1)>=px(2)
                       pxl=px(1)
                       pxs=px(2)
                  ELSE
                       pxl=px(2)
                       pxs=px(1)
                  ENDWHEN
                  WHEN py(1)>=py(2)
                       pyl=py(1)
                       pys=py(2)
                  ELSE
                       pyl=py(2)
                       pys=py(1)
                  ENDWHEN
            IF(&fd='2') DIGI (MLOC,NOWAIT,'',' Digitize the<#
> lower position of the flap please. :')px(3),py(3),pz(3)
            IF(&fd='2') PRINT {''}
       ENDWHEN
ENDWHEN
pxl=px(1)
pyl=py(1)
px3=px(3)
py3=py(3)
#999 RETURN
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|                        Square patch pocket (PCPA1)                    |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE REAL x(8),y(8)
DECLARE LOCATION P(8)
DECLARE ENTITY L(8),cp1(4),cp2(4)
ERTRAP 999
<#
<#
CALLP PCPASIZE(&flap,&loc,&corner,&fd,pwidth,pdepth,<#
>fdepth,pxl,pyl,pxl,pyl,pxs,pys,px3,py3)
<#
<#
PRINT {''}
PRINT      Select the position for the left lower<#
> corner of the pocket please.
PRINT   ----------------------------------<#
>----------------------------------
PRINT    * Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT      at the position you require.<#
> Press the left button.
PRINT   ----------------------------------<#
>----------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
<#
<#
DISP OFF
WHEN &loc='1'
          WHEN &corner='1'
                    x(1)=x(3)=:pxl
                    y(1)=y(2)=:pyl-pdepth
                    x(2)=x(4)=:x(1)+pwidth
                    y(3)=y(4)=:pyl
            OR &corner='2'
                    x(1)=x(3)=:pxl-pwidth
                    y(1)=y(2)=:pyl-pdepth
                    x(2)=x(4)=:pxl
                    y(3)=y(4)=:pyl
    OR &corner='3'
            x(1)=x(3)=:pxl
            y(1)=y(2)=:pyl
            x(2)=x(4)=:x(1)+pwidth
            y(3)=y(4)=:pyl+pdepth
    ELSE
            x(1)=x(3)=:pxl-pwidth
            y(1)=y(2)=:pyl
            x(2)=x(4)=:pxl
            y(3)=y(4)=:pyl+pdepth
ENDWHEN
REPEAT 100: i=1,1,i>4
P(i)=VECT(x(i),y(i),0)
#100 CONTINUE
L(1)=LINE/P(1),P(2)
```

```
          L(2)=LINE/P(2),P(4)
          L(3)=LINE/P(4),P(3)
          L(4)=LINE/P(3),P(1)
          $L=LINE/(x(1)+x(2))/2,y(3)-2,0,(x(1)+x(2))/2,<#
>y(1)+2,0,FONT,'ARROW'
          @V=VECT(X-x(1),Y-y(1),0)
          TRANSL (COPY,1,cp1(1)) 4,L(1),INDIRV,@V
       TRANSL $L,INDIRV,@V
          WHEN &flap='2'
                    x(5)=x(1)
                    x(6)=x(4)
                    y(5)=y(6)=:y(3)-fdepth
                    P(5)=VECT(x(5),y(5),0)
                    P(6)=VECT(x(6),y(6),0)
                    L(5)=LINE/P(5),P(6)
                    L(6)=LINE/P(3),P(5)
                    L(7)=LINE/P(4),P(6)
                    TRANSL (COPY,1,cp2) L(5),INDIRV,@V
                    TRANSL 2,L(6),INDIRV,@V
                    MIRROR cp2(1),L(6),L(7),MY,X,Y+y(3)-y(1),0
    OR &flap='3'
          x(5)=x(7)=:x(1)
          y(5)=y(6)=:y(3)-fdepth+1.5
          x(6)=x(8)=:x(4)
          y(7)=y(8)=:y(3)+1.5
          P(5)=VECT(x(5),y(5),0)
          P(6)=VECT(x(6),y(6),0)
          P(7)=VECT(x(7),y(7),0)
          P(8)=VECT(x(8),y(8),0)
          L(5)=LINE/P(5),P(6)
          L(6)=LINE/P(6),P(8)
          L(7)=LINE/P(8),P(7)
          L(8)=LINE/P(7),P(5)
          TRANSL (COPY,1,cp2) 4,L(5),INDIRV,@V
          MIRROR 4,cp2(1),MY,X,Y+y(3)-y(1)+1.5,0
       ENDWHEN
    OR &loc='2'
          x(1)=x(3)=:pxs
          y(1)=y(2)=:pys
          x(2)=x(4)=:pxl
          y(3)=y(4)=:pyl
          REPEAT 200: i=1,1,i>4
             P(i)=VECT(x(i),y(i),0)
          #200 CONTINUE
          L(1)=LINE/P(1),P(2)
          L(2)=LINE/P(2),P(4)
          L(3)=LINE/P(4),P(3)
          L(4)=LINE/P(3),P(1)
       $L=LINE/(x(1)+x(2))/2,y(3)-2,0,(x(1)+x(2))/2,<#
>y(1)+2,0,FONT,'ARROW'
          @V=VECT(X-x(1),Y-y(1),0)
          TRANSL (COPY,1,cp1) 4,L(1),INDIRV,@V
       TRANSL $L,INDIRV,@V
          WHEN &flap='2'
                   x(5)=x(1)
                   x(6)=x(4)
                     WHEN &fd='1'
                          y(5)=y(6)=:y(3)-fdepth
```

```
      OR &fd='2'
        y(5)=y(6)=:py3
    ENDWHEN
P(5)=VECT(x(5),y(5),0)
P(6)=VECT(x(6),y(6),0)
L(5)=LINE/P(5),P(6)
L(6)=LINE/P(3),P(5)
L(7)=LINE/P(4),P(6)
TRANSL (COPY,1,cp2) L(5),INDIRV,@V
TRANSL 2,L(6),INDIRV,@V
MIRROR cp2(1),L(6),L(7),MY,X,Y+y(3)-y(1),0
OR &flap='3'
     x(5)=x(7)=:x(1)
     x(6)=x(8)=:x(4)
        WHEN &fd='1'
             y(5)=y(6)=:y(3)-fdepth+1.5
             y(7)=y(8)=:y(3)+1.5
          OR &fd='2'
             y(5)=y(6)=:py3
             y(7)=y(8)=:y(3)+1.5
        ENDWHEN
     P(5)=VECT(x(5),y(5),0)
     P(6)=VECT(x(6),y(6),0)
     P(7)=VECT(x(7),y(7),0)
     P(8)=VECT(x(8),y(8),0)
     L(5)=LINE/P(5),P(6)
     L(6)=LINE/P(6),P(8)
     L(7)=LINE/P(8),P(7)
     L(8)=LINE/P(7),P(5)
     TRANSL(COPY,1,cp2) 4,L(5),INDIRV,@V
     MIRROR 4,cp2(1),MY,X,Y+y(3)-y(1)+1.5,0
  ENDWHEN
ENDWHEN
DISP ALL
#999 END
```

```
+-----------------------------------------------------------------+
|                                                                 |
|                Chop-corner patch pocket (PCPA2)                 |
|                                                                 |
+-----------------------------------------------------------------+
<#
<#
DECLARE REAL x(12),y(12)
DECLARE LOCATION P(12)
DECLARE ENTITY L(12),cp1(6),cp2(6)
ERTRAP 999
CALLP PCPASIZE(&flap,&loc,&corner,&fd,pwidth,pdepth,<#
>fdepth,px1,py1,px1,py1,pxs,pys,px3,py3)
PRINT {''}
PRINT   Select the position for the left lower corner<#
> of pocket pattern please.
PRINT   ------------------------------------<#
>----------------------------
PRINT   * Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT     at the position you require.<#
> Press the left button.
PRINT   ------------------------------------<#
>----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
DISP OFF
WHEN &loc='1'
        WHEN &corner='1'
          x(1)=x(3)=:px1
                y(1)=y(2)=:py1
          x(2)=x(4)=:x(1)+pwidth
          y(3)=y(4)=:y(1)-pdepth+pwidth/10
          x(5)=x(1)+pwidth/10
          x(6)=x(2)-pwidth/10
          y(5)=y(6)=:py1-pdepth
          OR &corner='2'
                x(1)=x(3)=:px1-pwidth
          y(1)=y(2)=:py1
          x(2)=x(4)=:px1
          y(3)=y(4)=:py1-pdepth+pwidth/10
          x(5)=x(1)+pwidth/10
                x(6)=x(2)-pwidth/10
                y(5)=y(6)=:y(1)-pdepth
    OR &corner='3'
          x(1)=x(3)=:px1
    y(1)=y(2)=:py1+pdepth
    x(2)=x(4)=:x(1)+pwidth
    y(3)=y(4)=:py1+pwidth/10
    x(5)=x(1)+pwidth/10
          x(6)=x(2)-pwidth/10
          y(5)=y(6)=:py1
ELSE
    x(1)=x(3)=:px1-pwidth
       y(1)=y(2)=:py1+pdepth
    x(2)=x(4)=:px1
    y(3)=y(4)=:py1+pwidth/10
    x(5)=x(1)+pwidth/10
```

```
                x(6)=x(2)-pwidth/10
                y(5)=y(6)=:pyl
    ENDWHEN
    REPEAT 100: i=1,1,i>6
    P(i)=VECT(x(i),y(i),0)
    #100 CONTINUE
L(1)=LINE/P(1),P(3)
L(2)=LINE/P(3),P(5)
L(3)=LINE/P(5),P(6)
L(4)=LINE/P(6),P(4)
      L(5)=LINE/P(4),P(2)
      L(6)=LINE/P(2),P(1)
      $L=LINE/(x(1)+x(2))/2,y(1)-2,0,(x(1)+x(2))/2,<#
>y(5)+2,0,FONT,'ARROW'
         @V=VECT(X-x(1),Y-y(5),0)
         TRANSL (COPY,1,cp1(1)) 6,L(1),INDIRV,@V
      TRANSL $L,INDIRV,@V
      WHEN &flap='2'
           x(7)=x(2)-pwidth
                x(8)=x(2)
                y(7)=y(8)=:y(1)-fdepth
                P(7)=VECT(x(7),y(7),0)
                P(8)=VECT(x(8),y(8),0)
                L(7)=LINE/P(7),P(8)
                L(8)=LINE/P(1),P(7)
                L(9)=LINE/P(2),P(8)
                TRANSL (COPY,1,cp2) L(7),INDIRV,@V
           TRANSL 2,L(8),INDIRV,@V
           MIRROR cp2(1),L(8),L(9),MY,X,Y+y(1)-y(5),0
OR &flap='3'
x(7)=x(9)=:x(2)-pwidth
           x(8)=x(10)=:x(2)
           y(7)=y(8)=:y(1)-fdepth+1.5+pwidth/15
           y(9)=y(10)=:y(1)+1.5
x(11)=x(7)+pwidth/15
x(12)=x(8)-pwidth/15
y(11)=y(12)=:y(1)-fdepth+1.5
REPEAT 200: i=7,1,i>12
P(i)=VECT(x(i),y(i),0)
#200 CONTINUE
L(7)=LINE/P(7),P(11)
L(8)=LINE/P(11),P(12)
L(9)=LINE/P(8),P(12)
           L(10)=LINE/P(10),P(8)
           L(11)=LINE/P(9),P(10)
           L(12)=LINE/P(9),P(7)
                TRANSL (COPY,1,cp2) 6,L(7),INDIRV,@V
                MIRROR 6,cp2(1),MY,X,Y+y(1)-y(5)+1.5,0
      ENDWHEN
OR &loc='2'
      x(1)=x(3)=:pxs
      y(1)=y(2)=:pyl
      x(2)=x(4)=:pxl
      y(3)=y(4)=:pys+(pxl-pxs)/10
      x(5)=x(1)+(pxl-pxs)/10
      x(6)=x(2)-(pxl-pxs)/10
      y(5)=y(6)=:pys
      REPEAT 300: i=1,1,i>6
```

```
        P(1)=VECT(x(1),y(1),0)
        #300 CONTINUE
        L(1)=LINE/P(1),P(3)
        L(2)=LINE/P(3),P(5)
        L(3)=LINE/P(5),P(6)
        L(4)=LINE/P(6),P(4)
        L(5)=LINE/P(4),P(2)
        L(6)=LINE/P(2),P(1)
     $L=LINE/(x(1)+x(2))/2,y(1)-2,0,(x(1)+x(2))/2,<#
>y(5)+2,0,FONT,'ARROW'
   @V=VECT(X-x(1),Y-y(5),0)
   TRANSL (COPY,1,cp1) 6,L(1),INDIRV,@V
TRANSL $L,INDIRV,@V
   WHEN &flap='2'
     x(7)=x(2)-(px1-pxs)
          x(8)=x(2)
          WHEN &fd='1'
                y(7)=y(8)=:y(1)-fdepth
             OR &fd='2'
                y(7)=y(8)=:py3
          ENDWHEN
          P(7)=VECT(x(7),y(7),0)
          P(8)=VECT(x(8),y(8),0)
          L(7)=LINE/P(7),P(8)
          L(8)=LINE/P(1),P(7)
          L(9)=LINE/P(2),P(8)
          TRANSL (COPY,1,cp2) L(7),INDIRV,@V
          TRANSL 2,L(8),INDIRV,@V
          MIRROR cp2(1),L(8),L(9),MY,X,Y+y(1)-y(5),0
    OR &flap='3'
          x(7)=x(9)=:x(2)-(px1-pxs)
          x(8)=x(10)=:x(2)
   y(9)=y(10)=:y(1)+1.5
   x(11)=x(7)+pwidth/15
   x(12)=x(8)-pwidth/15
          WHEN &fd='1'
                y(7)=y(8)=:y(1)-fdepth+1.5+pwidth/15
          y(11)=y(12)=:y(1)-fdepth+1.5
             OR &fd='2'
                y(7)=y(8)=:py3+pwidth/15
          y(11)=y(12)=:py3
          ENDWHEN
   REPEAT 400: i=7,1,i>12
          P(i)=VECT(x(i),y(i),0)
          #400 CONTINUE
                L(7)=LINE/P(7),P(11)
                L(8)=LINE/P(11),P(12)
                L(9)=LINE/P(12),P(8)
                L(10)=LINE/P(10),P(8)
          L(11)=LINE/P(9),P(10)
          L(12)=LINE/P(9),P(7)
                TRANSL(COPY,1,cp2) 6,L(7),INDIRV,@V
                MIRROR 6,cp2(1),MY,X,Y+y(1)-y(5)+1.5,0
       ENDWHEN
ENDWHEN
DISP ALL
#999 END
```

+----------+
| |
| Round-corner patch pocket (PCPA3) |
| |
+----------+

```
<#
<#
DECLARE REAL x(12),y(12)
DECLARE LOCATION P(12)
DECLARE ENTITY L(8),C(4),cp1(6),cp2(6)
ERTRAP 999
CALLP PCPASIZE(&flap,&loc,&corner,&fd,pwidth,pdepth,<#
>fdepth,pxl,pyl,pxl,pyl,pxs,pys,px3,py3)
PRINT {''}
PRINT   Select the position for the left lower<#
> corner of pocket pattern please.
PRINT   ----------------------------------<#
>--------------------------------
PRINT   * Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT     at the position you require.<#
> Press the left button.
PRINT   ----------------------------------<#
>--------------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
DISP OFF
WHEN &loc='1'
        WHEN &corner='1'
                x(1)=x(3)=:pxl
                y(1)=y(2)=:pyl
                x(2)=x(4)=:x(1)+pwidth
                y(3)=y(4)=:y(1)-pdepth+pwidth/5
           x(5)=x(1)+pwidth/5
           x(6)=x(2)-pwidth/5
           y(5)=y(6)=:pyl-pdepth
        OR &corner='2'
           x(1)=x(3)=:pxl-pwidth
                y(1)=y(2)=:pyl
                x(2)=x(4)=:pxl
                y(3)=y(4)=:pyl-pdepth+pwidth/5
                x(5)=x(1)+pwidth/5
                x(6)=x(2)-pwidth/5
           y(5)=y(6)=:y(1)-pdepth
    OR &corner='3'
           x(1)=x(3)=:pxl
           y(1)=y(2)=:pyl+pdepth
           x(2)=x(4)=:x(1)+pwidth
           y(3)=y(4)=:pyl+pwidth/5
           x(5)=x(1)+pwidth/5
           x(6)=x(2)-pwidth/5
           y(5)=y(6)=:pyl
    ELSE
           x(1)=x(3)=:pxl-pwidth
           y(1)=y(2)=:pyl+pdepth
           x(2)=x(4)=:pxl
           y(3)=y(4)=:pyl+pwidth/5
           x(5)=x(1)+pwidth/5
```

```
            x(6)=x(2)-pwidth/5
            y(5)=y(6)=:pyl
    ENDWHEN
    REPEAT 100: i=1,1,i>6
    P(i)=VECT(x(i),y(i),0)
    #100 CONTINUE
    L(1)=LINE/P(1),P(3)
L(2)=LINE/P(1),P(2)
L(3)=LINE/P(2),P(4)
L(4)=LINE/P(5),P(6)
C(1)=CIRCLE/RADIUS,pwidth/5,RIGHT,P(3),P(5),<#
>PGO,P(3),PEND,P(5)
     C(2)=CIRCLE/RADIUS,pwidth/5,LEFT,P(6),P(4),<#
>PGO,P(6),PEND,P(4)
     $L=LINE/(x(1)+x(2))/2,y(1)-2,0,(x(1)+x(2))/2,<#
>y(5)+2,0,FONT,'ARROW'
     @V=VECT(X-x(1),Y-y(5),0)
        TRANSL (COPY,1,cp1(1)) 4,L(1),INDIRV,@V
     TRANSL (COPY,1,cp1(5)) 2,C(1),INDIRV,@V
     TRANSL $L,INDIRV,@V
     WHEN &flap='2'
          x(7)=x(2)-pwidth
               x(8)=x(2)
          y(7)=y(8)=:y(1)-fdepth+pwidth/8
          x(9)=x(7)+pwidth/8
          x(10)=x(8)-pwidth/8
          y(9)=y(10)=:y(1)-fdepth
          REPEAT 200: i=7,1,i>10
          P(i)=VECT(x(i),y(i),0)
          #200 CONTINUE
          L(5)=LINE/P(9),P(10)
          L(6)=LINE/P(1),P(7)
          L(7)=LINE/P(2),P(8)
          C(3)=CIRCLE/RADIUS,pwidth/8,RIGHT,P(7),P(9),<#
>PGO,P(7),PEND,P(9)
          C(4)=CIRCLE/RADIUS,pwidth/8,LEFT,P(10),P(8),<#
>PGO,P(10),PEND,P(8)
          TRANSL (COPY,1,cp2) L(5),INDIRV,@V
          TRANSL (COPY,1,cp2(2)) 2,C(3),INDIRV,@V
               TRANSL 2,L(6),INDIRV,@V
          MIRROR 3,cp2(1),MY,X,Y+y(1)-y(5),0
          MIRROR 2,L(6),MY,X,Y+y(1)-y(5),0
     OR &flap='3'
          x(7)=x(11)=:x(2)-pwidth
          x(8)=x(12)=:x(2)
          y(7)=y(8)=:y(1)-fdepth+1.5+pwidth/8
          y(11)=y(12)=:y(1)+1.5
          x(9)=x(7)+pwidth/8
          x(10)=x(8)-pwidth/8
          y(9)=y(10)=:y(1)-fdepth+1.5
          REPEAT 300: i=7,1,i>12
          P(i)=VECT(x(i),y(i),0)
               #300 CONTINUE
          L(5)=LINE/P(7),P(11)
          L(6)=LINE/P(11),P(12)
          L(7)=LINE/P(12),P(8)
          L(8)=LINE/P(9),P(10)
          C(3)=CIRCLE/RADIUS,pwidth/8,RIGHT,P(7),P(9),<#
```

```
>PGO,P(7),PEND,P(9)
                    C(4)=CIRCLE/RADIUS,pwidth/8,LEFT,P(10),P(8),<#
>PGO,P(10),PEND,P(8)
            TRANSL (COPY,1,cp2) 4,L(5),INDIRV,@V
            TRANSL (COPY,1,cp2(5)) 2,C(3),INDIRV,@V
            MIRROR 6,cp2(1),MY,X,Y+y(1)-y(5)+1.5,0
          ENDWHEN
  OR &loc='2'
      x(1)=x(3)=:pxs
          y(1)=y(2)=:py1
          x(2)=x(4)=:px1
          y(3)=y(4)=:pys+(px1-pxs)/5
          x(5)=x(1)+(px1-pxs)/5
          x(6)=x(2)-(px1-pxs)/5
          y(5)=y(6)=:pys
          REPEAT 400: i=1,1,i>6
          P(i)=VECT(x(i),y(i),0)
          #400 CONTINUE
        L(1)=LINE/P(3),P(1)
        L(2)=LINE/P(1),P(2)
        L(3)=LINE/P(2),P(4)
        L(4)=LINE/P(5),P(6)
        C(1)=CIRCLE/RADIUS,(px1-pxs)/5,RIGHT,P(3),P(5),<#
>PGO,P(3),PEND,P(5)
        C(2)=CIRCLE/RADIUS,(px1-pxs)/5,LEFT,P(6),P(4),<#
>PGO,P(6),PEND,P(4)
        $L=LINE/(x(1)+x(2))/2,y(1)-2,0,(x(1)+x(2))/2,<#
>y(5)+2,0,FONT,'ARROW'
        @V=VECT(X-x(1),Y-y(5),0)
           TRANSL (COPY,1,cp1) 4,L(1),INDIRV,@V
        TRANSL (COPY,1,cp1(5)) 2,C(1),INDIRV,@V
        TRANSL $L,INDIRV,@V
        WHEN &flap='2'
             x(7)=x(2)-(px1-pxs)
                  x(8)=x(2)
             x(9)=x(7)+(px1-pxs)/8
             x(10)=x(8)-(px1-pxs)/8
                  WHEN &fd='1'
                            y(7)=y(8)=:y(1)-fdepth+(px1-pxs)/8
               y(9)=y(10)=:y(1)-fdepth
          OR &fd='2'
               y(7)=y(8)=:py3+(px1-pxs)/8
               y(9)=y(10)=:py3
             ENDWHEN
             REPEAT 500: i=7,1,i>10
             P(i)=VECT(x(i),y(i),0)
             #500 CONTINUE
             L(5)=LINE/P(9),P(10)
             L(6)=LINE/P(1),P(7)
             L(7)=LINE/P(2),P(8)
             C(3)=CIRCLE/RADIUS,(px1-pxs)/8,RIGHT,P(7),P(9),<#
>PGO,P(7),PEND,P(9)
             C(4)=CIRCLE/RADIUS,(px1-pxs)/8,LEFT,P(10),P(8),<#
>PGO,P(10),PEND,P(8)
             TRANSL (COPY,1,cp2) L(5),INDIRV,@V
             TRANSL (COPY,1,cp2(2)) 2,C(3),INDIRV,@V
             TRANSL 2,L(6),INDIRV,@V
             MIRROR 3,cp2(1),MY,X,Y+y(1)-y(5),0
```

```
    MIRROR 2,L(6),MY,X,Y+y(1)-y(5),0
OR &flap='3'
    x(7)=x(11)=:x(2)-(px1-pxs)
    x(8)=x(12)=:x(2)
    x(9)=x(7)+(px1-pxs)/8
          x(10)=x(8)-(px1-pxs)/8
    y(11)=y(12)=:y(1)+1.5
    WHEN &fd='1'
                    y(7)=y(8)=:y(1)-fdepth+1.5+(px1-pxs)/8
        y(9)=y(10)=:y(1)-fdepth+1.5
      OR &fd='2'
                    y(7)=y(8)=:py3+(px1-pxs)/8
          y(9)=y(10)=:py3
        ENDWHEN
        REPEAT 600: i=7,1,i>12
        P(i)=VECT(x(i),y(i),0)
             #600 CONTINUE
        L(5)=LINE/P(9),P(10)
        L(6)=LINE/P(7),P(11)
        L(7)=LINE/P(11),P(12)
        L(8)=LINE/P(12),P(8)
        C(3)=CIRCLE/RADIUS,(px1-pxs)/8,RIGHT,P(7),P(9),<#
>PGO,P(7),PEND,P(9)
        C(4)=CIRCLE/RADIUS,(px1-pxs)/8,LEFT,P(10),P(8),<#
>PGO,P(10),PEND,P(8)
        TRANSL(COPY,1,cp2) 4,L(5),INDIRV,@V
        TRANSL(COPY,1,cp2(5)) 2,C(3),INDIRV,@V
          MIRROR 6,cp2,MY,X,Y+y(1)-y(5)+1.5,0
      ENDWHEN
ENDWHEN
DISP ALL
#999 END
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|                    Square-V patch pocket (PCPA4)                      |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE REAL x(10),y(10)
DECLARE LOCATION P(10)
DECLARE ENTITY L(10),cp1(5),cp2(5)
ERTRAP 999
CALLP PCPASIZE(&flap,&loc,&corner,&fd,pwidth,pdepth,<#
>fdepth,pxl,pyl,pxl,pyl,pxs,pys,px3,py3)
PRINT {''}
PRINT   Select the position for the left lower corner<#
> of pocket pattern please.
PRINT   ------------------------------------<#
>----------------------------
PRINT   * Move the mouse on the pad to place the<#
> cursor on the graphic window
PRINT     at the position you require.<#
> Press the left button.
PRINT   ------------------------------------<#
>----------------------------
DIGI (MLOC,NOWAIT) X,Y,Z
PRINT {''}
DISP OFF
WHEN &loc='1'
        WHEN &corner='1'
          x(1)=x(3)=:pxl
                y(1)=y(2)=:pyl
                x(2)=x(4)=:x(1)+pwidth
          y(3)=y(4)=:y(1)-pdepth*3/4
          x(5)=(x(1)+x(2))/2
          y(5)=y(1)-pdepth
        OR &corner='2'
                x(1)=x(3)=:pxl-pwidth
                y(1)=y(2)=:pyl
                x(2)=x(4)=:pxl
                y(3)=y(4)=:pyl-pdepth*3/4
          x(5)=(x(1)+x(2))/2
                y(5)=y(1)-pdepth
          OR &corner='3'
                x(1)=x(3)=:pxl
                y(1)=y(2)=:pyl+pdepth
                x(2)=x(4)=:x(1)+pwidth
                y(3)=y(4)=:pyl+pdepth/4
        x(5)=(x(1)+x(2))/2
                y(5)=pyl
   ELSE
        x(1)=x(3)=:pxl-pwidth
                y(1)=y(2)=:pyl+pdepth
                x(2)=x(4)=:pxl
                y(3)=y(4)=:pyl+pdepth/4
                x(5)=(x(1)+x(2))/2
                y(5)=pyl
ENDWHEN
REPEAT 100: i=1,1,i>5
```

```
        P(i)=VECT(x(i),y(i),0)
        #100 CONTINUE
      L(1)=LINE/P(1),P(3)
      L(2)=LINE/P(3),P(5)
      L(3)=LINE/P(5),P(4)
      L(4)=LINE/P(4),P(2)
      L(5)=LINE/P(1),P(2)
      $L=LINE/(x(1)+x(2))/2,y(1)-2,0,(x(1)+x(2))/2,<#
>y(5)+2,0,FONT,'ARROW'
      @V=VECT(X-x(1),Y-y(5),0)
   TRANSL (COPY,1,cp1(1)) 5,L(1),INDIRV,@V
TRANSL $L,INDIRV,@V
WHEN &flap='2'
      x(6)=x(1)
      x(7)=x(2)
      y(6)=y(7)=:y(1)-fdepth+pdepth/8
      x(8)=x(5)
      y(8)=y(1)-fdepth
      P(6)=VECT(x(6),y(6),0)
      P(7)=VECT(x(7),y(7),0)
      P(8)=VECT(x(8),y(8),0)
      L(6)=LINE/P(6),P(8)
      L(7)=LINE/P(7),P(8)
      L(8)=LINE/P(1),P(6)
      L(9)=LINE/P(2),P(7)
      TRANSL (COPY,1,cp2) 2,L(6),INDIRV,@V
      TRANSL 2,L(8),INDIRV,@V
   MIRROR 2,cp2(1),MY,X,Y+y(1)-y(5),0
         MIRROR 2,L(8),MY,X,Y+y(1)-y(5),0
OR &flap='3'
   x(6)=x(9)=:x(1)
   x(7)=x(10)=:x(2)
   y(6)=y(7)=:y(1)-fdepth+pdepth/8+1.5
   x(8)=x(5)
   y(8)=y(1)-fdepth+1.5
   y(9)=y(10)=:y(1)+1.5
   REPEAT 200: i=6,1,i>10
   P(i)=VECT(x(i),y(i),0)
   #200 CONTINUE
   L(6)=LINE/P(6),P(8)
   L(7)=LINE/P(7),P(8)
   L(8)=LINE/P(7),P(10)
   L(9)=LINE/P(9),P(10)
      L(10)=LINE/P(9),P(6)
      TRANSL (COPY,1,cp2) 5,L(6),INDIRV,@V
      MIRROR 5,cp2(1),MY,X,Y+y(1)-y(5)+1.5,0
   ENDWHEN
OR &loc='2'
   x(1)=x(3)=:pxs
      y(1)=y(2)=:pyl
      x(2)=x(4)=:pxl
      y(3)=y(4)=:pys+(pyl-pys)/4
   x(5)=(x(1)+x(2))/2
   y(5)=pys
   REPEAT 300: i=1,1,i>5
   P(i)=VECT(x(i),y(i),0)
   #300 CONTINUE
   L(1)=LINE/P(1),P(3)
```

```
      L(2)=LINE/P(3),P(5)
      L(3)=LINE/P(4),P(5)
      L(4)=LINE/P(2),P(4)
      L(5)=LINE/P(1),P(2)
      $L=LINE/(x(1)+x(2))/2,y(1)-2,0,(x(1)+x(2))/2,<#
>y(5)+2,0,FONT,'ARROW'
      @V=VECT(X-x(1),Y-y(5),0)
         TRANSL (COPY,1,cp1) 5,L(1),INDIRV,@V
      TRANSL $L,INDIRV,@V
      WHEN &flap='2'
           x(6)=x(1)
           x(7)=x(2)
           x(8)=x(5)
           WHEN &fd='1'
                          y(6)=y(7)=:y(1)-fdepth+(py1-pys)/8
                y(8)=y(1)-fdepth
             OR &fd='2'
                y(6)=y(7)=:py3+(py1-pys)/8
                y(8)=py3
           ENDWHEN
           P(6)=VECT(x(6),y(6),0)
           P(7)=VECT(x(7),y(7),0)
           P(8)=VECT(x(8),y(8),0)
           L(6)=LINE/P(6),P(8)
           L(7)=LINE/P(7),P(8)
           L(8)=LINE/P(1),P(6)
           L(9)=LINE/P(2),P(7)
           TRANSL (COPY,1,cp2) 2,L(6),INDIRV,@V
           TRANSL 2,L(8),INDIRV,@V
           MIRROR cp2(1),cp2(2),L(8),L(9),<#
>MY,X,Y+y(1)-y(5),0
         OR &flap='3'
           x(6)=x(9)=:x(1)
           x(7)=x(10)=:x(2)
           x(8)=x(5)
           y(9)=y(10)=:y(1)+1.5
           WHEN &fd='1'
                y(6)=y(7)=:y(1)+1.5-fdepth+(py1-pys)/8
                y(8)=y(1)+1.5-fdepth
             OR &fd='2'
                          y(6)=y(7)=:py3+(py1-pys)/8
                y(8)=py3
           ENDWHEN
           REPEAT 400: i=6,1,i>10
                    P(i)=VECT(x(i),y(i),0)
           #400 CONTINUE
           L(6)=LINE/P(6),P(8)
           L(7)=LINE/P(7),P(8)
           L(8)=LINE/P(7),P(10)
           L(9)=LINE/P(9),P(10)
           L(10)=LINE/P(9),P(6)
           TRANSL(COPY,1,cp2) 5,L(6),INDIRV,@V
           MIRROR 5,cp2,MY,X,Y+y(1)-y(5)+1.5,0
      ENDWHEN
ENDWHEN
DISP ALL
#999 END
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|                  Horizontal worked buttonholes (BT1)                  |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE ENTITY cp1(30),cp2(30)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the number of buttonholes please. ---> )bt
PRINT {''}
#10 CONTINUE
READ( Which units do you want to use? <#
>   Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'
        READ( Enter the diameter of the button please.<#
>  unit; cm ---> )btdm
        READ( Enter the thickness of the button please.<#
>  unit; cm ---> )btht
  OR &unit='IN'.OR.&unit='in'
        READ( Enter the diameter of the button please.<#
>  unit; in ---> )btdm
        READ( Enter the thickness of the button please.<#
>  unit; in ---> )btht
        btdm=btdm*2.54
        btht=btht*2.54
  ELSE
        PRINT Wrong answer, enter again please.
        GOTO 10
ENDWHEN
btlngt=btdm+btht+0.3
<#
<#
PRINT {''}
WHEN bt=1
        PRINT Enter the location of the buttonhole on<#
> the button placement line please.
        DIGI (MLOC,NOWAIT)x1,y1,z1
        PRINT {''}
  ELSE
        PRINT Enter 2 locations for the first and <#
>the last buttonhole on the button placement line please.
        DIGI (MLOC,NOWAIT)x2,y2,z2
        PRINT {''}
        DIGI (MLOC,NOWAIT)x3,y3,z3
        PRINT {''}
ENDWHEN
<#
<#
#20 CONTINUE
PRINT {''}
PRINT Enter the location to that direction<#
> buttonhole will be made.
DIGI (MLOC,NOWAIT)x4,y4,z4
PRINT {''}
```

```
DISP OFF
IF (bt>1) GOTO 30
WHEN x4<x1
          $L1=LINE/x1+0.3,y1,0,x1-btlngt+0.3,y1,0
  OR x4>x1
          $L1=LINE/x1-0.3,y1,0,x1+btlngt-0.3,y1,0
  ELSE
PRINT The location for the buttonhole direction was wrong.
PRINT Enter again please.
          GOTO 20
ENDWHEN
GOTO 900
<#
<#
#30 CONTINUE
WHEN x3>x2
          D=(y3-y2)*(x4-x2)/(x3-x2)+y2
          WHEN D>y4
                    &dir='left'
            OR D<y4
                    &dir='right'
            ELSE
     PRINT The location for the buttonhole direction was wrong.
     PRINT Enter again please.
                    GOTO 20
          ENDWHEN
  OR x3<x2
          D=(y2-y3)*(x4-x3)/(x2-x3)+y3
          WHEN D>y4
                    &dir='right'
            OR D<y4
                    &dir='left'
            ELSE
   PRINT The location for the buttonhole direction was wrong.
   PRINT Enter again please.
                    GOTO 20
          ENDWHEN
ELSE
          WHEN x4>x2
                    &dir='right'
            OR x4<x2
                    &dir='left'
            ELSE
     PRINT The location for the buttonhole direction was wrong.
     PRINT Enter again please.
                    GOTO 20
          ENDWHEN
ENDWHEN
<#
<#
$E=LINE/x2,y2,0,x3,y3,0
$P=POINT/x2,y2,0
WHEN &dir='left'
          $L1=LINE/$P,PERP,$E,LENGTH,0.3
          $L2=LINE/$P,PERP,$E,LENGTH,-btlngt+0.3
  OR &dir='right'
          $L1=LINE/$P,PERP,$E,LENGTH,-0.3
          $L2=LINE/$P,PERP,$E,LENGTH,btlngt-0.3
```

```
ENDWHEN
@V=VECT((x3-x2)/(bt-1),(y3-y2)/(bt-1),0)
TRANSL(COPY,bt-1,cp1) $L1,INDIRV,@V
TRANSL(COPY,bt-1,cp2) $L2,INDIRV,@V
DELETE $P,$E
#900 CONTINUE
DISP ALL
#999 END
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|                  Horizontal bound buttonholes (BT2)                   |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE ENTITY P(7),L(7),cpl(100)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the number of buttonholes please. ---> )bt
PRINT {''}
#10 CONTINUE
READ( Which units do you want to use?<#
>     Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'
          READ( Enter the diameter of the button please.<#
>   unit; cm ---> )btdm
          READ( Enter the thickness of the button please.<#
>   unit; cm ---> )btht
   OR &unit='IN'.OR.&unit='in'
          READ( Enter the diameter of the button please.<#
>   unit; in ---> )btdm
          READ( Enter the thickness of the button please.<#
>   unit; in ---> )btht
          btdm=btdm*2.54
          btht=btht*2.54
   ELSE
          PRINT Wrong answer, enter again please.
          GOTO 10
ENDWHEN
btlngt=btdm+btht
<#
<#
PRINT {''}
WHEN bt=1
          PRINT Enter the location of the buttonhole on<#
> the button placement line please.
          DIGI (MLOC,NOWAIT)x1,y1,z1
          PRINT {''}
   ELSE
          PRINT Enter 2 locations for the first and the<#
> last buttonhole on the button placement line please.
          DIGI (MLOC,NOWAIT)x2,y2,z2
          PRINT {''}
          DIGI (MLOC,NOWAIT)x3,y3,z3
          PRINT {''}
ENDWHEN
<#
<#
#20 CONTINUE
PRINT {''}
PRINT Enter the location to that direction buttonhole<#
> will be made.
DIGI (MLOC,NOWAIT)x4,y4,z4
PRINT {''}
```

```
DISP OFF
IF (bt>1) GOTO 30
WHEN x4<x1
        L(1)=LINE/x1+0.3,y1,0,x1-btlngt+0.3,y1,0
        L(2)=LINE/x1+0.3,y1+0.3,0,x1-btlngt+0.3,y1+0.3,0
        L(3)=LINE/x1+0.3,y1-0.3,0,x1-btlngt+0.3,y1-0.3,0
        L(4)=LINE/x1+0.3,y1+0.3,0,x1+0.3,y1-0.3,0
        L(5)=LINE/x1-btlngt+0.3,y1+0.3,0,x1-btlngt+0.3,y1-0.3,0
   OR x4>x1
        L(1)=LINE/x1-0.3,y1,0,x1+btlngt-0.3,y1,0
        L(2)=LINE/x1-0.3,y1+0.3,0,x1+btlngt-0.3,y1+0.3,0
        L(3)=LINE/x1-0.3,y1-0.3,0,x1+btlngt-0.3,y1-0.3,0
        L(4)=LINE/x1-0.3,y1+0.3,0,x1-0.3,y1-0.3,0
        L(5)=LINE/x1+btlngt-0.3,y1+0.3,0,x1+btlngt-0.3,y1-0.3,0
   ELSE
        PRINT The location for the buttonhole direction was wrong.
        PRINT Enter again please.
        GOTO 20
ENDWHEN
GOTO 900
<#
<#
#30 CONTINUE
WHEN x3>x2
          D=(y3-y2)*(x4-x2)/(x3-x2)+y2
          WHEN D>y4
                    &dir='left'
            OR D<y4
                    &dir='right'
            ELSE
PRINT The location for the buttonhole direction was wrong.
PRINT Enter again please.
                    GOTO 20
          ENDWHEN
  OR x3<x2
          D=(y2-y3)*(x4-x3)/(x2-x3)+y3
          WHEN D>y4
                    &dir='right'
            OR D<y4
                    &dir='left'
            ELSE
PRINT The location for the buttonhole direction was wrong.
PRINT Enter again please.
                    GOTO 20
          ENDWHEN
   ELSE
          WHEN x4>x2
                    &dir='right'
            OR x4<x2
                    &dir='left'
            ELSE
PRINT The location for the buttonhole direction was wrong.
PRINT Enter again please.
                    GOTO 20
          ENDWHEN
ENDWHEN
<#
<#
```

```
$E=LINE/x2,y2,0,x3,y3,0
P(1)=POINT/x2,y2,0
WHEN &dir='left'
        L(1)=LINE/P(1),PERP,$E,LENGTH,0.3
        L(2)=LINE/P(1),PERP,$E,LENGTH,-btlngt+0.3
         P(2)=POINT/P(1),DISTAN,btlngt-0.3,L(2)
         L(3)=LINE/P(2),PERP,L(2),LENGTH,-0.3
         P(3)=POINT/P(2),DISTAN,0.3,L(3)
         L(4)=LINE/P(2),PERP,L(2),LENGTH,0.3
         P(4)=POINT/P(2),DISTAN,0.3,L(4)
        L(5)=LINE/P(3),PARLEL,L(2),LENGTH,-btlngt
        P(5)=POINT/P(3),DISTAN,btlngt,L(5)
        L(6)=LINE/P(4),PARLEL,L(2),LENGTH,-btlngt
        P(6)=POINT/P(4),DISTAN,btlngt,L(6)
        L(7)=LINE/P(5),P(6)
OR &dir='right'
        L(1)=LINE/P(1),PERP,$E,LENGTH,-0.3
        L(2)=LINE/P(1),PERP,$E,LENGTH,btlngt-0.3
        P(2)=POINT/P(1),DISTAN,btlngt-0.3,L(2)
        L(3)=LINE/P(2),PERP,L(2),LENGTH,0.3
        P(3)=POINT/P(2),DISTAN,0.3,L(3)
        L(4)=LINE/P(2),PERP,L(2),LENGTH,-0.3
        P(4)=POINT/P(2),DISTAN,0.3,L(4)
        L(5)=LINE/P(3),PARLEL,L(2),LENGTH,-btlngt
        P(5)=POINT/P(3),DISTAN,btlngt,L(5)
        L(6)=LINE/P(4),PARLEL,L(2),LENGTH,-btlngt
        P(6)=POINT/P(4),DISTAN,btlngt,L(6)
        L(7)=LINE/P(5),P(6)
ENDWHEN
DELETE 7,P(1)
DELETE $E
@V=VECT((x3-x2)/(bt-1),(y3-y2)/(bt-1),0)
TRANSL(COPY,bt-1,cp1) 7,L(1),INDIRV,@V
#900 CONTINUE
DISP ALL
#999 END
```

```
+-----------------------------------------------------------------------+
|                                                                       |
|                  Vertical worked buttonholes (BT3)                    |
|                                                                       |
+-----------------------------------------------------------------------+
<#
<#
DECLARE ENTITY cp1(30),cp2(30)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the number of buttonholes please. ---> )bt
PRINT {''}
#10 CONTINUE
READ( Which units do you want to use?<#
>     Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'
          READ( Enter the diameter of the button please.<#
>  unit; cm ---> )btdm
          READ( Enter the thickness of the button please.<#
>  unit; cm ---> )btht
   OR &unit='IN'.OR.&unit='in'
          READ( Enter the diameter of the button please.<#
>  unit; in ---> )btdm
          READ( Enter the thickness of the button please.<#
>  unit; in ---> )btht
          btdm=btdm*2.54
          btht=btht*2.54
   ELSE
          PRINT Wrong answer, enter again please.
          GOTO 10
ENDWHEN
btlngt=btdm+btht+0.3
<#
<#
PRINT {''}
WHEN bt=1
          PRINT Enter the location of the buttonhole on<#
> the button placement line please.
          DIGI (MLOC,NOWAIT)x1,y1,z1
          PRINT {''}
   ELSE
          PRINT Enter 2 locations for the first and the<#
> last buttonhole on the button placement line please.
          DIGI (MLOC,NOWAIT)x2,y2,z2
          PRINT {''}
          DIGI (MLOC,NOWAIT)x3,y3,z3
          PRINT {''}
ENDWHEN
<#
<#
DISP OFF
IF (bt>1) GOTO 30
$L1=LINE/x1,y1+0.3,0,x1,y1-btlngt+0.3,0
GOTO 900
<#
<#
```

```
#30 CONTINUE
$E=LINE/x2,y2,0,x3,y3,0
$P=POINT/x2,y2,0
$L1=LINE/$P,PARLEL,$E,LENGTH,-0.3
$L2=LINE/$P,PARLEL,$E,LENGTH,btlngt-0.3
@V=VECT((x3-x2)/(bt-1),(y3-y2)/(bt-1),0)
TRANSL(COPY,bt-1,cp1) $L1,INDIRV,@V
TRANSL(COPY,bt-1,cp2) $L2,INDIRV,@V
DELETE $P,$E
#900 CONTINUE
DISP ALL
#999 END
```

Vertical bound buttonholes (BT4)

```
<#
<#
DECLARE ENTITY P(7),L(7),cpl(100)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the number of buttonholes please. ---> )bt
PRINT {''}
#10 CONTINUE
READ( Which units do you want to use?<#
>     Enter "IN" or "CM" ---> )&unit
WHEN &unit='CM'.OR.&unit='cm'
          READ( Enter the diameter of the button please.<#
>   unit; cm ---> )btdm
          READ( Enter the thickness of the button please.<#
>   unit; cm ---> )btht
   OR &unit='IN'.OR.&unit='in'
          READ( Enter the diameter of the button please.<#
>   unit; in ---> )btdm
          READ( Enter the thickness of the button please.<#
>   unit; in ---> )btht
          btdm=btdm*2.54
          btht=btht*2.54
   ELSE
          PRINT Wrong answer, enter again please.
          GOTO 10
ENDWHEN
btlngt=btdm+btht
<#
<#
PRINT {''}
WHEN bt=1
          PRINT Enter the location of the buttonhole<#
> on the button placement line please.
          DIGI (MLOC,NOWAIT)x1,y1,z1
          PRINT {''}
   ELSE
          PRINT Enter 2 locations for the first and the<#
> last buttonhole on the button placement line please.
          DIGI (MLOC,NOWAIT)x2,y2,z2
          PRINT {''}
          DIGI (MLOC,NOWAIT)x3,y3,z3
          PRINT {''}
ENDWHEN
<#
<#
DISP OFF
IF (bt>1) GOTO 30
L(1)=LINE/x1,y1+0.3,0,x1,y1-btlngt+0.3,0
L(2)=LINE/x1-0.3,y1+0.3,0,x1-0.3,y1-btlngt+0.3,0
L(3)=LINE/x1+0.3,y1+0.3,0,x1+0.3,y1-btlngt+0.3,0
L(4)=LINE/x1-0.3,y1+0.3,0,x1+0.3,y1+0.3,0
```

```
L(5)=LINE/x1-0.3,y1-btlngt+0.3,0,x1+0.3,y1-btlngt+0.3,0
GOTO 900
<#
<#
#30 CONTINUE
$E=LINE/x2,y2,0,x3,y3,0
P(1)=POINT/x2,y2,0
L(1)=LINE/P(1),PARLEL,$E,LENGTH,-0.3
L(2)=LINE/P(1),PARLEL,$E,LENGTH,btlngt-0.3
P(2)=POINT/P(1),DISTAN,btlngt-0.3,L(2)
L(3)=LINE/P(2),PERP,L(2),LENGTH,-0.3
L(4)=LINE/P(2),PERP,L(2),LENGTH,0.3
P(3)=POINT/P(2),DISTAN,0.3,L(3)
P(4)=POINT/P(2),DISTAN,0.3,L(4)
L(5)=LINE/P(3),PARLEL,L(2),LENGTH,-btlngt
P(5)=POINT/P(3),DISTAN,btlngt,L(5)
L(6)=LINE/P(4),PARLEL,L(2),LENGTH,-btlngt
P(6)=POINT/P(4),DISTAN,btlngt,L(6)
L(7)=LINE/P(5),P(6)
DELETE 7,P(1)
DELETE $E
@V=VECT((x3-x2)/(bt-1),(y3-y2)/(bt-1),0)
TRANSL(COPY,bt-1,cp1) 7,L(1),INDIRV,@V
#900 CONTINUE
DISP ALL
#999 END
```

Button placement (BT5)

```
<#
<#
DECLARE ENTITY cpl(30)
ERTRAP 999
<#
<#
PRINT {''}
READ( Enter the number of buttons please. ---> )bt
PRINT {''}
WHEN bt=1
        PRINT Enter the location of the button on the<#
> button placement line please.
        DIGI (MLOC,NOWAIT)x1,y1,z1
        PRINT {''}
        $P=POINT/x1,y1,0
  ELSE
        PRINT Enter 2 locations for the first and the<#
> last button on the button placement line please.
        DIGI (MLOC,NOWAIT)x2,y2,z2
        PRINT {''}
        DIGI (MLOC,NOWAIT)x3,y3,z3
        PRINT {''}
        $P=POINT/x2,y2,0
        @V=VECT((x3-x2)/(bt-1),(y3-y2)/(bt-1),0)
        TRANSL(COPY,bt-1,cpl) $P,INDIRV,@V
ENDWHEN
#999 END
```

Grain line (GRAINL)

```
<#
<#
DECLARE REAL x(20),y(20),z(20)
DECLARE ENTITY G(10)
ERTRAP 999
<#
<#
#10 CONTINUE
PRINT {''}
PRINT                Which grainline do you want?
PRINT    ---------------------------------------
PRINT    Vertical grainline ------------------------ 1
PRINT    Horizontal grainline ---------------------- 2
PRINT    Bias grainline (an angle of 45 degrees) --- 3
PRINT    Bias grainline (an angle of 135 degrees) -- 4
PRINT    Any two locations as you want ------------- 5
PRINT    ---------------------------------------
READ(          Enter the number please. ---> )&grain
WHEN &grain='1'.OR.&grain='2'.OR.&grain='3'.OR.<#
>&grain='4'.OR.&grain='5'
         CONTINUE
  ELSE
         PRINT Wrong answer, enter again please.
         GOTO 10
ENDWHEN
<#
<#
PRINT {''}
DIGI(MLOC,20,n,'',' Enter the locations for<#
> grainlines up to 20 :') x,y,z
DISP OFF
WHEN &grain='1'
      REPEAT 100: i=2,2,i>n
      G(i)=LINE/x(i-1),y(i-1),0,x(i-1),y(i),0,FONT,'ARROW'
      #100 CONTINUE
  OR &grain='2'
       REPEAT 200: i=2,2,i>n
       G(i)=LINE/x(i-1),y(i-1),0,x(i),y(i-1),0,FONT,'ARROW'
      #200 CONTINUE
  OR &grain='3'
      REPEAT 300: i=2,2,i>n
      G(i)=LINE/x(i-1),y(i-1),0,x(i-1)+(y(i)-y(i-1)+x(i)-<#
>x(i-1))/2,y(i-1)+(y(i)-y(i-1)+x(i)-x(i-1))/2,0,FONT,'ARROW'
      #300 CONTINUE
 OR &grain='4'
      REPEAT 400: i=2,2,i>n
      G(i)=LINE/x(i-1),y(i-1),0,x(i-1)+(y(i-1)-y(i)+x(i)-<#
>x(i-1))/2,y(i-1)+(y(i)-y(i-1)+x(i-1)-x(i))/2,0,FONT,'ARROW'
      #400 CONTINUE
  ELSE
      REPEAT 500: i=2,2,i>n
      G(i)=LINE/x(i-1),y(i-1),0,x(i),y(i),0,FONT,'ARROW'
      #500 CONTINUE
```

```
ENDWHEN
DISP ALL
#999 END
```

Notch point (NOTCH)

```
<#
<#
DECLARE LOCATION L(30)
DECLARE ENTITY N(30)
ERTRAP 999
<#
<#
DIGI(MLOC,30,n,'',' Enter the locations for notch<#
> points up to 30 :')L
DISP OFF
REPEAT 100: i=1,1,i>n
N(i)=CIRCLE/CENTER,L(i),RADIUS,0.4
#100 CONTINUE
DISP ALL
#999 END
```

APPENDIX C

GRADING PROGRAMS

The grading program for bodice patterns. (GDBD)

```
<#
<#
DECLARE REAL BUST(12),NATOWA(12),s(10),dx(9),dy(9),<#
>x(100,9),y(100,9),z(100,9)
DECLARE REAL layer(9),size(9)
DECLARE LOCATION GP(1000)
DECLARE ENTITY GL(1000),GB(9),GPT(1000),GNP(1000),T(9)
DATA BUST/80,84,88,92,97,102,107,112,117,122,127,132/
DATA NATOWA/39,39.5,40,40.5,41,41.5,42,42.5,43,43.2,<#
>43.4,43.6/
ERTRAP 999
GETPRM (LAYER) lyr
<#
<#
#10 CONTINUE
s(1)=0
PRINT The size must be an even number between 8 and 30.
READ( Enter the current pattern size please. ---> )s(1)
WHEN s(1)=8.OR.s(1)=10.OR.s(1)=12.OR.s(1)=14.OR.<#
>s(1)=16.OR.s(1)=18.OR.s(1)=20.OR.s(1)=22.OR.s(1)=24<#
>.OR.s(1)=26.OR.s(1)=28.OR.s(1)=30
        CONTINUE
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
REPEAT 30: i=2,1,i>10
       s(i)=0
#30 CONTINUE
PRINT {''}
PRINT ----------------------------------------------
PRINT * You can grade at most 9 different sizes, which must be
PRINT   even numbers between 8 and 30.
PRINT * Enter required sizes separated by space or comma.
PRINT ----------------------------------------------
READ, FORMAT = 'FREE', (0) s(2),s(3),s(4),s(5),s(6),s(7)<#
>,s(8),s(9),s(10)
j=0
REPEAT 100: i=2,1,i>10
          WHEN s(i)=8.OR.s(i)=10.OR.s(i)=12.OR.s(i)=14<#
>.OR.s(i)=16.OR.s(i)=18.OR.s(i)=20.OR.s(i)=22.OR.s(i)=24<#
>.OR.s(i)=26.OR.s(i)=28.OR.s(i)=30
                    j=j+1
                    size(j)=s(i)
                    k1=(s(i)-6)/2
                    k2=(s(1)-6)/2
                    dx(j)=(BUST(k1)-BUST(k2))/BUST(k2)
                    dy(j)=(NATOWA(k1)-NATOWA(k2))/NATOWA(k2)
                    layer(j)=lyr+j
               OR s(i)=0
```

```
                GOTO 100
          ELSE
            PRINT Wrong value entered, enter again please.
               GOTO 20
        ENDWHEN
#100 CONTINUE
<#
<#
PRINT {''}
PRINT Wait a moment please.
REPEAT 110: i=1,1,i>j
  WHEN i=1
          &color='MAGENTA'
    OR i=2
          &color='BLUE'
    OR i=3
          &color='RED'
    OR i=4
          &color='GREEN'
    OR i=5
          &color='CYAN'
    OR i=6
          &color='GRAY'
    OR i=7
          &color='YELLOW'
    OR i=8
          &color='GREEN'
      OR i=9
            &color='RED'
        ENDWHEN
        !SELECT LDISCRIMINATION {&color} LAYER {layer(i)}
        !ECHO LAYER INCLUDE {layer(i)}
#110 CONTINUE
!DISCRIMINATE LAYER
EXECDF
<#
<#
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location for the zero<#
> point please. :') X,Y,Z
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location for the text<#
> <size name> please. :')<#
>t1,t2,t3
PRINT {''}
<#
<#
REPEAT 160: i=1,1,i>j
        SELECT (LAYER) layer(i)
        &size=size(i)
        T(i)=TEXT/'size '+&size,t1,t2,t3,TJST,'LJT',THGT,1
#160 CONTINUE
SELECT (LAYER) lyr
<#
<#
#200 CONTINUE
PRINT {''}
&type=''
```

```
PRINT   ----------------------------------------<#
>-------------------------
PRINT   Lines -->1     Bspline (curve) -->2    <#
> Points -->3     Notch points -->4
PRINT   ----------------------------------------<#
>-------------------------
READ( Enter the number of the entity type, <#
>or Q to quit. ---> )&type
WHEN &type='1'
        GOSUB 1000
  OR &type='2'
        GOSUB 2000
  OR &type='3'
        GOSUB 3000
  OR &type='4'
        GOSUB 4000
  OR &type='Q'.OR.&type='q'
        PRINT{''}
        PRINT Every size is on a different layer as follows.
        REPEAT 210: i=1,1,i>j
        WHEN size(i)<10
PRINT Size  {size(i)} pattern pieces are on the layer {layer(i)}.
          ELSE
PRINT Size {size(i)} pattern pieces are on the layer {layer(i)}.
        ENDWHEN
        #210 CONTINUE
        END
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 200
ENDWHEN
GOTO 200
#999 END
<#
<#
#1000 CONTINUE
PRINT {''}
font=1
PRINT   ----------------------------------------<#
>-------------------------
PRINT   Solid --> 1     Dash --> 2    <#
> Arrow(grain line) --> 3     1-dash --> 4
PRINT   ----------------------------------------<#
>-------------------------
READ( Enter the number of the line font please. <#
>  default:1 ---> )font
WHEN font=1
        &F='SOLID'
  OR font=2
        &F='DASH'
  OR font=3
        &F='ARROW'
  OR font=4
        &F='1-DASH'
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 1000
ENDWHEN
```

```
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded<#
> :')x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 1100:i=1,1,i>n
        REPEAT 1200: g=1,1,g>j
                SELECT (LAYER) layer(g)
                x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
                IF (i>1) GL(j*(i-2)+g)=LINE/GP(j*(i-2)+g),<#
>GP(j*(i-1)+g),FONT,&F
        #1200 CONTINUE
#1100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#2000 CONTINUE
PRINT {''}
PRINT          ----------------------------------<#
>------------------------
PRINT                You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ----------------------------------<#
>------------------------
PRINT          * In order to grade a bspline (curve)<#
> accurately, a few points
PRINT            on the current bspline are needed.
PRINT            If you want to insert points on the<#
> current bspline now, use
PRINT            the icon (GENERATE POINT ON N -> : ).
PRINT          * When you are ready for return to the<#
> program, please press
PRINT            keys "Control" and "x" at the same time.
PRINT          ----------------------------------<#
>------------------------
!<VAR>
EXECDF
#2010 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter up to 9 locations to be <#
>graded :')x(1,1),y(1,1),z(1,1)
PRINT {''}
REPEAT 2100: i=1,1,i>n
        REPEAT 2150: g=1,1,g>j
                x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
        #2150 CONTINUE
#2100 CONTINUE
DISP OFF
REPEAT 2200: g=1,1,g>j
        SELECT (LAYER) layer(g)
        WHEN n=3
             GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2)
```

```
            OR n=4
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),GP(g+j*3)
            OR n=5
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4)
            OR n=6
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5)
            OR n=7
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6)
            OR n=8
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7)
            OR n=9
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7),GP(g+j*8)
         ENDWHEN
#2200 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#3000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded<#
> :')x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 3100: i=1,1,i>n
         REPEAT 3150: g=1,1,g>j
                  SELECT (LAYER) layer(g)
                  x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                  y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                  GPT(j*(i-1)+g)=POINT/x(i,g+1),y(i,g+1),0
         #3150 CONTINUE
#3100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#4000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded <#
>:')x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 4100: i=1,1,i>n
         REPEAT 4150:g=1,1,g>j
                  SELECT (LAYER) layer(g)
                  x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                  y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                  GNP(j*(i-1)+g)=CIRCLE/CENTER,x(i,g+1),<#
>y(i,g+1),0,RADIUS,0.4
         #4150 CONTINUE
#4100 CONTINUE
```

```
DISP ALL
SELECT (LAYER) lyr
RTNSUB
```

```
+-----------------------------------------------------------+
|                                                           |
|        The grading program for skirt patterns. (GDSK)     |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL HIP(12),WATOKN(12),s(10),dx(9),dy(9),<#
>x(100,9),y(100,9),z(100,9)
DECLARE REAL layer(9),size(9)
DECLARE LOCATION GP(1000)
DECLARE ENTITY GL(1000),GB(9),GPT(1000),GNP(1000),T(9)
DATA HIP/85,89,93,97,102,107,112,117,122,127,132,137/
DATA WATOKN/57.5,58,58.5,59,59.5,60,60.5,61,61.25,<#
>61.5,61.75,62/
ERTRAP 999
GETPRM (LAYER) lyr
<#
<#
#10 CONTINUE
s(1)=0
PRINT The size must be an even number between 8 and 30.
READ( Enter the current pattern size please. ---> )s(1)
WHEN s(1)=8.OR.s(1)=10.OR.s(1)=12.OR.s(1)=14.OR.s(1)=16.<#
>OR.s(1)=18.OR.s(1)=20.OR.s(1)=22.OR.s(1)=24.OR.s(1)=26<#
>.OR.s(1)=28.OR.s(1)=30
        CONTINUE
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
REPEAT 30: i=2,1,i>10
        s(i)=0
#30 CONTINUE
PRINT {''}
PRINT ----------------------------------------
PRINT * You can grade at most 9 different sizes, which must be
PRINT   even numbers between 8 and 30.
PRINT * Enter required sizes separated by space or comma.
PRINT ----------------------------------------
READ, FORMAT = 'FREE', (0) s(2),s(3),s(4),s(5),s(6),s(7)<#
>,s(8),s(9),s(10)
j=0
REPEAT 100: i=2,1,i>10
          WHEN s(i)=8.OR.s(i)=10.OR.s(i)=12.OR.s(i)=14.<#
>OR.s(i)=16.OR.s(i)=18.OR.s(i)=20.OR.s(i)=22.OR.s(i)=24<#
>.OR.s(i)=26.OR.s(i)=28.OR.s(i)=30
                    j=j+1
                    size(j)=s(i)
                    k1=(s(i)-6)/2
                    k2=(s(1)-6)/2
                    dx(j)=(HIP(k1)-HIP(k2))/HIP(k2)
                    dy(j)=(WATOKN(k1)-WATOKN(k2))/WATOKN(k2)
                    layer(j)=lyr+j
            OR s(i)=0
```

```
                GOTO 100
            ELSE
                PRINT Wrong value entered, enter again please.
                GOTO 20
        ENDWHEN
#100 CONTINUE
<#
<#
PRINT {''}
PRINT Wait a moment please.
REPEAT 110: i=1,1,i>j
WHEN i=1
            &color='MAGENTA'
  OR i=2
            &color='BLUE'
  OR i=3
            &color='RED'
  OR i=4
            &color='GREEN'
  OR i=5
            &color='CYAN'
  OR i=6
            &color='GRAY'
  OR i=7
            &color='YELLOW'
  OR i=8
            &color='GREEN'
    OR i=9
              &color='RED'
        ENDWHEN
        !SELECT LDISCRIMINATION {&color} LAYER {layer(i)}
        !ECHO LAYER INCLUDE {layer(i)}
        EXECDF
#110 CONTINUE
!DISCRIMINATE LAYER
EXECDF
<#
<#
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location for zero<#
> point please. :') X,Y,Z
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location for text<#
> please. :')t1,t2,t3
PRINT {''}
<#
<#
REPEAT 160: i=1,1,i>j
        SELECT (LAYER) layer(i)
        &size=size(i)
        T(i)=TEXT/'size '+&size,t1,t2,t3,TJST,'LJT',THGT,1
#160 CONTINUE
SELECT (LAYER) lyr
<#
<#
#200 CONTINUE
PRINT {''}
&type=''
```

```
PRINT   ----------------------------------------<#
>------------------------
PRINT   Lines -->1     Bspline (curve) -->2    <#
> Points -->3     Notch points -->4
PRINT   ----------------------------------------<#
>------------------------
READ( Enter the number of the entity type, or Q to quit.<#
> ---> )&type
WHEN &type='1'
        GOSUB 1000
  OR &type='2'
        GOSUB 2000
  OR &type='3'
        GOSUB 3000
  OR &type='4'
        GOSUB 4000
  OR &type='Q'.OR.&type='q'
        PRINT{''}
        PRINT Every size is on a different layer as follows.
        REPEAT 210: i=1,1,i>j
        WHEN size(i)<10
PRINT Size  {size(i)} pattern pieces are on the layer {layer(i)}.
          ELSE
PRINT Size {size(i)} pattern pieces are on the layer {layer(i)}.
        ENDWHEN
        #210 CONTINUE
        END
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 200
ENDWHEN
GOTO 200
#999 END
<#
<#
#1000 CONTINUE
PRINT {''}
font=1
PRINT   ----------------------------------------<#
>------------------------
PRINT   Solid --> 1     Dash --> 2    <#
> Arrow(grain line) --> 3     1-dash --> 4
PRINT   ----------------------------------------<#
>------------------------
READ( Enter the number of the line font please. <#
>  default:1 ---> )font
WHEN font=1
        &F='SOLID'
  OR font=2
        &F='DASH'
  OR font=3
        &F='ARROW'
  OR font=4
        &F='1-DASH'
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 1000
ENDWHEN
```

```
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded<#
> :')x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 1100:i=1,1,i>n
        REPEAT 1200: g=1,1,g>j
                  SELECT (LAYER) layer(g)
                  x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                  y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                  GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
                  IF (i>1) GL(j*(i-2)+g)=LINE/GP(j*(i-2)+g)<#
>,GP(j*(i-1)+g),FONT,&F
        #1200 CONTINUE
#1100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#2000 CONTINUE
PRINT {''}
PRINT          ------------------------------------<#
>-----------------------
PRINT                You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ------------------------------------<#
>-----------------------
PRINT          * In order to grade a bspline (curve)<#
> accurately, a few points
PRINT            on the current bspline are needed.
PRINT            If you want to insert points on the<#
> current bspline now, use
PRINT            the icon (GENERATE POINT ON N -> : ).
PRINT          * When you are ready for return to the<#
> program, please press
PRINT            keys "Control" and "x" at the same time.
PRINT          ------------------------------------<#
>-----------------------
!<VAR>
EXECDF
#2010 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter up to 9 locations to be<#
> graded :')x(1,1),y(1,1),z(1,1)
PRINT {''}
REPEAT 2100: i=1,1,i>n
        REPEAT 2150: g=1,1,g>j
                  x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                  y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                  GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
        #2150 CONTINUE
#2100 CONTINUE
DISP OFF
REPEAT 2200: g=1,1,g>j
        SELECT (LAYER) layer(g)
         WHEN n=3
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2)
```

```
            OR n=4
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),GP(g+j*3)
            OR n=5
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4)
            OR n=6
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5)
            OR n=7
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6)
            OR n=8
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7)
            OR n=9
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7),GP(g+j*8)
         ENDWHEN
#2200 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#3000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded<#
> :')x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 3100: i=1,1,i>n
         REPEAT 3150: g=1,1,g>j
                  SELECT (LAYER) layer(g)
                  x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                  y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                  GPT(j*(i-1)+g)=POINT/x(i,g+1),y(i,g+1),0
         #3150 CONTINUE
#3100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#4000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded<#
> :')x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 4100: i=1,1,i>n
         REPEAT 4150:g=1,1,g>j
                  SELECT (LAYER) layer(g)
                  x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                  y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                  GNP(j*(i-1)+g)=CIRCLE/CENTER,x(i,g+1),<#
>y(i,g+1),0,RADIUS,0.4
         #4150 CONTINUE
#4100 CONTINUE
```

```
DISP ALL
SELECT (LAYER) lyr
RTNSUB
```

The grading program for trouser patterns. (GDTR)

```
<#
<#
DECLARE REAL HIP(12),WATOFL(12),s(10),dx(9),dy(9),<#
>x(100,9),y(100,9),z(100,9)
DECLARE REAL layer(9),size(9)
DECLARE LOCATION GP(1000)
DECLARE ENTITY GL(1000),GB(9),GPT(1000),GNP(1000),T(9)
DATA HIP/85,89,93,97,102,107,112,117,122,127,132,137/
DATA WATOFL/102,103,104,105,106,107,108,109,109.5,110,<#
>110.5,111/
ERTRAP 999
GETPRM (LAYER) lyr
<#
<#
#10 CONTINUE
s(1)=0
PRINT The size must be an even number between 8 and 30.
READ( Enter the current pattern size please. ---> )s(1)
WHEN s(1)=8.OR.s(1)=10.OR.s(1)=12.OR.s(1)=14.OR.s(1)=16.<#
>OR.s(1)=18.OR.s(1)=20.OR.s(1)=22.OR.s(1)=24.OR.s(1)=26<#
>.OR.s(1)=28.OR.s(1)=30
          CONTINUE
   ELSE
          PRINT Wrong value entered, enter again please.
          GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
REPEAT 30: i=2,1,i>10
          s(i)=0
#30 CONTINUE
PRINT {''}
PRINT ----------------------------------------------------
PRINT * You can grade at most 9 different sizes, which must be
PRINT    even numbers between 8 and 30.
PRINT * Enter required sizes separated by space or comma.
PRINT ----------------------------------------------------
READ, FORMAT = 'FREE', (0) s(2),s(3),s(4),s(5),s(6),s(7)<#
>,s(8),s(9),s(10)
j=0
REPEAT 100: i=2,1,i>10
          WHEN s(i)=8.OR.s(i)=10.OR.s(i)=12.OR.s(i)=14.OR<#
>.s(i)=16.OR.s(i)=18.OR.s(i)=20.OR.s(i)=22.OR.s(i)=24<#
>.OR.s(i)=26.OR.s(i)=28.OR.s(i)=30
                    j=j+1
                    size(j)=s(i)
                    k1=(s(i)-6)/2
                    k2=(s(1)-6)/2
                    dx(j)=(HIP(k1)-HIP(k2))/HIP(k2)
                    dy(j)=(WATOFL(k1)-WATOFL(k2))/WATOFL(k2)
                    layer(j)=lyr+j
             OR s(i)=0
```

```
                GOTO 100
          ELSE
                PRINT Wrong value entered, enter again please.
                GOTO 20
        ENDWHEN
#100 CONTINUE
<#
<#
PRINT {''}
PRINT Wait a moment please.
REPEAT 110: i=1,1,i>j
          WHEN i=1
                &color='MAGENTA'
            OR i=2
                &color='BLUE'
            OR i=3
                &color='RED'
            OR i=4
                &color='GREEN'
            OR i=5
                &color='CYAN'
            OR i=6
                &color='GRAY'
            OR i=7
                &color='YELLOW'
            OR i=8
                &color='GREEN'
              OR i=9
                    &color='RED'
          ENDWHEN
          !SELECT LDISCRIMINATION {&color} LAYER {layer(i)}
          !ECHO LAYER INCLUDE {layer(i)}
          EXECDF
#110 CONTINUE
!DISCRIMINATE LAYER
EXECDF
<#
<#
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location for zero point<#
> please. :') X,Y,Z
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location for text please.<#
> :')t1,t2,t3
PRINT {''}
REPEAT 160: i=1,1,i>j
          SELECT (LAYER) layer(i)
          &size=size(i)
          T(i)=TEXT/'size '+&size,t1,t2,t3,TJST,'LJT',THGT,1
#160 CONTINUE
SELECT (LAYER) lyr
<#
<#
#200 CONTINUE
PRINT {''}
&type=''
PRINT   ----------------------------------------<#
>------------------------
```

```
PRINT   Lines -->1     Bspline (curve) -->2    <#
> Points -->3     Notch points -->4
PRINT   ---------------------------------------<#
>------------------------
READ( Enter the number of the entity type,<#
> or Q to quit. ---> )&type
WHEN &type='1'
         GOSUB 1000
  OR &type='2'
         GOSUB 2000
  OR &type='3'
         GOSUB 3000
  OR &type='4'
         GOSUB 4000
  OR &type='Q'.OR.&type='q'
         PRINT{''}
         PRINT Every size is on a different layer as follows.
         REPEAT 210: i=1,1,i>j
         WHEN size(i)<10
PRINT Size  {size(i)} pattern pieces are on the layer {layer(i)}.
           ELSE
PRINT Size {size(i)} pattern pieces are on the layer {layer(i)}.
         ENDWHEN
         #210 CONTINUE
         END
  ELSE
         PRINT Wrong value entered, enter again please.
         GOTO 200
ENDWHEN
GOTO 200
#999 END
<#
<#
#1000 CONTINUE
PRINT {''}
font=1
PRINT   ---------------------------------------<#
>------------------------
PRINT   Solid --> 1     Dash --> 2   <#
>  Arrow(grain line) --> 3     1-dash --> 4
PRINT   ---------------------------------------<#
>------------------------
READ( Enter the number of the line font please. <#
>  default:1 ---> )font
WHEN font=1
         &F='SOLID'
  OR font=2
         &F='DASH'
  OR font=3
         &F='ARROW'
  OR font=4
         &F='1-DASH'
  ELSE
         PRINT Wrong value entered, enter again please.
         GOTO 1000
ENDWHEN
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded :')<#
```

```
>x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 1100:i=1,1,i>n
        REPEAT 1200: g=1,1,g>j
                SELECT (LAYER) layer(g)
                x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
                IF (i>1) GL(j*(i-2)+g)=LINE/GP(j*(i-2)+g)<#
>,GP(j*(i-1)+g),FONT,&F
        #1200 CONTINUE
#1100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#2000 CONTINUE
PRINT {''}
PRINT          ----------------------------------------<#
>----------------------
PRINT                You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ----------------------------------------<#
>----------------------
PRINT          * In order to grade a bspline (curve)<#
> accurately, a few points
PRINT            on the current bspline are needed.
PRINT            If you want to insert points on the<#
> current bspline now, use
PRINT            the icon (GENERATE POINT ON N -> : ).
PRINT          * When you are ready for return to the<#
> program, please press
PRINT            keys "Control" and "x" at the same time.
PRINT          ----------------------------------------<#
>--------------------
!<VAR>
EXECDF
#2010 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter up to 9 locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
REPEAT 2100: i=1,1,i>n
        REPEAT 2150: g=1,1,g>j
                x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
        #2150 CONTINUE
#2100 CONTINUE
DISP OFF
REPEAT 2200: g=1,1,g>j
        SELECT (LAYER) layer(g)
        WHEN n=3
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2)
          OR n=4
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),GP(g+j*3)
```

```
            OR n=5
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4)
            OR n=6
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5)
            OR n=7
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6)
            OR n=8
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7)
            OR n=9
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7),GP(g+j*8)
         ENDWHEN
#2200 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#3000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 3100: i=1,1,i>n
         REPEAT 3150: g=1,1,g>j
                  SELECT (LAYER) layer(g)
                  x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                  y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                  GPT(j*(i-1)+g)=POINT/x(i,g+1),y(i,g+1),0
         #3150 CONTINUE
#3100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#4000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 4100: i=1,1,i>n
         REPEAT 4150:g=1,1,g>j
                  SELECT (LAYER) layer(g)
                  x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                  y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                  GNP(j*(i-1)+g)=CIRCLE/CENTER,x(i,g+1),<#
>y(i,g+1),0,RADIUS,0.4
         #4150 CONTINUE
#4100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
```

RTNSUB

```
+-----------------------------------------------------------+
|                                                           |
|      The grading program for sleeve patterns. (GDSL)      |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL ARMHOLE(12),SLLNGT(12),s(10),dx(9),dy(9),<#
>x(100,9),y(100,9),z(100,9)
DECLARE REAL layer(9),size(9)
DECLARE LOCATION GP(1000)
DECLARE ENTITY GL(1000),GB(9),GPT(1000),GNP(1000),T(9)
DATA SLLNGT/57.2,57.8,58.4,59,59.5,60,60.5,61,61.2,61.4,<#
>61.6,61.8/
ERTRAP 999
GETPRM (LAYER) lyr
<#
<#
#10 CONTINUE
s(1)=0
PRINT The size must be an even number between 8 and 30.
READ( Enter the current pattern size please. ---> )s(1)
WHEN s(1)=8.OR.s(1)=10.OR.s(1)=12.OR.s(1)=14.OR.s(1)=16<#
>.OR.s(1)=18.OR.s(1)=20.OR.s(1)=22.OR.s(1)=24.OR.s(1)=26<#
>.OR.s(1)=28.OR.s(1)=30
          CONTINUE
  ELSE
          PRINT Wrong value entered, enter again please.
          GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
REPEAT 30: i=2,1,i>10
          s(i)=0
#30 CONTINUE
PRINT {''}
PRINT ------------------------------------------------
PRINT * You can grade at most 9 different sizes, which must be
PRINT    even numbers between 8 and 30.
PRINT * Enter required sizes separated by space or comma.
PRINT ------------------------------------------------
READ, FORMAT = 'FREE', (0) s(2),s(3),s(4),s(5),s(6),<#
>s(7),s(8),s(9),s(10)
j=0
REPEAT 100: i=2,1,i>10
          WHEN s(i)=8.OR.s(i)=10.OR.s(i)=12.OR.s(i)=14.OR.<#
>s(i)=16.OR.s(i)=18.OR.s(i)=20.OR.s(i)=22.OR.s(i)=24.OR.<#
>s(i)=26.OR.s(i)=28.OR.s(i)=30
                    j=j+1
                    size(j)=s(i)
                    layer(j)=lyr+j
            OR s(i)=0
                    GOTO 100
            ELSE
                    PRINT Wrong value entered, enter again please.
                    GOTO 20
          ENDWHEN
```

```
#100 CONTINUE
<#
<#
PRINT {''}
PRINT Wait a moment please.
REPEAT 110: i=1,1,i>j
          WHEN i=1
                  &color='MAGENTA'
            OR i=2
                  &color='BLUE'
            OR i=3
                  &color='RED'
            OR i=4
                  &color='GREEN'
            OR i=5
                  &color='CYAN'
            OR i=6
                  &color='GRAY'
            OR i=7
                  &color='YELLOW'
            OR i=8
                  &color='GREEN'
            OR i=9
                  &color='RED'
          ENDWHEN
          !SELECT LDISCRIMINATION {&color} LAYER {layer(i)}
          !ECHO LAYER INCLUDE {layer(i)}
          EXECDF
#110 CONTINUE
!DISCRIMINATE LAYER
EXECDF
<#
<#
PRINT {''}
PRINT          ----------------------------------------<#
>----------------------
PRINT                You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ----------------------------------------<#
>----------------------
PRINT          * Measure the length of the armhole of<#
> all the bodice pieces
PRINT            using the icon (MEASURE LENGTH).
PRINT            The unit of the length will be "cm".
PRINT          * After Measuring, please press keys <#
>"Control" and "x" at the
PRINT            same time  to restart the program.
PRINT          ----------------------------------------<#
>----------------------
PRINT {''}
PRINT Measure the size {s(1)} armhole of bodice pieces please.
!<VAR>
EXECDF
READ( Enter the sum of that armhole please.<#
> <unit: cm> ---> )AH
REPEAT 200: i=1,1,i>j
          PRINT {''}
          PRINT Measure the size {size(i)} armhole of<#
```

```
> bodice pieces please.
        !<VAR>
        EXECDF
        READ( Enter the sum of that armhole please.<#
> <unit: cm> ---> )ARMHOLE(i)
        k1=(size(i)-6)/2
        k2=(s(1)-6)/2
        dx(i)=(ARMHOLE(i)-AH)/AH
        dy(i)=(SLLNGT(k1)-SLLNGT(k2))/SLLNGT(k2)
#200 CONTINUE
<#
<#
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location of the zero<#
> point please. :') X,Y,Z
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location for text<#
> please. :')t1,t2,t3
PRINT {''}
<#
<#
REPEAT 260: i=1,1,i>j
        SELECT (LAYER) layer(i)
        &size=size(i)
   T(i)=TEXT/'size '+&size,t1,t2,t3,TJST,'LJT',THGT,1
#260 CONTINUE
SELECT (LAYER) lyr
<#
<#
#300 CONTINUE
PRINT {''}
&type=''
PRINT   ----------------------------------------<#
>--------------------
PRINT   Lines -->1    Bspline (curve) -->2   <#
> Points -->3    Notch points -->4
PRINT   ----------------------------------------<#
>--------------------
READ( Enter the number of the entity type, <#
>or Q to quit. ---> )&type
WHEN &type='1'
        GOSUB 1000
  OR &type='2'
        GOSUB 2000
  OR &type='3'
        GOSUB 3000
  OR &type='4'
        GOSUB 4000
  OR &type='Q'.OR.&type='q'
        PRINT{''}
        PRINT Every size is on a different layer as follows.
        REPEAT 210: i=1,1,i>j
        WHEN size(i)<10
              PRINT Size  {size(i)} pattern pieces <#
>are on the layer {layer(i)}.
          ELSE
              PRINT Size {size(i)} pattern pieces <#
>are on the layer {layer(i)}.
```

```
        ENDWHEN
        #210 CONTINUE
        END
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 300
ENDWHEN
GOTO 300
#999 END
<#
<#
#1000 CONTINUE
PRINT {''}
font=1
PRINT   ----------------------------------------<#
>------------------------
PRINT   Solid --> 1     Dash --> 2     <#
>Arrow(grain line) --> 3     1-dash --> 4
PRINT   ----------------------------------------<#
>------------------------
READ( Enter the number of line font please. <#
>  default:1 ---> )font
WHEN font=1
        &F='SOLID'
  OR font=2
        &F='DASH'
  OR font=3
        &F='ARROW'
  OR font=4
        &F='1-DASH'
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 1000
ENDWHEN
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 1100:i=1,1,i>n
        REPEAT 1200: g=1,1,g>j
                SELECT (LAYER) layer(g)
                x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
                IF (i>1) GL(j*(i-2)+g)=LINE/GP(j*(i-2)+g)<#
>,GP(j*(i-1)+g),FONT,&F
        #1200 CONTINUE
#1100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#2000 CONTINUE
PRINT {''}
PRINT           ----------------------------------------<#
>------------------------
```

```
PRINT                You can use the icon tool, <#
>while the prompt is "<VAR>".
PRINT          ------------------------------------<#
>----------------------
PRINT          * In order to grade a bspline (curve)<#
> accurately, a few points
PRINT            on the current bspline are needed.
PRINT            If you want to insert points on the <#
>current bspline now, use
PRINT            the icon (GENERATE POINT ON N -> : ).
PRINT          * When you are ready for return to the<#
> program, please press
PRINT            keys "Control" and "x" at the same time.
PRINT          ------------------------------------<#
>----------------------
!<VAR>
EXECDF
#2010 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter up to 9 locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
REPEAT 2100: i=1,1,i>n
        REPEAT 2150: g=1,1,g>j
                x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
        #2150 CONTINUE
#2100 CONTINUE
DISP OFF
REPEAT 2200: g=1,1,g>j
        SELECT (LAYER) layer(g)
        WHEN n=3
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2)
          OR n=4
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),GP(g+j*3)
          OR n=5
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4)
          OR n=6
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5)
          OR n=7
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6)
          OR n=8
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7)
          OR n=9
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7),GP(g+j*8)
        ENDWHEN
#2200 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
```

```
#3000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 3100: i=1,1,i>n
        REPEAT 3150: g=1,1,g>j
                SELECT (LAYER) layer(g)
                x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                GPT(j*(i-1)+g)=POINT/x(i,g+1),y(i,g+1),0
        #3150 CONTINUE
#3100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#4000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 4100: i=1,1,i>n
        REPEAT 4150:g=1,1,g>j
                SELECT (LAYER) layer(g)
                x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                GNP(j*(i-1)+g)=CIRCLE/CENTER,x(i,g+1),<#
>y(i,g+1),0,RADIUS,0.4
        #4150 CONTINUE
#4100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
```

The grading program for collar patterns. (GDCL)

```
<#
<#
DECLARE REAL NECK(12),NATOWA(12),s(10),dx(9),dy(9),<#
>x(100,9),y(100,9),z(100,9)
DECLARE REAL layer(9),size(9)
DECLARE LOCATION GP(1000)
DECLARE ENTITY GL(1000),GB(9),GPT(1000),GNP(1000),T(9)
DATA NATOWA/39,39.5,40,40.5,41,41.5,42,42.5,43,43.2,<#
>43.4,43.6/
ERTRAP 999
GETPRM (LAYER) lyr
<#
<#
#10 CONTINUE
s(1)=0
PRINT The size must be an even number between 8 and 30.
READ( Enter the current pattern size please. ---> )s(1)
WHEN s(1)=8.OR.s(1)=10.OR.s(1)=12.OR.s(1)=14.OR.s(1)=16.<#
>OR.s(1)=18.OR.s(1)=20.OR.s(1)=22.OR.s(1)=24.OR.s(1)=26<#
>.OR.s(1)=28.OR.s(1)=30
          CONTINUE
   ELSE
          PRINT Wrong value entered, enter again please.
          GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
REPEAT 30: i=2,1,i>10
          s(i)=0
#30 CONTINUE
PRINT {''}
PRINT ----------------------------------------
PRINT * You can grade at most 9 different sizes, which must be
PRINT   even numbers between 8 and 30.
PRINT * Enter required sizes separated by space or comma.
PRINT ----------------------------------------
READ, FORMAT = 'FREE', (0) s(2),s(3),s(4),s(5),s(6),s(7)<#
>,s(8),s(9),s(10)
j=0
REPEAT 100: i=2,1,i>10
          WHEN s(i)=8.OR.s(i)=10.OR.s(i)=12.OR.s(i)=14<#
>.OR.s(i)=16.OR.s(i)=18.OR.s(i)=20.OR.s(i)=22.OR.s(i)=24<#
>.OR.s(i)=26.OR.s(i)=28.OR.s(i)=30
                    j=j+1
                    size(j)=s(i)
                    layer(j)=lyr+j
             OR s(i)=0
                    GOTO 100
             ELSE
              PRINT Wrong value entered, enter again please.
                    GOTO 20
          ENDWHEN
```

```
#100 CONTINUE
<#
<#
PRINT {''}
PRINT Wait a moment please.
REPEAT 110: i=1,1,i>j
          WHEN i=1
                  &color='MAGENTA'
            OR i=2
                  &color='BLUE'
            OR i=3
                  &color='RED'
            OR i=4
                  &color='GREEN'
            OR i=5
                  &color='CYAN'
            OR i=6
                  &color='GRAY'
            OR i=7
                  &color='YELLOW'
            OR i=8
                  &color='GREEN'
            OR i=9
                  &color='RED'
        ENDWHEN
        !SELECT LDISCRIMINATION {&color} LAYER {layer(i)}
        !ECHO LAYER INCLUDE {layer(i)}
        EXECDF
        #110 CONTINUE
        !DISCRIMINATE LAYER
        EXECDF
        <#
        <#
        PRINT {''}
        PRINT          ----------------------------------<#
        >--------------------------
        PRINT                  You can use the icon tool,<#
        > while the prompt is "<VAR>".
        PRINT          ----------------------------------<#
        >--------------------------
        PRINT          * Measure the length of a half neck line<#
        > on the bodice pieces
        PRINT            using the icon (MEASURE LENGTH).
        PRINT            The unit of the length will be "cm".
        PRINT          * After Measuring, please press keys<#
        > "Control" and "x" at the
        PRINT            same time  to restart the program.
        PRINT          ----------------------------------<#
        >--------------------------
        PRINT {''}
        PRINT Measure the size {s(1)} a half neck line on<#
        > the bodice pieces please.
        !<VAR>
        EXECDF
        READ( Enter the sum of that half neck please.<#
        > <unit: cm> ---> )neck
        REPEAT 200: i=1,1,i>j
                 PRINT {''}
```

```
          PRINT Measure the size {size(i)} a half neck line<#
> on the bodice please.
          !<VAR>
          EXECDF
          READ( Enter the sum of that half neck please.<#
> <unit: cm> ---> )NECK(i)
          k1=(size(i)-6)/2
          k2=(s(1)-6)/2
          dx(i)=(NECK(i)-neck)/neck
          dy(i)=(NATOWA(k1)-NATOWA(k2))/NATOWA(k2)
#200 CONTINUE
<#
<#
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location of the zero <#
>point please. :') X,Y,Z
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location for text<#
> please. :')t1,t2,t3
PRINT {''}
<#
<#
REPEAT 260: i=1,1,i>j
          SELECT (LAYER) layer(i)
          &size=size(i)
          T(i)=TEXT/'size '+&size,t1,t2,t3,TJST,'LJT',THGT,1
#260 CONTINUE
SELECT (LAYER) lyr
<#
<#
#300 CONTINUE
PRINT {''}
&type=''
PRINT   ----------------------------------------<#
>-------------------------
PRINT   Lines -->1     Bspline (curve) -->2    <#
> Points -->3     Notch points -->4
PRINT   ----------------------------------------<#
>-------------------------
READ( Enter the number of entity type to be generated,<#
> or Q to quit. ---> )&type
WHEN &type='1'
          GOSUB 1000
  OR &type='2'
          GOSUB 2000
  OR &type='3'
          GOSUB 3000
  OR &type='4'
          GOSUB 4000
  OR &type='Q'.OR.&type='q'
          PRINT{''}
          PRINT Every size is on a different layer as follows.
          REPEAT 210: i=1,1,i>j
          WHEN size(i)<10
PRINT Size  {size(i)} pattern pieces are on the layer {layer(i)}.
            ELSE
PRINT Size {size(i)} pattern pieces are on the layer {layer(i)}.
          ENDWHEN
```

```
          #210 CONTINUE
          END
   ELSE
          PRINT Wrong value entered, enter again please.
          GOTO 300
ENDWHEN
GOTO 300
#999 END
<#
<#
#1000 CONTINUE
PRINT {''}
font=1
PRINT   ----------------------------------------<#
>-----------------------
PRINT   Solid --> 1     Dash --> 2    <#
> Arrow(grain line) --> 3     1-dash --> 4
PRINT   ----------------------------------------<#
>-----------------------
READ( Enter the number of line font please.  <#
> default:1 ---> )font
WHEN font=1
          &F='SOLID'
   OR font=2
          &F='DASH'
   OR font=3
          &F='ARROW'
   OR font=4
          &F='1-DASH'
   ELSE
          PRINT Wrong value entered, enter again please.
          GOTO 1000
ENDWHEN
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded<#
> :')x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 1100: i=1,1,i>n
          REPEAT 1200: g=1,1,g>j
                    SELECT (LAYER) layer(g)
                    x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                    y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                    GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
                    IF (i>1) GL(j*(i-2)+g)=LINE/GP(j*(i-2)+g)<#
>,GP(j*(i-1)+g),FONT,&F
          #1200 CONTINUE
#1100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#2000 CONTINUE
PRINT {''}
PRINT          ----------------------------------------<#
>-----------------
PRINT               You can use the icon tool,<#
```

```
> while the prompt is "<VAR>".
PRINT        ----------------------------------------<#
>-----------------
PRINT          * In order to grade a bspline (curve)<#
> accurately, a few points
PRINT            on the current bspline are needed.
PRINT            If you want to insert points on the<#
> current bspline now, use
PRINT            the icon (GENERATE POINT ON N -> : ).
PRINT          * When you are ready for return to the<#
> program, please press
PRINT            keys "Control" and "x" at the same time.
PRINT        ----------------------------------------<#
>-----------------
!<VAR>
EXECDF
#2010 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter up to 9 locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
REPEAT 2100: i=1,1,i>n
        REPEAT 2150: g=1,1,g>j
                x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
        #2150 CONTINUE
#2100 CONTINUE
DISP OFF
REPEAT 2200: g=1,1,g>j
         SELECT (LAYER) layer(g)
         WHEN n=3
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2)
           OR n=4
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),GP(g+j*3)
           OR n=5
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4)
           OR n=6
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5)
           OR n=7
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6)
           OR n=8
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7)
           OR n=9
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7),GP(g+j*8)
         ENDWHEN
#2200 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#3000 CONTINUE
```

```
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded<#
> :')x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 3100: i=1,1,i>n
        REPEAT 3150: g=1,1,g>j
                SELECT (LAYER) layer(g)
                x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                GPT(j*(i-1)+g)=POINT/x(i,g+1),y(i,g+1),0
        #3150 CONTINUE
#3100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#4000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded<#
> :')x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 4100: i=1,1,i>n
        REPEAT 4150:g=1,1,g>j
                SELECT (LAYER) layer(g)
                x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                GNP(j*(i-1)+g)=CIRCLE/CENTER,x(i,g+1),<#
>y(i,g+1),0,RADIUS,0.4
        #4150 CONTINUE
#4100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
```

```
+-----------------------------------------------------------+
|                                                           |
|          The grading program for height (GDHGHT)          |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL NATOWA(12),WATOKN(12),WATOFL(12),SLLNGT(12)
DECLARE REAL x(100,3),y(100,3),z(100,3)
DECLARE LOCATION GP(1000)
DECLARE ENTITY GL(1000),GB(9),GPT(1000),GNP(1000)
DATA NATOWA/39,39.5,40,40.5,41,41.5,42,42.5,43,43.2,43.4,43.6/
DATA WATOKN/57.5,58,58.5,59,59.5,60,60.5,61,61.25,61.5,61.75,62/
DATA WATOFL/102,103,104,105,106,107,108,109,109.5,110,110.5,111/
DATA SLLNGT/57.2,57.8,58.4,59,59.5,60,60.5,61,61.2,61.4,<#
>61.6,61.8/
ERTRAP 999
GETPRM (LAYER) lyr
<#
<#
#10 CONTINUE
PRINT   ----------------------------------------<#
>-----------------------
PRINT   Bodice -->1      Skirt -->2      <#
>Trousers -->3      Sleeves -->4
PRINT   ----------------------------------------<#
>-----------------------
READ(   Enter the number of the pattern type please. ---> )&type
WHEN &type='1'
          diff=2
  OR &type='2'
          diff=3
  OR &type='3'
          diff=5
  OR &type='4'
          diff=2.5
  ELSE
          PRINT Wrong value entered, enter again please.
          GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
PRINT {''}
s=0
PRINT The size must be an even number between 8 and 30.
READ( Enter the current pattern size please. ---> )s
WHEN s=8.OR.s=10.OR.s=12.OR.s=14.OR.s=16.OR.s=18.OR.s=20<#
>.OR.s=22.OR.s=24.OR.s=26.OR.s=28.OR.s=30
          k=(s-6)/2
          WHEN &type='1'
                    basic=NATOWA(k)
            OR &type='2'
                    basic=WATOKN(k)
            OR &type='3'
                    basic=WATOFL(k)
            OR &type='4'
                    basic=SLLNGT(k)
```

```
        ENDWHEN
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 20
ENDWHEN
<#
<#
#30 CONTINUE
PRINT {''}
PRINT Enter the current pattern height<#
> (short, medium or tall) please.
READ( 'S' or 'M' or 'T' ---> )&h
WHEN &h='S'.OR.&h='s'
        dy1=diff/basic
        dy2=diff*2/basic
        &T1='medium'
        &T2='tall'
  OR &h='M'.OR.&h='m'
        dy1=-diff/basic
        dy2=diff/basic
        &T1='short'
        &T2='tall'
  OR &h='T'.OR.&h='t'
        dy1=-diff*2/basic
        dy2=-diff/basic
        &T1='short'
        &T2='medium'
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 30
ENDWHEN
<#
<#
PRINT {''}
PRINT Wait a moment please.
!SELECT LDISCRIMINATION MAGENTA LAYER {lyr+1}
!SELECT LDISCRIMINATION CYAN LAYER {lyr+2}
!ECHO LAYER INCLUDE {lyr+1}
!ECHO LAYER INCLUDE {lyr+2}
!DISCRIMINATE LAYER
EXECDF
<#
<#
PRINT {''}
DIGI(MLOC,NOWAIT,'','  Enter the location for the zero<#
> point please. :') X,Y,Z
PRINT {''}
DIGI(MLOC,NOWAIT,'','  Enter the location for the text<#
> please. :')t1,t2,t3
PRINT {''}
SELECT (LAYER) lyr+1
$T1=TEXT/'height: '+&T1,t1,t2,t3,TJST,'LJT',THGT,1
SELECT (LAYER) lyr+2
$T2=TEXT/'height: '+&T2,t1,t2,t3,TJST,'LJT',THGT,1
SELECT (LAYER) lyr
<#
<#
#200 CONTINUE
```

```
PRINT {''}
&type=''
PRINT   ------------------------------------------<#
>-------------------------
PRINT   Lines -->1     Bspline (curve) -->2    <#
> Points -->3     Notch points -->4
PRINT   ------------------------------------------<#
>-------------------------
READ( Enter the number of the entity type, or<#
> Q to quit. ---> )&type
WHEN &type='1'
        GOSUB 1000
  OR &type='2'
        GOSUB 2000
  OR &type='3'
        GOSUB 3000
  OR &type='4'
        GOSUB 4000
  OR &type='Q'.OR.&type='q'
        PRINT{''}
PRINT Different height of patterns are on<#
> separate layers as follows.
PRINT Height "{&T1}" pattern pieces are on the layer {lyr+1}.
PRINT Height "{&T2}" pattern pieces are on the layer {lyr+2}.
        END
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 200
ENDWHEN
GOTO 200
#999 END
<#
<#
#1000 CONTINUE
PRINT {''}
font=1
PRINT   ------------------------------------------<#
>-------------------------
PRINT   Solid --> 1     Dash --> 2     <#
>Arrow(grain line) --> 3     1-dash --> 4
PRINT   ------------------------------------------<#
>-------------------------
READ( Enter the number of the line font please. <#
>  default:1 ---> )font
WHEN font=1
        &F='SOLID'
  OR font=2
        &F='DASH'
  OR font=3
        &F='ARROW'
  OR font=4
        &F='1-DASH'
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 1000
ENDWHEN
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded<#
```

```
> :')x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 1100: i=1,1,i>n
          x(i,2)=x(i,3)=:x(i,1)
          y(i,2)=y(i,1)+dy1*(y(i,1)-Y)
          y(i,3)=y(i,1)+dy2*(y(i,1)-Y)
          GP(2*(i-1)+1)=VECT(x(i,2),y(i,2),0)
          GP(2*(i-1)+2)=VECT(x(i,3),y(i,3),0)
          IF(i=1) GOTO 1100
          SELECT (LAYER) lyr+1
          GL(2*(i-2)+1)=LINE/GP(2*(i-2)+1),GP(2*(i-1)+1),FONT,&F
          SELECT (LAYER) lyr+2
          GL(2*(i-2)+2)=LINE/GP(2*(i-2)+2),GP(2*(i-1)+2),FONT,&F
#1100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#2000 CONTINUE
PRINT {''}
PRINT          ------------------------------------<#
>----------------------
PRINT                You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ------------------------------------<#
>----------------------
PRINT          * In order to grade a bspline (curve)<#
> accurately, a few points
PRINT            on the current bspline are needed.
PRINT            If you want to insert points on the<#
> current bspline now, use
PRINT            the icon (GENERATE POINT ON N -> : ).
PRINT          * When you are ready for return to the<#
> program, please press
PRINT            keys "Control" and "x" at the same time.
PRINT          ------------------------------------<#
>----------------------
!<VAR>
EXECDF
#2010 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter up to 9 locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
REPEAT 2100: i=1,1,i>n
          x(i,2)=x(i,3)=:x(i,1)
          y(i,2)=y(i,1)+dy1*(y(i,1)-Y)
          y(i,3)=y(i,1)+dy2*(y(i,1)-Y)
          GP(2*(i-1)+1)=VECT(x(i,2),y(i,2),0)
          GP(2*(i-1)+2)=VECT(x(i,3),y(i,3),0)
#2100 CONTINUE
DISP OFF
REPEAT 2200: g=1,1,g>2
          SELECT (LAYER) lyr+g
          WHEN n=3
             GB(g)=BSPLIN/3,GP(g),GP(g+2),GP(g+4)
```

```
            OR n=4
             GB(g)=BSPLIN/3,GP(g),GP(g+2),GP(g+4),GP(g+6)
            OR n=5
             GB(g)=BSPLIN/3,GP(g),GP(g+2),GP(g+4),GP(g+6),GP(g+8)
            OR n=6
             GB(g)=BSPLIN/3,GP(g),GP(g+2),GP(g+4),GP(g+6),<#
>GP(g+8),GP(g+10)
            OR n=7
             GB(g)=BSPLIN/3,GP(g),GP(g+2),GP(g+4),GP(g+6),<#
>GP(g+8),GP(g+10),GP(g+12)
            OR n=8
             GB(g)=BSPLIN/3,GP(g),GP(g+2),GP(g+4),GP(g+6),<#
>GP(g+8),GP(g+10),GP(g+12),GP(g+14)
          OR n=9
           GB(g)=BSPLIN/3,GP(g),GP(g+2),GP(g+4),GP(g+6),<#
>GP(g+8),GP(g+10),GP(g+12),GP(g+14),GP(g+16)
        ENDWHEN
#2200 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#3000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 3100: i=1,1,i>n
        x(i,2)=x(i,3)=:x(i,1)
        y(i,2)=y(i,1)+dy1*(y(i,1)-Y)
        y(i,3)=y(i,1)+dy2*(y(i,1)-Y)
        SELECT (LAYER) lyr+1
        GPT(2*(i-1)+1)=POINT/x(i,2),y(i,2),0
        SELECT (LAYER) lyr+2
        GPT(2*(i-1)+2)=POINT/x(i,3),y(i,3),0
#3100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#4000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 4100: i=1,1,i>n
        x(i,2)=x(i,3)=:x(i,1)
        y(i,2)=y(i,1)+dy1*(y(i,1)-Y)
        y(i,3)=y(i,1)+dy2*(y(i,1)-Y)
        SELECT (LAYER) lyr+1
     GNP(2*(i-1)+1)=CIRCLE/CENTER,x(i,2),y(i,2),0,RADIUS,0.4
     SELECT (LAYER) lyr+2
     GNP(2*(i-1)+2)=CIRCLE/CENTER,x(i,3),y(i,3),0,RADIUS,0.4
#4100 CONTINUE
```

```
DISP ALL
SELECT (LAYER) lyr
RTNSUB
```

```
+-----------------------------------------------------------+
|                                                           |
|        The program for horizontal grading (GDHORIZ)       |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL leng(9),s(10),dx(9),x(100,9),y(100,9),z(100,9)
DECLARE REAL layer(9),size(9)
DECLARE LOCATION GP(1000)
DECLARE ENTITY GL(1000),GB(9),GPT(1000),GNP(1000),T(9)
ERTRAP 999
GETPRM (LAYER) lyr
<#
<#
#10 CONTINUE
s(1)=0
PRINT The size must be an even number between 8 and 30.
READ( Enter the current pattern size please. ---> )s(1)
WHEN s(1)=8.OR.s(1)=10.OR.s(1)=12.OR.s(1)=14.OR.s(1)=16<#
>.OR.s(1)=18.OR.s(1)=20.OR.s(1)=22.OR.s(1)=24.OR.s(1)=26.<#
>OR.s(1)=28.OR.s(1)=30
          CONTINUE
  ELSE
          PRINT Wrong value entered, enter again please.
          GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
REPEAT 30: i=2,1,i>10
          s(i)=0
#30 CONTINUE
PRINT {''}
PRINT ----------------------------------------------------
PRINT * You can grade at most 9 different sizes, which must be
PRINT   even numbers between 8 and 30.
PRINT * Enter required sizes separated by space or comma.
PRINT ----------------------------------------------------
READ, FORMAT = 'FREE', (0) s(2),s(3),s(4),s(5),s(6),s(7),<#
>s(8),s(9),s(10)
j=0
REPEAT 100: i=2,1,i>10
          WHEN s(i)=8.OR.s(i)=10.OR.s(i)=12.OR.s(i)=14.OR.<#
          >s(i)=16.OR.s(i)=18.OR.s(i)=20.OR.s(i)=22.OR.s(i)=24.OR.<#
          >s(i)=26.OR.s(i)=28.OR.s(i)=30
                    j=j+1
                    size(j)=s(i)
                    layer(j)=lyr+j
               OR s(i)=0
                    GOTO 100
               ELSE
                  PRINT Wrong value entered, enter again please.
                    GOTO 20
            ENDWHEN
#100 CONTINUE
<#
<#
```

```
PRINT {''}
PRINT Wait a moment please.
REPEAT 110: i=1,1,i>j
          WHEN i=1
                    &color='MAGENTA'
            OR i=2
                    &color='BLUE'
            OR i=3
                    &color='RED'
            OR i=4
                    &color='GREEN'
            OR i=5
                    &color='CYAN'
            OR i=6
                    &color='GRAY'
            OR i=7
                    &color='YELLOW'
            OR i=8
                    &color='GREEN'
            OR i=9
                    &color='RED'
          ENDWHEN
          !SELECT LDISCRIMINATION {&color} LAYER {layer(i)}
          !ECHO LAYER INCLUDE {layer(i)}
          EXECDF
#110 CONTINUE
!DISCRIMINATE LAYER
EXECDF
<#
<#
#120 CONTINUE
&m=''
PRINT {''}
PRINT          ----------------------------------------<#
>----------------------
PRINT          * For grade rule, one same part horizontal<#
> length (or distance)
PRINT            of the each size have to be entered.
PRINT          * If you need to measure them now,<#
> please press "M",
PRINT            or not, press "RETURN"
PRINT          ----------------------------------------<#
>----------------------
READ(            Press "M" or "RETURN" please. ---> )&m
WHEN &m='M'.OR.&m='m'
PRINT {''}
PRINT          ----------------------------------------<#
>----------------------
PRINT              You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ----------------------------------------<#
>----------------------
PRINT          * Measure the horizontal length<#
> (or distance) of the part of
PRINT            required size using the icon (MEASURE LENGTH).
PRINT            The unit of the length will be "cm".
PRINT          * After Measuring, please press keys<#
> "Control" and "x" at the
```

```
PRINT            same time  to restart the program.
PRINT          ------------------------------------<#
>----------------------
PRINT {''}
PRINT Measure the length of the part of the size {s(1)} please.
!<VAR>
EXECDF
READ( Enter that length please. <unit: cm> ---> )lngt
REPEAT 200: i=1,1,i>j
        PRINT {''}
        PRINT Measure the length of the part of the <#
>size {size(i)} please.
        !<VAR>
        EXECDF
      READ( Enter that length please. <unit: cm> ---> )leng(i)
        k1=(size(i)-6)/2
        k2=(s(1)-6)/2
        dx(i)=(leng(i)-lngt)/lngt
#200 CONTINUE
OR &m=''
PRINT {''}
PRINT Enter the length of the part of the size {s(1)} please.
READ(  ---> ) length
REPEAT 250: i=1,1,i>j
PRINT {''}
PRINT Enter the length of the part of the size {size(i)} please.
        READ(  ---> ) leng(i)
        k1=(size(i)-6)/2
        k2=(s(1)-6)/2
        dx(i)=(leng(i)-length)/length
#250 CONTINUE
ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 120
ENDWHEN
<#
<#
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location of the zero point<#
> please. :') X,Y,Z
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location for text please.<#
> :')t1,t2,t3
PRINT {''}
REPEAT 260: i=1,1,i>j
        SELECT (LAYER) layer(i)
        &size=size(i)
        T(i)=TEXT/'size '+&size,t1,t2,t3,TJST,'LJT',THGT,1
#260 CONTINUE
SELECT (LAYER) lyr
<#
<#
#300 CONTINUE
PRINT {''}
&type=''
PRINT   ----------------------------------------------<#
>--------------------
PRINT   Lines -->1    Bspline (curve) -->2    Points -->3<#
```

```
>     Notch points -->4
PRINT   ----------------------------------------<#
>--------------------
READ( Enter the number of the entity type, or Q to quit.<#
> ---> )&type
WHEN &type='1'
         GOSUB 1000
  OR &type='2'
         GOSUB 2000
  OR &type='3'
         GOSUB 3000
  OR &type='4'
         GOSUB 4000
  OR &type='Q'.OR.&type='q'
         PRINT{''}
         PRINT Every size is on a different layer as follows.
         REPEAT 210: i=1,1,i>j
         WHEN size(i)<10
PRINT Size  {size(i)} pattern pieces are on the layer {layer(i)}.
           ELSE
PRINT Size {size(i)} pattern pieces are on the layer {layer(i)}.
         ENDWHEN
         #210 CONTINUE
         END
  ELSE
         PRINT Wrong value entered, enter again please.
         GOTO 300
ENDWHEN
GOTO 300
#999 END
<#
<#
#1000 CONTINUE
PRINT {''}
font=1
PRINT   ----------------------------------------<#
>----------------------
PRINT   Solid --> 1    Dash --> 2   <#
> Arrow(grain line) --> 3    1-dash --> 4
PRINT   ----------------------------------------<#
>----------------------
READ( Enter the number of the line font please. <#
>  default:1 ---> )font
WHEN font=1
         &F='SOLID'
  OR font=2
         &F='DASH'
  OR font=3
         &F='ARROW'
  OR font=4
         &F='1-DASH'
  ELSE
         PRINT Wrong value entered, enter again please.
         GOTO 1000
ENDWHEN
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded <#
>:')x(1,1),y(1,1),z(1,1)
```

```
PRINT {''}
DISP OFF
REPEAT 1100: i=1,1,i>n
         REPEAT 1200: g=1,1,g>j
                   SELECT (LAYER) layer(g)
                   x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                   y(i,g+1)=y(i,1)
                   GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
                   IF (i>1) GL(j*(i-2)+g)=LINE/GP(j*(i-2)+g),<#
>GP(j*(i-1)+g),FONT,&F
         #1200 CONTINUE
#1100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#2000 CONTINUE
PRINT {''}
PRINT          ----------------------------------------<#
>----------------------
PRINT                You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ----------------------------------------<#
>----------------------
PRINT          * In order to grade a bspline (curve)<#
> accurately, a few points
PRINT            on the current bspline are needed.
 PRINT           If you want to insert points on the<#
> current bspline now, use
PRINT            the icon (GENERATE POINT ON N -> : ).
PRINT          * When you are ready for return to the<#
> program, please press
PRINT            keys "Control" and "x" at the same time.
PRINT          ----------------------------------------<#
>----------------------
!<VAR>
EXECDF
#2010 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter up to 9 locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
REPEAT 2100: i=1,1,i>n
         REPEAT 2150: g=1,1,g>j
                   x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                   y(i,g+1)=y(i,1)
                   GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
         #2150 CONTINUE
#2100 CONTINUE
DISP OFF
REPEAT 2200: g=1,1,g>j
         SELECT (LAYER) layer(g)
         WHEN n=3
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2)
           OR n=4
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),GP(g+j*3)
           OR n=5
```

```
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4)
             OR n=6
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5)
             OR n=7
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6)
             OR n=8
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7)
             OR n=9
                GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7),GP(g+j*8)
          ENDWHEN
#2200 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#3000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 3100: i=1,1,i>n
          REPEAT 3150: g=1,1,g>j
                   SELECT (LAYER) layer(g)
                   x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                   y(i,g+1)=y(i,1)
                   GPT(j*(i-1)+g)=POINT/x(i,g+1),y(i,g+1),0
         #3150 CONTINUE
#3100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#4000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 4100: i=1,1,i>n
          REPEAT 4150:g=1,1,g>j
                   SELECT (LAYER) layer(g)
                   x(i,g+1)=x(i,1)+dx(g)*(x(i,1)-X)
                   y(i,g+1)=y(i,1)
                   GNP(j*(i-1)+g)=CIRCLE/CENTER,x(i,g+1),<#
>y(i,g+1),0,RADIUS,0.4
          #4150 CONTINUE
#4100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
```

```
+-----------------------------------------------------------+
|                                                           |
|          The program for vertical grading (GDVERT)        |
|                                                           |
+-----------------------------------------------------------+
<#
<#
DECLARE REAL leng(9),s(10),dy(9),x(100,9),y(100,9),z(100,9)
DECLARE REAL layer(9),size(9)
DECLARE LOCATION GP(1000)
DECLARE ENTITY GL(1000),GB(9),GPT(1000),GNP(1000),T(9)
ERTRAP 999
GETPRM (LAYER) lyr
<#
<#
#10 CONTINUE
s(1)=0
PRINT The size must be an even number between 8 and 30.
READ( Enter the current pattern size please. ---> )s(1)
WHEN s(1)=8.OR.s(1)=10.OR.s(1)=12.OR.s(1)=14.OR.s(1)=16<#
>.OR.s(1)=18.OR.s(1)=20.OR.s(1)=22.OR.s(1)=24.OR.<#
>s(1)=26.OR.s(1)=28.OR.s(1)=30
          CONTINUE
  ELSE
          PRINT Wrong value entered, enter again please.
          GOTO 10
ENDWHEN
<#
<#
#20 CONTINUE
REPEAT 30: i=2,1,i>10
          s(i)=0
#30 CONTINUE
PRINT {''}
PRINT ----------------------------------------
PRINT * You can grade at most 9 different sizes, which must be
PRINT   even numbers between 8 and 30.
PRINT * Enter required sizes separated by space or comma.
PRINT ----------------------------------------
READ, FORMAT = 'FREE', (0) s(2),s(3),s(4),s(5),s(6)<#
>,s(7),s(8),s(9),s(10)
j=0
REPEAT 100: i=2,1,i>10
          WHEN s(i)=8.OR.s(i)=10.OR.s(i)=12.OR.s(i)=14.OR<#
          >.s(i)=16.OR.s(i)=18.OR.s(i)=20.OR.s(i)=22.OR.s(i)=24.OR.<#
          >s(i)=26.OR.s(i)=28.OR.s(i)=30
                    j=j+1
                    size(j)=s(i)
                    layer(j)=lyr+j
               OR s(i)=0
                    GOTO 100
               ELSE
                    PRINT Wrong value entered, enter again please.
                    GOTO 20
          ENDWHEN
#100 CONTINUE
<#
<#
```

```
PRINT {''}
PRINT Wait a moment please.
REPEAT 110: i=1,1,i>j
        WHEN i=1
                &color='MAGENTA'
          OR i=2
                &color='BLUE'
          OR i=3
                &color='RED'
          OR i=4
                &color='GREEN'
          OR i=5
                &color='CYAN'
          OR i=6
                &color='GRAY'
          OR i=7
                &color='YELLOW'
          OR i=8
                &color='GREEN'
          OR i=9
                &color='RED'
        ENDWHEN
        !SELECT LDISCRIMINATION {&color} LAYER {layer(i)}
        !ECHO LAYER INCLUDE {layer(i)}
        EXECDF
#110 CONTINUE
!DISCRIMINATE LAYER
EXECDF
<#
<#
#120 CONTINUE
&m=''
PRINT {''}
PRINT          ------------------------------------<#
>----------------------
PRINT          * For grade rule, one same part vertical<#
> length (or distance)
PRINT            of the each size have to be entered.
PRINT          * If you need to measure them now, <#
>please press "M",
PRINT            or not, press "RETURN"
PRINT          ------------------------------------<#
>----------------------
READ(            Press "M" or "RETURN" please. ---> )&m
WHEN &m='M'.OR.&m='m'
PRINT {''}
PRINT          ------------------------------------<#
>----------------------
PRINT              You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ------------------------------------<#
>----------------------
PRINT          * Measure the vertical length<#
> (or distance) of the part of
PRINT            required size using the icon (MEASURE LENGTH).
PRINT            The unit of the length will be "cm".
PRINT          * After Measuring, please press keys<#
> "Control" and "x" at the
```

```
PRINT              same time  to restart the program.
PRINT          ------------------------------------<#
>-----------------------
PRINT {''}
PRINT Measure the length of the part of the size {s(1)} please.
!<VAR>
EXECDF
READ( Enter that length please. <unit:cm> ---> )lngt
REPEAT 200: i=1,1,i>j
         PRINT {''}
PRINT Measure the length of the part of the size {size(i)} please.
         !<VAR>
         EXECDF
         READ( Enter the length please. <unit:cm> ---> )leng(i)
         k1=(size(i)-6)/2
         k2=(s(1)-6)/2
         dy(i)=(leng(i)-lngt)/lngt
#200 CONTINUE
OR &m=''
PRINT {''}
PRINT Enter the length of the part of the size {s(1)} please.
READ(  ---> ) length
REPEAT 250: i=1,1,i>j
         PRINT {''}
PRINT Enter the length of the part of the size {size(i)} please.
         READ(  ---> ) leng(i)
         k1=(size(i)-6)/2
         k2=(s(1)-6)/2
         dy(i)=(leng(i)-length)/length
#250 CONTINUE
ELSE
         PRINT Wrong value entered, enter again please.
         GOTO 120
ENDWHEN
<#
<#
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location of the zero<#
> point please. :') X,Y,Z
PRINT {''}
DIGI(MLOC,NOWAIT,'',' Enter the location for text <#
>please. :')t1,t2,t3
PRINT {''}
REPEAT 260: i=1,1,i>j
         SELECT (LAYER) layer(i)
         &size=size(i)
         T(i)=TEXT/'size '+&size,t1,t2,t3,TJST,'LJT',THGT,1
#260 CONTINUE
SELECT (LAYER) lyr
<#
<#
#300 CONTINUE
PRINT {''}
&type=''
PRINT   ------------------------------------------<#
>-----------------------
PRINT   Lines -->1     Bspline (curve) -->2    <#
> Points -->3     Notch points -->4
```

```
PRINT   ----------------------------------------<#
>------------------------
READ( Enter the number of the entity type,<#
> or Q to quit. ---> )&type
WHEN &type='1'
        GOSUB 1000
  OR &type='2'
        GOSUB 2000
  OR &type='3'
        GOSUB 3000
  OR &type='4'
        GOSUB 4000
  OR &type='Q'.OR.&type='q'
        PRINT{''}
        PRINT Every size is on a different layer as follows.
        REPEAT 210: i=1,1,i>j
        WHEN size(i)<10
PRINT Size  {size(i)} pattern pieces are on the layer {layer(i)}.
          ELSE
PRINT Size {size(i)} pattern pieces are on the layer {layer(i)}.
        ENDWHEN
        #210 CONTINUE
        END
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 300
ENDWHEN
GOTO 300
#999 END
<#
<#
#1000 CONTINUE
PRINT {''}
font=1
PRINT   ----------------------------------------<#
>------------------------
PRINT   Solid --> 1    Dash --> 2   <#
> Arrow(grain line) --> 3    1-dash --> 4
PRINT   ----------------------------------------<#
>------------------------
READ( Enter the number of the line font please. <#
>  default:1 ---> )font
WHEN font=1
        &F='SOLID'
  OR font=2
        &F='DASH'
  OR font=3
        &F='ARROW'
  OR font=4
        &F='1-DASH'
  ELSE
        PRINT Wrong value entered, enter again please.
        GOTO 1000
ENDWHEN
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
```

```
DISP OFF
REPEAT 1100: i=1,1,i>n
         REPEAT 1200: g=1,1,g>j
                  SELECT (LAYER) layer(g)
                  x(i,g+1)=x(i,1)
                  y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                  GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
                  IF (i>1) GL(j*(i-2)+g)=LINE/GP(j*(i-2)+g)<#
>,GP(j*(i-1)+g),FONT,&F
         #1200 CONTINUE
#1100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#2000 CONTINUE
PRINT {''}
PRINT          ------------------------------------<#
>--------------------------
PRINT                You can use the icon tool,<#
> while the prompt is "<VAR>".
PRINT          ------------------------------------<#
>--------------------------
PRINT          * In order to grade a bspline (curve)<#
> accurately, a few points
PRINT            on the current bspline are needed.
PRINT            If you want to insert points on the<#
> current bspline now, use
PRINT            the icon (GENERATE POINT ON N -> : ).
PRINT          * When you are ready for return to the<#
> program, please press
PRINT            keys "Control" and "x" at the same time.
PRINT          ------------------------------------<#
>--------------------------
!<VAR>
EXECDF
#2010 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter up to 9 locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
REPEAT 2100: i=1,1,i>n
         REPEAT 2150: g=1,1,g>j
                  x(i,g+1)=x(i,1)
                  y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                  GP(j*(i-1)+g)=VECT(x(i,g+1),y(i,g+1),0)
         #2150 CONTINUE
#2100 CONTINUE
DISP OFF
REPEAT 2200: g=1,1,g>j
         SELECT (LAYER) layer(g)
         WHEN n=3
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2)
           OR n=4
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),GP(g+j*3)
           OR n=5
              GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
```

```
>GP(g+j*3),GP(g+j*4)
            OR n=6
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5)
            OR n=7
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6)
            OR n=8
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7)
            OR n=9
               GB(g)=BSPLIN/3,GP(g),GP(g+j),GP(g+j*2),<#
>GP(g+j*3),GP(g+j*4),GP(g+j*5),GP(g+j*6),GP(g+j*7),GP(g+j*8)
         ENDWHEN
#2200 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#3000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 3100: i=1,1,i>n
         REPEAT 3150: g=1,1,g>j
                  SELECT (LAYER) layer(g)
                  x(i,g+1)=x(i,1)
                  y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                  GPT(j*(i-1)+g)=POINT/x(i,g+1),y(i,g+1),0
         #3150 CONTINUE
#3100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
<#
<#
#4000 CONTINUE
PRINT {''}
DIGI(MLOC,100,n,'',' Enter locations to be graded :')<#
>x(1,1),y(1,1),z(1,1)
PRINT {''}
DISP OFF
REPEAT 4100: i=1,1,i>n
         REPEAT 4150:g=1,1,g>j
                  SELECT (LAYER) layer(g)
                  x(i,g+1)=x(i,1)
                  y(i,g+1)=y(i,1)+dy(g)*(y(i,1)-Y)
                  GNP(j*(i-1)+g)=CIRCLE/CENTER,x(i,g+1)<#
>,y(i,g+1),0,RADIUS,0.4
         #4150 CONTINUE
#4100 CONTINUE
DISP ALL
SELECT (LAYER) lyr
RTNSUB
```