/AUTOMATION OF THE MODULAR PATTERN SYSTEM BASIC SKIRT PATTERN DRAFTING METHODOLOGY USING TURBO PASCAL AND DBASEII/

by

MIRIAM SHAHEED CLARK

B.S., KANSAS STATE UNIVERSITY, 1982

A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1986

Approved by:

Major Professor

Acknowledgements

I wish to express my appreciation and thanks to Dr. Roger T. Hartley, for discussing ideas and providing suggestions during the course of the implementation project. My sincere gratitude is extended to Dr. Elizabeth B. Unger, for her guidance and assistance as my major professor. Thanks are also extended to Dr. Richard A. McBride and Dr. Austin C. Melton, Jr. for serving as the advisory committee members.

Special thanks are extended to Professor Emeritus Helen Brockman for providing an exciting project and for her guidance.

I would like to thank my parents Fahim and Christa Shaheed and my husband Timothy John Clark for their continuous love and support during the course of my education.

THIS BOOK CONTAINS NUMEROUS PAGES WITH DIAGRAMS THAT ARE CROOKED COMPARED TO THE REST OF THE INFORMATION ON THE PAGE.

THIS IS AS RECEIVED FROM CUSTOMER.

THIS BOOK CONTAINS NUMEROUS PAGE NUMBERS THAT ARE ILLEGIBLE

THIS IS AS RECEIVED FROM THE CUSTOMER

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

page

Figure

Chapter 1

Review of literature and statement of problem

1.1 Broad perspective

In the past, clothing was made at home by hand. Thus, it was made to fit the wearer. With the advent of modern manufacturing techniques, clothing started being mass produced. Patterns used to sew one's own clothing at home were also being mass produced.

Today's standard system of sizing was developed during World War II. Enormous amounts of uniforms were needed for the troops. Measurements were taken to mass produce these uniforms. The measurements were compiled, analyzed, and used as a basis for today's standard sizes. Since then, no such large-scale analysis of the human figure has been done.

Currently there is a problem with the fit of ready-made patterns and ready-to-wear clothing. They don't fit a majority of the people, because people don't tend to come in standard sizes [bro 86]. This is one of the main reasons for the current popularity of knit fabrics. Where the garment does not fit properly, the fabric gives. The fashion and garment industry have also tried to alleviate the fitting problem by creating clothing with loose-fitting silhouettes. These approaches simply avoid the fitting problem. The pattern industry has attempted to solve the problem by offering patterns with a range of three different sizes. The rationale be-

ing that you may be a size 12 at the bust, a size 14 at the waist, and a size 16 at the hips. Unfortunately, a person's measurements may not fall within the size range offered, and there are many other variables which must be taken into consideration for a proper fit, besides circumference.

Helen Brockman, Professor Emeritus at Kansas State University, has developed a pattern drafting method that will produce a pattern that fits exactly. By gathering empirical data on over 800 women, Brockman has developed this methodology for a basic skirt pattern, a basic bodice pattern, and some design variations of the basic skirt and bodice. She is currently developing pattern drafting methods for the basic overblouse and some variations.

This methodology requires expertise in pattern-making and consumes time that today's career woman can not afford. Helen feels that, in order for this method to benefit the woman of today, this pattern-drafting method should be automated and made easily available.

1.2 History

In the past, several companies have sold computer produced clothing patterns. These included Surefit, Compusize by Uno, and Fashion Futures of Seattle, Washington. They are no longer in business, probably because their patterns simply did not fit [bro 86]. There are some current attempts and successes at automation of various pattern drafting methodologies. Laura Varney, Ph.D. and

her son Douglas Varney, have completed and are currently marketing the basic pants, skirt, and bodice patterns through their company, Clothing Design Concepts [var 86]. Their drafting method has been implemented in Basic and is being run on an IBM personal computer. The patterns are being drawn on a Houston Instruments plotter. The patterns are marketed by mail through the Vogue pattern books. Laura checks and adjusts each pattern individually, before it goes out in the mail. Although the details of pattern drafting methodology are not known to this author, Clothing Design Concepts appears to be quite successful in the market place.

With at least one successful computerized, individualized pattern effort available by mail; it seems reasonable to explore the probability that such a system could be made available more conveniently.

In order for the automated pattern drafting system to be of any value, it must be easily available to a large percentage of the population. In many ways, the ideal place to set such a system up, would be a piece goods store. Trained personnel would take the proper measurements. The numbers could then be punched in at the terminal and a pattern would be drawn on the plotter. Since most people's bodies are not symmetrical, alterations would probably still have to be made to the pattern at this point to further customize it for a particular individual. To avoid errors during the alteration phase, there should be a separate room with muslin and sewing machines, where the customer could sew a muslin using the

pattern. After the muslin is sewn, the trained personnel would make any needed alteration and enter those alterations to produce a corrected pattern. Once a correct basic pattern is achieved, the customer could choose from the style variations offered and a pattern with that styling could be plotted.

The system should consist of a personal computer and a plotter. The use of a personal computer, rather than a mainframe, would allow a financially feasible on-site system. The use of the plotter would allow immediate results, in the form of a printed pattern.

In the past, there were too many ambiguities left up to the judgement of the pattern maker, which made it impossible to automate pattern drafting. Brockman has removed all of these ambiguities by creating tables which contain unambiguous information needed to draft a pattern. These tables provide the key to the development of an automated pattern drafting program.

1.3 Scope of the project

The scope of this research project is automation of the basic skirt drafting method using Turbo Pascal. The front and back of the basic skirt pattern will be simulated on the screen. Since Brockman continues to modify and improve her pattern drafting methodology; this project is limited to the implementation of the method as it existed in January of 1985, when this research was commenced.

1.4 Overview

This paper is broken up into four chapters. Chapter one contains a review of the related work, a statement of the problem, and the scope of the research project. Chapter two covers the specific problem from the point of view of the pattern maker. Brockman's entire drafting process is explained, beginning with how certain measurements are taken and transformed to the needed information. Also included are tables of information used during the pattern drafting process. Chapter three contains an overview of the program design, a hierarchy diagram showing the calling structure, and a table of the inputs and outputs for each module. Chapter four discusses what the program does and possible improvements. The appendix contains a user's manual and the source code.

Chapter 2

Drafting the Basic Skirt Pattern Using the Mod-u-lar Pattern System

The Mod-u-lar Pattern System is the methodology developed by Professor Emeritus Helen Brockman. Brockman likened the system to the concept of modular furniture, since "the user chooses separate parts and assembles them". [bro 85] This chapter will cover the steps for drafting by hand a basic skirt pattern using the Mod-u-lar Pattern System. The basic skirt is not meant to be worn, but is rather the "blueprint" *from which* various skirt styles may be developed.

This chapter describes in detail that system and its usage as a hand methodology. Chapter 3 uses this work to create a computer system for the production of computerized patterns.

2.1 Necessary Supplies for Drafting a Basic Skirt Pattern

Listed below are the supplies and tools needed to draft a basic skirt pattern. These must be available before the pattern drafting process can begin.

plastic #60 curve
metric aluminum hipcurve
scissors or razor blades
metric tape measure
tracing paper
magic tape
H8 pencil
straight pins
yardstick
rubberband
men's medium undershirt
muslin
skirt template & worksheet (Furnished with the Skirt Book)

SKIRT WORKSHEET for_____

	BODY MEASUREMENTS	
1.	HEIGHT (without shoes)	
2.	WEIGHT (without undergarments)	
3.	TORSOLINE GIRTH	
4.	WAISTLINE GIRTH	
5.	DIFFERENTIAL (TL - WL)	
6.	CENTER-BACK LENGTH (WL to TL)	
	PATTERN TEMPLATE INFORMATION	
7.	SIDESEAM NUMBER: 1 2 3 4 5 6 7	
8.	PATTERN WIDTH AT WAISTLINE	
9.	PATTERN WIDTH AT HIPLINE	
	DART TEMPLATE SELECTION	
10.	BODYTYPE: (A) (B) (C)	
11.	SUBTYPE: (-) (=) (+) (^)	
12.	DART TEMPLATE PAGE NUMBER	
	FINAL DRAFTING VERIFICATION	
13.	BACK WAISTLINE (with darts closed)	
14.	FRONT WAISTLINE (with darts closed)	
15.	COMBINED BACK AND FRONT WAISTLINES	
16.	MEASURED WL GIRTH / 2 (in mm)	
17.	WL EASE	

Figure 2-1. The skirt worksheet

2.2 The Skirt Worksheet

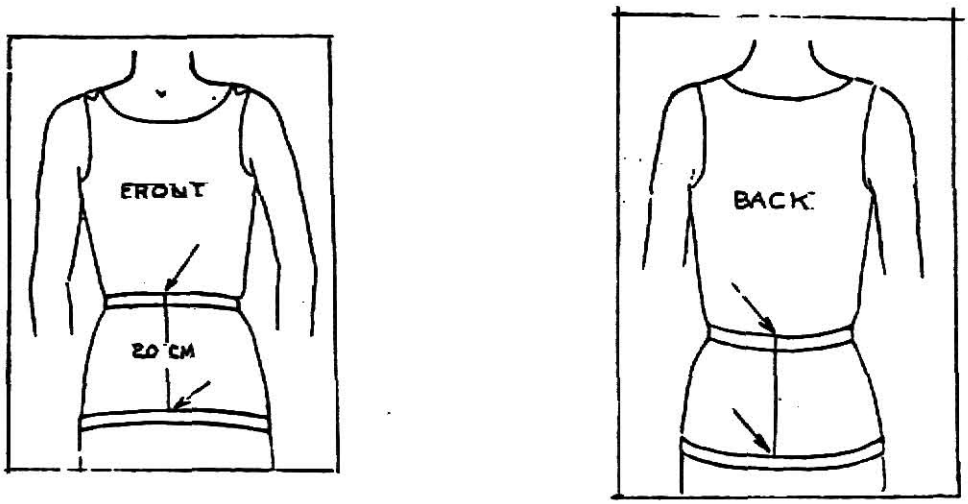
The skirt worksheet, shown in figure 2-1, is used to make the process of taking body measurements and calculating pattern measurements easier. Measurements 1, 2, 3, 4, and 6 of the skirt worksheet are the five measurements that need to be taken on the body. The remaining entries are calculated from these five body measurements.

2.2.1 Preparation for Measurement

In order that the resulting pattern fits properly, it is essential that the measurements be taken by a trained person to ensure correctness.

Several items are needed in preparation for body measurement to facilitate the process. First, two one-inch wide muslin bands are needed. One should be long enough to encircle the waist and the other should be slightly longer than the hip circumference. A body shirt is also needed to provide a smooth measuring surface. A men's undershirt, with opened shoulder seams, is needed. The undershirt is to be put on, over any undergarments normally worn, and pinned at the shoulder as shown in figure 2-2(a). The two sturdy bands are now needed. The waistband should be pinned snugly around the waist and the torsoband should be pinned 20 centimeters down from the centerfront of the waist. This torsoband must be parallel to the floor. A yardstick or meterstick may be used to measure the distance from the front torsoband to the floor. A rubberband may

be used to mark that distance. The back torsoband should be pinned so that it is the same distance from the floor. Preparations are now in order for body measurement to begin.



(a) (b)

Figure 2-2. Body measurements (a) front (b) back

2.2.2 Taking the Body Measurements

Five body measurements are needed in order to calculate all the measurements to draft the basic skirt. These are the height, weight, waistline girth, torsoline girth, and centerback length.

Body height is to be recorded without shoes. The height must be known in centimeters or be converted to centimeters from another system, e.g., feet and inches (to convert height into centimeters, determine the height in inches and multiply by 2.54).

Record the body weight in pounds with the undergarments on.

The torsoline girth measurement is to be taken with the tape

measure firmly around the torsoband. A metric tape measure must be used for both this and the remaining measurements. Round the resulting measurement to the nearest centimeter.

The waistline girth measurement should be adjusted to be smaller than the torsoband girth by an even number of centimeters.

The final body measurement to be recorded is the centerback length. This is done by measuring at the centerback from the top of the waistband to the top of the torsoband. See figure 2-2(b).

2.2.3 Transforming the Body Measurements

Entries 5, 7, 8, 9, 10, 11 and, 12 on the skirt worksheet are determined through the combined use of the input measurements and lookup tables.

5. The differential is calculated by subtracting the waistline girth from the torsoline girth. The differential sizes range in even numbers from 14 to 40.

7. The sideseam number is determined by looking up the associated differential in the pattern width table, see appendix D. The sideseam number ranges from one to seven.

8. The pattern width at the waistline can be found by differential and waistline girth in the pattern width at the waistline table.

9. The pattern width at the hipline can be found by differential and waistline girth in the pattern width at the hipline table, see appendix D.

10. The bodytype is determined by the difference between the centerfront and centerback length. Bodytype "A" has a long front and a short back. Bodytype "B" has a normal front and a normal back; less than one centimeter's difference. Bodytype "C" has a short front and a long back.

11. Each bodytype has three possible subtypes. If the bodytype is "B", then the subtype is determined by weight. Subtype "-" is at least 15 pounds underweight, subtype "+" is at least 20 pounds overweight, and subtype "=" is between 15 pounds under and 20 pounds over the normal weight. The normal weights depending on height are given in the height/weight table, see appendix D. If the bodytype is "A" or "C", then the subtype is determined by the amount of difference between the front and back length. Subtype "=" is a one centimeter's difference, subtype "+" is a two centimeter's difference, and subtype "^" is a three centimeter's difference.

12. The dart template page number can be found in the lookup table shown in figure 2-3. It is determined by bodytype, subtype, and differential.

Items 13 to 17 on the skirt worksheet are for drafting verification of the final pattern. This step is beyond the scope of the implementation project as the skirt pattern will only be simulated to the screen.

2.2.4 An Example of Transforming the Body Measurements

This example will illustrate the methods used to derive the skirt worksheet entries from five body measurements. Assume, for this

BODY TYPE	SIDE SEAM	DF SIZE	DART TEMPLATE PAGE NUMBERS			
			(-)	(=)	(+)	(^)
A	1	14 / 16		64	71	78
	2	18 / 20		65	72	79
	3	22 / 24		66	73	80
	4	26 / 28		67	74	81
	5	30 / 32		68	75	82
	6	34 / 36		69	76	83
	7	38 / 40		70	77	
B	1	14 / 16	84	88	95	
	2	18 / 20	85	89	96	
	3	22 / 24	86	90	97	
	4	26 / 28	87	91	98	
	5	30 / 32		92	99	
	6	34 / 36		93	100	
	7	38 / 40		94	101	
C	1	14 / 16		102	109	
	2	18 / 20		103	110	
	3	22 / 24		104	111	116
	4	26 / 28		105	112	117
	5	30 / 32		106	113	118
	6	34 / 36		107	114	119
	7	38 / 40		108	115	120

Figure 2-3. Dart template page number table

example, that the given measurements are items 1, 2, 3, 4, and 6, shown in figure 2-4. The combined use of these input measurements and lookup tables determines entries 5, 7, 8, 9, 10, 11, and 12

SKIRT WORKSHEET for_____Example_____

	BODY MEASUREMENTS	
1.	HEIGHT (without shoes)	5'2" or 158 cm
2.	WEIGHT (without undergarments)	118 lbs
3.	TORSOLINE GIRTH	93 cm
4.	WAISTLINE GIRTH	67 cm
5.	DIFFERENTIAL (TL - WL)	26 cm
6.	CENTER-BACK LENGTH (WL to TL)	19.5 cm

	PATTERN TEMPLATE INFORMATION	
7.	SIDESEAM NUMBER: 1 2 3 4 5 6 7	4
8.	PATTERN WIDTH AT WAISTLINE	227 cm
9.	PATTERN WIDTH AT HIPLINE	246 cm

	DART TEMPLATE SELECTION	
10.	BODYTYPE: (A) (B) (C)	B
11.	SUBTYPE: (-) (=) (+) (^)	=
12.	DART TEMPLATE PAGE NUMBER	91

	FINAL DRAFTING VERIFICATION	
13.	BACK WAISTLINE (with darts closed)	
14.	FRONT WAISTLINE (with darts closed)	
15.	COMBINED BACK AND FRONT WAISTLINES	
16.	MEASURED WL GIRTH / 2 (in mm)	
17.	WL EASE	

Figure 2-4. A skirt worksheet example

of this same worksheet. All steps will be numbered by the corresponding worksheet number for clarity.

4. The first step is to verify that the waistline girth (67 cm) is smaller than the torsoline girth (93 cm) by an even number. If the difference between the two is odd, then 1 cm is subtracted from the waistline. In this case, the waistline girth needs no adjustment.

5. The differential is calculated by subtracting the waistline girth (67 cm) from the torsoline girth (93 cm). The result is a differential of 26 cm.

7. The sideseam number is determined by looking up the associated differential (26 cm) in the pattern width table, see appendix D. The resulting sideseam number is 4.

8. The pattern width at the waistline can be found by differential (26 cm) and waistline girth (67 cm) in the pattern width at the waistline table. The resulting pattern width at the waistline is 227 mm.

9. The pattern width at the hipline can be found by differential (26 cm) and hipline girth (93 cm) in the pattern width at the hipline table. The resulting pattern width at the hipline is 246 mm.

10. The bodytype is determined by the difference between the centerfront (20 cm) and centerback length (19.5 cm). The resulting difference is .5 cm; less than one centimeter's difference. The bodytype is "B" with a normal front and a normal back.

11. Each bodytype has three possible subtypes. If the bodytype is "B", then the subtype is determined by deviation from normal weight. The normal weights depending on height are given in the

height/weight table, see appendix D. According to this table, the normal weight for a height of 158cm and a differential of 26cm is 117 pounds. The input weight was 118 pounds; 1 pound over the norm. Weights 15 pounds under to 20 pounds over fall into the Subtype "=" category.

12. The dart template page number can be found in the lookup table shown in figure 2-3. It is determined by bodytype, subtype, and differential. The dart template for bodytype "B", subtype "=", and differential 26 may be found on page 91 of the Mod-u-lar Pattern System skirt book.

2.3 Drafting the Pattern

2.3.1 Drawing the Back Pattern Framework

A pattern template is provided with the Skirt Book to facilitate drawing the guidelines and the sideseams. The back pattern framework consists of five guidelines; the waistline, control line, and hipline, which are parallel, and the centerline, which is perpendicular. These guidelines are simply traced from the template to produce the back pattern template. Then the pattern width at the waistline(figure 2-1.8) and the pattern width at the hipline(figure 2-1.9) are marked on the waistline and hipline guidelines, respectively.

2.3.2 Drawing the Sideseam Line

The pattern template, shown in figure 2-5, is also used to draw

the sideseam line. Seven different sideseam templates are provided. Each has a corresponding number. In order to determine the appropriate sideseam, match with the corresponding number on skirt worksheet item number seven.

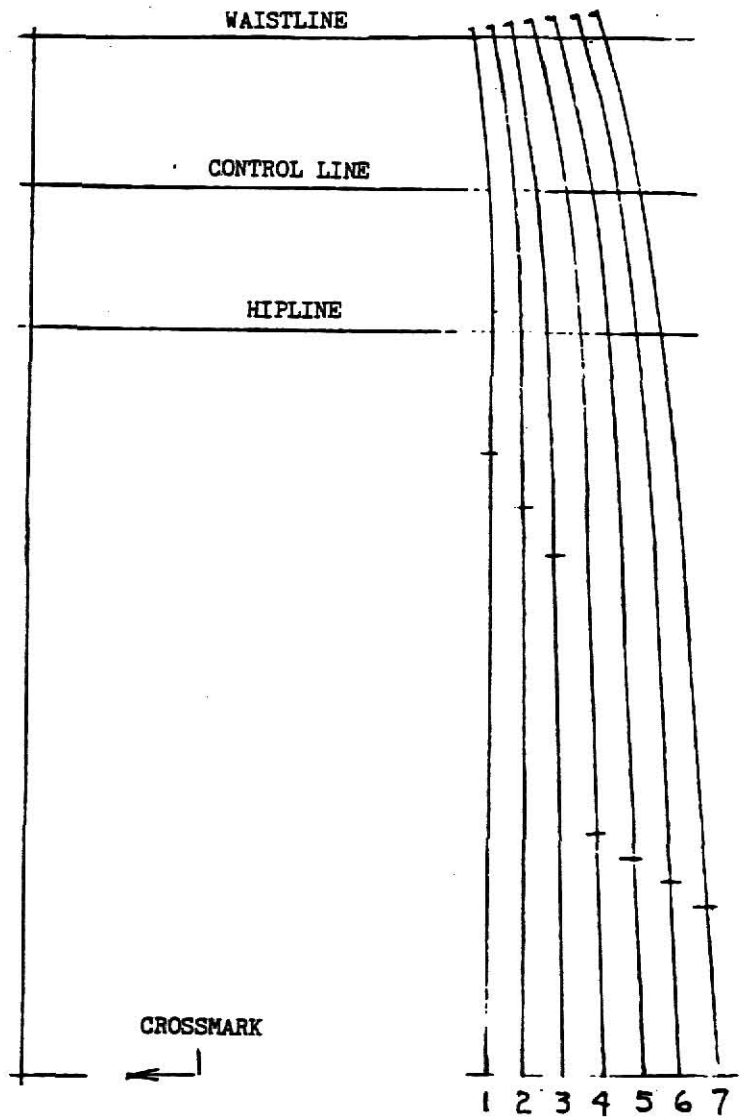


Figure 2-5. Skirt pattern template

In order to trace the appropriate sideseam, first align the tracing paper over the sideseam template. The horizontal guidelines should be aligned and the sideseam line should intersect at the waistline pattern width marked previously. Sideseam 1,2, and 3 may

be drawn using the plastic #60 curve and sideseams 4, 5, 6, and 7 can be drawn with the metal hipcurve to the matchmark on the template. The line below the match point may be drawn using a straight ruler as it is a straight line.

2.3.3 Marking the Locations of the Dart Center Lines

There are two darts on the back basic skirt pattern; the one closest to the centerline is known as the panel dart and the other, which is closer to the sideseam, is known as the side dart. The center line of the panel dart is a third of the distance from the centerback line and the sideseam line on the waistline, shown in figure 2-6. The side dart's center line is the midpoint of the distance between the panel dart center and the sideseam line on the waistline.

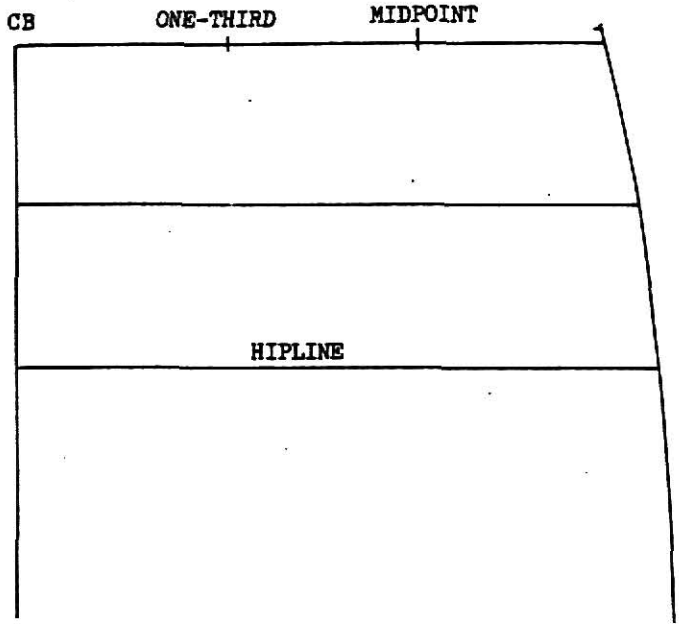


Figure 2-6. Location of the dart center lines

2.3.4 Drawing the Front Pattern Framework

The front pattern draft is developed on the lower half of the back pattern, since it will be a mirror image of the back pattern at this point. In preparation, the pattern must be turned face down, folded in half with the centerline folded up on itself and the bottom crossmark matching the waistline, and pinned in position. The waistline, control line, sideseam line, and dart center marks may now be copied from the back pattern framework. The resulting front pattern, shown in figure 2-7, will be a mirror image of the back, since the differences between the front and back pattern will not occur until the darts and waistline are added.

2.3.5 Drawing the Pattern Darts

A dart template is provided with the Skirt Book to facilitate drawing the basic skirt pattern's darts. The appropriate template is determined by bodytype and differential size. The dart template is used to trace the four waistline darts and the centerfront and centerback waistline levels.

2.3.5.1 Dart Center Lines

The dart center lines for the front and back pattern are drawn first. The center line for the front and back panel darts is drawn by connecting the previously marked panel crossmarks. The center line for the front and back side darts is drawn by connecting the previously marked side crossmarks.

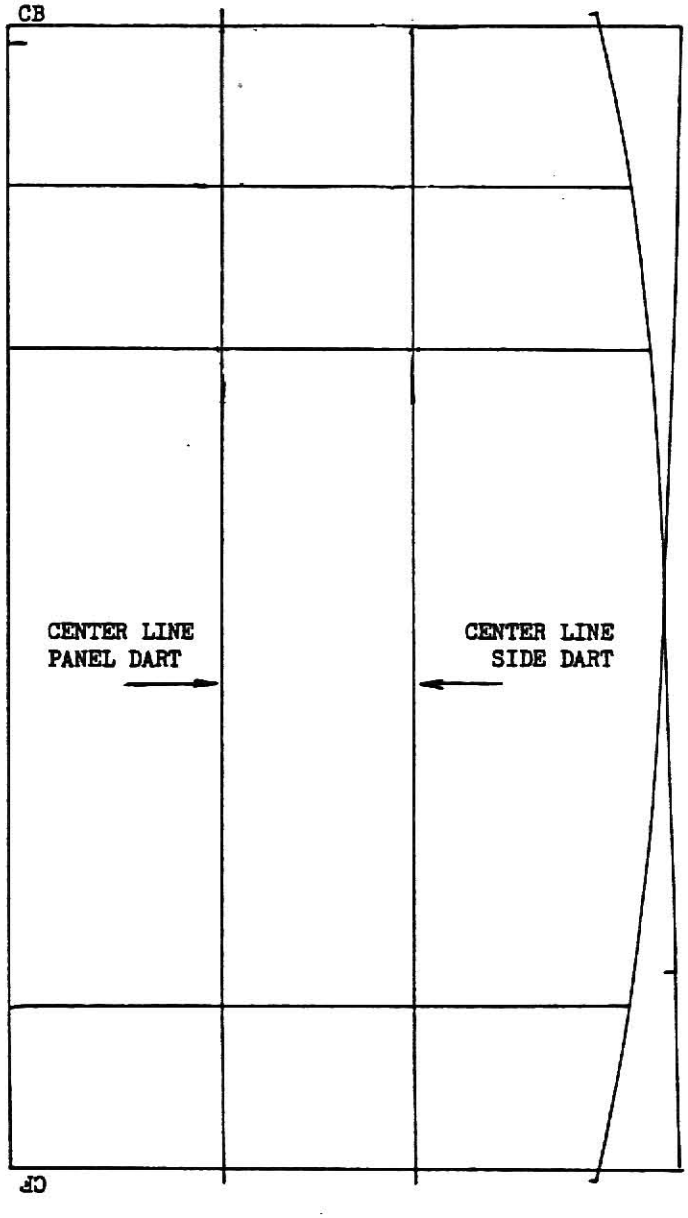


Figure 2-7. Front and back pattern with dart center lines

2.3.5.2 Back Pattern Darts

The back skirt pattern is placed over the back dart template to draft the back pattern darts. The waistline and control line on

the pattern must always be matched to the waistline and control line on the template.

First, the center back level is marked, if it is different from the waistlevel. This is accomplished by matching the center back lines on the skirt pattern and dart template.

In order to draw the back panel dart, the pattern is shifted to the left so its panel dart center line matches that on the template. First the top line of dart is traced. Then the horizontal line at tip of dart is traced over to the side dart center line. Then the dart is drawn using a ruler, as the back panel dart is always straight.

In order to draw the back side dart, the pattern is shifted to the left so its side dart center line matches that on the template. First the top line of dart is traced. Then the straight portion of the dart is drawn using a ruler, and the curved portion is drawn with a #60 curve.

2.3.5.3 Front Pattern Darts

The front skirt pattern is placed over the front dart template to draft the front pattern darts. The waistline and control line on the pattern must always be matched to the waistline and control line on the template.

First, the center front level is marked, if it is different from the waistlevel. This is accomplished by matching the center front lines on the skirt pattern and dart template.

In order to draw the front panel dart, the pattern is shifted to the right so its panel dart center line matches that on the template. First the top line of dart is traced. Then the horizontal line at tip of dart is traced over to the side dart center line. The front panel dart may be curved or straight. The straight part of the dart is drawn using a ruler. If there is a curved portion, it is copied using a #60 curve.

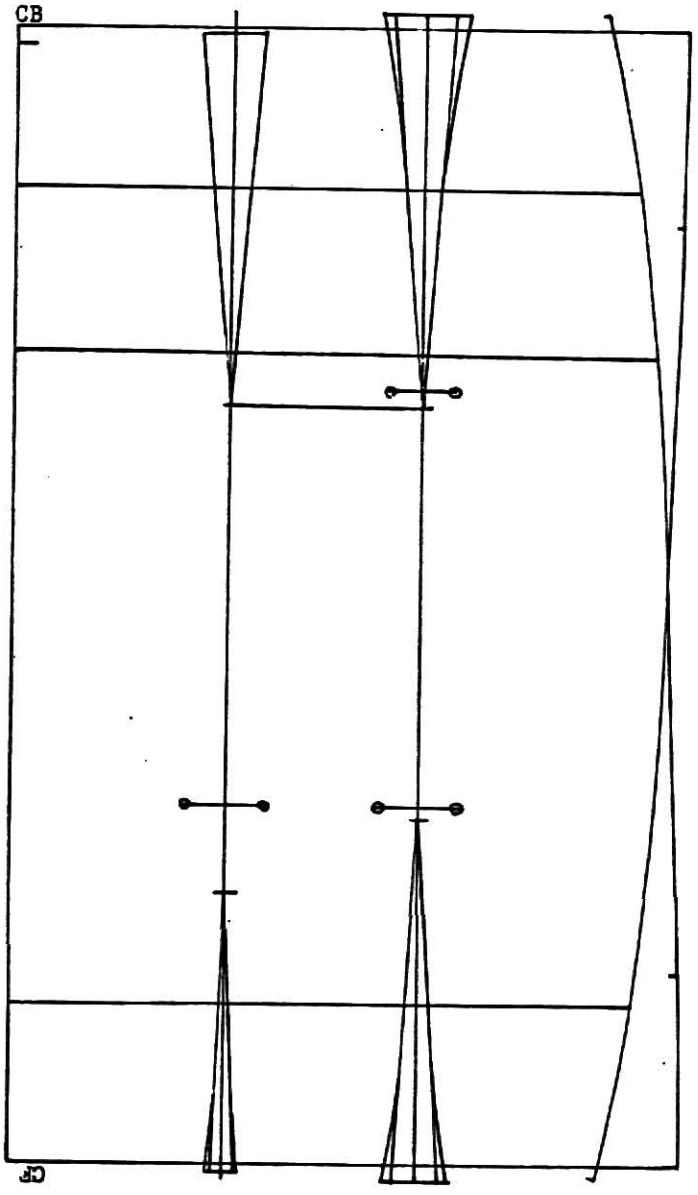


Figure 2-8. Front and back pattern with darts

In order to draw the back side dart, the pattern is shifted to the right so its side dart center line matches that on the template. First the top line of dart is traced. Then the straight portion of the dart is drawn using a ruler, and the curved portion is drawn with a #60 curve. The resulting combined front and back pattern is shown in figure 2-8.

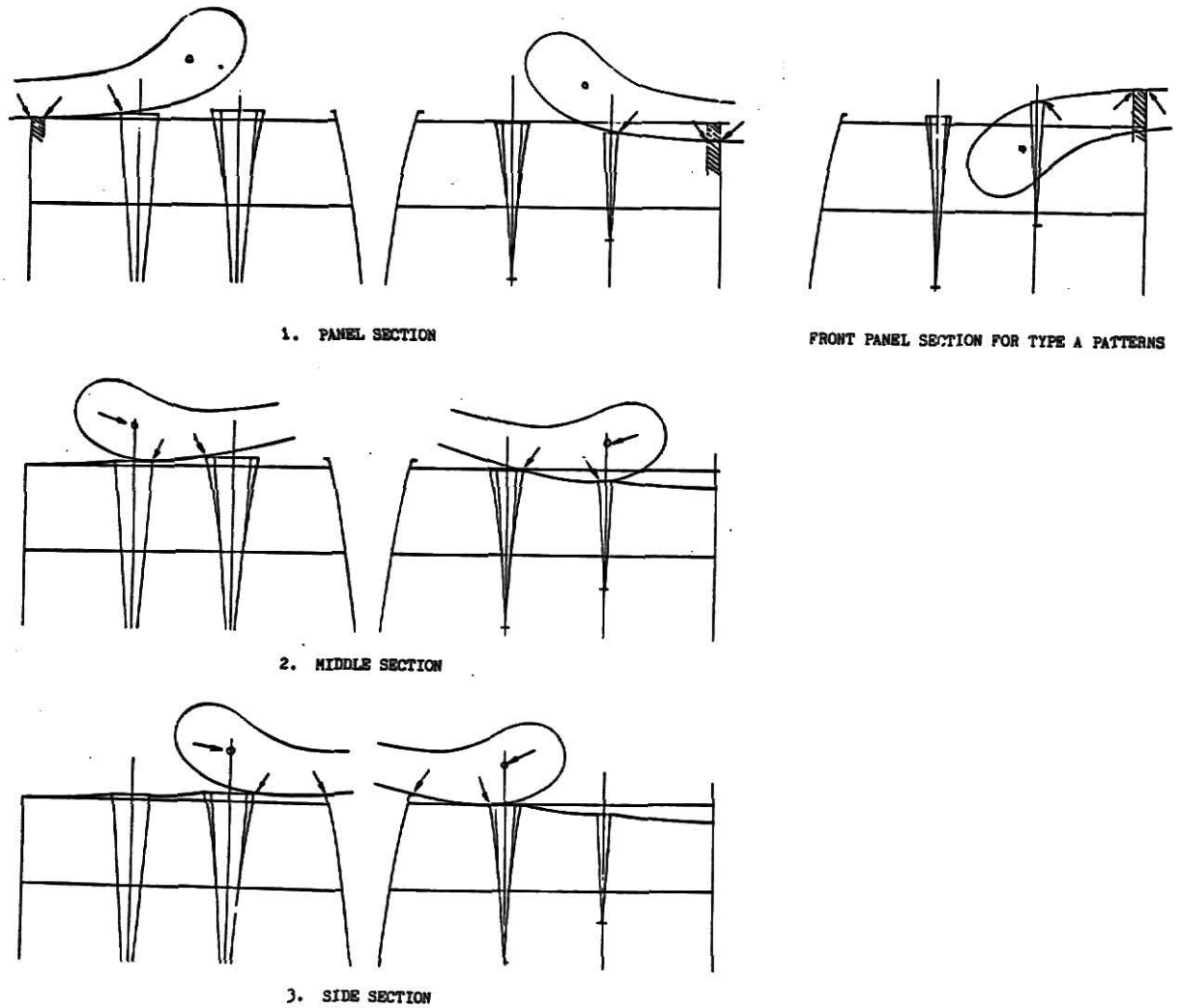


Figure 2-9. Waistline curve drafting steps

2.3.6 Waistline Curve

The #60 curve is used to draw the waistline curve from the waistline to the sideseam. The waistline curve is made up of three sections; the panel section, middle section, and side section. It is drawn one section at a time, so that a smooth curve results when the darts are closed. The following steps explain the procedure, which is illustrated in figure 2-9. The steps for each section are drafted on both the front and back pattern piece.

The first step in drafting the panel section is drawing a straight line on the waistline from the centerline 10 millimeters towards the panel dart. From that point a curve is drawn to the near side of the panel dart. For figure types "B" and "C" this line curves downward and for figure type "A" this curve, on the front pattern, bows upward.

The middle section lying in between the panel dart and the side dart is drawn next. In order to draft a smooth curve, the guide hole on the #60 curve must be directly over the panel dart centerline before the curve may be drawn from the edge of the panel dart to the edge of the side dart.

The side section lies between the side dart and the sideseam crossmark. The guidehole of the #60 curve must again be directly above the centerline of the panel dart before the curve is drawn. This step completes the waistline.

2.4 Conclusion

The patterns are now ready to be transferred onto muslin. The muslins are then sewn and used to check for proper pattern fit. Revisions are made to the front and back pattern for any necessary alterations. Once the pattern fits correctly a master copy is made on durable paper. This pattern may then be used as a "blueprint" for skirt styling or as a pattern for the sheath skirt. The skirt styles available through this pattern method are six-gore or a-line, eight-gore or basic flare, and pleated.

This chapter describes in detail that system and its usage as a hand methodology. Chapter 3 uses this work to create a computer system for the production of computerized patterns.

Chapter 3

Overview of program design

3.1 Introduction

As the name Mod-u-lar Pattern System implies, this methodology is modular in its approach. Just as a program is modular, each section of this drafting methodology is "reserved for on major function and tasks closely related to that function".[lee 85] As a result, the software structure of the implementation program strongly resembles the outline of the Mod-u-lar Pattern System.

3.2 System Description

The research project was implemented using Turbo Pascal and dBaseII. First, the information tables developed by Brockman were stored in dBaseII database files. The Turbo Pascal routines, written to automate the skirt drafting methodology, were able to access all necessary information from user input and the dBaseII files.

3.2.1 Why Turbo Pascal Was Used

Turbo Pascal was used to automate this basic skirt drafting methodology, because it is available for use on a personal computer, is capable of doing mathematical calculation of points, has graphics capabilities, and can access dBaseII text files. Since it was previously decided that the use of a personal computer would allow a financially feasible on-site system, a language that was

accessible on a personal computer was needed. A language that has graphics capabilities was needed, because of the graphic nature of the skirt pattern drafting method. Turbo Pascal also makes access of dBaseII text files possible. This allows some manner of accessing the tables of information developed for and used by the pattern drafting methodology.

3.2.2 Why dBaseII Was Used

dBaseII was used to enter and access the tables of information used throughout the Mod-u-lar Pattern System methodology. dBaseII is especially suited for table entry and database access. A database can be created for each table needed. Options such as audible feedback and preset field widths aid in error checking during data entry. Command files, shown in appendix C, were used to output the database contents, therefore allowing a visual double-check of the entries. dBaseII's capability to produce text files, allows data access by Turbo Pascal routines.

3.2.3 dBaseII File Access by Turbo Pascal Routines

The research project was implemented using Turbo Pascal and dBaseII. First, the information tables developed by Professor Brockman were stored in dBaseII database files. The information in the database was then copied into text files, shown in appendix B. Then the basic skirt drafting methodology was implemented using Turbo Pascal, see appendix F. The Turbo Pascal routines were able to access all necessary information from the dBaseII text files.

3.2.4 A Simple Example of dBaseII File Access by Turbo Pascal

A command file can be created to transform a dBaseII database file into a text file. The command "modify comm transform" creates a command file called transform. The contents of "transform" follow, with a brief explanation of each statement's purpose.

```
set default to b                  - set default drive
accept 'enter name of file to transform: ' to example
use &example                      - allow user to specify filename
set talk off                      - only want file contents
set alternate to example.txt      - results are placed in a text file
set alternate on
list off                          - eliminates record number
set alternate off
set talk on                       - reset dBaseII environment
```

The command "do transform" will cause the command file to execute.

In order that the Turbo Pascal source program may use the contents of the text file, the following statements are necessary. A declaration statement is necessary to declare variable names for the file and the data items to be read in from the file. Then a dBaseII text file name is "assigned" to that variable name. The dBaseII file name may be a string or a variable which contains a string value. After the file read pointer is "reset" at the beginning of the file, the data may be read from the file. Examples of the necessary statements follow, with a brief explanation of each statement's purpose.

```
program example(input,output);
var
  turbo_filename: text;           - declare file of type text
  data_item_name: integer;        - data item to be read from file
begin
  assign(turbo_filename,'dbase_filename.txt');
  reset(turbo_filename);          - read pointer at beginning of file
  read(turbo_filename,data_item_name); - read data item from file
end.
```

3.3 Hierarchical Diagram

Following is a hierarchical diagram, which shows the calling structure of the basic skirt drafting program. The boxes denote modules in the program. Boxes composed of dots denote subroutines also called by modules shown on the following page.

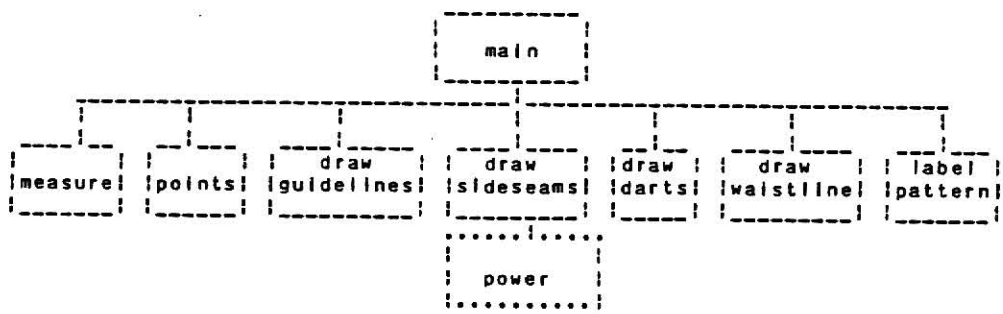


Figure 3-1. Hierarchy diagram for main program

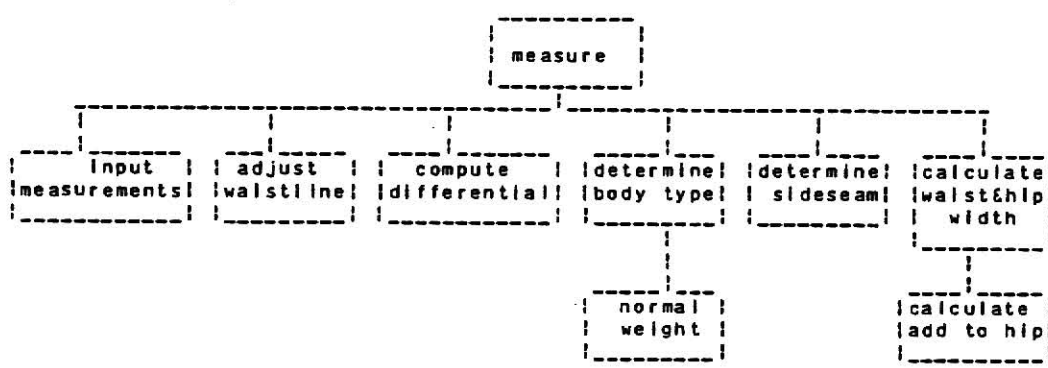


Figure 3-2. Hierarchy diagram for measure procedure

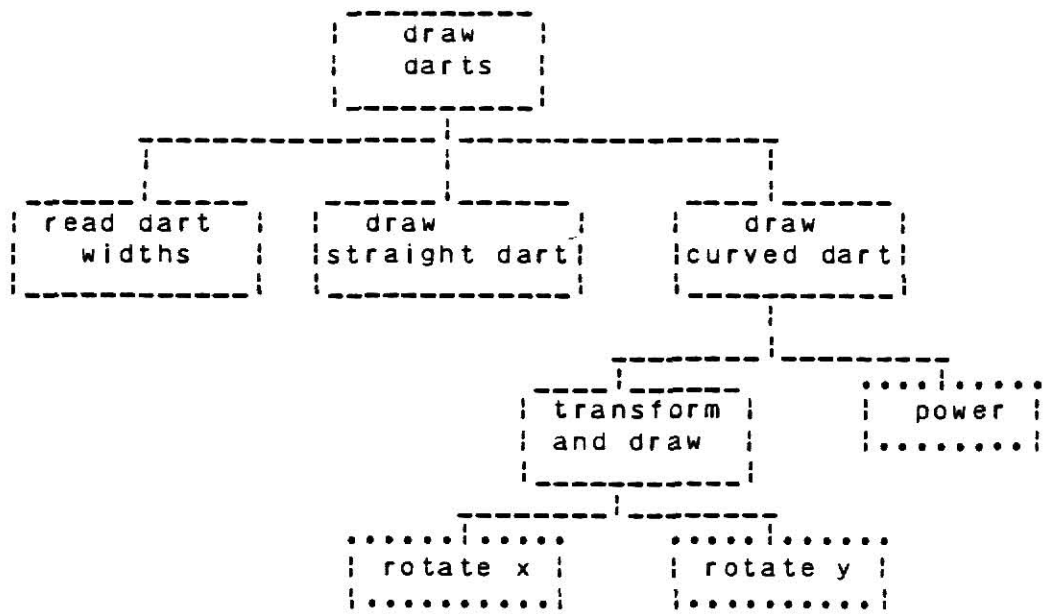


Figure 3-3. Hierarchy diagram for draw darts procedure

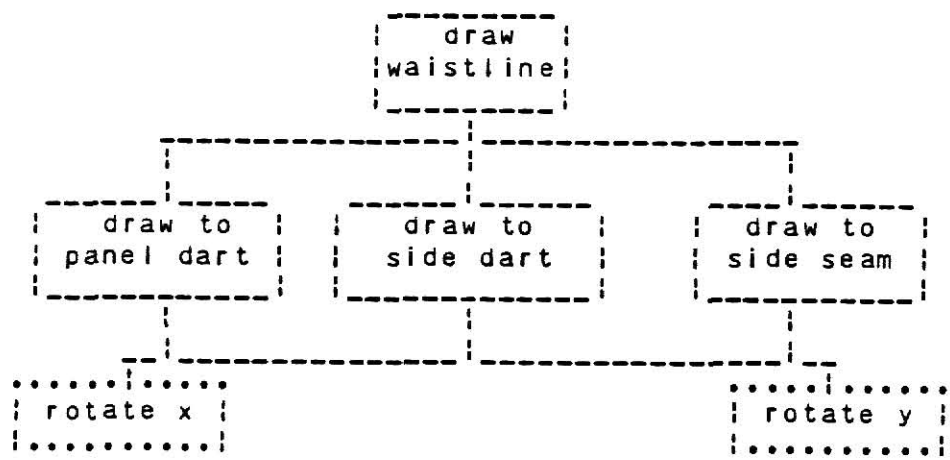


Figure 3-4. Hierarchy diagram for draw waistline procedure

3.4 Overview of the Program Design

The basic software design was modeled upon the same organization of tasks as the Mod-u-lar Pattern System methodology. As illustrated in the hierarchical diagram above the design is divided into seven major tasks; measure, points, draw guidelines, draw sideseams, draw waistline, and label pattern. This section will discuss the functions, design problems, and design decisions associated with each major task.

The first module to be executed is measure. Its functions include a user interface for the input of the five measurements and calculation of the remaining items of the skirt worksheet from those five inputs. The main problems addressed while designing this module were the user interface, detection of erroneous input, and the transformation of the input measurements.

The user interface consists of a series of prompts; one for each of the five input measurements. Each input measurement is "echoed" back to the user for visual verification. After all five measurements are input the user is queried as to whether he/she desires to correct any inputs or continue.

Error detection is possible by the user and/or the program. The user may detect errors through the visual feedback mentioned above. The program may detect errors through various methods of range-checking. Once an error is detected there must be a means to allow user to correct erroneous input. If the user detects an error during input, she/he is given the option to re-enter the input. If an

error is detected by the program, the user is first notified as to what type of error occurred. Then, the user is given the option to re-enter the measurements or quit.

The transformation of the input measurements was accomplished through the combined use of mathematical calculation and information table access. A problem encountered in this module was that the limitation of 32 fields for each dBaseII database. This prohibited entering the two pattern width tables, each of which required 64 fields. Therefore it was necessary to find an alternative to the pattern width at the waist and the pattern width at the hip table. Since part of the data in the tables was extrapolated by Professor Brockman, it was possible to find a pattern. This pattern was used to derive the pattern widths through a mathematical calculation.

Points is the second major module. It calculates coordinates and translations and determines scaling factors for tne graphic functions of the program. The problems encountered while designing this module were associated with the graphics mode and the scaling method. The scaling method is dependent on the graphics mode chosen as it determines the number of pixels. The graphics mode must be capable of graphical as well a textual representation. The scaling method must be easily modified to actual scale for plotting on paper in the future. The scale must also be able to represent the basic skirt in proportion on the screen.

The Turbo Pascal high resolution mode was chosen, because it is capable of simultaneous textual and graphical representation. The "Hires" mode the screen consists of 640 by 200 pixels, with the x-

coordinates ranging from 0 to 639 and the y-coordinates ranging from 0 to 199.

In order to represent the basic skirt in proportion on the screen, the number of pixels on the x and y axis and the screen dimensions must be used to determine the scale. For actual size representation of the skirt on plotter paper this is not necessary. The scaling method had to be easily adjustable for both representations. In order to allow this there are three separate interrelated scales. The first "scale" is used like a constant and is dependent on the method of representation; screen or plotter paper. The other two are an "xscale" and a "yscale" which are used for screen representation. The "xscale" and "yscale" are multiples of scale. If plotter representation is desired then these should be the scale multiplied by one. As of now, they are scaled for proportional screen representation of the basic skirt.

The third module, Draw Guidelines' function is to plot the pattern framework onto the screen. This module was rather straightforward and without problems. It was simply a matter of plotting the coordinates generated by the two previous modules.

Draw Sideseams is the fourth module. Its function is to draw the basic skirt's sideseams on the screen. The major problem associated with this module was generating the #60 curve and hipcurve necessary to plot the skirt sideseam.

The options were the use of a digitizer, Lagrange interpolation, or SAS subroutines order to determine the formula for a needed

curve. The digitizer available could only digitize a maximum area of 8 1/2 X 11 inches. This was too small for both the hipcurve and the #60 curve. The process of Lagrange interpolation proved too tedious, time consuming, and error prone. The alternative chosen was to use the prewritten SAS subroutines to generate the formula for each of the seven sideseam curves.

Draw Darts plots the four skirt darts on the screen. The problem associated with this module was the large number of dart templates available. As a result, there were too many different curves to allow a separate calculation for each. A general purpose module had to be developed. This module should have the ability to produce any of the dart templates by using information from a dart table. This table must be appropriate for the given bodytype and subtype. Through the combined use of this table and a series of curve formulation, rotation, scaling, and translation, any of the dart templates can be reproduced.

Draw Waistline draws a smooth waistline curve between all the darts from the sideseam to the centerline. The problem encountered with this module is in achieving a smooth waistline curve once the darts are joined. This is achieved by positioning the #60 curve exactly as specified by Professor Brockman's methodology. The problem results when attempting to generalize the module for the variety of curves which must be generated. The curves may vary in length, so that the endpoint of the curve usually does not lay on the waistline.

The solution to the problem is to calculate the endpoint of the

curve for the needed curve length. Then, the curve may be rotated, so that it can intersect with the end of the dart or sideseam which is its destination.

Label Pattern identifies the various parts of the basic skirt. The simultaneous representation of graphical and textual output to the screeen is possible through the use of high resolution mode in Turbo pascal graphics. This is a straightforward module with no design problems.

3.5 Data Flow Descriptions

An analysis of the data flow was done for each module. A list of all the input and output variables for each module is described in table form in appendix E. This analysis aided in the removal of global access of variables by the modules, in ensuring that all necessary inputs and outputs existed, and in organizing the large number of variables present due to all the input needed.

Chapter 4

Scope of the Research Project

The scope of this research project is automation of the basic skirt drafting method using Turbo Pascal. The project is limited to simulating the front and back of the basic skirt pattern on the screen, because of time constraints. Since Brockman continues to modify and improve her pattern drafting methodology; this project is also limited to the implementation of the method as it existed in January of 1985, when we commenced this research.

4.1 Current Capabilities

4.1.1 User Interface

Five body measurements used in concert with the lookup tables and skirt templates are required to produce the basic skirt pattern. This system queries the user for the necessary body measurements. The measurements are validated after they are read in. If an error is detected, the user is given the option either to continue by re-entering the measurements, or to exit the system. If there are no detectable errors, the user is given the option to correct any errors known to her/him. Once the skirt pattern is simulated on the screen, the user may enter new measurements to produce another basic skirt pattern.

4.1.2 User Interface for Table Entry

There is another level of users that will be accessing this system. Since this is an evolving methodology, there is a need to modify the databases which contain the information tables. There are currently two categories of command files, listed in appendix , which allow table entry to be verified and accessed by the Turbo Pascal program. The first type of command file prints out a formatted listing with appropriate headings of the requested database file. This listing allows the user to have a hardcopy for visual verification of the table entries. The second type of command file produces a dBaseII text file for the requested dBaseII database. This format permits access of the information by the Turbo Pascal routines.

4.2 Future Extensions

Possibilities for future extensions may include the following:

- add the most current modifications of the method to the pattern drafting program.
- implement alterations to basic patterns.
- allow saving of skirt pattern once alterations are done.
- draw the pattern on paper using a plotter.
- automate the basic bodice pattern.
- automate design variations of the basic skirt and bodice. - ie., gored,pleated,etc skirt and bodice.
- allow user to design interactively.
- add some statistics gathering capabilities to the program to add to the empirical data already compiled.

References

[bor 85] Borland, Frank, Turbo Pascal Language Manual, Borland International, 1985

[bro 85] Brockman, Professor Emeritus Helen, The Mod-u-lar Pattern System: The Skirt Book, Kansas State Research Foundation, 1985

[bro 86] Brockman, Professor Emeritus Helen, personal communication, 1986

[han 74] Hanford, Jack, Professional Patternmaking for Designers of Women's Wear, Hanford Enterprises, Inc., 1974

[jaf 73] Jaffe, Hilde, Relis, Nurie, Draping for Fashion Design, Reston Publishing Company, 1973

[lee 85] Leeson, Marjorie M., Computer Information: A Modular System, Science Research Associates, 1985

[pri 78] Price, Jeanne, Zamkoff, Bernard, Grading Techniques for Modern Design, Fairchild Publications, 1978

[sch 84] Schweers, Cecelia, A Comparison of Garment Fit Resulting from Skirt Patterns Generated by the Brockman Pattern System and Traditional Unit Methods of Alternating a Commercial Pattern, unpublished master's thesis, 1984

[var 86] Varney, Laura Ph.D., personal communication, 1986

Appendix A

User's Manual

User's Manual

This user's manual assumes little prior knowledge regarding the use of personal computers. The ability of the user to insert a disk, and distinguish between the various disk drives is assumed. It is also assumed that trained personnel will take the five body measurements which are to be typed in by the user.

Equipment Needed

The Mod-u-lar Pattern System skirt program can be used on any IBM or IBM compatible personal computer. A Mod-u-lar Pattern System Skirt disk is required in addition to a personal computer.

How to Start the System

Press the monitor and unit switches to the "on" position. Insert the Mod-u-lar Pattern System Skirt diskette. When prompted for the date and time, press the <return> key. Once the system is started, as many skirt patterns may be drafted on the screen as desired.

Drafting a Basic Skirt Pattern on the Screen

The user must enter five body measurements, in order to enable the basic skirt to be drawn onto the screen. These measurements are the height, weight, waistline girth, torsoline girth, and center back length. The unit of measurement is centimeters, with the exception of the weight, which is measured in pounds. The measurements must be taken by trained personnel in order to guaran-

tee a correct fit.

The system will prompt the user for each of the five measurements. skirt pattern. Each measurement is "echoed" back to the screen to allow the user a visual double-check of the entered measurements. The measurements are validated after they are read in. If an error is detected, the user is given the option either to continue by re-entering the measurements, or to exit the system. If there are no detectable errors, the user is given the option to correct any errors known to her/him. Once the skirt pattern is simulated on the screen, the user is given the option either to enter new measurements to produce another basic skirt pattern or to quit.

Appendix B

Creating a dBaseII Text File from a dBaseII Database

In order to enable a Turbo Pascal routine to access the information in the database file, the information must be copied into a dBaseII text file. The following command file executes the necessary series of dBaseII statements to achieve this.

```
*** Creates a text file for requested database file ***

set default to b
accept 'enter file to transform to textfile ' to example
use &example

set talk off
set alternate to &example
set alternate  on

list off

set alternate off
set talk on
```

Appendix C

Printing the Contents of a dBaseII Database

```
*** Prints all information in the height/weight database ***

set default to a
use ht_wt
set talk off
set print on
?
? '                                  height / weight table '
?
? '                                   differential sizes'
? '.-------------------------------------------------------------------------',;
  '----------------.'
? '! df | df | df | df | df | df | df | df | df | df | df | df | df | df | ',;
  'df | df | skirt!'
? '! ht | 12 | 14 | 16 | 18 | 20 | 22 | 24 | 26 | 28 | 30 | 32 | 34 | 36 | ',;
  '38 | 40 !length!'
?
do while .not. eof

   ? height,d12,d14,d16,d18,d20,d22,d24,d26,d28,d30,d32,d34,d36,d38,d40,;
     '  ',skirt:len
   skip

enddo
set print off
set talk on
return
```

```
*** Prints all the information for the requested dart database ***

set default to b
accept 'enter name of table to be printed: ' to darts

use &darts
set talk off
set print on
?
?
? 'dart file name: ',darts
?
? '.-------------------------------------------------------------------------------.'
? '|   |          dart width at w1             |      dart width at c1         |'
? '|dif|   back    |      front     |   total   |  back | front |    total      |'
? '|   | p | s | s | s | s | p | p | b | f |b+f| p | s | s | p | b | f |b+f|'

?
do while .not. eof

   ? diff,w1bpanel,w1bside1,w1bside2,w1fside1,w1fside2,w1fpan11,w1fpan12,;
     w1totalb,w1totalf,w1totbf,c1bpanel,c1bside,c1fside,c1fpanel,c1totalb,;
     c1totalf,c1totbf
   skip

enddo

go top
?
?
? '.-------------------------------------------------------------------------------.'
? '|     dart degrees      |       waistline level      |  dart length |guide|'
? '|    back   |   front   |     back     |     front     |back|  front  |hole |'
? '|  p  |  s  |  s  |  p  | cb | p | s | s | p |  cf |both| s |  p  |level|'

?

do while .not. eof

   ? bpdegree,bsdegree,fsdegree,fpdegree,cbwlevel,bpwlevel,;
     bswlevel,fswlevel,fpwlevel,cfwlevel,blength,fslength,fplength,guidelev
   skip

enddo
set print off
set talk on
return
```

Appendix D

Information Tables

Database structure for all Dart Width Measurement tables

STRUCTURE FOR FILE: A:BEQUAL .DBF
NUMBER OF RECORDS: 00007
DATE OF LAST UPDATE: 01/01/80
PRIMARY USE DATABASE

FLD	NAME	TYPE	WIDTH	DEC
001	DIFF	N	002	
002	WLBPANEL	N	002	
003	WLBSIDE1	N	002	
004	WLBSIDE2	N	002	
005	WLFSIDE1	N	002	
006	WLFSIDE2	N	002	
007	WLFPANL1	N	002	
008	WLFPANL2	N	002	
009	WLTOTALB	N	002	
010	WLTOTALF	N	002	
011	WLTOTBF	N	003	
012	CLBPANEL	N	002	
013	CLBSIDE	N	002	
014	CLFSIDE	N	002	
015	CLFPANEL	N	002	
016	CLTOTALB	N	002	
017	CLTOTALF	N	002	
018	CLTOTBF	N	002	
019	BPDEGREE	N	004	001
020	BSDEGREE	N	004	001
021	FSDEGREE	N	004	001
022	FPDEGREE	N	003	001
023	CBWLEVEL	N	003	
024	BPWLEVEL	N	002	
025	BSWLEVEL	N	002	
026	FSWLEVEL	N	002	
027	FPWLEVEL	N	003	
028	CFWLEVEL	N	003	
029	BLENGTH	N	003	
030	FSLENGTH	N	003	
031	FPLENGTH	N	003	
032	GUIDELEV	N	003	
** TOTAL **			00080	

Database structure for height / weight table

```
STRUCTURE FOR FILE:  A:HT_WT    .DBF
NUMBER OF RECORDS:   00017
DATE OF LAST UPDATE: 01/01/80
PRIMARY USE DATABASE
FLD        NAME        TYPE  WIDTH    DEC
001       HEIGHT         N     003
002       D12            N     003
003       D14            N     003
004       D16            N     003
005       D18            N     003
006       D20            N     003
007       D22            N     003
008       D24            N     003
009       D26            N     003
010       D28            N     003
011       D30            N     003
012       D32            N     003
013       D34            N     003
014       D36            N     003
015       D38            N     003
016       D40            N     003
017       SKIRT:LEN      N     002
** TOTAL **                    00051
```

dart file name: aequal

	dart width at w1									dart width at c1					
dif	back			front				total		back		front		total	
	p	s	s	s	s	p	p	b	f	b+f	p	s	s	p	b	f	b+f
16	18	21	28	13	16	7	7	46	23	69	9	10	5	0	19	5	24
20	21	24	33	15	18	7	9	54	27	81	12	12	6	1	24	7	31
24	24	27	38	16	20	8	11	62	31	93	15	15	7	2	30	9	38
28	27	29	43	17	22	10	13	70	35	105	17	17	8	3	34	11	45
32	29	32	47	18	25	11	15	76	40	116	18	19	10	4	37	14	51
36	31	35	51	20	28	12	17	82	45	127	20	21	11	5	41	16	57
40	34	40	54	22	31	13	19	88	50	138	23	23	12	6	46	18	64

dart degrees					waistline level						dart length			guide
back		front			back		front				back	front		hole
p	s	s	p	cb	p	s	s	p	cf		both	s	p	level
8.0	9.0	7.0	5.5	-10	-5	4	4	6	10		130	100	60	115
9.0	9.0	7.5	5.5	-10	-5	5	4	6	10		140	110	70	125
9.0	9.0	7.5	5.5	-10	-5	6	5	7	10		150	120	80	135
10.0	10.0	7.5	5.5	-10	-4	7	6	7	10		160	130	90	145
10.0	10.5	7.5	5.5	-10	-4	8	7	8	10		170	140	100	155
10.0	10.5	7.5	5.5	-10	-3	9	8	8	10		180	150	110	165
10.5	10.5	7.5	5.5	-10	-3	10	9	9	10		190	160	120	175

dart file name: apluss

	dart width at w1									dart width at c1					
dif	back			front				total		back		front		total	
	p	s	s	s	s	p	p	b	f	b+f	p	s	s	p	b	f	b+f
16	19	22	30	10	14	6	6	49	20	69	9	10	3	0	19	3	22
20	22	25	35	12	16	8	8	57	24	81	12	13	4	0	25	4	29
24	26	28	39	13	18	8	10	65	28	93	15	15	5	1	30	6	36
28	30	32	43	14	20	9	12	73	32	105	17	18	6	2	35	8	43
32	33	34	47	15	22	10	14	80	36	116	20	20	7	3	40	10	50
36	35	37	52	16	24	11	16	87	40	127	22	23	8	4	45	12	57
40	38	40	56	17	26	12	18	94	44	138	25	25	9	5	50	14	64

dart degrees					waistline level						dart length			guide
back		front			back		front				back	front		hole
p	s	s	p	cb	p	s	s	p	cf		both	s	p	level
8.5	9.0	6.0	5.5	-5	0	6	6	18	25		130	90	50	110
9.0	9.5	6.0	5.5	-5	0	7	7	18	25		140	100	60	120
10.0	10.0	6.0	5.5	-5	0	8	8	18	25		150	110	70	130
10.5	11.0	6.0	5.5	-5	1	9	9	18	25		160	120	80	140
11.0	11.0	6.0	5.5	-5	1	10	10	18	25		170	130	90	150
11.0	11.5	6.0	5.5	-5	2	11	11	18	25		180	140	100	160
11.5	11.5	6.0	5.5	-5	3	12	12	18	25		190	150	110	170

dart file name: adelta

dif	dart width at wl: back p	back s	front s	front s	total p	total b	total f	total b+f	dart width at cl: back p	back s	front s	front p	total b	total f	total b+f
16	21	22	30	9	14	4	4	51	18	69	9	9	2	0	18	2	20
20	25	26	35	10	16	5	5	60	21	81	12	12	3	0	24	3	27
24	28	29	40	12	19	6	6	68	25	93	15	15	4	0	30	4	34
28	31	33	44	14	23	7	7	75	30	105	17	17	6	0	34	6	41
32	34	36	47	16	26	9	9	81	35	116	20	20	7	1	40	8	48
36	37	39	51	18	29	10	10	88	39	127	22	23	8	2	45	10	55

dart degrees: back p	back s	front s	front p	waistline level: back cb	back p	back s	front s	front p	cf	dart length: back both	front s	front p	guide hole level
9.5	9.5	5.0	4.0	0	5	10	12	25	35	120	90	30	105
10.5	10.5	5.0	4.0	0	5	10	12	25	35	130	100	40	115
11.0	11.0	5.0	4.0	0	5	10	13	25	35	140	110	50	125
11.5	11.5	5.0	4.0	0	5	10	13	25	35	150	120	60	135
12.0	12.0	6.0	5.0	0	6	11	14	26	35	160	130	70	145
12.0	12.5	6.5	5.5	0	7	12	14	27	35	170	140	80	155

dart file name: bequal

dif	dart width at wl: back p	back s	front s	front s	total p	total b	total f	total b+f	dart width at cl: back p	back s	front s	front p	total b	total f	total b+f
16	20	21	25	16	19	7	7	45	26	71	9	9	6	1	18	7	25
20	22	23	29	19	22	8	9	51	31	82	11	11	8	2	22	10	32
24	24	25	33	22	25	9	11	57	36	93	13	13	10	3	26	13	39
28	26	27	37	25	29	10	13	63	42	105	15	15	12	4	30	16	46
32	28	29	41	27	32	11	15	69	47	116	17	17	14	5	34	19	53
36	31	32	44	29	35	12	17	75	52	127	19	19	16	6	38	22	60
40	33	34	48	31	38	13	19	81	57	138	21	21	18	7	42	25	67

dart degrees: back p	back s	front s	front p	waistline level: back cb	back p	back s	front s	front p	cf	dart length: back both	front s	front p	guide hole level
10.0	10.0	8.0	5.0	-10	-4	0	4	3	0	120	110	80	115
10.0	10.0	8.5	5.0	-10	-4	1	5	3	0	130	120	90	125
10.0	10.0	9.0	5.0	-10	-3	2	6	3	0	140	130	100	135
10.0	10.0	10.0	5.0	-10	-3	3	7	3	0	150	140	110	145
10.0	10.0	10.0	5.0	-10	-2	4	7	3	0	160	150	120	155
10.5	10.5	10.0	5.0	-10	-2	5	8	3	0	170	160	130	165
10.5	10.5	10.0	5.0	-10	-2	6	8	3	0	180	170	140	175

dart file name: bpluss

dif	dart width at w1 back p	back s	front s	front p	total b	total f	total b+f	dart width at c1 back p	back s	front s	front p	total b	total f	total b+f
16	19	20	26	17	20	6	6	45	26	71	8	9	6	1	17	7	24
20	22	23	30	19	22	7	8	52	30	82	10	11	8	2	21	10	31
24	24	26	34	21	25	8	10	58	35	93	12	13	10	3	25	13	38
28	26	28	39	24	28	10	12	65	40	105	14	15	12	4	29	16	45
32	28	30	43	26	31	11	14	71	45	116	16	17	14	5	33	19	52
36	31	33	48	28	33	12	15	79	48	127	18	19	16	6	37	22	50
40	33	35	53	30	36	13	16	86	52	138	20	21	18	7	41	25	66

dart degrees back p	back s	front s	front p	waistline level back cb	back p	back s	front s	front p	front cf	dart length back both	front s	front p	guide hole level
9.0	9.5	9.0	4.0	-5	1	6	6	6	5	115	105	75	110
9.5	10.0	9.0	5.0	-4	2	7	6	6	5	125	115	85	120
10.0	10.5	9.5	5.0	-4	3	8	6	6	5	135	125	95	130
10.0	10.5	9.5	5.0	-3	4	8	7	6	5	145	135	105	140
10.0	10.5	10.0	5.0	-3	5	8	7	7	5	155	145	115	150
10.5	11.0	10.0	5.5	-3	5	9	8	7	5	165	155	125	160
10.5	11.0	10.0	5.5	-3	5	10	8	7	5	175	165	135	170

dart file name: bdelta

dif	dart width at w1 back p	back s	front s	front p	total b	total f	total b+f	dart width at c1 back p	back s	front s	front p	total b	total f	total b+f
16	19	20	26	13	20	5	5	45	25	70	10	10	6	0	20	6	26
20	22	23	29	15	24	6	6	51	30	81	12	12	8	1	24	9	33
24	24	25	32	18	28	6	8	56	36	92	14	14	10	2	28	12	40
28	26	28	36	21	32	7	10	62	42	104	16	16	12	3	32	15	47

dart degrees back p	back s	front s	front p	waistline level back cb	back p	back s	front s	front p	front cf	dart length back both	front s	front p	guide hole level
9.0	9.0	6.0	4.0	-15	-9	0	2	-2	-5	130	120	70	125
9.5	9.5	6.5	4.0	-15	-8	1	2	-2	-5	140	130	80	135
9.5	9.5	7.0	4.0	-15	-8	2	2	-2	-5	150	140	90	145
9.5	9.5	8.0	4.0	-15	-7	3	2	-2	-5	160	150	100	155

dart file name: cequal

dif	dart width at w1: back p	back s	front s	front s	total b	total f	total b+f	dart width at c1: back p	back s	front s	front p	total b	total f	total b+f
16	21	20	25	13	17	6	6	46	22	68	11	10	4	0	21	4	25
20	24	23	29	15	19	8	8	54	26	80	13	12	5	2	25	7	32
24	27	27	34	16	21	8	10	61	31	92	16	15	7	2	31	9	40
28	30	31	39	18	23	9	12	69	35	104	18	18	9	3	36	12	48
32	33	34	43	19	25	10	14	76	39	115	21	20	10	4	41	14	55
36	36	37	47	20	27	11	16	83	43	126	23	23	11	5	46	16	62
40	39	39	51	21	29	12	18	90	47	137	26	25	12	6	51	18	69

dart degrees: back p	back s	front s	front p	waistline level: back cb	back p	back s	front s	front p	front cf	dart length: back both	front s	front p	guide hole level
9.0	8.5	8.0	6.0	0	3	4	2	1	-4	135	95	65	115
9.5	9.0	8.0	6.0	0	3	5	2	0	-5	145	105	75	125
10.0	9.5	8.0	6.0	0	3	6	2	-1	-6	155	115	85	135
10.5	10.5	8.0	6.0	0	3	7	2	-2	-7	165	125	95	145
11.0	10.5	8.0	6.0	0	3	8	2	-3	-8	175	135	105	155
11.0	11.0	8.0	6.0	0	4	9	2	-4	-9	185	145	115	165
11.5	11.0	8.0	6.0	0	4	10	2	-5	-10	195	155	125	175

dart file name: cpluss

dif	dart width at w1: back p	back s	front s	front s	total b	total f	total b+f	dart width at c1: back p	back s	front s	front p	total b	total f	total b+f
16	22	21	26	13	16	5	5	48	21	69	11	10	4	0	21	4	25
20	25	24	30	14	20	6	6	55	26	81	14	13	5	0	27	5	32
24	27	27	35	15	23	6	8	62	31	93	16	15	6	2	31	8	39
28	30	30	40	16	25	7	10	70	35	105	18	18	7	3	36	10	46
32	33	33	44	17	28	8	11	77	39	116	21	20	9	3	41	12	53
36	36	36	48	19	30	9	13	84	43	127	23	23	10	4	46	14	60
40	39	39	52	20	32	10	15	71	47	138	26	25	11	5	51	16	67

dart degrees: back p	back s	front s	front p	waistline level: back cb	back p	back s	front s	front p	front cf	dart length: back both	front s	front p	guide hole level
9.0	8.0	7.5	5.0	0	3	5	-1	-4	-8	140	90	60	115
9.5	9.0	7.5	5.0	0	3	6	-1	-5	-10	150	100	70	125
9.5	9.0	7.5	5.0	0	4	7	-1	-6	-12	160	110	80	135
10.0	10.0	7.5	5.0	0	4	8	-1	-7	-14	170	120	90	145
10.5	10.0	7.5	5.0	0	4	9	-1	-8	-16	180	130	100	155
10.5	10.5	7.5	5.0	0	5	10	-1	-9	-18	190	140	110	165
11.0	10.5	7.5	5.0	0	6	11	-1	-10	-20	200	150	120	175

dart file name: cdelta

	dart width at w1										dart width at c1					
dif	back			front			total			back	front		total			
	p	s	s	s	s	p	p	b	f	b+f	p	s	s	p	b	f	b+f
24	29	27	35	12	23	6	6	64	29	93	17	16	5	1	33	6	39
28	32	30	40	14	25	6	8	72	33	105	19	18	6	2	37	8	45
32	34	34	44	15	28	7	10	78	38	116	21	21	7	2	42	9	51
36	36	37	49	17	30	8	12	85	42	127	24	23	8	3	47	11	58
40	39	40	53	18	32	9	14	92	46	138	26	26	10	4	52	14	66

	dart degrees					waistline level						dart length			guide
	back		front			back			front			back	front		hole
p	s		s	p	cb	p	s		s	p	cf	both	s	p	level
9.0	9.0		7.0	5.0	0	5	8		-1	-11	-17	165	105	75	135
10.0	9.5		7.0	5.0	0	5	9		-1	-12	-19	175	115	85	145
10.0	10.0		7.0	5.0	0	6	10		-1	-13	-21	185	125	95	155
10.5	10.0		7.0	5.0	0	6	11		-1	-14	-23	195	135	105	165
10.5	10.5		7.0	5.0	0	7	12		-1	-15	-25	205	145	115	175

height / weight table

differential sizes

df ht	df 12	df 14	df 16	df 18	df 20	df 22	df 24	df 26	df 28	df 30	df 32	df 34	df 36	df 38	df 40	skirt length
143	91	92	93	94	95	96	97	98	99	101	103	0	0	0	0	55
145	94	95	96	97	98	99	100	101	103	105	107	0	0	0	0	57
148	97	98	99	100	101	102	103	104	106	108	110	112	0	0	0	58
150	100	101	102	103	104	105	106	107	109	111	113	115	0	0	0	59
153	103	104	105	106	107	108	109	111	113	115	117	119	121	0	0	61
155	106	107	108	109	110	111	112	114	116	118	120	122	124	0	0	62
158	109	110	111	112	113	114	115	117	119	121	123	125	127	130	0	63
160	112	113	114	115	116	117	118	120	122	124	126	128	130	133	0	64
163	115	116	117	118	119	120	121	123	125	127	129	131	133	136	140	65
165	118	119	120	121	122	123	124	126	128	130	132	134	136	140	144	66
168	0	122	123	124	125	126	127	129	131	133	135	137	140	144	148	67
170	0	126	127	128	129	130	131	133	135	137	139	141	144	148	152	68
173	0	0	130	131	132	133	134	136	138	140	142	145	148	152	156	69
175	0	0	133	134	135	136	137	139	141	143	145	148	152	156	160	70
178	0	0	0	137	138	139	140	142	144	146	149	152	156	160	164	71
180	0	0	0	141	142	143	144	146	148	150	153	156	160	164	168	72
183	0	0	0	0	146	147	148	150	152	154	157	160	164	168	172	73

PATTERN WIDTH - BY DIFFERENTIAL AND WAISTLINE GIRTH MEASUREMENT

SS	DF	AT THE WAISTLINE																			
		60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
1	14	191	194	196	199	201	204	206	209	211	214	216	219	221	224	226	229	231	234	236	239
	16	194	197	199	202	204	207	209	212	214	217	219	222	224	227	229	232	234	237	239	242
2	18	197	200	202	205	207	210	212	215	217	220	222	225	227	230	232	235	237	240	242	245
	20	200	203	205	208	210	213	215	218	220	223	225	228	230	233	235	238	240	243	245	248
3	22	203	206	208	211	213	216	219	221	223	226	228	231	233	236	238	241	243	246	248	251
	24	206	209	211	214	216	219	221	224	226	229	231	234	236	239	241	244	246	249	251	254
4	26	209	212	214	217	219	222	224	227	229	232	234	237	239	242	244	247	249	252	254	257
	28	212	215	217	220	222	225	227	230	232	235	237	240	242	245	247	250	252	255	257	260
5	30	215	218	220	223	225	228	230	233	235	238	240	243	245	248	250	253	255	258	260	263
	32	218	221	223	226	228	231	233	236	238	241	243	246	248	251	253	256	258	261	263	266
6	34	221	224	226	229	231	234	236	239	241	244	246	249	251	254	256	259	261	264	266	269
	36	224	227	229	232	234	237	239	242	244	247	249	252	254	257	259	262	264	267	269	272
7	38	227	230	232	235	237	240	242	245	247	250	252	255	257	260	262	265	267	270	272	275
	40	230	233	235	238	240	243	245	248	250	253	255	258	260	263	265	268	270	273	275	278

SS	DF	AT THE WAISTLINE																			
		80	82	84	86	88	90	92	94	96	98	100	102	104	106	108	110	112	114	116	118
1	14	241	246	251	256	261	266	271	276	281	286	291	296	301	306	311	316	321	326	331	336
	16	244	249	254	259	264	269	274	279	284	289	294	299	304	309	314	319	324	329	334	339
2	18	247	252	257	262	267	272	277	282	287	292	297	302	307	312	317	322	327	332	337	342
	20	250	255	260	265	270	275	280	285	290	295	300	305	310	315	320	325	330	335	340	345
3	22	253	258	263	268	273	278	283	288	293	298	303	308	313	318	323	328	333	338	343	348
	24	256	261	266	271	276	281	286	291	296	301	306	311	316	321	326	331	336	341	346	351
4	26	259	264	269	274	279	284	289	294	299	304	309	314	319	324	329	334	339	344	349	354
	28	262	267	272	277	282	287	292	297	302	307	312	317	322	327	332	337	342	347	352	357
5	30	265	270	275	280	285	290	295	300	305	310	315	320	325	330	335	340	345	350	355	360
	32	268	273	278	283	288	293	298	303	308	313	318	323	328	333	338	343	348	353	358	363
6	34	271	276	281	286	291	296	301	306	311	316	321	326	331	336	341	346	351	356	361	366
	36	274	279	284	289	294	299	304	309	314	319	324	329	334	339	344	349	354	359	364	369
7	38	277	282	287	292	297	302	307	312	317	322	327	332	337	342	347	352	357	362	367	372
	40	280	285	290	295	300	305	310	315	320	325	330	335	340	345	350	355	360	365	370	375

SS	DF	AT THE HIPLINE																			
		80	82	84	86	88	90	92	94	96	98	100	102	104	106	108	110	112	114	116	118
1	14	251	256	261	266	271	276	281	286	291	296	301	306	311	316	321	326	331	336	341	346
	16	254	259	264	269	274	279	284	289	294	299	304	309	314	319	324	329	334	339	344	349
2	18	260	265	270	275	280	285	290	295	300	305	310	315	320	325	330	335	340	345	350	355
	20	263	268	273	278	283	288	293	298	303	308	313	318	323	328	333	338	343	348	353	358
3	22	269	274	279	284	289	294	299	304	309	314	319	324	329	334	339	344	349	354	359	364
	24	272	277	282	287	292	297	302	307	312	317	322	327	332	337	342	347	352	357	362	367
4	26	278	283	288	293	298	303	308	313	318	323	328	333	338	343	348	353	358	363	368	373
	28	281	286	291	296	301	306	311	316	321	326	331	336	341	346	351	356	361	366	371	376
5	30	287	292	297	302	307	312	317	322	327	332	337	342	347	352	357	362	367	372	377	382
	32	290	295	300	305	310	315	320	325	330	335	340	345	350	355	360	365	370	375	380	385
6	34	296	301	306	311	316	321	326	331	336	341	346	351	356	361	366	371	376	381	386	391
	36	299	304	309	314	319	324	329	334	339	344	349	354	359	364	369	374	379	384	389	394
7	38	305	310	315	320	325	330	335	340	345	350	355	360	365	370	375	380	385	390	395	400
	40	308	313	318	323	328	333	338	343	348	353	358	363	368	373	378	383	388	393	398	403

SS	DF	AT THE HIPLINE																			
		60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
1	14	201	204	206	209	211	214	216	219	221	224	226	229	231	234	236	239	241	244	246	249
	16	204	207	209	212	214	217	219	222	224	227	229	232	234	237	239	242	244	247	249	252
2	18	210	213	215	218	220	223	225	228	230	233	235	238	240	243	245	248	250	253	255	258
	20	213	216	218	221	223	226	228	231	233	236	238	241	243	246	248	251	253	256	258	261
3	22	219	222	224	227	229	232	234	237	239	242	244	247	249	252	254	257	259	262	264	267
	24	222	225	227	230	232	235	237	240	242	245	247	250	252	255	257	260	262	265	267	270
4	26	228	231	233	236	238	241	243	246	248	251	253	256	258	261	263	266	268	271	273	276
	28	231	234	236	239	241	244	246	249	251	254	256	259	261	264	266	269	271	274	276	279
5	30	237	240	242	245	247	250	252	255	257	260	262	265	267	270	272	275	277	280	282	285
	32	240	243	245	248	250	253	255	258	260	263	265	268	270	273	275	278	280	283	285	288
6	34	246	249	251	254	256	259	261	264	266	269	271	274	276	279	281	284	286	289	291	294
	36	249	252	254	257	259	262	264	267	269	272	274	277	279	282	284	287	289	291	294	297
7	38	255	258	260	263	265	268	270	273	275	278	280	283	285	288	290	293	295	298	300	303
	40	258	261	263	266	268	271	273	276	278	281	283	286	288	291	293	296	298	301	303	306

Appendix E

Data Flow Description Tables

Procedure name	Input variables	Output variables
MAIN PROGRAM	quit (measure) enter (screen) error (measure)	
1. MEASURE		error
1a. INPUT_MEASUREMENTS	height (screen) weight (screen) waistline_girth (screen) torsoline_girth (screen) back (screen) error (parameter) quit (screen)	height weight waistline_girth torsoline_girth back correct quit
1b. ADJUST_WAISTLINE	torsoline_girth (param) waistline_girth (param)	ADJUST_WAISTLINE
1c. COMPUTE_DIFFERENTIAL	torsoline_girth (param) waistline_girth (param)	COMPUTE_DIFFERENTIAL
1d. BODY_TYPE	weight (param) back (param) differential (param)	BODY_TYPE
1d1. NORMAL_WEIGHT	ht (param) wt (param) diff (param) height (ht_weight table) norm_wt (ht_weight table) skirt_length(ht_weight table)	norm_wt skirt_length
1f. DETERMINE_SIDESEAM	diff (param)	DETERMINE_SIDESEAM
1g. CALC_WAISTHIP_WIDTHS	waistline_girth (param) differential (param) calc_add_to_hip (funct)	waist_pattern_width hip_pattern_width
1g1. CALC_ADD_TO_HIP	differential (param)	CALC_ADD_TO_HIP

Procedure name	Input variables	Output variables
2. POINTS	skirt_length (param) waist_pattern_width	f_sstranslation b_sstranslation ytranslation waistlevel controllevel hiplevel hemlevel cb cf xscale yscale
3. DRAW_GUIDELINES	f_sstranslation (param) b_sstranslation (param) waistlevel (param) hemlevel (param) cb (param) cf (param)	Front and back guidelines to the screen
4. DRAW_SIDESEAMS	sideseam (param) f_sstranslation (param) b_sstranslation (param) ytranslation (param) hemlevel (param) xscale (param) yscale (param) ss (ss_table) x (ss_table) y (ss_table) delta_y (ss_table) xend (ss_table) next_x (ss_table) next_y (ss_table)	f_xold b_xold yold f_xnew b_xnew y_new Front and back sideseams to the screen
5. DRAW_DARTS	f_sstranslation b_sstranslation xtranslation hip_pattern_width xscale cf cb control_level ytranslation wl_front_panel1 wl_front_panel2	cl_panel_dart front_panel_translation back_panel_translation front_side_translation back_side_translation back_dart_length back_panel_level back_side_degrees back_side_level front_side_dart_length front_side_dart_degrees front_side_level front_panel_dart_length front_panel_level

Procedure name	Input variables	Output variables
5a. READ_DART_WIDTHS	differential (param) dart_table (param)	diff wl_back_panel wl_back_side1 wl_back_side2 wl_front_side1 wl_front_side2 wl_front_panel1 wl_front_panel2 wl_total_back wl-total_front wl_tot_back_and_front cl_back_panel cl_back_side cl_front_side cl_front_panel cl_tot_back cl_tot_front cl_tot_back_and_front back_panel_degrees back_side_degrees front_side_degrees front_panel_degrees centerback_level back_panel_level back_side_level front_side_level front_panel_level centerfront_level back_dart_length front_side_dart_length front_panel_dart_length guide_hole_level
5b. DRAW_STRAIGHT_DART	dart_length (param) dart_center (param) dart_level (param) dart_width (param) xscale (param) yscale (param) ytranslation (param)	l_xnew r_xnew ynew xold yold {straight dart output to the screen}
5c. DRAW_CURVED_DART	xold (param) yold dart_degrees waist_level guide_hole_level dart_length controlline	

Procedure name	Input variables	Output variables
5c1. TRANSFORM_AND_DRAW	x (param) y (param) dart_degrees (param) xold (param) yold (param) guidehole_level (param) xscale (param) yscale (param) dart_center (param) y_translation (param)	xold (screen) yold (screen) ynew (screen) r_xnew (screen) l_xnew (screen) {left and right curved portion of dart to screen}
5c1a. ROTATE_X	x (param) y (param) theta (param)	ROTATE_X
5c1b. ROTATE_Y	x (param) y (param) theta (param)	ROTATE_Y
6. DRAW_WAISTLINE		
6a. DRAW_TO_PANEL_DART	center_line_level (param) left_panel_dart_end(param)	{waistline curve from centerline to panel dart to screen}
6b. DRAW_TO_SIDE_DART	right_panel_dart_end(param) left_side_dart_end(param)	{waistline curve from panel dart to side dart to screen}
6c. DRAW_TO_SIDESEAM	right_side_dart_end(param) sideseam_level (param)	{waistline curve from side dart to sideseam to screen}
7. LABEL_PATTERN		{textual pattern labels to screen}
POWER	x (param) y (param)	POWER

Appendix F

Source Code

External Documentation

The main purpose of this documentation is to aid in completing the implementation of the Mod-U-Lar Pattern System basic skirt pattern. The documentation will consist of five sections. The first portion will explain the current state of the implementation. The second section will discuss some knowledge that was gathered during this implementation, which will aid in completing the work. The third portion will discuss possible causes for the current errors. The fourtn section will discuss some suggestions on improving the program. The last section will discuss some helpful information, including some problems to be wary of with the Turbo Pascal language.

Current State of the Implementation

The hierarchy diagram is shown in figures 3.1-3.4 and may be used to understand the overall structure of the program. Most of the modules shown have been completed. Specifically, the modules measure, points, draw_guidelines, draw_sideseams, and label_pattern are completed. The module draw_darts is designed and coded, but is still in the debugging phase. The module draw_waistline is designed, but uncoded and untested. The following sections are intended as an aid in completing the testing phase on the draw_darts module and the coding and testing phase for the draw_waistline module.

Background Information

The purpose of this section is to discuss some background knowledge which may be needed to complete the implementation of the basic skirt pattern. The Mod-U-Lar Pattern System methodology may be used in conjunction with the skirt worksheet, information tables, and templates to draft a correctly fitting pattern. The following paragraphs will discuss how these were implemented.

The portion of the program dealing with the skirt worksheet has already been implemented. If any further information is desired on the subject, refer to the internal documentation of the program or the report.

The information tables were typed into dBaseII databases. There is one database for each table. The dBaseII databases were transformed into text files to allow access by Turbo Pascal routines. These files may be found in appendix D. The naming convention aids in identifying the tables; all the dart tables are listed by body type and subtype.

The templates for the skirt sideseams, darts, and waistline are all curves. It was decided that these curves would be derived from formulas to allow for the generation of more coordinates as they are needed. These formulas were derived using SAS subroutines which are accessible through the IBM/370 mainframe. The steps used for deriving a formula for a curve will be discussed in great detail, as they will be needed for the draw_waistline module.

The first step in deriving the formula for a curve is to plot it onto graph paper. The graph paper should be ten squares per inch, in order that the curve be scaled the same as the other curves. The next step is to create a file in the CMS environment using xedit. An example file is shown below.

```
DATA;
    INPUT Y X;
CARDS;
0 0
-40 1
-53 2
-61 3
-67 4
-72 5
-76 6
;
PROC GLM;
MODEL X = Y Y*Y Y*Y*Y Y*Y*Y*Y/P;
```

The inputs to this program are the x and y coordinates plotted on the graph paper. They follow the "CARDS;" line in the file above. The y coordinates are in the first column and are the first input item on each line. The x coordinates are in the second column and are in the second column of each line. Once this file is entered, exit xedit and save the file.

The next step is to generate an output file. Before execution of this program is possible, more memory is needed. More memory is accessible by typing the command "RESTOR 600K". Now, the program may be executed with the command "SAS fn", where fn is the file name. The output may be found ion a listing file.

Once the output file is generated, the formula must be examined for accuracy. The formula itself, as well as its predicted output,

may be found in the listing file. The predicted output of the formula may be found at the bottom of the listing under the heading "PREDICTED VALUE". IF the "OBSERVED VALUE" is equal to the rounded "PREDICTED VALUE" then the formula is accurate. Sometimes the curve must be broken up to achieve an accurate formula. Once it is determined that the formula is accurate, it may be found in the middle section of the listing, under the heading "ESTIMATE". The first item under "ESTIMATE" is multiplied times "1", the next item is multiplied times "X", the next item is multiplied times "X" squared, the next item is multiplied times "X" cubed, and so on. It is important to note that in this example "X" is the dependent variable, in other words, the value of "Y" is known and the value of "X" is derived by the formula.

The last step in deriving a formula is to test it using a small Turbo Pascal test program. This will determine if it actually generates the desired output.

Possible Sources of Errors

The draw_darts module is currently in the debugging phase. The straight portion of the dart, below the control line, appears to be correct. But there are some problems with the curved portion of the dart, which is above the control line. The major problem seems to be with the rotation. The curve should be continuing from the straight line in an upward V-shaped curve. Instead, it is going downward, making the dart resemble an M-shape. There also seems to be a problem with the beginning of the curve and the end of the

curve. The beginning of the curve moves downward, before curving up. And, the end of the curve is also at a downward angle. The second problem may also be caused by the incorrect rotation. Therefore, the best approach is probably to correct the problem with the rotation first.

The code for rotating has been written out as a separate program and tested separately. It is shown on pages 67-8. This program accepts screen input of the desired rotation in degrees. Execution is halted by entering a number of degrees greater than 400. The program output consists of screen as well as file output. The x and y coordinates, before and after rotation are sent to a file called "out.txt". Screen output consists of a line before and after rotation. The x and y coordinates are scaled as well as rotated before they are output to the screen, so that they appear in the proper proportions.

The screen output for the rotation program will be used to discuss the possible source of the problem. The angles of rotation tested were those with obvious results; the 45, 90, 180, 270, and 360 degree angles. The resulting angles where obviously incorrect. Another item which was noted was the difference in the length of the rotated and unrotated line. This difference implies that the scaling factor may be incorrect. It is also possible that the pixel sizes may vary on different parts of the screen. It is also possible that the formula used for the rotation is incorrect, but it has been checked and appears to be correct. This may imply a possibility of some sort of round-off error.

```
PROGRAM PATTERN (input,output);
const
  outfilename = 'out.txt';
var
  outfile: text;
  pi,x,y,degrees,scale,xscale,yscale: real;
  xold,yold,xnew,ynew: integer;

{To rotate a point (x,y) through a clockwise angle (theta) about}
{the origin of the coordinate system, x is rotated by:          }
{               x' = x cos theta + y sin theta                  }

FUNCTION ROTATE_X(x,y,theta: real): real;
begin
  rotate_x := x * cos(theta) + y * sin(theta)
END {ROTATE X};


{To rotate a point (x,y) through a clockwise angle (theta) about }
{the origin of the coordinate system, y is rotated by:          }
{               y' = -x sin theta + y cos theta                 }

FUNCTION ROTATE_Y(x,y,theta: real): real;
begin
  rotate_y := (-x) * sin(theta) + y * cos(theta)
END {ROTATE Y};
```

```
begin {main}

  pi := 3.14159;
  scale := 1.5;
  xscale := 1.0 * 1.5;
  yscale := (35/79) * scale;
  assign(outfile,outfilename);
  rewrite(outfile);
  xold := 0;
  yold := 0;
  x := 0.0;
  y := 100.0;

  write('Enter degrees of rotation to be plotted(quit >= 400): ');
  readln(degrees); writeln;

  while(degrees < 400) do
  begin
    HiRes;
    HiResColor(10);

    xnew := round(x);                    {Draw line before rotating}
    ynew := round(y);
    draw(round(xold*xscale+200),round(yold*yscale+100),
         round(xnew*xscale+200),round(ynew*yscale+100),1);
    writeln(outfile,'UNROTATED  ','xold= ',xold:4,' yold= ',yold:4);
    writeln(outfile,'           ','xnew= ',xnew:4,' ynew= ',ynew:4);

    xnew := round(rotate_x(x,y,degrees));     {Rotate and draw line}
    ynew := round(rotate_y(x,y,degrees));
    draw(round(xold*xscale+200),round(yold*yscale+100),
         round(xnew*xscale+200),round(ynew*yscale+100),1);
    writeln(outfile,'  ROTATED  ','xold= ',xold:4,' yold= ',yold:4);
    writeln(outfile,'           ','xnew= ',xnew:4,' ynew= ',ynew:4);
    writeln(outfile,'ROTATION is ',degrees:5:1,' degrees');
    writeln(outfile);

    write('Enter degrees of rotation to be plotted : ');
    readln(degrees); writeln;
  end;

  close(outfile);
end.
```

Once the reason for the incorrect rotation is determined and the error is corrected, the correction may be added to the source code itself. The source code includes trace messages which may aid in continued debugging and testing. These trace messages may be turned on and off by setting the constants "debug1" and "debug2" to true or false respectively. The constant "debug1" was used mainly for the first phases of testing. It may be desirable to turn only the "debug2" flag on. The trace messages will be output to a file "out.txt".

Suggestions for Improvement

There are two items in the current source program which could reduce the program's complexity. As was discussed previously, the curve templates were plotted on graph paper that was ten boxes per inch. Thus, each point calculated on the curve was at this scale. The other data, which was read from an information table or input by the user, was being treated as one point for each centimeter. This resulted in two different scales being used. To alleviate this problem, a conversion factor for the data in centimeters was used. It would be better if this conversion were used on the curves, because there are less of them than the input data.

Another item which might simplify the program greatly, is to divide the screen and the pattern coordinates. That is to make all points on the pattern relative to the pattern and all screen points relative to the screen. In other words, all rotation, scaling, and translation is done right before plotting. The scaling done for

the previously mentioned conversion should not be included in this.

Helpful Information

There are a few more items which should be mentioned. There are some bugs in Turbo Pascal, which will be less of a problem, if one is aware of them. Turbo Pascal does not detect undefined variables. If a variable has no value and it is used in a statement, then some "junk" value is used. An unformated variable in an output statement will produce incorrect output. Integer type constants used in conjunction with real values or variables will cause incorrect results.

Some correct test data will also be necessary as program input. Following are some sample measurements provided by Helen Brockman.

HEIGHT	WEIGHT	TORSOLINE GIRTH	WAISTLINE GIRTH	BACK LENGTH	BODY TYPE	DIFFERENTIAL
5'4"	208	129	93	21.5	C+	36
5'3"	165	113	83	18.0	A+	30
5'3"	155	109	87	20.0	B+	22
5'2"	118	93	67	19.5	B=	26

In conclusion, it should be mentioned that studying and understanding Brockman's "Mod-U-Lar Pattern System Skirt Book" is an important factor in understanding the source code. The overall system discussion in this report will also be helpful to the next implementor.

```
PROGRAM PATTERN (input,output);
const
  debug1 = false;
  debug2 = true;
  outfilename = 'out.txt';
type
  filename = string[12];
var
  outfile: text;

  differential,                    {Remaining skirt worksheet measurements}
  sideseam,
  skirt_length,
  waist_pattern_width,hip_pattern_width:integer;

  f_sstranslation,b_sstranslation,ytranslation,xtranslation, {Translations}
  waistlevel,controllevel,hiplevel,hemlevel,                 {Grainline levels}
  cb,cf: integer;                                {Centerfront and centerback}
  xscale,yscale: real;                                       {Scaling factors}

  quit,enter: char;                                {Error checking variables}
  error: boolean;
  i: integer;                                                  {loop variable}

  dart_table:filename;                {Table containing dart measurements}

  diff: integer;                                             {dart table entries}
  wl_back_panel: integer;
  wl_back_side1: integer;
  wl_back_side2: integer;
  wl_front_side1: integer;
  wl_front_side2: integer;
  wl_front_panel1: integer;
  wl_front_panel2: integer;
  wl_tot_back: integer;
  wl_tot_front: integer;
  wl_tot_back_and_front: integer;
  cl_back_panel: integer;
  cl_back_side: integer;
  cl_front_side: integer;
  cl_front_panel: integer;
  cl_tot_back: integer;
  cl_tot_front: integer;
  cl_tot_back_and_front: integer;
  back_panel_degrees: real;
  back_side_degrees: real;
  front_side_degrees: real;
  front_panel_degrees: real;
  centerback_level: integer;
  back_panel_level: integer;
  back_side_level: integer;
  front_side_level: integer;
  front_panel_level: integer;
```

```
  centerfront_level: integer;
  back_dart_length: integer;
  front_side_dart_length: integer;
  front_panel_dart_length: integer;
  guidehole_level: integer;



{Input or compute all measurements needed for the basic skirt pattern}
PROCEDURE MEASURE(var differential, sideseam,
                  waist_pattern_width, hip_pattern_width: integer;
                  var dart_table: filename; var error: boolean);
var
  height,weight,                          {Person's input measurements}
  waistline_girth,torsoline_girth: integer;
  back: real;


  {Input the person's height, weight, waistline girth, torsoline}
  {girth, and back length.                                       }


  PROCEDURE INPUT_MEASUREMENTS(var height, weight, waistline_girth,
                               torsoline_girth:integer; var back: real);
  var
    correct: char;
  begin
    correct := 'y';

    {Accept all input measurements, until user is satisfied that}
    {they are correct.                                          }
    while (correct <> 'n') do
    begin
      HiRes;
      HiResColor(10);
      writeln; writeln; writeln;
      write('     enter height in cm: '); readln(height);
      writeln('      ',height:3,' cm');
      write('     enter weight in pounds: '); readln(weight);
      writeln('      ',weight:3,' lbs');
      write('     enter waistline girth in cm: '); readln(waistline_girth);
      writeln('      ',waistline_girth:3,' cm');
      write('     enter torsoline girth in cm: '); readln(torsoline_girth);
      writeln('      ',torsoline_girth:3,' cm');
      write('     enter center back length in cm: '); readln(back);
      writeln('      ',back:4:1,' cm');
      writeln;
      write('     Do you wish to correct any of the measurements? y or n');
      readln(correct); writeln

    end{while}
  END {INPUT_MEASUREMENTS};
```

```
{Adjust the waistline girth to be smaller than the torsoline}
{by an even number of centimeters.                          }

FUNCTION ADJUST_WAISTLINE(waistline_girth: integer):integer;
begin
  if odd(torsoline_girth - waistline_girth) then
     waistline_girth := waistline_girth - 1;
  adjust_waistline := waistline_girth;
END {ADJUST_WAISTLINE};


{Compute the differential, which is the difference between }
{the waistline and the torsoline girth.                    }

FUNCTION COMPUTE_DIFFERENTIAL(torsoline_girth,waistline_girth: integer)
                              :integer;
begin
  compute_differential := torsoline_girth - waistline_girth;
END {COMPUTE_DIFFERENTIAL};



{Determine the body type from the height and weight and the    }
{length of the centerback measured from waistline to torsoline.}

FUNCTION BODY_TYPE(height,weight,differential:integer; back: real;
                   var skirt_length: integer):filename;
var
  dart_table: filename;
  difference: real;
  norm_wt: integer;


  {determine the normal weight for the person's height}

  PROCEDURE NORMAL_WEIGHT(ht,wt,diff: integer;
                          var norm_wt,skirt_length:integer);
  var
    i,height: integer;
    ht_weight: text;
  begin
    assign(ht_weight,'a:ht_wt.txt');
    reset(ht_weight);
    height := 0;

    {Find entries for person's height}
    while (height < ht) and (not eof(ht_weight)) do
    begin
      readln(ht_weight);
      if not (eof(ht_weight)) then
      begin
        read(ht_weight,height);
      end{if}
    end{while};
```

```
    if not (eof(ht_weight)) then
    begin
      {Determine the normal weight for person's height and differential}
      i := 10;
      repeat
        i := i + 2;
        read(ht_weight,norm_wt);
      until (i = diff);

      {Determine the skirt length}
      repeat
        i := i + 2;
        read(ht_weight,skirt_length);
      until (i = 42);
    end{if};

  END {NORMAL_WEIGHT};


begin {BODY_TYPE}

  dart_table := 'a:';

  {compute difference between front and back length}
  difference :=  20 - back;

  {Determine if figure type is a, b, or c                                    }
  {A normal front and back, less than 1 cm difference is figure type 'b'}
  {A long front and a short back is figure type 'a'                          }
  {A long back and short front is figure type 'c'                            }
  if abs(difference) < 1 then  dart_table :=  dart_table + 'b'
  else
    if difference >= 1 then  dart_table := dart_table + 'a'
    else dart_table := dart_table + 'c';

  {Determine subtypes. If figure type is 'b' then subtype is determined}
  {by deviation from normal weight.  If figure type is 'a' or 'c' then }
  {subtype is determined by front and back difference                   }
  if dart_table[3] = 'b' then
  begin
    normal_weight(height,weight,differential,norm_wt,skirt_length);
    if debug1 then writeln(outfile,'returning from normal_weight');

    {Subtype 'delta' is at least 15 pounds underweight}
    if weight <= norm_wt - 15 then
      dart_table := dart_table + 'delta'
    else
      {Subtype 'pluss' is at least 20 pounds overweight}
      if weight >= norm_wt + 20 then
        dart_table := dart_table + 'pluss'
      else
        {Subtype 'equal' is from 15 lbs under to 20 lbs over}
        dart_table := dart_table + 'equal'
```

```
  end {if figure type is 'b'}

  else {if figure type is 'a' or 'c'}
  begin
    difference := abs(difference);

    {subtype is 'delta' if difference between front and back is > 3 cm}
    if difference >= 3 then
      dart_table := dart_table + 'delta'
    else
      {Subtype is 'pluss' if difference between front & back is >= 2 cm}
      if difference >= 2 then
        dart_table := dart_table + 'pluss'
      else
        {Subtype is 'equal' if difference between front & back is < 2cm}
        dart_table := dart_table + 'equal'
  end {else figure type is 'a' or 'c'};

  {The dart measurement file is a text file}
  dart_table := dart_table + '.txt';
  if debug2 then writeln('IN BODYTYPE the table is ',dart_table:12);
  body_type := dart_table;

END {BODY_TYPE};


{Determine the number of the sideseam template to be used.}

FUNCTION DETERMINE_SIDESEAM(diff:integer):integer;
{case diff of
  14 : sideseam := 1;
  16 : sideseam := 1;
  18 : sideseam := 2;
        ..
  40 : sideseam := 7
 end}
begin
  determine_sideseam := round((diff / 2 - 6) / 2)
END {DETERMINE_SIDESEAM};



{Compute the width of the pattern at the waist and hip}
PROCEDURE CALC_WAIST_HIP_WIDTHS(var waist_pattern_width,
                                hip_pattern_width:integer;
                                waistline_girth,differential:integer);


  {Calculate the hipwidth.  Needed to compute hiplength pattern width.}
  function CALC_ADD_TO_HIP(differential: integer):integer;
  var
    i,add_to_hip: integer;
```

```
    begin
      add_to_hip := 0;

      {every other differential, a 3 is added,}
      {then a 6 is added to the hip width.    }
      For i := 1 to ((differential - 14) mod 2) do

        if i mod 2 = 0 then
          add_to_hip := add_to_hip + 3
        else
          add_to_hip := add_to_hip + 6;
      {endfor}

      calc_add_to_hip := add_to_hip
    END {CALC_ADD_TO_HIP};


  begin {CALC_WAIST_HIP_WIDTHS}
    {Compute pattern widths for people with waistlines smaller than 80 cm}
    if waistline_girth < 80 then
    begin
      waist_pattern_width := 191 + trunc((waistline_girth - 60) * 2.5
                               + (differential - 14) * 1.5);
      hip_pattern_width := 201 + trunc((waistline_girth - 60) * 2.5
                           + calc_add_to_hip(differential));
    end{if}
    else {waistlines 80 cm or larger}
    begin
      waist_pattern_width := 191 + trunc((waistline_girth - 60) * 5
                               + (differential - 14) * 1.5);
      hip_pattern_width := 201 + trunc((waistline_girth - 60) * 5
                           + calc_add_to_hip(differential))
    end{else}
  END {CALC_WAIST_HIP_WIDTHS};


begin {MEASURE}

  {Check for people whose measurements are beyond}
  {the limits, while inputting measurements.     }

  input_measurements(height,weight,waistline_girth,torsoline_girth,back);

  {Adjust the waistline}
  waistline_girth := adjust_waistline(waistline_girth);

  {Compute person's differential}
  differential := compute_differential(torsoline_girth,waistline_girth);

  {Determine the person's body type in order to determine}
  {the correct dart table name.                          }
  dart_table := body_type(height,weight,differential,back,skirt_length);
```

```
  {Check for values that are out of the bounds of current tables}
  if (not error) then

   {This condition is more logically formatted on the source code disk}
   if ((differential >= 16) and (differential <= 40) and
       (dart_table[2] <> 'd')) or ((dart_table[2] = 'd')
       and (((dart_table[1] = 'a')and (differential >= 16) and
       (differential <= 36)) or ((dart_table[1] = 'b') and
       (differential >= 16) and (differential <= 28))
       or((dart_table[1] = 'c')and (differential >= 24) and
       (differential <= 40)))) then
   begin
     {Determine sideseam template to use}
     sideseam := determine_sideseam(differential);

     {Determine the pattern width at the waistline}
     calc_waist_hip_widths(waist_pattern_width,hip_pattern_width,
                           waistline_girth,differential);
   end{if}
   else
   begin
     error := true;
     writeln('**** ERROR - The differential of ',differential:4,
             ' is out of bounds!')
   end{else}

END {MEASURE};



{Calculate the points to be plotted by the graphics routines}

PROCEDURE POINTS(skirt_length,waist_pattern_width: integer;
                 var f_sstranslation,b_sstranslation,xtranslation,
                 ytranslation, waistlevel,controllevel,hiplevel,hemlevel,
                 cb,cf: integer; var xscale,yscale: real);
var
  waistwidth,screen_right,screen_left: integer;
 scale: real;
begin
  scale := 1.5;                      {               scale               }
  xscale := 1.0 * scale;             {               xscale              }
  yscale := (35/79) * scale;     {yscale(35mm of x are equiv to 79 mm of y)}
  screen_right := 639;               {  coordinate of screen's right side }
  screen_left := 0;                  {  coordinate of screen's left side  }
  xtranslation := round(40 * xscale); {pattern distance from screen's side}
  ytranslation := round(40 * yscale); {pattern distance from top of screen}
  cf := screen_left + xtranslation;     {x coordinate of center front line}
  cb := screen_right - xtranslation;    {x coordinate of center back line }
  waistlevel := ytranslation;                  { level of the waist line }
  controllevel := round(waistlevel + 10/0.254 * yscale);{controlline level}
  hiplevel := round(waistlevel + 20/0.254 * yscale);    {level of hip line}
```

```
  {scale and translate the skirt's hem}
  hemlevel := round((skirt_length/0.254) * yscale + ytranslation);

  {convert the waistwidth from mm to cm and scale}
  waistwidth := round(waist_pattern_width / 10 / 0.254 * xscale);

  {calculate the front and back sideseam translations}
  f_sstranslation := cf + waistwidth;
  b_sstranslation := cb - waistwidth;

END {POINTS};



{Draw the guidelines for the skirt pattern}

PROCEDURE DRAW_GUIDELINES(f_sstranslation,b_sstranslation,
                          waistlevel,hemlevel,cf,cb: integer);
begin
  {This activates 640X200 high resolution graphics screen, and gives x}
  {coordinates between 0 and 639 and y coordinates between 0 and 199. }
  HiRes;
  HiResColor(10);    {set color to green}

  {draw the center front line}
  draw(cf,waistlevel,cf,hemlevel,1);

  {draw the center back line}
  draw(cb,waistlevel,cb,hemlevel,1);

  {draw the front waistline}
  draw(cf,waistlevel,f_sstranslation,waistlevel,1);

  {draw the back waistline}
  draw(cb,waistlevel,b_sstranslation,waistlevel,1);

END {DRAW_GUIDELINES};



{This procedure draws the front and back sideseams of the skirt}

PROCEDURE DRAW_SIDESEAMS(sideseam,f_sstranslation,b_sstranslation,
                         ytranslation,hemlevel:integer;xscale,yscale:real);
var
  ss_table: text;
  x,y: real;
  delta_y,delta_x,ss,
  f_xold,b_xold,yold,f_xnew,b_xnew,ynew,xend,next_x,next_y: integer;


  {This function is used by curve-plotting routines to find the}
```

```
  {result of the first argument to the second argument power.  }
  {since exp(ln(x)) = x
   then  x**y    =
   and   exp(ln(x**y))  =
   and   exp(y*ln(x)) }

  FUNCTION POWER(x,y : real):real;
  begin
    if x = 0 then power := 1
    else power := exp(y*ln(x))
  END {POWER};


begin {DRAW_SIDESEAMS}
  assign(ss_table,'a:sideseam.txt');
  reset(ss_table);

  {Get the starting location of the sideseam and }
  {the slope at the bottom of the skirt sideseam.}
  repeat
    readln(ss_table);
    read(ss_table,ss);

  until (ss = sideseam);

  if debug1 then writeln(outfile,'ss= ',ss:3);

  read(ss_table,x,y,delta_y,xend);

  if debug1 then
    writeln(outfile,'x= ',x:5:1,' y= ',y:5:1,' delta_y= ',
            delta_y:3,' xend= ',xend:3);

  {scale and translate starting positions}
  f_xnew := f_sstranslation + round(x * xscale);
  b_xnew := b_sstranslation - round(x * xscale);
  ynew := ytranslation + round(y * yscale);

  if debug1 then
  begin
    writeln(outfile);
    writeln(outfile,'drawing the curved portion of the sideseam')
  end{if};

  x := 0.0;
  {draw the curved portion of the sideseam}
  while (x <= xend) do
  begin

    {calculate the next y coordinate depending on the sideseam number}
    case sideseam of

      1: y := 9.19167 * x - 0.4375 * sqr(x) + 0.420833 * power(x,3);
```

```
    2: y := round(5.72587 * x + 0.523755 * sqr(x) +
                  0.0396148 * power(x,3) + 0.00851986 * power(x,4));

    3: y := round(2.42556 * x + 2.221102 * sqr(x) - 0.414444 * power(x,3)
                  + 0.0362731 * power(x,4));

    4: y := round(-0.01680672 + 6.6394958 * x - 0.93267974 * sqr(x) +
                  + 0.22679739 * power(x,3) - 0.01111111 * power(x,4));

    5: y := round(-0.00061735 + 3.57417713 * x + 0.21739133 * sqr(x)
                  + 0.00437014 * power(x,3) + 0.00061293 * power(x,4));

    6: y := round(-0.01587302 + 4.02910053 * x - 0.02777778 * sqr(x) +
                  0.01851852 * power(x,3) - 7.0753588e-15 * power(x,4));

    7: y := round(0.00623389 + 3.66619266 * x - 0.01191289 * sqr(x)
                  + 0.01465435 * power(x,3) - 0.00051276 * power(x,4));

  end{case};

  yold := ynew;
  f_xold := f_xnew;
  b_xold := b_xnew;
  ynew := ytranslation + round(y * yscale);       {scale and translate y}
  f_xnew := f_sstranslation + round(x * xscale);  {scale and translate x}
  b_xnew := b_sstranslation - round(x * xscale);  {scale and translate x}

  if debug1 then
  begin
    writeln(outfile);
    writeln(outfile,'fold=',f_xold:4,' bold=',b_xold:4,' yold=',yold:4);
    writeln(outfile,'fnew=',f_xnew:4,' bnew=',b_xnew:4,' ynew=',ynew:4)
  end{if};

  {front sideseam}
  draw(f_xold,yold,f_xnew,ynew,1);

  {draw back sideseam}
  draw(b_xold,yold,b_xnew,ynew,1);

  x := x + 0.5;   {increment x}

end{while};

f_xold := f_xnew;
b_xold := b_xnew;
yold := ynew;

{finish drawing the curved part of sideseams 4, 5, 6, and 7}
read(ss_table,next_x,next_y);
while (next_x <> 0) do
begin

  if debug1 then
```

```
  begin
    writeln(outfile);
    writeln(outfile,'next_x=',next_x:4,' next_y=',next_y:4);
  end{if};

  {scale and translate the new coordinates}
  b_xnew := round(b_sstranslation - next_x * xscale);
  f_xnew := round(f_sstranslation + next_x * xscale);
  ynew := round(ytranslation + next_y * yscale);

  {draw the front sideseam}
  draw(f_xold,yold,f_xnew,ynew,1);

  {draw the back sideseam}
  draw(b_xold,yold,b_xnew,ynew,1);

  if debug1 then
  begin
    writeln(outfile);
    writeln(outfile,'continuation of sideseam curve');
    writeln(outfile,'fold=',f_xold:4,' bold=',b_xold:4,' yold=',yold:4);
    writeln(outfile,'fnew=',f_xnew:4,' bnew=',b_xnew:4,' ynew=',ynew:4)
  end{if};

  f_xold := f_xnew;
  b_xold := b_xnew;
  yold := ynew;
  read(ss_table,next_x,next_y);

end{while};

{The remainder of the sideseam is a straight   }
{line, which may be calculated using its slope.}
ynew := hemlevel;
delta_x := round((ynew - yold) / delta_y);
f_xnew := f_xold + delta_x;
b_xnew := b_xold - delta_x;

if debug1 then
begin
  writeln(outfile);
  writeln(outfile,'straight end of sideseam');
  writeln(outfile,'f old=',f_xold:4,' b old=',b_xold:4,' y old=',yold:4);
  writeln(outfile,'f new=',f_xnew:4,' b new=',b_xnew:4,' y new=',ynew:4)
end{if};

draw(f_xold,yold,f_xnew,ynew,1);  {draw remainder of the front sideseam}
draw(b_xold,yold,b_xnew,ynew,1);  {draw remainder of the back sideseam }

{now that we know what x is at the hem, draw the front hemline}
draw(cf,hemlevel,f_xnew,ynew,1);

{also, draw the back hemline}
draw(cb,ynew,b_xnew,ynew,1)
```

```
END {DRAW_SIDESEAMS};



{To rotate a point (x,y) through a clockwise angle (theta) about the}
{origin of the coordinate system, x is rotated by:                   }
{              x' = x cos theta + y sin theta                        }

FUNCTION ROTATE_X(x,y,theta: real): real;
begin
  rotate_x := x * cos(theta) + y * sin(theta)
END {ROTATE X};



{To rotate a point (x,y) through a clockwise angle (theta) about the}
{origin of the coordinate system, y is rotated by:                   }
{              y' = -x sin theta + y cos theta                       }

FUNCTION ROTATE_Y(x,y,theta: real): real;
begin
  rotate_y := (-x) * sin(theta) + y * cos(theta)
END {ROTATE Y};



{Draw skirt darts}
PROCEDURE DRAW_DARTS(dart_table: filename; differential, f_sstranslation,
                     b_sstranslation,xtranslation,waistlevel: integer);
var
  hip_width: real;
  cl_panel_dart,front_panel_translation,back_panel_translation,dart_level,
  l_xnew,r_xnew,ynew,front_side_translation,back_side_translation: integer;
  front_panel_dart_is_straight: boolean;


  {This power function allows calculation of a negative number}
  {to a positive power.                                       }
  function power(number,exponent:real): real;
  var
    product: real;
    i: integer;
  begin
    product := 1.0;

    for i := 1 to round(exponent) do
      product := product * number;

    power := product;
  end;
```

```
{Read in all the measurements for the darts' widths, degrees,}
{levels and lengths for a given differential and figure type.}
{Send information to the printer.                            }

PROCEDURE READ_DART_WIDTHS(differential:integer;dart_table:filename);
var
  dart_width: text;
begin
  assign(dart_width,dart_table);
  reset(dart_width);
  diff := 0;
  if debug2 then
  begin
    writeln(outfile,'READING DART WIDTHS FROM ',dart_table:12,' TABLE');
    writeln(outfile,'differential= ',differential:3,' diff= ',diff:3);
  end;

  while (not eof(dart_width)) and (diff < differential) do
  begin
    writeln(outfile,'IN LOOP ... diff is ',diff:5);
    readln(dart_width,diff,wl_back_panel,wl_back_side1,
          wl_back_side2,wl_front_side1,wl_front_side2,wl_front_panel1,
          wl_front_panel2,wl_tot_back,wl_tot_front,wl_tot_back_and_front,
          cl_back_panel,cl_back_side,cl_front_side,cl_front_panel,
          cl_tot_back,cl_tot_front,cl_tot_back_and_front,
          back_panel_degrees,back_side_degrees,front_side_degrees,
          front_panel_degrees,centerback_level,back_panel_level,
          back_side_level,front_side_level,front_panel_level,
          centerfront_level,back_dart_length,front_side_dart_length,
          front_panel_dart_length,guidehole_level)
  end{while};

  if debug2 then
  begin
    writeln(outfile,'In READ DARTS');
    writeln(outfile,'diff= ',diff:5,wl_back_panel:5,wl_back_side1:5);
    writeln(outfile,wl_back_side2:5,wl_front_side1:5,wl_front_side2:5,
            wl_front_panel1:5);
    writeln(outfile,wl_front_panel2:5,wl_tot_back:5,wl_tot_front:5,
            wl_tot_back_and_front:5);
    writeln(outfile,cl_back_panel:5,cl_back_side:5,cl_front_side:5,
            cl_front_panel:5);
    writeln(outfile,cl_tot_back:5,cl_tot_front:5,
            cl_tot_back_and_front:5);
    writeln(outfile,'dart degrees= ',back_panel_degrees:10:5,
            back_side_degrees:10:5,front_side_degrees:10:5);
    writeln(outfile,front_panel_degrees:10:5,centerback_level:5,
            back_panel_level:5);
    writeln(outfile,back_side_level:5,front_side_level:5,
            front_panel_level:5);
    writeln(outfile,centerfront_level:5,back_dart_length:5,
            front_side_dart_length:5);
    writeln(outfile,front_panel_dart_length:5,'  guidehole level = ',
            guidehole_level:5)
```

```
  end{if}

END {READ_DART_WIDTHS};


{This procedure converts the dart measurements from mm to cm.  It also }
{converts the measurements to the same scale as the graph paper used to}
{plot the curves.  Each square on the graph paper is represented by one}
{x-coordinate on the screen and measured .254 cm.  All measurements are}
{scaled by appropriate scale factor.                                    }

PROCEDURE CONVERT_DART_MEAS;
var
  conversion_factor: real;
begin
  {Divide every measurement by 10 for conversion to cm, then by .254}
  {for same scale as curves plotted on graph paper.                 }
  conversion_factor := 10 * 0.254;

  {Convert and scale all dart widths}
  wl_back_panel := round(wl_back_panel / conversion_factor);
  wl_back_side1 := round(wl_back_side1 / conversion_factor);
  wl_back_side2 := round(wl_back_side2 / conversion_factor);
  wl_front_side1 := round(wl_front_side1 / conversion_factor);
  wl_front_side2 := round(wl_front_side2 / conversion_factor);
  wl_front_panel1 := round(wl_front_panel1 / conversion_factor);
  wl_front_panel2 := round(wl_front_panel2 / conversion_factor);
  cl_back_panel := round(cl_back_panel / conversion_factor);
  cl_back_side  := round(cl_back_side / conversion_factor);
  cl_front_side := round(cl_front_side / conversion_factor);
  cl_front_panel := round(cl_front_panel / conversion_factor);

  {Convert and scale all dart levels}
  centerback_level := round(centerback_level / conversion_factor);
  back_panel_level := round(back_panel_level / conversion_factor);
  back_side_level  := round(back_side_level / conversion_factor);
  front_side_level := round(front_side_level / conversion_factor);
  front_panel_level := round(front_panel_level / conversion_factor);
  centerfront_level := round(centerfront_level / conversion_factor);
  guidehole_level := round(guidehole_level / conversion_factor);

  {Convert and scale all dart lengths}
  back_dart_length := round(back_dart_length / conversion_factor);
  front_side_dart_length := round(front_side_dart_length
                                  / conversion_factor);
  front_panel_dart_length := round(front_panel_dart_length
                                   / conversion_factor);
END {CONVERT_DART_MEAS};


{This procedure draws the straight portion of a dart. This may be the}
{entire dart, or it may only be the portion of the dart below the     }
{control line.                                                        }
PROCEDURE DRAW_STRAIGHT_DART(dart_length,                {dart length }
```

```
                                dart_level,      {waist or control level}
                                dart_center,           {dart centerline}
                                dart_width,      {waist or control width}
                                ytranslation: integer;
                                xscale,yscale: real;
                                var l_xnew,r_xnew,ynew: integer);
var
  xold,yold: integer;
  x: real;
begin
  if debug2 then
  begin
    writeln(outfile,'dart length = ',dart_length:10);
    writeln(outfile,'dart level = ',dart_level:10);
    writeln(outfile,'dart center = ',dart_center:10);
    writeln(outfile,'dart width = ',dart_width:10);
  end;

  {Set the starting position. x is the center of the dart.}
  xold := dart_center;

  {y is the length of the dart from the waistline}
  yold := round(dart_length + ytranslation);

  {Compute the left and right endpoints of the dart at the waistline.}
  l_xnew := round(dart_center - (dart_width / 2));
  r_xnew := round(dart_center + (dart_width / 2));
  ynew := dart_level;

  {Draw the right side of the dart}
  draw(xold,yold,r_xnew,ynew,1);

  {draw the left side of the dart}
  draw(xold,yold,l_xnew,ynew,1)
END {DRAW_STRAIGHT_DART};



{This procedure continues drawing the dart. It draws the curved portion}
{above the control line.  The last point plotted is passed to the      }
{procedure.  This will be the starting point for the curve.            }
PROCEDURE DRAW_CURVED_DART(l_xold,r_xold,yold,guidehole_level,waist_level
                           ,dart_center, dart_width,dart_length: integer;
                           dart_degrees: real);
var
  curve_begin,curve_end,increment,x,y,conversion_factor,controlline:real;

  {NOTE: 1st rotate, then scale, then translate}
  {This procedure transforms the x,y coordinates given by rotating,}
  {scaling, and translating.  Then it draws to that point.        }
```

```
  PROCEDURE TRANSFORM_AND_DRAW(x,y,dart_degrees:real; var l_xold,r_xold,
                               yold: integer;
                               dart_length,dart_center: integer);
  var
    r_xnew,l_xnew,ynew: integer;

  begin {TRANSFORM_AND_DRAW}
    if debug2 then
    begin
      writeln(outfile,'Entering TRANSFORM AND DRAW');
      writeln(outfile,'x= ',x:10:5,'  y= ',y:10:5,
              '  dart degrees=',dart_degrees:10:5);
      write(outfile,'dart length= ',dart_length:10,'  dart center= ');
      writeln(outfile,dart_center:5,'  ytranslation= ',ytranslation:5)
    end;

    {1st rotate, then scale, and then translate each point.}
    r_xnew := dart_center + round(rotate_x(x,y,dart_degrees/2) * xscale);
    l_xnew := dart_center - round(rotate_x(x,y,dart_degrees/2) * xscale);
    ynew := round ((dart_length * yscale + ytranslation)
                  + rotate_y (x,y,dart_degrees/2) * yscale);

    if debug2 then
    begin
      writeln(outfile,'r x old= ',r_xold:5,'l x old= ',l_xold:5,
              '  yold= ',yold:5);
      writeln(outfile,'r x new= ',r_xnew:5,'l x new= ',l_xnew:5,
              '  ynew= ',ynew:5);
    end;

    {draw the right side of the dart}
    draw(r_xold,yold,r_xnew,ynew,1);

    {draw the left side of the dart}
    draw(l_xold,yold,l_xnew,ynew,1);

    l_xold := l_xnew;
    r_xold := r_xnew;
    yold := ynew;
  END {TRANSFORM AND DRAW};


begin {DRAW_CURVED_DART}
  {The control line is 10 cm down from the waistline.}
  conversion_factor := 0.254;
  controlline := 10 / conversion_factor;

  {Determine which part of the #60 curve to start on. The curved portion}
  {of the dart starts on the control line.                              }
  curve_begin := -(guidehole_level - controlline);

  {Set the Y coordinate to the beginning of the curve.}
  y := curve_begin;
```

```
{Determine where to stop on the curve.}
curve_end := curve_begin - (controlline + waist_level);

if debug2 then
begin
  writeln(outfile,'In DRAW CURVED DART');
  write(outfile,'guidehole level= ',guidehole_level:4);
  writeln(outfile,' control line= ',controlline:5,' y= ',y:10:5,
          '  waist level= ', waist_level:5);
  writeln(outfile,' curve begin',curve_begin:10:5,' curve end= ',
          curve_end:10:5);
end;

{Set increment value, may make smaller for greater accuracy.}
increment := 1.0;

{Derive x coordinate from curve begin, which is a y coordinate.}
x := round(0.00071341 + 0.12809169 * y + 0.0076817 * power(y,2) +
           0.00013192 * power(y,3) + 8.7836639e-07 * power(y,4));

{Calculate the next point, because the dart is already drawn}
{to the control line                                        }
x := x + increment;   {increment x}
y := (-0.06746032) - 61.08597884 * x + 26.71527773 * power(x,2)
     - 5.66203704 * power(x,3) + 0.4375 * power(x,4); {derive y}

{Derive all y coordinates from x for greater accuracy.}
{1st formula will derive y accurately, until x > 5.   }
while x <= 5 do
begin
  transform_and_draw(x,y,dart_degrees,r_xold,l_xold,yold,
                     dart_length,dart_center);

  x := x + increment;   {increment x}
  y := (-0.06746032) - 61.08597884 * x + 26.71527778 * power(x,2)
       - 5.66203704 * power(x,3) + 0.4375 * power(x,4); {derive y}

end{while};

{2nd formula will derive y accurately, if x > 5.   }
while y > curve_end do
begin
  y := (-38.60317459) - 9.55291006 * x + 0.66666667 * power(x,2)
       - 0.01851852 * power(x,3) + 3.2812826e-12 * power(x,4);{derive y}

  transform_and_draw(x,y,dart_degrees,l_xold,r_xold,yold,
                     dart_length,dart_center);

  x := x + increment   {increment x}

end{while};

{If Y coordinate is past the end of the dart,}
{then set it to the endpoint                 }
```

```
    if y > curve_end then
      y := curve_end;

    {Draw to the end of the dart}
    transform_and_draw(x,y,dart_degrees,r_xold,l_xold,yold,dart_length,
                       dart_center);

  end {DRAW_CURVED_DART};



begin {DRAW DARTS}
  {read in information from dart data table, for given differential}
  read_dart_widths(differential,dart_table);

  {Determine if the front panel dart is curved or straight}
  front_panel_dart_is_straight := wl_front_panel1 = wl_front_panel2;

  {Convert dart measurements from cm to mm and scale}
  convert_dart_meas;

  {Calculate hip width. Convert cm to mm and}
  {points not plotted on graph paper        }
  hip_width := hip_pattern_width / 10 / 0.254 * xscale;

  {Calculate panel darts' centerline positions. 1/3 of hip width}
  cl_panel_dart := round(hip_width/3);
  front_panel_translation := cf + cl_panel_dart;
  back_panel_translation := cb - cl_panel_dart;

  {Calculate side darts' centerline position. }
  {Midpoint between panel dart and sideseam.   }
  front_side_translation := (front_panel_translation + f_sstranslation)
                            div 2;
  back_side_translation := (back_panel_translation + b_sstranslation)
                           div 2;

  if debug2 then
  begin
    write(outfile,'back dart length = ',back_dart_length:3);
    writeln(outfile,' waist level =', waistlevel:3,' back panel level = ',
            back_panel_level:3);
    writeln(outfile,' back panel transl = ',back_panel_translation:3,
          ' wl back panel width = ',wl_back_panel:3,' ytranslation = ',
          ytranslation:3,' xscale = ',xscale: 5:2);
    writeln(outfile,' yscale = ',yscale:5:2,' l_xnew = ',l_xnew:3,
            ' r_xnew = ', r_xnew:3,' ynew = ',ynew:3);
  end{if};

  if debug2 then writeln(outfile,'Drawing straight back panel dart');
  {draw back panel dart, which will always be straight}
  dart_level := waistlevel + (-back_panel_level);
```

```
  draw_straight_dart(back_dart_length,dart_level,back_panel_translation,
                     wl_back_panel,ytranslation,xscale,yscale,
                     l_xnew,r_xnew,ynew);

  if debug2 then writeln(outfile,'Drawing curved back side dart');
  {draw the back side dart, which will always be straight beneath the}
  {control line, and curved above it.                                 }
  draw_straight_dart(back_dart_length,controllevel,back_side_translation,
                     cl_back_side,ytranslation,xscale,yscale,
                     l_xnew,r_xnew,ynew);

  draw_curved_dart(l_xnew,r_xnew,ynew,guidehole_level,back_side_level,
                   back_side_translation,wl_back_side2,
                   back_dart_length,back_side_degrees);

  if debug2 then writeln(outfile,'Drawing curved front side dart');
  {draw the front side dart,which will always be straight beneath the}
  {control line, and curved above it.                                 }
  draw_straight_dart(front_side_dart_length,controllevel,
                     front_side_translation, cl_front_side,ytranslation,
                     xscale,yscale, l_xnew,r_xnew,ynew);

  draw_curved_dart(l_xnew,r_xnew,ynew,guidehole_level,front_side_level,
                   front_side_translation,wl_front_side2,
                   front_side_dart_length,front_side_degrees);

  {draw the front panel dart, which is straight below the control line,}
  {but may be curved or straight above the control line.               }
  dart_level := waistlevel + (- front_panel_level);
  if front_panel_dart_is_straight then
  begin
    if debug2 then writeln(outfile,'Drawing straight front panel dart');
    draw_straight_dart(front_panel_dart_length,dart_level,
                       front_panel_translation, wl_front_panel2,
                       ytranslation,xscale,yscale, l_xnew,r_xnew,ynew)
  end
  else
  begin
    if debug2 then writeln(outfile,'Drawing curved front panel dart');
    draw_straight_dart(front_panel_dart_length,controllevel,
                       front_panel_translation, cl_front_panel,ytranslation
                       ,xscale,yscale, l_xnew,r_xnew,ynew);
    draw_curved_dart(l_xnew,r_xnew,ynew,guidehole_level,front_panel_level,
                     front_panel_translation,wl_front_panel2,
                     front_panel_dart_length,front_panel_degrees)
  end{else}
END {DRAW_DARTS};



procedure LABEL_PATTERN;
begin
```

```
  writeln;
  writeln('SKIRT':20,'SKIRT':49);
  writeln('CF':8,' ':7,'FRONT',' ':35,'CB',' ':7,'BACK');

end {LABEL_PATTERN};




begin {main}

  quit := 'n';
  enter := 'y';
  assign(outfile,outfilename);

  while (quit <> 'y') and (enter = 'y') do
  begin

    rewrite(outfile);
    error := false;       {Turn off the error flag}

    measure(differential,sideseam,waist_pattern_width,hip_pattern_width,
            dart_table,error);

    if (error = false) then
    begin

      points(skirt_length,waist_pattern_width,f_sstranslation,
             b_sstranslation,xtranslation,ytranslation,waistlevel,
             controllevel,hiplevel, hemlevel,cb,cf,xscale,yscale);

      if debug1 then
      begin
        writeln(outfile,'skirt length= ',skirt_length:5,
                '  waist_pattern_width=', waist_pattern_width:4);
        writeln(outfile,'front transl= ',f_sstranslation:4,
                ' back transl= ', b_sstranslation:4,'  y transl= ',
                ytranslation:4,'  xscale= ', xscale:6:1,'  yscale= ',
                yscale:6:1);
        writeln(outfile,'waistlevel= ',waistlevel:3,' controllevel= ',
                controllevel:3,' hiplevel= ',hiplevel:3,' hemlevel= '
                ,hemlevel:4);
        writeln(outfile,'cb= ',cb:4,'  cf=',cf:4);
      end{if};

      draw_guidelines(f_sstranslation,b_sstranslation,waistlevel,hemlevel,
                      cf,cb);

      draw_sideseams(sideseam,f_sstranslation,b_sstranslation,
                     ytranslation,hemlevel,xscale,yscale);

      draw_darts(dart_table,differential,f_sstranslation,b_sstranslation,
```

```
                xtranslation,waistlevel);

      {draw_waistline;}

      label_pattern;

      for i := 1 to 22 do writeln;
      write('Do you want to enter measurements again?  y or n : ');
      readln(enter); writeln;

    end{if}
    else
    begin
      writeln;
      writeln('Since there is an error in the in the input, you have ',
              'the option to correct it or quit.');
      write('Do you wish to quit? y or n: '); readln(quit); writeln
    end{else};

    close(outfile);
  end{while};
end.
```

AUTOMATION OF THE MODULAR PATTERN SYSTEM BASIC SKIRT PATTERN DRAFTING METHODOLOGY USING TURBO PASCAL AND DBASEII

by

MIRIAM SHAHEED CLARK

B.S., KANSAS STATE UNIVERSITY, 1982

ABSTRACT FOR A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1986

Abstract

This report details the automation of the Mod-u-lar Pattern System skirt pattern drafting methodology and the resulting automated drafting system. The Mod-u-lar Pattern System drafting methodology is still under development by Helen Brockman, professor emeritus at Kansas State University. The first stage of automation is for the basic skirt and has been implemented using dBaseII and Turbo Pascal. The current implementation was limited to drawing the front and back of the basic skirt pattern on a cathode ray screen.

The problem from the point of view of the pattern maker is clarified. Professor Brockman's entire drafting process is explained, beginning with how certain measurements are taken and transformed to the final pattern. An overview of the program design, a hierarchy diagram showing the calling structure, and a table of the inputs and outputs for each module are given. A user's manual is included along with the source code and all data tables required during the automated pattern drafting process.

This project includes the automation of but a small part of the Mod-u-lar Pattern System methodology. Because of the perceived need for properly fitting clothing by a large percentage of the United States population , the automation of professor Brockman's entire drafting methodology would be a great financial advantage to Kansas State University.