

COMPETITION DECISION - ASSIST PACKAGE FOR THE
MICROCOMPUTER(U) ARMY PROCUREMENT RESEARCH OFFICE FORT
LEE VA M G NORTON ET AL. FEB 87 APRO-85-86

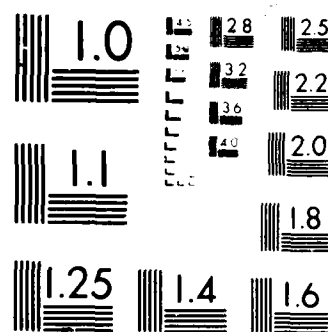
UNCLASSIFIED

F/G 5/1

NL

1

IND
4 9/
DHC



PHOTOCOPY RESOLUTION TEST CHART

APRO 85-06

FINAL

COMPETITION DECISION-ASSIST PACKAGE
FOR THE MICROCOMPUTER

by

MONTE G. NORTON

and

FRED A. FRYE

Information and data contained in this document are based on input available at the time of preparation. This document represents the view of the authors and should not be construed to represent the official position of the United States Army.

The pronouns "he", "his", and "him", when used in this publication, represent both the masculine and feminine genders unless otherwise specifically stated.

Approved for Public Release; Distribution Unlimited

OFFICE OF THE DEPUTY CHIEF OF STAFF FOR LOGISTICS
US ARMY PROCUREMENT RESEARCH OFFICE
Fort Lee, Virginia 23801-6045

EXECUTIVE SUMMARY

A. BACKGROUND. Now more than ever Project Managers must assess the extent to which their system ought to be competed. While current guidance, including the current AMC Pamphlet 715-9, the Competition Decision-Assist Package, is useful in analyzing the competition decision, a need exists for an improved automated process for providing decision support information to managers.

B. STUDY OBJECTIVE. The objective of this study was to develop, test, and effectively operate a transportable microcomputer system version of the Competition Decision-Assist Package (CDAP) in order to enhance its versatility in the field.

C. SUMMARY. The Competition Decision-Assist Package for the Microcomputer is a microcomputer FORTRAN version of CDAP that operates on an IBM Personal Computer and compatibles. It performs just as the CDAP does, but it is more easily transported to the field via floppy disk. *Keywords: flow charting*



Accession For	
NTIS CRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and/or Special
A-1	

TABLE OF CONTENTS

	<u>PAGE</u>
EXECUTIVE SUMMARY.....	i
SECTION:	
I. <u>INTRODUCTION</u>	
A. Background.....	1
B. Objective.....	2
C. Approach.....	2
II. <u>CDAP FOR THE MICROCOMPUTER</u>	
A. Competition Analysis.....	3
B. Competition Decision-Assist Package for the Microcomputer (CDAPM).....	4
III. <u>Using CDAPM</u>	
A. What CDAPM Needs.....	5
B. Protecting CDAPM and Data.....	5
C. Making CDAPM Go.....	6
<u>APPENDIX A</u>	
CDAPM User's Guide.....	A-1
<u>APPENDIX B</u>	
Flowchart for CDAPM.....	B-1
<u>APPENDIX C</u>	
CDAPM Program Listing.....	C-1
<u>APPENDIX D</u>	
CDAPM Technical Reference.....	D-1

SECTION I

INTRODUCTION

A. BACKGROUND

With the enactment of the Competition in Contracting Act (CICA) in 1984, competition is expected in the production of major weapon systems. Now more than ever Project Managers (PM) must assess the extent to which their system ought to be competed. In APRO 82-08, Competition Decision-Assist Package (CDAP), the Army Procurement Research Office (APRO) developed a decision-assist package consisting of a computer program and a guidance document to assist the PM's analytical staff in evaluating the likely cost effects and non-cost effects of competitive acquisition strategies. In APRO 84-09, CDAP-Enhanced, the Army Procurement Research Office expanded the competition analysis capability and improved the output presentations of the model.

While current guidance is helpful in analyzing the competition decision, a need exists for an improved automated process for providing decision support information to managers. The model should provide a general framework for analysis that can be readily modified to accommodate the acquisition peculiarities of a particular system under consideration. Further, most of the analytical information needs to be available to the manager/decisionmaker in a form which will readily enable him to assess significant total cost differences between alternatives. In order to better serve the needs of a manager/decisionmaker, a transportable and

automated CDAP tool must be available to the PM's analysts. Recent hardware advances and integrated state-of-the-art software systems provide an environment which lends itself to the transformation of CDAP from a mainframe computer mode to a microcomputer mode.

B. OBJECTIVE

The objective of this effort was to develop, test, and effectively operate a transportable microcomputer system version of the CDAP model in order to enhance its versatility in the field. It was the intent of this project to provide a timely solution to the problem of transportability of CDAP to the field.

C. APPROACH

The study approach to provide a Competition Decision-Assist Package for the Microcomputer (CDAPM) was as follows:

1. Obtain a "state-of-the-art" transportable microcomputer for use within APRO.
2. Download the mainframe version of CDAP and tailor the FORTRAN coding as required to ensure the precise transition of the CDAP model from the mainframe computer to the microcomputer.
3. Test the microcomputer version(s) for accuracy and structural integrity.
4. Develop guidance for the microcomputer version of CDAP.
5. Recommend pamphlet changes as appropriate to provide adequate guidance to users.

SECTION II

CDAP FOR THE MICROCOMPUTER

A. COMPETITION ANALYSIS

It is generally accepted that competition can reduce costs, improve quality and performance, and enhance the industrial base; and this has usually been the case. In order to maximize the benefits from competition, a thorough analysis should be made of each system to structure the appropriate competition strategy. This analysis typically focuses on both the qualitative and quantitative factors impacting the competition and further breaks the quantitative cost factors into non-recurring investment costs and recurring unit costs.

AMC Pam 715-9 describes these factors and offers the CDAP as a means to analyze primarily the recurring cost savings. The economic analysis model selected for the approved CDAP guidance is based upon production improvement (learning curve) theory that involves making a comparison on non-recurring cost (i.e., Government and contractor investment) and recurring cost savings. Although this approach requires experienced procurement judgement, it is reasonably quantitative and repetitive in nature.

To increase the acceptance and use in the field, some enhancements to the current CDAP were required. These include improved output presentation formats, expanded analysis capability, and increased "transportability" of the package to the field. It is the intent of this study to meet the transportability requirements. Planned improvements for CDAP in

the future include incorporating the non-recurring investment cost analysis directly into the CDAP. Currently it is done separately to supplement CDAP and provide a complete analysis.

B. COMPETITON DECISION-ASSIST PACKAGE FOR THE MICROCOMPUTER

The Competition Decision-Assist Package for the Microcomputer (CDAPM) was written in FORTRAN for the IBM PC and compatible computers. It is virtually the same program as the mainframe Competition Decision-Assist Package--Enhanced (CDAP-E, APRO report 84-09 by V. Gail Lankford). Some changes were made to the way CDAP-E handles output from the user and the way it displays its results. Other changes, invisible to the user, had to be made to adapt CDAP-E to the software environment of the PC.

SECTION III

USING CDAPM

A. WHAT CDAPM NEEDS

CDAPM absolutely requires the following items:

1. An IBM PC or compatible computer with 128K of user memory and one double-sided, double-density 5.25 inch diskette drive.
2. DOS for that PC: 2.0 or a higher version.
3. This document.
4. The diskette accompanying this document.

Recommended for CDAPM, but NOT required:

1. A printer.
2. An additional disk drive (hard disk is best).
3. A math coprocessor chip (Intel 8087 or similar).

B. PROTECTING CDAPM AND DATA

Users MUST NOT use the distribution diskette to run CDAPM.

The file CDAPM.EXE should be copied from the distribution onto a hard disk or a dedicated CDAPM execution diskette. The distribution should then be put in a safe place. If two backups are desired, the distribution diskette can be duplicated with the DOS commands DISKCOPY or COPY; CDAPM is not copy-protected.

CDAPM data files should be backed up before every CDAPM session.

On the distribution diskette is the original FORTRAN source file for CDAPM. All users are free to modify CDAPM in any fashion they desire, but APRO requests that all modified versions of CDAPM be documented to prevent confusion with APRO's version.

C. MAKING CDAPM GO

The computer should be up and running, and the DOS command prompt should be on the screen.

The default drive and directory should be set to the place where the data files will be stored.

If the file CDAPM.EXE is not there, the PATH must be set to include the place where CDAPM.EXE exists.

CDAPM is started by typing CDAPM and a carriage return.

Appendix A of this document contains step-by-step instructions on how to make CDAPM perform a given analysis.

APPENDIX A

CDAPM USER'S GUIDE

(This appendix is adapted from APRO Report 84-09, Competition Decision-Assist Package Enhanced, by V. Gail Lankford.)

TABLE OF CONTENTS - APPENDIX A

<u>SECTION</u>	<u>PAGE</u>
A. Model Overview.....	A-3
B. Assumptions/Constraints.....	A-5
C. Execution of Computer Program.....	A-7
1. Data File Create Mode.....	A-7
2. File Modify Mode.....	A-9
3. Simulate Mode.....	A-10
4. Sensitivity Analysis.....	A-11
D. Example Input Data Files.....	A-11
E. Program Execution Example.....	A-13
F. Example Program Output.....	A-19

APPENDIX A
CDAPM USERS' GUIDE

A. MODEL OVERVIEW.

This interactive computer program has been designed to calculate estimates of recurring costs associated with two producers involved in a competitive unit production effort. The concepts employed in this program are those described in Chapter III of APRO 82-08, Competition Decision-Assist Package. The model is based on learning curve or cost-improvement curve theory, which states that a relationship exists between production quantity and unit price such that, as the quantity produced doubles, the unit price will be a fixed percentage less than the unit price prior to the doubling. For example, if a 95% learning curve is being observed, the cost of the 200th item will be 95% of the cost of the 100th item. According to this theory, total recurring cost (c) can be computed as the sum of these declining unit costs, and represented by the following function:

$$c = \sum_{q=q_1}^{q_2} a q^b$$

where a represents first unit cost, b is the rate of cost improvement, and the production quantity extends from unit q_1 to q_2 .

Certain modifications to this fundamental expression are necessary to accomodate any shifts in unit price and/or rotation of the cost-improvement curve which are attributable to the introduction of competition into the process. A shift in unit price is represented by an equivalent shift in the first unit cost. A change in the cost-improvement curve is modeled by

adjusting the expression exponent, b , an appropriate amount. Also, since it is common for the analysis to consider production efforts spanning several delivery years, it may be desirable to include the time value of money in the analysis. This can be accomplished by multiplying each year's production cost by an appropriate discount factor. Thus, the basic relationship becomes for year i :

$$c_i = \alpha_i (1-p_i) \sum_{q=q_{i-1}+1}^{q_i} aq^{(b-\sigma_i)}$$

where α_i represents the discount factor for year i , q_i represents the production experience of a producer at the end of year i , p_i is the percent reduction in unit cost at year i due to a shift in unit price and σ_i is the relative change in the cost-improvement rate at year i due to rotation.

Any combination of adjustments can be applied for any production year, as appropriate. When no adjustments are made, the expression reverts to its original form.

The total program recurring cost can be computed by evaluating the given expression for each production year and summing the results. The total cost, C , for n years is represented by:

$$C = \sum_{i=1}^n \alpha_i (1-p_i) \sum_{q=q_{i-1}+1}^{q_i} aq^{(b-\sigma_i)}$$

Costs may be computed in constant or in discounted dollars at the option of the user.

The result of the effort to compute the total program recurring costs, based on the cost effects of competition, depends greatly on the accuracy of

the input data which are developed by the user and provided interactively to the model. If all the parameter values were known with certainty, the alternative strategy costs could be easily computed. However, when the parameter values are not known with certainty, as is often the case, a range of values must be judgementally assigned to each data element, with the range reflecting the general level of uncertainty. In this CDAPM model, each factor of the basic cost relationship is treated as a triangular distribution of values with minimum, most likely, and maximum magnitudes. The model uses Monte Carlo techniques, randomly sampling from these distributions for each simulation. By repeating the process many times, a range of probable costs is developed.

Based on the simulation, the program output provides both the mean and median estimates of the total program recurring costs, minimum and maximum probable costs, the range of those costs, and the fourth-spread values: the values which bound the middle fifty percent of the simulated results. For multiple production periods, the cost of each period is given for each producer along with an aggregate lot cost. The program will also determine which of the two producers is most likely to win a split buy award and will display the relative win percentage. Finally, options also allow a cumulative probability versus total program recurring cost and probability density plots to be displayed.

B. ASSUMPTIONS/CONSTRAINTS.

It is well recognized that program logic influences model results and can introduce biases into those results. Assumptions inherent in the model logic

should be known to the user so he can better assess the appropriateness of the model as a tool for his analysis. The following assumptions and constraints are embedded within the algorithms and code of CDAP and CDAPM:

1. The first producer whose data is input to the model is considered the "Prime".

2. The model is constrained to consider not more than two competitors in a simulation.

3. "All or nothing" lot awards go the "Prime" until there is a split award. Thereafter, "all or nothing" lots are awarded to the producer with the lowest cumulative average unit cost prior to that lot.

4. On split awards, the major portion of the lot goes to the producer with the lowest cumulative average unit cost prior to that lot being awarded. Exception: If the first lot in the simulation is a split award, the major portion of the lot is awarded to the "Prime."

5. If a lot is split into two equal parts for award, it is assumed that the two producers are fully competitive and would bid the same price. The model sets the "Second Source" lot cost equal to that computed for the "Prime" for this lot.

6. For lots which are split into two equal parts, the Prime split win percentage displayed in the output result will be 100%. This statistic is meaningless in this instance, as no computation is made in the model to determine which producer would have won the major portion of the lot if the lot had not been split equally. The user can determine the percentage of time each producer would win the major portion by splitting the lot unequally and running a sensitivity analysis.

7. When a range of values is input to the model for a data element, a triangular distribution is assumed and values are computed as needed using a Monte Carlo sampling technique.

8. Ten percent (10%) discount factors are used when the option to discount is selected.

9. The magnitude of the analysis is limited to twenty-five (25) production lots.

10. The simulation is limited to 5000 iterations.

C. EXECUTION OF COMPUTER PROGRAM.

The program has been divided into three basic operational modes: data file create, data file modification, and simulate. Each of these modes are described in the sections which follow and example data files are given. An example session for each mode is also provided.

1. Data File Create Mode.

This program mode allows data files to be created interactively by following prompts given at the user's terminal. A number of the input parameters require three values to be entered. If conditions do not warrant a range of values, the same value is entered for each of the three required entries. The following information is requested:

a. Desired file name.

When a proposed file name is entered, its existence is checked to determine if another file with the entered name already exists. If the name does not already exist, it is accepted for use with the file being created. If the name already exists in the user's directory, the program

will ask for another name. This is done to prevent over writing an existing file.

- b. Number of production lots to be evaluated. (Limited to 25)
- c. First unit cost for prime producer. (minimum, most likely, maximum)
- d. First unit cost for the second source. (minimum, most likely, maximum)
- e. Prime producer performance curve slope. (minimum, most likely, maximum) for example: .88, .92, .95
- f. Second source performance curve slope. (minimum, most likely, maximum)
- g. Individual lot data.

(1) Major split quantity/minor split quantity. If it is desired to establish a sole source producer baseline, all lot quantity is entered in the major split value. The output will then only reflect results for the prime producer under sole source conditions. Computation of lot costs for split buy awards are based on a major/minor quantity award to each producer. For a given cycle of the simulation when unequal values have been entered for the major and minor split quantities, the program awards the major split quantity to the producer having the lowest unit price prior to the given lot (adjusted for any shifts). This approach to dealing with split awards may not be appropriate in all cases. An alternate method is available within the program. By entering equal values for the major and minor split quantities, a "composite" performance approach is invoked. The total lot will be split evenly between the two

producers, and the model will assign equal lot costs for each producer. This simulates both producers behaving in exactly the same manner.

(2) Prime producer competition shift percentage (minimum, most likely, maximum)/second source competition shift percentage (minimum, most likely, maximum). Shifts are entered relative to a base of 1.0. If no shift is expected to take place, enter 1.0. If, due to competition, you expect the prime producer to lower his unit price by 8 percent, that would be entered as .92 ($1 - .08$). If the range is expected to be a 5 to 12 percent decrease, then the values entered would be .88, .92 and .95 (in each case modifying the base of 1.0 by the percent shift). The model can also handle an increase in unit price. A 2 percent increase would be entered as 1.02.

(3) Prime producer curve competition rotation (minimum, most likely, maximum)/second source curve competition rotation (minimum, most likely, maximum). Entry of performance curve rotation values is based on slope percentage points. If the basic curve slope had been entered as .93 and is expected to reach a value of .91 after competition, the rotation entry would be .02 to indicate a downward rotation. Upward rotations may be indicated by using negative input values. An example of inputs for a downward rotation is .01, .02, .03.

To maximize ease of data review, appropriate headings are stored along with the data entries. It is thus possible to quickly review the contents of individual files for accuracy without utilizing the available file modification mode, if desired.

2. File Modify Mode.

Changes to existing data files can easily be made using the modify

mode. When this mode is entered, the name of the data file to be modified is requested. All information in the file is then displayed along with a reference number. Entry of the appropriate reference number causes the data entry prompt for the affected element to be displayed.

While in this mode it is possible to extend the number of lots to be evaluated by simply changing the entry for number of lots. When that value is increased the program will automatically display the lot data prompts for the new lot or lots. It should also be noted that if one entry for a given lot is to be modified, all data entry prompts for that lot will be given. The data elements that remain unchanged are simply reentered.

3. Simulation Mode.

When the simulation mode is entered, the user is queried by prompts for the information necessary to execute the simulation. The name of the appropriate data file for the simulation is requested and verified for its existence. Once the proper data file has been established the user is asked for an arbitrary whole number between 1 and 999,999, and then whether a cumulative probability plot and a probability density plot are desired. The desired number of simulation cycles must be input (5000 is the maximum allowed). Choosing the same arbitrary whole number and the same number of cycles on two different runs of CDAPM will cause CDAPM to produce identical results on the two runs. Then the user selects whether the option to discount is to be implemented. If a negative response is given to this prompt, output results will be in constant base year dollars. A positive response will cause the program to request the lot number that discounting should begin. When discounting is selected the output will be annotated accordingly. Discounting in this program is based on yearly midpoints.

Basic program output consists of a presentation of the total average cost and other statistical indicators of central tendency and spread, along with individual lot average costs. Basic output also displays the individual input parameters derived from the specified file. Display of the probability plots was made optional to minimize printing time if that information is not actually needed for a given analysis.

4. Sensitivity Analysis.

The analyst is encouraged to check the sensitivity of the simulation output results to input parameters of concern. This is readily accomplished by modifying one parameter in the input file, rerunning the simulation, and comparing the new results with those of the initial run. For this comparison purpose the simulation should be run for the same number of iterations and same arbitrary whole number each time. Although Monte Carlo techniques are used in the simulation, this will cause no problem when comparing the results of the sensitivity analysis with the baseline, so long as both runs are performed on the same ADP system.

D. EXAMPLE INPUT DATA FILES.

1. Example 1.

This example sets up data for a sole source baseline. The production period covers four years. Note that all lot quantities are placed in the major split category, and that no performance curve shifts or rotations are entered.

```

PRIME FIRST UNIT COST---MIN> 80000. MOST LIKELY> 100000. MAX> 127000.
SECOND SOURCE-----MIN> 0. MOST LIKELY> 0. MAX> 0.
PRIME PCURVE SLOPE-----MIN> .910 MOST LIKELY> .930 MAX> .950
SECOND SOURCE-----MIN>1.000 MOST LIKELY>1.000 MAX>1.000

```

LOT #	LOT QUAN	SHIFT FACTOR PRIME			SHIFT FACTOR SECOND SOURCE			ROTATION FACT PRIME			ROTATION FACT SECOND SOURCE				
		MAX	MIN	M.L.	MAX	MIN	M.L.	MAX	MIN	M.L.	MAX	MIN	M.L.	MAX	
		1	1000.	0.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	.00	.00	.00	.00
2	1800.	0.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	.00	.00	.00	.00	.00	.00
3	2500.	0.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	.00	.00	.00	.00	.00	.00
4	3000.	0.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	.00	.00	.00	.00	.00	.00

Figure A-1. Example 1 Input Data File

2. Example 2.

This example sets up a file for a six year production program where a split buy competition is introduced in the first year with a buy-out in the fifth and sixth years. Note that shift factors are introduced for both producers in the second year, and both shift and rotation are introduced in the third year as the producers respond to competitive pressures.

```

PRIME FIRST UNIT COST---MIN> 100000. MOST LIKELY> 140000. MAX> 200000.
SECOND SOURCE-----MIN> 90000. MOST LIKELY> 140000. MAX> 210000.
PRIME PCURVE SLOPE-----MIN> .870 MOST LIKELY> .940 MAX> .980
SECOND SOURCE-----MIN> .850 MOST LIKELY> .940 MAX> .970

```

LOT #	LOT QUAN	SHIFT FACTOR PRIME			SHIFT FACTOR SECOND SOURCE			ROTATION FACT PRIME			ROTATION FACT SECOND SOURCE			
		MAX	MIN	M.L.	MAX	MIN	M.L.	MAX	MIN	M.L.	MAX	MIN	M.L.	MAX
		1	500	300	1.00	1.00	1.00	1.00	1.00	1.00	.00	.00	.00	.00
2	1500	500	.94	.96	.95	.93	.98	1.00	.00	.00	.00	.00	.00	.00
3	2500	1200	.96	.98	1.00	.94	.96	.98	.02	.04	.06	.01	.03	.04
4	4000	2000	1.00	1.00	1.00	.95	.97	1.00	.00	.00	.00	.02	.04	.05
5	4000	0	1.00	1.00	1.00	.96	.98	1.00	.00	.00	.00	.00	.00	.00
6	4000	0	1.00	1.00	1.00	1.00	1.00	1.00	.00	.00	.00	.00	.00	.00

Figure A-2. Example 2 Input Data File

E. PROGRAM EXECUTION EXAMPLES.

FILE CREATE MODE EXAMPLE SESSION

The following program modes are available:

1. Create a new data file (C)
2. Modify an existing data file (M)
3. Run using an existing data file (R)

Enter the letter shown in () that corresponds to the desired mode
C

What is the data file name?
Testfile

How many lots are there in this data set?
2

First unit cost for prime
Minimum: 100000

Most likely: 120000

Maximum: 170000

Second source first unit cost
Minimum: 90000

Most likely: 115000

Maximum: 170000

Prime performance curve slope (.XXX)
Minimum: .88

Most likely: .95

Maximum: .95

Second source performance curve slope (.XXX)
Minimum: .86

Most likely: .89

Maximum: .95

Data for lot # 1

Major split quantity 600

Minor split quantity 300

Prime shift factor (.XXX)

Minimum: .90

Most likely: .95

Maximum: .99

Second source shift factor (.XXX)

Minimum: .88

Most likely: .93

Maximum: .98

Prime rotation factor (.XXX)

Minimum: .01

Most likely: .02

Maximum: .03

Second source rotation factor (.XXX)

Minimum: .01

Most likely: .02

Maximum: .03

Data for lot # 2

Major split quantity 900

Minor split quantity 100

Prime shift factor (.XXX)

Minimum: 1.00

Most likely: 1.00

Maximum: 1.00

Second source shift factor (.XXX)

Minimum: 1.00

Most likely: 1.00

Maximum: 1.00

Prime rotation factor (.XXX)

Minimum: 0

Most likely: 0

Maximum: 0

Second source rotation factor (.XXX)

Minimum: 0

Most likely: 0

Maximum: 0

Another file (Y,N)?

N

Would you like to enter another mode (Y,N)

Y

FILE MODIFY MODE EXAMPLE SESSION

The following program modes are available:

1. Create a new data file (C)
2. Modify an existing data file (M)
3. Run using an existing data file (R)

Enter the letter shown in () that corresponds to the desired mode
M

What is the name of the file you wish to modify
Testfile

(1) NUMBER OF LOTS : 2
(2) PRIME FIRST UNIT COST-MIN> 100000 MOST LIKELY> 120000 MAX> 170000
(3) SECOND SOURCE-----MIN> 90000 MOST LIKELY> 115000 MAX> 170000
(4) PRIME PCURVE SLOPE----MIN> 880 MOST LIKELY> 950 MAX> 950
(5) SECOND SOURCE-----MIN> 860 MOST LIKELY> 890 MAX> 950
(6)

LOT	LOT QUAN	SHIFT FACTOR			SHIFT FACTOR			ROTATION FACT			ROTATION FACT				
		PRIME			SECOND SOURCE			PRIME			SECOND SOURCE				
#		MAX	MIN	MIN	M.L.	MAX	MIN	M.L.	MAX	MIN	M.L.	MAX	MIN	M.L.	MAX
1	600	300	90	95	99	88	93	98	01	02	03	01	02	03	
2	900	100	1.00	1.00	1.00	1.00	1.00	1.00	00	00	00	00	00	00	

Enter the number in the () that corresponds to the line you want to modify
4

Prime performance curve slope (.XXX)
Minimum: .88

Most likely: .90

Maximum: .95

Any more changes (Y,N) ? Y

Enter the number in the () that corresponds to the line you want to
modify
6

What lot number to you want to modify
2

Data for lot # 2
Major split quantity 950

Minor split quantity 400

Prime shift factor (.XXX)
Minimum: 1.00

Most likely: 1.00

Maximum: 1.00

Second source shift factor (.XXX)
Minimum: 1.00

Most Likely: 1.00

Maximum: 1.00

Prime rotation factor (.XXX)
Minimum: 0

Most likely: 0

Maximum: 0

Second source rotation factor (.XXX)
Minimum: 0

Most likely: 0

Maximum: 0

Any more changes (Y,N) ? N

Would you like to enter another mode (Y,N)
Y

SIMULATION MODE EXAMPLE SESSION

The following program modes are available:

1. Create a new data file (C)
2. Modify an existing data file (M)
3. Run using an existing data file (R)

Enter the letter shown in () that corresponds to the desired mode
R

What is the data file name?
STRATEGY3

Input random number seed (1-999999)?
123

Do you want a cumulative probability display (Y,N) ? Y

Do you want a probability density display (Y,N) ? Y

How many simulation cycles would you like (5000 is max) ? 1001

Do you want the results in discounted dollars (Y,N) N

The following output modes are available

(1) Output to the CRT

(2) Output to the line printer

Select desired output mode (1, 2)?
2

Computing results now.

Another run (Y,N) ? N

Would you like to enter another mode (Y,N) N

F. EXAMPLE PROGRAM OUTPUT

THE FOLLOWING PROGRAM MODES ARE AVAILABLE :

1. CREATE A NEW DATA FILE (C)
2. MODIFY AN EXISTING DATA FILE (M)
3. RUN USING AN EXISTING DATA FILE (R)

ENTER THE LETTER SHOWN IN () FOR DESIRED MODE?

R

WHAT IS THE DATA FILE NAME ?

RATT

INPUT RANDOM NUMBER SEED (1 - 9999999)?

123

DO YOU WANT A CUMULATIVE PROBABILITY DISPLAY (Y,N)?

Y

DO YOU WANT A PROBABILITY DENSITY DISPLAY (Y,N)?

Y

HOW MANY SIMULATION CYCLES WOULD YOU LIKE (5000 IS MAX) ?

1001

DO YOU WANT THE RESULTS IN DISCOUNTED DOLLARS (Y,N)?

Y

WHAT LOT WILL DISCOUNTING BEGIN ?

1

THE FOLLOWING OUTPUT MODES ARE AVAILABLE

(1) OUTPUT TO THE CRT

(2) OUTPUT TO THE LINE PRINTER

SELECT DESIRED OUTPUT MODE (1,2)?

1

COMPUTING RESULTS NOW!

ARMY PROCUREMENT RESEARCH OFFICE (APRO)

MICROCOMPUTER VERSION OF COMPETITION DECISION ASSIST PACKAGE

--- RESULTS ARE IN DISCOUNTED DOLLARS ---
(10%) , BEGINNING WITH LOT NUMBER 1

***** AVERAGE COST = 261220 *****
***** MEDIAN COST = 260746 *****
***** MAXIMUM COST = 401861 *****
***** MINIMUM COST = 141896 *****
***** RANGE = 259965 *****
***** FOURTH-SPREAD: 230975 *****
***** 290609 *****

Pause.

Press (enter) to continue.

--- RESULTS ARE IN DISCOUNTED DOLLARS ---
(10%) , BEGINNING WITH LOT NUMBER 1

PRIME SPLIT WIN PERCENTAGE		LOT QUANTITY	AVERAGE LOT COST	AVERAGE UNIT COSTS		
LOT #	%			PRIME	SECOND SOURCE	COMPOSITE
1	100.00	600	74537 *	128 *	117 *	124 *
2	36.36	800	64931 *	97 *	81 *	81 *
3	36.36	1200	79923 *	83 *	65 *	67 *
4	34.87	900	41829 *	50 *	45 *	46 *

TOTAL NUMBER OF UNITS = 3500

Pause.

Press (enter) to continue.

PRIME FIRST UNIT COST--MIN> 100 MOST LIKELY> 200 MAX> 300
SECOND SOURCE-----MIN> 100 MOST LIKELY> 220 MAX> 250
PRIME PCURVE SLOPE----MIN> 900 MOST LIKELY> 950 MAX> 980
SECOND SOURCE-----MIN> 900 MOST LIKELY> 930 MAX> 960
NUMBER OF CYCLES---> 1001

LOT #	LOT QUAN MAX MIN	SHIFT FACTOR PRIME			SHIFT FACTOR SECOND SOURCE			ROTATION FACT PRIME			ROTATION FACT SECOND SOURCE		
		MIN	M.L.	MAX	MIN	M.L.	MAX	MIN	M.L.	MAX	MIN	M.L.	MAX
1	400. 200.	1.00	1.00	1.00	1.00	1.00	1.00	.00	.00	.00	.00	.00	.00
2	600. 200.	.90	.94	.98	.90	.92	.95	.01	.02	.03	.01	.02	.03
3	900. 300.	1.00	1.00	1.00	1.00	1.00	1.00	.01	.01	.01	.01	.01	.01
4	900. 0.	.95	.96	.98	.90	.92	.95	.00	.00	.00	.00	.00	.00

Pause.

Press <enter> to continue.

RUN DATE (MO/DY/YR) --->		11/22/1985				PAGE 2							
STRATEGY COST		CUMULATIVE PROBABILITY											
		0.00	.10	.20	.30	.40	.50	.60	.70	.80	.90	1.00	
		+-----+-----+-----+-----+-----+-----+-----+-----+-----+											
141896.	:												
144496.	:												
149695.	:												
154894.	:												
160093.	:												
165293.	:*												
170492.	:*												
175691.	:*												
180891.	**												
186090.	**												
191289.	***												
196489.	****												
201688.	*****												
206887.	*****												
212086.	*****												
217286.	*****												
222485.	*****												
227684.	*****												
232884.	*****												
238083.	*****												
243282.	*****												
248481.	*****												
253681.	*****												
258880.	*****												
264079.	*****												
269279.	*****												
274478.	*****												
279677.	*****												
284876.	*****												

290076	*****	7632
295275	*****	8012
300474	*****	8292
305674	*****	8521
310873	*****	8761
316072	*****	9061
321271	*****	9201
326471	*****	9321
331670	*****	9421
336869	*****	9560
342069	*****	9620
347268	*****	9660
352467	*****	9750
357666	*****	9810
362866	*****	9840
368065	*****	9880
373264	*****	9920
378464	*****	9940
383663	*****	9970
388862	*****	9980
394061	*****	9990
399261	*****	1 0000

+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
 0 00 10 20 30 40 50 60 70 80 90 1 00

STRATEGY COST

CUMULATIVE PROBABILITY

Pause.

Press (enter) to continue.

 RUN DATE (MO/DY/YR) ---> 11/22/1985

PAGE 3

STRATEGY COST

PROBABILITY

0 00 01 02 03 04 05 06 07 08 09 10

141896	:	0000
144496	:*	0030
149695	:	0010
154894	:*	0020
160093	**	0040
165293	**	0040
170492	****	0090
175691	***	0070
180891	*****	0130
186090	***	0070
191289	****	0090
196489	*****	0220
201688	*****	0250
206887	*****	0230
212086	*****	0260

WOULD YOU LIKE TO ENTER ANOTHER MODE (Y,N)?

N

CDAP FINISHED -- GOOD DAY!

Stop - Program terminated.

APPENDIX B

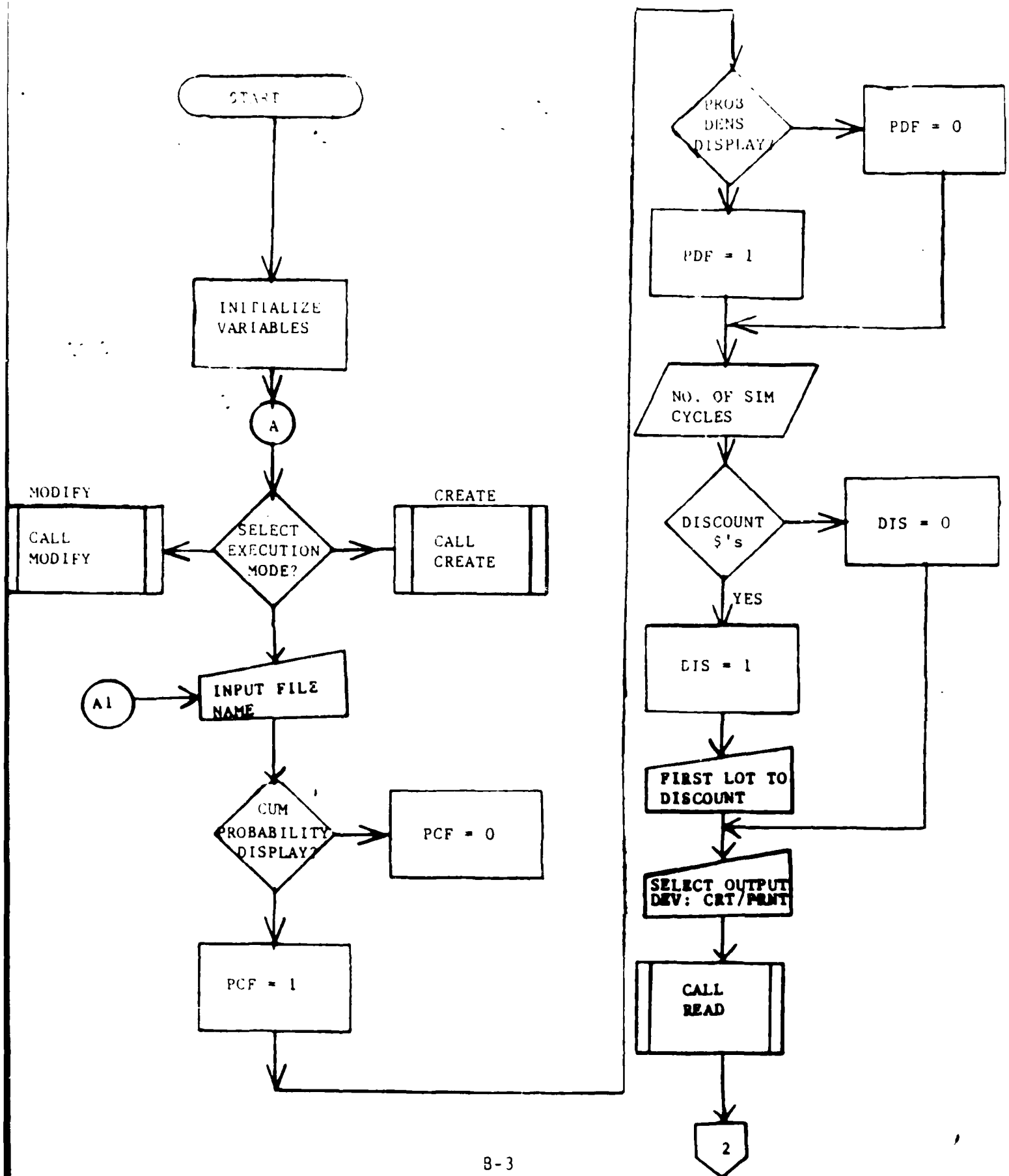
CDAPM

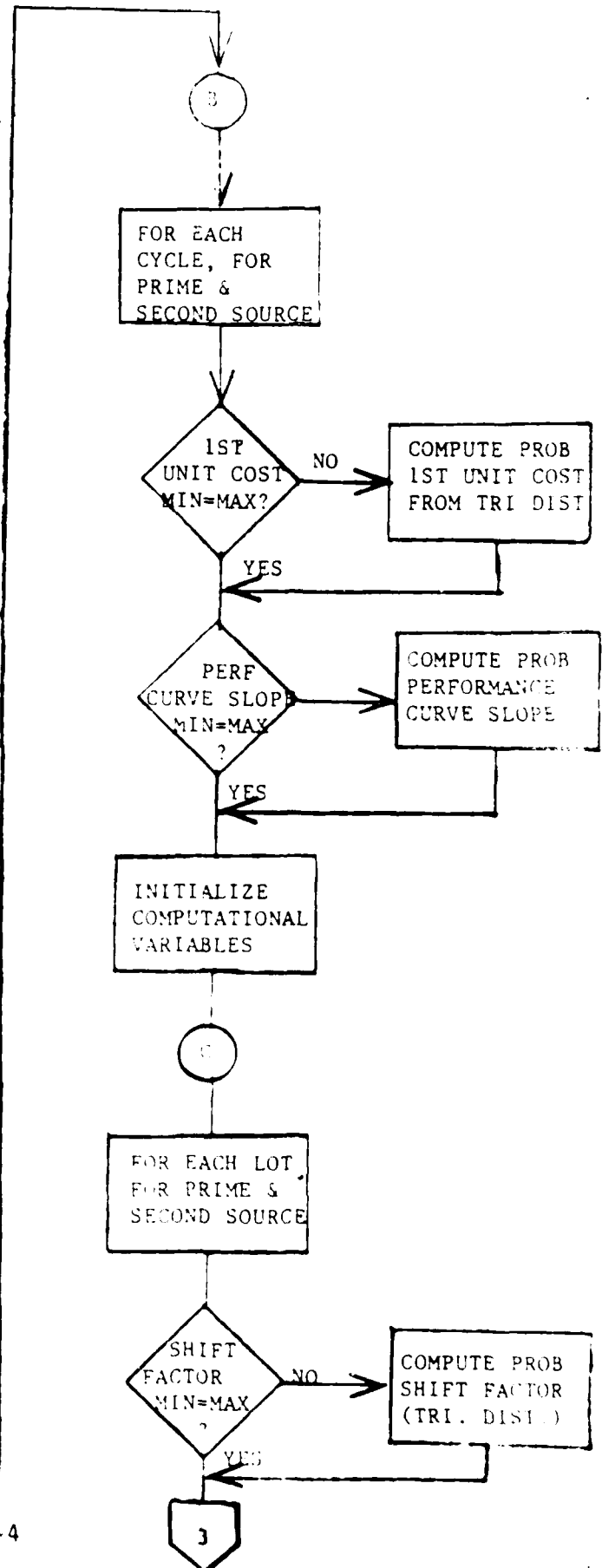
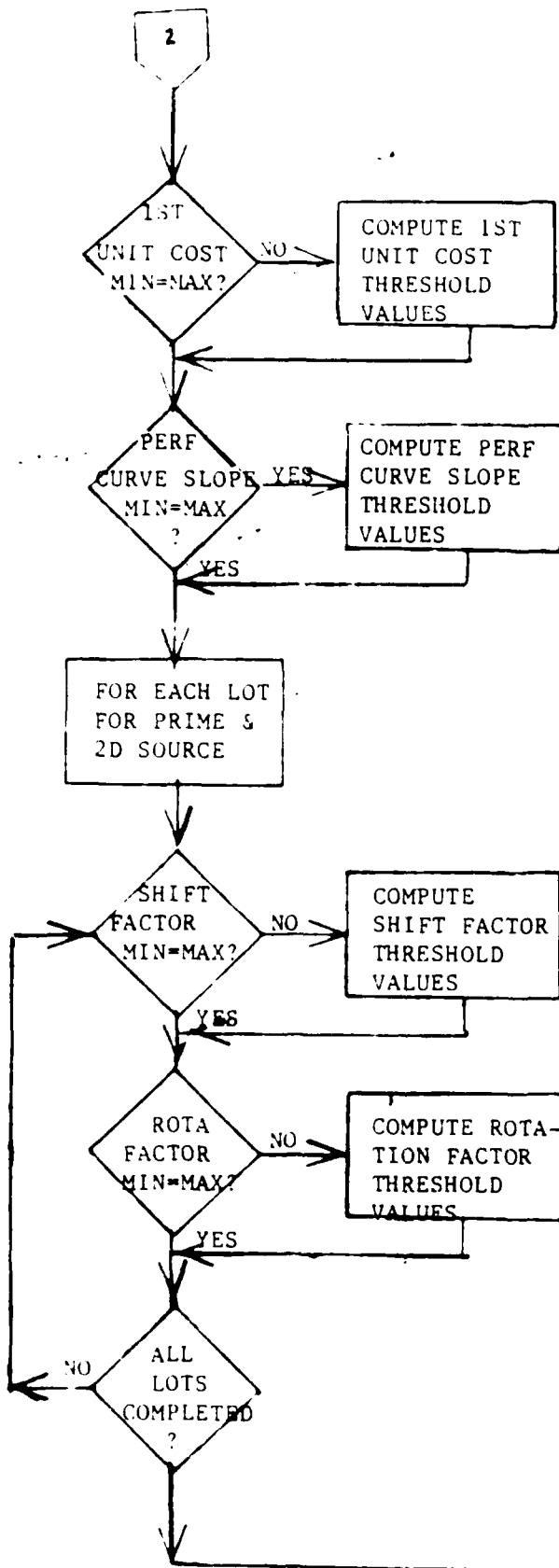
SYSTEM FLOWCHART

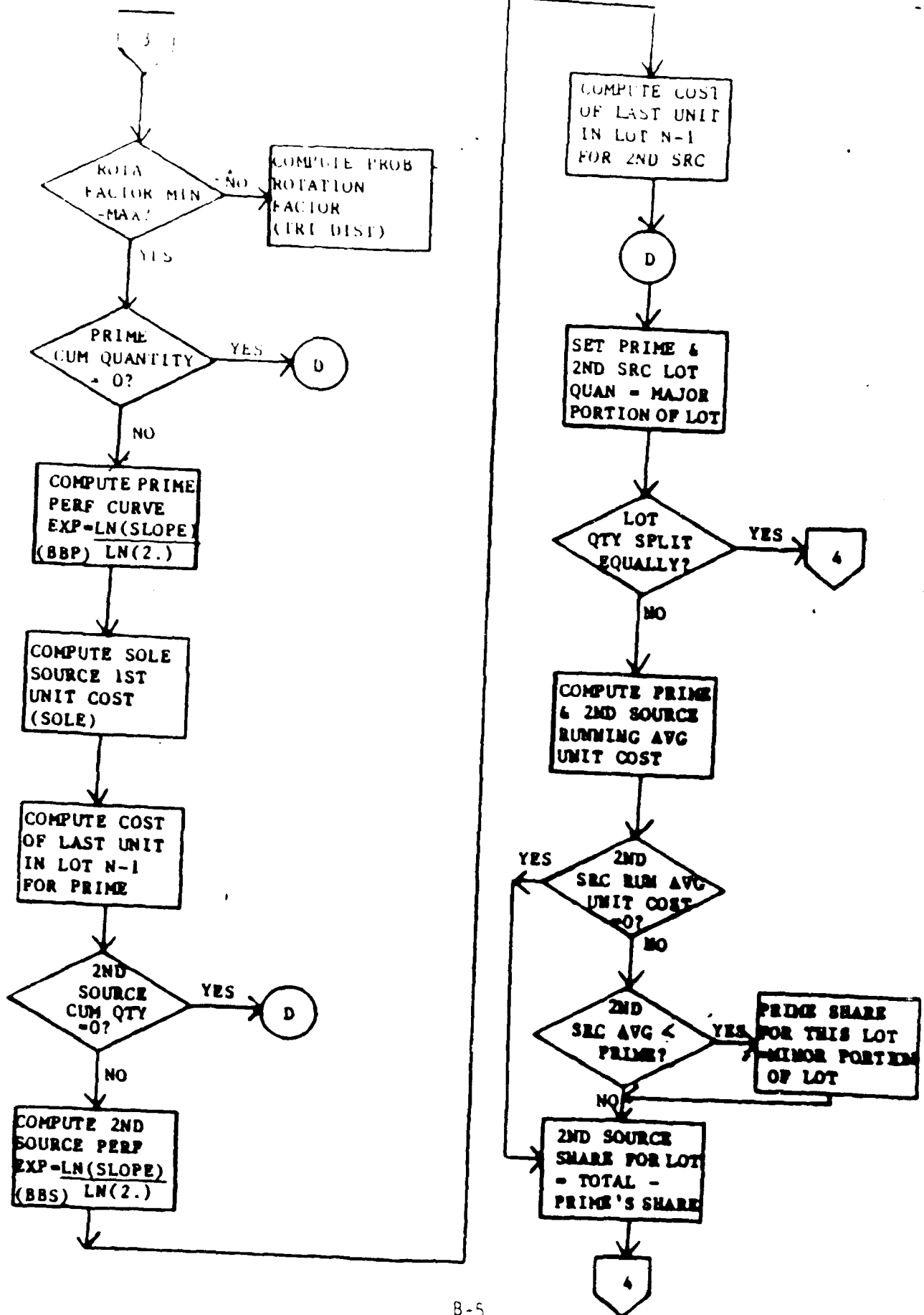
The CDAPM computer program consists of a main routine containing all the simulation logic and 12 subroutines which are called from the main program. The subroutines are used for housekeeping and formatting chores, are straightforward, do not affect the program logic and are therefore not flowcharted. They are listed alphabetically for reference in Figure B-1.

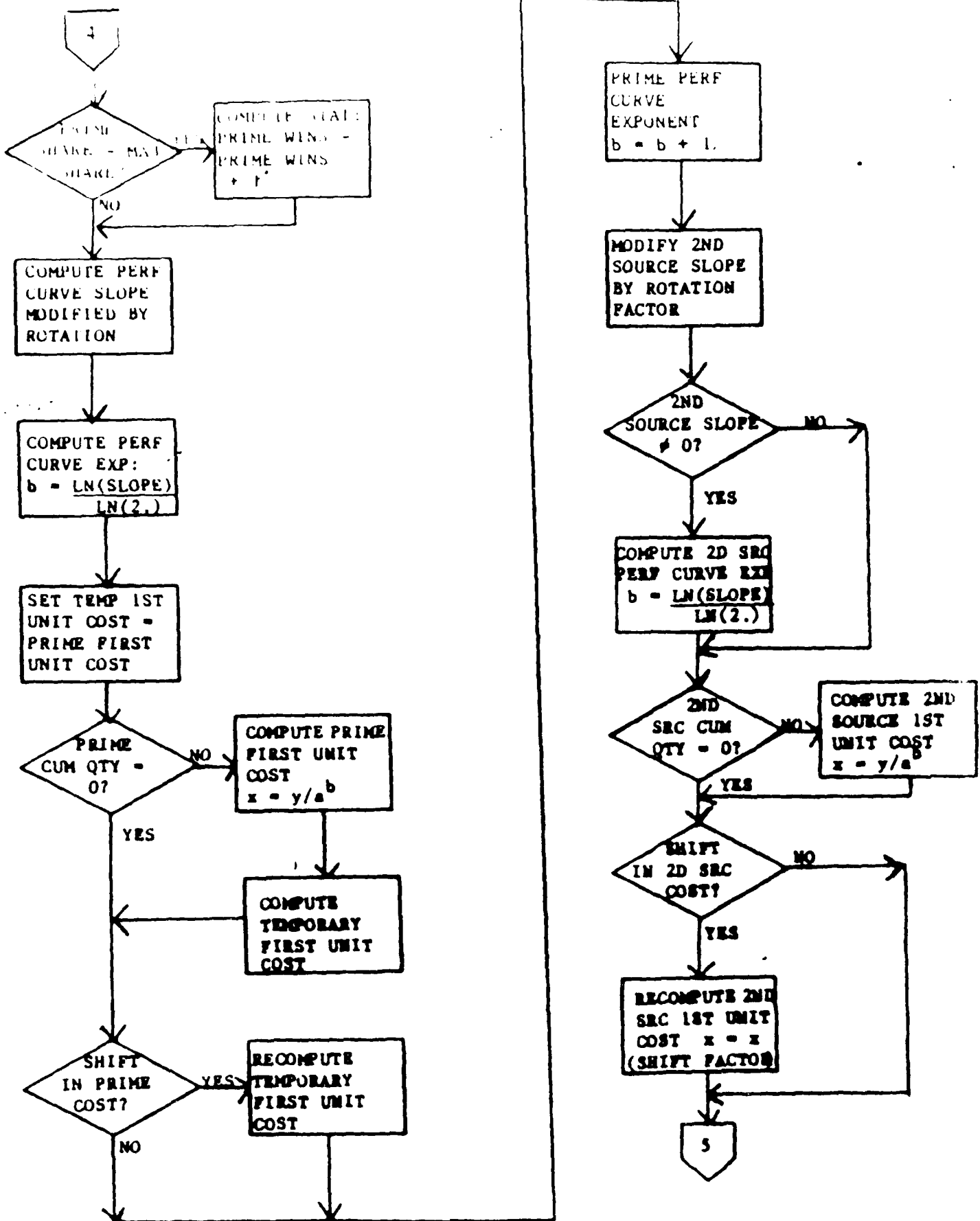
CREATE	PAGE
ENDP	PLOTA
HEAD1	RANDOM
HEAD2	RREAD
LINE	RWRITE
MODIFY	SCALE

FIGURE B-1. CDAPM Subroutines

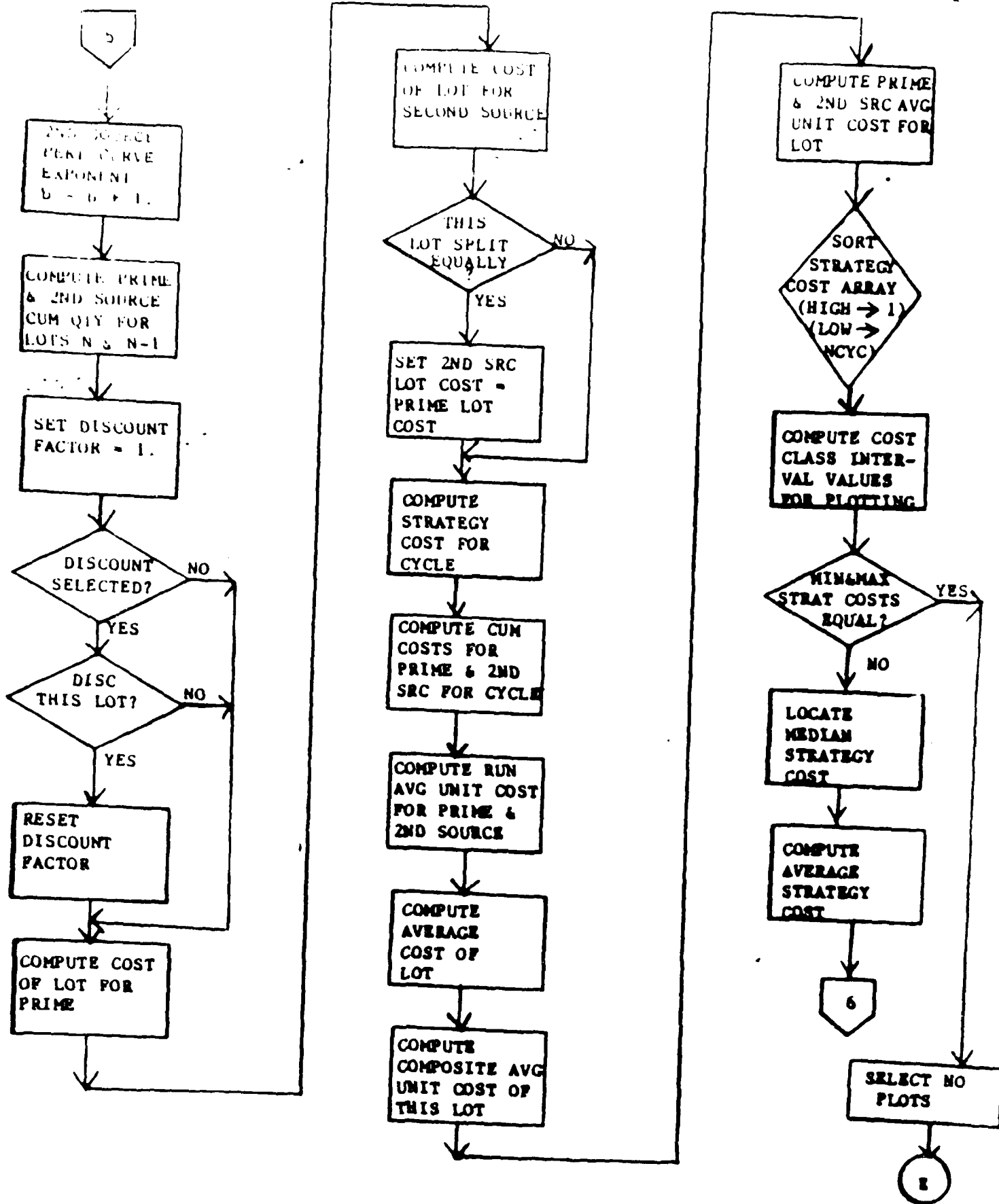


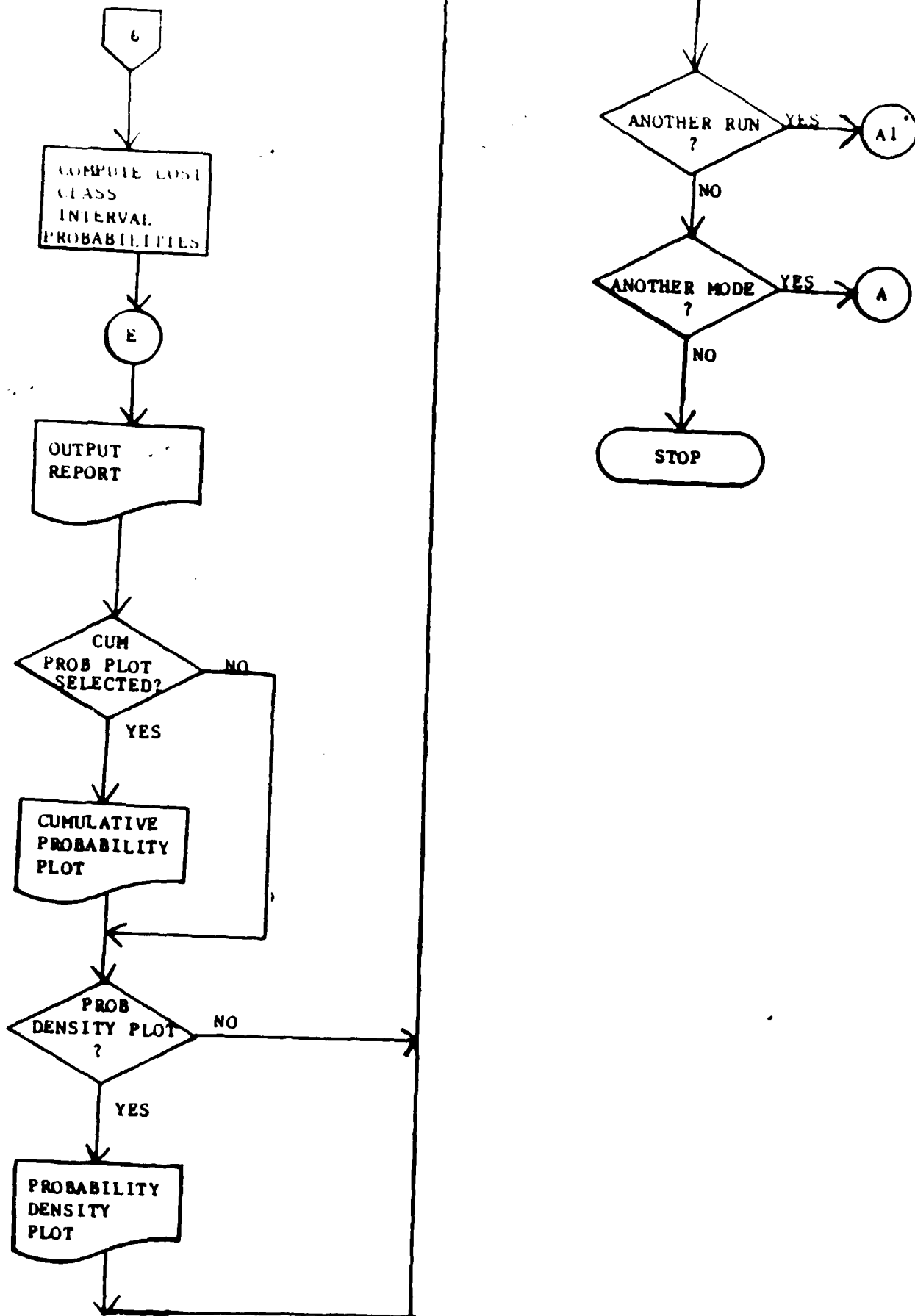






CDAPM Main Program (Con't)





APPENDIX C

CDAPM

SOURCE CODE LISTING

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
1 *****
2 C*
3 C* VARIABLES DEFINITION
4 C*
5 C*****
6 C ACLOT(25)    AVERAGE COST OF LOT N
7 C ALUC(25)    AVERAGE UNIT COST FOR LOT N (COMPOSITE)
8 C AMID        MEDIAN STRATEGY COST FROM SIMULATION RESULTS
9 C AP          PRIME SHIFTED FIRST UNIT COST AT LOT N AND A GIVEN SIM CYC
10 C APUCL(25)   AVERAGE UNIT COST FOR THE PRIME IN LOT N
11 C ASUCL(25)   AVERAGE UNIT COST FOR THE SECOND SOURCE IN LOT N
12 C AS         SS SHIFTED FIRST UNIT COST FOR LOT N AND A GIVEN SIM CYC
13 C AVG        AVERAGE STRATEGY COST
14 C BBP        ROTATED PRIME SLOPE AT LOT N AND A GIVEN SIM CYCLE
15 C BBS        ROTATED SECOND SOURCE SLOPE AT LOT N AND A GIVEN SIM CYC
16 C BP(3)      PRIME PERFORMANCE CURVE SLOPE VALUES--MIN, ML, MAX
17 C BPT        PRIME SLOPE TRIANGULAR DISTRIBUTION THRESHOLD VALUE
18 C BSS(3)     SECOND SOURCE PERFORMANCE CURVE SLOPE VALUES(MIN,ML,MAX)
19 C BSST       SECOND SOURCE SLOPE TRIANGULAR DIST THRESHOLD VALUE
20 C CIP(3)     FIRST UNIT COST FOR THE PRIME--MIN, ML, MAX
21 C CIPT       PRIME FIRST UNIT COST TRIANGULAR DIST THRESHOLD VALUE
22 C CISS(3)    FIRST UNIT COST FOR THE SECOND SOURCE--MIN, ML, MAX
23 C CISST      SECOND SOURCE FIRST UNIT COST TRIANG DIST THRESHOLD VALUE
24 C COST       VARIABLE USED TO PRINT THE COST VALUES USED IN OUTPUT PLOT
25 C CUM        SELECTS PROPER MODE FOR CUM PROB DISPLAY(1=CUM, 0=PDF)
26 C CWP(25)    PRIME WIN PERCENTAGE FOR LOT N
27 C CYCLES     FLOATING POINT VERSION OF NCYC (SIMULATION CYCLES)
28 C DEV        OUTPUT DEVICE ID (0 = CONSOLE, 1 = LPT1)
29 C DF         DISCOUNT FACTOR USED IN CALCULATING LOT COSTS
30 C DIS        SELECTION PARAMETER FOR IMPLEMENTING DISCOUNTING
31 C DISC(25)   INDIVIDUAL DISCOUNT FACTORS USING A 10% RATE
32 C DP         VALUE USED IN CALCULATING AVG UNIT COST OF PRIME AT LOT
33 C DS         VALUE USED IN CALCULATING AVG UNIT COST OF SS AT LOT N
34 C FINC       CLASS INTERVAL VALUE FOR SETTING UP THE OUTPUT PLOTS
35 C FNAME      VARIABLE USED TO READ THE DATA FILE NAME
36 C HOLD       TEMPORARY VARIABLE USED IN SORTING THE SIMULATION VALUES
37 C ICK        TEMPORARY VARIABLE USED DURING VALUE SORTING AND TESTS
38 C IDF        LOT NUMBER THAT DISCOUNTING WILL BEGIN
39 C ITT        TOP OF FORM PRINT VARIABLE FOR IMPACT TERMINALS
40 C IVAL       INTERGER VARIABLE USED TO SCALE THE OUTPUT PLOT LINES
41 C LINE       DATA FILE MODIFICATION ENTRY POINT (*MODIFY)
42 C LOTN       DATA FILE MODIFICATION LOT ENTRY POINT (*MODIFY)
43 C M1         ARGUMENT OF SUBROUTINE *CREATE -- ENTRY POINT FROM MODIFY
44 C M2         ARGUMENT OF SUBROUTINE *CREATE -- BEGINNING LOT NUMBER
45 C M3         ARGUMENT OF SUBROUTINE *CREATE -- ENDING LOT NUMBER
46 C NCYC       NUMBER OF SIMULATION CYCLES ---- INPUT PRIOR TO RUN
47 C NLN        LENGTH OF DATA FILE NAME
48 C NLOT       NUMBER OF LOTS READ FROM THE DATA FILE
49 C NLOTM      MODIFIED NUMBER OF LOTS FROM *MODIFY
50 C NLOTP      CURRENT NUMBER OF LOTS+1 -- USED IN *MODIFY
51 C PAVG       PRIME RUNNING AVERAGE UNIT COST AT LOT N AND A GIVEN SIM
52 C PB         PRIME PERFORMANCE CURVE SLOPE FOR CYCLE N OF THE SIM
53 C PC(50)     CALCULATED CLASS INTERVAL PROBABILITY
54 C PCOST(50)  CALCULATED COST CLASS INTERVAL PROBABILITY
55 C PCF        CUMULATIVE PROBABILITY DENSITY DISPLAY OPTION(1=YES,0=NO)
56 C PDF        PROBABILITY DENSITY DISPLAY OPTION (0=NO, 1=YES)
57 C PFUC       PRIME FIRST UNIT COST FOR CYCLE N OF THE SIMULATION
58 C PPCOST     PRIME RUNNING CUMULATIVE COST FOR A GIVEN SIMULATION CYC
59 C PROT       PRIME ROTATION FACTOR FOR LOT N AND A GIVEN SIM CYC

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
60 C PSF      PRIME SHIFT FACTOR FOR LOT N AND A GIVEN SIM CYCLE
61 C PUCOST    PRIME UNIT COST FOR LAST ITEM IN LOT N-1 FOR A SIM CYCLE
62 C PV(50)    HISTOGRAM PLOT VAR (DEFINES PHYSICAL NUM OF PLOT POINTS)
63 C Q1P      LAST UNIT NUMBER FOR PRIME AT LOT N-1 AND A GIVEN SIM CYC
64 C Q1S      LAST UNIT NUMBER FOR THE S S AT LOT N-1 AND SIM CYCLE
65 C Q2P      LAST UNIT NUMBER FOR PRIME A LOT N AND A GIVEN SIM CYCLE
66 C Q2S      LAST UNIT NUM FOR SECOND SOURCE FOR LOT N AND GIVEN SIM CY
67 C QM(25)    MAJOR SPLIT QUANTITY FOR LOT N
68 C QP      PRIME LOT QUANTITY FOR LOT N FOR A GIVEN SIMULATION CYCLE
69 C QPC      PRIME CUMULATIVE QUANTITY FOR CYCLE N OF THE SIMULATION
70 C QS(25)    MINOR SPLIT QUANTITY FOR LOT N
71 C QSS      SECOND SOURCE QUANTITY FOR LOT N FOR A GIVEN SIM CYCLE
72 C QSSC     SECOND SOURCE CUM QUANTITY FOR CYCLE N OF THE SIMULATION
73 C RN      RANDOM NUMBER USED IN THE SIMULATION
74 C ROTP(25,3) PRIME SLOPE ROTATION FACTOR FOR LOT N --- MIN, ML, MAX
75 C ROTPT    PRIME SLOPE RCTATION TRIANGULAR DIST THRESHOLD VALUE
76 C ROTSS(25,3) SECOND SOURCE SLOPE ROTATION FACTOR FOR LOT N(MIN,ML,MAX)
77 C ROTSST   SECOND SOURCE ROTATION TRIANGULAR DIST THRESHOLD VALUE
78 C SAVG     SECOND SOURCE RUNNING AVERAGE UNIT COST AT LOT N AND CYC
79 C SCOST(5000) STRATEGY COST FOR THE NTH SIMULATION CYCLE
80 C SF      SCALE FACTOR USED IN SCALE SUBROUTINE
81 C SFP(25,3) PRIME FIRST UNIT COST SHIFT FACTOR FOR LOT N(MIN,ML,MAX)
82 C SFPT     PRIME SHIFT FACTOR TRIANGULAR DISTRIBUTION THRES VALUE
83 C SFSS(25,3) SECOND SOURCE FIRST UNIT COST SHIFT FOR LOT N(MIN,ML,MAX)
84 C SFSST    SECOND SOURCE SHIFT FACTOR TRIANGULAR DIST THRES VALUE
85 C SOLE     SOLE SOURCE FIRST UNIT COST-- USED FOR REFERENCING SHIFT
86 C SSB      SECOND SOURCE PERFORMANCE CURVE SLOPE FOR CYCLE N OF THE
87 C SSCOST   SECOND SOURCE RUNNING CUMULATIVE COST FOR A GIVEN SIM CY
88 C SSFUC    SECOND SOURCE FIRST UNIT COST FOR CYCLE N OF THE SIM
89 C SSROT    SECOND SOURCE ROTATION FACT FOR LOT N AND A GIVEN SIM CY
90 C SSSF     SECOND SOURCE SHIFT FACTOR FOR LOT N AND A GIVEN SIM CYC
91 C SUCOST   SECOND SOURCE UNIT COST FOR LAST UNIT OF LOT N-1 AND CYC
92 C TCK      VARIABLE USED TO INDICATE TYPE OF TERMINAL BEING USED
93 C TFUC     TEMP FIRST UNIT COST USED BETWEEN LOTS SPECIFYING SHIFTS
94 C TOTAL    VALUE USED TO PRINT TOTAL LOT QUANTITIES
95 C TTYP     VARIABLE USED TO INPUT TERMINAL TYPE (IMPACT OR THERMAL)
96 C VAL      VARIABLE USED TO PRINT PROBABILITY VALUES IN OUTPUT PLOTS
97 C YP      COST OF LOT N FOR THE PRIME FOR A GIVEN SIMULATION CYCLE
98 C YS      COST OF LOT N FOR THE SS FOR A GIVEN SIMULATION CYCLE
99 C
100 C *****
101 C
102 C *****
103 C* THIS PROGRAM SETS UP A SIMULATION OF TWO PRODUCERS OPERATING IN A
104 C* COMPETITIVE ENVIRONMENT THE PURPOSE OF THE SIMULATION IS TO
105 C* CALCULATE THE COST OF ACQUIRING THE TOTAL QUANTITY OF ITEMS MADE BY
106 C* BOTH SOURCES CONSIDERED IN THIS SIMULATION PROGRAM ARE THE EFFECTS
107 C* OF QUANTITY PRODUCED ON UNIT PRICE (PRICE PERF CURVES, OR LEARNING
108 C* CURVES), INFLUENCE OF COMPETITION ON UNIT PRICE UNDER CONDITIONS OF
109 C* MULTIPLE CONTRACT AWARDS AND THE EFFECT OF COMPETITION ON RELATIVE
110 C* PRODUCTION EFFICIENCY DATA FOR THIS PROGRAM IS READ FROM A FILE
111 C* THAT IS PREPARED AND MAINTAINED BY A UTILITY WITHIN THIS PROGRAM
112 C* THIS PROGRAM WAS ORIGINALLY CODED FOR USE ON A PRIME COMPUTER BY
113 C*
114 C* US ARMY TANK AUTOMOTIVE COMMAND
115 C* SYSTEMS AND COST ANALYSIS DIRECTORATE
116 C* DRSTA-VS (MR PAUL BRADLEY)
117 C* WARREN MICHIGAN 48090
118 C*

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
119 C*****
120 C*
121 C* THIS VERSION OF THE PROGRAM HAS BEEN CODED SPECIFICALLY FOR THE
122 C* COMPAQ PLUS MICROCOMPUTER AND OTHER IBM COMPATIBLE MICROCOMPUTER SYS
123 C* USING IBM FORTRAN COMPILER VERSION 2 00 (MICROSOFT FORTRAN)
124 C* IMPROVEMENTS, AND CORRECTIONS TO THE PROGRAM LOGIC HAVE BEEN MADE, AS
125 C* NEEDED. A REPORT DOCUMENTING THE OPERATION AND FEATURES OF THIS
126 C* MODIFIED VERSION IS APRO 85-06 AND IS AVAILABLE FROM:
127 C*
128 C*      ARMY PROCUREMENT RESEARCH OFFICE
129 C*      ATTN: DALO-PRO (MR ANDERSON J LATTIMORE)
130 C*      FORT LEE, VA 23801-6045
131 C*
132 C*****
133      COMMON/A/PV(51),PCOST(50),SCOST(5000)
134      COMMON/RD/FID(12),CIP(3),CISS(3),BP(3),BSS(3),QM(25),
135      +QS(25),SFP(25,3),SFSS(25,3),ROTP(25,3),ROTSS(25,3),ACLOT(25),
136      +ALUC(25),APUCL(25),ASUCL(25),FNAME,NLOT,IYEAR,IMONTH,IDAY
137      REAL*8 PV,PCOST,SCOST,FID,CIP,CISS,ACLOT,ALUC,BP,BSS,SFP,SFSS,
138      +ROTP,ROTSS,SFPT(25),SFSST(25),ROTP(25),ROTSST(25),QM,QS,CWP(25)
139      +,AVG,COST,PPCOST,SSCOST,PAVG,SAVG,YP,YS,SSFUC,PFUC,ASUCL,APUCL
140      REAL DISC(25)
141      CHARACTER CH,ANS,TICK
142      CHARACTER*12 FNAME
143      INTEGER TCK,DEV
144      INTEGER*2 IYEAR,IMONTH,IDAY
145      INTEGER*4 TOTAL
146      DOUBLE PRECISION IX,RANDOM
147      LOGICAL OPN,SHOW
148      DATA DISC/ 954, 867, 788, 717, 652, 592, 538, 489, 445, 405,
149      + 368, 334, 304, 276, 251, 228, 208, 189, 172, 156, 142, 129,
150      + 117, 107, 097/,M1/0/,M2/0/,M3/0/
151      ITT=12
152      CH=CHAR(27)
153      DEV = 0
154      OPEN(1,FILE='LPT1')
155      CALL GETDAT(IYEAR,IMONTH,IDAY)
156 C*****
157 C*
158 C* DISPLAY MENU OF RUN MODES CREATE A NEW FILE, MODIFY EXISTING FILE
159 C* AND RUN THE SIMULATION FROM AN EXISTING FILE
160 C*
161 C*****
162      510 WRITE(*,520)
163      WRITE(*,522)
164      WRITE(*,524)
165      WRITE(*,526)
166      WRITE(*,540)
167      520 FORMAT(/IX,'THE FOLLOWING PROGRAM MODES ARE AVAILABLE  ')
168      522 FORMAT(/5X,'1 CREATE A NEW DATA FILE (C)')
169      524 FORMAT(/5X,'2 MODIFY AN EXISTING DATA FILE (M)')
170      526 FORMAT(/5X,'3 RUN USING AN EXISTING DATA FILE (R)')
171      540 FORMAT(/IX,'ENTER THE LETTER SHOWN IN ( ) FOR DESIRED MODE?')
172      READ(*,1250) ANS
173      IF( (ANS EQ 'C') OR (ANS EQ 'c') ) CALL CREATE(M1,M2,M3)
174      IF( (ANS EQ 'M') OR (ANS EQ 'm') ) CALL MODIFY
175      IF( (ANS EQ 'R') OR (ANS EQ 'r') ) GOTO 570
176      550 WRITE(*,560)
177      560 FORMAT(/IX,'WOULD YOU LIKE TO ENTER ANOTHER MODE (Y,N)? ')

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
178      READ(*,1250) ANS
179      IF ( ( ANS NE 'Y' ) AND ( ANS NE 'y' ) ) GOTO 1255
180      GOTO 510
181      570 WRITE(*,580)
182      580 FORMAT(/1X,'WHAT IS THE DATA FILE NAME ?')
183      READ(*,590) FNAME
184      590 FORMAT(A12)
185      C*****
186      C*
187      C* SET UP RUN PARAMETERS
188      C*
189      C*****
190      C
191      600      WRITE(*,610)
192      610      FORMAT(/1X,'INPUT RANDOM NUMBER SEED (1 - 999999)? ')
193      READ(*,625) IY
194      IF ( ( IY LT 1 ) OR ( IY GT 999999 ) ) goto 600
195      C***** a future version of CDAPM will use date/time as seed
196      IX=IY
197      DO 615 I=1,100
198      RN=RANDOM(IX)
199      615 CONTINUE
200      PDF=0
201      PCF=0
202      WRITE(*,620)
203      620 FORMAT(/1X,'DO YOU WANT A CUMULATIVE PROBABILITY DISPLAY (Y,N)? ')
204      READ(*,1250) ANS
205      IF ( ( ANS EQ 'Y' ) OR ( ANS EQ 'y' ) ) PCF=1
206      WRITE(*,630)
207      630 FORMAT(/1X,'DO YOU WANT A PROBABILITY DENSITY DISPLAY (Y,N)? ')
208      READ(*,1250) ANS
209      IF ( ( ANS EQ 'Y' ) OR ( ANS EQ 'y' ) ) PDF=1
210      640 WRITE(*,650)
211      650 FORMAT(/1X,'HOW MANY SIMULATION CYCLES WOULD YOU LIKE ')
212      1 '(5000 IS MAX) ?')
213      READ(*,625) NCYC
214      625 FORMAT(I10)
215      IF ( ( NCYC GT 5000 ) OR ( NCYC LT 1 ) ) GOTO 640
216      CYCLES=NCYC
217      DIS=0
218      WRITE(*,660)
219      660 FORMAT(/1X,'DO YOU WANT THE RESULTS IN DISCOUNTED DOLLARS (Y,N)? ')
220      READ(*,1250) ANS
221      IF ( ( ANS EQ 'Y' ) OR ( ANS EQ 'y' ) ) DIS=1
222      IF(DIS EQ 0 ) GOTO 673
223      WRITE(*,670)
224      670 FORMAT(/1X,'WHAT LOT WILL DISCOUNTING BEGIN ?')
225      READ(*,625) IDF
226      673 WRITE(*,674)
227      674 FORMAT(/1X,'THE FOLLOWING OUTPUT MODES ARE AVAILABLE /5X,(1) '
228      + 'OUTPUT TO THE CRT /5X,(2) OUTPUT TO THE LINE PRINTER '
229      + /1X,'SELECT DESIRED OUTPUT MODE (1 2)??')
230      675 READ(*,625) ID
231      IF(ID LT 1 OR ID GT 2) GOTO 673
232      IF (ID EQ 1) DEV = 0
233      IF(ID EQ 2) DEV = 1
234      678 FORMAT(/1X,'COMPUTING RESULTS NOW')
235      WRITE (* 678)
236      680 CALL RREAD(NLN)

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
237 C*****
238 C*
239 C* COMPUTE TRIANGULAR DISTRIBUTION CUMULATIVE PROBABILITY THRESHOLD VALUE
240 C* FOR USE IN GENERATING DESIRED VARIABLE VALUES FROM A RANDOM NUMBER
241 C* IF THE MIN, ML, AND MAX VALUES ARE EQUAL SKIP THE CALCULATION
242
243 C*
244 C*****
245   700 RN=RANDOM(IX)
246     CIPT=CIP(3)
247     CISST=CISS(3)
248     BPT=BP(3)
249     BSST=BSS(3)
250     IF(CIP(1) NE CIP(3)) CIPT=(CIP(2)-CIP(1))/(CIP(3)-CIP(1))
251     IF(CISS(1) NE CISS(3)) CISST=(CISS(2)-CISS(1))/(CISS(3)-CISS(1))
252     IF(BP(1) NE BP(3)) BPT=(BP(2)-BP(1))/(BP(3)-BP(1))
253     IF(BSS(1) NE BSS(3)) BSST=(BSS(2)-BSS(1))/(BSS(3)-BSS(1))
254     DO 780 I=1,NLOT
1 255     CWP(I)=0
1 256     SFPT(I)=SFP(I,3)
1 257     IF(SFP(I,1) EQ SFP(I,3)) GO TO 750
1 258     SFPT(I)=(SFP(I,2)-SFP(I,1))/(SFP(I,3)-SFP(I,1))
1 259   750 SFSST(I)=SFSS(I,3)
1 260     IF(SFSS(I,1) EQ SFSS(I,3)) GO TO 760
1 261     SFSST(I)=(SFSS(I,2)-SFSS(I,1))/(SFSS(I,3)-SFSS(I,1))
1 262   760 ROTPT(I)=ROTP(I,3)
1 263     IF(ROTP(I,1) EQ ROTP(I,3)) GO TO 770
1 264     ROTPT(I)=(ROTP(I,2)-ROTP(I,1))/(ROTP(I,3)-ROTP(I,1))
1 265   770 ROTSST(I)=ROTSS(I,3)
1 266     IF(ROTSS(I,1) EQ ROTSS(I,3)) GO TO 780
1 267     ROTSST(I)=(ROTSS(I,2)-ROTSS(I,1))/(ROTSS(I,3)-ROTSS(I,1))
1 268   780 CONTINUE
269 C*****
270 C*
271 C* BEGIN THE SIMULATION -----
272 C*
273 C* GENERATE A FIRST UNIT COST AND PERFORMANCE CURVE SLOPE FOR THE PRIME
274 C* AND SECOND SOURCE SUPPLIERS
275 C*
276 C*****
277     DO 910 I=1,NCYC
1 278     PFUC=CIP(3)
1 279     SSFUC=CISS(3)
1 280     PB=BP(3)
1 281     SSB=BSS(3)
1 282     IF(CIP(1) EQ CIP(3)) GO TO 790
1 283     RN=RANDOM(IX)
1 284     IF(RN GE CIPT) PFUC=CIP(3)-DSQRT((1-RN)*(CIP(3)-CIP(1))*
1 285     +(CIP(3)-CIP(2)))
1 286     IF(RN LT CIPT) PFUC=CIP(1)+DSQRT(RN*(CIP(3)-CIP(1))*
1 287     +(CIP(2)-CIP(1)))
1 288   790 IF(CISS(1) EQ CISS(3)) GO TO 800
1 289     RN=RANDOM(IX)
1 290     IF(RN GE CISST) SSFUC=CISS(3)-DSQRT((1-RN)*(CISS(3)-CISS(1))*
1 291     +(CISS(3)-CISS(2)))
1 292     IF(RN LT CISST) SSFUC=CISS(1)+DSQRT(RN*(CISS(3)-CISS(1))*
1 293     +(CISS(2)-CISS(1)))
1 294   800 IF(BP(1) EQ BP(3)) GO TO 810
1 295     RN=RANDOM(IX)

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
1 296      IF(RN GE BPT) PB=BP(3)-DSQRT((1-RN)*(BP(3)-BP(1))*
1 297      +(BP(3)-BP(2)))
1 298      IF(RN LT BPT) PB=BP(1)+DSQRT(RN*(BP(3)-BP(1))*
1 299      +(BP(2)-BP(1)))
1 300      810 IF(BSS(1) EQ BSS(3)) GO TO 820
1 301      RN=RANDOM(IX)
1 302      IF(RN GE BSST) SSB=BSS(3)-DSQRT((1-RN)*(BSS(3)-BSS(1))*
1 303      +(BSS(3)-BSS(2)))
1 304      IF(RN LT BSST) SSB=BSS(1)+DSQRT(RN*(BSS(3)-BSS(1))*
1 305      +(BSS(2)-BSS(1)))
1 306      820 SCOST(1)=0
1 307      PPCOST=0
1 308      SSCOST=0
1 309      GPC=0
1 310      QSSC=0
1 311      PAVG=0
1 312      SAVG=0
1 313      C*****
1 314      C*
1 315      C* CALCULATE FIRST UNIT COST AND SLOPE ADJUSTMENT FOR LOT N DURING THE
1 316      C* GIVEN SIMULATION CYCLE
1 317      C*
1 318      C*****
1 319      DO 910 J=1,NLOT
2 320      PSF=SFP(J,3)
2 321      IF(SFP(J,1) EQ SFP(J,3)) GO TO 830
2 322      RN=RANDOM(IX)
2 323      IF(RN GE SFPT(J)) PSF=SFP(J,3)-DSQRT((1-RN)*(SFP(J,3)-SFP(J,1))*
2 324      +(SFP(J,3)-SFP(J,2)))
2 325      IF(RN LT SFPT(J)) PSF=SFP(J,1)+DSQRT(RN*(SFP(J,3)-SFP(J,1))*
2 326      +(SFP(J,2)-SFP(J,1)))
2 327      830 SSSF=SFSS(J,3)
2 328      IF(SFSS(J,1) EQ SFSS(J,3)) GO TO 840
2 329      RN=RANDOM(IX)
2 330      IF(RN GE SFSST(J)) SSSF=SFSS(J,3)-DSQRT((1-RN)*
2 331      +(SFSS(J,3)-SFSS(J,1))*(SFSS(J,3)-SFSS(J,2)))
2 332      IF(RN LT SFSST(J)) SSSF=SFSS(J,1)+DSQRT(RN*(SFSS(J,3)-SFSS(J,1))*
2 333      +(SFSS(J,2)-SFSS(J,1)))
2 334      840 PROT=ROTP(J,3)
2 335      IF(ROTP(J,1) EQ ROTP(J,3)) GO TO 850
2 336      RN=RANDOM(IX)
2 337      IF(RN GE ROTPT(J)) PROT=ROTP(J,3)-DSQRT((1-RN)*
2 338      +(ROTP(J,3)-ROTP(J,1))*(ROTP(J,3)-ROTP(J,2)))
2 339      IF(RN LT ROTPT(J)) PROT=ROTP(J,1)+DSQRT(RN*
2 340      +(ROTP(J,3)-ROTP(J,1))*(ROTP(J,2)-ROTP(J,1)))
2 341      850 SSROT=ROTSS(J,3)
2 342      IF(ROTSS(J,1) EQ ROTSS(J,3)) GO TO 860
2 343      RN=RANDOM(IX)
2 344      IF(RN GE ROTST(J)) SSROT=ROTSS(J,3)-DSQRT((1-RN)*
2 345      +(ROTSS(J,3)-ROTSS(J,1))*(ROTSS(J,3)-ROTSS(J,2)))
2 346      IF(RN LT ROTST(J)) SSROT=ROTSS(J,1)+DSQRT(RN*
2 347      +(ROTSS(J,3)-ROTSS(J,1))*(ROTSS(J,2)-ROTSS(J,1)))
2 348      C*****
2 349      C*
2 350      C* BASED ON THE RUNNING UNIT AVG COST FOR EACH SOURCE, DETERMINE WHICH
2 351      C* SOURCE WILL WIN THE MAJOR SPLIT QUANTITY WHEN SPLIT BUYS ARE BEING
2 352      C* USED USING THE SELECTED QUANTITIES. CALCULATE LOT COSTS FOR EACH
2 353      C* SOURCE INCREMENT RUNNING CUM COSTS AND UPDATE UNIT AVG COSTS
2 354      C* CONTINUE UNTIL ALL LOTS HAVE BEEN EVALUATED FOR THE GIVEN SIM

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
2  355 C*
2  356 C*****
2  357      860 IF(QPC EQ 0 ) GO TO 870
2  358      BBP=ALOG(PB)/ALOG(2 )
2  359      PUCOST=PFUC*QPC**BBP
2  360      IF(QSSC EQ 0 ) GO TO 870
2  361      BBS=ALOG(SSB)/ALOG(2 )
2  362      SUCOST=SSFUC*QSSC**BBS
2  363      870 QP=QM(J)
2  364      QSS=QP
2  365 C*
2  366 C*
2  367 C* IF QM AND QS ARE EQUAL, A COMPOSITE PERFORMANCE CURVE IS BEING USED
2  368 C* THEREFORE BYPASS THE LOT SPLIT PROCESS
2  369 C*
2  370 C*
2  371      IF(QS(J) EQ QM(J)) GO TO 890
2  372      PAVG=PAVG* PSF
2  373      SAVG=SAVG*SSSF
2  374      IF(SAVG EQ 0 ) GO TO 880
2  375      IF(SAVG LT PAVG) QP=QS(J)
2  376 C*****
2  377 C* The following line of code may be used in an alternate single-source
2  378 C* baseline strategy. it guarantees that the second source is ignored
2  379 C* Use it only when the input data implies there is no split award
2  380 C* i e . only in a no-competition, prime-source-only situation
2  381 C*      IF(QS(J) EQ 0 ) QP=QM(J)
2  382 C*****
2  383 C*
2  384 C*
2  385 C* DECISION RULE FOR SPLIT BUY AWARDS THE SOURCE HAVING THE LOWEST AVG
2  386 C* PRICE AS OF THIS LOT FOR A GIVEN SIMULATION CYCLE IS GIVEN THE MAJOR
2  387 C* SPLIT QUANTITY
2  388 C*
2  389 C*
2  390      880 QSS=QM(J)+QS(J)-QP
2  391      890 IF(QP EQ QM(J)) CWP(J)=CWP(J)+1
2  392      PB=ABS(PB-PROT)
2  393      BBP=ALOG(PB)/ALOG(2 )
2  394      IF(QPC NE 0 ) PFUC=PUCOST/(QPC**BBP)
2  395      IF(PSF NE 1 ) PFUC=PFUC*PSF
2  396      BBP=BBP+1
2  397      SSB=ABS(SSB-SSROT)
2  398      IF(SSB NE 0 ) BBS=ALOG(SSB)/ALOG(2 )
2  399      IF(QSSC NE 0 ) SSFUC=SUCOST/(QSSC**BBS)
2  400      IF(SSSF NE 1 ) SSFUC=SSFUC*SSSF
2  401      BBS=BBS+1
2  402      Q1P=QPC
2  403      Q2P=QPC+QP
2  404      Q1S=QSSC
2  405      Q2S=QSSC+QSS
2  406      QPC=Q2P
2  407      QSSC=Q2S
2  408      DF=1
2  409      IF(DIS EQ 0 ) GO TO 900
2  410      IF(J LT IDF) GO TO 900
2  411      IJ=J-IDF+1
2  412      DF=DISC(IJ)
2  413      900 YP=DF*PFUC*((Q2P+ 5)**BBP-(Q1P+ 5)**BBP)/BBP

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
2   414      YS=DF*SSFUC*((Q2S+ 5)**BBS-(Q1S+ 5)**BBS)/BBS
2   415 C*
2   416 C*
2   417 C* IF QM AND QS ARE EQUAL. A COMPOSITE PERFORMANCE CURVE IS BEING USED
2   418 C* THEREFORE THE LOT COST FOR EACH SOURCE IS EQUAL
2   419 C*
2   420 C*
2   421      IF(QS(J) EQ QM(J)) YS=YP
2   422      SCOST(I)=SCOST(I)+YP+YS
2   423      PPCOST=PPCOST+YP
2   424      SSCOST=SSCOST+YS
2   425      PAVG=PPCOST/Q2P
2   426      IF(Q2S NE 0 ) SAVG=SSCOST/Q2S
2   427      ACLOT(J)=ACLOT(J)+(YP+YS)/CYCLES
2   428      ALUC(J)=ALUC(J)+(YP+YS)/(CYCLES*(QM(J)+QS(J)))
2   429      IF(QP NE 0 ) APUCL(J)=APUCL(J)+YP/QP
2   430      IF(QSS NE 0 ) ASUCL(J)=ASUCL(J)+YS/QSS
2   431      910 CONTINUE
2   432 C*
2   433 C*
2   434 C* COMPUTE THE AVERAGE UNIT COST FOR EACH SOURCE AT LOT N THIS PROCESS
2   435 C* REQUIRED SINCE A BUYOUT PERIOD CHANGES THE NUMBER OF CYCLES A GIVEN
2   436 C* SOURCE WILL APPEAR IN A GIVEN SIMULATION RUN DUE TO THE AWARD RULE
2   437 C*
2   438 C*
2   439      DO 920 I=1 NLOT
1   440      DP=CYCLES
1   441      DS=CYCLES
1   442      IF(QS(I) EQ 0 ) DS=CYCLES-CWP(I)
1   443      IF(QS(I) EQ 0 ) DP=CWP(I)
1   444      IF(DS EQ 0 ) LS=CYCLES
1   445      IF(DP EQ 0 ) DP = CYCLES
1   446      APUCL(I)=APUCL(I)/DP
1   447      ASUCL(I)=ASUCL(I)/DS
1   448      920 CONTINUE
2   449 C*****
2   450 C*
2   451 C* SORT THE SIMULATION RESULTS AND COMPUTE THE CLASS INTERVAL SIZE ---
2   452 C* (MAX - MIN)/50 ---- ALSO COMPUTE THE AVERAGE COST AND DETERMINE THE
2   453 C* MEDIAN COST
2   454 C*
2   455 C*****
2   456      N=NCYC
2   457      930 ICK=0
2   458      N=N-1
2   459      IF(N EQ 1) GO TO 960
2   460      DO 950 I=1,N
1   461      IF(SCOST(I) GE SCOST(I+1)) GO TO 950
1   462      940 HOLD=SCOST(I)
1   463      SCOST(I)=SCOST(I+1)
1   464      SCOST(I+1)=HOLD
1   465      ICK=1
1   466      950 CONTINUE
2   467      IF(ICK GT 0) GO TO 930
2   468      960 FINC=(SCOST(1)-SCOST(NCYC))/50
2   469      IF (FINC GT 1) GO TO 965
2   470      PDF=0
2   471      PCF=0
2   472      WRITE(DEV 963) SCOST(1)

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
473 963 FORMAT(1X,'DETERMINISTIC RUN MIN COST=MAX COST=',F11.0)
474      AVG=SCOST(1)
475      AMID=SCOST(1)
476      GO TO 992
477 965 ICK=NCYC/2+1
478      AMID=SCOST(ICK)
479      DO 970 I=1,50
1 480      PCOST(I)=0
1 481 970 CONTINUE
482      AVG=0
483      DO 980 I=1,NCYC
1 484      AVG=AVG+SCOST(I)/CYCLES
1 485      J=((SCOST(I)-SCOST(NCYC))/FINC)+1
1 486      IF(J GT 50) J=50
1 487      PCOST(J)=PCOST(J)+1
1 488 980 CONTINUE
489      DO 990 I=1,50
1 490      PCOST(I)=PCOST(I)/CYCLES
1 491 990 CONTINUE
492 992 DO 1000 I=1,50
1 493      PV(I)=' '
1 494 1000 CONTINUE
495      LC=1
496      IPAGE=0
497      WRITE(DEV,1010)
498 1010 FORMAT(1H1)
499      CALL PAGE(IPAGE,LC,TCK,DEV)
500      WRITE(DEV,1115)
501 1115 FORMAT(10X,'ARMY PROCUREMENT RESEARCH OFFICE (APRO)',/,/,1X,
502      + 'MICROCOMPUTER VERSION OF COMPETITION DECISION ASSIST PACKAGE')
503      LC=LC+4
504      IF(DIS EQ 0) GO TO 1050
505      WRITE(DEV,1020) IDF
506      LC=LC+3
507 1020 FORMAT(/17X,'--- RESULTS ARE IN DISCOUNTED DOLLARS ---',
508      + /15X,'( 10% ) BEGINNING WITH LOT NUMBER ',I2)
509 1050 WRITE(DEV,1060) AVG,AMID
510      LC = LC + 4
511 1060 FORMAT(/20X,'***** AVERAGE COST = ',F11.0,' *****')
512      + /20X,'***** MEDIAN COST = ',F11.0,' *****')
513      WRITE(DEV,1070) SCOST(1), SCOST(NCYC)
514      LC = LC + 4
515 1070 FORMAT(/20X,'***** MAXIMUM COST = ',F11.0,' *****')
516      + /20X,'***** MINIMUM COST = ',F11.0,' *****')
517      WRITE(DEV,1075) SCOST(1) - SCOST(NCYC)
518      LC = LC + 2
519 1075 FORMAT(/20X,'***** RANGE = ',F11.0,' *****')
520      IFORTH = NCYC/4
521      WRITE(DEV,1076) SCOST(NCYC-IFORTH), SCOST(1+IFORTH)
522 1076 FORMAT(/20X,'***** FOURTH-SPREAD ',F11.0,' *****')
523      + /20X,'*****',16X,F11.0,' *****')
524      LC = LC + 5
525      IF(DEV EQ 0) PAUSE
526      IF(DIS EQ 1) WRITE(DEV,1020) IDF
527      IF(DIS EQ 1) LC=LC+3
528      IF(TCK EQ 1) WRITE(*, '(1X,2A)') CH,'(2J)'
529      CALL HEAD1(DEV)
530      LC=LC+5
531      DO 1100 I=1,NLOT

```

```

D Line# 1      7
1 532      TICK=' '
1 533      IF(DIS EQ 0) GO TO 1080
1 534      IF(I GE IDF) TICK='*'
1 535 1080 CWP(I)=CWP(I)/CYCLES*100
1 536      TOTAL=QM(I)+QS(I)
1 537      WRITE(DEV,1090) I,CWP(I),TOTAL,ACLOT(I),TICK,APUCL(I),TICK,
1 538      +ASUCL(I),TICK,ALUC(I),TICK
1 539 1090 FORMAT(2X,12.3X,F6.2,4X,17.4X,F11.0,A1,1X,F7.0,A1,4X,
1 540      +F7.0,A1,5X,F7.0,A1)
1 541      LC=LC+1
1 542      IF(LC LT 57) GOTO 1100
1 543      IF(LC LT 63) GO TO 1100
1 544      CALL ENDP(LC,TCK,ITT,DEV)
1 545      CALL PAGE(IPAGE,LC,TCK,DEV)
1 546      CALL HEAD1(DEV)
1 547      LC=LC+5
1 548 1100 CONTINUE
549      TOTAL=QPC+QSSC
550      WRITE(DEV,1110) TOTAL
551      IF (TCK EQ 1) WRITE(*,1120) 'CH, '02J'
552      LC=LC+2
553      IF(LC LT 52) GOTO 1150
554      IF(LC LT 56) GO TO 1150
555      CALL ENDP(LC,TCK,ITT,DEV)
556 1105 CALL PAGE(IPAGE,LC,TCK,DEV)
557 1110 FORMAT(1X,'TOTAL NUMBER OF UNITS = ',I7)
558 1120 FORMAT(//)
559 1150 LC=LC+4
560      WRITE(DEV,1120)
561 1160 FORMAT(//1X,'DATA USED IN RUN---> ',2A6,2X,A8,2X,
562      +//1X,79(' '))
563      IF(DEV EQ 0) PAUSE
564      WRITE(DEV,1170) CIP
565      LC=LC+1
566      IF(LC LT 57) GOTO 1175
567      IF(LC LT 63) GO TO 1175
568      CALL ENDP(LC,TCK,ITT,DEV)
569      CALL PAGE(IPAGE,LC,TCK,DEV)
570 1170 FORMAT(//1X,'PRIME FIRST UNIT COST--MIN>',F8.0,2X,'MOST LIKELY>',
571      +F8.0,2X,'MAX>',F8.0)
572 1175 WRITE(DEV,1180) CISS
573      LC=LC+1
574      IF(LC LT 57) GOTO 1185
575      IF(LC LT 63) GO TO 1185
576      CALL ENDP(LC,TCK,ITT,DEV)
577      CALL PAGE(IPAGE,LC,TCK,DEV)
578 1180 FORMAT(1X,'SECOND SOURCE-----MIN>',F8.0,2X,'MOST LIKELY>',
579      +F8.0,2X,'MAX>',F8.0)
580 1185 WRITE(DEV,1190) BP
581      LC=LC+1
582      IF(LC LT 59) GOTO 1195
583      IF(LC LT 63) GO TO 1195
584      CALL ENDP(LC,TCK,ITT,DEV)
585      CALL PAGE(IPAGE,LC,TCK,DEV)
586 1190 FORMAT(1X,'PRIME PCURVE SLOPE----MIN>',1X,F5.3,4X,'MOST LIKELY>',
587      +1X,F5.3,4X,'MAX>',1X,F5.3)
588 1195 WRITE(DEV,1200) BSS
589      LC=LC+1
590      IF(LC LT 57) GOTO 1205

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
591      IF(LC LT 63) GO TO 1205
592      CALL ENDP(LC,TCK,ITT,DEV)
593      CALL PAGE(IPAGE,LC,TCK,DEV)
594 1200  FORMAT(1X,'SECOND SOURCE-----MIN)',1X,F5 3,4X,' MOST LIKELY'
595      +1X,F5 3,4X,'MAX)',1X,F5 3)
596 1205  WRITE(DEV,1210) NCYC
597      IF (TCK EQ 1) WRITE(*,'(1X,2A)')CH,'[2J'
598      LC=LC+1
599      IF(LC LT 52) GOTO 1215
600      IF(LC LT 56) GO TO 1215
601      CALL ENDP(LC,TCK,ITT,DEV)
602      CALL PAGE(IPAGE,LC,TCK,DEV)
603 1210  FORMAT(1X,'NUMBER OF CYCLES---> ',14)
604 1215  CALL HEAD2(DEV)
605      LC=LC+5
606      DO 1240 I=1,NLOT
1 607      WRITE(DEV,1230) I,QM(I),QS(I),(SFP(I,J),J=1,3),(SFSS(I,J),J=1,3),
1 608      +(ROTP(I,J),J=1,3),(ROTSS(I,J),J=1,3)
1 609      LC=LC+1
1 610      IF(LC LT 57) GOTO 1240
1 611      IF(LC LT 63) GO TO 1240
1 612      IF(I EQ NLOT) GO TO 1240
1 613      IF (TCK EQ 1) WRITE(*,'(1X,2A)')CH,'[2J'
1 614      CALL ENDP(LC,TCK,ITT,DEV)
1 615      CALL PAGE(IPAGE,LC,TCK,DEV)
1 616      CALL HEAD2(DEV)
1 617      LC=LC+5
1 618 1230  FORMAT(1X,12,1X,F5 0,1X,F5 0,12(1X,F4 2))
1 619 1240  CONTINUE
620      IF (TCK EQ 1) WRITE(*,'(1X,2A)')CH,'[2J'
621      CALL ENDP(LC,TCK,ITT,DEV)
622      IF(DEV EQ 0) PAUSE
623 C*****
624 C*
625 C* IF SELECTED GENERATE A CUMULATIVE PROBABILITY PLOT
626 C*
627 C*****
628      IF(PCF EQ 0 ) GO TO 1246
629      CALL PAGE(IPAGE,LC,TCK,DEV)
630      WRITE(DEV,1245)
631      SF=10
632      CALL SCALE(SF,DEV)
633      CALL LINE(DEV)
634      CUM=1
635      CALL PLOT1(SF,CUM,NCYC,FINC,DEV)
636      CALL LINE(DEV)
637      CALL SCALE(SF,DEV)
638      IF (DEV NE 1) WRITE(DEV,1245)
639 1245  FORMAT(1X,'STRATEGY COST',18X,'CUMULATIVE PROBABILITY'
640      LC=LC+57
641      IF (TCK EQ 1) WRITE(*,'(1X,2A)')CH,'[2J'
642      IF(DEV EQ 0) PAUSE
643      CALL ENDP(LC,TCK,ITT,DEV)
644 1246  IF(PDF EQ 0) GO TO 1248
645      CALL PAGE(IPAGE,LC,TCK,DEV)
646 C*****
647 C*
648 C* IF SELECTED GENERATE A PROBABILITY DENSITY PLOT
649 C*

```

D Line# 1 7 IBM Personal Computer FORTRAN Compiler V2 00

```

650 C*****
651     CUM=0
652     SF=100
653     WRITE(DEV,1247)
654     CALL SCALE(SF,DEV)
655     CALL LINE(DEV)
656     CALL PLOTA(SF,CUM,NCYC,FINC,DEV)
657     CALL LINE(DEV)
658     CALL SCALE(SF,DEV)
659     WRITE(DEV,1247)
660 1247 FORMAT(1X,'STRATEGY COST',24X,'PROBABILITY')
661     LC=LC+57
662     IF (TCK EQ 1) WRITE(*,'(1X,2A)')CH,'[2J'
663     CALL ENDP(LC,TCK,ITT,DEV)
664     IF(DEV EQ 0) PAUSE
665 1248 IPAGE='E'
666     CALL PAGE(IPAGE,LC,TCK,DEV)
667     WRITE(*,1249)
668 1249 FORMAT(//,1X,'ANOTHER RUN (Y,N) ? ')
669     READ(*,1250) ANS
670 1250 FORMAT(A1)
671     DEV = 0
672     IF ( ( ANS EQ 'Y' ) OR ( ANS EQ 'y' ) ) GOTO 570
673     GO TO 550
674 1255 WRITE (*,1256)
675 1256 FORMAT(1X,'CDAP FINISHED  --  GOOD DAY!')
676     STOP
677     END

```

Name	Type	Offset	P	Class
------	------	--------	---	-------

ABS				INTRINSIC
ACLOT	REAL*8	2992	/RD	/
ALOG				INTRINSIC
ALUC	REAL*8	3192	/RD	/
AMID	REAL	2462		
ANS	CHAR*1	1362		
APUCL	REAL*8	3392	/RD	/
ASUCL	REAL*8	3592	/RD	/
AVG	REAL*8	2454		
BBP	REAL	2308		
BBS	REAL	2316		
BP	REAL*8	144	/RD	/
BPT	REAL	2140		
BSS	REAL*8	168	/RD	/
BSST	REAL	2144		
CH	CHAR*1	1118		
CHAR				INTRINSIC
CIP	REAL*8	96	/RD	/
CIPT	REAL	2132		
CISS	REAL*8	120	/RD	/
CISST	REAL	2136		
COST	REAL*8	*****		
CUM	REAL	3704		
CWP	REAL*8	802		
CYCLES	REAL	1802		
DEV	INTEGER*4	1120		
DF	REAL	2348		
DIS	REAL	1806		

IBM Personal Computer FORTRAN Compiler V2 00

D Line# 1	7		
DISC	REAL	1002	
DP	REAL	2376	
DS	REAL	2380	
DSQRT			INTRINSIC
FID	REAL*8	0	/RD /
FINC	REAL	2400	
FNAME	CHAR*12	3792	/RD /
HOLD	REAL	2396	
I	INTEGER*4	1572	
ICK	INTEGER*4	2388	
ID	INTEGER*4	2084	
IDAY	INTEGER*2	3812	/RD /
IDF	INTEGER*4	1912	
IFORTH	INTEGER*4	2976	
IJ	INTEGER*4	2352	
IMONTH	INTEGER*2	3810	/RD /
IPAGE	INTEGER*4	2474	
ITT	INTEGER*4	1114	
IX	REAL*8	1564	
IY	INTEGER*4	1560	
IYEAR	INTEGER*2	3808	/RD /
J	INTEGER*4	2252	
LC	INTEGER*4	2470	
M1	INTEGER*4	1102	
M2	INTEGER*4	1106	
M3	INTEGER*4	1110	
N	INTEGER*4	2384	
NCYC	INTEGER*4	1792	
NLN	INTEGER*4	2120	
NLOT	INTEGER*4	3804	/RD /
OPN	LOGICAL*4	*****	
PAVG	REAL*8	2236	
PB	REAL	2172	
PCF	REAL	1592	
PCOST	REAL*8	408	/A /
PDF	REAL	1588	
PFUC	REAL*8	2156	
PPCOST	REAL*8	2212	
PROT	REAL	2284	
PSF	REAL	2260	
PUCOST	REAL	2312	
PV	REAL*8	0	/A /
Q1P	REAL	2332	
Q1S	REAL	2340	
Q2P	REAL	2336	
Q2S	REAL	2344	
QM	REAL*8	192	/RD /
QP	REAL	2324	
QPC	REAL	2228	
QS	REAL*8	392	/RD /
QSS	REAL	2328	
QSSC	REAL	2232	
RANDOM	REAL*8		FUNCTION
RN	REAL	1576	
ROTP	REAL*8	1792	/RD /
ROTPT	REAL*8	602	
ROTSS	REAL*8	2392	/RD /
ROTSST	REAL*8	402	
SAVG	REAL*8	2244	

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
SCOST REAL*8      808  /A  /
SF     REAL      3700
SFP    REAL*8      592  /RD  /
SFPT   REAL*8      202
SFSS   REAL*8     1192  /RD  /
SFSST  REAL*8        2
SHOW   LOGICAL*4   *****
SSB    REAL      2176
SSCOST REAL*8     2220
SSFUC  REAL*8     2164
SSROT  REAL      2296
SSSF   REAL      2272
SUCOST REAL      2320
TCK    INTEGER*4   2484
TICK   CHAR*1      3076
TOTAL  INTEGER*4   3078
YP     REAL*8     2356
YS     REAL*8     2364

```

```

678 C*****
679 C*
680 C* THIS RANDOM(IX) FUNCTION GENERATES A UNIFORM RANDOM NUMBER
681 C* BETWEEN 0 0 AND 1 0
682 C* THE RANDOM NUMBER GENERATOR IS BASED ON DRAND(IX) GIVEN IN
683 C* 'SIMULATION MODELING AND ANALYSIS' BY LAW AND KELTON
684 C*
685 C*****
686     DOUBLE PRECISION FUNCTION RANDOM(IX)
687     DOUBLE PRECISION A,P,IX,B15,B16,XHI,XALO,LEFTLO,FHI,K
688     DATA A/16807 D0/,B15/32768 D0/,B16/65536 D0/P/2147483647 D0/
689     XHI=IX/B16
690     XHI=XHI-DMOD(XHI,1 D0)
691     XALO=(IX-XHI*B16)*A
692     LEFTLO=XALO/B16
693     LEFTLO=LEFTLO-DMOD(LEFTLO,1 D0)
694     FHI=XHI*A+LEFTLO
695     K=FHI/B15
696     K=K-DMOD(K,1 D0)
697     IX=((XALO-LEFTLO*B16)-P)+(FHI-K*B15)*B16)+K
698     IF(IX LT 0 D0)IX=IX+P
699     RANDOM=IX*4 656612875D-10
700     RETURN
701     END

```

Name	Type	Offset	P	Class
A	REAL*8	3872		
B15	REAL*8	3880		
B16	REAL*8	3888		
DMOD				INTRINSIC
FHI	REAL*8	3928		
IX	REAL*8	0	*	
K	REAL*8	3936		
LEFTLO	REAL*8	3920		
P	REAL*8	3896		
XALO	REAL*8	3912		
XHI	REAL*8	3904		

```

702 C*****

```

D Line# 1 7 IBM Personal Computer FORTRAN Compiler V2 00

```

703 C*
704 C* SCALE ----- ARGUMENT IS THE SCALE FACTOR TO BE USED IN LABELING THE
705 C* PLOT AND LATER CALCULATING PLOT POINTS.
706 C*
707 C*****
708 SUBROUTINE SCALE(SF,IDEV)
709 DIMENSION D(10)
710 DO 1260 I=1,10
1 711 D(I)=I/SF
1 712 1260 CONTINUE
713 WRITE(IDEV,1270) D
714 1270 FORMAT(15X,'0.00',10(1X,F4.2))
715 RETURN
716 END

```

Name	Type	Offset	P	Class
------	------	--------	---	-------

D	REAL	3944		
I	INTEGER*4	3984		
IDEV	INTEGER*4	4	*	
SF	REAL	0	*	

```

717 C*****
718 C*
719 C* LINE ----- THIS SUBROUTINE SIMPLY LAYS OUT THE REFERENCE MARKS FOR
720 C* SELECTED SCALE
721 C*
722 C*****
723 SUBROUTINE LINE(IDEV)
724 WRITE(IDEV,1280)
725 1280 FORMAT(16X,'+',10('----+'))
726 RETURN
727 END

```

Name	Type	Offset	P	Class
------	------	--------	---	-------

IDEV	INTEGER*4	0	*	
------	-----------	---	---	--

```

728 C*****
729 C*
730 C* PLOTA ----- ARGUMENTS ARE THE SCALE FACTOR, SELECTION PARAMETER FOR
731 C* CUMULATIVE OR DENSITY PLOT(CUM), VALUE OF THE
732 C* NUMBER OF SIMULATION CYCLES(NCYC), AND CLASS INTERVAL
733 C* VALUE(FINC)
734 C*
735 C*****
736 SUBROUTINE PLOTA(SF,CUM,NCYC,FINC,IDEV)
737 REAL*8 PV,PCOST,SCOST
738 COMMON/A/PV(51),PCOST(50),SCOST(5000)
739 DIMENSION AV(51)
740 CHARACTER AV
741 COST=SCOST(NCYC)
742 VAL=0
743 DO 1290 I=1,50
1 744 AV(I)=' '
1 745 1290 CONTINUE
746 WRITE(IDEV,1320) COST,(AV(J),J=1,50),VAL

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
      747      COST=COST-FINC/2
      748      DO 1330 I=1,50
1      749      IF(CUM.EQ 0 ) VAL=PCOST(I)
1      750      IF(CUM.EQ 1 ) VAL=VAL+PCOST(I)
1      751      IVAL=VAL*5 *SF+ 5
1      752      IF(IVAL GT 51)IVAL=51
1      753      DO 1300 J=1,IVAL
2      754      AV(J)='*'
2      755      1300 CONTINUE
1      756      IVAL=IVAL+1
1      757      DO 1310 J=IVAL,50
2      758      AV(J)=' '
2      759      1310 CONTINUE
1      760      COST=COST+FINC
1      761      WRITE(IDEV,1320) COST,(AV(J),J=1,50),VAL
1      762      1320 FORMAT(4X,F11.0,1X,' ',50A1,2X,F6.4)
1      763      1325 FORMAT(4X,F11.0,1X,' ',50A1,2X,F6.4)
1      764      1330 CONTINUE
      765      RETURN
      766      END

```

Name	Type	Offset	P	Class
AV	CHAR*1	4034		
COST	REAL	4086		
CUM	REAL	4	*	
FINC	REAL	12	*	
I	INTEGER*4	4094		
IDEV	INTEGER*4	16	*	
IVAL	INTEGER*4	4106		
J	INTEGER*4	4098		
NCYC	INTEGER*4	8	*	
PCOST	REAL*8	408	/A	/
PV	REAL*8	0	/A	/
SCOST	REAL*8	808	/A	/
SF	REAL	0	*	
VAL	REAL	4090		

```

767 C*****
768 C*
769 C* RREAD ----- ARGUMENT IS THE FILE NAME LENGTH THIS SUBROUTINE
770 C*          READS THE SELECTED DATA FILE FOR THOSE PROGRAM MODES THAT
771 C*          USE AN EXISTING FILE
772 C*
773 C*****
774      SUBROUTINE RREAD(NLN)
775      COMMON/RD/FID(12),CIP(3),CISS(3),BP(3),BSS(3),QM(25),
776      +QS(25),SFP(25,3),SFSS(25,3),ROTP(25,3),ROTSS(25,3),ACLOT(25),
777      +ALUC(25),APUCL(25),ASUCL(25),FNAME,NLOT,IYEAR,IMONTH,IDAY
778      REAL*8 FID,CIP,CISS,BP,BSS,QM,QS,SFP,SFSS,ROTP,ROTSS,ACLOT,
779      +ALUC,APUCL,ASUCL
780      CHARACTER*12 FNAME,DUMMY
781      INTEGER*2 IYEAR,IMONTH,IDAY
782      CALL GETFILE(FNAME,3)
783      READ(3,1350,END=1415)NLOT
784      1350 FORMAT(5X,I2)
785      READ(3,1360)CIP
786      1360 FORMAT(29X,F8.0,1X,F8.0,6X,F8.0)

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
      787      READ(3,1360) CISS
      788      READ(3,1370) BP
      789 1370  FORMAT(29X,F5.3,17X,F5.3,9X,F5.3)
      790      READ(3,1370) BSS
      791      DO 1390 I=1,4
1      792      READ(3,1380) DUMMY
1      793 1380  FORMAT(A12)
1      794 1390  CONTINUE
      795      DO 1410 I=1,NLOT
1      796      READ(3,1400,END=1415)QM(I),QS(I),(SFP(I,J),J=1,3),(SFSS(I,J),
1      797      +J=1,3),(ROTP(I,J),J=1,3),(ROTSS(I,J),J=1,3)
1      798 1400  FORMAT(3X,F5.0,1X,F5.0,12(1X,F4.2))
1      799      ACLOT(I)=0
1      800      ALUC(I)=0
1      801      APUCL(I)=0
1      802      ASUCL(I)=0
1      803 1410  CONTINUE
      804 1415  CLOSE(3)
      805      RETURN
      806      END

```

Name	Type	Offset	P	Class
------	------	--------	---	-------

ACLOT	REAL*8	2992	/RD	/
ALUC	REAL*8	3192	/RD	/
APUCL	REAL*8	3392	/RD	/
ASUCL	REAL*8	3592	/RD	/
BP	REAL*8	144	/RD	/
BSS	REAL*8	168	/RD	/
CIP	REAL*8	96	/RD	/
CISS	REAL*8	120	/RD	/
DUMMY	CHAR*12	4246		
FID	REAL*8	0	/RD	/
FNAME	CHAR*12	3792	/RD	/
I	INTEGER*4	4242		
IDAY	INTEGER*2	3812	/RD	/
IMONTH	INTEGER*2	3810	/RD	/
IYEAR	INTEGER*2	3808	/RD	/
J	INTEGER*4	4268		
NLN	INTEGER*4	0	*	
NLOT	INTEGER*4	3804	/RD	/
QM	REAL*8	192	/RD	/
QS	REAL*8	392	/RD	/
ROTP	REAL*8	1792	/RD	/
ROTSS	REAL*8	2392	/RD	/
SFP	REAL*8	592	/RD	/
SFSS	REAL*8	1192	/RD	/

```

807 C*****

```

```

808 C*

```

```

809 C* CREATE ---- THIS SUBROUTINE IS USED TO CREATE A NEW DATA FILE
810 C*      REMOVAL OF FILES IS HANDLED WHILE IN THE OPERATING
811 C*      SYSTEM COMMAND MODE      ARGUMENTS REPRESENT ENTRY
812 C*      POINTS WHEN CALLED FROM *MODIFY      M1 SETS THE
813 C*      BASIC ENTRY. M2 IS THE STARTING LOT NUMBER. AND M3
814 C*      IS THE ENDING LOT NUMBER

```

```

815 C*

```

```

816 C*****

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
817      SUBROUTINE CREATE(M1,M2,M3)
818      COMMON/RD/FID(12),CIP(3),CISS(3),BP(3),BSS(3),QM(25),
819      +QS(25),SFP(25,3),SFSS(25,3),ROTP(25,3),ROTSS(25,3),ACLOT(25),
820      +ALUC(25),APUCL(25),ASUCL(25),FNAME,NLOT,IYEAR,IMONTH,IDAY
821      REAL*8 PV,PCOST,SCOST,FID,CIP,CISS,ACLOT,ALUC,BP,BSS,SFP,SFSS,
822      +ROTP,ROTSS,SFPT(25),SFSST(25),ROTPT(25),ROTSST(25),QM,QS,CWP(25)
823      +,AVG,COST,PPCOST,SSCOST,PAVG,SAVG,YP,YS,SSFUC,PFUC,ASUCL,APUCL
824      REAL DISC(25)
825      CHARACTER*12 FNAME,ANS
826      INTEGER*2 IYEAR,IMONTH,IDAY
827      LOGICAL OPN
828      IF(M1 EQ 1)GO TO 1555
829      IF(M1 EQ 2)GO TO 1495
830      IF(M1 EQ 3)GO TO 1525
831      IF(M1 EQ 4)GO TO 1535
832      IF(M1 EQ 5)GO TO 1545
833      IF(M1 EQ 6)GO TO 1555
834 1420 WRITE(*,1430)
835 1430 FORMAT(/1X,'WHAT IS THE DATA FILE NAME')
836      READ(*,1440) FNAME
837      WRITE(*,1445) FNAME
838 1440 FORMAT(A12)
839 1445 FORMAT(1X,'THE DATA FILE NAME IS ',A12)
840 1448 FORMAT(I2)
841 1450 FORMAT(F8.0)
842 1485 WRITE(*,1490)
843 1490 FORMAT(/1X,'HOW MANY LOTS ARE THERE IN THIS DATA SET ')
844      READ(*,1448)NLOT
845      M2=1
846      M3=NLOT
847 1493 IF (M1 EQ 1) RETURN
848 1495 WRITE(*,1500)
849 1500 FORMAT(/1X,'FIRST UNIT COST FOR PRIME'/1X,'MINIMUM ')
850      READ(*,1450)CIP(1)
851      WRITE(*,1510)
852 1510 FORMAT(/1X,'MOST LIKELY ')
853      READ(*,1450)CIP(2)
854      WRITE(*,1520)
855 1520 FORMAT(/1X,'MAXIMUM ')
856      READ(*,1450)CIP(3)
857      IF(M1 EQ 2)RETURN
858 1525 WRITE(*,1530)
859 1530 FORMAT(/1X,'SECOND SOURCE FIRST UNIT COST'/1X,'MINIMUM ')
860      READ(*,1450)CISS(1)
861      WRITE(*,1510)
862      READ(*,1450)CISS(2)
863      WRITE(*,1520)
864      READ(*,1450)CISS(3)
865      IF(M1 EQ 3)RETURN
866 1535 WRITE(*,1540)
867 1540 FORMAT(/1X,'PRIME PERFORMANCE CURVE SLOPE (XXX)'/1X
868      + 'MINIMUM ')
869      READ(*,1450)BP(1)
870      WRITE(*,1510)
871      READ(*,1450)BP(2)
872      WRITE(*,1520)
873      READ(*,1450)BP(3)
874      IF(M1 EQ 4)RETURN
875 1545 WRITE(*,1550)

```

D Line# 1 7 IBM Personal Computer FORTRAN Compiler V2 00

```

876 1550 FORMAT(/1X,'SECOND SOURCE PERFORMANCE CURVE SLOPE ( XXX)'/
877      + 1X,'MINIMUM ')
878      READ(*,1450)BSS(1)
879      WRITE(*,1510)
880      READ(*,1450)BSS(2)
881      WRITE(*,1520)
882      READ(*,1450)BSS(3)
883      IF(M1 EQ 5)RETURN
884 1555 DO 1620 I=M2,M3
1 885      WRITE(*,1560)I
1 886 1560 FORMAT(/1X,'DATA FOR LOT # ',I2/1X,'MAJOR SPLIT QUANTITY')
1 887      READ(*,1450)QM(I)
1 888      WRITE(*,1570)
1 889 1570 FORMAT(/1X,'MINOR SPLIT QUANTITY')
1 890      READ(*,1450)QS(I)
1 891      WRITE(*,1580)
1 892 1580 FORMAT(/1X,'PRIME SHIFT FACTOR ( XXX)'/1X,'MINIMUM ')
1 893      READ(*,1450)SFP(I,1)
1 894      WRITE(*,1510)
1 895      READ(*,1450)SFP(I,2)
1 896      WRITE(*,1520)
1 897      READ(*,1450)SFP(I,3)
1 898      WRITE(*,1590)
1 899 1590 FORMAT(/1X,'SECOND SOURCE SHIFT FACTOR ( XXX)'/1X,'MINIMUM ')
1 900      READ(*,1450)SFSS(I,1)
1 901      WRITE(*,1510)
1 902      READ(*,1450)SFSS(I,2)
1 903      WRITE(*,1520)
1 904      READ(*,1450)SFSS(I,3)
1 905      WRITE(*,1600)
1 906 1600 FORMAT(/1X,'PRIME ROTATION FACTOR ( XXX)'/1X,'MINIMUM ')
1 907      READ(*,1450)ROTP(I,1)
1 908      WRITE(*,1510)
1 909      READ(*,1450)ROTP(I,2)
1 910      WRITE(*,1520)
1 911      READ(*,1450)ROTP(I,3)
1 912      WRITE(*,1610)
1 913 1610 FORMAT(/1X,'SECOND SOURCE ROTATION FACTOR ( XXX)'/1X,
1 914      + 'MINIMUM ')
1 915      READ(*,1450)ROTSS(I,1)
1 916      WRITE(*,1510)
1 917      READ(*,1450)ROTSS(I,2)
1 918      WRITE(*,1520)
1 919      READ(*,1450)ROTSS(I,3)
1 920 1620 CONTINUE
921      IF(M1 EQ 1)RETURN
922      IF(M1 EQ 6)RETURN
923      CALL RWRITE
924      WRITE(*,1630)
925 1630 FORMAT(/1X,'ANOTHER FILE (Y,N) ? ')
926      READ(*,1640)ANS
927 1640 FORMAT(A1)
928      IF ( ( ANS EQ 'Y' ) OR ( ANS EQ 'y' ) ) GOTO 1420
929      RETURN
930      END

```

Name Type Offset P Class

ACLOT REAL*8 2992 /RD /

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2.00
ALUC  REAL*8      3192  /RD  /
ANS   CHAR*12      6205
APUCL REAL*8      3392  /RD  /
ASUCL REAL*8      3592  /RD  /
AVG   REAL*8      *****
BP    REAL*8       144   /RD  /
BSS   REAL*8       168   /RD  /
CIP   REAL*8        96   /RD  /
CISS  REAL*8       120   /RD  /
COST  REAL*8      *****
CWP   REAL*8      5118
DISC  REAL       5318
FID   REAL*8        0    /RD  /
FNAME CHAR*12     3792  /RD  /
I     INTEGER*4    5856
IDAY  INTEGER*2    3812  /RD  /
IMONTH INTEGER*2   3810  /RD  /
IYEAR INTEGER*2    3808  /RD  /
M1    INTEGER*4     0 *
M2    INTEGER*4     4 *
M3    INTEGER*4     8 *
NLOT  INTEGER*4    3804  /RD  /
OPN   LOGICAL*4    *****
PAVG  REAL*8      *****
PCOST REAL*8      *****
PFUC  REAL*8      *****
PPCOST REAL*8     *****
PV    REAL*8      *****
QM    REAL*8       192   /RD  /
QS    REAL*8       392   /RD  /
ROTP  REAL*8      1792  /RD  /
ROTPT REAL*8      4918
ROTSS REAL*8      2392  /RD  /
ROTSST REAL*8     4718
SAVG  REAL*8      *****
SCOST REAL*8      *****
SFP   REAL*8       592   /RD  /
SFPT  REAL*8      4518
SFSS  REAL*8      1192  /RD  /
SFSST REAL*8      4318
SSCOST REAL*8     *****
SSFUC REAL*8      *****
YP    REAL*8      *****
YS    REAL*8      *****

```

```

931 C*****

```

```

932 C*

```

```

933 C*  RWRITE ----- THIS SUBROUTINE WRITES DATA IN TO A PERMANENT FILE

```

```

934 C*          IT IS USED BY *CREATE AND *MODIFY

```

```

935 C*

```

```

936 C*****

```

```

937      SUBROUTINE RWRITE

```

```

938      COMMON/RD/FID(12),CIP(3),CISS(3),BP(3),BSS(3),QM(25),

```

```

939      +QS(25),SFP(25,3),SFSS(25,3),ROTP(25,3),ROTSS(25,3),ACLOT(25)

```

```

940      +ALUC(25),APUCL(25),ASUCL(25),FNAME,NLOT,IYEAR,IMONTH,IDAY

```

```

941      REAL*8 PV,PCOST,SCOST,FID,CIP,CISS,ACLOT,ALUC,BP,BSS,SFP,SFSS

```

```

942      +ROTP,ROTSS,SFPT(25),SFSST(25),ROTPT(25),ROTSST(25),QM,QS,CWP(25)

```

```

943      +AVG,COST,PPCOST,SSCOST,PAVG,SAVG,YP,YS,SSFUC,PFUC,ASUCL,APUCL

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
  944      CHARACTER*12 FNAME
  945      INTEGER*2 IYEAR,IMONTH,IDAY
  946      REAL DISC(25)
  947      CALL MAKEFILE( FNAME, 2 )
  948      WRITE(2,1650)NLOT
  949 1650 FORMAT('ID : ',I2)
  950      WRITE(2,1660) (CIP(J),J=1,3)
  951 1660 FORMAT(' PRIME FIRST UNIT COST---MIN)',F8.0,2X,'MOST LIKELY)'
  952      + F8.0,2X,'MAX)',F8.0)
  953      WRITE(2,1670)(CISS(J),J=1,3)
  954 1670 FORMAT(' SECOND SOURCE-----MIN)',F8.0,2X,'MOST LIKELY)'
  955      + F8.0,2X,'MAX)',F8.0)
  956      WRITE(2,1680)(BP(J),J=1,3)
  957 1680 FORMAT(' PRIME PCURVE SLOPE-----MIN)',F5.3,5X,'MOST LIKELY)'
  958      + F5.3,5X,'MAX)',F5.3)
  959      WRITE(2,1690)(BSS(J),J=1,3)
  960 1690 FORMAT(' SECOND SOURCE-----MIN)',F5.3,5X,'MOST LIKELY)'
  961      + F5.3,5X,'MAX)',F5.3)
  962      WRITE(2,1700)
  963 1700 FORMAT(/' LOT LOT QUAN  SHIFT FACTOR',3X,'SHIFT FACTOR',3X,
  964      + ' ROTATION FACT',2X,'ROTATION FACT'/1X,'#',16X,'PRIME',7X,
  965      + 'SECOND SOURCE',5X,'PRIME',7X,'SECOND SOURCE'/5X,'MAX MIN',
  966      + 2X,4('MIN  M L  MAX  '))
  967      DO 1720 I=1,NLOT
1  968      WRITE(2,1710) I,QM(I),QS(I),(SFP(I,J),J=1,3),(SFSS(I,J),J=1,3),
1  969      + (ROTP(I,J),J=1,3),(ROTSS(I,J),J=1,3)
1  970 1710 FORMAT(I2,1X,F5.0,1X,F5.0,12(1X,F4.2))
1  971 1720 CONTINUE
  972      CLOSE(2)
  973      RETURN
  974      END

```

Name	Type	Offset	P	Class
ACLOT	REAL*8	2992	/RD	/
ALUC	REAL*8	3192	/RD	/
APUCL	REAL*8	3392	/RD	/
ASUCL	REAL*8	3592	/RD	/
AVG	REAL*8	*****		
BP	REAL*8	144	/RD	/
BSS	REAL*8	168	/RD	/
CIP	REAL*8	96	/RD	/
CISS	REAL*8	120	/RD	/
COST	REAL*8	*****		
CWP	REAL*8	7022		
DISC	REAL	7222		
FID	REAL*8	0	/RD	/
FNAME	CHAR*12	3792	/RD	/
I	INTEGER*4	7888		
IDAY	INTEGER*2	3812	/RD	/
IMONTH	INTEGER*2	3810	/RD	/
IYEAR	INTEGER*2	3808	/RD	/
J	INTEGER*4	7334		
NLOT	INTEGER*4	3804	/RD	/
PAVG	REAL*8	*****		
PCOST	REAL*8	*****		
PFUC	REAL*8	*****		
PPCOST	REAL*8	*****		
PV	REAL*8	*****		

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
QM    REAL*8      192    /RD    /
QS    REAL*8      392    /RD    /
ROTP  REAL*8      1792   /RD    /
ROTPT REAL*8      6822
ROTSS REAL*8      2392   /RD    /
ROTSST REAL*8      6622
SAVG  REAL*8      *****
SCOST REAL*8      *****
SFP   REAL*8      592    /RD    /
SFPT  REAL*8      6422
SFSS  REAL*8      1192   /RD    /
SFSST REAL*8      6222
SSCOST REAL*8      *****
SSFUC REAL*8      *****
YP    REAL*8      *****
YS    REAL*8      *****

```

```

975 C*****

```

```

976 C*

```

```

977 C* MODIFY-----THIS SUBROUTINE ALLOWS MODIFICATION OF AN EXISTING DATA
978 C* FILE. WHEN CALLED THE CURRENT DATA IS DISPLAYED ALONG
979 C* WITH LOCATOR REFERENCES A GIVEN DATA ELEMENT CAN BE
980 C* MODIFIED BY ENTERING THE LOCATOR WHICH WILL CAUSE THE
981 C* APPROPRIATE PROMPT TO BE GIVEN THIS ROUTINE WILL ALSO
982 C* ALLOW THE LOT NUMBER TO BE EXPANDED WITH CORRESPONDING
983 C* INPUT PROMPTS FOR THE EXPANDED LOT DATA IF THE LOT
984 C* NUMBER IS REDUCED EXISTING DATA WILL BE RETAINED AND
985 C* SIMPLY IGNORED WHEN THE SIMULATION IS RUN

```

```

986 C*

```

```

987 C*****

```

```

988 SUBROUTINE MODIFY

```

```

989 COMMON/RD/FID(12),CIP(3),CISS(3),BP(3),BSS(3),QM(25),

```

```

990 +QS(25),SFP(25,3),SFSS(25,3),ROTP(25,3),ROTSS(25,3),ACLOT(25),

```

```

991 +ALUC(25),APUCL(25),ASUCL(25),FNAME,NLOT,IYEAR,IMONTH,IDAY

```

```

992 REAL*8 PV,PCOST,SCOST,FID,CIP,CISS,ACLOT,ALUC,BP,BSS,SFP,SFSS,

```

```

993 +ROTP,ROTSS,SFPT(25),SFSST(25),ROTPT(25),ROTSST(25),QM,QS,CWP(25)

```

```

994 +,AVG,COST,PPCOST,SSCOST,PAVG,SAVG,YP,YS,SSFUC,PFUC,ASUCL,APUCL

```

```

995 CHARACTER*12 FNAME,ANS

```

```

996 INTEGER*2 IYEAR,IMONTH,IDAY

```

```

997 REAL DISC(25)

```

```

998 1730 WRITE(*,1740)

```

```

999 1740 FORMAT(/1X,'WHAT IS THE NAME OF THE FILE YOU WISH TO MODIFY')

```

```

1000 READ(*,1750) FNAME

```

```

1001 1750 FORMAT(A12)

```

```

1002 1770 CALL RREAD(NLN)

```

```

1003 NLOTP=NLOT+1

```

```

1004 WRITE(*,1780)NLOT

```

```

1005 1780 FORMAT(/1X,'(1) NUMBER OF LOTS ',12)

```

```

1006 WRITE(*,1790) (CIP(J),J=1,3)

```

```

1007 1790 FORMAT(1X,'(2) PRIME FIRST UNIT COST-MIN',F8,0.2X,'MOST LIKELY',

```

```

1008 +,F8,0.2X,'MAX',F8,0)

```

```

1009 WRITE(*,1800) (CISS(J),J=1,3)

```

```

1010 1800 FORMAT(' (3) SECOND SOURCE-----MIN',F8,0.2X,'MOST LIKELY',

```

```

1011 +,F8,0.2X,'MAX',F8,0)

```

```

1012 WRITE(*,1810) (BP(J),J=1,3)

```

```

1013 1810 FORMAT(' (4) PRIME PCURVE SLOPE---MIN',F5,3.5X,'MOST LIKELY',

```

```

1014 +,F5,3.5X,'MAX',F5,3)

```

```

1015 WRITE(*,1920) (BSS(J),J=1,3)

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2.00
1016 1820 FORMAT(' (5) SECOND SOURCE-----MIN--F5 3,5X,'MOST LIKELY')
1017      + F5 3,5X,'MAX--F5 3)
1018      WRITE(*,1830)
1019      WRITE(*,1835)
1020      WRITE(*,1837)
1021 1830 FORMAT(' (6) ' ' LOT LOT QUAN SHIFT FACTOR',3X,'SHIFT FACTOR',3X,
1022      + 'ROTATION FACT',2X,'ROTATION FACT')
1023 1835 FORMAT(1X,'*',16X,'PRIME',7X,
1024      + 'SECOND SOURCE',5X,'PRIME',7X,'SECOND SOURCE')
1025 1837 FORMAT(4X,'MAX MIN',
1026      + 2X,4('MIN M L MAX '))
1027      DO 1850 I=1,NLOT
1 1028      WRITE(*,1840) I,QM(I),QS(I),(SFP(I,J),J=1,3),(SFSS(I,J),J=1,3),
1 1029      + (ROTP(I,J),J=1,3),(ROTSS(I,J),J=1,3)
1 1030 1840 FORMAT(12,1X,F5 0,1X,F5 0,12(1X,F4 2))
1 1031 1850 CONTINUE
1032 1860 WRITE(*,1870)
1033 1870 FORMAT('/1X,'ENTER THE NUMBER IN THE ( ) THAT '
1034      + 'CORRESPONDS'/1X,'TO THE LINE YOU WANT TO MODIFY '
1035      + '/1X,'A ZERO (0) ABORTS MODIFICATION ')
1036 1875 FORMAT(I6)
1037      READ(*,1875)LINE
1038      IF (LINE EQ 0) RETURN
1039      IF ((LINE LT 1) OR (LINE GT 6)) GOTO 1860
1040      IF (LINE NE 1) GO TO 1890
1041      WRITE(*,1880)
1042 1880 FORMAT('/1X,'PLEASE INPUT THE NEW NUMBER OF LOTS')
1043      READ(*,1875)NLOTM
1044      IF (NLOTM GT NLOT) CALL CREATE(LINE,NLOTP,NLOTM)
1045      NLOT=NLOTM
1046      GO TO 1920
1047 1890 IF (LINE NE 6) GO TO 1910
1048      WRITE(*,1900)
1049 1900 FORMAT('/1X,'WHAT LOT NUMBER DO YOU WANT TO MODIFY')
1050      READ(*,1875)LOTN
1051      CALL CREATE(LINE,LOTN,LOTN)
1052      GO TO 1920
1053 1910 CALL CREATE(LINE,NLOT,NLOT)
1054 1920 WRITE(*,1930)
1055 1930 FORMAT('/1X,'ANY MORE CHANGES (Y,N) ?')
1056      READ(*,1940)ANS
1057 1940 FORMAT(A1)
1058      IF ((ANS EQ 'Y') OR (ANS EQ 'y')) GOTO 1860
1059      CALL RWRITE
1060 C**** M1=0
1061 C**** M2=0
1062 C**** M3=0
1063      RETURN
1064      END

```

Name	Type	Offset	P	Class
------	------	--------	---	-------

ACLOT	REAL*8	2992	/RD	/
ALUC	REAL*8	3192	/RD	/
ANS	CHAR*12	10129		
APUCL	REAL*8	3392	/RD	/
ASUCL	REAL*8	3592	/RD	/
AVC	REAL*8	*****		
BP	REAL*8	144	/RD	/

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
BSS  REAL*8      168  /RD  /
CIP   REAL*8      96  /RD  /
CISS  REAL*8     120  /RD  /
COST  REAL*8     *****
CWP   REAL*8     8740
DISC  REAL      8940
FID   REAL*8      0  /RD  /
FNAME CHAR*12    3792 /RD  /
I     INTEGER*4   9768
IDAY  INTEGER*2   3812 /RD  /
IMONTH INTEGER*2  3810 /RD  /
IYEAR INTEGER*2   3808 /RD  /
J     INTEGER*4   9144
LINE  INTEGER*4   9992
LOTN  INTEGER*4  10090
NLN   INTEGER*4   9102
NLOT  INTEGER*4   3804 /RD  /
NLOTM INTEGER*4  10040
NLOTP INTEGER*4   9106
PAVG  REAL*8     *****
PCOST REAL*8     *****
PFUC  REAL*8     *****
PPCOST REAL*8    *****
PV    REAL*8     *****
QM    REAL*8      192  /RD  /
QS    REAL*8      392  /RD  /
ROTP  REAL*8     1792  /RD  /
ROTPT REAL*8     8540
ROTSS REAL*8     2392  /RD  /
ROTSST REAL*8    8340
SAVG  REAL*8     *****
SCOST REAL*8     *****
SFP   REAL*8      592  /RD  /
SFPT  REAL*8     8140
SFSS  REAL*8     1192  /RD  /
SFSST REAL*8     7940
SSCOST REAL*8    *****
SSFUC REAL*8     *****
YP    REAL*8     *****
YS    REAL*8     *****

```

```
1065 C*****
```

```
1066 C*
```

```
1067 C* PAGE-----THIS SUBROUTINE KEEPS TRACK OF INDIVIDUAL OUTPUT
```

```
1068 C*          PAGES   THE ARGUMENT IS THE RUNNING PAGE NUMBER AND
```

```
1069 C*          LINE COUNT
```

```
1070 C*
```

```
1071 C*****
```

```
1072      SUBROUTINE PAGE(IPAGE,LC,TCK,IDEV)
```

```
1073      COMMON/RD/FID(12),CIP(3),CISS(3),BP(3),BSS(3),QM(25)
```

```
1074      +QS(25),SFP(25,3),SFSS(25,3),ROTP(25,3),ROTSS(25,3),ACLOT(25,
```

```
1075      +ALUC(25),APUCL(25),ASUCL(25),FNAME,NLOT,IYEAR,IMONTH,IDAY
```

```
1076      REAL*8 PV,PCOST,SCOST,FID,CIP,CISS,ACLOT,ALUC,BP,BSS,SFP,SFSS
```

```
1077      +ROTP,ROTSS,SFPT(25),SFSST(25),ROTPT(25),ROTSST(25),QM,QS,CWP,CSS
```

```
1078      +AVG,COST,PPCOST,SSCOST,PAVG,SAVG,YP,YS,SSFUC,PFUC,ASUCL,APUCL
```

```
1079      CHARACTER*12 FNAME
```

```
1080      INTEGER*2 IYEAR,IMONTH,IDAY
```

```
1081      IF (TCK EQ 1) GOTO 2005
```

IBM Personal Computer FORTRAN Compiler V2 00

```

D Line# 1      7
1082          WRITE(IDEV,2000)
1083 2000      FORMAT(79(1H-))
1084 2005      IF(IPAGE EQ 'E')RETURN
1085          IPAGE=IPAGE+1
1086          LC=2
1087 2006      FORMAT(1H,A6)
1088          WRITE(IDEV,2010) IMONTH, IDAY, IYEAR, IPAGE
1089 2010      FORMAT(1X, 'RUN DATE (MO/DY/YR) ---) ', I2, '/', I2, '/', I4,
1090          +10X 'PAGE ', I2)
1091          IF (IPAGE GT 1) GOTO 2015
1092          WRITE (IDEV,2013)
1093 2013      FORMAT(//)
1094          LC = LC + 2
1095 2015      RETURN
1096          END

```

Name	Type	Offset	P	Class
------	------	--------	---	-------

ACLOT	REAL*8	2992	/RD	/
ALUC	REAL*8	3192	/RD	/
APUCL	REAL*8	3392	/RD	/
ASUCL	REAL*8	3592	/RD	/
AVG	REAL*8	*****		
BP	REAL*8	144	/RD	/
BSS	REAL*8	168	/RD	/
CIP	REAL*8	96	/RD	/
CISS	REAL*8	120	/RD	/
COST	REAL*8	*****		
CWP	REAL*8	10946		
FID	REAL*8	0	/RD	/
FNAME	CHAR*12	3792	/RD	/
IDAY	INTEGER*2	3812	/RD	/
IDEV	INTEGER*4	12 *		
IMONTH	INTEGER*2	3810	/RD	/
IPAGE	INTEGER*4	0 *		
IYEAR	INTEGER*2	3808	/RD	/
LC	INTEGER*4	4 *		
NLOT	INTEGER*4	3804	/RD	/
PAVG	REAL*8	*****		
PCOST	REAL*8	*****		
PFUC	REAL*8	*****		
PPCOST	REAL*8	*****		
PV	REAL*8	*****		
QM	REAL*8	192	/RD	/
QS	REAL*8	392	/RD	/
ROTP	REAL*8	1792	/RD	/
ROTPT	REAL*8	10746		
ROTSS	REAL*8	2392	/RD	/
ROTSST	REAL*8	10546		
SAVC	REAL*8	*****		
SCOST	REAL*8	*****		
SFP	REAL*8	592	/RD	/
SFPT	REAL*8	10346		
SFSS	REAL*8	1192	/RD	/
SFSST	REAL*8	10146		
SSCOST	REAL*8	*****		
SSFUC	REAL*8	*****		
TCK	REAL	8 *		
YP	REAL*8	*****		

D Line# 1 7

IBM Personal Computer FORTRAN Compiler V2 00

YS REAL*8 *****

```

1097 C*****
1098 C*
1099 C* HEAD1-----THIS SUBROUTINE PRINTS THE HEADING ASSOCIATED WITH
1100 C*          SIMULATION LOT DATA RESULTS  IT IS CALLED WHENEVER
1101 C*          A PAGE BREAK OCCURS IN THE GIVEN PORTION OF THE
1102 C*          OUTPUT
1103 C*
1104 C*****
1105     SUBROUTINE HEAD1(IDEV)
1106     WRITE(IDEV,2020)
1107 2020 FORMAT(/1X,'PRIME SPLIT WIN',32X,'AVERAGE UNIT COSTS'/1X,
1108     + 'PERCENTAGE',17X,'AVERAGE LOT',2X,'-----',
1109     + '-----'/1X,'LOT #',3X,'% ',5X,'LOT QUANTITY',5X,'COST',7X,
1110     + 'PRIME',3X,'SECOND SOURCE',2X,'COMPOSITE'/1X,'-----',1X,'-----',
1111     + 2X,'-----',2X,'-----',2X,'-----',2X,
1112     + '-----',2X,'-----')
1113     RETURN
1114     END

```

Name	Type	Offset	P	Class
IDEV	INTEGER*4	0	*	

```

1115 C*****
1116 C*
1117 C* HEAD2-----THIS SUBROUTINE PRINTS THE HEADING ASSOCIATED WITH
1118 C*          THE DATA FILE DISPLAY  IT IS CALLED FOR EACH PAGE
1119 C*          BREAK
1120 C*
1121 C*****
1122     SUBROUTINE HEAD2(IDEV)
1123     WRITE(IDEV,2030)
1124 2030 FORMAT(/16X,'SHIFT FACTOR',3X,'SHIFT FACTOR',3X,'ROTATION FACT',
1125     + 2X,'ROTATION FACT'/1X,'LOT LOT QUAN',5X,'PRIME',7X,
1126     + 'SECOND SOURCE',5X,'PRIME',7X,'SECOND SOURCE'/2X,'# ',3X,
1127     + 'MAX MIN',2X,4('MIN M L MAX ')/1X,'-----',1X,'-----',
1128     + 4('-----'))
1129     RETURN
1130     END

```

Name	Type	Offset	P	Class
IDEV	INTEGER*4	0	*	

```

1131 C*****
1132 C*
1133 C* ENDP-----THIS SUBROUTINE PADS THE BOTTOM OF EACH PAGE WITH BLANK
1134 C*          LINES TO INSURE EACH PAGE IS PRINTED IN 8 1/2 X 11 INCH
1135 C*          FORMAT
1136 C*
1137 C*****
1138     SUBROUTINE ENDP(LC,TCK,ITT,IDEV)
1139     IF (TCK EQ 0) GOTO 2037
1140     WRITE(IDEV,2035)

```

IBM Personal Computer FORTRAN Compiler V2 00

```

D Line# 1      7
1141 2035 FORMAT(1H1)
1142      RETURN
1143 2037 LC=LC+1
1144      DO 2050 I=LC,67 2
1 1145      WRITE(IDEV,2040)
1 1146 2040 FORMAT(//)
1 1147 2050 CONTINUE
1148      RETURN
1149      END

```

Name	Type	Offset	P	Class
------	------	--------	---	-------

I	INTEGER*4	11886		
IDEV	INTEGER*4	12	*	
ITT	INTEGER*4	8	*	
LC	INTEGER*4	0	*	
TCK	REAL	4	*	

```

1150 C*
1151 C*
1152 C*****
1153 C*
1154 C*
1155 C*  GetFile ---- This routine is used to open previously existing
1156 C*  data files. If given a non-existent filename, GetFile will
1157 C*  prompt for a new filename until it gets an existing file.
1158 C*
1159 C*  GetFile gets the filename and channel passed as parameters
1160 C*  rather than through COMMON. It changes the COMMON filename
1161 C*  DON'T, DON'T, DON'T use it to change channel/unit numbers
1162 C*
1163 C*  NOTE the function FindFile CAN'T use the COMMON filename
1164 C*
1165 C*****
1166 C*
1167 C*
1168      SUBROUTINE GetFile( filename, channel
1169      CHARACTER*12 filename
1170      INTEGER*2 channel
1171 C*
1172 C*  local variables
1173      CHARACTER*12 TempNameString
1174      INTEGER*2 StringLength, ParseFilename
1175      LOGICAL*2 FindFile
1176 C*
1177 C*  begin
1178 100      StringLength = ParseFilename( filename
1179      TempNameString = filename
1180 C*  if the file exists
1181      IF FindFile( StringLength, TempNameString ) .NOT. .FALSE.
1182      GO TO 200
1183 C*  then open it
1184 110      OPEN( channel, FILE = TempNameString
1185      RETURN
1186 C*  else show the file has to be created
1187 200      WRITE( * , 1187 ) TempNameString
1188 110      FORMAT( ' The file ', A, ' does not exist.' )
1189      WRITE( * , 1189 )

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
1190 350      FORMAT( 1X, 'What file would you like to use?' )
1191          READ( *, 400 ) filename
1192 400      FORMAT( A12 )
1193 C*      check to see if this file is legit
1194          GOTO 100
1195          END

```

Name	Type	Offset	P	Class
------	------	--------	---	-------

CHANNE	INTEGER*2	4	*	
FILENA	CHAR*12	0	*	
FINDFI	LOGICAL*2			FUNCTION
PARSEF	INTEGER*2			FUNCTION
STRING	INTEGER*2	11894		
TEMPNA	CHAR*12	11896		

```

1196 C*
1197 C*
1198 C******
1199 C*
1200 C*
1201 C*      MakeFile ---- This routine is used to create data files
1202 C*      If the file already exists. MakeFile prompts for
1203 C*      permission to overwrite the existing file or use
1204 C*      a different filename
1205 C*
1206 C*      Filename and channel number are passed as parameters
1207 C*      rather than through COMMON. The COMMON filename will
1208 C*      be changed. DON'T change channel numbers!
1209 C*
1210 C*      NOTE: the function FindFile CAN'T use the COMMON filename
1211 C*
1212 C******
1213 C*
1214 C*
1215      SUBROUTINE MakeFile( filename, channel )
1216      CHARACTER*12 filename
1217      INTEGER*2 channel
1218 C*
1219 C*      Local variables
1220      CHARACTER*12 TempNameString
1221      CHARACTER*1 response
1222      INTEGER*2 StringLength ParseFilename
1223      LOGICAL*2 FindFile
1224 C*
1225 C*      begin
1226 100      StringLength = ParseFilename( filename )
1227      TempNameString = filename
1228 C*      if the file exists
1229      IF ( FindFile( StringLength, TempNameString ) ) GOTO 300
1230 C*      then ask permission to overwrite it
1231 C*      else it can be legitimately created
1232 100      OPEN ( channel, FILE = filename, STATUS = 'NEW' )
1233      RETURN
1234 C*      given the file exists, ask for permission to overwrite
1235 100      WRITE ( *, 400 ) filename
1236 400      FORMAT ( 1X, 'The file ', A12, ' already exists. ' )
1237      WRITE ( *, 450 )

```

```

D Line# 1      7      IBM Personal Computer FORTRAN Compiler V2 00
1238 450      FORMAT( 1X, 'Do you want to remake it? ( Y or N )' )
1239          READ( *, 500 ) response
1240 500      FORMAT( A1 )
1241 C*      if permission given then recreate the file
1242          IF ( ( response .EQ. 'Y' ) OR ( response .EQ. 'y' ) ) GOTO 200
1243 C*      else get a new filename
1244          WRITE( *, 600 )
1245 600      FORMAT( /1X, 'Type the new filename in ' )
1246          READ( *, 700 ) filename
1247 700      FORMAT( A12 )
1248 C*      check to see if this file is legit
1249          GOTO 100
1250          END

```

Name	Type	Offset	P	Class
------	------	--------	---	-------

CHANNE	INTEGER*2	4	*	
FILENA	CHAR*12	0	*	
FINDFI	LOGICAL*2	*****		
PARSEF	INTEGER*2	*****		
RESPON	CHAR*1	12165		
STRING	INTEGER*2	12058		
TEMPNA	CHAR*12	12060		

```

1251 C*
1252 C*
1253 C*****
1254 C*
1255 C*      ParseFilename scans the FilenameString for blanks to see how
1256 C*      long the string is      DO NOT MODIFY the string, ParseFilename
1257 C*      is effectively EQUIVALENCEing [sic] the filename in COMMON
1258 C*
1259 C*****
1260 C*
1261 C*
1262      INTEGER*2 FUNCTION ParseFilename( FilenameString )
1263      CHARACTER*1 FilenameString( 12 )
1264 C*
1265 C*      local variables
1266      INTEGER*2 i
1267      CHARACTER*1 space
1268 C*
1269 C*      local constants
1270      DATA space / ' ' /
1271 C*
1272 C* begin
1273 C*
1274      DO 100 i = 1, 12
1 1275 C*      if character is an ascii space,
1 1276      IF ( FilenameString( i ) .NE. space ) GOTO 100
1 1277 C*      then end-of-string is found
1 1278      ParseFilename = i - 1
1 1279      RETURN
1 1280 C*      else keep looking
1 1281 100      CONTINUE
1282      ParseFilename = 12
1283 200      RETURN
1284      END

```

D Line# 1 7

IBM Personal Computer FORTRAN Compiler V2 00

Name	Type	Offset	P	Class
------	------	--------	---	-------

FILENA	CHAR*1	0	*	
I	INTEGER*2	12322		
SPACE	CHAR*1	12320		

```

1285 C*
1286 C*
1287 C*****
1288 C*
1289 C*    LOGICAL*2 FUNCTION FindFile( StringLength, FilenameString )
1290 C*    INTEGER*2 Stringlength
1291 C*    CHARACTER*12 FilenameString
1292 C*
1293 C*    is external to this source file because it was written
1294 C*    in Assembler
1295 C*
1296 C*    NOTE.  the parameters passed to it CAN'T be in COMMON
1297 C*
1298 C*****
1299 C*
1300 C*

```

Name	Type	Size	Class
A		40808	COMMON
CREATE			SUBROUTINE
ENDP			SUBROUTINE
FINDFI	LOGICAL*2		FUNCTION
GETDAT			SUBROUTINE
GETFIL			SUBROUTINE
HEAD1			SUBROUTINE
HEAD2			SUBROUTINE
LINE			SUBROUTINE
MAIN			PROGRAM
MAKEFI			SUBROUTINE
MODIFY			SUBROUTINE
PAGE			SUBROUTINE
PARSEF	INTEGER*2		FUNCTION
PLOTA			SUBROUTINE
RANDOM	REAL*8		FUNCTION
RD		3814	COMMON
RREAD			SUBROUTINE
RWRITE			SUBROUTINE
SCALE			SUBROUTINE

Pass One No Errors Detected
 1300 Source Lines

APPENDIX D

CDAPM

TECHNICAL REFERENCE

APPENDIX D

CDAPM TECHNICAL REFERENCE

1. The Competition Decision-Assist Package for the Microcomputer (CDAPM) was written in FORTRAN for the IBM Personal Computer FORTRAN Compiler Version 2.0.
2. To get results as accurate and consistent as possible, CDAPM uses the 8087-math options built into the compiler. Since only a few machines in the field have 8087s or the like, CDAPM will emulate the 8087 when the chip is not there. Emulation costs execution speed, but APRO feels that consistency in CDAPM is more important than speed.
3. CDAPM has been tried out on several different makes of PC; CDAPM works on all the computers it was tried on. APRO does not mean to slight any manufacturer whose computer is not listed below; the machines listed below are simply those that were available around Fort Lee, VA for APRO to use.

<u>Computer</u>	<u>Run Time (1001 Cycles)</u>
IBM PC w/PC-DOS 2.1	44 min.
Wyse 1100-1 PC w/MS-DOS 2.11	43 min.
Compaq Plus w/MS-DOS 2.11	49 min.
Compaq Plus w/MS-DOS 2.11; w/8087	7 min.
Compaq Deskpro w/MS-DOS 3.0	41 min.
Compaq Deskpro w/MS-DOS 3.0; 8 MHz	23 min.
Tandy 1000 w/MS-DOS 2.1	52 min.
Leading Edge PC w/PC-DOS 2.1	33 min.
Leading Edge PC w/MS-DOS 2.11	30 min.

These are not intended to be definitive benchmarks; in several cases, background processes were running, but these time tests generally indicate what to expect. Users should note the big difference in "old" and "new" processor chip makes.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER 85-06	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Competition Decision-Assist Package for the Microcomputer		5. TYPE OF REPORT & PERIOD COVERED Final
7. AUTHOR(s) Monte G. Norton and Fred A. Frye		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS Office of the Deputy Chief of Staff for Logistics US Army Procurement Research Office Fort Lee, VA 23801-6045		8. CONTRACT OR GRANT NUMBER(s)
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army Materiel Command ATTN: AMCPP 5001 Eisenhower Ave., Alexandria, VA 22333-0001		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE February 1987
		13. NUMBER OF PAGES 75
		15. SECURITY CLASS. (of this report) Unclassified
		16a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Unclassified, Unlimited Distribution		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Competition, Decision Model		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) (III) Project Managers must assess the extent to which their system ought to be competed. The objective of this study was to develop, test, and operate a microcomputer version of the Competition Decision-Assist Package (CDAP). CDAPM provides a microcomputer FORTRAN version of CDAP to assist Program Managers in their competition decisions.		

END

4-87

DTIC