U. S. AIR FORCE

PROJECT RAND

DOCUMENT

SMAC MANUAL

M. I. Bernstein

D-5279

June 17, 1958

Assigned to _____

ACKNOWLEDGMENT

The author wishes to acknowledge his indebtedness to J. I. Derr whose JOHNNIAC Floating Point Interpretive System serves as the foundation upon which SMAC was built.

M.I.B.

SUMMARY

SMAC is a rather unsophisticated system for stating a limited class of problems for numerical solution on the RAND JOHNNIAC. The few elements of sophistication present were incorporated to enable the user to state his problems with ease.

This report was prepared with two views in mind:

1. To serve as one of the texts for the programming courses being given by the Numerical Analysis Department (starting April 8, 1958).
2. To serve as a complete manual from which the interested reader can learn to use the system without necessarily taking a formal course.

CONTENTS

APPENDICES

SMAC — A SMALL COMPILER

M. I. Bernstein

1. THE SIMPLE SYSTEM

1.0. Introduction

SMAC (for "SMALL Compiler") is a system for stating problems for numerical solution on the JOHNNIAC. The class of problems for which SMAC was designed is limited, in the main, to algebraic ones. The system embodies a small amount of sophistication so that this limited class of problems may be stated with ease.

2.0 Form of Algebraic Statements

The algebraic statements that may be written directly in SMAC language are limited, for the most part, by what one can write unambiguously without the use of parentheses.

Example 1

$a + bx + cx^2$ is an unambiguous statement.

Example 2

$a + b/c + d$ might mean

1. $\left[(a + b)/c\right] + d$

2. $(a + b)/(c + d)$ etc.

Rule 1

In algebraic statements all multiplications are done, then all divisions, then all additions and subtractions.

Thus, in SMAC, Example 2 would result in:

$$a + (b/c) + d.$$

2.1 Explicit Statement

SMAC provides for variables whose "names" may be up to four characters long. Thus, all multiplications must be explicitly stated (see example). Due to the limitations of key-punch equipment, the multiplication sign is an asterisk (*).+

Example 3

AB + C * D + E/F * G means

$AB + (C)\cdot(D) + \left[E/((F)\cdot(G))\right]$ where

AB $\equiv$ the variable whose name is "AB".

Note

All succeeding examples of statements in this chapter are illustrated in Appendix 1. We suggest that the reader separate this appendix from the rest of the paper for easy reference.

2.2 Fixed Format

The SMAC statement sheet allows for up to 6 operands and 5 operations per statement. Fixed format means that the i^{th} operand always occurs in the same place on the sheet.

Example 4 shows how Example 3 appears when written on the statement sheet.

+Throughout the remainder of this report, * will be used when writing SMAC statements and · will be used when writing equations to indicate multiply.

Rule 2

The first character of the i^{th} operand must be written at the left margin of the i^{th} field and must be alphabetic. Succeeding characters may be alphabetic or numeric.

3.0 Writing Equations

Typical algebraic problems are often stated somewhat as follows:

$$a = f_1 (x, y, z, ...)$$

$$b = f_2 (a, x, y, z, ...)$$

$$c = f_3 (a, b, x, y, z, ...)$$

For example:

$$a = x \cdot y$$

$$b = a^2 + a \cdot z$$

$$c = a \cdot x + (a \cdot y/b) + b \cdot z^2$$

Example 5 shows a way of writing these equations in SMAC. Note that the expression for C will not fit into one SMAC statement. Therefore, it is broken up into two statements, the first of which is merely an intermediate result which we have arbitrarily called "INTM." Any other name, other than A, B, C, X, Y, or Z would have done as well. (Experience indicates that it is wise to be systematic in assigning names to intermediate results; the most commonly used system is R0, R1, R2, ...)

3.1 Functions

SMAC has the following "built-in" functions:

SIN, CØS, ATAN (arc tangent) - radians

LØG, EXP (exponential) - base e

SQRT (square root)

ABS (absolute value of)

NEG (negative of)

INT (integer part of)

When a function is used, the remainder of the right-hand side of the equation is treated as if it were parenthesized. For example:

$$y_0 = \sin (a + b \cdot x)$$

$$y_1 = \log |y_0|$$

$$y_2 = - (a \cdot y_0 + b \cdot y_1)$$

are written as SMAC statements in Example 6. Note the need for the intermediate result, R0.

4.0 Decision Making

As should be apparent from Example 5 and Example 6, SMAC performs the algebra of each statement and then proceeds to the next statement in order. In each case the variable evaluated in one statement (the left-side of the =) is available for use in subsequent statements.

In practice, it often occurs that there is a choice of several formulae (say, for evaluating a variable) dependent upon a criterion. For example:

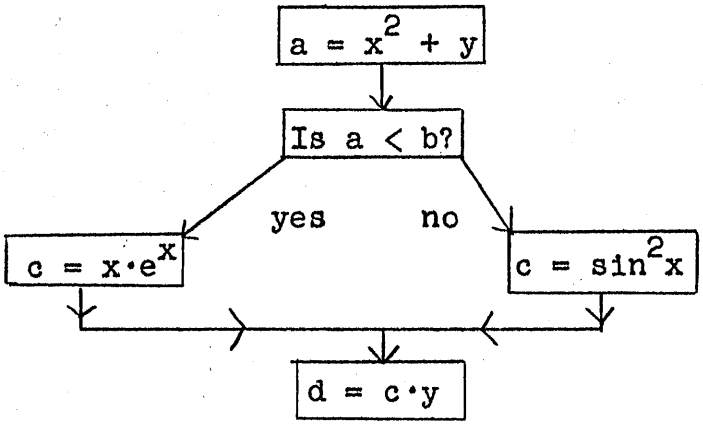
$a = x^2 + y$

if $a < b$, $c = x \cdot e^x$

if $a \geq b$, $c = \sin^2 x$

$d = c \cdot y$

Schematically, this example may be represented as follows:
(In computer parlance this is called a "flow diagram")



4.1 Statement Names

In order to do the example of section 4.0, we must be able to do several things.

1. Make comparisons.
2. Deviate from going from one statement to the next.

The second requirement implies that the statements must have "names" so that they may be referred to. In SMAC, a statement is named by the variable being evaluated. Thus, in Example 6, Y0, R0, Y1, Y2 serve dual purposes:

a. They are the names of the variables being evaluated.

b. They are the names of the statements doing the evaluation.

4.2 The Instruction GØ TØ

This instruction tells SMAC that the next statement to be performed is the one whose name is given.

4.3 The Instructions IS, YES, NØ

1. IS: This instruction sets a criterion. It compares two variables for one of three conditions.
   a. G (greater than)
   b. = (equal to)
   c. L (less than)
2. YES: If the criterion of the IS instruction has been met, this instruction tells SMAC what statement to perform next.
3. NØ: If the criterion of the IS instruction has not been met, this instruction tells SMAC what statement to perform next.

Rule 3

The instructions IS, YES, NØ may occur in the order IS, YES, NØ or IS, NØ, YES. However, there may not be any instructions or statements separating them from one another. (Also see Section 4.5)

4.4 An Example Involving Decision Making

Example 7 is a way of doing the problem of Section 4.0. The statements have been numbered in the left margin to

expedite following the flow:

1. Computes $a = x^2 + y$
2. Asks if a is less than b
3. If $a < b$ proceed to RO (5)
   - 5. Computes $RO = e^x$
   - 6. Computes $c = x \cdot e^x$
   - 7. Says proceed to D (10)
   - 10. Computes $d = c \cdot y$
4. If $a \geq b$ proceed to R1 (8)
   - 8. Computes $R1 = \sin x$
   - 9. Computes $c = \sin^2 x$
   - 10. Computes $c \cdot y$

Example 8 is a second way of doing the same problem: The YES and NØ have been interchanged.

4.5 The Implied YES or NØ

In SMAC, if the answer to an IS is not YES, then it must be NØ; if it is not NØ, then it must be YES. In Example 8, note that the YES (line 4) proceeds to RO (line 5). In this case, the YES statement could have been omitted and statement RO would be an implied YES. On the other hand, in Example 7 the NØ (line 4) could not be omitted, since RO (line 5) is not an implied NØ (it is the statement that is proceeded to from the YES).

4.6 "Dummy" Statement Names

Let us attempt to write SMAC statements to do the

following example:

$a = x^2 + y$

if $a < b$ $\quad c = \sqrt{x}$

if $a \geq b$ $\quad c = \sin x$

$d = c \cdot y$

Example 9 is an attempt to do this example. However, on line 3 we want to proceed to line 5, whose name is C; on line 4 we want to proceed to line 7, whose name is also C. The JOHNNIAC just isn't that smart! (If C replaced the question marks on lines 3 and 4, line 5 would result; where choices of this kind exist, SMAC picks the first occurrence.)

A solution to this dilemma is to introduce SMAC statements which are merely names; no evaluations are performed. Example 10 is a way to do the example of this section. The YES (line 3) proceeds to DUM (line 6) which does nothing and proceeds to the next statement in order. Note the implied NØ on line 4.

4.7 Expanded Definition of the IS

In the example of Section 4.6 the variable A was evaluated only to compare it against B; A was not needed for other computation. The IS in this case (Example 10) was used to compare A and B. The IS may also be used to compare a variable against the value of an expression.

First, let us restate the example of Section 4.6:

$$\text{If } b > x^2 + y \qquad c = \sqrt{x}$$

$$\text{if } b \leq x^2 + y \qquad c = \sin x$$

$$d = c \cdot y$$

Example 11 is a way to do this example.

5.0 The Instructions READ and PRINT

a. Referring to the example of Section 4.7, it is obvious that we can do no computing unless we have the numbers b, x, y in the machine. In addition, the computation doesn't do us much good unless the quantity d can be obtained from the machine.

In SMAC, all numbers must be input from cards. The format of the numbers is a 9-digit fraction and a two-digit power of 10.

Example 12 is an input sheet for:

B = 125.03

X = 17.2

Y = .003

EØF (line 4) stands for "End of File" and tells SMAC that this is the end of the current set of input data. The remainder of the EØF card is ignored.

b. READ: Instructs SMAC to input numbers from cards. When an EØF card is reached, proceed to the next statement. The READ statement may have a statement name so that it can be referred to from a YES, NØ, or GØ TØ statement.

c. Output from SMAC is in the form of printed lists.

The lists contain up to six numbers per line. The format of these numbers is identical to that of the input numbers. The instruction, PRINT, tells SMAC to list the variable whose name is given in the i^{th} column. Any blank operands will result in blank columns. The PRINT statement may have a statement name.

5.1 An Example

Compute:

$$y = 2.5\ x^2 + 750.3x + .075$$

for $x = .5$ and $x = .003$; list x and y

Example 13 shows the data forms for this computation.

Example 14 shows a way to write the SMAC statements for this problem. Note that line 7 introduces a new instruction, STØP: This operation stops the JOHNNIAC. If the start button is hit, proceed to the named statement (if no name is given, "stop dead"). Let us follow the example:

line 1. Inputs A, B, C, and X = .5
2. Computes Y
3. Lists X in column 1, and Y in column 3.
4. Inputs X
5. Computes Y
6. Prints X and Y
7. Stops. If more values of X are desired, proceeds to line 4.

6.0 Format Control of the Listing

a. HEDING: This instruction prints the 6 characters written in the i^{th} operand as a heading for the i^{th} column of the list. Blanks are legal characters and are written on the statement sheet as a "donut" ◎, when confusion might result (see Example 15).

The HEDING instruction may have a statement name so that it can be referred to from a YES, NØ, or GØ TØ

b. SPACE: Space one line on the printer. This instruction may have a statement name.

c. EJECT: Skip to the top of the next page (start a new page). This instruction may have a statement name.

7.0 The START Instruction

The last SMAC statement for any problem must be a START instruction. This instruction tells SMAC to begin computation with the SMAC statement whose name is given.

7.1 The Title Card

Each SMAC problem must be preceded by at least one title card. The total number of title cards is not limited. However, only the last title card may have any information in column 80 (the extreme right of the title form). The last card must have a + in column 80.

7.2 A Complete Problem

Example 16, Example 17, and Example 18 are respectively:

the title cards, SMAC statements, and input data for the example of Section 5.1. They would be entered into the JOHNNIAC in the given order.

8.0 Handwriting

One of the difficulties of preparing manuscripts for keypunching (as opposed to reports for typing) is that we are usually not writing sentences from which the keypunch operator can make reasonable guesses from context. One keypunch error will almost always result in wrong answers.

8.1 Some Conventions

At RAND, the following conventions hold for keypunch manuscripts:

Number	Letter
0	Ø
1	I
2	Z

In addition,  = blank

9.0 Some Restrictions

a. The number of SMAC statements for any one problem may not exceed 140. SMAC will tell you if you have violated this rule.

b. Problems of less than 140 statements may, under complex conditions, exceed capacity. SMAC will inform you if this is the case.

c. The number of variables (including "intermediate results") that may be introduced into one SMAC problem may not exceed 1200. SMAC will tell you if you have violated this rule.

d. The detection of the above and other errors is described in Appendix 5.

2. SUBROUTINES

10.0 Introduction

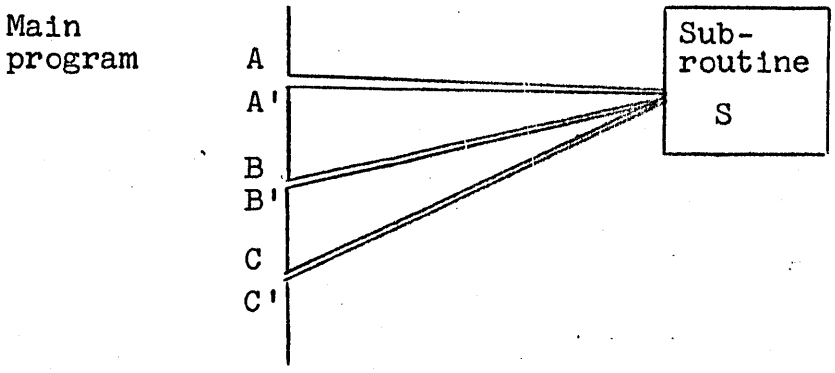
There are two basic reasons for using subroutines when preparing problems for solution on a computer:

a. They allow savings in writing (and, hence, clerical errors) and/or in specifying oft-repeated sequences of instructions. For example, the functional operations (SQRT, SIN, etc.) are used so much that they have been incorporated directly into SMAC. They are, in fact, subroutines within the system.

b. They aid in problem organization, checkout, and change.

10.1 A Diagram

The following diagram is a representation of what subroutine usage looks like:



At points A, B, C in the main program we wish to perform the computation in subroutine S and then proceed to A', B', C'

respectively. Another way of stating the same thing is to say that at A, B, C we wish to "enter" subroutine S, which subroutine, upon "exit," will return to A', B', C' respectively.

11.0 Naming a Subroutine

In SMAC a subroutine is named at its entry point. The instruction ENTRY gives the subroutine its name.

E.g., ENTRY F(X) tells SMAC that the succeeding SMAC statements are the subroutine F(X). The statements for F(X) are terminated by the instruction EXIT. The instruction ENTER tells SMAC to proceed to the subroutine whose name is given.

11.1 The instructions ENTER, ENTRY, EXIT

a. ENTER: proceed to the subroutine whose name is given. (Upon EXIT, the subroutine will return to the statement immediately following the ENTER.)

b. ENTRY: give this subroutine the specified name.

c. EXIT: return to the SMAC statement immediately following the last ENTER to this subroutine. The EXIT instruction may have a statement name.

Rule 5

There may not be any other ENTRY or EXIT between the ones which specify the statements for a given subroutine. There may, however, be ENTER's. (See Section 12.0).

11.2 An Example

Example 19 (Appendix 2) shows a set of SMAC statements to compute

$$y = f(a) + f(b) + f(c)$$

where $f(x) = \sin x + x^2 + xe^x$.

12.0 Cascading Subroutines

Example 20 is a way of coding:

Compute $y = f(a,b) + f(a,d)$

$$f(x,z) = \left[x^2/g(z)\right] + g(z)$$

$$g(z) = z^2 + \ell n\ z$$

12.1 A Note

The name used in an ENTRY instruction must be unique (i.e., it may not be used to name any other SMAC statements).

$$\delta(x,z) = \frac{x^2}{z^2 + \ln z} + z^2 + \ln z$$

3. COUNTERS AND SUBSCRIPTS

20.0 Introduction

The reader will have noted that in the examples for Chapters 1 and 2 the last two columns of (1) the variable field and (2) each of the 6 operand fields were never used. These columns are reserved for specifying subscripts and, conversely, when subscripts are used, they must appear in these columns.

Rule 6

Subscripts, when they appear alone, must be single alphabetic characters and must be written in the right-most column of the subscript field.

Rule 7

Subscripts, when they appear alone, do not "name" a SMAC statement.

Rule 8

In any one problem, a maximum of 6 different subscripts may be used.

20.1 Numbers and Subscripts

When numbers are used in association with subscripts, they are written directly on the coding sheet - in the subscript columns (see Example 21 - Appendix 3).

21.0 The SET Instruction

Before a subscript may be used in any SMAC statement it must have been initiated by a SET instruction (see Example 22). The SET instruction may not have a statement name.

21.1 The DEFINE Instruction

Whenever a subscript is used, its cardinality must be specified by a DEFINE instruction once at the beginning of the program*(see Example 23). By "cardinality" we mean its highest value + 1 (since, in SMAC, we count from zero). Thus in Example 23, Z = 0, 1, 2,...,9.

---

* Normally, the DEFINE cards occur immediately after the title cards.

22.0 Subscripts as Counters

In computing, it is often necessary to execute a portion of the code a fixed number of times. For example, in Newton's Method for iterating:

$$x_{n+1} = x_n - f(x_n)/f'(x_n),$$

quite often, we limit the number of iterations rather than depend upon convergence criteria. Let's take the case:

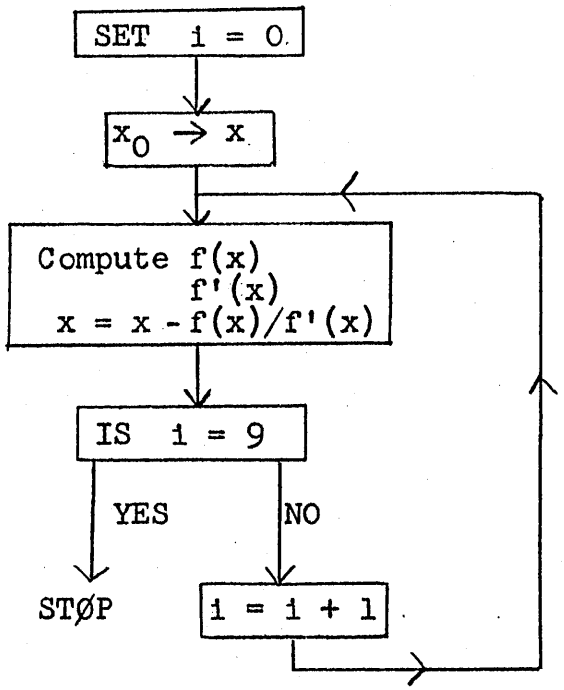
$f(x) = x \sin x - a = 0$

$f'(x) = x \cos x + \sin x$

initial guess = x_0

compute for 10 iterations.

22.1 The Flow Diagram



Example 24 shows a way to code this problem.

23.0 Arrays

When a subscript is appended to a variable (or operand), the variable is treated as a linear array and the value of the subscript determines the position in the array. In this case, the first column of the subscript field must be a comma (see Example 25).

23.1 An Example Using Subscripts

Example 26 is a way to code

$$x_i = k(y_i + z_i) \qquad i = 0(1)13$$

Note that the GØ TØ statement names X ,I as the statement to proceed to.

Rule 9

The name of a statement includes the subscript field. X ,I is different than X ,J (also see Rule 7).

23.2 "Definitive" Subscripts

It is often desired to pick a specific element from an array. In SMAC, this is done by appending a numeric subscript to the variable. Example 27 is a way to write the statement for

$$y_i = y_o + x_j$$

Appendix 4 contains a non-trivial example which uses most of the features of SMAC.

4. PROBLEM CHECK-OUT

31.0 Wha' hoppen?

Let us assume a problem has been coded in SMAC and the first run has been made on the JOHNNIAC. One of three things usually occurs:

a. Correct answers.

b. Wrong answers.

c. No answers.

"What do I do now?"

31.1 Range of Numbers

In SMAC all numbers must be in the range of $10^{-50} \leq X < 10^{50}$.

a. If $X < 10^{-50}$ we say an underflow has occurred.

b. If $X \geq 10^{50}$ we say an overflow has occurred.

If, during computation, the result of any operation produces an underflow or overflow SMAC will print ARITH,UNFLOW or ARITH,OVFLOW, respectively, and stop. SMAC will not inform you in which statement the out of range number occurred.

31.2 Other Numerical Errors

SMAC will inform you in the following way when the below-listed errors occur (but not in which statement they occurred) and stop.

Error	SMAC prints
Y/X, X = 0	DIV BY ZERO
EXP $X \geq 10^{50}$	EXP OVFLOW

Error	SMAC prints
EXP $X < 10^{-50}$	EXP UNFLOW
LOG X, $X < 0$	LØG, - ARG.
LOG X, $X = 0$	LØG, 0 ARG.
SQRT X, $X < 0$	SQRT, - ARG.
SIN, X, $X \geq \pm 10^{10}$	SIN MAX ARG.
COS X, $X \geq \pm 10^{10}$	CØS MAX ARG.
Others (?)	UNDETERMINED

32.0 What to do

Since, in general, SMAC informs you only of the fact that an error has occurred, rather than where the error is, we need to develop other devices to isolate mistakes. The remainder of this chapter is concerned with several "debugging" techniques which the user should find of some help. Certainly, these techniques are not all-inclusive; as one gains experience with the system, new devices for locating errors will be discovered.

32.1 Special Use of HEDING

Assume the machine stops and that all that has been printed are the title cards and the reason for stopping. We should like to know (approximately) which SMAC statement was being executed. By putting HEDING instructions with, say, A1, A2, A3,...., in the deck at strategic places we should be able to determine bounds for where the error lies. Since A_i will have printed and A_{i+1} will not, the error is somewhere between these two.

32.2 Breakpointing

In a manner identical to that of section 32.1, one can insert PRINT instructions at strategic places, to list values of critical variables.

32.3 The Post-mortem Dump

There exists a SMAC dump routine to print the names and values of every variable in the problem. Usually, an inspection of these quantities will supply an insight into the cause of the stop. A useful device to be used in conjunction with the dump is the creation of a "dummy" variable, say, R, which is incremented at strategic places in the code. By inspecting its value at the time of the dump, one can determine approximately where the difficulty arose.

CMAC STATEMENT

APPENDIX I

	INST	VAR OR NAME	G = L	FUNC	CP1	+ − * /	CP2	+ − * /	CP3	+ − * /	CP4	+ − * /	CP5	+ − * /	CP6	IDENT
Ex. 4					AB	+	C	*	D	+	E	/	F	*	G	
Ex. 5		A	=		X	*	Y									
		B	=		A	*	A	+	A	*	Z					
		INTM	=		A	*	X	+	A	*	Y	/	B			
		C	=		INTM	+	B	*	Z	*	Z					
Ex. 6		Y0	=	SIN	A	+	B	*	X							
		R0	=	ABS	Y0											
		Y1	=	LOG	R0											
		Y2	=	NEG	A	*	Y0	+	B	*	Y1					
Ex. 7 1		A	=		X	*	X	+	Y							
2	IS	A	L		B											
3	YES	R0														
4	NO	R1														
5		R0	=	EXP	X											
6		C	=		X	*	R0									
7	GO TO	D														
8		R1	=	SIN	X											
9		C	=		R1	*	R1									
10		D	=		C	*	Y									

APPENDIX I

CMAC STATEMENT

	INST	VAR OR NAME	G = L	FUNC	CP1	+−*/	CP2	+−*/	CP3	+−*/	CP4	+−*/	CP5	+−*/	CP6	IDENT
Ex. 8																
1		A	=		X	*	X	+	Y							
2	IS	A	L		B											
3	NØ	R1														
4	YES	RO														
5		RO	=	EXP	X											
6		C	=		X	*	RO									
7	GØ TØ	D														
8		R1	=	SIN	X											
9		C	=		R1	*	R1									
10		D	=		C	*	Y									
Ex. 9																
1		A	=		X	*	X	+	Y							
2	IS	A	L		B											
3	YES	?														
4	NØ	?														
5		C	=	SQRT	X											
6	GØ TØ	D														
7		C	=	SIN	X											
8		D	=		C	*	Y									

6-17-58
+27-

APPENDIX I

CMAC STATEMENT

	INST	VAR OR NAME	G = L	FUNC	OP1	+-*/	OP2	+-*/	OP3	+-*/	OP4	+-*/	OP5	+-*/	OP6	IDENT
Ex. 10,		A	=		X	*	X	+	Y							
2	IS	A	L		B											
3	YES	DUM														
4		C	=	SIN	X											
5	GO TO	D														
6		DUM														
7		C	=	SQRT	X											
8		D	=		C	*	Y									
Ex. 11	IS	B	G		X	*	X	+	Y							
	YES	DUM														
		C	=	SIN	X											
	GO TO	D														
		DUM														
		C	=	SQRT	X											
		D	=		C	*	Y									

6-17-58
-28-

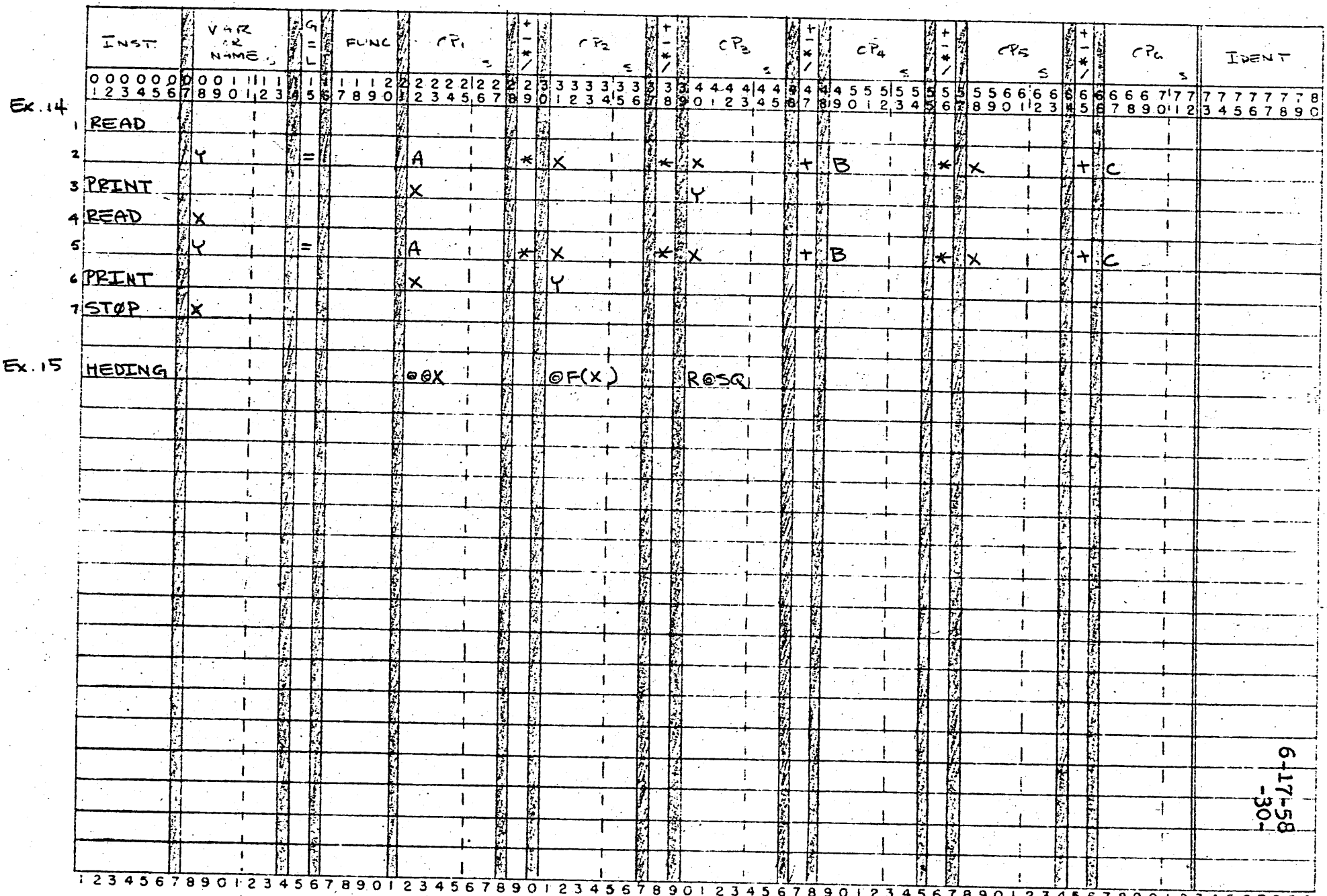
APPENDIX I

SMAC DATA FORM

	INIT (EXF)	NAME	=	NUMBER: ±	.	FRACTION	±	P	COMMENTS	IDENT
Ex. 12 1		B	=	+	.	12503	+	03		
2		X	=	+	.	172	+	02		
3		Y	=	+	.	3	-	02		
4	EØF									
Ex. 13		A	=	+	.	25	+	01		
		B	=	+	.	7503	+	03		
		C	=	+	.	75	-	01		
		X	=	+	.	5	+	00		
	EØF									
		X	=	+	.	3	-	02		
	EØF									

APPENDIX I

CMAC STATEMENT



APPENDIX I

SMAC TITLE FORM

NOTE: (+ IN COL 80 OF LAST LINE)

TITLE	IDENT	+
Ex. 16 SMAC ILLUSTRATIVE PROBLEM		
Y = A(X)X(X) + B(X) + C		

APPENDIX I

CMAC STATEMENT

Ex. 17

INST	VAR OR NAME	=	FUNC	OP1	+-*/	OP2	+-*/	OP3	+-*/	OP4	+-*/	OP5	+-*/	OP6	IDENT
READ	COEF														
EJECT															
HEDING				@@A		@@B		@@C							
SPACE															
PRINT				A		B		C							
SPACE															
SPACE															
HEDING				@@X				@@Y							
SPACE															
READ															
	Y	=		A	*	X	*	X	+	B	*	X	+	C	
PRINT				X				Y							
READ	X														
	Y	=		A	*	X	*	X	+	B	*	X	+	C	
PRINT				X				Y							
STOP	X														
START	COEF														

APPENDIX I

SMAC Data Form

Ex. 18

INST (EQF)	NAME	S	=	NUMBER t	.	FRACTION	±	P	COMMENTS	IDENT
	A		=	+	.	25	+	01		
	B		=	+	.	7503	+	03		
	C		=	+	.	75	−	01		
EQF										
	X		=	+	.	5	+	00		
EQF										
	X		=	+	.	3	−	02		
EQF										

APPENDIX 2

EXAMPLE 19

INST	NAME		FUNC	OP1	OP2	OP3	OP4	OP5	OP6
				•		•		•	
				•		•		•	
	X	=		A					
ENTER	EVAL								
	R0	=		F(X)					
	X	=		B					
ENTER	EVAL								
	R0	=		R0	+ F(X)				
	X	=		C					
ENTER	EVAL								
	Y	=		R0	+ F(X)				
				•		•		•	
				•		•		•	
ENTRY	EVAL								
	R1	=	SIN	X					
	R2	=	EXP	X					
	F(X)	=		R1	+ X	* X	+ X	* R2	
EXIT									
				•		•		•	
				•		•		•	

EXAMPLE 20

INST	NAME		FUNC	OP1	OP2	OP3	OP4	OP5	OP6
				•		•		•	
				•		•		•	
	X	=		A					
	Z	=		B					
ENTER	F								
	R0	=		FXZ					
	Z	=		D					
ENTER	F								
	Y	=		R0	+ FXZ				
				•		•		•	
				•		•		•	
ENTRY	F								
ENTER	G								
	FXZ	=		X	* X	/ G(Z)	+ G(Z)		
EXIT									
ENTRY	G								
	R1	=	LOG	Z					
	G(Z)	=		Z	* Z	+ R1			
EXIT									
				•		•		•	
				•		•		•	

APPENDIX 3

EXAMPLE 21

```
INST    NAME     FUNC  OP1        OP2       OP3       OP4       OP5       OP6

              I =           I +      01
IS            I =           24
```

EXAMPLE 22

```
INST    NAME     FUNC  OP1        OP2       OP3       OP4       OP5       OP6

SET           I =           00
SET           J =           10
SET           K =           99
SET           V =           00
```

EXAMPLE 23

```
INST    NAME     FUNC  OP1        OP2       OP3       OP4       OP5       OP6

DEFINE        Z =           10
```

EXAMPLE 24 ?

```
INST    NAME     FUNC  OP1        OP2       OP3       OP4       OP5       OP6

                  .                    .                    .
                  .                    .                    .
DEFINE        I =           10
SET           I =           00
        X         =     X0
        DUM1
ENTER   EVAL
IS            I =           09
NO      DUM2
                  .                    .                    .
                  .                    .                    .
        DUM2
              I =           I +      01
GO TO   DUM1
ENTRY   EVAL
        R0        = SIN  X
        R1        = COS  X
        F         =      X     * R0      - A
        FP        =      X     * R1      + R0
        X         =      X     - F       / FP
EXIT
                  .                    .                    .
                  .                    .                    .
```

how do you get out if I = 09
After is statement wouldn't it go to ([illegible]

APPENDIX 3

EXAMPLE 25

INST	NAME	FUNC	OP1	OP2	OP3	OP4	OP5	OP6
	X ,I =	SIN	Y ,I +	A	/ Z ,J			

EXAMPLE 26

INST	NAME	FUNC	OP1	OP2	OP3	OP4	OP5	OP6
		•		•		•		
		•		•		•		
DEFINE	I		14					
SET	I =		00					
	X ,I =	K		* Y ,I + K		* Z ,I		
IS	I =		13					
YES	•••							
	I =		I +	01				
GO TO	X ,I							
		•		•		•		
		•		•		•		

EXAMPLE 27

INST	NAME	FUNC	OP1	OP2	OP3	OP4	OP5	OP6
	Y ,I =		Y	00 + X ,J				

APPENDIX 4

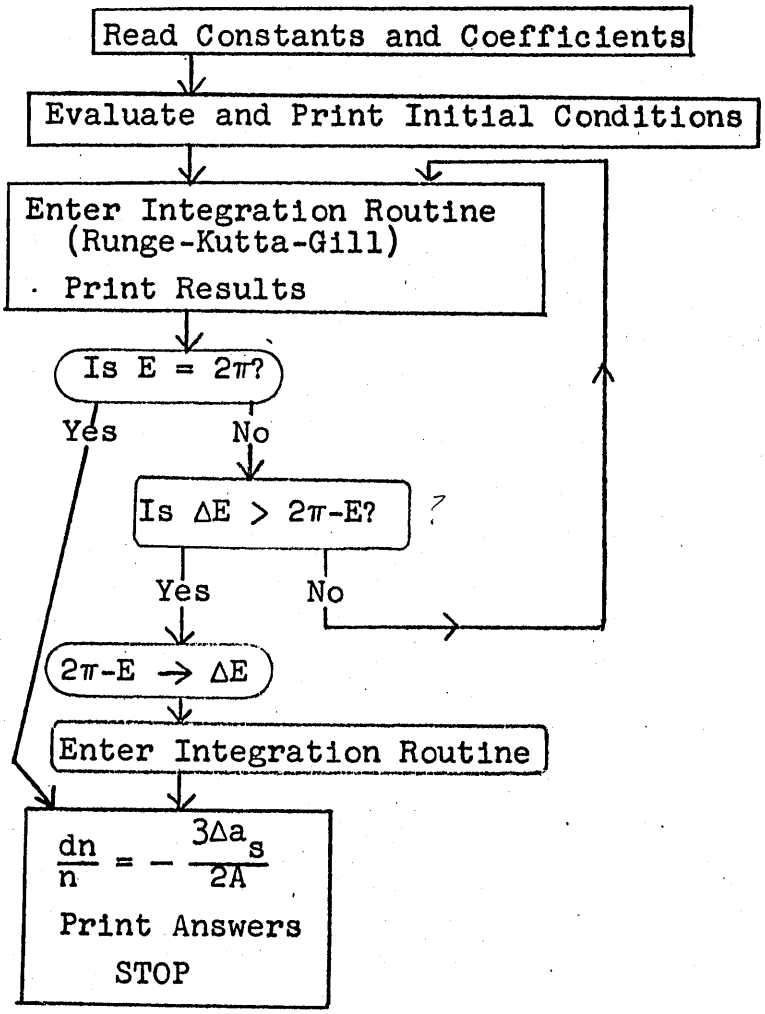
Evaluate $\frac{dn}{n} = -\frac{3\Delta a_s}{2A}$

where: $\Delta a_s = C_I \int_0^{2\pi} \rho(E) \frac{(1+e \cos E)^{3/2}}{(1-e \cos E)^{1/2}} dE$

$$h = a_r(1-e \cos E) - a_e$$

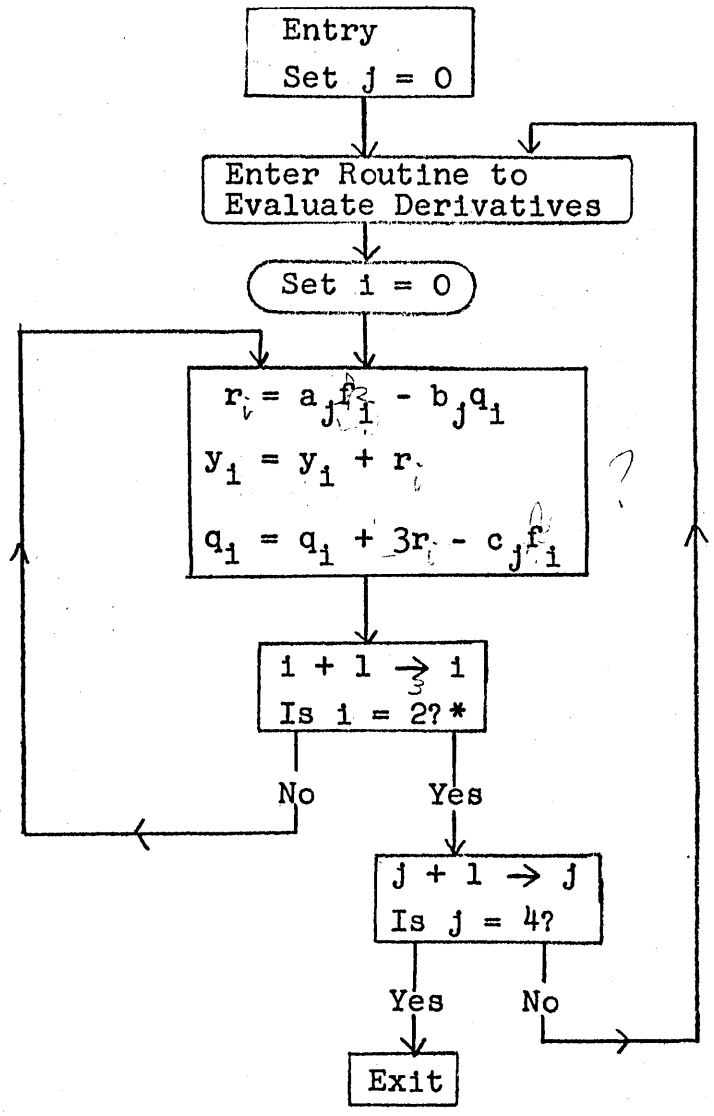
$$\log_{10} \rho(E) = a_0 + a_1 h + a_2 h^2 + a_3 h^3 + a_4 h^4 + a_5 h^5$$

Flow Diagram



APPENDIX 4

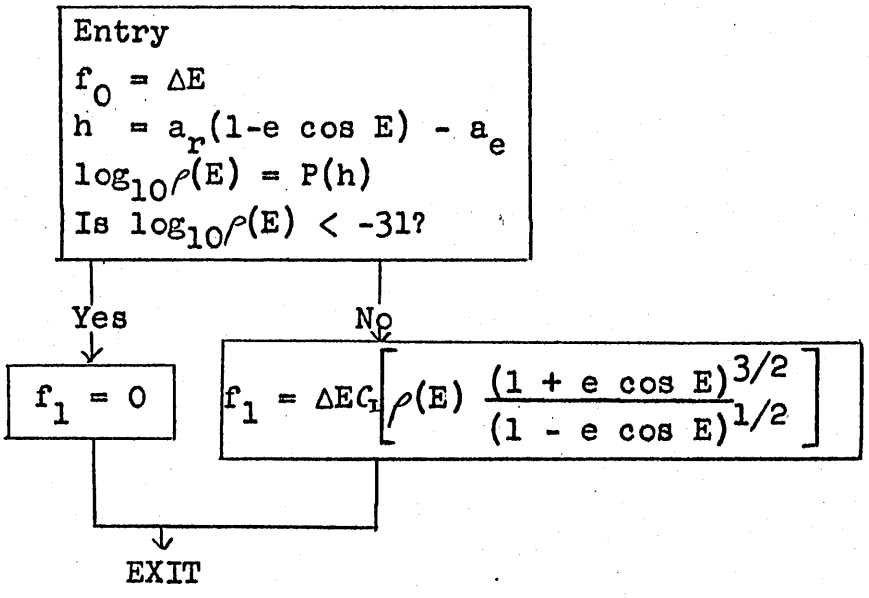
Flow Diagram for RKG Integration Routine



* Actually the number of equations to be integrated. In this case 2, counting the independent variable.

APPENDIX 4

Flow Diagram for Routine to Evaluate Derivatives



APPENDIX 4

SMAC 1 TEST CASE

```
INST    NAME      FUNC OP1      OP2       OP3       OP4       OP5       OP6

DEFINE      I =          2
DEFINE      J =          4
READ
HEDING                   E        DA/S      P(E)      H
SPACE
ENTER   EVAL
        Y  01 =      F   01 / DE
PRINT                Y   00    Y   01    P(E)      H
        DO
ENTER   RKG
PRINT                Y   00    Y   01    P(E)      H
IS      Y   00 =     EMAX
YES     DN/N
        Z       =    EMAX   - Y   00
IS      DF      G    Z
NO      DO
        DE      =    EMAX   - Y   00
ENTER   RKG
        DN/N    = NEG C3     * Y   01 / TWO    * A
SPACE
HEDING               E        DA/S      DN/N
SPACE
PRINT                Y   00    Y   01    DN/N
STOP
ENTRY   RKG
SET         J =          0
        RKG1
ENTER   EVAL
SET         I =          0
        R       =    A   ,J * F   ,I - B   ,J * Q   ,I
        Y   ,I  =    Y   ,I + R
        Q   ,I  =    Q   ,I + C3     * R     - C   ,J * F   ,I
            I   =        I +       1
IS          I   =        2
NO      R
            J   =        J +       1
IS          J   =        4
NO      RKG1
EXIT
ENTRY   EVAL
        F   00  =    DE
        I1      = COS Y   00
        H       =    A/R    - A/E    - A/R    * EX     * I1
        P0      =    A/4    + H      * A/5
```

APPENDIX 4

```
INST    NAME      FUNC OP1      OP2      OP3      OP4      OP5      OP6

        P0      =      A/3    + H      * P0
        P0      =      A/2    + H      * P0
        P0      =      A/1    + H      * P0
        L10P    =      A/0    + H      * P0
IS      L10P    L      C-31
YES     EV1
        P0      = LOG  TEN
        P(E)    = EXP  L10P   * P0
        I2      =      ONE    + EX     * I1
        I2      = SQRT I2     * I2     * I2
        I2      =      P(E)   * I2
        I3      = SQRT ONE    - EX     * I1
        F   01  = NEG  DE     * CI     * I2     / I3
GO TO   EV2
        EV1
        F   01  =      B   00
        EV2
EXIT
START
```

APPENDIX 4

INST	NAME		NUMBER	
	DE	=	+.1	+00
	EMAX	=	+.6283184	+01
	C3	=	+.3	+01
	TWO	=	+.2	+01
	Y	01 =	+.0	+00
	A	00 =	+.5	+00
	A	01 =	+.292893219	+00
	A	02 =	+.170710678	+01
	A	03 =	+.166666667	+00
	B	00 =	+.0	+00
	B	01 =	+.292893219	+00
	B	02 =	+.170710678	+01
	B	03 =	+.333333333	+00
	C	00 =	+.5	+00
	C	01 =	+.292893219	+00
	C	02 =	+.170710678	+01
	C	03 =	+.5	+00
	ONE	=	+.1	+01
	EX	=	+.0482	+00
	A/R	=	+.69265	+04
	A/0	=	+.61044486	+01
	A/1	=	-.635663623	-01
	A/2	=	+.208897299	-03
	A/3	=	-.421130308	-06
	A/4	=	+.457684827	-09
	A/5	=	-.205798723	-12
	TEN	=	+.10	+02
	CI	=	+.796545454	+00
	A/E	=	+.6378	+04
	C-31	=	-.31	+02
EOF				

APPENDIX 4

SMAC 1 TEST CASE

E	DA/S	P(E)	H
.000000000 00	-.550046864 -01	.627763053 -01	.214642700 03
.999999998 -01	-.603928615 -01	.590115447 -01	.216310598 03
.199999998 00	-.651616914 -01	.492098648 -01	.221297619 03
.299999997 00	-.689246097 -01	.368086554 -01	.229553939 03
.399999996 00	-.716078595 -01	.251086449 -01	.240997064 03
.499999995 00	-.733681258 -01	.159301727 -01	.255512655 03
.599999994 00	-.744517808 -01	.959298899 -02	.272955679 03
.699999993 00	-.750901583 -01	.558702451 -02	.293151852 03
.799999992 00	-.754563567 -01	.319758497 -02	.315899378 03
.899999991 00	-.756638865 -01	.182132092 -02	.340970973 03
.999999990 00	-.757814049 -01	.104262675 -02	.368116129 03
.109999998 01	-.758484917 -01	.604485049 -03	.397063618 03
.119999997 01	-.758873739 -01	.357163874 -03	.427524210 03
.129999996 01	-.759103860 -01	.216130544 -03	.459193551 03
.139999995 01	-.759243523 -01	.134339709 -03	.491755213 03
.149999994 01	-.759330584 -01	.857246476 -04	.524883850 03
.159999993 01	-.759386174 -01	.558322214 -04	.558248450 03
.169999992 01	-.759422208 -01	.366422907 -04	.591515648 03
.179999991 01	-.759445539 -01	.237418534 -04	.624353048 03
.189999990 01	-.759460276 -01	.147794109 -04	.656432550 03
.199999989 01	-.759469096 -01	.856810560 -05	.687433628 03
.209999988 01	-.759473949 -01	.448753457 -05	.717046522 03
.219999987 01	-.759476337 -01	.207383750 -05	.744975359 03
.229999986 01	-.759477370 -01	.836769493 -06	.770941078 03
.239999985 01	-.759477761 -01	.297107592 -06	.794684241 03
.249999984 01	-.759477893 -01	.956032712 -07	.815967612 03
.259999983 01	-.759477933 -01	.293379070 -07	.834578536 03
.269999982 01	-.759477944 -01	.919907869 -08	.850331058 03
.279999981 01	-.759477948 -01	.319286893 -08	.863067785 03
.289999980 01	-.759477949 -01	.133106708 -08	.872661455 03
.299999979 01	-.759477949 -01	.716800039 -09	.879016212 03
.309999978 01	-.759477949 -01	.526288907 -09	.882068560 03
.319999977 01	-.759477949 -01	.541623394 -09	.881788003 03
.329999976 01	-.759477949 -01	.779272986 -09	.878177342 03
.339999975 01	-.759477949 -01	.151738587 -08	.871272655 03
.349999974 01	-.759477949 -01	.377423999 -08	.861142931 03
.359999973 01	-.759477952 -01	.111257488 -07	.847889383 03
.369999972 01	-.759477965 -01	.358042226 -07	.831644436 03
.379999971 01	-.759478014 -01	.116259043 -06	.812570404 03
.389999970 01	-.759478171 -01	.356438116 -06	.790857868 03
.399999969 01	-.759478634 -01	.983945429 -06	.766723772 03
.409999968 01	-.759479833 -01	.238356688 -05	.740409258 03
.419999967 01	-.759482547 -01	.504445115 -05	.712177250 03
.429999966 01	-.759487950 -01	.945025531 -05	.682309833 03
.439999965 01	-.759497614 -01	.160714813 -04	.651105433 03
.449999964 01	-.759513580 -01	.255898492 -04	.618875834 03
.459999963 01	-.759538704 -01	.393407571 -04	.585943062 03
.469999962 01	-.759577445 -01	.599476012 -04	.552636172 03
.479999961 01	-.759637305 -01	.922965800 -04	.519287956 03
.489999960 01	-.759731395 -01	.145244245 -03	.486231616 03
.499999959 01	-.759883006 -01	.234748398 -03	.453797441 03

APPENDIX 4

.509999958	01	-.760133923	-01	.389597520	-03	.422309502	03
.519999957	01	-.760559527	-01	.661734969	-03	.392082417	03
.529999956	01	-.761296054	-01	.114434992	-02	.363418204	03
.539999955	01	-.762588703	-01	.200187329	-02	.336603267	03
.549999954	01	-.764872685	-01	.351439727	-02	.311905532	03
.559999953	01	-.768898202	-01	.612854239	-02	.289571770	03
.569999952	01	-.775892075	-01	.104764321	-01	.269825132	03
.579999951	01	-.787691938	-01	.172689855	-01	.252862922	03
.589999950	01	-.806682187	-01	.269276965	-01	.238854618	03
.599999949	01	-.835265703	-01	.389198272	-01	.227940189	03
.609999948	01	-.874719486	-01	.511427176	-01	.220228686	03
.619999947	01	-.923803387	-01	.601421826	-01	.215797162	03

E		DA/S		DN/N	
.628318399	01	-.968907868	-01	.290672360	00

APPENDIX 5

Error Detection

It is likely that sometime while using SMAC you will receive, instead of answers (right or wrong) from the machine, messages which indicate that all is not correct with your statement of the problem. These arise from the built-in error detection system in SMAC.

There are two classes of detectable errors. The first are general system errors, the second, statement errors.

The following are the messages which arise from general system errors and their meanings:

STATEMENT LIST FULL - Your problem contains more than 140 statements.

DEFINITION LIST FULL- You have defined more than 6 subscripts.

UNDEFINED SUBSCRIPT - You have appended a subscript to a variable without defining it.

DATA BLOCK FULL - Your data needs more storage than is available.

SOMEBODY GOOFED - The system does not have all of the information required to continue processing.

- This may arise from a possible machine failure.

APPENDIX 5 (Cont.)

When a statement error is detected, the erroneous statement is printed along with a message. In some cases the statement has been transformed in processing it, and therefore will not be in a readable form. When an error is grievous enough, all processing stops at that point. In most cases processing continues but only with the hope of detecting remaining errors. If any errors are detected execution of the instructions will not take place.

The following table of error messages indicates whether or not the statement printed with the message will be readable or have meaning and whether or not processing continued after the error was detected:

Message	Statement	Continues?
UNDEFINED SUBSCRIPT [1]	Not Always Readable	No
YOU CANT MIX SUBSCRIPTS	Readable	Yes
YOU CANT SET A VARIABLE	Readable	Yes
VARIABLE MISSING [2]	Readable	Yes
VAR MISSING IN GO TO	Readable	Yes
VAR MISSING IN ENTER	Readable	Yes
VAR MISSING IN ENTRY	Readable	Yes
TWO ENTRYS NO EXIT	Readable	Yes
EXIT BEFORE ENTRY	Readable	Yes
NO EXIT FROM LAST ENTRY	Has No Meaning	No
NO YES OR NO AFTER IS	Readable	Yes
ILLEGAL INSTRUCTION	Readable	Yes

APPENDIX 5 (Cont.)

Message	Statement	Continues?
ILLEGAL FUNCTION 3)	Not Readable	Yes
SOMEBODY GOOFED, RELOAD 4)	Not Readable	No
UNDEFINED NAMES 5)	Readable	No
PROBLEM TOO LONG 6)	Has No Meaning	No

1) This should occur only when the subscript is being used as a simple counter and has not been defined. In these cases the statement will be readable.

2) A completely blank card will cause this error.

3) The illegal function itself will be readable, therefore, it should be possible to locate the proper statement.

4) Here again, as above, is an indication of possible machine failure, but not always - an illegal form may have escaped earlier detection.

5) In this case the statement will be a list of the undefined names.

6) In processing the statements, though the problem contains less than 140 statements, it is possible that resulting machine instructions exceed 1200 in number.

SMAC DATA FORM

NAME _____ DIV _____ DEPT _____ PAGE _____ OF _____

PROBLEM TITLE _____ DATE _____

INST (EØF)	NAME S	=	NUMBER ± . FRACTION ± P	DESCRIPTION	BLOCK Nº 75 76
01 02 03 04 05 06	08 09 10 11 12 13	15	17 18 19 20 21 22 23 24 25 26 27	29 30 31	33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66	73 74 75 76 77 78 79 80

01
02
03
04
05
06
07
08
09
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25