Solution of Simultaneous Linear Equations
(Revised)

Solution of Simultaneous Linear Equations

Author: Mr. Burr Preston
IBM Corporation
1st Data Processing Unit
APO 343
San Francisco, California

Deck Key

1. Fortran Source Language Deck
   SLE 01 through 59

2. Object Deck
   0001 through 0413

3. Sample Problem Data Deck
   SP 1 01 - 1 05
   SP 2 01 - 2 19
   SP 3 01 - 3 07
   SP 4 01 - 4 07

Solution of Simultaneous Linear Equations

5.0.007

Author: Mr. Burr Preston

Direct Inquiries to: Mr. Burr Preston
IBM Corporation
1st Data Processing Unit
A.P.O. 343
San Francisco, California

A. Purpose/Description: This program solves sets of non-homogeneous simultaneous linear equations and provides either printed or punched output. It is designed for ease of use. Operating instructions and error messages are automatically typed. Data values are entered in free form notation as a group of digits with a decimal point. An optional power of ten may be added to each value.

B. Method: The Jordan method of elimination is used, a test for zero divisor is included. A typewriter message indicates when a pivotal element is smaller in absolute value than a value selected by the operator. At this point the solution may be continued or the next problem read in.

C. Restrictions and Range: A maximum of 25 equations in 25 unknowns may be solved. A maximum of 8 significant digits per matrix element is allowed.

D. Accuracy: Varies with the size of the matrix.

E. Machine Configuration: Memory 20K, Card Input-Output. No other special features are required.

F. Program Requirements: The entire core for 25 equations.

G. Source Language: The program is written in Fortran without Format. Alphabetic output and format control are accomplished by two subroutines incorporated in the program. All computation is done in standard Fortran single precision floating point arithmetic.

H. Program Execution Time: Read and compute time for 3 equations is 5 seconds. Typing the answer takes an additional 7 seconds. Read and compute time for 8 equations is 25 seconds. 19 additional seconds are required for typing the answer.

I. Check Out Status: The program has been used successfully on systems of 3 to 8 equations, and the logic has been verified for all other cases.

J. Sample Problem Running Time: N/A

K. Comments: This program and its documentation were written by an IBM employee. It was developed for a specific purpose and submitted for general distribution to interested parties in the hope that it might prove helpful to other members of the data processing community. The program and its documentation are essentially in the author's original form. IBM serves as the distribution agency in supplying this program. Questions concerning the use of the program should be directed to the author's attention.

TABLE OF CONTENTS

Description of the Program

This program solves sets of non-homogeneous simultaneous linear equations and provides either typewriter or punched card output. Up to 26 equations in 25 unknowns may be solved. The first card read for each set of equations states the number of equations following. The program automatically sizes itself for the proper matrix dimensions. Any number of sets of equations may be solved one after the other without reloading the program deck. When a solution has been obtained the word SOLUTION is typed. Under this heading the solution variables are typed one per line in the order in which they appear from left to right in the original equations. Punched card output is identical, with one card punched for each line typed.

If a set of equations is submitted which has more than one solution, a valid solution will be typed or punched, but no indication will be given that there are other solutions as well.

If the equations in the set are not all independent, a typewriter message will indicate that a diagonal element is zero. No solution is developed in this case, as there are infinite solutions to the set of equations.

Additional information regarding the characteristics of the program is contained in the discussion of the method of computation.

Method of Computation

General Description. The Jordan method of elimination[1] is used to accomplish the solution. Briefly this method consists of diagonalizing the augmented coefficient matrix. After diagonalization the right hand or augmented column contains the solution values for the variables.

The reduction of the matrix takes place in the same storage locations where the original coefficients are stored. Thus at the completion of the solution the original coefficients are lost, and in their place remain an identity matrix and the column of answers.

Division by a Zero Diagonal. In the process of diagonalization the elements of each row to the right of the diagonal and the diagonal itself are divided by the diagonal element. The diagonal elements are used in the same order in which they appear in the original matrix. No attempt is made to select these pivotal elements by magnitude. Hence a zero must not appear on the diagonal of the matrix. If such a condition exists, a message will by typed indicating which diagonal element is zero and control passes to reading the next problem.

[1] Alston S. Householder, Principles of Numerical Analysis (New York: McGraw-Hill Book Co. Inc., 1953), pp. 68-72.

The possibility of zero diagonal elements may be minimized by observing two rules in laying out the original matrix.

1. The upper left hand element should not be zero.
2. No diagonal element may be zero if all elements in the row to the left of it are zero.

There is a slight chance of generating a zero diagonal element in the course of calculations. Equations which are not independent (a unique solution undefined) will also generate a zero diagonal. If it is known that the coefficient matrix is non-singular, that is, that there is a unique solution, division by zero may be easily corrected and the problem submitted again. A simple rearrangement of the columns will usually suffice. In rearranging the columns the equality vector always must be kept on the right. As indicated later, rearrangement of columns may be accomplished simply by rearranging input cards. No additional punching is required.

Tolerance Warning Level. When the diagonal element is very small relative to the values to its right in the row, a loss of accuracy will result from round-off errors. In this program the operator has control over this situation. When the absolute value of the divisor is less than or equal to some tolerance level selected by the operator, a warning message is typed. The operator then has the choice of continuing with the solution or proceeding to the next problem. If he should elect to continue, he does so with the knowledge that the answer may contain significant rounding errors. Of course, checking the answers will quickly show the accuracy of the solution.

Input Card Format

The first card of the input for each set of equations must be a card with the number of equations punched anywhere between columns 1 and 72 inclusive. If the number of equations is less than 10, the leading zero need not be present, although its presence makes no difference. A decimal point must not appear with this number. The remainder of the card is blank.

The matrix of coefficients is punched column-wise beginning with the left-most column and ending with the equality column. Each column must begin a new card. A card must contain five values. The column may be continued over as many cards as necessary. (Five cards is the maximum number, as the program is limited to 25 unknowns.) If the number of rows is not a multiple of five, dummy values must be punched to finish out five entries on the last card of the column. It is never necessary to punch more than four dummy entries for a given column. The number of cards per column will be the same for all columns within a given problem, but may vary from problem to problem. The dummy entries used to fill out the five entries on the last card of a column need not be zero. These entries are accessed during the read in, but are never accessed during calculation. Hence they must be present, but their value is immaterial.

The numerical value of each entry is punched in a free form notation--the standard floating point variable form of 1620 FORTRAN. For those not familiar with this FORTRAN format, the specifics are detailed below.

Each value may contain as many as 8 significant digits with a decimal point at the beginning, among, or at the end of the group of digits. Zeros immediately to the right of the decimal point in a fractional number are not considered significant. A plus or minus sign may precede the number; the presence of the plus sign for positive numbers is optional. Any value may be followed by an E xx, where xx is a 1 or 2 digit power of 10 by which the number is to be multiplied. The plus sign is optional, but if omitted the space it would have occupied must also be omitted.

Examples of input values might be as follows:

+1. 23.004678 -8.4 .000000012345678 3.4E+8 -.4E10 267.E-9

In the last three examples the numbers represented are in order 340000000.,-4000000000., and .000000267 .

A DECIMAL POINT MUST APPEAR IN EACH ENTRY.

At least one blank column must be left between each of the five entries. The entries including leading blank columns may be in columns 1 - 72 inclusive.

Columns 73-80 of each input card are not read by the program and are available for any identification the user may desire.

Output Card Format

The first card of output for each problem will be a header card with SOLUTION$\pm$ punched in columns 1-10. Columns 77-80 of this card will contain $\bar{0}413$. The following cards contain the solution variables, one per card, in the order in which the variables appeared in the original matrix. The values are punched in standard 1620 FORTRAN floating point notation, that is, exactly like the input. However, the E$\pm$ xx form will appear only if the decimal point does not fall at the beginning, end, or among the 8 significant digits. The punching begins in column 1.

Output Typewriter Format

This output is identical to the card output except that one line is printed for each card punched and the record mark following the word SOLUTION is not printed. This record mark stops transmission so the constant $\bar{0}413$ is not printed.

Deck Identification

The FORTRAN source language deck has SLE in columns 73-75 and a sequence number 01-59 in column 79-80.

The machine language object deck in columns 77-80 contains a sequence number, $\bar{0}001$-$\bar{0}413$.

OPERATING INSTRUCTIONS

I. A. Initial Console Setting

	Program	Stop
Parity		X
I/O		X
Overflow	X	

B. Sense Switch Settings

Sense Switch 1	ON	Solution is punched.
	OFF	Solution is typed.
Sense Switch 2	ON	Next problem after tolerance stop.
	OFF	Continue after tolerance stop.
Sense Switch 3	ON	Tolerance level entered for each problem.
	OFF	Tolerance level entered only once.
Sense Switch 4	ON	To correct error in typing tolerance level.
	OFF	Tolerance level entered correctly.

See Section V for further comment on Switches 2, 3, and 4.

II. Input-Output

Card Reader

No. of Cards	Description
413	Program Deck
1	N card - no of equations in 1st set of equations
3-130	Coefficient cards for 1st set of equations
1	N card - no of equations in 2nd set of equations
3-130	Coefficient cards for 2nd set of equations
	Etc.

Card Punch

No. of Cards	Description
1	SOLUTION Header card
2-25	Solution values for 1st set of equations
1	SOLUTION Header card
2-25	Solution values for 2nd set of equations
	Etc.

Typewriter Output

Control	Description
Margins	Set left margin as desired
Tab Stops	None
Forms	Standard paper

III. Normal Loading Procedure

1. Clear Storage
2. Depress RESET
3. Depress LOAD button
4. Depress COMPUTER START after message LOAD DATA has been typed. See paragraph below.
5. Depress READER START when hopper is empty

When the LOAD DATA message appears the last few cards of the program deck will still be in the read hopper. If data cards are in the hopper following the program deck, pressing COMPUTER START will cause the remaining program cards to be read and a transfer to the program with subsequent reading of data cards. If no data follows the program, depressing COMPUTER START will cause loading to continue. The READER START must be depressed when the reader hopper is empty in order to read the last few program cards. When data cards are later placed in the hopper, READER START must be depressed to begin data entry and again when the hopper is empty to read the last data cards.

IV. Special Loading Instructions

At any time the program may be initialized and started by branching to 07548.

OK.

V. Other Instructions and Remarks

Tolerance Level. When the program begins, the typewriter will request the operator to enter the tolerance level as a floating point number, that is, in the same format as the input matrix coefficients--see page 5. A record mark must immediately follow this entry.

If the entry is typed correctly, set Sense Switch 4 OFF, press

RELEASE and COMPUTER START. The data cards will then be read.

If the entry is typed incorrectly, turn Sense Switch 4 ON, press RELEASE and START. The program will immediately return control to the typewriter so the entry may be made again. If the retry is correct, follow the procedure for a correct entry.

If it is desired to enter a tolerance level for each set of equations, turn Sense Switch 3 ON. If not, turn Switch 3 OFF, and the last tolerance level entered will automatically be applied to all succeeding sets of equations.

When a tolerance warning stop occurs, directions for the use of Switch 2 are typed. Only at this time is the setting of Switch 2 interrogated. If the tolerance level is greater than 9 integers or smaller than .000000001, the tolerance level will appear as an excess 50 floating point number in the message.

VI. Programmed Stops and Required Action

All programmed stops are accompanied by typewriter messages which are self explanatory and which indicate the required action.

As the program is written in FORTRAN, all the error stops and messages of that system apply. The more cryptic of these are of the form E1, E2, ..., E9, F1, F2, ..., F7.

The FORTRAN processor without format, 1620-FO-002, was used to compile this program. All references to FORTRAN refer to this processor.

SAMPLE PROBLEMS

The following four problems were used one after the other.

$$
\begin{aligned}
8x_1 - 5x_2 + 7x_3 &= 29 \\
+2x_2 - 2x_3 &= -2 \\
-x_1 + 9x_2 - 6x_3 &= 1
\end{aligned}
$$

Answer: $x_1 = 2,\ x_2 = 3,\ x_3 = 4$

$$
\begin{aligned}
x_1 + 2x_2 + 3x_3 - 4x_4 + 5x_5 - 6x_6 + 7x_7 - 8x_8 &= -28 \\
7x_2 - 2x_5 + x_7 + x_8 &= 19 \\
2x_1 + 3x_2 - 4x_3 - 5x_4 + 6x_6 &= 12 \\
x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 &= 36 \\
10x_1 + 3x_2 - 4x_3 + x_5 + 2x_6 - 9x_7 &= -42 \\
3x_1 - 3x_2 - 2x_3 + 2x_4 + x_6 &= 5 \\
-8x_1 - 9x_3 + 7x_5 + 6x_7 - 3x_8 &= 18 \\
+3x_7 - 5x_8 &= -19
\end{aligned}
$$

Answers: $x_1 = 1,\ x_2 = 2,\ x_3 = 3,\ x_4 = 4$
$x_5 = 5,\ x_6 = 6,\ x_7 = 7,\ x_8 = 8$

$$
\begin{aligned}
3x_1 + 8x_2 + 6x_3 + 10x_4 + 42x_5 &= -20 \\
+2x_2 + x_4 + 5x_5 &= 40 \\
+ 4x_4 + 6x_5 &= -18 \\
2x_1 + 4x_2 + 7x_3 + 9x_5 &= 12 \\
x_1 + 4x_2 + 6x_3 - 2x_4 + 10x_5 &= 7
\end{aligned}
$$

This problem will cause diagonal 3 to be zero. It was included merely to illustrate the typewriter message.

$$
\begin{aligned}
3x_1 + 8x_2 + 6 x_3 + 10x_4 + 42x_5 &= -20 \\
+2x_2 + x_4 + 5x_5 &= 40 \\
+ .01x_3 + 4x_4 + 6x_5 &= -18 \\
2x_1 + 4x_2 + 7 x_3 + 9x_5 &= 12 \\
x_1 + 4x_2 + 6 x_3 - 2x_4 + 10x_5 &= 7
\end{aligned}
$$

This problem will make diagonal 3 go below a tolerance level of 0.1 . It was included merely to illustrate this typewriter message.

The sample problem deck contains SP in columns 73-74; the problem number, 1, 2, 3, or 4, in column 77; and a sequence number in columns 79-80. The sequence numbers are 01-05 for problem 1, 01-19 for problem 2, and 01-07 for problems 3 and 4.

SAMPLE PROBLEMS

INPUT

```
3
8.  0.  -1.  0.  0.
-5.  2.  9.  0.  0.
7.  -2.  -6.  0.  0.
29.  -2.  1.  0.  0.
```

```
8
1.  0.  2.  1.  10.
3.  -8.  0.  0.  0.
2.  7.  3.  1.  3.
-3.  0.  0.  0.  0.
3.  0.  -4.  1.  -4.
-2.  -9.  0.  0.  0.
-4.  0.  -5.  1.  0.
2.  0.  0.  0.  0.
5.  -2.  0.  1.  1.
0.  7.  0.  0.  0.
-6.  0.  6.  1.  2.
1.  0.  0.  0.  0.
7.  1.  0.  1.  -9.
0.  6.  3.  0.  0.
-8.  1.  0.  1.  0.
0.  -3.  -5.  0.  0.
-28.  19.  12.  36.  -42.
5.  &18. -19.  0.  0.
```

```
   5
3.  0.  0.  2.  1.
8.  2.  0.  4.  4.
6.  0.  0.  7.  6.
10.  1.  4.  0.  -2.
42.  5.  6.  9.  10.
-20.  40.  -18.  12.  7.
```

```
   5
3.  0.  0.  2.  1.
8.  2.  0.  4.  4.
6.  0.  1.E-2  7.  6.
10.  1.  4.  0.  -2.
42.  5.  6.  9.  10.
-20.  40.  -18.  12.  7.
```

COMMENTS ON THE TYPEWRITER LOG

FOR SAMPLE PROBLEMS

On the typewriter log, which follows, several points should be noted. Sense Switch 3 was on at the beginning. Note that a tolerance level was entered for each of the first three problems. After the tolerance was entered for the third problem, Sense Switch 3 was turned off. When diagonal 3 went to zero and COMPUTER START was pressed to continue with the fourth problem, the solution began immediately without the entry of a tolerance level, as Switch 3 was off. Note the tolerance level (0.1) in the warning message of the fourth problem is the last entry made--that for problem 3. When this warning message appeared, Switch 2 was turned off and calculation continued to obtain a solution.

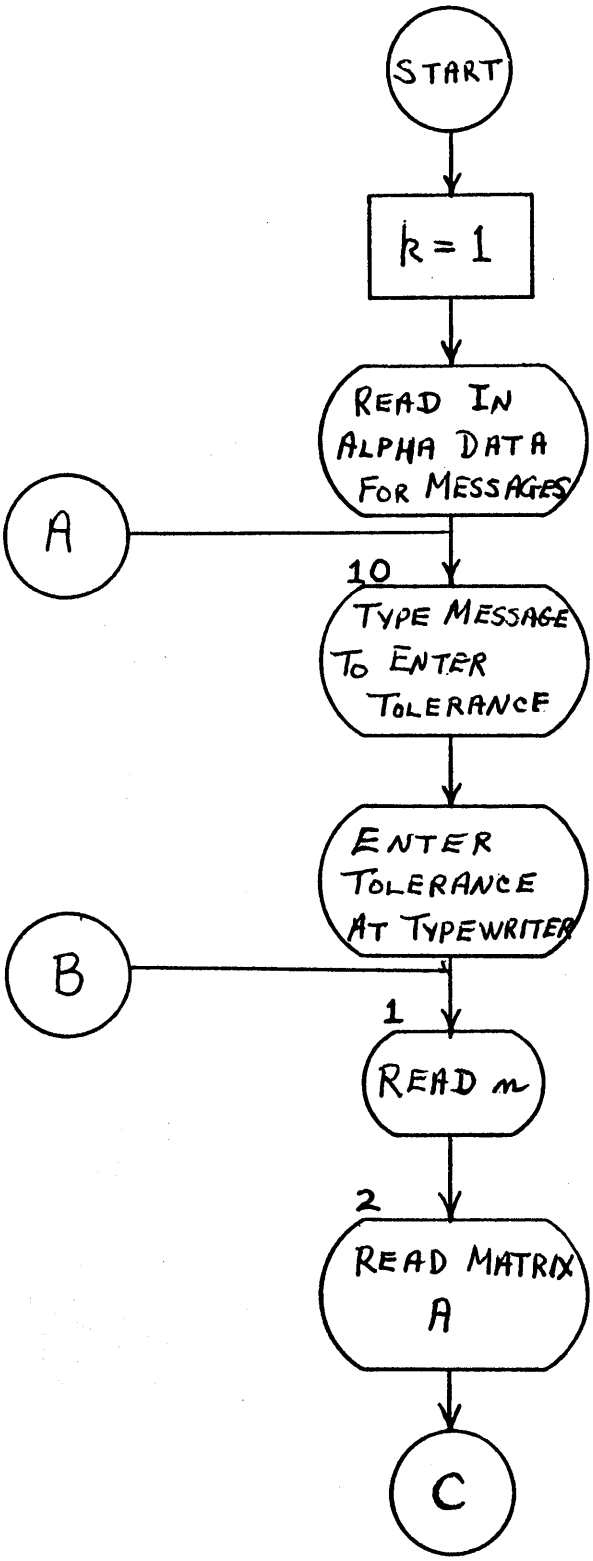
TYPEWRITER LOG

VERIFIED FOR ORIGINAL 05.0.007 MODIFIED

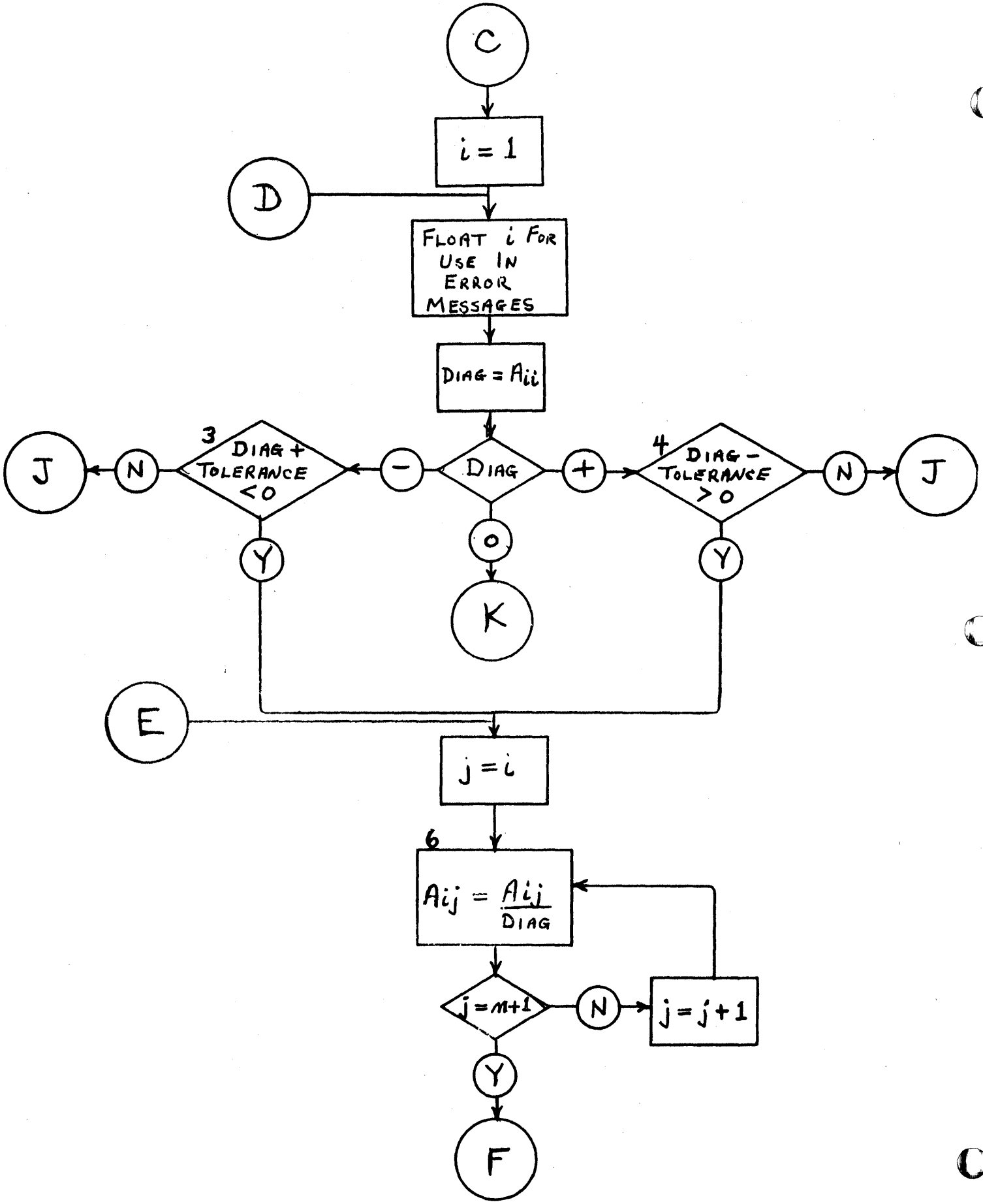
```
LOAD DATA


ENTER TOLERANCE LEVEL AS FP NO WITH RM
THEN PRESS RELEASE AND START
.0000000001‡


SOLUTION
 2.0000000
 3.0000000
 4.0000000


ENTER TOLERANCE LEVEL AS FP NO WITH RM
THEN PRESS RELEASE AND START
1.E-9‡


SOLUTION
 1.0000065
 1.9999976
 3.0000053
 3.9999963
 4.9999916
 5.9999990
 7.0000022
 8.0000012


ENTER TOLERANCE LEVEL AS FP NO WITH RM
THEN PRESS RELEASE AND START
.1‡


DIAGONAL 3 IS ZERO, PUSH START FOR NEXT PROBLEM


DIAGONAL  3 IS LESS THAN OR EQUAL TO          .100000000
TURN SWITCH 2 OFF TO CONTINUE, OR
TURN SWITCH 2 ON FOR NEXT PROBLEM
THEN PUSH START


SOLUTION
-6.7605150
 34.405115
-7.0773320
 5.9439480
-6.9508359
```

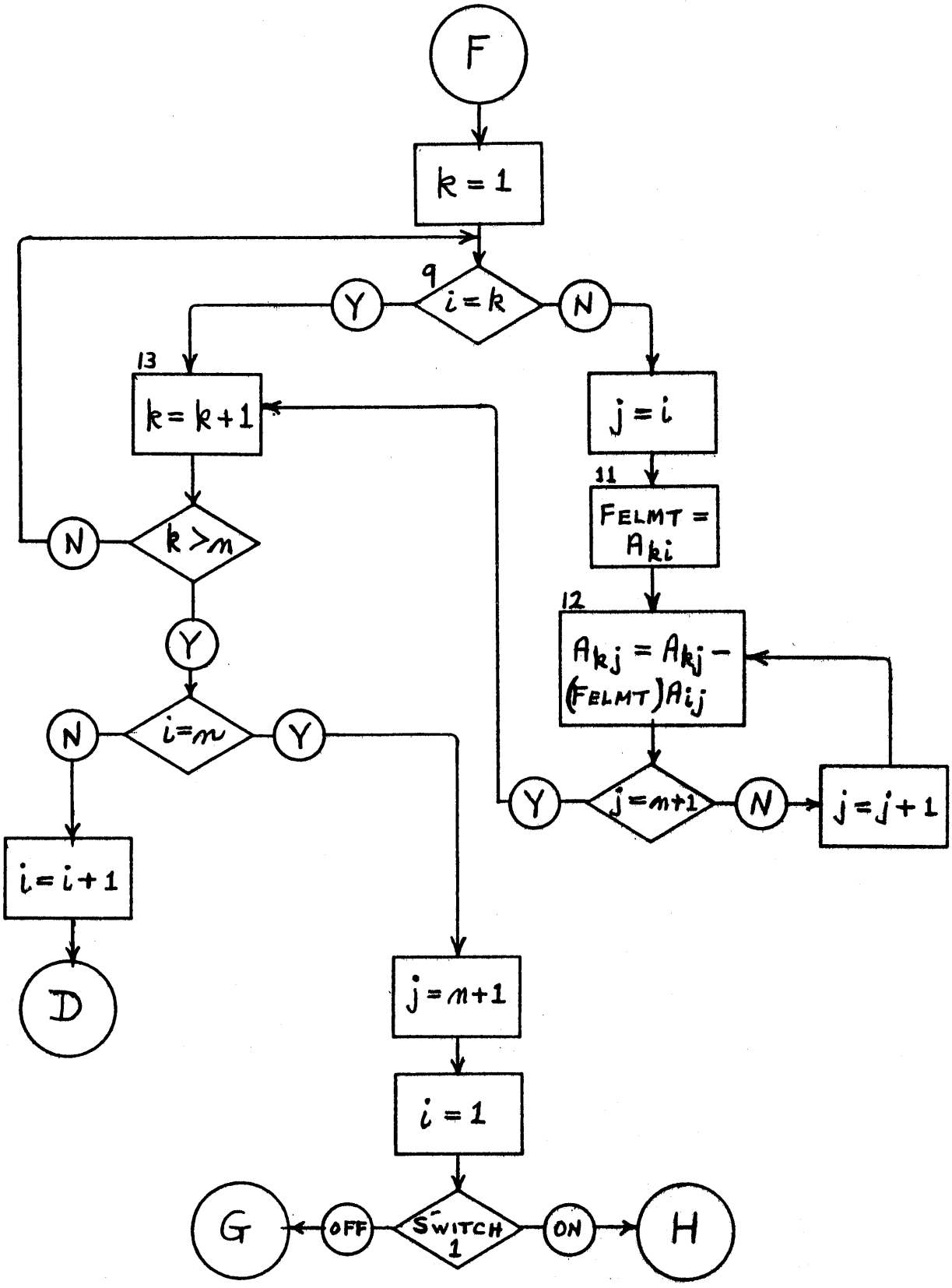
FLOW CHART

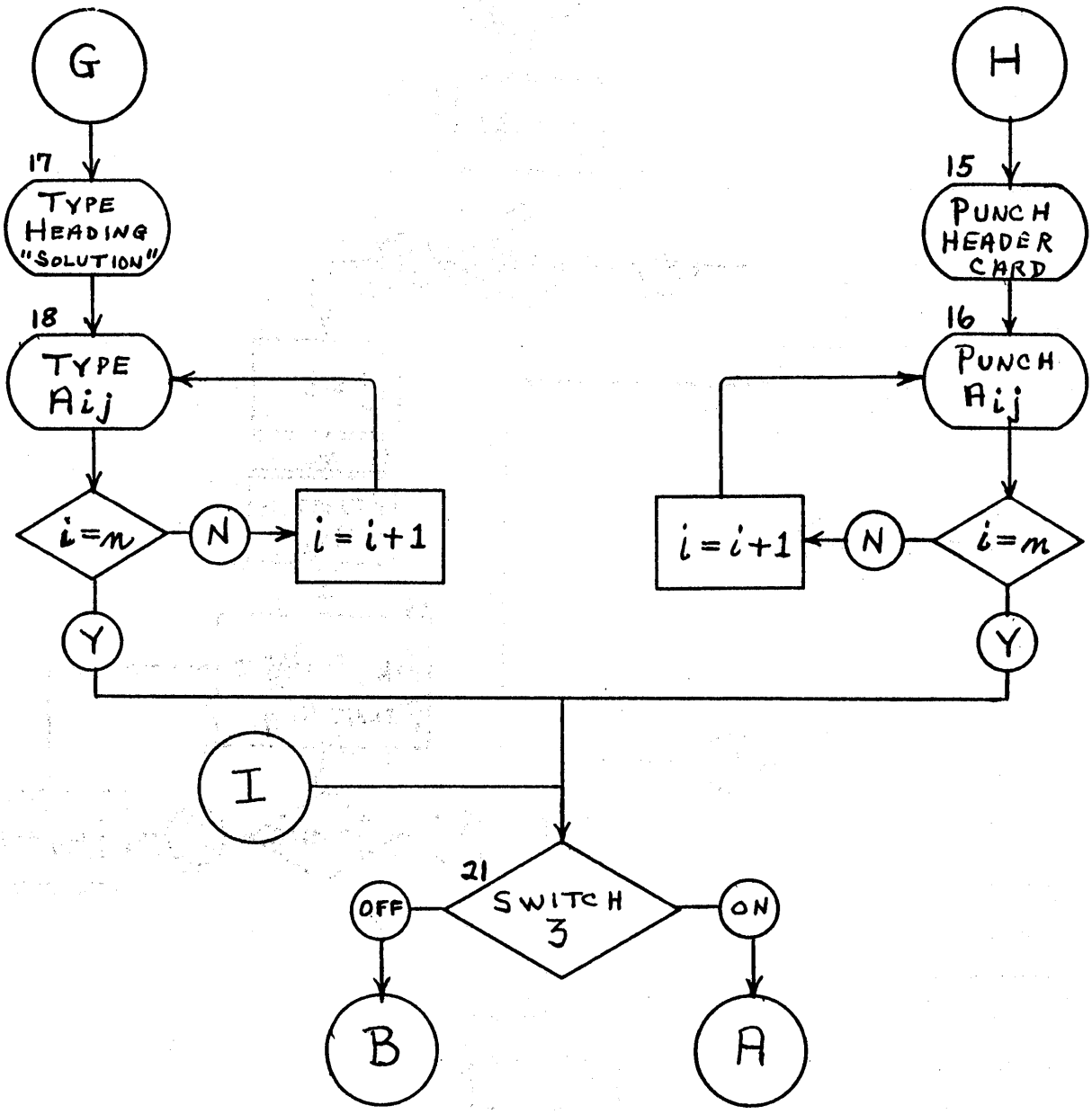
Solution of n Simultaneous Linear Equations
by the Jordan Method of Elimination

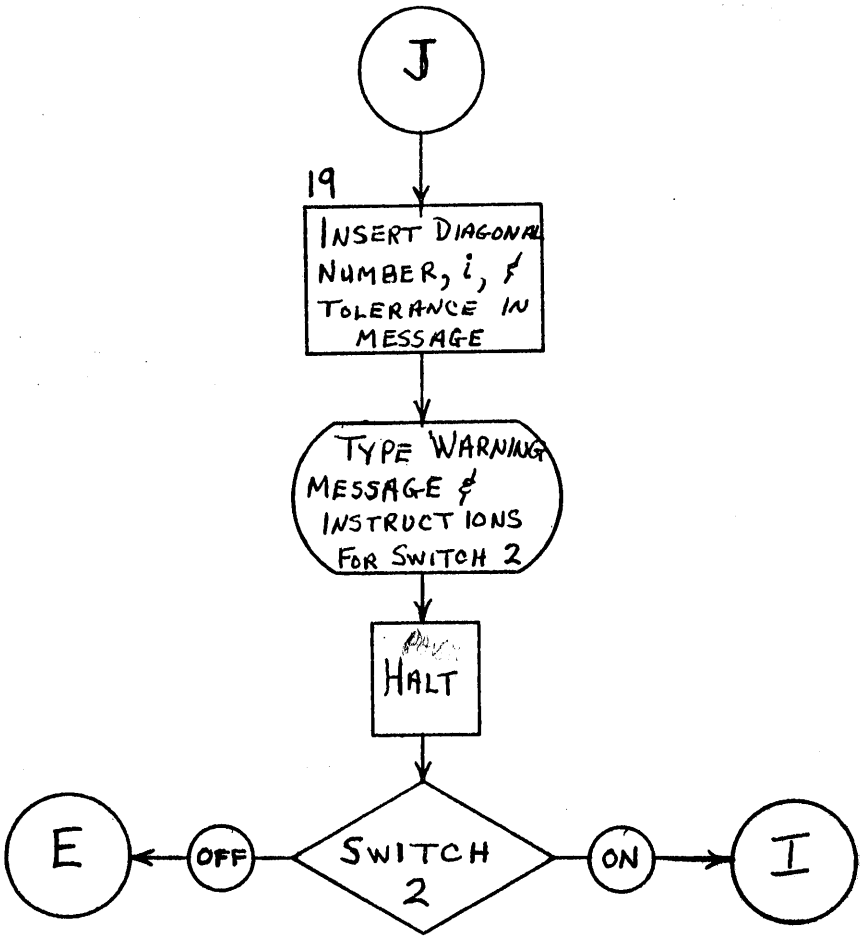


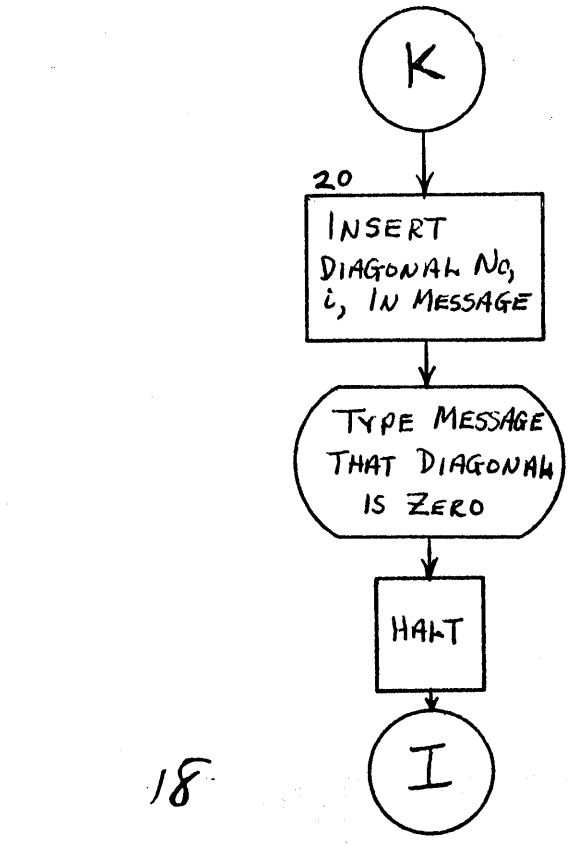
Dummy statement so that an alphabetic subroutine statement will not be the first executable statement


C
i = 1
D
FLOAT i FOR USE IN ERROR MESSAGES
DIAG = Aii
3
DIAG + TOLERANCE < 0
J
N
−
DIAG
+
4
DIAG − TOLERANCE > 0
N
J
Y
0
Y
K
E
j = i
6
Aij = Aij / DIAG
j = m+1
N
j = j+1
Y
F


F
k = 1
9
i = k
Y
N
13
k = k+1
j = i
11
FELMT = Aki
k > n
N
Y
12
Akj = Akj - (FELMT)Aij
i = n
N
Y
j = n+1
Y
N
j = j+1
i = i+1
D
j = n+1
i = 1
G
OFF
SWITCH 1
ON
H


G
17
TYPE HEADING "SOLUTION"
18
TYPE Aij
i=n
N
i = i+1
Y
H
15
PUNCH HEADER CARD
16
PUNCH Aij
i=m
N
i = i+1
Y
I
21
SWITCH 3
OFF
ON
B
A


J
19
INSERT DIAGONAL NUMBER, i, & TOLERANCE IN MESSAGE
TYPE WARNING MESSAGE & INSTRUCTIONS FOR SWITCH 2
HALT
SWITCH 2
OFF
ON
E
I


K
20
INSERT DIAGONAL No, i, IN MESSAGE
TYPE MESSAGE THAT DIAGONAL IS ZERO
HALT
I

```
C      SOLUTION OF N SIMULTANEOUS LINEAR EQUATIONS
C        BY THE JORDAN METHOD OF ELIMINATION
C      N MUST BE LESS THAN OR EQUAL TO 26                    *
       DIMENSION A(26,27)
       K = 1
       D = SAY(1950.)
   10  D = SAY(2200.)
       D = SAY(-7201.)
       ACCEPT, TOLER
    1  READ, N
       N1 = N + 1
       DO 2 J = 1, N1
       DO 2 I = 1, N, 5
    2  READ, A(I,J), A(I+1,J), A(I+2,J), A(I+3,J), A(I+4,J)
       DO 14 I = 1, N, 1
       DNO = I
       DIAG = A(I,I)
       IF (DIAG) 3, 20, 4
    3  IF (DIAG + TOLER) 5, 19, 19
    4  IF (DIAG - TOLER) 19, 19, 5
    5  DO 6 J = I, N1
    6  A(I,J) = A(I,J) / DIAG
       K = 1
    9  IF (K-I) 11, 13, 11
   11  FELMT = A(K,I)
       DO 12 J = I, N1
   12  A(K,J) = A(K,J) - FELMT * A(I,J)
   13  K = K + 1
       IF (K-N) 9, 9, 14
   14  CONTINUE
       J = N1
       IF (SENSE SWITCH 1) 15, 17
   15  D = SAY(9104.)
       DO 16 I = 1, N
   16  PUNCH, A(I,J)
       GO TO 21
   17  D = SAY(2200.)
       D = SAY(-9101.)
       DO 18 I = 1, N
   18  PRINT, A(I,J)
   21  IF (SENSE SWITCH 3) 10, 1
   19  D = SAY(2200.)
       D = SAY(-1101.)
       Y = 10001320.
       Z = OUT(DNO)
       D = SAY(3101.)
       Y = 10001399.
       Z = OUT(TOLER)
       D = SAY(-4301.)
       PAUSE
       IF (SENSE SWITCH 2) 21, 5
   20  D = SAY(2200.)
       D = SAY(-1101.)
       Y = 10001320.
       Z = OUT(DNO)
       D = SAY(2101.)
       PAUSE
       GO TO 21
       END
```

* Comments follow the program listing..

ALPHABETIC CONSTANT CARDS

(Last 9 cards of program deck)

```
DIAGONALZ
 IS ZERO, PUSH START FOR NEXT PROBLEMZ
 IS LESS THAN OR EQUAL TOZ
TURN SWITCH 2 OFF TO CONTINUE, ORZ
TURN SWITCH 2 ON FOR NEXT PROBLEMZ
THEN PUSH STARTZ
ENTER TOLERANCE LEVEL AS FP NO WITH RMZ
THEN PRESS RELEASE AND STARTZ
SOLUTION Z
```

Z is a record mark, 0-2-8 punch.

NOTES ON THE PROGRAM LISTING

Two functions are used in the program. The function SAY is a subroutine for alphabetic output and is described in 1620 Library File No. 1.6.006. The function OUT is a subroutine by R. F. Steinhart for format control of decimal points and is described in issue #38 of 1620 Technical Publications, Systems Engineering, IBM Midwestern Region, where it is referred to as the RFS subroutine. A description of this RFS subroutine is also included in 1620 Users Group Data, Distribution #1, July, 1961.

The alphabetic constants for messages are the last 9 cards of the object program deck. They are not generated by the FORTRAN program, but are actually input data cards to the SAY function. They are read by the second FORTRAN statement, D = SAY (1950.), and therefore are placed immediately behind the object deck generated by FORTRAN. These 9 cards need be of no concern to the user, as they are read only once when the program is loaded.

The FORTRAN processor, 1620-FO-002, used to compile this program was modified to include these two functions.

END OF PROGRAM T0136
START OF TABLE T2359

UNSUBSCRIPTED VARIABLES

T9999	EXP
T9989	EXPF
T9979	LOG
T9969	LOGF
T9959	SQR
T9949	SQRF
T9939	SIN
T9929	SINF
T9919	COS
T9909	COSF
T9899	ATN
T9889	ATNF
T9879	OUT
T9869	OUTF
T9859	ABS
T9849	ABSF
T9839	SAY
T9829	SAYF
T2799	K
T2779	D
T2719	TOLER
T2699	N
T2689	N1
T2669	J
T2659	I
T2639	DNO
T2629	DIAG
T2529	FELMT
T2429	Y
T2409	Z

SUBSCRIPTED VARIABLES

T9819	T2809	A	0026

FLOATING POINT CONSTANTS

T2769	5419500000
T2739	5422000000
T2729	5472010000
T2489	5491040000
T2459	5491010000
T2439	5411010000
T2419	5810001320
T2399	5431010000
T2389	5810001399
T2379	5443010000
T2369	5421010000

FIXED POINT CONSTANTS

T2789	0001

STATEMENT NUMBERS

T2749	07548	0010
T2709	07632	0001
T2679	07704	0002
T2649	09232	0014
T2619	08324	0003
T2609	09972	0020
T2599	08392	0004
T2589	08460	0005
T2579	09700	0019
T2569	08472	0006
T2559	08688	0009
T2549	08756	0011
T2539	09128	0013
T2519	08852	0012
T2509	09312	0015
T2499	09476	0017
T2479	09348	0016
T2469	09680	0021
T2449	09560	0018

TEMPORARY ACCUMULATORS

T2759	000

END OF MAPPING

Comments follow;

COMMENTS ON THE FORTRAN SYMBOL TABLE MAP

The data in the symbol table map is defined as follows:

END OF PROGRAM — The first available location after the last program instruction--excluding subroutines

START OF TABLE — The first available location preceding the expanded symbol table.

UNSUBSCRIPTED VARIABLES

Field address	Variable name

SUBSCRIPTED VARIABLES

Field address first element	Field address last element	Variable name	Number of rows

FLOATING POINT CONSTANTS

Field address	Constant

FIXED POINT CONSTANTS

Field address	Constant

STATEMENT NUMBERS

Field address of compiled branch	Instruction address of first instruction compiled for statement	Statement Number

TEMPORARY ACCUMULATORS

Field address	Accumulator number

Fields are addressed from their units position (high core position); instructions are addressed from their high order position (low core position).