



**TECHNICAL
DOCUMENTATION**

for

U N I C O D E

**Automatic Programming System for
Univac Scientific 1103A and 1105**

Volume II

April , 1961

PX 1790

Remington Rand Univac®

DIVISION OF SPERRY RAND CORPORATION

UNIVAC PARK, ST. PAUL 16, MINNESOTA

VOLUME I

	Page
Table of Contents	I-v
I. INTRODUCTION.	3
II. GENERAL	
1. UNICODE Service Routines.	7
2. Library Routines.	49
3. UNICODE System Tape Package	123
4. UNICODE Sample Coding	153
5. UNICODE Card Input.	163
6. Statistical Miscellany.	185
III. TRANSLATION AND CORRECTION	
1. UNICODE Sentinel Blocks	203
2. Tape Merge.	217
3. Translation Phase	
a. Translation Subroutines	291
b. Translators	434

VOLUME II

Table of Contents	II-v
III. TRANSLATION AND CORRECTION	
3. Translation Phase	
b. Translators (cont.)	569
IV. GENERATION PHASE	
1. Generation Set-up and Drum Loader . . .	949
2. Generation Subroutines.	959
3. Generators.	1013

VOLUME III

Table of Contents	III-v
IV. GENERATION PHASE	
3. Generators (cont.).	1193
V. ALLOCATION PHASE	
1. Segmentor	1461
2. Allocator	1551
3. Initialization Generator.	1607
VI. PROCESSING PHASE.	1671
VII. PROGRAM LISTING PHASE	1747

VOLUME II

TABLE OF CONTENTS

III. TRANSLATION AND CORRECTION

3. TRANSLATION PHASE

b. Translators

LIST String-Out (cont.)	
Coding	569
PRINT String-Out	
Write-Up	636
Flow Charts	638
Coding	640
IF String-Out	
Notes	646
Flow Charts	664
Coding.	692
VARY Translation Routine	
Notes	724
Flow Charts	729
Coding.	742
RESUME String-Out	
Flow Chart	773
Coding	774
JUMP String-Out	
Flow Charts	776
Write-Up	777
Coding	778
STOP String-Out	
Flow Chart	781
Coding	782
END OF TAPE String-Out	
Write-Up	783
Flow Charts	784
Coding	788
EXIT String-Out	
Flow Chart	792
Coding	793

START String-Out	
Flow Chart.	794
Coding.	795
EQUATION Translation Routine	
Notes	796
Flow Charts	806
Coding	
Regions	812
Setups and Subroutines.	816
Constants and Variables	823
Translation Left.	830
Translation Right	850
Error Prints.	893
PSEUDO-OPERATION HEADING Translator	
Write-Up.	921
Flow Charts	922
Coding.	928

IV. GENERATION PHASE

1. GENERATION SET-UP AND DRUM LOADER

Write-Up	949
Flow Charts	950
Coding	952

2. GENERATION SUBROUTINES

Regions	959
Control Generation	961
Computer Error	969
Sentence Heading Error Routine	973
Op Control Subroutine	
Write-Up	975
Flow Charts	977
Coding	980
Constant Call Word Routine for Generation	986
Routine to Get Call Word of Referenced Line Number	989
Line Number Check Routine C - Generation	992
KI Illegal Line Jump Check Routine - Generation	995
LS Library List Routine - Generation	999
Routine for Conversion of Excess-Three to Flex Code	
Write-Up	1002

Excess-Three to Flex Code Table	1003
Unityper Modifications for UNICODE	1005
Flow Chart	1007
Coding	1008

3. GENERATORS

START Generator	1013
JUMP Generation	
Write-Up	1016
Flow Chart	1017
Coding	1018
IF Generation	
Notes	1019
Flow Charts	1027
Coding	1033
PRINT Generation Routine	
Write-Up	1045
Flow Chart	1047
Coding	1048
COMPUTE Generation Routine	
Write-Up	1051
Flow Charts	1054
Coding	1060
VARY Generation Routine	
Write-Up	1072
Flow Charts.	1073
Coding	1084
RESUME Generation	1100
EXIT Generator	1105
TYPE Generator	
Flow Charts	1107
Coding	1115
LIST Generator	
Flow Charts.	1126
Coding	1136
READ Generator	
Flow Charts.	1145
Coding	1153
STOP Generator	
Flow Charts.	1161
Coding	1166
DIMENSION Generator	1174
PSEUDO-OPERATION HEADING	1176
END OF TAPE Generation	
Write-Up	1178
Flow Charts	1180
Coding	1183

List String-out Regions
(String-out Subroutine Regions also Required)

RE	LA4000
RE	LB4013
RE	LC4042
RE	MC4067
RE	LD4105
RE	MD4132
RE	ND4165
RE	LE4217
RE	LF4241
RE	NF4305
RE	LG4344
RE	LH4374
RE	LI4441
RE	LJ4503
RE	LQ4545
RE	LP4551
RE	LQ4612
RE	LR4642
RE	LS4653
RE	LT4721
RE	LU4762
RE	LV5011
RE	LW5065
RE	LX5114
RE	ZA5141
RE	ZB5204
RE	ZC5240
RE	ZD5271
RE	ZE5306
RE	ZF5347
RE	ZG5404
RE	ZH5444
RE	ZI5503
RE	ZJ5544
RE	ZK5602
RE	ZL5634
RE	ZM5664
RE	PB5675
RE	PC5730
RE	PD5773
RE	PE6025
RE	PG6040
RE	PH6072
RE	PI6130
RE	PJ6163
RE	PK6211
RE	PL6253
RE	PM6301
RE	PN6334

RE P06354
 RE PP6405
 RE PQ6445
 RE PR6476
 RE PT6541
 RE PU6603
 RE PV6646
 RE PW6716
 RE PX6743
 RE PY6767
 RE PZ7037
 RE SA7063
 RE SB7125
 RE SC7163
 RE SD7207
 RE SE7245
 RE SF7306
 RE SG7365
 RE SH7424
 RE SI7463
 RE SK7506
 RE SL7550

RE LY7607

RE LZ7660

RE WR50023

RE WP22

Indicators, Counters,
 Temporaries, etc.
 Heading List

Rewind List
 Maximum number of Call Words
 in Rewind List

List String-Out Routine

	IA	LA		Begin List String-out
0	MJ	0	CT	Exit to string-out control
1	TV	LW12	LP17	Preset add. 1st var. ind. word -1
2	TU	LW12	LP23	Preset inst. to preset add. in assem. blk.
3	TP	LW13	EW3	Preset add. of 1st C.W. in S.O. -1 in "C.W. to S.O." rtn.
4	TP	LV12	WL5	Zeroize variable count in String-out List.
5	RP	10074	LA7	
6	TP	LV12	LZ	Zeroize Heading List
7	TP	LV22	Q	} Set "list" indicator in fixed location 12.
10	QS	Q	12	
11	RP	10051	LB	
12	TP	LV12	LY	Zeroize assem. blk & temps.
	CA	LA13		
	IA	LB		
0	RJ	SY	SY1	Get next symbol (1st six char. sym. → A)
1	EJ	LX14	LB	Sym. = comma?
2	EJ	LX15	LB	Sym. = semi-colon?
3	EJ	LX12	ZA	1st var. = "tape"? Yes → Alarm #1
4	TP	SY7	Q	Var. ind. → Q
5	QJ	LC yes	LB6 no	Sym. = variable?
6	TP	SY11	Q	Digit or dec. pt. ind. → Q
7	QJ	LB10 yes	LB12 no	Sym. possibly = constant?
10	TP	SY12	Q	Sym. contains letter ind. → Q
11	QJ	LB12 no	ZA4 yes	Sym. = constant? Yes → Alarm #2
12	EJ	LX13	ZA13	Sym. = "Δ ."? Yes → Alarm #3
13	EJ	LX14	LB22 yes	Sym. = comma?
14	EJ	LX15	LB22 yes	Sym. = semi-colon?
15	EJ	LX16	ZA17 yes	Sym. = open parenthesis? Yes ⇒ Alarm #4
16	EJ	LX17	LB20	Sym. = close parenthesis?
17	MJ	0	ZA30	No ⇒ Alarm #6
20	IJ	LY44	LB22 no	Decrease parenthesis level; close parenthesis on level zero?
21	MJ	0	ZA24	Yes ⇒ Alarm #5
22	RJ	SY	SY1	Get next symbol
23	EJ	LX12	LB25	Sym. = tape? Yes → tape designation phase
24	MJ	0	LB4	No → ①
25	TP	LY44	A	
26	ZJ	ZA37 no	LF yes	"Tape" on parenthesis level zero?
	CA	LB27		No ⇒ printout #7

①

③

③A

Variable Phase (Fix & Fl. Pt. Var. Section)

④		IA	LC		
	0	RJ	RH	RH1	Check for legal variable sym.
	1	TP	SY2	LY16	Variable Sym. → temp.
	2	TP	LV2	A	6 in "V" → A
	3	TJ	SY5	MC6 yes	# char. in sym. > 6?
	4	RJ	TA	TA1	Var. in Combination List?
	5	MJ	0	LC17	Not in List
	6	TP	TA4	Q	Var. in Comb. List → CW → Q _v
	7	TP	Q	TF2	
	10	QT	LV5	A	1st two octal digits of CW → A
	11	EJ	LV5	LD	Subscripted var. (77---) CW?
	12	EJ	LV6	LE	Function (66---) CW?
	13	EJ	LV7	MC6	Floating point var. (65---) CW?
	14	EJ	LV10	MC2	Fixed point var. (64---) CW?
	15	TJ	LV43	ZB	Pseudo Op. (5----) CW? → Alarm #8
	16	MJ	0	ZB5	No; Library (4----) CW? ⇒ Alarm #9
	17	TP	LV11	TF	Number of words in file (3 in "u") →
					1st line file build-up
	20	TP	SY2	TF1	XS3 sym. → 2nd line file build-up
	21	TP	LV12	TF3	Format (zero) → 4th line file build-up
	22	RJ	TK	TK1	Obtain avail. last 3 digits for 65----
					or 64--- CW
	23	TP	SY10	Q	Fix. pt. ind. → Q
	24	QJ	MC yes	MC4 no	
		CA	LC25		

Non-Subs (Fix & Fl. Pt.) Var. Section

		IA	MC		
	0	AT	LV10	TF2	64--- C.W. → 3rd line file build-up
	1	RJ	TE	TE1	Add fix pt. var. file → Combination List
Fix.	2	RJ	LP	LP4	Adv. & ck. var. count & set fix pt.
Pt.					ind. bit.
Var.	3	MJ	0	MC7	
	4	AT	LV7	TF2	65--- C.W. → 3rd line file build-up
	5	RJ	TE	TE1	Add fl. pt. var. file → Combination List
Fl.	6	RJ	LP	LP1	Adv. & ck. var. count & set fl. pt. ind.
Pt.					bit
Var.	7	TP	TF2	Q	64--- or 65--- C.W. → Q _v (input var. list S.O.)
	10	RJ	L0	L01	64--- or 65--- C.W. → var. list S.O.
	11	RJ	LR	LR1	Store XS3 sym. for Hdg. Edit
	12	RJ	SY	SY1	Get next sym.
⑦A	13	EJ	LX14	LB22	Sym. = comma?
	14	EJ	LX15	LB22	Sym. = semi-colon?
	15	MJ	0	ZB12	No. ⇒ Alarm #10
		CA	MC16		

Subscripted Variable Section

⑧

	IA	LD	
0	RJ	LO	LO1
1	RJ	LP	LP1
2	RJ	LR	LR1
3	TV	TA5	LY
4	TP	LV12	LY45
5	RJ	SY	SY1
6	EJ	LX16	LD10
7	MJ	O	ZB21
10	TP	LV12	LY46

77----- CW → Var. List S.O.
 Adv. & check var. count & set Fl.
 Pt. Ind. bit
 Store XS3 sym. for Heading Edit
 No. of subscripts → Index ctr
 (c₁)
 Zeroize count of subscripts processed
 Get next sym.
 Sym. = open parenthesis?
 No ⇒ Alarm #11
 Set subscript parenthesis level
 = zero

⑧A
 ⑩
 ⑩A

11	RJ	LR	LR1
12	RJ	SY	SY1
13	TP	SY7	Q
14	QJ	MD yes	LD15 no
15	TP	SY11	Q
16	QJ	ND yes	LD17 no
17	EJ	LX17	ND20
20	EJ	LX13	ZB27
21	EJ	LX14	LD12
22	EJ	LX15	LD12
23	EJ	LX16	ZC
24	MJ	O	ZC6
	CA	LD25	

Store XS3 sym. for Hdg. Edit
 Get next sym.
 Variable ind. → Q
 Sym. = variable?
 Digit or dec. pt. ind. → Q
 Sym. possible = constant? (1st
 char. = digit or dec. pt.)
 Sym. = close parenthesis?
 Sym. = Δ .? Yes ⇒ Alarm #12
 Sym. = comma?
 Sym. = semi-colon?
 Sym. = open parenthesis? Yes
 → Alarm #13
 No ⇒ Alarm #14

11	0	IA	MD	ZC14	Variable = "Tape"? Yes $\Rightarrow$ Alarm #15
	1	EJ	LX12	RH1	Legal variable sym. (No; Alarm string-out subs)
	2	RJ	RH		6 in "v" $\rightarrow$ A
	3	TP	LV2	A	No. char. in sym. > 6?
	4	TJ	SY5	MD31 yes	Var. in Comb. List?
	5	RJ	TA	TA1	Not in list
	6	MJ	O	MD17	Var. in list $\Rightarrow$ CW $\rightarrow$ Q _v
	7	TP	TA4	Q	1st two octal digits of CW $\rightarrow$ A
	10	QT	LV5	A	64--- CW? (Fix Pt. Var.)
	11	EJ	LV10	ND7	4---- or 5---- CW? (Lib. Rtn. or Pseudo Op.)
	12	TJ	LV7	MD15	65--- CW? (fl. pt. var.) Yes $\Rightarrow$ Alarm #16
13	13	TJ	LV5	ZD	66--- C.W.? (function) Yes $\Rightarrow$ Alarm #17
	14	MJ	O	ZE	77--- C.W.? (subs. var.) $\Rightarrow$ Alarm #18
	15	TJ	LV43	ZF6	4---- C.W.? (pseudo op.) Yes $\Rightarrow$ Alarm #20
16	16	MJ	O	ZF	5---- CW? (library rtn.) $\Rightarrow$ Alarm #19
	17	TP	SY10	Q	Fix pt. var. ind. $\rightarrow$ Q
	20	QJ	MD21	ZC21 no	Sym.= fix pt. var? No $\Rightarrow$ Alarm #16
	21	TP	LV11	TF	No. of words in file (3 in "u") $\rightarrow$ 1st line file build-up
	22	TP	SY2	TF1	XS3 sym. $\rightarrow$ 2nd line file build-up
	23	TP	LV12	TF3	Format (zero) $\rightarrow$ 4th line file build-up
	24	RJ	TK	TK1	Obtain avail. last 3 digits for 64--- CW
	25	AT	LV10	TF2	64--- CW $\rightarrow$ 3rd line file build-up
	26	RJ	TE	TE1	Add fix pt. var. file $\rightarrow$ Combination List
	27	TP	TF2	Q	64--- CW $\rightarrow$ Q _v (input for var. list S.O.)
17	30	MJ	O	ND7	
	31	TP	SY10	Q	Fix pt. var. ind. $\rightarrow$ Q
	32	QJ	ND11 yes	ZC21 no	Sym.= fix pt. var.? No $\Rightarrow$ Alarm #16
		CA	MD33		

(18)	0	IA	ND			Superscript indicator → Q
	1	TP	SY13	Q		Sym.= Superscript const.; Yes ⇒ Alarm #21
		QJ	ZF14	yes	ND2 no	
	2	RJ	RD		RD1	Const.= fix pt.? (No; Alarm in string-out subs)
	3	TP	SY2		RS4	XS3 fix point constant → input conversion routine
	4	RJ	RS2		RS	Convert XS3 const. → octal
(18A)	5	TP	RS3		A	Octal constant → A
	6	RJ	GW		GW1	Constant → Const. Pool & Const. CW
	7	RJ	LO		LO1	Q _v 64---- or 67---- CW → var. list string-out
	10	RJ	LR		LR1	Store XS3 sym. for Hdg. Edit
	11	RA	LY45		LV	Adv. count of subscripts processed
	12	RJ	SY		SY1	Get next sym.
(18B)	13	EJ	LX14		LD11	Symbol = comma?
	14	EJ	LX15		LD11	Symbol = semi-colon?
	15	EJ	LX13		ZB27	Sym.= End sent. sym? Yes ⇒ Alarm #12
	16	EJ	LX17		ND20	Sym.= close parent.?
	17	MJ	0		ZF24	No ⇒ Alarm #22
	20	IJ	LY46		LD12	Is this close subscript parent.? (i.e., parent. on level zero)
(19)	21	RJ	LR		LR1	Store XS3 close parent. for Hdg. Edit
	22	TP	LY45		A	No. of valid subscripts processed → A
	23	EJ	LY		ND25	Correct no. of subscripts for subs. var.?
	24	MJ	0		ZG	No ⇒ Alarm #23
	25	RJ	SY		SY1	Get next sym.
	26	EJ	LX14		LB22	Sym.= comma?
(20)	27	EJ	LX15		LB22	Sym.= semi-colon?
	30	EJ	LX13		ZG5	Sym.= end sent. sym.? Yes ⇒ Alarm #24
	31	MJ	0		ZG12	No ⇒ Alarm #25
		CA	ND32			

Variable Phase (Function Section)

Subroutine to handle arguments of function. (Ref. by Alarm #17A & 26)	21	0	IA	LE		
		0	RJ	LO	LO1	66--- CW → Var. list string-out
		1	RJ	LP	LP1	Adv. & check var. count. & set ind. bit (fl. pt.)
		2	RJ	LR	LR1	Store XS3 symbol for Hdg. Edit
		3	RJ	SY	SY1	Get next sym.
		4	EJ	LX16	ZG16	Sym.= open parent.? Yes⇒ Alarm #26
		5	MJ	0	MC13	No⇒function symbol w/o arguments
		6	TV	LW26	LE11	
		7	TP	LV12	LY47	Set function parent. level → zero
		10	TP	SZ2	LY50	Function symbol → temp.
		11	RJ	SY	[SY1]	
		12	EJ	LX17	LE20 yes	Sym.= close parenthesis?
		13	EJ	LX16	LE17 yes	Sym.= open parenthesis?
		14	EJ	LX12	ZG27 yes	Sym.= "Tape"? Yes⇒ Alarm #27
		15	EJ	LX13	ZG33 yes	Sym.= space-period (Δ.)? Yes⇒ Alarm #28
		16	MJ	0	LE11	
		17	RA	LY47	LV1	Advance function parent. level by two
		20	IJ	LY47	LE11	Close parent. for arguments of function?
		21	MJ	0	[30000]	
			CA	LE22		

Tape Designation Phase

②②	0	IA	LF		
	1	RJ	SY	SY1	Get next symbol
	2	EJ	LX13	ZH	Sym.= " Δ ."? Yes ⇒ Alarm #29
	3	TP	SY11	Q	Digit or dec. pt. ind. → Q
	4	QJ	NF yes	LF4 no	Tape sym.= constant?
	5	TP	SY7	Q	Var. ind. → Q
	6	QJ	LF13 yes	LF6 no	Tape sym.= variable?
②③	7	EJ	LX14	ZH4 yes	Sym.= comma? Yes ⇒ Alarm #30
	10	EJ	LX15	ZH4 yes	Sym.= semi-colon? Yes ⇒ Alarm #30
	11	EJ	LX16	ZH10 yes	Sym.= open parent.? Yes → Alarm #31
	12	EJ	LX17	ZH14 yes	Sym.= close parent.? Yes → Alarm #32
②④	13	MJ	0	ZH20	No ⇒ Alarm #33
	14	RJ	RH	RH1	Sym.= legal variable? (Alarm in string-out subs)
	15	TP	LV2	A	6 in "v" → A
	16	TJ	SY5	LF42 yes	No. of char. in sym. > 6?
	17	RJ	TA	TA1	Var. in Combination List
	20	MJ	0	LF26	Not in list
	21	TP	TA4	Q	Var. in list ⇒ CW > Qv
	22	QT	LV5	A	1st two octal digits of CW → A
	23	EJ	LV10	LF40 yes	64--- CW? (fix pt. var.)
	24	TJ	LV43	ZH32 yes	4--- CW? Pseudo Op.) Yes ⇒ Alarm #35
②⑤	25	TJ	LV10	ZH25 yes	5--- CW? (Lib. Rtn.) Yes ⇒ Alarm #34
	26	MJ	0	ZI	No ⇒ Alarm #36
	27	TP	SY10	Q	Fix. pt. var. ind. → Q
	30	QJ	LF30 yes	ZI no	Sym.= fix pt. var.? No ⇒ Alarm #36
	31	TP	LV11	TF	No. of words in file (3 in "u") → 1st line file build-up
	32	TP	SY2	TF1	XS3 sym. → 2nd line file build-up
	33	TP	LV12	TF3	Format (zero) → 4th line file build-up
	34	RJ	TK	TK1	Obtain avail. last 3 digits for 64--- CW
	35	AT	LV10	TF2	64--- CW → 3rd line file build-up
	36	RJ	TE	TE1	Add. file to Comb. List
②⑥	37	SP	TF2	17	64--- CW → "u" of A
	40	MJ	0	NF12	
	41	SP	TA4	17	64--- CW → "u" of A
	42	MJ	0	NF12	
	43	TP	SY10	Q	Fix. pt. ind. → Q
	44	QJ	NF26 yes	ZI no	Sym.= fix. pt var.? No ⇒ Alarm #36
	45	CA	LF44		

Tape Designation Phase (cont.)

(27)	0	IA	NF		
	1	RP	20011	NF2	
	2	EJ	LX1	NF5	
	3	EJ	LX	ZM	
(28)	4	RJ	RD	RD1	
	5	MJ	0	ZI7	
	6	TP	A	RS4	
	7	RJ	RS2	RS	
(28A)	10	TP	RS3	A	
	11	RJ	GW	GW1	
	12	SP	Q	17	
	13	TP	A	WL4	
(29)	14	TU	WR	NF14	
	15	RP	[30000]	NF16	
	16	EJ	WR1	NF26 yes	
	17	SP	WR	0	
(30)	20	TJ	LW24	NF22 no	
	21	TP	A	Q	
	22	QJ	NF26	ZM4	
	23	RA	WR	LV15	
(30)	24	SA	LW25	0	
	25	TV	A	NF25	
	26	TP	WL4	[30000]	
	27	RJ	SY	SY1	
(30)	30	EJ	LX13	NF35	
	31	EJ	LX14	NF35	
	32	EJ	LX15	NF35	
	33	EJ	LX16	ZI16	
(30)	34	EJ	LX17	ZI22	
	35	MJ	0	ZI26	
	36	TP	UZ3	A	
	37	ZJ	ZI35 yes	LG no	
(30)	38	CA	NF37		
	39				
	40				
	41				

Tape sym.= XS3 {2,3,4,5,6,7,8,9,10
Tape number =1? Yes⇒Alarm #56
Tape number = fixed point? (Alarm in
string-out subs)
⇒ Alarm #37

Convert XS3 tape number.
Octal tape # → A
Tape number → const. pool & const. CW
⇒ Q_v
67---- CW (const. tape #) → "u" of A
Tape # CW (67---- or 64----) → S.O.
list.

Tape # CW in Rewind List
Count of Tape # CW's in Rewind List
→ A

Max. no. CW's in Rewind List (18₁₀)

(Q+) ⇒ Alarm #57 printout
(Q-) ⇒ Printout made previously
Adv. (1 in "u" & "v") count of CW's
in Rewind List
Form next avail. add. in Rewind List

Tape # CW → Rewind List

Get next symbol

Sym.= "Δ"?

Sym.= comma?

Sym.= semi-colon?

Sym.= open parent.? Yes → Alarm
#38

Sym.= close parent.? Yes → Alarm
#39

No ⇒ Alarm #40

Error count for sentence

Has there been error? Yes ⇒ Warning
#41

Heading Phase (Edit Variable Names)

32	0	IA	LG		
	1	TP	LV17	A	5 in "v" → A
	2	SS	WL5	1	(5 - # var.) x 2 → "v" of A
	3	AT	LW21	LY2	(A) + add. in Hdg. List → preset add.
	4	ST	LV42	LY3	1st var. in Hdg. List - 4
	5	TU	LW22	LG22	Address 1st col. Hdg. - 4 → temp.
33 34	6	TP	WL5	LY	Preset add. of 1st var. ind. word
	7	TP	LW23	LY1	Variable count → index counter (C ₁)
	8				Preset avail. add. in assem. blk. → Init. add.
	9	IJ	LY	LG21 no	All variable names edited?
	10	TP	SY2	A	
	11	EJ	LX13	LJ	Sym. = " Δ ."? Yes → End list string-out
	12	RJ	SY	SY1	Get next symbol
	13	EJ	LX14	LG12 yes	Sym. = comma?
	14	EJ	LX15	LG12 yes	Sym. = semi-colon?
	15	EJ	LX16	LG26	Sym. = open parent.? Yes → Title or column heading.
	16	EJ	LX17	ZJ	Sym. = close parent.? Yes → Alarm #42
	17	EJ	LX13	LJ	Sym. = " Δ ."? Yes → end list string-out
	20	MJ	0	ZJ4	No ⇒ Warning #43
	21	RA	LG22	LV22	Adv. "u" of NI → add. next var. ind. word
	22	TP	[30000]	LT1	Var. ind. word → input edit. var. subroutine
	23	RJ	LT	LT2	Edited variable → Hdg. List
	24	RA	LY1	LV32	Adv. avail. assem. blk. add. by 4 → add. next var.
	25	MJ	0	LG7	
	26	RP	10025	LH	
	27	TP	LV12	LY17	Zeroize assem. blk. (258 words)
		CA	LG30		

Heading Phase (Column Heading Section)

(35)	0	IA	LH		
	1	TP	GN4	A	Get next character → A
(42)	2	EJ	LV30	LI	Char.= open parent.? Yes → title section
	3	TU	LW22	LH21	Preset "u" of TP → Add. 1st var. ind. word - 1
(42A)	4	TP	LY3	LY2	Preset avail. add. in Hdg. List → add. 1st col. hdg. - 4
	5	RA	LY13	LV13	Set col. hdg. bit in hdg. ind. → 1
(43)	6	RJ	LP	LP27	Adv. & ck. col. hdg. count
	7	TP	LV12	LY	Set level indicator (C ₁) → zero
(44)	10	TP	GN4	A	
	11	EJ	LX20	LH16	Char. = close parent.
(45)	12	EJ	LV30	LH34	Char. = open parent.
	13	EJ	LV	LH41	Char. = "△"?
(46)	14	RJ	LQ	LQ1	Store XS3 character for hdg. edit.
	15	RJ	GN	GN1	Get next char. → A
(47)	16	MJ	0	LH10	
	17	IJ	LY	LH13 no	Close parent. on level zero?
(48)	20	TP	LW23	LY1	Preset avail. assem. blk. add. → initial add.
	21	RA	LH21	LV22	Adv. "u" of NI → Add. next var. ind. word
(49)	22	TP	[30000]	LT1	Var. ind. word → input edit col. hdg. routine
	23	TV	LY15	LT1	Char. count → input edit col. hdg. routine
(50)	24	RJ	LT	LT2	Edit col. hdg.
	25	RJ	GN	GN1	Get next char. (throw away close parent.)
(51)	26	RJ	SY	SY1	Get next sym.
	27	EJ	LX14	LH25	Sym.= comma?
(52)	30	EJ	LX15	LH25	Sym.= semi-colon?
	31	EJ	LX13	LJ	Sym.= "△."? → end list S.O.
(53)	32	EJ	LX16	LH36	Sym.= open parent.?
	33	EJ	LX17	ZJ13	Sym.= close parent.? Yes → Warning #44
(54)	34	MJ	0	ZJ17	No → Warning #45
	35	RA	LY	LV	Adv. parent. level by 1
(55)	36	MJ	0	LH13	
	37	TP	GN4	A	Char. → A
(56)	40	RP	10025	LH5	
	41	TP	LV12	LY17	Zeroize assem. blk.
(57)	42	RJ	LQ	LQ1	Store XS3 char. for Hdg. Edit
	43	RJ	GN	GN1	Get next char.
(58)	44	EJ	LX21	ZJ25	Char. = period? Yes ⇒ Warning #46
	45	MJ	0	LH10	
(59)	46	CA	LH45		

Heading Phase (Title Section)

	IA	LI		
0	TP	LV12	LY	Set level ind. = zero
1	RA	LY13	LV33	Set title bit in hdg.indicator → 1
2	TP	LV23	LY11	Preset index (C ₂)
3	TV	LV20	LQ10	Preset char. shift
4	TP	LW4	LQ11	Preset initial add. in assem. blk.
5	TP	LV12	LY15	Zeroize char. count.
③⑥	6	RJ	GN	Get next char.
③⑥④	7	EJ	LX20	Char.= close parent.?
③⑧	10	EJ	LV30	Char.= open parent.?
	11	EJ	LV	Char.= " Δ "?
③⑦	12	RJ	LQ	Store XS3 character for title edit
	13	MJ	0	→ ③⑥
	14	RA	LY	Adv. parent. level by 1
	15	MJ	0	
	16	IJ	LY	
③⑨	17	RJ	GN	Close parent. on level zero?
	20	EJ	LX20	Get next char.
	21	MJ	0	Char.= close parent.
	22	RJ	LS	No ⇒ Alarm #47
④①	23	RJ	GN	Edit and store title for edit
	24	RJ	SY	Get next char. (Throw away close parent.)
	25	EJ	LX14	Get next sym.
	26	EJ	LX15	Sym.= comma?
	27	EJ	LX16	Sym.= semi-colon?
	30	EJ	LX13	Sym.= open parent.?
	31	EJ	LX17	Sym.= " Δ ."? → end list S.O.
	32	MJ	0	Sym.= close parent.? Yes → Printout #48
	33	TP	GN4	No ⇒ Printout #49
	34	RP	10025	
	35	TP	LV12	→ ④②
	36	RJ	LQ	Zeroize assem. blk.
	37	RJ	GN	Store char. (Δ) for title edit
	40	EJ	LX21	Get next char.
	41	MJ	0	Char.= period? Yes ⇒ Warning #50
		CA	LI42	

End List String-out

④9	0	IA	LJ		
		TP	EW3	LY15	Add. of last entry in string-out → temp.
	1	RA	LY15	LV15	1 in "u" & "v" adv. → initial add. for headings in string-out.
	2	TP	LY13	Q	Hdg. ind. → Q
⑤0	3	QJ	LJ4 yes	LJ5 no	Are there column hdgs.?
	4	QJ	LJ6 yes	LJ13 no	Is there title? (w/col. hdgs.)
	5	QJ	LJ16 yes	LJ27 no	Is there title? (w/o col. hdgs.)
	6	TV	LY15	LJ10	
Title, col. hdgs. & var. names	7	RP	30074	LJ11	
	10	TP	LZ	[30000]	Hdg. list (title-col. hdgs.-var. names) → S.O.
	11	TP	LV37	WL6	Hdg. count (60 ₁₀) → S.O.
	12	MJ	0	LJ33	
⑤1 Col. hdgs. & var. names	13	TV	LY15	LJ15	
	14	RP	30050	LJ25	
	15	TP	LZ24	[30000]	Hdg. List (col. hdg. and var. names) → S.O.
	16	TV	LY15	LJ20	
Title & var. names	17	RP	30024	LJ21	
	20	TP	LZ	[30000]	Title → S.O.
	21	TV	LJ20	LJ24	
	22	RA	LJ24	LV42	Adv. by 20 ₁₀ → Add. following title in string-out
⑤2	23	RP	30024	LJ25	
	24	TP	LZ50	[30000]	Var. names → S.O.
	25	TP	LV41	WL6	Hdg. count (40 ₁₀) → S.O.
	26	MJ	0	LJ33	
⑤3 Var. names	27	TV	LY15	LJ31	
	30	RP	30024	LJ32	
	31	TP	LZ50	[30000]	Hdg. List (var. names) → S.O.
	32	TP	LV42	WL6	Hdg. count (20 ₁₀) → S.O.
⑤4	33	TU	LY15	LJ34	Preset "u" of NI → add. of 1st word of hdgs. in string-out
	34	CC	30000	LX22	Fast feed 1 sym. → 1st char. of hdgs. in string-out
	35	RS	LY15	LW16	No. of words in S.O. w/o hdgs. → "u" & "v" of A
	36	AT	WL6	Q	No. of words in S.O. including hdgs. → Q
	37	QT	LV40	WL	Word count → "v" of 1st word of S.O.
	40	RJ	WT	WT1	String-out → tape
	41	MJ	0	LA	→ String-out Exit
		CA	LJ42		

Adv. and Ck. Var. (Col. Hdg.) Count Subroutine

	0	IA	LP		
		MJ	0	[30000]	
Fl. Pt.	1	TU	LW	LP17	Set up inst. for fl. pt. ind.
Ent.	2	TU	LW2	LQ3	Preset add. of fl. pt. char. limit
	3	MJ	0	LP6	
Fix Pt.	4	TU	LW1	LP17	
Ent.	5	TU	LW3	LQ3	Preset add. of fix pt. char. limit
(55)	6	SP	WL5	0	Var. count $\rightarrow$ A
Delete	7	TJ	LV17	LP15 yes	5 > # variables?
sym.	10	MJ	0	ZK17	No $\Rightarrow$ Printout #51
before	11	RJ	SY	SY1	Get next sym.
"Tape".	12	EJ	LX12	LF	Sym. = tape? $\rightarrow$ tape designation phase
(56)	13	EJ	LX13	ZA13	Sym. = " Δ ."? Yes $\Rightarrow$ Alarm #3
	14	MJ	0	LP11	
(57)	15	AT	LV	WL5	Adv. var. count $\rightarrow$ list string-out
	16	RA	LP17	LV	
Preset	17	TP	[30000]	[30000]	Fix or fl. pt. ind. $\rightarrow$ var. ind. word
at be-	20	SP	LP17	17	
gin	21	TU	A	LQ2	Preset add. of char. count (var. ind. word)
list S.O.	22	RA	LP23	LV22	Adv. NI to preset next add. in assem. blk.
Preset	23	TP	[30000]	LQ11	Preset add. in assem. blk.
at	24	TV	LW20	LQ1	Preset ent. $\rightarrow$ store XS3 char. for var. name
begin					
list	25	TU	LQ11	ZL2	Add in assem. blk. $\rightarrow$ trans. inst. for warning print
S.O.					
	26	MJ	0	LP36	
Col.	27	SP	LY14	0	Col. hdg. count $\rightarrow$ A
Hdg.	30	TJ	WL5	LP32 yes	# Variables > # col. hdgs.?
Ent.	31	MJ	0	ZK26	No $\Rightarrow$ Warning #52
	32	AT	LV	LY14	Adv. col. hdg. count by 1
	33	TP	LW4	LQ11	Preset add. in assem. blk.
	34	TP	LV12	LY15	Zeroize char. count
	35	TV	LW11	LQ1	Preset ent. $\rightarrow$ store XS3 char. for column hdg.
(58)	36	TP	LV23	LY11	Preset index (C2)
	37	TV	LV20	LQ10	Preset char. shift
	40	MJ	0	LP	Exit
		CA	LP41		

Call Word $\rightarrow$ Var. List String-out

	IA	LO		
0	MJ	0	[30000]	
1	TP	Q	EW2	Call word $\rightarrow$ "v" of EW2
2	RJ	EW	EW1	Call word $\rightarrow$ string-out
3	MJ	0	LO	
	CA	L04		

Store XS3 Char. for Hdg. Edit

⑥1 Preset in var. count rtn. Var. Ent. Preset in var. count rtn.	0	IA	LQ		Input: XS3 char. in "v" of A
	1	MJ	0	[30000]	"v" preset → LQ2 or LQ17
	2	TP	[30000]	A	Char. count → A
	3	TJ	[30000]	LQ5	23 ₁₀ (27 ₈) > # char.?
	4	MJ	0	ZL	No ₁₀ ⇒ Warning #53
	5	TU	LQ2	LQ6	Preset "u" of NI → char. count add.
	6	RA	[30000]	LV	Adv. char. count by 1 in "v"
⑤9 Preset in Var. count rtn.	7	RS	LQ10	LV2	Decrease shift count
	10	SP	Q	[30000]	Position char. in A
Preset in var. count rtn.	11	AT	[30000]	[30000]	Char. → current word in assem. blk.
⑥0 Col. → Hdg. Ent. Title → Ent.	12	IJ	LY11	LQ	Current word full? (index preset by var. count routine)
	13	RA	LQ11	LV15	Adv. current assem. blk. address
	14	TV	LV16	LQ10	Reset shift count
	15	TP	LV17	LY11	Reset index
	16	MJ	0	LQ	
	17	TP	LY15	A	Char. count → A
	20	TJ	LV14	LQ25 yes	23 ₁₀ (27 ₈) > # char.?
	21	MJ	0	ZL10	No ₁₀ ⇒ Warning #54
	22	[TP	LY15	A]	Title char. count → A (reset → MJ-0-LQ after printout; not preset, transferred from drum before operating)
	23	TJ	LV34	LQ25 yes	119 ₁₀ (167 ₈) > # char.?
⑥2	24	MJ	0	ZL20	No ₁₀ ⇒ Warning #55
	25	RA	LY15	LV	Adv. title (col. hdg.) char. count by 1 in "v"
	26	TP	GN4	Q	
	27	MJ	0	LQ7	
		CA	LQ30		

Store XS3 Sym. for Heading Edit

⑥3	0	IA	LR		Input: XS3 sym. in SY2, # char. in SY5
	1	MJ	0	[30000]	
	2	TP	SY5	LY12	# Char in sym. → index ctr. (C ₃)
	3	TP	SY2	LY3	XS3 sym. → temp.
	4	IJ	LY12	LR5	All char. trans. → Assem. blk.
	5	MJ	0	LR	
	6	LQ	LY3	6	Next XS3 char. in sym. → "v" of Q
	7	QT	LV52	Q	XS3 char. → "v" of Q
	8	RJ	LQ	LQ1	Store XS3 char. for Hdg. Edit
	9	MJ	0	LR3	
	10	CA	LR11		

Edit Title Subroutine

64	0	IA	LS		
		MJ	0	[30000]	
	1	TV	LW14	LS31	Preset hdg. list add. → middle add. of title
	2	SP	LY15	0	Char. count → A
	3	DV	LV31	Q	#char.; # full words in 1/2 title → Q
					12
65	4	ZJ	LS5 yes	LS6 no	Is there partial word?
	5	RA	Q	LV	Adv. Q by 1 ⇒ total # words in 1/2 title → Q _v
	6	RS	LS31	Q	Decrease hdg. list add. → add. initial title word in hdg. list
	7	SP	Q	1	(# words in 1/2 title) * 2 = total # words in title → A
	10	ST	LV	LY11	# Words in title - 1 → index ctr. (C ₂)
	11	SP	LY15	0	# Char. → A
66	12	EJ	LV34	LS44	# Char. = 119
	13	SA	LV26	0	# Char. + 3 ¹⁰ → A
	14	DV	LV1	A	#char. + 3 → A
					2
	15	DV	LV2	Q	# Char. to shift → A
	16	ZJ	LS17 no	LS37 yes	# Char. to shift = zero? No ⇒ shift 1 char.
67	17	TJ	LV1	LS23	2 > # char. to shift?
	20	TU	LW23	LS27	Preset "SP" → add. of 1st word in assem. blk.
	21	TU	LW4	LS30	Preset "SA" → add. of 2nd word in assem. blk.
	22	MJ	0	LS25	
	23	TU	LW4	LS27	Preset "SP" → add. of 2nd word in assem. blk.
	24	TU	LW26	LS30	Preset "SA" → add. of 3rd word in assem. blk.
68	25	MP	A	LV2	(# char. to shift) x 6 = shift count
	26	TV	A	LS30	Preset shift count in "SA"
	27	SP	[30000]	44	Current title word from assem. blk. → A
	30	SA	[30000]	[30000]	Position edited title word in A _L
	31	LT	0	[30000]	Edited title word → hdg. list add.
	32	RA	LS27	LV22	Adv. "u" of "SP" by 1
	33	RA	LS30	LV22	Adv. "u" of "SA" by 1
	34	RA	LS31	LV	Adv. "v" of "LT" by 1
	35	IJ	LY11	LS27 no	All of edited title → hdg. list?
	36	MJ	0	LS	
	37	SP	Q	20	# words edited title → "u" of A
	40	AT	LW15	LS42	Add. "w" to dummy repeat
	41	TV	LS31	LS43	Preset add. of initial title word in hdg. list
	42	[RP	30000	LS]	
	43	TP	LY20	[30000]	Trans. edited title → hdg. list

44	RP	30024	LJ
45	TP	LY20	LZ
	CA	LS46	

Trans. 119₁₀ char. title → hdg. list.

Edit Variable (Col. Hdg.) Subroutine

⑧3	0	IA	LT		Exit
	1	MJ	0	[30000]	Input
	2	[O	30000	30000]	Adv. avail. add. hdg. list by 4 in "u"
		RA	LY2	LV23	→ add. next var. in hdg. list
	3	TV	A	LU15	Preset "v" of trans. inst. → add. next
					var. in hdg. list.
	4	TU	LY1	LU13	Preset "SP" inst. → add. next var. - 1
					in assem. blk.
	5	TU	LY1	LU14	Preset "SA" inst. → add. next var. - 1
					in assem. blk.
	6	RA	LU14	LV22	Adv. "u" of "SA" by 1 → add. next var.
					- in assem. blk.
	7	TP	LT1	Q	Var. ind. word → Q
	10	QT	LV40	LT1	# Char. → A _v & input line
⑥9	11	EJ	LV25	LU23 yes	# Char. = 23 ₁₀ ^v (278)?
	12	QJ	LT24 no	LT13 yes	Variable floating pt. quan?
	13	TJ	LV27	LT16 yes	13 ₁₀ > # char.?
	14	TP	LV26	LY11	Index = 3 to trans. 4 words → hdg. list
	15	MJ	0	LT20	→ ⑦1
⑦0	16	TP	LV	LY11	Index = 1 to trans. 2 words → hdg. list
	17	RA	LU15	LV	Adv. "v" trans. inst. by 1 → next add.
					hdg. list
⑦1	20	TP	LT1	A	# Char. → A
	21	SA	LV26	0	# Char. + 3 → A
	22	DV	LV1	A	# Char. + 3; Quot → A
					→ ⑦2
⑦2	23	MJ	0	LU2	→ ⑦7
	24	TJ	LV31	LT33 yes	12 ₁₀ > # char.?
	25	TJ	LV53	LT30 yes	19 ₁₀ > # char.?
	26	TP	LV26	LY11	Index = 3 to trans. 4 words → hdg. list
	27	MJ	0	LT35	→ ⑦5
⑦3	30	TP	LV1	LY11	Index = 2 to trans. 3 words → hdg. list
	31	RA	LU15	LV	Adv. "v" trans. inst. by 1 → next add.
					hdg. list
					→ ⑦5
⑦4	32	MJ	0	LT35	→ ⑦5
	33	TP	LV	LY11	Index = 1 to trans. 2 words → hdg. list
	34	RA	LU15	LV1	Adv. "v" of trans. inst. by 2 → add. in
					hdg. list.
⑦5	35	TP	LT1	A	# Char. → A _v
	36	TJ	LV30	LU yes	15 ₁₀ > # char.?
	37	SA	LV	0	# Char. + 1 → A
	40	MJ	0	LU2	
		CA	LT41		

Edit Var. (Col. Hdg.) Subroutine (cont.)

		IA	LU		
⑦⑥	0	DV	LV1	A	#char/2; Quot → A
	1	LV1	LV1	0	Quot + 2 → A
⑦⑦	2	DV	LV2	Q	(A)/6 ⇒ Rem. = # char. to shift → A
	3	ZJ	LU4 no	LU23 yes	# Char. to shift = zero?
	4	MP	A	LV2	# Char. to shift x 6 = shift count → A
⑦⑧	5	TV	A	LU14	Preset shift count in "v" of SA inst.
	6	TJ	LV24	LU11 yes	7 > shift count? (i.e., # char. to shift = 1)
	7	TP	LV12	A	Zero → A
	10	MJ	0	LU14	
	11	RA	LU13	LV22	Adv. "u" of SP by 1 → add. of 1st var. word
	12	RA	LU14	LV22	Adv. "u" of SA by 1 → add. of 2nd var. word
⑦⑨	13	SP	[30000]	44	Variable word from assem. blk. → A _L
⑧⑩	14	SA	[30000]	[30000]	Add. next word to A _R & shift to position in A _L
	15	LT	0	[30000]	Edited word from assembly block → heading list
	16	RA	LU13	LV22	Adv. "u" of SP by 1
	17	RA	LU14	LV22	Adv. "u" of SA by 1
	20	RA	LU15	LV	Adv "v" of trans. inst. by 1 → next add. in hdg. list
⑧⑪	21	IJ	LY11	LU13	All words trans. from assem. blk. → hdg. list
	22	MJ	0	LT	
	23	TU	LU14	LU26	Add. 1st word of var. in assem. blk. → "u" of TP
	24	TV	LU15	LU26	Add. for variable in hdg. list → "v" of TP
	25	RP	30004	LT	
	26	TP	[30000]	[30000]	Trans. words from assem. blk. → hdg. list w/o editing.
		CA	LU27		

Fixed Constants

	IA	LV		
0	0	0	1	XS3 space char. (Δ)
1	0	0	2	
2	0	0	6	
3	0	0	11	
4	0	0	12	
5	0	0	77000	
6	0	0	66000	
7	0	0	65000	
10	0	0	64000	
11	0	3	3	
12	0	0	0	Fl. pt. ind.
13	40	0	0	Fix pt. ind.
14	0	0	27	Fl. pt. char. limit (23 = maximum # char.)
15	0	1	1	
16	0	0	44	
17	0	0	5	
20	0	0	36	
21	40	0	27	
22	0	1	0	
23	0	0	4	
24	0	0	7	
25	0	0	27	
26	0	0	3	
27	0	0	15	
30	0	0	17	XS3 open parent. character
31	0	0	14	
32	0	4	4	
33	20	0	0	
34	0	0	167	
35	0	0	170	
36	0	24	24	
37	0	0	74	
40	0	0	77777	
41	0	0	50	
42	0	0	24	
43	0	0	50000	
44	0	55000	0	
45	0	550	0	
46	0	55740	0	
47	0	557	40000	
50	17	0	0	
51	0	0	40000	
52	0	0	77	
53	0	0	23	
	CA	LV54		

Relative Constants

	IA	LW		
0	0	LV12	0	Add. fl. pt. ind.
1	0	LV13	0	Add. fix. pt. ind.
2	0	LV14	0	Add. fl. pt. character limit
3	0	LV21	0	Add. fix pt. character limit
4	AT	LY20	LY20	To preset add. in assembly block for XS3 sym. or character store routine
5	AT	LY24	LY24	
6	AT	LY30	LY30	
7	AT	LY34	LY34	
10	AT	LY40	LY40	
11	0	0	LQ17	
12	0	LW3	LY3	To preset inst. to present add. in assembly block. To preset add. of first var. ind. word - 1
13	0	WL6	WL6	To preset EW3 → Add. of 1st CW in S.O. - 1
14	0		LZ12	Middle add. of title in hdg. list.
15	RP	30000	LS	
16	0	WL	WL	Initial add. in S.O. list
17	0	0	LQ	
20	0	0	LQ2	
21	0	0	LZ44	To preset add. in hdg. list.
22	0	LY3	0	To preset add. of var. ind. word
23	0	LY17	LY17	Initial address in assem. block
24	0	WP20000	WP	WP = max. no. of tape CW's in Rewind List
25	0	0	WR	Initial add. in Rewind List
26	0	LY21	SY1	
	CA	LW27		

XS3 Codes

	IA	LX			
0	04	77777	77777	1	} Servo numbers
1	05	77777	77777	2	
2	06	77777	77777	3	
3	07	77777	77777	4	
4	10	77777	77777	5	
5	11	77777	77777	6	
6	12	77777	77777	7	
7	13	77777	77777	8	
10	14	77777	77777	9	
11	04	03777	77777	10	
12	66	24523	07777	TAPE	
13	01	22777	77777	△	
14	21	77777	77777	Comma symbol	
15	23	77777	77777	Semi-colon symbol	
16	17	77777	77777	Open parent. symbol "("	
17	43	77777	77777	Close parent. symbol ")"	
20	00	00000	00043	Close parent. char.	
21	00	00000	00022	Period char.	
22	37	00000	00000	Fast feed 1 sym. (packed to left w/zero fill)	
23	00	00000	00021	Comma character.	
24	00	00000	00023	Semi-colon character	
	CA	LX25			

		IA	ZA		
Alarm	0	RJ	WA	WA1	
#1	1	TP	PB	UP3	
	2	RJ	UP2	UP	
	3	MJ	0	LA	→ Exit
Alarm	4	RJ	WA	WA1	
#2	5	TP	SY2	PB22	
	6	TP	SY3	PB23	
	7	TP	SY4	PB24	
	10	TP	PB17	UP3	
	11	RJ	UP2	UP	
	12	MJ	0	LB22	→ Exit
Alarm	13	RJ	WA	WA1	
#3	14	TP	PC	UP3	
	15	RJ	UP2	UP	
	16	MJ	0	LA	→ Exit
Alarm	17	RJ	WA	WA1	
#4	20	TP	PC17	UP3	
	21	RJ	UP2	UP	
	22	RA	LY44	LV	Adv. Parenthesis Level
	23	MJ	0	LB22	→ Var. Phase
Alarm	24	RJ	WA	WA1	
#5	25	TP	PD	UP3	
	26	RJ	UP2	UP	
	27	MJ	0	LB22	→ Var. Phase
Alarm	30	RJ	WA	WA1	
#6	31	TP	SY2	PD22	
	32	TP	SY3	PD23	
	33	TP	SY4	PD24	
	34	TP	PD17	UP3	
	35	RJ	UP2	UP	
	36	MJ	0	LB22	→ Var. Phase
Alarm	37	RJ	WA	WA1	
#7	40	TP	PE	UP3	
	41	RJ	UP2	UP	
	42	MJ	0	LF	→ Tape Designation Phase
		CA	ZA43		

		IA	ZB		
Alarm	0	RJ	WA	WA1	Print hdg. & set error bit
#8	1	TP	SY2	PG6	Pseudo op. sym. → printout
	2	TP	PG	UP3	
	3	RJ	UP2	UP	Printout #8
	4	MJ	0	LB22	
Alarm	5	RJ	WA	WA1	Print hdg. & set error bit
#9	6	TP	SY2	PG23	Library rtn. sym. → printout
	7	TP	PG16	UP3	
	10	RJ	UP2	UP	Printout #9
	11	MJ	0	LB22	
Alarm	12	RJ	WA	WA1	Print hdg. & set error bit
#10	13	TP	SZ2	PH7	
	14	TP	SZ3	PH10	} Symbol → Printout
	15	TP	SZ4	PH11	
	16	TP	PH	UP3	
	17	RJ	UP2	UP	Printout #10
	20	MJ	0	ZB25	
Alarm	21	RJ	WA	WA1	Print hdg. & set error bit
#11	22	TP	LY16	PH20	Subs. var. symbol → printout
	23	TP	PH13	UP3	
	24	RJ	UP2	UP	Printout #11
	25	TP	SY2	A	
	26	MJ	0	LB23	
Alarm	27	RJ	WA	WA1	Print hdg. & set error bit
#12	30	TP	LY16	PI11	Subs. var. sym. → printout
	31	TP	PI	UP3	
	32	RJ	UP2	UP	
	33	MJ	0	LA	→ Exit
	34	CA	ZB34		

		IA	ZC		
Alarm	0	RJ	WA	WA1	Print hdg. and set error bit
#13	1	TP	LY16	PI31	Subs. Var. sym. → printout
	2	TP	PI21	UP3	
	3	RJ	UP2	UP	Printout #13
	4	RA	LY46	LV	Adv. subs. parent. level by 1
	5	MJ	0	LD12	
Alarm	6	RJ	WA	WA1	Print hdg. & set error bit
#14	7	TP	SY2	PJ3	Illegal sym. → printout
	10	TP	LY16	PJ12	Subs. var. sym. → printout
	11	TP	PJ	UP3	
	12	RJ	UP2	UP	Printout #14
	13	MJ	0	LD12	
Alarm	14	RJ	WA	WA1	Print hdg. & set error bit
#15	15	TP	LY16	PJ24	Subs. var. sym. → printout
	16	TP	PJ14	UP3	
	17	RJ	UP2	UP	Printout #15
	20	MJ	0	LF	
Alarm	21	RJ	WA	WA1	Print hdg. and set error bit
#16	22	TP	SY2	PK15	} Symbol → Printout
	23	TP	SY3	PK16	
	24	TP	SY4	PK17	
	25	TP	LY16	PK4	
	26	TP	PK	UP3	
	27	RJ	UP2	UP	
	30	MJ	0	LD12	
		CA	ZC31		
		IA	ZD		
Alarm	0	RJ	WA	WA1	Print hdg. & set error bit
#17	1	TP	SY2	PK25	Function symbol → printout
	2	TP	LY16	PK32	Sub. var. sym. → printout
	3	RJ	SY	SY1	Get next symbol
	4	EJ	LX16	ZD11 yes	Sym. = open parenthesis? (i.e., are there arguments w/function)
	5	TP	PK21	UP3	Parameter for #17 → Uniprint
	6	RJ	UP2	UP	Printout #17
	7	TP	SY2	A	Current sym. → A
	10	MJ	0	LD13	→ Subs. var. section
	11	TP	PK22	UP3	Parameter for #17A → Uniprint
	12	RJ	UP2	UP	Printout #17A
	13	RJ	LE21	LE6	Delete arguments of function
	14	MJ	0	LD12	→ Subs. var. section
		CA	ZD15		

Alarm	0	IA	ZE		
#18	1	RJ	WA	WA1	Print hdg. & set error bit
	2	TP	SY2	PL6	Latest subs. var. → printout
	3	TP	LY16	PL14	Prior subs. var. → printout
	4	RJ	SY	SY1	Get next symbol
	5	EJ	LX16	ZE11 yes	Symbol = open parenthesis
	6	TP	PL	UP3	Parameter for #18 → printout
	7	RJ	UP2	UP	Printout #18
	10	TP	SY2	A	Current sym. → A
	11	MJ	0	LD13	→ Subs. var. section
	12	TP	PL1	UP3	Parameter for #18A → Printout
	13	RJ	UP2	UP	Printout #18A
	14	TP	LV12	LY47	Set parenthesis level = zero
	15	TP	SZ2	LY50	Latest subs. var. sym. → temp.
(14)	16	RJ	SY	SY1	Get next symbol
	17	EJ	LX17	ZE23 yes	Sym. = close parenthesis?
	20	EJ	LX16	ZE25	Sym. = open parenthesis?
	21	EJ	LX12	ZE27 yes	Sym. = "tape"? yes ⇒ printout #15
			LX13	ZE34 yes	Sym. = space-period (Δ .)? yes → printout #12
	22	MJ	0	ZE15	
	23	IJ	LY47	ZE15 no	Close parent. for subscripts? (level zero)
	24	MJ	0	LD12	
	25	RA	LY47	LV	Adv. parent. level by 1
	26	MJ	0	ZE15	
	27	RJ	WA	WA1	
	30	TP	LY50	PJ24	
	31	TP	PJ14	UP3	
	32	RJ	UP2	UP	Printout #15
	33	MJ	0	LF	→ Tape designation phase
	34	RJ	WA	WA1	
	35	TP	LY50	PI11	
	36	TP	PI	UP3	Printout #12
	37	RJ	UP2	UP	
	40	MJ	0	LA	→ Exit
		CA	ZE41		

		IA	ZF		
Alarm	0	RJ	WA	WA1	Print hdg. & set error bit
#19	1	TP	SY2	PM5	Lib. rtn. sym. → printout
	2	TP	LY16	PM13	
	3	TP	PM	UP3	
	4	RJ	UP2	UP	Printout #19
	5	MJ	0	LD12	
Alarm	6	RJ	WA	WA1	Print hdg. & set error bit
#20	7	TP	SY2	PM23	
	10	TP	LY16	PM31	
	11	TP	PM15	UP3	
	12	RJ	UP2	UP	Printout #20
	13	MJ	0	LD12	
Alarm	14	RJ	WA	WA1	Print hdg. & set error bit
#21	15	TP	SY2	PN14	
	16	TP	SY3	PN15	
	17	TP	SY4	PN16	
	20	TP	LY16	PN4	
	21	TP	PN	UP3	
	22	RJ	UP2	UP	
	23	MJ	0	LD12	
Alarm	24	RJ	WA	WA1	Print hdg. & set error bit
#22	25	TP	SZ2	P013	
	26	TP	SZ3	P014	
	27	TP	SZ4	P015	
	30	TP	LY16	P07	
	31	TP	P0	UP3	
	32	RJ	UP2	UP	
	33	TP	SY2	A	
	34	MJ	0	LD13	
		CA	ZF35		

		IA	ZG		
Alarm	0	RJ	WA	WA1	
#23	1	TP	LY16	P027	
	2	TP	P017	UP3	
	3	RJ	UP2	UP	
	4	MJ	0	LB22	→ Var. phase
Alarm	5	RJ	WA	WA1	
#24	6	TP	LY16	PP15	
	7	TP	PP	UP3	
	10	RJ	UP2	UP	
	11	MJ	0	LA	→ Exit
Alarm	12	RJ	WA	WA1	
#25	13	TP	LY16	PP36	
	14	TP	PP23	UP3	
	15	MJ	0	ZB24	
Alarm	16	RJ	WA	WA2	
#26	17	TP	SZ2	PQ5	
	20	TP	PQ	UP3	
	21	RJ	UP2	UP	
	22	RJ	LE11	LE7	} Save arguments of function
	23	RJ	LR	LR1	
	24	RJ	LE21	SY1	
	25	RJ	LR	LR1	
	26	MJ	0	LB22	
Alarm	27	RJ	WA	WA1	
#27	30	TP	LY50	PQ27	
	31	TP	PQ15	UP3	
	32	MJ	0	LB22	
Alarm	33	RJ	WA	WA1	
#28	34	TP	LY50	PR12	
	35	TP	PR	UP3	
	36	RJ	UP2	UP	
	37	MJ	0	LA	→ Exit
		CA	ZG40		

		IA	ZH		
Alarm	0	RJ	WA	WA1	
#29	1	TP	PR21	UP3	
	2	RJ	UP2	UP	
	3	MJ	0	LA	→ Exit
Alarm	4	RJ	WA	WA1	
#30	5	TP	PT	UP3	
	6	RJ	UP2	UP	
	7	MJ	0	LA	→ Exit
Alarm	10	RJ	WA	WA1	
#31	11	TP	PT21	UP3	
	12	RJ	UP2	UP	
	13	MJ	0	LA	→ Exit
Alarm	14	RJ	WA	WA1	
#32	15	TP	PU	UP3	
	16	RJ	UP2	UP	
	17	MJ	0	LA	→ Exit
Alarm	20	RJ	WA	WA1	
#33	21	TP	SY2	PU30	
	22	TP	PU21	UP3	
	23	RJ	UP2	UP	
	24	MJ	0	LA	→ Exit
Alarm	25	RJ	WA	WA1	
#34	26	TP	SY2	PV12	
	27	TP	PV	UP3	
	30	RJ	UP2	UP	
	31	MJ	0	LA	→ Exit
Alarm	32	RJ	WA	WA1	
#35	33	TP	SY2	PV36	
	34	TP	PV24	UP3	
	35	RJ	UP2	UP	
	36	MJ	0	LA	→ Exit
	37	CA	ZH37		

		IA	ZI		
Alarm	0	RJ	WA	WA1	
#36	1	TP	SY2	PW16	
	2	TP	SY3	PW17	
	3	TP	SY4	PW20	
	4	TP	PW	UP3	
	5	RJ	UP2	UP	
	6	MJ	0	LA	→ Exit
Alarm	7	RJ	WA	WA1	
#37	10	TP	SY2	PX15	
	11	TP	SY3	PX16	
	12	TP	SY4	PX17	
	13	TP	PX	UP3	
	14	RJ	UP2	UP	
	15	MJ	0	LA	→ Exit
Alarm	16	RJ	WA	WA1	
#38	17	TP	PY	UP3	
	20	RJ	UP2	UP	
	21	MJ	0	LA	→ Exit
Alarm	22	RJ	WA	WA1	
#39	23	TP	PY24	UP3	
	24	RJ	UP2	UP	
	25	MJ	0	LA	→ Exit
Alarm	26	RJ	WA	WA1	
#40	27	TP	SY2	PZ14	
	30	TP	SY3	PZ15	
	31	TP	SY4	PZ16	
	32	TP	PZ	UP3	
	33	RJ	UP2	UP	
	34	MJ	0	LA	→ Exit
Alarm	35	RJ	WA	WA2	
#41	36	TP	SA	UP3	
	37	RJ	UP2	UP	
	40	MJ	0	LA	→ Exit
		CA	ZI41		

		IA	ZJ		
Warn-	0	RJ	WA	WA2	Print hdg.; do not set error bit
ing	1	TP	SA16	UP3	
#42	2	RJ	UP2	UP	
	3	MJ	0	LJ	→ End list string-out
Warn-	4	RJ	WA	WA2	Print hdg.; do not set error bit
ing	5	TP	SY2	SB26	
#43	6	TP	SY3	SB27	Symbol → printout
	7	TP	SY4	SB30	
	10	TP	SB	UP3	
	11	RJ	UP2	UP	
	12	MJ	0	LJ	→ End list string-out
Warn-	13	RJ	WA	WA2	
ing	14	TP	SC	UP3	
#44	15	RJ	UP2	UP	
	16	MJ	0	LJ	→ End list string-out
Warn-	17	RJ	WA	WA2	
ing	20	TP	SY2	SD26	
#45	21	TP	SY3	SD27	Symbol → printout
	22	TP	SY4	SD30	
	23	TP	SD	UP3	
	24	MJ	0	ZJ15	
Warn-	25	RJ	WA	WA2	
ing	26	TP	SE	UP3	
#46	27	RJ	UP2	UP	
	30	TP	LX13	SY2	
	31	MJ	0	LJ	
Warn-	32	RJ	WA	WA2	
ing	33	TP	SE16	UP3	
#47	34	RJ	UP2	UP	
	35	MJ	0	LJ	→ End list string-out
		CA	ZJ36		

		IA	ZK		
Warn-	0	RJ	WA	WA2	
ing	1	TP	SF	UP3	
#48	2	RJ	UP2	UP	
	3	MJ	0	LJ	→ End list string-out
Warn-	4	RJ	WA	WA2	
ing	5	TP	SY2	SF47	
#49	6	TP	SY3	SF50	
	7	TP	SY4	SF51	
	10	TP	SF22	UP3	
	11	RJ	UP2	UP	
	12	MJ	0	LJ	→ End list string-out
Warn-	13	RJ	WA	WA2	Print hdg.; do not set error bit
ing	14	TP	SG	UP3	
#50	15	RJ	UP2	UP	
	16	MJ	0	LJ	
Alarm	17	RJ	WA	WA1	Print hdg.; set error bit
#51	20	TP	SY2	SG31	
	21	TP	SY3	SG32	
	22	TP	SY4	SG33	
	23	TP	SG15	UP3	
	24	RJ	UP2	UP	
	25	MJ	0	LP11	→ Var. count subroutine
Warn-	26	RJ	WA	WA2	Print hdg.; do not set error bit
ing	27	TP	SH	UP3	
#52	30	RJ	UP2	UP	
	31	MJ	0	LJ	→ End list string-out
		CA	ZK32		

		IA	ZL		
Warn-	0	RJ	WA	WA2	
ing#53	1	RP	30004	ZL3	
Preset	2	TP	[30000]	SH32	} Truncated var. name (23 char.)→ printout
in var.					
count	3	RA	SH32	LV50	Put open parent. preceding var. name in printout
rtn.					
	4	TP	SH14	UP3	
	5	RJ	UP2	UP	
	6	TV	LW17	LQ1	Reset ent. → LQ
	7	MJ	0	LQ	
Warn-	10	RJ	WA	WA2	
ing	11	RP	30004	ZL13	
#54	12	TP	LY20	SI16	} Truncated column heading (23 char.)→ printout
	13	RA	SI16	LV50	Put open parent. preceding col. hdg. in printout
	14	TP	SI	UP3	
	15	RJ	UP2	UP	
	16	TV	LW17	LQ1	Reset ent. → LQ
	17	MJ	0	LQ	
Warn-	20	RJ	WA	WA2	
ing	21	RP	30024	ZL23	
#55	22	TP	LY20	SK15	} Truncated title (20 words) → Printout
	23	RA	SK15	LV50	
	24	TP	SK	UP3	Put open parent. preceding title in printout
	25	RJ	UP2	UP	
	26	TP	ZL27	LQ22	Reset to by-pass printout for remaining char. of title
	27	MJ	0	LQ	NB → this instruction used to reset by preceding instruction
		CA	ZL30		
		IA	ZM		
Alarm	0	RJ	WA	WA1	
#56	1	TP	SL	UP3	
	2	RJ	UP2	UP	
	3	MJ	0	LA	→ Exit
Alarm	4	RJ	WA	WA1	
#57	5	CC	WR	LV13	
	6	TP	SL21	UP3	
	7	RJ	UP2	UP	
	10	MJ	0	NF26	
		CA	ZM11		

List String-out Alarm Texts (Alarm Heading 27₁₀)

0	IA	PB									
1	40	PB1	16								Printout #1
2	71	51542	72101	W	O	R	D	,	△		
3	66	24523	02101	T	A	P	E	,	△		
4	24	52523	02454	A	P	P	E	A	R		
5	65	01246	50131	S	△	A	S	△	F		
6	34	54656	60170	I	R	S	T	△	V		
7	24	54342	42546	A	R	I	A	B	L		
10	30	01665	10125	E	△	T	O	△	B		
11	30	01463	46566	E	△	L	I	S	T		
12	30	27220	10154	E	D	.	△	△	R		
13	30	65660	15131	E	S	T	△	O	F		
14	01	65305	06630	△	S	E	N	T	E		
15	50	26300	15051	N	C	E	△	N	O		
16	66	01263	33026	T	△	C	H	E	C		
17	45	30272	27777	K	E	D	.	77	77		
17	40	PB20	13								Printout #2
20	26	51506	56624	C	O	N	S	T	A		
21	50	66210	17777	N	T	,	△	77	77		
22	0	0	0	-	-	-	-	-	-		
23	0	0	0	-	-	-	-	-	-		
24	0	0	0	-	-	-	-	-	-		
25	21	01244	75150	,	△	A	M	O	N		
26	32	01702	45434	G	△	V	A	R	I		
27	24	25463	06501	A	B	L	E	S	△		
30	66	51012	53001	T	O	△	B	E	△		
31	46	34656	63027	L	I	S	T	E	D		
32	22	77777	77777	.	77	77	77	77	77		
	CA	PB33									

	IA	PC			
0	40	PC1	16		Printout #3
1	30	50270	15131	E N D △ O F	
2	01	65305	06630	△ S E N T E	
3	50	26300	16573	N C E △ S Y	
4	47	25514	60124	M B O L △ A	
5	47	51503	20170	M O N G △ V	
6	24	54342	42546	A R I A B L	
7	30	65016	65101	E S △ T O △	
10	25	30014	63465	B E △ L I S	
11	66	30272	20101	T E D . △ △	
12	54	30656	60151	R E S T △ O	
13	31	01653	05066	F △ S E N T	
14	30	50263	00150	E N C E △ N	
15	51	66012	63330	O T △ C H E	
16	26	45302	72201	C K E D . △	
17	40	PC20	23		Printout #4
20	51	52305	00152	O P E N △ P	
21	24	54305	06633	A R E N T H	
22	30	65346	52101	E S I S , △	
23	71	33342	63301	W H I C H △	
24	34	65015	05166	I S △ N O T	
25	01	52543	02630	△ P R E C E	
26	27	30270	12573	D E D △ B Y	
27	01	24274	42426	△ A D J A C	
30	30	50660	10165	E N T △ △ S	
31	67	25652	65434	U B S C R I	
32	52	66302	70170	P T E D △ V	
33	24	54342	42546	A R I A B L	
34	30	21012	45252	E , △ A P P	
35	30	24546	50124	E A R S △ A	
36	47	51503	20170	M O N G △ V	
37	24	54342	42546	A R I A B L	
40	30	65016	65101	E S △ T O △	
41	25	30014	63465	B E △ L I S	
42	66	30272	27777	T E D . 77 77	
	CA	PC43			

	IA	PD			
0	40	PD1	16		Printout #5
1	26	46516	53001	C L O S E △	
2	52	24543	05066	P A R E N T	
3	33	30653	46501	H E S I S △	
4	71	34663	35167	W I T H O U	
5	66	01265	15454	T △ C O R R	
6	30	65525	15027	E S P O N D	
7	34	50320	15152	I N G △ O P	
10	30	50012	45252	E N △ A P P	
11	30	24546	50124	E A R S △ A	
12	47	51503	20170	M O N G △ V	
13	24	54342	42546	A R I A B L	
14	30	65016	65101	E S △ T O △	
15	25	30014	63465	B E △ L I S	
16	66	30272	27777	T E D . 77 77	
17	40	PD20	12		Printout #6
20	65	73472	55146	S Y M B O L	
21	21	01777	77777	, △ 77 77 77 77	
22	0	0	0		
23	0	0	0		
24	0	0	0		
25	21	01244	75150	, △ A M O N	
26	32	01346	63047	G △ I T E M	
27	65	01665	10125	S △ T O △ B	
30	30	01463	46566	E △ L I S T	
31	30	27227	77777	E D . 77 77 77	
	CA	PD32			

	IA	PE			
0	40	PE1	12		Printout #7
1	52	24543	05066	P A R E N T	
2	33	30653	06501	H E S E S △	
3	52	54302	63027	P R E C E D	
4	34	50320	17151	I N G △ W O	
5	54	27210	16624	R D , △ T A	
6	52	30210	15051	P E , △ N O	
7	66	01525	45152	T △ P R O P	
10	30	54467	30101	E R L Y △ △	
11	01	01010	10152	△ △ △ △ △ P	
12	24	34543	02722	A I R E D .	
	CA	PE13			

	IA	PG			
0	40	PG1	15		Printout #8
1	52	65306	72751	P S E U D O	
2	01	51523	05424	△ O P E R A	
3	66	34515	00165	T I O N △ S	
4	73	47255	14621	Y M B O L ,	
5	01	77777	77777	△ 77 77 77 77 77	
6	0	0	0	- - - - -	
7	21	01244	75150	, △ A M O N	
10	32	01702	45434	G △ V A R I	
11	24	25463	06501	A B L E S △	
12	66	51010	10101	T O △ △ △ △	
13	01	01012	53001	△ △ △ B E △	
14	46	34656	63027	L I S T E D	
15	22	77777	77777	. 77 77 77 77 77	
16	40	PG17	13		Printout #9
17	46	34255	42454	L I B R A R	
20	73	01545	16766	Y △ R O U T	
21	34	50300	16573	I N E △ S Y	
22	47	25514	62101	M B O L , △	
23	0	0	0	- - - - -	
24	21	01244	75150	, △ A M O N	
25	32	01702	45434	G △ V A R I	
26	24	25463	06501	A B L E S △	
27	66	51012	53001	T O △ B E △	
30	01	01010	14634	△ △ △ △ L I	
31	65	66302	72277	S T E D . 77	
	CA	PG32			

	IA	PH								
0	40	PH1	12							Printout #10
1	26	51474	72401	C	O	M	M	A	△	
2	47	34656	53450	M	I	S	S	I	N	
3	32	01315	14646	G	△	F	O	L	L	
4	51	71345	03201	O	W	I	N	G	△	
5	70	24543	42425	V	A	R	I	A	B	
6	46	30210	17777	L	E	,	△	77	77	
7	0	0	0	-	-	-	-	-	-	
10	0	0	0	-	-	-	-	-	-	
11	0	0	0	-	-	-	-	-	-	
12	22	77777	77777	.	77	77	77	77	77	
13	40	PH14	22							Printout #11
14	65	67256	52654	S	U	B	S	C	R	
15	34	52663	02701	I	P	T	E	D	△	
16	70	24543	42425	V	A	R	I	A	B	
17	46	30210	17777	L	E	,	△	77	77	
20	0	0	0	-	-	-	-	-	-	
21	21	01505	16601	,	△	N	O	T	△	
22	31	51464	65171	F	O	L	L	O	W	
23	30	27012	57301	E	D	△	B	Y	△	
24	51	52305	00101	O	P	E	N	△	△	
25	01	01010	10101	△	△	△	△	△	△	
26	52	24543	05066	P	A	R	E	N	T	
27	33	30653	46522	H	E	S	I	S	.	
30	01	65672	56526	△	S	U	B	S	C	
31	54	34526	66501	R	I	P	T	S	△	
32	24	65656	74730	A	S	S	U	M	E	
33	27	01665	10125	D	△	T	O	△	B	
34	30	01473	46565	E	△	M	I	S	S	
35	34	50322	27777	I	N	G	.	77	77	
	CA	PH36								

	IA	PI			
0	40	PI1	20		Printout #12
1	30	50270	15131	E N D Δ O F	
2	01	65305	06630	Δ S E N T E	
3	50	26300	16573	N C E Δ S Y	
4	47	25514	60124	M B O L Δ A	
5	47	51503	20165	M O N G Δ S	
6	67	25652	65434	U B S C R I	
7	52	66650	13151	P T S Δ F O	
10	54	01777	77777	R Δ 77 77 77 77	
11	0	0	0	- - - - -	
12	22	01010	10101	. Δ Δ Δ Δ Δ	
13	01	01015	43065	Δ Δ Δ R E S	
14	66	01513	10165	T Δ O F Δ S	
15	30	50663	05026	E N T E N C	
16	30	01505	16601	E Δ N O T Δ	
17	26	33302	64530	C H E C K E	
20	27	22777	77777	D . 77 77 77 77	
21	40	PI22	11		Printout #13
22	51	52305	00152	O P E N Δ P	
23	24	54305	06633	A R E N T H	
24	30	65346	50124	E S I S Δ A	
25	47	51503	20165	M O N G Δ S	
26	67	25652	65434	U B S C R I	
27	52	66650	13151	P T S Δ F O	
30	54	01777	77777	R Δ 77 77 77 77	
31	0	0	0	- - - - -	
32	22	77777	77777	. 77 77 77 77 77	
	CA	PI33			

	IA	PJ								
0	40	PJ1	13							Printout #14
1	65	73472	55146	S	Y	M	B	O	L	
2	21	01777	77777	,	Δ	77	77	77	77	
3	0	0	0	-	-	-	-	-	-	
4	21	01245	25230	,	Δ	A	P	P	E	
5	24	54650	12447	A	R	S	Δ	A	M	
6	51	50320	16567	O	N	G	Δ	S	U	
7	25	65265	43452	B	S	C	R	I	P	
10	66	65013	15154	T	S	Δ	F	O	R	
11	01	77777	77777	Δ	77	77	77	77	77	
12	0	0	0	-	-	-	-	-	-	
13	22	77777	77777	.	77	77	77	77	77	
14	40	PJ15	11							Printout #15
15	71	51542	72101	W	O	R	D	,	Δ	
16	66	24523	02101	T	A	P	E	,	Δ	
17	24	52523	02454	A	P	P	E	A	R	
20	65	01244	75150	S	Δ	A	M	O	N	
21	32	01656	72565	G	Δ	S	U	B	S	
22	26	54345	26665	C	R	I	P	T	S	
23	01	31515	40177	Δ	F	O	R	Δ	77	
24	0	0	0	-	-	-	-	-	-	
25	22	77777	77777	.	77	77	77	77	77	
	CA	PJ26								

	IA	PK							
0	40	PK1	20						Printout #16
1	65	67256	52654	S	U	B	S	C	R
2	34	52666	50131	I	P	T	S	△	F
3	51	54017	77777	O	R	△	77	77	77
4	0	0	0	-	-	-	-	-	-
5	01	34502	64667	△	I	N	C	L	U
6	27	30016	63330	D	E	△	T	H	E
7	01	31465	12466	△	F	L	O	A	T
10	34	50320	15251	I	N	G	△	P	O
11	34	50660	10101	I	N	T	△	△	△
12	01	01010	10101	△	△	△	△	△	△
13	01	70245	43424	△	V	A	R	I	A
14	25	46302	10177	B	L	E	,	△	77
15	0	0	0	-	-	-	-	-	-
16	0	0	0	-	-	-	-	-	-
17	0	0	0	-	-	-	-	-	-
20	22	77777	77777	.	77	77	77	77	77
21	40	PK23	11						Printout #17
22	40	PK23	17						#17A
23	31	67502	66634	F	U	N	C	T	I
24	51	50210	17777	O	N	,	△	77	77
25	0	0	0	-	-	-	-	-	-
26	21	01244	75150	,	△	A	M	O	N
27	32	01656	72565	G	△	S	U	B	S
30	26	54345	26665	C	R	I	P	T	S
31	01	31515	40177	△	F	O	R	△	77
32	0	0	0	-	-	-	-	-	-
33	22	01010	10101	.	△	△	△	△	△
34	01	01010	10101	△	△	△	△	△	△
35	01	01010	10101	△	△	△	△	△	△
36	24	54326	74730	A	R	G	U	M	E
37	50	66650	16567	N	T	S	△	S	U
40	52	30543	14667	P	E	R	F	L	U
41	51	67652	27777	O	U	S	.	77	77
	CA	PK42							

Printout #17
#17A

Printout #18
#18A

	IA	PM			
0	40	PM1	14		Printout #19
1	46	34255	42454	L I B R A R	
2	73	01545	16766	Y Δ R O U T	
3	34	50300	16573	I N E Δ S Y	
4	47	25514	62101	M B O L , Δ	
5	0	0	0	- - - - -	
6	21	01244	75150	, Δ A M O N	
7	32	01656	72565	G Δ S U B S	
10	26	54345	26665	C R I P T S	
11	01	31515	40101	Δ F O R Δ Δ	
12	01	01010	17777	Δ Δ Δ Δ 77 77	
13	0	0	0	- - - - -	
14	22	77777	77777	. 77 77 77 77 77	
15	40	PM16	15		Printout #20
16	52	65306	72751	P S E U D O	
17	01	51523	05424	Δ O P E R A	
20	66	34515	00165	T I O N Δ S	
21	73	47255	14621	Y M B O L ,	
22	01	77777	77777	Δ 77 77 77 77 77	
23	0	0	0	- - - - -	
24	21	01244	75150	, Δ A M O N	
25	32	01656	72565	G Δ S U B S	
26	26	54345	26665	C R I P T S	
27	01	31515	40101	Δ F O R Δ Δ	
30	01	01017	77777	Δ Δ Δ 77 77 77	
31	0	0	0	- - - - -	
32	22	77777	77777	. 77 77 77 77 77	
	CA	PM33			

	IA	PN			
0	40	PN1	17		Printout #21
1	65	67256	52654	S U B S C R	
2	34	52666	50131	I P T S Δ F	
3	51	54017	77777	O R Δ 77 77 77	
4	0	0	0	- - - - -	
5	01	34502	64667	Δ I N C L U	
6	27	30016	56752	D E Δ S U P	
7	30	54652	65434	E R S C R I	
10	52	66012	65150	P T Δ C O N	
11	65	66245	06621	S T A N T ,	
12	01	01010	10101	Δ Δ Δ Δ Δ Δ	
13	01	77777	77777	Δ 77 77 77 77 77	
14	0	0	0	- - - - -	
15	0	0	0	- - - - -	
16	0	0	0	- - - - -	
17	22	77777	77777	. 77 77 77 77 77	
	CA	PN20			

	IA	PP			
0	40	PP1	22		Printout #24
1	30	50270	15131	E N D Δ O F	
2	01	65305	06630	Δ S E N T E	
3	50	26300	16573	N C E Δ S Y	
4	47	25514	60131	M B O L Δ F	
5	51	46465	17165	O L L O W S	
6	01	26465	16530	Δ C L O S E	
7	01	65672	56526	Δ S U B S C	
10	54	34526	60101	R I P T Δ Δ	
11	01	01010	10152	Δ Δ Δ Δ Δ P	
12	24	54305	06633	A R E N T H	
13	30	65346	50131	E S I S Δ F	
14	51	54017	77777	O R Δ 77 77 77	
15	0	0	0	- - - - -	
16	22	01543	06566	. Δ R E S T	
17	01	51310	16530	Δ O F Δ S E	
20	66	30502	63001	T E N C E Δ	
21	50	51660	12633	N O T Δ C H	
22	30	26453	02722	E C K E D Δ	
23	40	PP24	14		Printout #25
24	26	51474	72401	C O M M A Δ	
25	47	34656	53450	M I S S I N	
26	32	01315	14646	G Δ F O L L	
27	51	71345	03201	O W I N G Δ	
30	26	46516	53001	C L O S E Δ	
31	65	67256	52654	S U B S C R	
32	34	52660	15224	I P T Δ P A	
33	54	30506	63330	R E N T H E	
34	65	34650	10131	S I S Δ Δ F	
35	51	54017	77777	O R Δ 77 77 77	
36	0	0	0	- - - - -	
37	22	77777	77777	. 77 77 77 77 77	
	CA	PP40			

[illegible]

Printout #27

	IA	PU			
0	40	PU1	20		Printout #32
1	34	50702	44634	I N V A L I	
2	27	01662	45230	D Δ T A P E	
3	01	27306	53432	Δ D E S I G	
4	50	24663	45150	N A T I O N	
5	22	01264	65165	. Δ C L O S	
6	30	01522	45430	E Δ P A R E	
7	50	66333	06534	N T H E S I	
10	65	01315	14646	S Δ F O L L	
11	51	71650	10171	O W S Δ Δ W	
12	51	54272	10166	O R D , Δ T	
13	24	52302	20154	A P E . Δ R	
14	30	65660	15131	E S T Δ O F	
15	01	65305	06630	Δ S E N T E	
16	50	26300	15051	N C E Δ N O	
17	66	01263	33026	T Δ C H E C	
20	45	30272	27777	K E D . 77 77	
21	40	PU22	21		Printout #33
22	34	50702	44634	I N V A L I	
23	27	01662	45230	D Δ T A P E	
24	01	27306	53432	Δ D E S I G	
25	50	24663	45150	N A T I O N	
26	22	01657	34725	. Δ S Y M B	
27	51	46210	17777	O L , Δ 77 77	
30	0	0	0	- - - - -	
31	21	01315	14646	, Δ F O L L	
32	51	71650	10101	O W S Δ Δ Δ	
33	01	01010	10101	Δ Δ Δ Δ Δ Δ	
34	71	51542	72101	W O R D , Δ	
35	66	24523	02201	T A P E . Δ	
36	54	30656	60151	R E S T Δ O	
37	31	01653	05066	F Δ S E N T	
40	30	50263	00150	E N C E Δ N	
41	51	66012	63330	O T Δ C H E	
42	26	45302	72277	C K E D . 77	
	CA	PU43			

	IA	PV			
0	40	PV1	23		Printout #34
1	34	50702	44634	I N V A L I	
2	27	01662	45230	D Δ T A P E	
3	01	27306	53432	Δ D E S I G	
4	50	24663	45150	N A T I O N	
5	22	01463	42554	. Δ L I B R	
6	24	54730	15451	A R Y Δ R O	
7	67	66345	03001	U T I N E Δ	
10	65	73472	55146	S Y M B O L	
11	21	01010	10177	, Δ Δ Δ Δ 77	
12	0	0	0	- - - - -	
13	21	01315	14646	, Δ F O L L	
14	51	71650	17151	O W S Δ W O	
15	54	27210	16624	R D , Δ T A	
16	52	30220	15430	P E . Δ R E	
17	65	66015	13101	S T Δ O F Δ	
20	65	30506	63050	S E N T E N	
21	26	30015	05166	C E Δ N O T	
22	01	26333	02645	Δ C H E C K	
23	30	27227	77777	E D . 77 77 77	
24	40	PV25	23		Printout #35
25	34	50702	44634	I N V A L I	
26	27	01662	45230	D Δ T A P E	
27	01	27306	53432	Δ D E S I G	
30	50	24663	45150	N A T I O N	
31	22	01526	53067	. Δ P S E U	
32	27	51015	15230	D O Δ O P E	
33	54	24663	45150	R A T I O N	
34	01	65734	72551	Δ S Y M B O	
35	46	21010	10177	L , Δ Δ Δ 77	
36	0	0	0	- - - - -	
37	21	01315	14646	, Δ F O L L	
40	51	71650	17151	O W S Δ W O	
41	54	27210	16624	R D , Δ T A	
42	52	30220	15430	P E . Δ R E	
43	65	66015	13101	S T Δ O F Δ	
44	65	30506	63050	S E N T E N	
45	26	30015	05166	C E Δ N O T	
46	01	26333	02645	Δ C H E C K	
47	30	27227	77777	E D . 77 77 77	
	CA	PV50			

	IA	PW			
0	40	PW1	24		Printout #36
1	34	50702	44634	I N V A L I	
2	27	01662	45230	D Δ T A P E	
3	01	65734	72551	Δ S Y M B O	
4	46	23015	43065	L ; Δ R E S	
5	66	01513	10165	T Δ O F Δ S	
6	30	50663	05026	E N T E N C	
7	30	01675	02633	E Δ U N C H	
10	30	26453	02722	E C K E D .	
11	01	50515	00231	Δ N O N - F	
12	34	72302	70152	I X E D Δ P	
13	51	34506	60170	O I N T Δ V	
14	24	54342	42546	A R I A B L	
15	30	21017	77777	E , Δ 77 77 77	
16	0	0	0	- - - - -	
17	0	0	0	- - - - -	
20	0	0	0	- - - - -	
21	21	01315	14646	, Δ F O L L	
22	51	71650	17151	O W S Δ W O	
23	54	27210	16624	R D , Δ T A	
24	52	30227	77777	P E . 77 77 77	
	CA	PW25			

	IA	PX			
0	40	PX1	23		Printout #37
1	34	50702	44634	I N V A L I	
2	27	01662	45230	D Δ T A P E	
3	01	50674	72530	Δ N U M B E	
4	54	23015	43065	R ; Δ R E S	
5	66	01513	10165	T Δ O F Δ S	
6	30	50663	05026	E N T E N C	
7	30	01505	16601	E Δ N O T Δ	
10	26	33302	64530	C H E C K E	
11	27	22010	10124	D . Δ Δ Δ A	
12	65	65674	73027	S S U M E D	
13	01	26515	06566	Δ C O N S T	
14	24	50662	10177	A N T , Δ 77	
15	0	0	0	- - - - -	
16	0	0	0	- - - - -	
17	0	0	0	- - - - -	
20	21	01315	14646	, Δ F O L L	
21	51	71650	17151	O W S Δ W O	
22	54	27210	16624	R D , Δ T A	
23	52	30227	77777	P E . 77 77 77	
	CA	PX24			

	IA	PZ			
0	40	PZ1	23		Printout #40
1	26	51474	72401	C O M M A Δ	
2	51	54013	05027	O R Δ E N D	
3	01	51310	16530	Δ O F Δ S E	
4	50	66305	02630	N T E N C E	
5	01	65734	72551	Δ S Y M B O	
6	46	01473	46565	L Δ M I S S	
7	34	50322	20166	I N G . Δ T	
10	24	52300	16573	A P E Δ S Y	
11	47	25514	60131	M B O L Δ F	
12	51	46465	17130	O L L O W E	
13	27	01257	30177	D Δ B Y Δ 77	
14	0	0	0	- - - - -	
15	0	0	0	- - - - -	
16	0	0	0	- - - - -	
17	22	01543	06566	. Δ R E S T	
20	01	51310	16530	Δ O F Δ S E	
21	50	66305	02630	N T E N C E	
22	01	67502	63330	Δ U N C H E	
23	26	45302	72277	C K E D . 77	
	CA	PZ24			

	IA	SB	35	
0	40	SB1		
1	71	24545	03450	W A R N I N
2	32	21010	16624	G , Δ Δ T A
3	52	30012	73065	P E Δ D E S
4	34	32502	46634	I G N A T I
5	51	50016	53026	O N Δ S E C
6	66	34515	00165	T I O N Δ S
7	30	51674	62701	H O U L D Δ
10	25	30013	15146	B E Δ F O L
11	46	51713	02701	L O W E D Δ
12	25	73016	63330	B Y Δ T H E
13	01	30502	70151	Δ E N D Δ O
14	31	01653	05066	F Δ S E N T
15	30	50263	00165	E N C E Δ S
16	73	47255	14601	Y M B O L Δ
17	51	54012	45001	O R Δ A N Δ
20	51	52305	00133	O P E N Δ H
21	30	24273	45032	E A D I N G
22	01	52245	43050	Δ P A R E N
23	66	33306	53465	T H E S I S
24	21	34506	56630	, I N S T E
25	24	27015	13101	A D Δ O F Δ
26	0	0	0	- - - - -
27	0	0	0	- - - - -
30	0	0	0	- - - - -
31	22	01543	06566	. Δ R E S T
32	01	51310	16530	Δ O F Δ S E
33	50	66305	02630	N T E N C E
34	01	34325	05154	Δ I G N O R
35	30	27227	77777	E D . 77 77 77
	CA	SB36		

Printout #43

	IA	SC			
0	40	SC1	23		Printout #44
1	71	24545	03450	W	A
2	32	21010	12646	G	,
3	51	65300	15224	O	S
4	54	30506	63330	R	E
5	65	34650	17134	S	I
6	66	33516	76601	T	H
7	26	51545	43065	C	O
10	52	51502	73450	P	O
11	32	01010	10151	G	△
12	52	30500	12452	P	E
13	52	30245	46501	P	E
14	24	47515	03201	A	M
15	26	51466	74750	C	O
16	01	33302	42734	△	H
17	50	32652	20154	N	G
20	30	65660	15131	E	S
21	01	65305	06630	△	S
22	50	26300	13432	N	C
23	50	51543	02722	N	O
	CA	SC24			

	IA	SD			
0	40	SD1	35		
1	71	24545	03450	W	A R N I N
2	32	21010	15254	G	, Δ Δ P R
3	30	26302	73450	E	C E D I N
4	32	01265	14667	G	Δ C O L U
5	47	50013	33024	M	N Δ H E A
6	27	34503	20165	D	I N G Δ S
7	33	51674	62701	H	O U L D Δ
10	25	30013	15146	B	E Δ F O L
11	46	51713	02701	L	O W E D Δ
12	25	73015	15230	B	Y Δ O P E
13	50	01265	14667	N	Δ C O L U
14	47	50013	33024	M	N Δ H E A
15	27	34503	20152	D	I N G Δ P
16	24	54305	06633	A	R E N T H
17	30	65346	50151	E	S I S Δ O
20	54	01305	02701	R	Δ E N D Δ
21	51	31016	53050	O	F Δ S E N
22	66	30502	63001	T	E N C E Δ
23	65	73472	55146	S	Y M B O L
24	21	34506	56630	,	I N S T E
25	24	27015	13101	A	D Δ O F Δ
26	0	0	0	-	- - - - -
27	0	0	0	-	- - - - -
30	0	0	0	-	- - - - -
31	22	01543	06566	.	Δ R E S T
32	01	51310	16530	Δ	O F Δ S E
33	50	66305	02630	N	T E N C E
34	01	34325	05154	Δ	I G N O R
35	30	27227	77777	E	D . 77 77 77
	CA	SD36			

Printout #45

[illegible]

	IA	SF	21		
0	40	SF1	21		Printout #48
1	71	24545	03450	W A R N I N	
2	32	21010	12646	G , Δ Δ C L	
3	51	65300	15224	O S E Δ P A	
4	54	30506	63330	R E N T H E	
5	65	34650	17134	S I S Δ W I	
6	66	33516	76601	T H O U T Δ	
7	26	51545	43065	C O R R E S	
10	52	51502	73450	P O N D I N	
11	32	01010	10151	G Δ Δ Δ Δ O	
12	52	30500	13151	P E N Δ F O	
13	46	46517	16501	L L O W S Δ	
14	66	34664	63022	T I T L E .	
15	01	54306	56601	Δ R E S T Δ	
16	51	31016	53050	O F Δ S E N	
17	66	30502	63001	T E N C E Δ	
20	34	32505	15430	I G N O R E	
21	27	22777	77777	D . 77 77 77 77	
22	40	SF23	34		Printout #49
23	71	24545	03450	W A R N I N	
24	32	21010	16634	G , Δ Δ T I	
25	66	46300	16530	T L E Δ S E	
26	26	66345	15001	C T I O N Δ	
27	65	33516	74627	S H O U L D	
30	01	25300	13151	Δ B E Δ F O	
31	46	46517	13027	L L O W E D	
32	01	25730	13050	Δ B Y Δ E N	
33	27	01513	10165	D Δ O F Δ S	
34	30	50663	05026	E N T E N C	
35	30	01657	34725	E Δ S Y M B	
36	51	46015	15401	O L Δ O R Δ	
37	51	52305	00126	O P E N Δ C	
40	51	46674	75001	O L U M N Δ	
41	33	30242	73450	H E A D I N	
42	32	01522	45430	G Δ P A R E	
43	50	66333	06534	N T H E S I	
44	65	21013	45065	S , Δ I N S	
45	66	30242	70101	T E A D Δ Δ	
46	51	31017	77777	O F Δ 77 77 77	
47	0	0	0	- - - - -	
50	0	0	0	- - - - -	
51	0	0	0	- - - - -	
52	22	01543	06566	. Δ R E S T	
53	01	51310	16530	Δ O F Δ S E	
54	50	66305	02630	N T E N C E	
55	01	34325	05154	Δ I G N O R	
56	30	27227	77777	E D . 77 77 77	
	CA	SF57			

	IA	SG								
0	40	SG1	14							Printout #50
1	71	24545	03450	W	A	R	N	I	N	
2	32	21010	13050	G	.	Δ	Δ	E	N	
3	27	01513	10165	D	Δ	O	F	Δ	S	
4	30	50663	05026	E	N	T	E	N	C	
5	30	01657	34725	E	Δ	S	Y	M	B	
6	51	46013	45001	O	L	Δ	I	N	Δ	
7	66	34664	63022	T	I	T	L	E	.	
10	01	54306	56601	Δ	R	E	S	T	Δ	
11	51	31010	10165	O	F	Δ	Δ	S	Δ	
12	30	50663	05026	E	N	T	E	N	C	
13	30	01343	25051	E	Δ	I	G	N	O	
14	54	30272	27777	R	E	D	.	77	77	
15	40	SG16	21							Printout #51
16	47	51543	00166	M	O	R	E	Δ	T	
17	33	24500	13134	H	A	N	Δ	F	I	
20	70	30017	02454	V	E	Δ	V	A	R	
21	34	24254	63065	I	A	B	L	E	S	
22	01	66510	12530	Δ	T	O	Δ	B	E	
23	01	46346	56630	Δ	L	I	S	T	E	
24	27	22017	02454	D	.	Δ	V	A	R	
25	34	24254	63065	I	A	B	L	E	S	
26	01	01010	10131	Δ	Δ	Δ	Δ	Δ	F	
27	51	46465	17134	O	L	L	O	W	I	
30	50	32017	77777	N	G	Δ	77	77	77	
31	0	0	0	-	-	-	-	-	-	
32	0	0	0	-	-	-	-	-	-	
33	0	0	0	-	-	-	-	-	-	
34	01	50516	60126	Δ	N	O	T	Δ	C	
35	33	30264	53027	H	E	C	K	E	D	
36	22	77777	77777	.	77	77	77	77	77	
	CA	SG37								

	IA	SH			
0	40	SH1	13		Printout #52
1	71	24545	03450	W A R N I N	
2	32	21010	14751	G , Δ Δ M O	
3	54	30012	65146	R E Δ C O L	
4	67	47500	13330	U M N Δ H E	
5	24	27345	03265	A D I N G S	
6	01	66332	45001	Δ T H A N Δ	
7	70	24543	42425	V A R I A B	
10	46	30652	20130	L E S . Δ E	
11	72	26306	56501	X C E S S Δ	
12	27	30463	06630	D E L E T E	
13	27	22777	77777	D . 77 77 77 77	
14	40	SH15	22		Printout #53
15	71	24545	03450	W A R N I N	
16	32	21010	14751	G , Δ Δ M O	
17	54	30016	63324	R E Δ T H A	
20	50	01050	60126	N Δ 2 3 Δ C	
21	33	24542	42666	H A R A C T	
22	30	54650	13450	E R S Δ I N	
23	01	70245	43424	Δ V A R I A	
24	25	46300	15024	B L E Δ N A	
25	47	30220	10130	M E . Δ Δ E	
26	72	26306	56501	X C E S S Δ	
27	27	30463	06630	D E L E T E	
30	27	01463	02470	D Δ L E A V	
31	34	50320	17777	I N G Δ 77 77	
32	0	0	0	- - - - -	
33	0	0	0	- - - - -	
34	0	0	0	- - - - -	
35	0	0	0	- - - - -	
36	43	22777	77777	) . 77 77 77 77	
	CA	SH37			

0	IA	SI	22	
1	40	SI1	03450	W A R N I N
2	71	24545	14751	G , Δ Δ M O
3	32	21010	63324	R E Δ T H A
4	54	30016	60126	N Δ 2 3 Δ C
5	50	01050	42666	H A R A C T
6	33	24542	13450	E R S Δ I N
7	30	54650	66747	Δ C O L U M
10	01	26514	02427	N Δ H E A D
11	50	01333	20130	I N G . Δ E
12	34	50322	56501	X C E S S Δ
13	72	26306	06630	D E L E T E
14	27	30463	02470	D Δ L E A V
15	27	01463	17777	I N G Δ 77 77
16	34	50320	0	- - - - -
17	0	0	0	- - - - -
20	0	0	0	- - - - -
21	0	0	0	- - - - -
22	0	0	0	- - - - -
22	43	22777	77777	) . 77 77 77 77
	CA	SI23		

Warning #54

	IA	SK		
0	40	SK1	41	
1	71	24545	03450	W A R N I N
2	32	21010	14751	G , Δ Δ M O
3	54	30016	63324	R E Δ T H A
4	50	01040	41401	N Δ 1 1 9 Δ
5	26	33245	42426	C H A R A C
6	66	30546	50134	T E R S Δ I
7	50	01663	46646	N Δ T I T L
10	30	22013	07226	E . Δ E X C
11	30	65650	10127	E S S Δ Δ D
12	30	46306	63027	E L E T E D
13	01	46302	47034	Δ L E A V I
14	50	32017	77777	N G Δ 77 77 77
15	0	0	0	- - - - -
16	0	0	0	- - - - -
17	0	0	0	- - - - -
20	0	0	0	- - - - -
21	0	0	0	- - - - -
22	0	0	0	- - - - -
23	0	0	0	- - - - -
24	0	0	0	- - - - -
25	0	0	0	- - - - -
26	0	0	0	- - - - -
27	0	0	0	- - - - -
30	0	0	0	- - - - -
31	0	0	0	- - - - -
32	0	0	0	- - - - -
33	0	0	0	- - - - -
34	0	0	0	- - - - -
35	0	0	0	- - - - -
36	0	0	0	- - - - -
37	0	0	0	- - - - -
40	0	0	0	- - - - -
41	43	22777	77777	) . 77 77 77 77
	CA	SK42		

Warning #55

Explanation of Indicators, Counters, Temporaries, etc.

LY0	Index counter (C_1) - (# subscripts)					
1	Avail. add. in assem. blk. ("u" & "v")					
2	Avail. add. in hdg. list.					
3	Temp. or add. for 1st col. hdg. in hdg. list - 1					
4	Indicator 1st var.					
	<table><tr><td>00</td><td>← fl. pt. ind.</td><td rowspan="2">char. count</td></tr><tr><td>40</td><td>← fix pt. ind.</td></tr></table>	00	← fl. pt. ind.	char. count	40	← fix pt. ind.
00	← fl. pt. ind.	char. count				
40	← fix pt. ind.					
5	Indicator 2nd var.					
6	Indicator 3rd var.					
7	Indicator 4th var.					
10	Indicator 5th var.					
11	Index counter (C_2) - (#char. in assem. blk. word)					
12	Index counter (C_3)					
13	Heading indicator (1st bit = 1 $\Rightarrow$ col. hdg. present) (2nd bit = 1 $\Rightarrow$ title present)					
14	Col. hdg. count					
15	Char. count (v) or string-out count ("u" & "v")					
16	} Current variable XS3 symbol					
17						
20						
21						
22						
23						
24						
25						
26						
27						
30						
31						
32	Assembly Block (25_8)					
33						
34						
35						
36						
37						
40						
41						
42						
43						
44	Parenthesis level					
45	Count of subscripts processed					
46	Subscript parenthesis level					
47	Function parent. level (subs. var. within subs.)					
50	Function symbol or subs. var. within subscript sym.					

PRINT STRING-OUT

Print string-out uses three lines of GN (Get Next Character Routine) with the following understanding concerning their function. GN2 holds the buffer input VK address in u and v of the line from which the last character has been obtained. GN3 holds the shift that has been needed to extract the last character. It will vary from 6 to 44_8 . GN6 holds the number of the blockette from which the last character has been obtained. This number varies from 0 to 5 for the 6 blockettes of the block of input data.

When the Print Routine has been entered from CT, Control Routine, the divider following PRINT is the last character. It should be a space, but whatever it is, it is not included in the group of XS3 codes stored in the output for later conversion and printing.

The current line containing the last character is replaced by a line in which 77's replace all characters already obtained from it. GN3 is used to make up the masking QS instruction to secure this replacement.

The current blockette is examined from the last line backward to find the last line in the blockette which is not a line of spaces. From the address of this line is subtracted the address of the current line to get the number of lines of print data to be transferred to the buffer region output, VN. A search for Δ is made of the last two lines in the blockette that are not lines of spaces. If a Δ is found, the Print String-out is terminated. If it is not found, the first line of the next blockette is examined. If this is a line of spaces, all lines from the second on to the last line of the blockette which is not a line of spaces are transferred to the output. If this next blockette first line is not a line of spaces, it is assumed to be a line number, and this causes a termination of the Print String-out.

Continuing in this manner, each blockette is successively examined until a Δ is found as indicated, or until a line number is found in the following blockette. However, if or when print data in 6 blockettes has been transferred to output, the routine is automatically terminated.

When the routine is terminated, GN2 is set to the address of the first line of the blockette following the last one from which data has been transferred.

GN6 is given the proper number of this blockette. The exit to the Control Routine eventually puts future analysis in the Get Next Sentence Routine. This routine checks GN2 against a set of addresses of blockette first lines and if an equality is found, gets the next sentence from the blockette in which the address in GN2 is found. Otherwise, if no equality is found, the Get Next Sentence Routine gets the next sentence from the following blockette. Following a print instruction, of course, such an equality is always found.

When blockette 5, the last of 6 in the block, is one of the group examined, a new block of data is read in by the routine and GN2 and GN6 reset as needed.

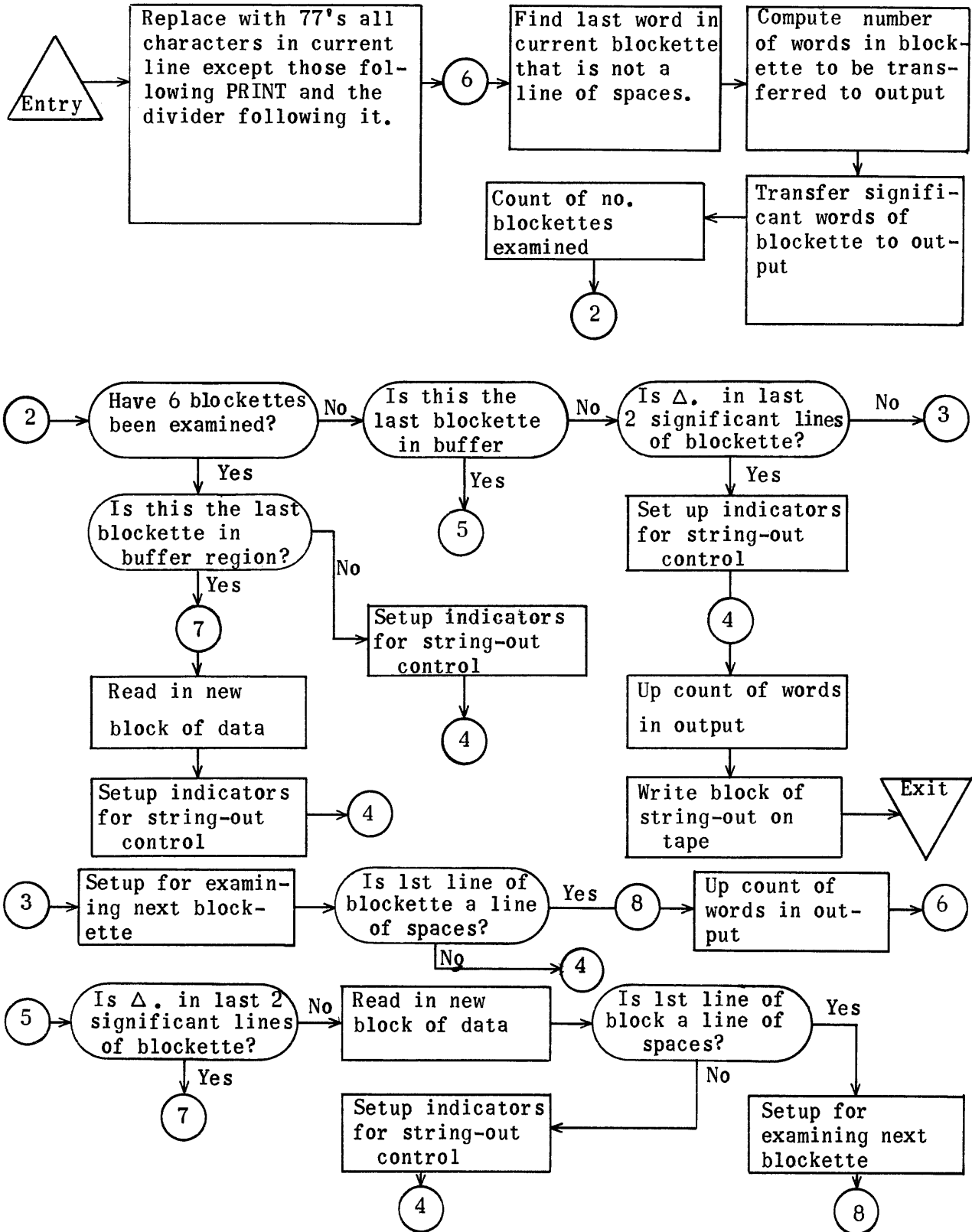
When a print instruction is set on the Unityper, a line number goes in the first 6 places on the first Unityper line. The tab key should be set and used for all runover lines so that a hanging indentation of 6 spaces starts each runover line. Failure to do so will cause a premature termination of the Print String-out and an error print-out originating from a string-out subroutine.

Preferably words should be set right out to the end of any one Unityper line without any excess spacing. No hyphens should be used to break words at the end. If all of a word does not go in one line, as many letters as possible should be put on the top line and the remaining letters should start on the 7th position following the 6 spaces in the runover line.

However, the routine does not require such close typing right out to the end preceding a runover. Much of the excess spacing that might be put in by ragged indentions on the right is eliminated by the routine. As much as 5 spaces may be left to occur in the print-out by the present system of assembling data from blockettes. If excess spacing is left starting from the 7th position on, or if excess spacing is left between words in a line, such excess will be fully duplicated in the final print-out.

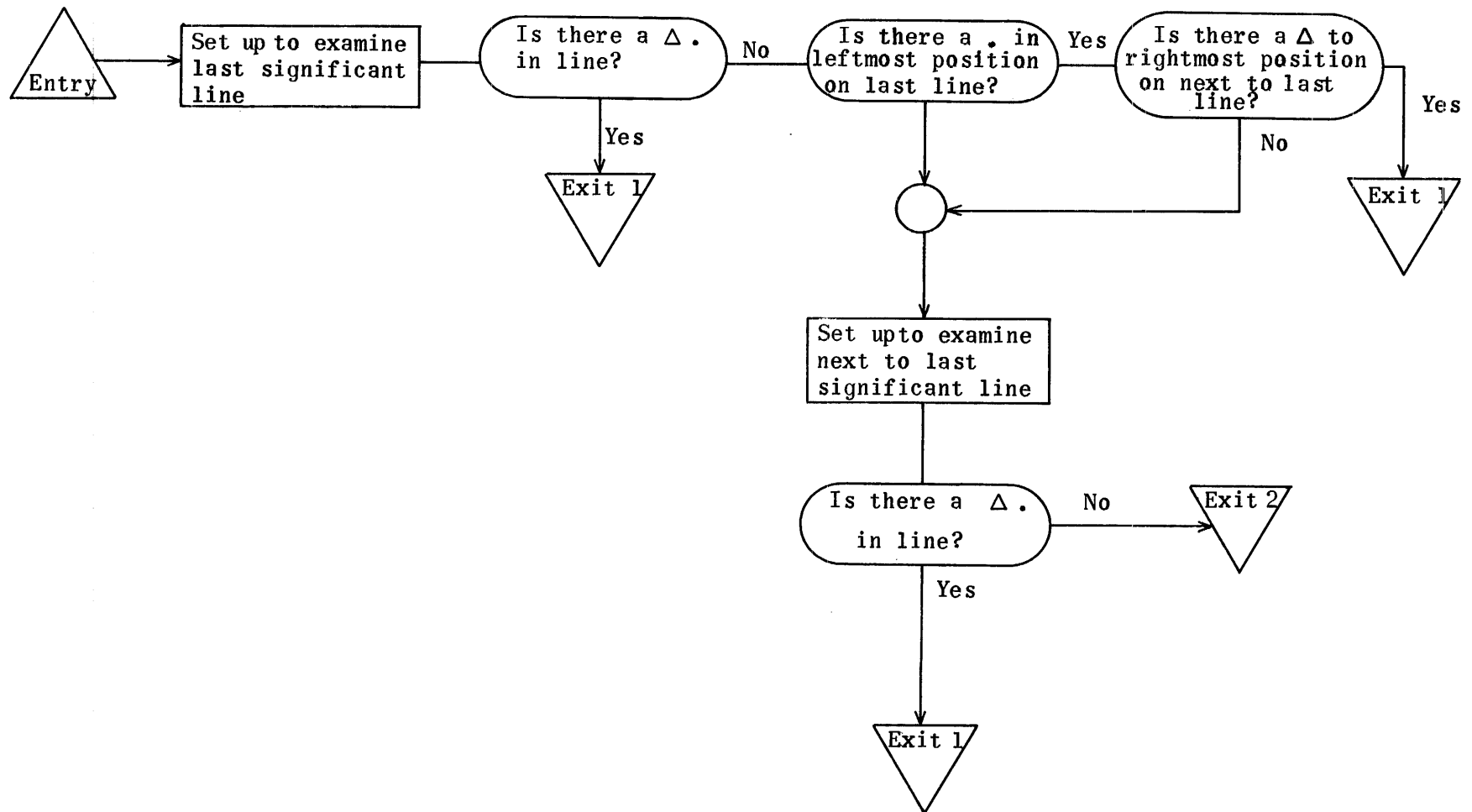
Δ .'s, as explained, when occurring in the last two non-space-filler words in a Unityper line or blockette, have the same effect of ending a sentence as in the rest of UNICODE. However, their use elsewhere in the print-out data does not have this effect because this data is not examined symbol by symbol. The blanket rule of eliminating Δ .'s, except at the end of a sentence, prevents possible confusion concerning their use. Likewise, they may be eliminated from the end of a Print instruction since a following line number in the next blockette serves the same purpose of termination. However, their inclusion at the end speeds up the program.

Flow Chart for PRINT String-Out Routine Page 1 of 2



Flow Chart for PRINT String-Out Routine-Cont. Page 2 of 2

Flow Chart for Subroutine that looks for Δ . in Two Lines



PRINT STRING-OUT REGIONS

RE VN3507	}	String-out subroutine regions used.
RE WT3207		
RE VK3317		
RE GT21		
RE GN1324		
RE CT714		

RE PS4400
RE TZ4445
RE HT4477
RE NR4527
RE VR4551
RE BB4562
RE ZP4605
RE WP4631

Print String-Out

	IA	PS			
	0 MJ	0	CT		Exit
	1 TV	GN2	PS7		Current address to v of PS7
	2 TP	GN3	A	}	<u>Shift</u> → A
	3 DV	ZP	A		
	4 SA	BB16	17	}	Getting proper mask into Q
	5 TU	A	PS6		
	6 TP	30000	Q		
	7 QS	ZP23	30000		Replacing current line with 77s and any characters other than divider to be printed
	10 TP	ZP6	WP5		Clearing counter of blockettes used
	11 TP	ZP14	WP6		Number of lines accumulated (4) to line accumulative counter
	12 TP	GN6	A	}	Blockette count to A
	13 SA	BB15	17		Getting address of address of last line of current blockette to u of PS15
	14 TU	A	PS15		
	15 SP	30000	17		
	16 TU	A	PS17		Getting address of last line of current blockette to u of PS17
Loop to find last line in a blockette that is not a line of spaces.	17 TP	30000	A		Contents of last line of current blockette to A
	20 EJ	ZP2	PS22		Is it a line of spaces?
	21 MJ	0	PS24		
	22 RS	PS17	ZP3		Reducing u of PS17 by one
Computation of number of lines to be transferred to VN from VK blockette	23 MJ	0	PS17		
	24 TU	PS17	WP1	}	Address of last significant line in a blockette to WP1.
	25 LQ	WP1	25		
	26 QT	ZP4	WP1	}	
	27 TV	GN2	WP2		
	30 TP	ZP4	Q	}	Current address to v of WP2 and v of A
	31 QT	WP2	WP2		
	32 TN	A	A	}	Number of lines in blockette to be transferred to WP2
	33 AT	WP1	A		
	34 AT	ZP1	WP2	}	Setting up u of PS43 so that right number of lines will be transferred to output region
	35 LA	A17	17		
	36 TU	A	PS43	}	Setting up u of PS44 so transfer to VN can be started
	37 RA	PS43	ZP12		
	40 TU	GN2	PS44		Getting right address in VN to which transfer of lines is to be made
	41 TV	WP6	PS44	}	Transfer of excess-three-coded lines of print data to output region
	42 RA	PS44	BB20		
	43 RP	30000	HT	}	
	44 TP	30000	30000		
	CA	PS45			
	IA	TZ			
	0 RA	WP6	WP2		Number of lines accumulated plus number of lines transferred from last blockette gives total accumulated in WP6 to date.
	1 TP	A	VN		Total number of lines in output to first line of output.

2	RJ	WT	WT1	Getting output written on tape
3	MJ	0	PS	Exit
4	RJ	HT24	HT20	Getting new block read in and getting GN6 reset to zero
5	TP	VK	A	Is 1st line of new block a line of spaces? If not, it is assumed to be a line number and print is terminated
6	EJ	ZP2	TZ26	
7	SP	BB	17	Putting 0 VK VK into GN2
10	AT	BB	GN2	
11	MJ	0	TZ	
12	RJ	TZ25	TZ17	Setting up GN to proper 1st line address of next blockette
13	TU	GN2	TZ14	Is 1st line of blockette a line of spaces? If not, it is assumed to be a line number and print is terminated
14	TP	30000	A	
15	EJ	ZP2	TZ27	
16	MJ	0	TZ	
17	TP	GN6	A	Getting address of address of 1st line of current blockette to A _u Setting up u of next 2 instructions to this address of address of 1st line. Address of 1st line to A _u Address of 1st line to GN2 both in u and v Setting up GN2 to proper 2d line address of blockette to be examined Bringing up accumulative addition to date of number of lines in output
20	SA	BB21	17	
21	TU	A	TZ23	
22	TU	A	TZ24	
23	SP	30000	17	
24	AT	30000	GN2	
25	MJ	0	30000	
26	RJ	TZ25	TZ17	
27	RA	GN2	ZP15	
30	RA	WP6	WP2	
31	MJ	0	PS12	
	CA	TZ32		
	IA	HT		
0	RA	WP5	ZP1	Counts no. blockettes used
1	EJ	ZP	HT11	Have 6 blockettes been used?
2	TP	GN6	A	Is this the last blockette in the block?
3	EJ	ZP13	HT25	
4	RA	GN6	ZP1	Adding one to ordinal number of blockette to be checked
5	TV	BB22	NR1	TZ12 to v of NR1
6	RJ	NR	NR2	Check to see if Δ is in last blockette in last 2 significant lines.
				If not, return to TZ12.
7	RJ	TZ25	TZ17	Setting up GN2 to proper 1st line address of next blockette
10	MJ	0	TZ	Is the current blockette No. 5 and hence the last one in block?
11	TP	GN6	A	
12	EJ	ZP13	HT16	Increasing blockette number by one as we go on to next
13	AT	ZP1	GN6	
14	RJ	TZ25	TZ20	Setting up GN2 to address of 1st line of blockette next to be examined by String-out Control

Setting up GN2 to proper 1st line address of next blockette to be examined

Getting new block in.	15	MJ	0	TZ	
	16	RJ	HT24	HT20	Getting a new block of data in
	17	MJ	0	TZ7	
	20	TP	BB17	GT3	Reading in another block of data to VK
	21	RA	13	ZP1	Keeping up count of number of blocks read in by string-out
	22	RJ	GT2	GT	
	23	TP	ZP6	GN6	
	24	MJ	0	30000	
	25	TV	BB6	NR1	TZ4 to v of NR1
	26	RJ	NR	NR2	Check for Δ . in last 2 significant lines in blockette. (Goes to TZ4 if no Δ . is found)
	27	MJ	0	HT16	Returns here if Δ . has been found
		CA	HT30		
		IA	NR		
	0	MJ	0	30000	Exit if a Δ . is found
	1	MJ	0	30000	Exit if no Δ . is found
	2	SP	WP1	17	Address of last significant line of blockette is sent to u of VR5
	3	TU	A	VR5	
	4	RJ	VR10	VR	Check for Δ . in last significant line
	5	TU	VR5	NR7	Check for . in leftmost position on last line
	6	TP	ZP5	Q	
	7	QT	30000	A	
	10	EJ	ZP7	NR12	
	11	MJ	0	NR17	Check for Δ in rightmost position on next to last significant line
	12	TU	VR5	NR15	
	13	RS	NR15	ZP3	
	14	TP	ZP16	Q	
	15	QT	30000	A	Check for Δ . in next to last significant line
	16	EJ	ZP1	NR	
	17	RS	VR5	ZP3	
	20	RJ	VR10	VR	
	21	MJ	0	NR1	
		CA	NR22		
		IA	VR		
	0	TP	ZP14	WP	Index 4 to WP
	1	TP	ZP10	WP4	Mask to WP4
	2	TP	ZP11	WP3	Δ . to WP3
	3	LQ	WP3	6	Δ . shifted 6 places
	4	LQ	WP4	6	7777 shifted 6 places
	5	QT	30000	A	Portion of line is masked out
	6	EJ	WP3	NR	Is it equal to Δ . in shifted position?
	7	IJ	WP	VR3	
	10	MJ	0	30000	
		CA	VR11		
Loop to locate Δ .					

	IA	BB		
0	0	0	VK	Address of beginning lines of blockettes in input
1	0	0	VK24	
2	0	0	VK50	
3	0	0	VK74	
4	0	0	VK120	
5	0	0	VK144	Addresses of last lines of blockettes in input
6	0	0	TZ4	
7	0	0	VK23	
10	0	0	VK47	
11	0	0	VK73	
12	0	0	VK117	Address of address of last line of 1st blockette
13	0	0	VK143	
14	0	0	VK167	Used to help to get proper mask for changing 1st line of blockette that is to be printed
15	0	0	BB7	
16	0	0	ZP15	Parameter to use GT in reading in another block
17	50	00105	VK	Address of 1st line of output
20	0	0	VN	Address of address of 1st line of 1st blockette
21	0	0	BB	
22	0	0	TZ12	
	CA	BB23		

	IA	ZP		
0	0	0	6	Line of spaces
1	0	0	1	
2	01	01010	10101	
3	0	1	0	Masks
4	0	0	77777	
5	77	0	0	
6	0	0	0	Period
7	22	0	0	
10	77	0	77	Mask
11	22	0	01	. 0 Δ
12	0	30000	0	Index
13	0	0	5	
14	0	0	4	
15	0	1	1	Masks
16	0	0	77	
17	0	0	7777	
20	0	7	77777	
21	0	777	77777	
22	0	77777	77777	
23	77	77777	77777	
	CA	ZP24		

Temporary Region - WP

- 0 Index for search for Δ .
- 1 Holds address of last significant line in blockette
- 2 Number of lines transferred per blockette
- 3 Holds Δ . shifted for comparison
- 4 Holds mask shifted
- 5 Counter for number of blockettes examined
- 6 Number of output lines - accumulative

IF STRING-OUT

There are two separate analyses in this routine - one of the first clause started by IF subroutine and the second of one or two succeeding clauses started in IU subroutine. Each of these control routines has in turn several sub-routines which handle different facets of the string-out.

Examples of typical IF sentences are shown below:

- 24. If $X < Y$ jump to sentence 7, if $X > Y$ jump to sentence 74,
if $X = Y$ jump to sentence 32.5 Δ .
- 36. If $X > = Y$ jump to sentence 65, if $X < Y$ jump to sentence 44 Δ .
- 13.1 If $X \text{ NOT} = Y$ jump to sentence 2, if $X = Y$ jump to sentence 36 Δ .
- 12. If $X > Y$ jump to sentence 6, if $X < = Y$ jump to sentence 52 Δ .
- 11. If $X = Y$ jump to sentence 41, if $X \text{ NOT} = Y$ jump to sentence 31 Δ .
- 40.4 If $X < Y$ jump to sentence 30, if $X = Y$ jump to sentence 4 Δ .
- 76. If $X = Y$ jump to sentence 50 Δ .
- 72.3 If $X < = Y$ jump to sentence 42 Δ .
- 21.2 If $X(i,j,k,1) > Y(i,j)$ jump to sentence 3, if $X(i,j,k,1) < = Y(i,j)$ jump to sentence 5.2

Only one set of variables or constants is permitted in any one IF sentence. Each of the first five examples above exhausts all the possible relations between X and Y. No duplication of relations is permitted in separate clauses.

Throughout this write-up the set of variables or constants will be referred to as X and Y, X being the left-hand value and Y the value on the right of the relation symbol. In actual use, of course, X and Y may assume any and all of the combinations of letters and figures that constitute variables and constants. No distinction is made between X and Y. Restrictions on any one apply equally to the other.

X and Y may be numbers in scientific notation form. The latter ideally is a number in decimal form between 1 and 10 times a power of 10. Actually any number up to 12 decimal digits is permitted to be the left-hand member of a scientific notation number. Other variations taken care of by the program are revealed in the examples given below:

$1.234 \text{ e } 34$
 $3.4567235641 * 10^{20}$
 $6.8924 * 10 \text{ POW } 29$
 $9.67 * 10 \text{ POW }^{18}$
 8.3276 e ^{-23}
 $-5.298765 * 10 \text{ }^{33}$
 $2.3678987654 * 10^{-16}$
 $4.7 * 10^{-22}$
 $3.56 * 10^{-34}$
 -2.6784 e ^{-39}
 -2.6784 e ^{-39}
 -2.6784 e ^{-39}
 $3.3786 * 10^{-23}$

The asterisk, used as a multiplication sign in UNICODE, is the only binary operator permitted in the IF sentence, and it is only allowed in scientific notation. Note that the superior negative sign may be used in front of the power of 10 instead of the regular negative sign, but the superior sign may not be used in front of the left-hand member of a scientific notation number. Following an asterisk 10 must appear. The next number is assumed to be the power of 10 whether superior or lower-case. Following e (exponent) the next number occurring is assumed to be the power of 10.

Too high a value of exponent may give a floating-point number which is too large to represent. Ordinarily such a number would cause a machine fault and stop. A subroutine IQ is used in this connection to locate such discrepancies, avoid the machine-faulting stop, and give an error print-out. See separate write-up on this subroutine.

No expressions are allowed in the IF sentence. No plus signs are permitted. The absence of a negative sign indicates positive. Thus, some examples of what X and Y may not be are: $a + b$, $a - b$, a/b , $Z + a * b$, a^2 , b^a , $a^{3/2}$.

If such evaluations are to be compared, they must be computed separately by other instructions first and then referred to in the IF sentence by the simple variable to which they have been equated.

In the following chart are given the relation symbols permitted by the IF routine. NOT, occurring in the second position of a symbol set, is not

acceptable. Thus, NOT NOT is not acceptable, but other double-up relation symbols are interpreted as single symbols. For example, << is taken as meaning <. If NOT occurs alone as a single relation, it is accepted without error reference with the print-out: (NOT) interpreted to mean (NOT EQUAL).

Preferred Form	Acceptable Variations			Code Figures Assigned to Relations
<	< <			2
=	= =			3
>	> >			4
< =	= <	NOT >		5
NOT =	<>	><	NOT	6
> =	= >	NOT <		7

The code figures assigned to these relations are used by the routine to check on non-duplication of relations and to facilitate the reversal of a relation symbol when needed. $6 - \{2,3,4\} = \{4,3,2\}$ and $12 - \{5,6,7\} = \{7,6,5\}$. This enables < to be changed to > and $\leq$ to $\geq$, and vice versa. = is never changed to NOT =, and vice versa. A total of three single-relation code figures is always 9 when all possible relations of X and Y have been postulated and there is no duplication. Similarly, the total of a double-relation and a single-relation, non-duplicating symbol is 9.

The X and Y initially examined in the first clause form the set that is stored for later comparison in the running program. The sets of X and Y that appear in the second and third clauses are compared to the first set as a check on the validity of the sentence.

Ideally any set of X and Y should be repeated with the same order and with the same set of signs in the second and third clauses as it had in the first. If care is taken to do this, the routine operation is speeded up. However, failure to do this does not necessarily invalidate the sentence. If it can be done without changing the meaning, the program will reverse the relation symbol to correct altered order and signs. See section describing technical operation of JF for details on the theory back of relation-symbol reversal. An example

of how it works is the following: Let $X < -Y$ be contained in the first clause and $-Y < X$ in the second clause. The routine interprets the latter as $X > -Y$, and stores the $>$ symbol in output.

If the second or third variable set cannot be equated to the initial set by a consistent sign change, an error print-out will occur. All that is stored from the second and third clauses of an IF sentence for use in the running program are the line numbers to which jumps are to be made and the relation code figure for the second test, if any.

No more than two tests are needed in a running program for an IF sentence. A third IF situation following two tests is always an unconditional jump. Similarly, if the first test is for a double relation, its failure makes a second IF hypothesis true, necessitating an unconditional jump, and vice versa. In the last situation VN6 and VN7 (the second test and line number storage) are left empty, VN10 gets the line number of the unconditional jump, and VN4 and VN5 take the first test code and line number, respectively. See attached sheet explaining output.

To return to additional specifications in writing X and Y, a superior minus sign is not permitted to indicate negative value of X or Y. If an absolute sign has already appeared, a negative sign following it will be ignored since the absolute sign negates its meaning. An absolute sign causes the indicator for such to be put into the proper line, regardless of the previous appearance of a negative sign.

An absolute sign appearing in front of a variable or constant applies only to that variable or constant. The absence of a closing absolute sign after the value is not noted. However, the appearance of a closing absolute sign causes a check to see if an open absolute sign has been recorded. If not, there is an error print-out.

Only a single value is stored for a function in UNICODE. The use of a variable function in an IF sentence presupposes its computation or read-in via a previous instruction. Hence arguments following a function, whether of constants or variables, are ignored both in analysis of the first clause and in analysis of the second and third clauses. Also differences between these superfluous arguments between clauses are not noted.

If either X or Y is a fixed-point variable, the other must be a fixed-point constant or fixed-point variable. If either X or Y is a floating-point variable, the other must be a floating-point number or variable. Based upon such considerations, the routine translates numbers to octal or floating point, gives them call words, and stores them in list CL.

If both X and Y are constants, they are sent to the subroutine IT, where an immediate comparison is made of them. If the test shows the relation true, the sentence is changed to a jump sentence and the string-out output modified accordingly. G_9 (see print-out schedule) alerts the operator to this decision without error reference.

If the test fails, a jump is made to the beginning of the IF routine and the remaining sentence clauses (if any) are processed as a separate IF sentence. This is the reason recognition of the IF symbol is built into the first part of IF string-out.

If specific sets of arrays of values of unknowns are to be compared, subscripted variables should be used. Up to four subscripts may be used for each variable. Subscripts in an IF sentence may not be expressions. However, a subscripted variable used for X or Y in the second or third clauses must agree in every particular with that used in the first clause. As an example, suppose " $X(i,j,k) < Y(i,j,k,l)$ " is in the first clause; $X(i,j,k)$ and $Y(i,j,k,l)$ must also appear in the second and third clauses.

All print-outs include as a first element, Sentence_____(IF), where the dash indicates a line number which will be typed out. Some print-outs merely give information on how the program has handled an unusual situation. These do not cause a reference to the error routine or termination of the IF string-out. Other print-outs are accompanied by an error reference but not termination. The more serious type includes both error-referencing and IF string-out termination. Some without built-in termination are used most of the time with external termination. In the attached chart of print-outs, subscripted G's are assigned for simplification of following technical description of the segments of the IF string-out.

Error Reference	Identification	Print-Out	Built-In If Termination
	G ₁	Sentence_____ (IF)	
✓	G ₂	Inconsistent sign change	✓
	G ₃	Symbol rejected_____	
✓	G ₄	Disallowable character in Exponent_____	
✓	G ₅	Incorrectly written_____*	
✓	G ₆	Scientific notation in- correctly written	✓
✓	G ₇	Open absolute sign missing	
	G ₈	T0 should follow jump	
	G ₉	Becomes Unconditional Jump to Sentence_____	
✓	G ₁₀	Space period occurs before sufficient data given	
✓	G ₁₁	Comparison symbols ambiguous	✓
✓	G ₁₂	Set of variables differs from initial set	✓
	G ₁₃	(NOT) interpreted as (not equal)	
✓	G ₁₄	Fixed and floating point values are not comparable	✓

*The symbol being examined when this print-out occurs is given here. It indicates how far in the sentence analysis has gone before external termination.

IF Control

IF subroutine is the control of analysis of the first clause of the IF sentence. It clears VN output lines (except for first four) and clears temporary storage lines VN71-121. See attached charts for explanation of data stored in these addresses. VN71-121 is used first to accumulate data on X. Then this data is transferred to VN40-70, leaving VN71-121 for use in accumulating data on Y.

The divider routine II is set up so that it can be entered only once without error during the first-clause analysis. The first two symbol output lines are sent to temporary storage. A pseudo-op indicator is put in VN33, if needed.

Recognition of , ; IF (causes a return to the SY referencing instruction that gets the next symbol. Δ . recognition sends analysis to the IK termination routine. If the symbol is a constant, control goes to IE. IG takes over for a fixed-point variable and IH assumes control for all other variables.

A character not identified causes print-out G_3 and then a return to get next symbol.

IK-- Δ . Termination

Zero value is checked for successively in VN5, VN4, VN14, and VN24. If found in any, print-out G_{10} ensues with termination. If not found, WT, the tape-write routine, is referenced to get completed data written on tape before return to CT, the String-out Control Routine.

From the analyses of the second or third clauses, an entry is made to IK14. The absence of zero in VN10 sends the routine to the loop mentioned above. If a zero is found in VN10, a check is made if VN34 equals 00 00003 00000. If yes, G_{10} occurs with termination. If no, VN7 and VN6 are successively examined for zero. If found, G_{10} occurs with termination. If not found, a jump is made to the loop of zero checking described above.

IE Constant Analysis

Reference is made to RB for checking validity of a constant. A constant indicator is stored in VN31 for X and VN32 for Y.

If the number of characters is > 6 , the number is converted to floating point via proper subroutine. Also, a floating-point constant indicator is filled with 40 0 0. This is necessary because a floating-point number may be zero and its presence as such in regular floating-point location could not be recognized.

Either a superior or regular dash is permitted to indicate a negative sign following e or * in scientific notation. Failure of a figure to follow this sign causes G_6 . 10 not following * causes G_6 . 10 may not be a superior figure. After 10, there may be POW, but after this either a negative sign or figure must be next to avoid G_6 .

An exponent is checked by RD subroutine. If superscript, it is sent to IN, the Superior to Lower-Case Translator, where G_4 may occur. See separate write-up on this subroutine.

The exponent is then converted to octal by use of RS subroutine and transferred to proper input line IQ2 either as a positive or negative number. After checking and conversion to floating point in IQ (see separate write-up), the number is changed to negative, if desired.

Failure to find an open absolute sign when a closed one has shown up causes G_7 .

A relation symbol $<$, $>$, $=$, or NOT causes a jump to II subroutine. JUMP symbol sends control to IJ subroutine. Characters () , ; initiate a return to beginning of loop to get next symbol. An unrecognized symbol causes G_3 without termination. This loop, starting at IE115, is used frequently as an exit by other subroutines.

IG--Fixed-Point Variable Analysis

After a reference to RH to check for validity of variable, the call word is obtained from TS if it is there, or from TA when it is in the Combination List. If the call word is not in the latter or the pseudo-op list, it is secured by adding 64000 to a number put in A by RJ tk tk1. It is then added to the combination list by using TF and TE. Explanation of uses of these routines is found in separate write-ups on each.

IH and IL--Floating-Point Variable Analysis

On entry to IH, two separate courses are taken, depending on whether the sentence is within a pseudo operation or not.

If a subscripted variable is identified, the modulus (the number of words stored for the variable) is assigned a call word, and this call word put in the u position of an output line. In the same word in the v position is put the number of subscripts.

The format of the data on a subscripted variable obtained from the combination list is as follows:

TA4	0	0	(call word)
TA5	0	(modulus)	(number of subscripts)
TA6	0	(2d subscript multiplier)	(1st subscript multiplier)
TA7	0	0	(3d subscript multiplier)

The proper multipliers for the subscripts of variables not in the pseudo-op list are obtained from region TA by means of a small subroutine entered via 2 one-shot RJ switches. The order of entry (only three for any set of subscripts) determines which multiplier is secured. The Constant call word assigned to it is then transferred to the u position of the output line. The v position of this line is later filled with the call word of the fixed-point variable or constant involved.

The call word of the pseudo-op subscripted variable is in the form 0 0 76ZRR where Z is the number of subscripts and RR the current pseudo-op variable number. Thus, if 3 single-valued variables have been assigned positions in the pseudo-op list, (0,1,2), RR will equal 3. The routine extracts the Z and sets up an index for processing the subscripts. To the extracted RR is added 63000. This number becomes the call word of the subscripted variable. The number of subscripts is added to 630RR to give a call word assigned to the modulus. The call words of the subscript multipliers start with (630RR + 1). As subscripts are encountered, they are supposed to have call words starting with (630RR + Z + 1). Below is given an example of pseudo-op input region for the case of three subscripts.

Assigned Call word

63ORR	Subscripted variable
63ORR + 1	I_M - 1st multiplier
63ORR + 2	J_M - 2d multiplier
63ORR + 3	Modulus
63ORR + 4	Subscript 1
63ORR + 5	Subscript 2
63ORR + 6	Subscript 3

A subscripted variable in a pseudo operation may be a regular floating-point variable or pseudo-op variable. Whichever it is, it may have fixed-point variable subscripts located either in the pseudo-op list or combination list. If these subscripts are found in neither list, they are assigned call words in the 64000 range and put in the combination list. Note that if a subscript is a constant, it will not be in the pseudo-op list.

If a subscript of a subscripted variable fails to be either a constant or a fixed-point variable, G_5 occurs with termination.

When the call word of a nonsubscripted variable has been stored, it is examined to see if it is that of a function. If so, analysis goes to an IL exit subroutine in which any arguments that might be inadvertently put following the function are ignored. Exits and print-outs are the same as those in the IE exit subroutine. If the call word is not that of a function, the exit from IL is made via the IE exit subroutine.

II--Relation Symbol Analysis

No more than one entry can be made to this routine during an IF instruction. A second entry causes G_5 with termination. Failure to have a second relation symbol in the clause follow immediately after the first will create this error. A "NOT" occurring in the second position causes the same error termination.

The data gathered on X is transferred by this coding from VN71-121 to VN40-70 and the first region is cleared for succeeding data to be gathered on Y.

IJ--Jump Subroutine

Failure of the next symbol to be "TO" causes G_8 without error reference. A loop of symbol recognition similar to that used in the separate jump instruction string-out is included at the start. See the write-up on this routine. Failure to meet the specifications of the loop causes G_5 with termination. Getting the assumed line number into proper form terminates this first loop and may cause error print-outs explained in the line-number-routine write-up. Regardless of what happens in the line-number routine, its output is transferred to VN5 and the string-out continued. The referenced number is sent via IX routine to reference list IZ where it is later given a call word by other routines.

Variables or constants X and Y are compared in the second part of this subroutine. Combinations of floating-point and fixed-point values cause G_{14} .

Fixed-point numbers or those assumed to be such are not translated from excess-three decimal code until the analysis of this subroutine. Then they may be translated either to octal or floating point, depending upon the nature of the corresponding variable or constant.

VN11 to VN30 output lines are now filled with the data that has been gathered in the temporary storage regions.

The closing loop checks for parentheses, comma, semicolon, space period, and IF. G_5 ensues with termination if none occur. Recognition of IF sends the analysis to IU, the beginning control routine for analysis of the second clause.

IU--Second and Third Clause Control

The absolute sign indicators for X and Y are transferred to the temporary storage lines directly above the excess-three representation of X and Y, respectively. Thus 11 identifying lines on each of X and Y are in one location for future reference.

Exits from this control routine are IY, IY33, JB, and IK14, depending on whether X or Y is a constant, fixed-point variable, or floating-point variable, or whether Δ . has been encountered. Temporary storage region VN150-164 is cleared for gathering of data on X or Y. G_3 occurs without error reference for an unrecognized symbol.

IY--Number Analysis

This routine (the counterpart to IE) has similar checks on scientific notation. G_6 or G_{12} follow a misuse of the latter. Excess-three storage of data on X and Y is continued in it. The exit loop is in JB6-JB17 and is shared with variable subroutines. G_3 occurs if indicated following use of this loop.

JB--Floating-Point Variables

If a floating-point-variable symbol is not found in either the combination list or the pseudo-op list, it cannot be the same as any of the symbols encountered in the first clause. Hence G_{12} follows.

Excess-three representations of subscripts are stored in VN156-161.

The exit loop sends analysis to JC for a relation symbol, JF for a jump symbol, and IK14 for Δ . Additional errors and print-outs are similar to those of IH and IL.

JC--Relation Symbol Subroutine

Count and storage of excess-three relation symbols is the function of this routine. Also data on X is transferred to VN130-144, and region VN150-164 is cleared for use of Y data accumulation. If two relation symbols have been found, control is transferred to IU5 to get next symbol. Otherwise, the jump is made to IU6 since an unidentified symbol has still to be examined. G_5 followed by termination occurs if routine is referenced twice in a single clause analysis or if second symbol is "NOT".

JI--Relation-Symbol Check

Referenced from JF, JI examines the count of relation symbols, checks their consistency with previous counts, and assigns a code figure to the symbol either in VN6 or temporary location JI123 if all relations have been used in a sentence. An accumulative excess number (over 3) of relation symbols causes G_{11} .

JF--Second Jump Subroutine

Let us call the X symbol the second or third time through X_2 and the Y symbol Y_2 . This routine compares excess-three data gathered on X_2 in VN131-141 with similar data gathered on X the first time through. If an inequality is found, Y_2 data in VN151-161 is compared to X data. Similarly, as the situation requires, Y data is compared with Y_2 data and later, if needed, with X_2 data. Next signs are compared.

Since the only values obtained from the second and third clauses other than the line numbers are the relation symbols, it was convenient in a few lines to use a theory of symbol reversal to correct what might ordinarily have been errors in position and sign of variables. $X(s)$ means sign of X. Following are the rules that govern such reversal.

<u>Variables</u>	<u>Signs</u>	<u>Symbol Reversal</u>
$X = X_2$	$\left\{ \begin{array}{l} X(s) = X_2(s) \\ Y(s) = Y_2(s) \end{array} \right\}$	No
$Y = Y_2$	$\left\{ \begin{array}{l} X(s) \neq X_2(s) \\ Y(s) \neq Y_2(s) \end{array} \right\}$	Yes
$X = Y_2$	$\left\{ \begin{array}{l} X(s) = Y_2(s) \\ Y(s) = X_2(s) \end{array} \right\}$	Yes
$Y = X_2$	$\left\{ \begin{array}{l} X(s) \neq Y_2(s) \\ Y(s) \neq X_2(s) \end{array} \right\}$	No

All sign combinations other than the above cause G_2 . Failures to find equality of $\{X, Y\}$ with $\{X_2, Y_2\}$, exclusive of sign, in any order causes G_{12} .

After symbol reversal, if needed, the relation codes are compared for uniqueness and internal consistency. A loop to get the line number operates similarly to the one in IJ.

If VN10 has been filled with a line number, the next loop is the final exit. If only VN7 has a line number, a loop is entered which, on recognition of an IF symbol, will send analysis back to IU2 for the third and final analysis of an IF clause. The final exit loop permits only a closed parenthesis and Δ . The other loop permits (), ; Δ . IF. Failure of either to recognize a symbol causes G_5 with termination.

Prior to analysis of the third clause, this routine puts 00 00003 00000 in VN34. This indicator is later used to check if sufficient data is given in the third clause to complete the sentence string-out.

Output VN

0	0	0	34	Number of lines of output in v
1	Line Number			
2	IF77777777			Name of instruction
3	0	0	v	v = call word of line number.
4				Holds code figure $\in \{2,3,4,5,6,7\}$ to indicate relation test
5				Line number of sentence to be jumped to on first test.
6				Holds code figure in v for second relation test
7				Holds line number of sentence to be jumped to on second test
10				Holds line number of sentence to be jumped to unconditionally
11	0	u	v	u = Call word of modulus. v = number of subscripts of X
12				Negative value desired if 40 0 0, otherwise cleared
13				Absolute value desired if 40 0 0, otherwise clear
14	0	0	v	Call word of X
15	0	u	v	u = call word of I multiplier. v = call word of I
16	0	u	v	u = call word of J multiplier. v = call word of J
17	0	u	v	u = call word of K multiplier. v = call word of K
20	0	0	v	v = call word of L
21	0	u	v	u = call word of modulus. v = number of subscripts of Y
22				Negative value indicator
23				Absolute value indicator
24	0	0	v	Call word of Y
25	0	u	v	$\left. \begin{array}{l} u = \text{call word of subscript multipliers} \\ v = \text{call word of subscripts of Y} \end{array} \right\}$
26	0	u	v	
27	0	u	v	
30	0	0	v	
31				If X is a constant, 40 0 0, otherwise clear
32				If Y is a constant, 40 0 0, otherwise clear
33				Pseudo-op indicator, 40 0 0, otherwise clear

*Temporary Storage in VN

X	Y	
40	71	Floating-point constant
41	72	Fixed-point constant in excess-three
42	73	Used in making call word of multiplier
43	74	{ Indicator of fixed-point variable (40 0 0) Holds absolute-sign indicator during comparison Excess-three representation of variable
44	75	
45	76	
46	77	Sign of exponent indicator (40 0 0)
47	100	Exponent of 10 in excess three
50	101	Subscript 1 in excess three
51	102	Subscript 2 in excess three
52	103	Subscript 3 in excess three
53	104	Subscript 4 in excess three
54	105	u = call word of modulus; v = number of subscripts
55	106	Negative sign indicator (40 0 0)
56	107	Absolute sign indicator (40 0 0)
57	110	Call word in v
60	111	u = call word of I multiplier; v = call word of I subscript
61	112	u = call word of J multiplier; v = call word of J subscript
62	113	u = call word of K multiplier; v = call word of K subscript
63	114	v = call word of L subscript
64	115	Index for subscript assembly
65	116	Count of relation symbols
66	117	First relation symbol
67	120	Second relation symbol
70	121	Floating-point constant indicator (40 0 0)
X ₂	Y ₂	
130	150	Negative sign indicator (40 0 0)
131	151	{ Absolute sign indicator (40 0 0) Excess-three representation of variable
132	152	
133	153	

{ 134	154 }	Sign of exponent indicator (40 0 0)
135	155	Exponent of 10 in excess-three
{ 136	156 }	Subscript 1 in excess-three
137	157	Subscript 2 in excess-three
140	160	Subscript 3 in excess-three
{ 141	161 }	Subscript 4 in excess-three
142	162	Count of relation symbols
143	163	First relation symbol
144	164	Second relation symbol

*Braced portions are those compared during analyses of second and third clauses to determine equality of variables.

Scientific Notation Checking Routine

The purpose of this routine is to check the size of a number in scientific notation form and then convert it to floating point. If it is too large for floating-point representation, the error routine is referenced and the following error print-out is given: Sentence_____(_____)--Absolute Value of Number Too Large. If too small, the floating-point representation is set to zero and this print-out occurs: Sentence_____(_____)--Absolute Value of Number Too Small--Given Zero Value. In the latter case, the error routine is not referenced.

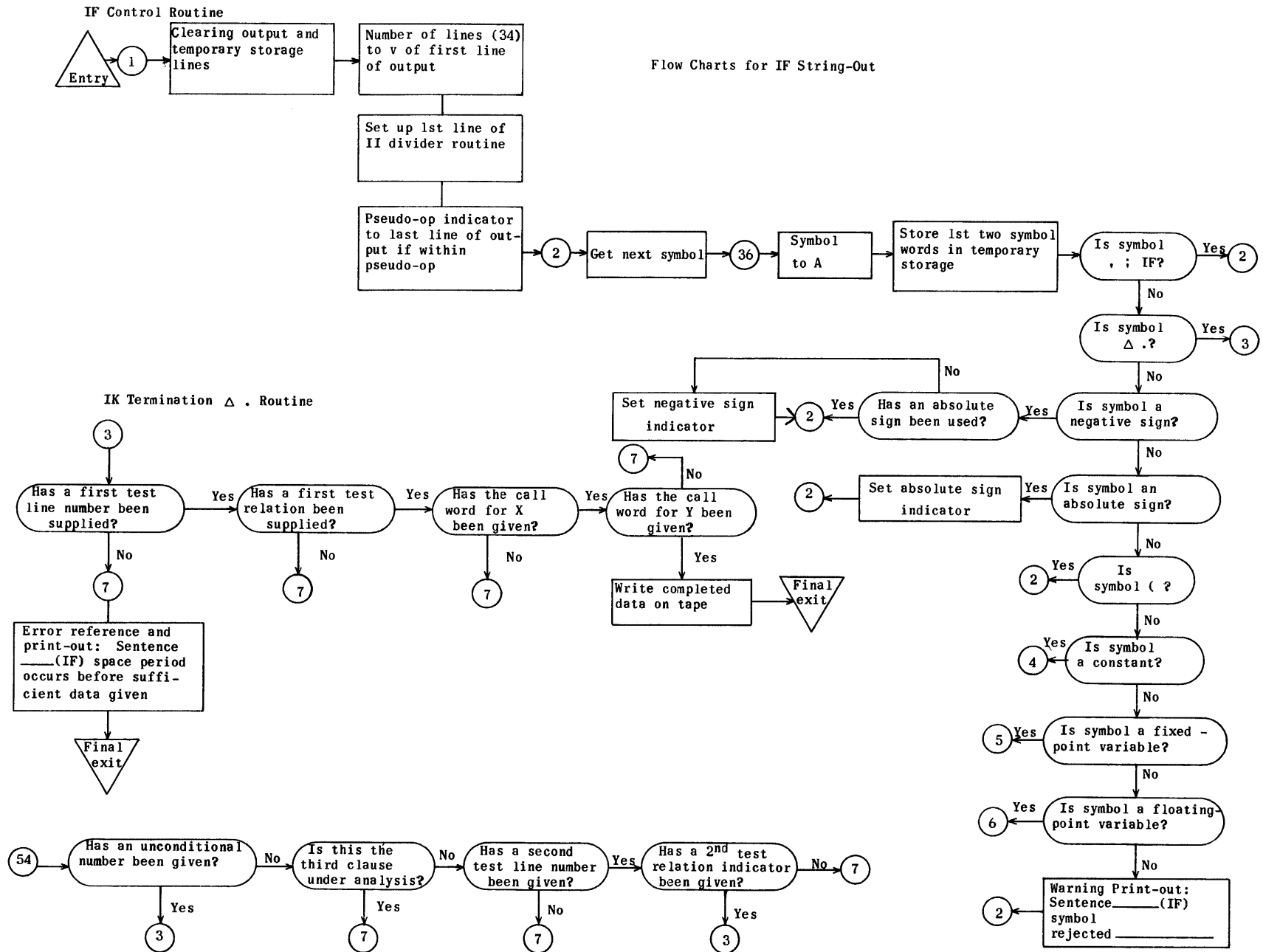
To use the routine a number to the left of e or * is first converted to floating point by use of the "excess-three decimal to floating-point" routine. This floating point number, either positive or negative, is put in input line IQ2. The exponent of 10 after conversion to octal by means of the proper routine is put in input line IQ3. It also may be either positive or negative. Instruction RJ IQ IQ1 then performs the check and, if the absolute value of the number is not too large, puts the correct floating-point representation into IQ2 as the output.

Superior To Lower-Case Translation of Figures

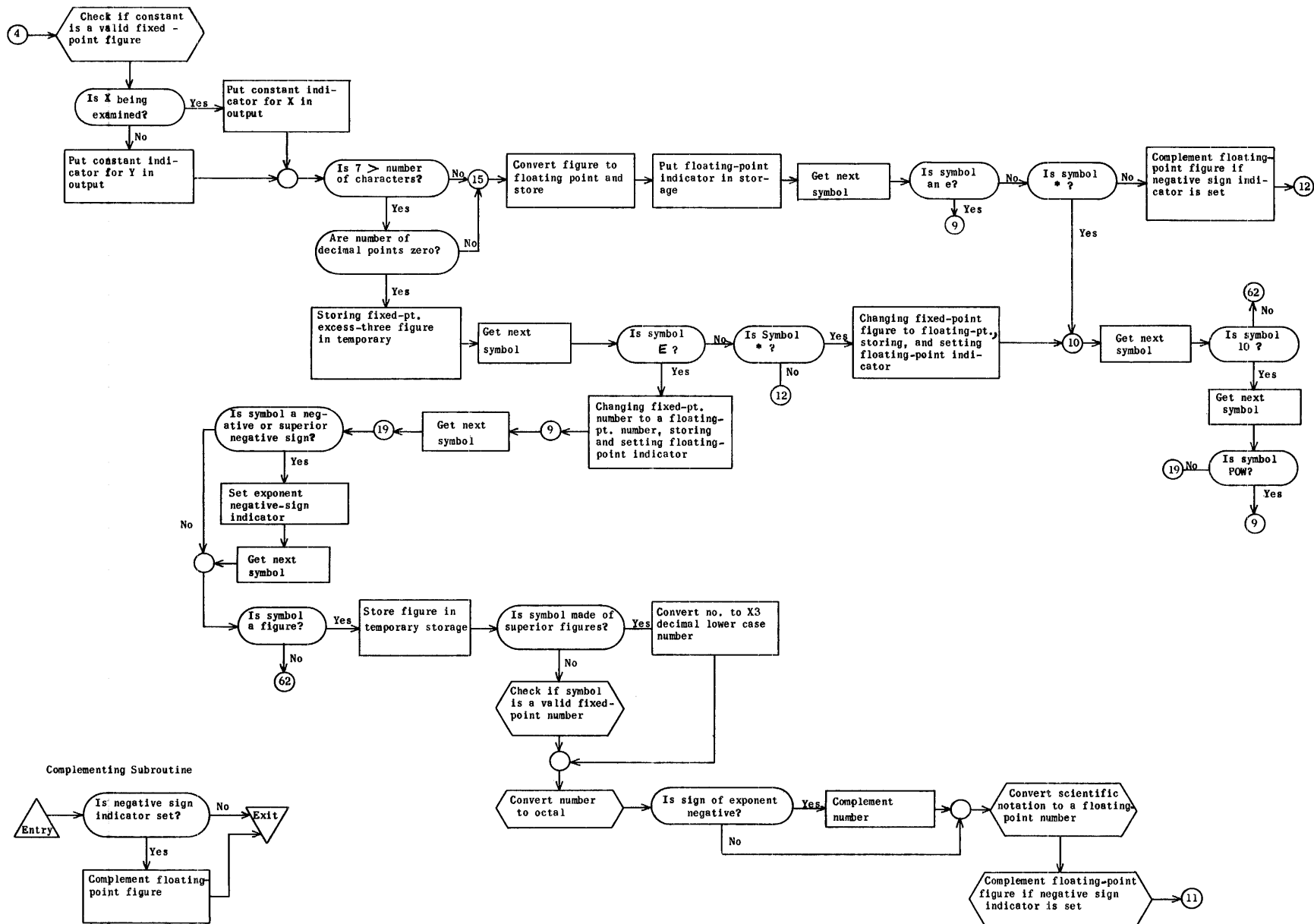
Input to this routine is one line of superscript figures packed starting at the left with 77 fillers at the right. No characters other than superscript figures are permitted. Thus periods are not allowed, and only integral figures are translated. Both the input and output are in excess-threedecimal code. The input line, in2, becomes the output line on termination of the referencing instruction, RJ in in1.

If the first character on the left is 77, no translation is performed. The input line, in effect, is untouched.

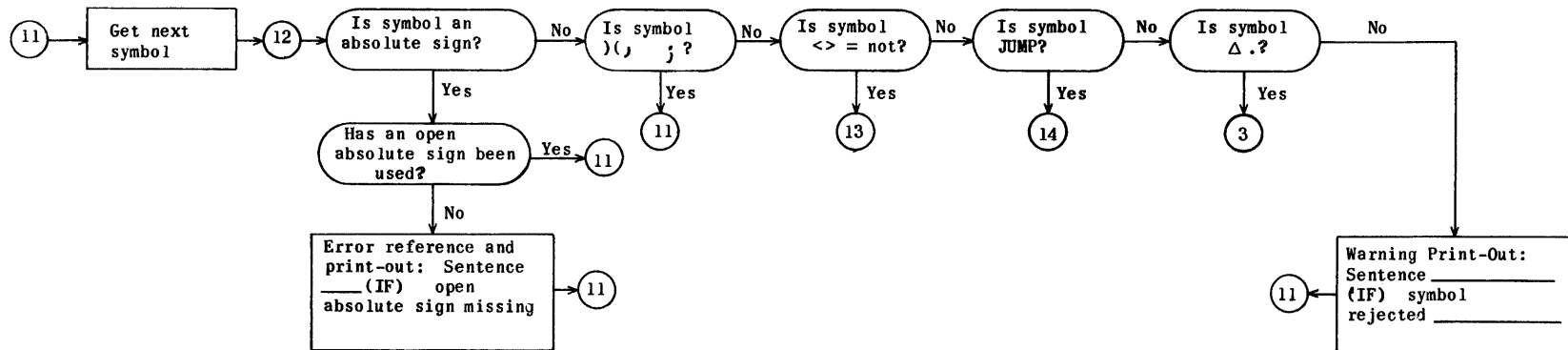
If a character is not a figure or a 77, the error routine uz is referenced and there is a print-out: Sentence_____(____). Disallowable Character in Exponent_____. In the last blank is given the print-out of the character. This print-out occurs for every erroneous character in the line prior to the first 77. When a 77 is encountered, no further analysis or translation of the line occurs. What has been translated is packed at the left, and the balance of the line is retained the same as it was in the original input.



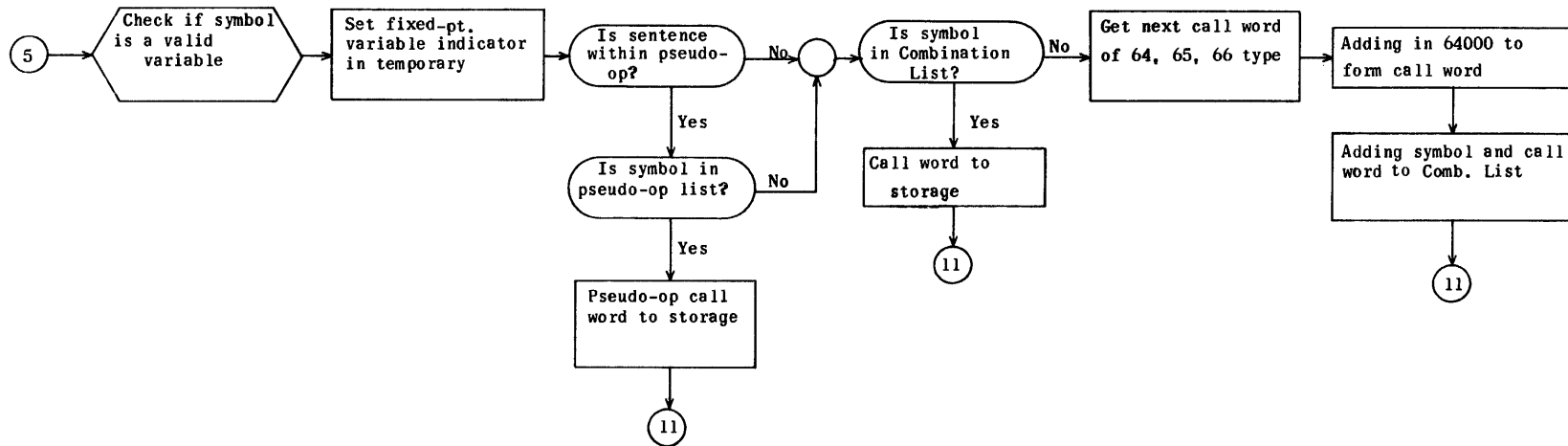
IE Constant Routines



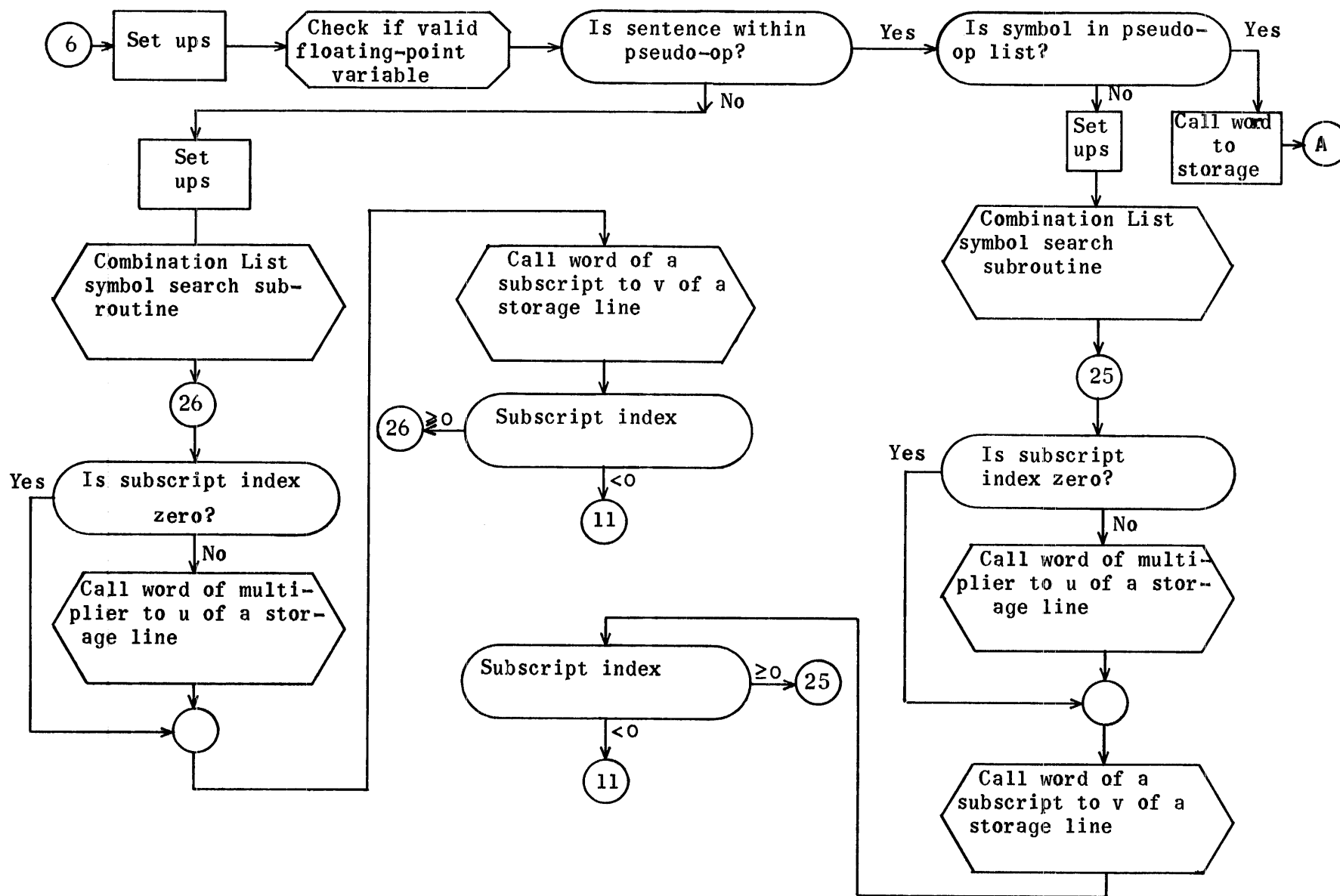
Constant Termination Loop

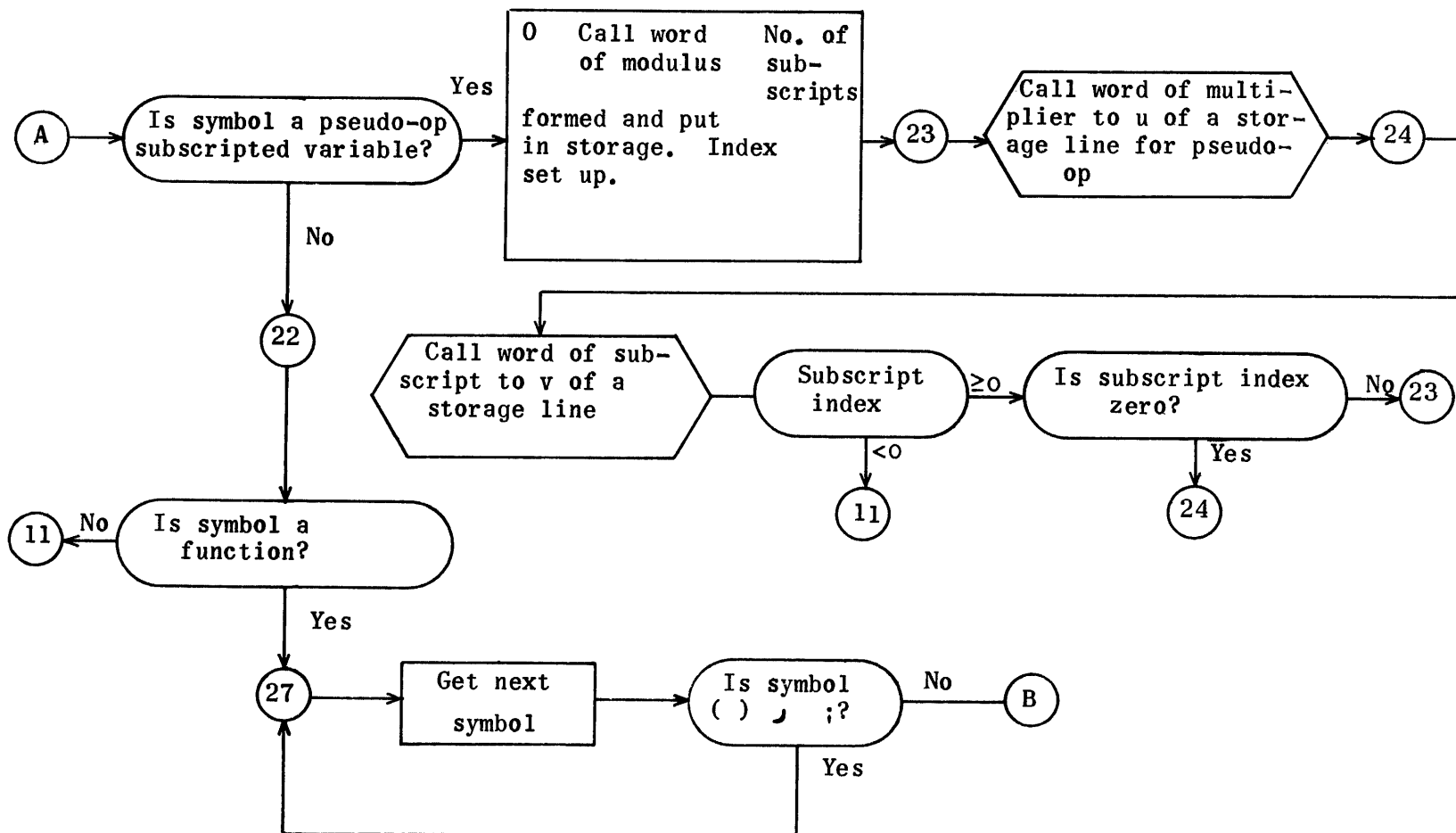


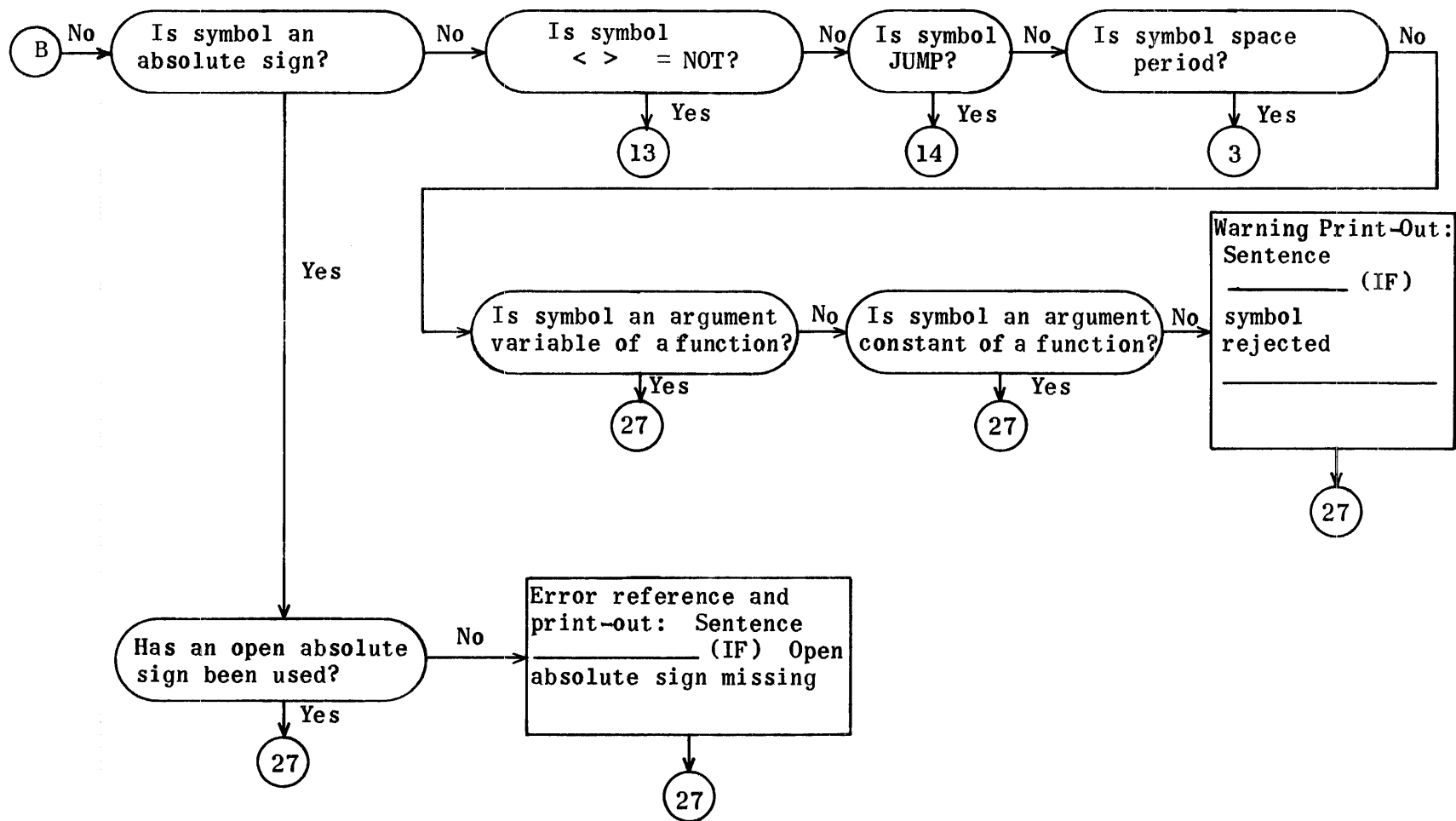
IG Fixed-Point Variable Routine



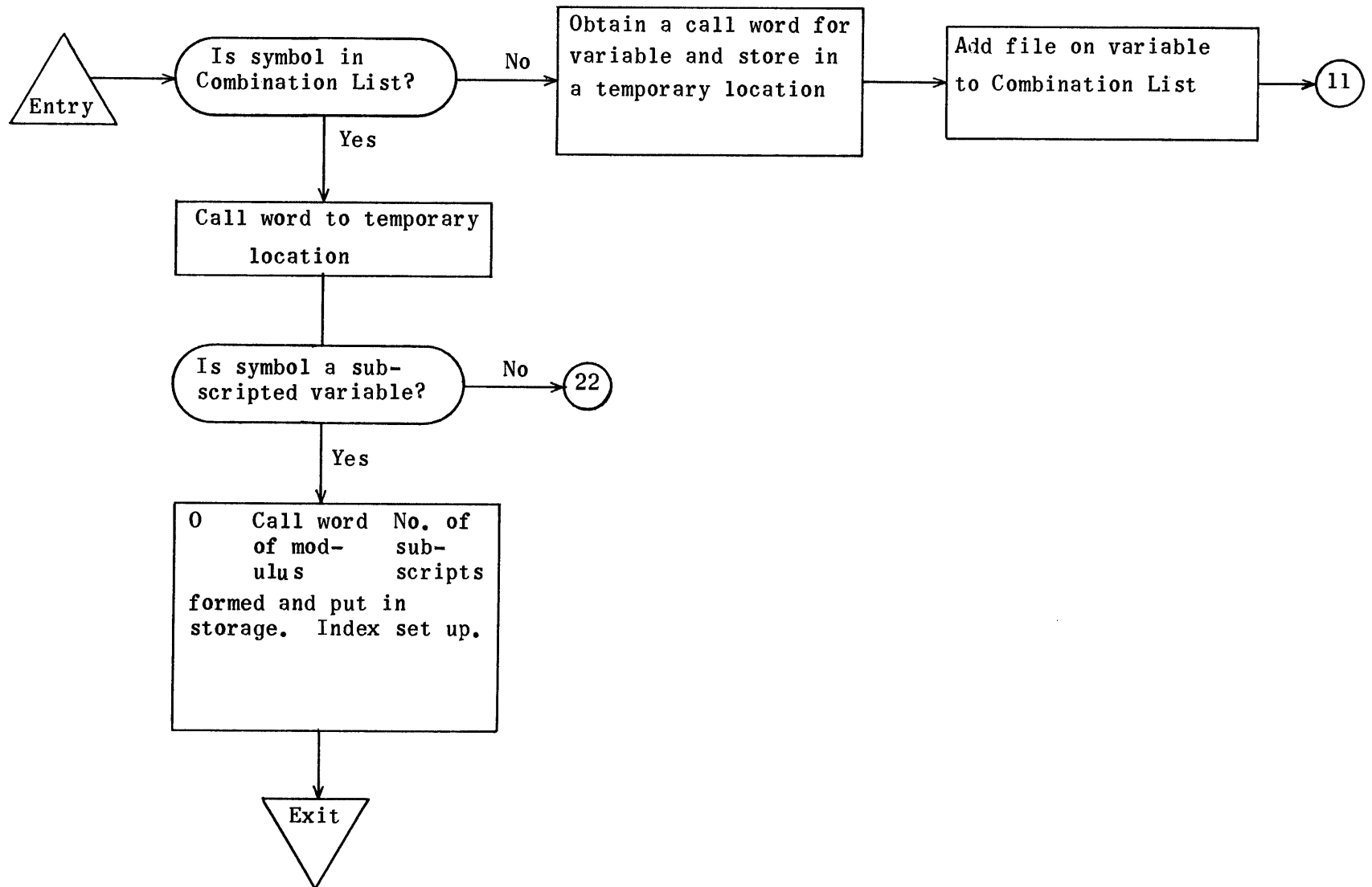
IH and IL Floating-Point Variable Routines



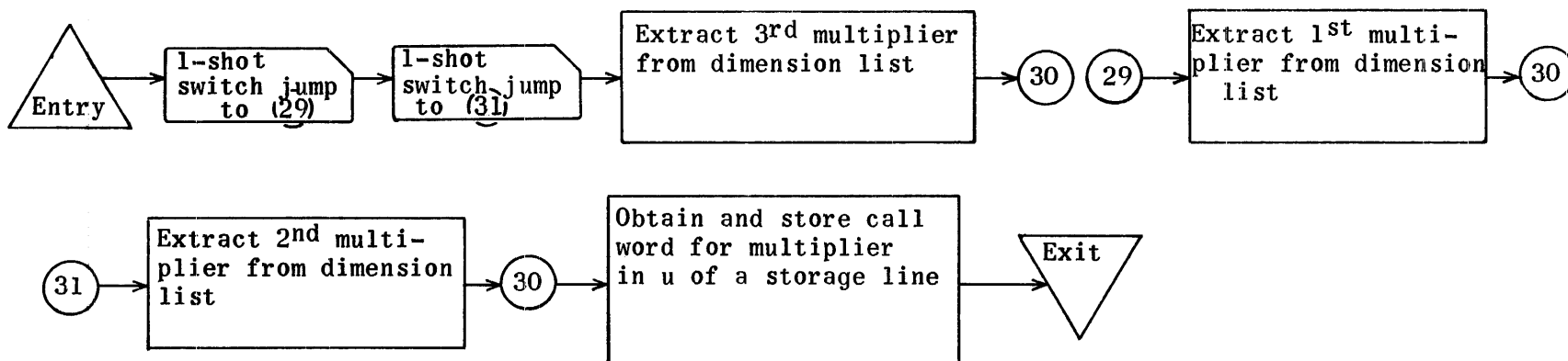




Combination List Symbol Search Subroutine

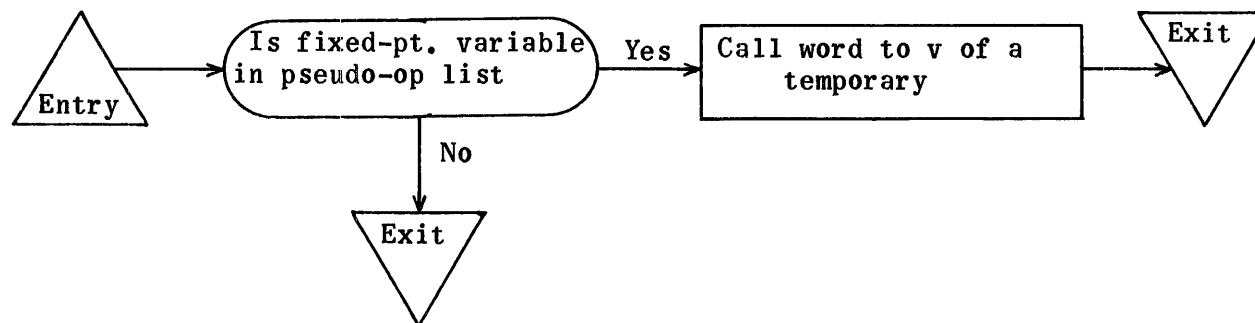


Subroutine to put call word of subscript multiplier (not for a pseudo-op sub variable) to u of a storage line

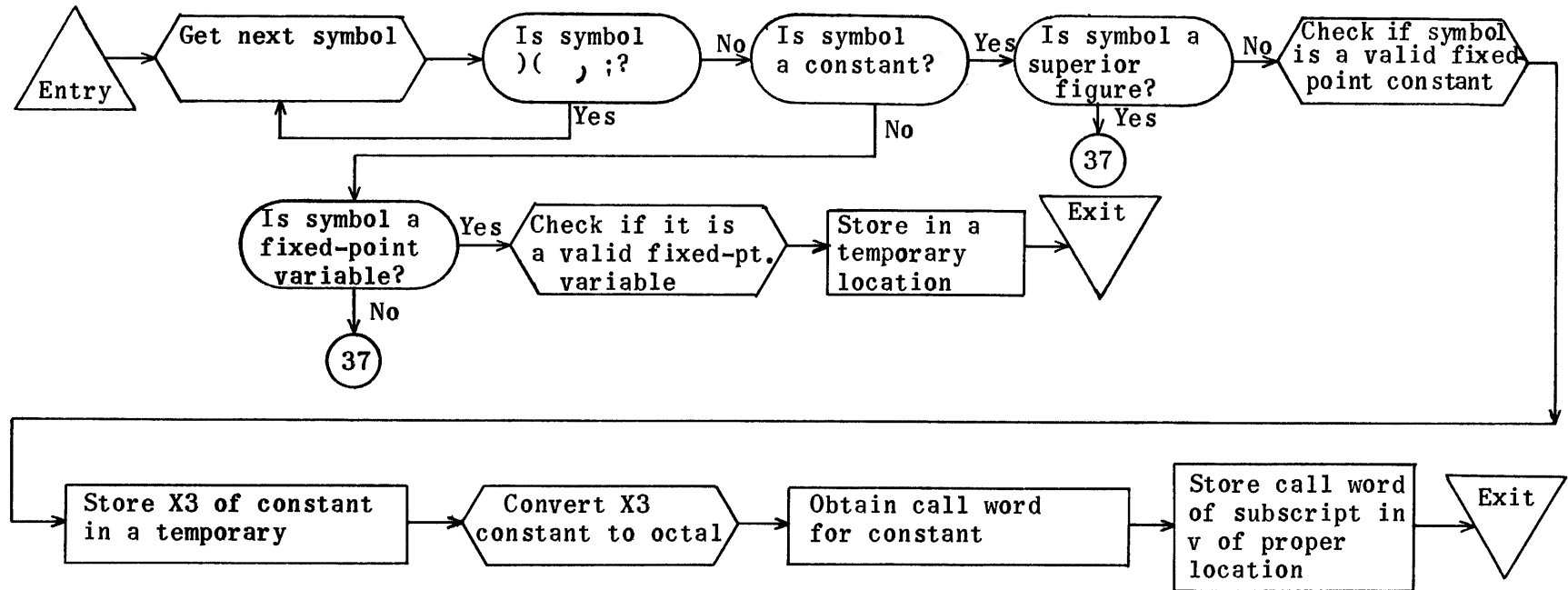


3 Subroutines to put call word of a subscript to v of a storage line:

2 - Checking if fixed-point variable is in pseudo-op list and getting call word, if so.

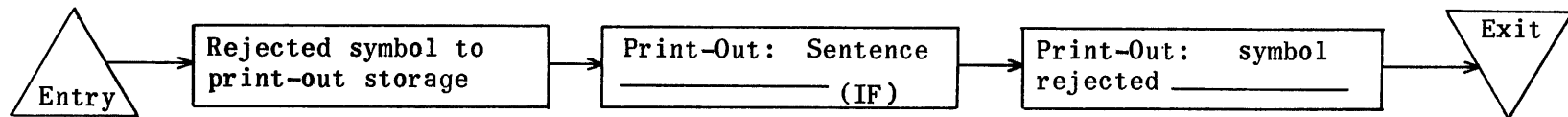


1-Handling subscript if a constant and checking if it is a fixed-point variable

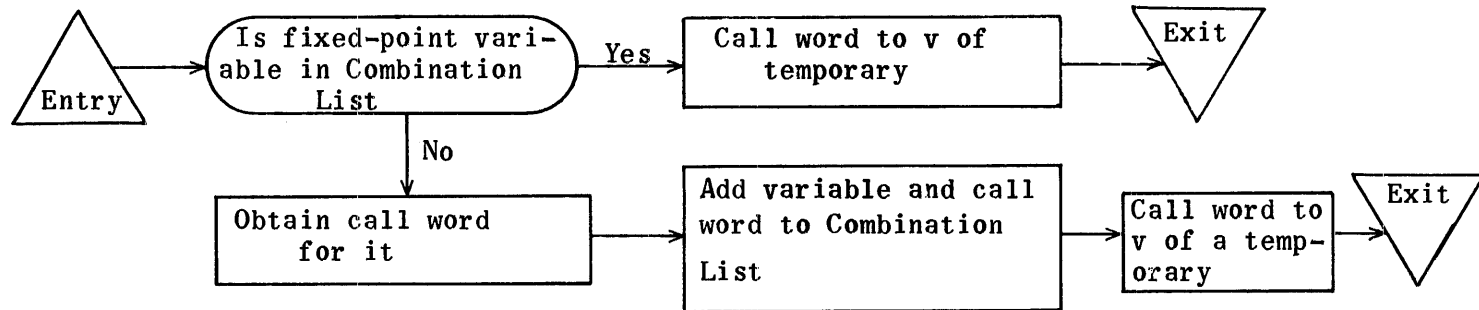


672

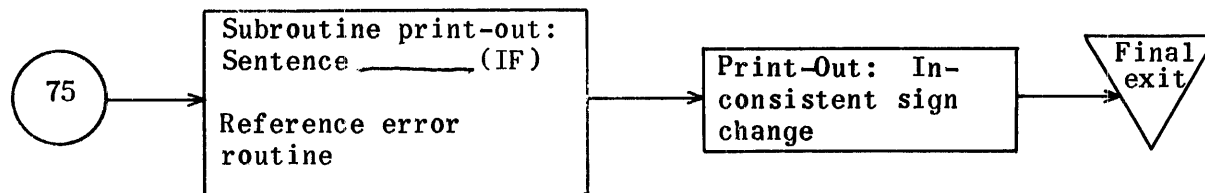
Sentence _____(IF) Symbol Rejected _____(Warning only)



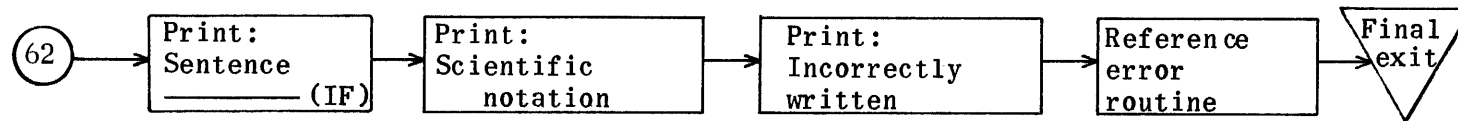
3-Checking if fixed-point variable is in Combination List and, if not, adding it in. Call word to location.



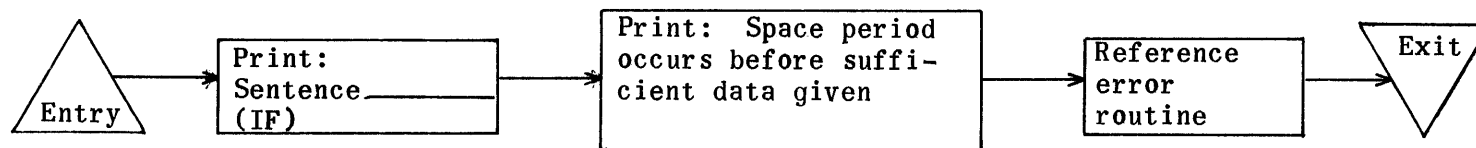
ID Print-Out Subroutine Sentence____(IF) Inconsistent Sign Change



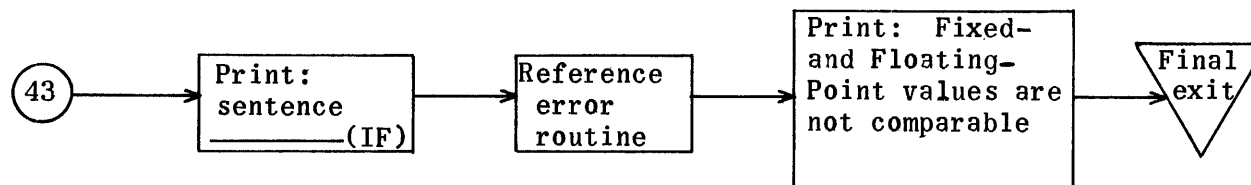
Sentence _____ (IF) Scientific Notation Incorrectly Written



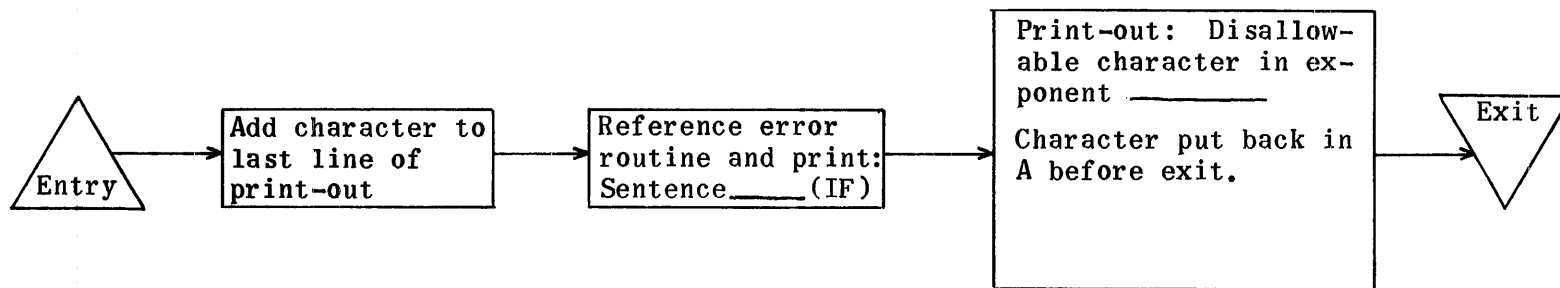
Sentence _____ (IF) Space Period Occurs Before Sufficient Data Given



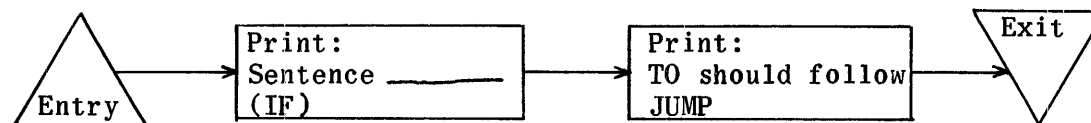
Sentence _____ (IF) Fixed- and Floating-Point Values Are Not Comparable



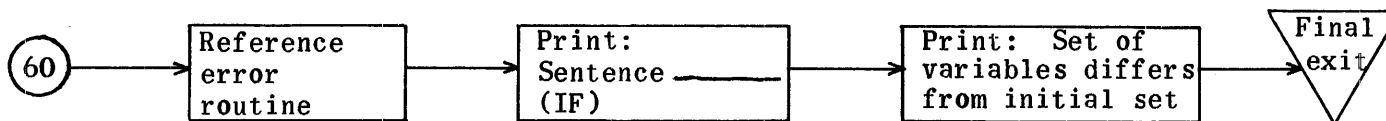
Sentence____(IF) Disallowable Character in Exponent _____



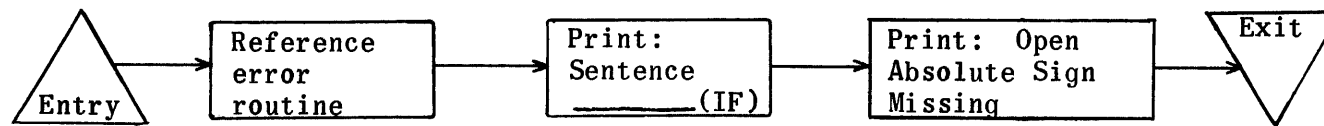
Sentence____(IF) TO Should Follow JUMP (Warning only)



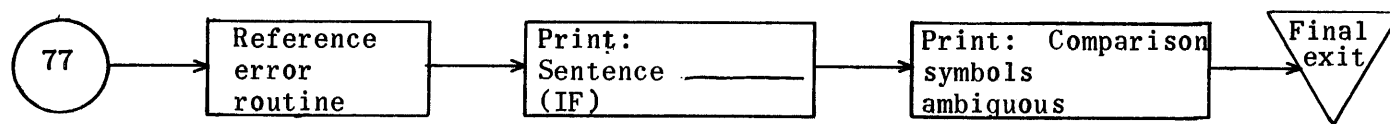
Sentence____(IF) Set of Variables Differs From Initial Set



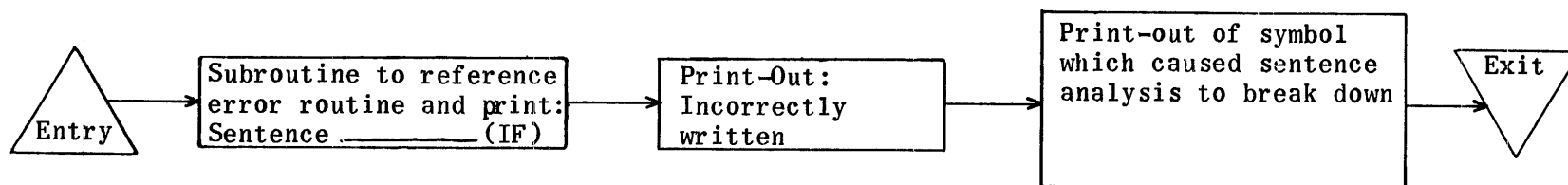
Sentence _____ (IF) Open Absolute Sign Missing



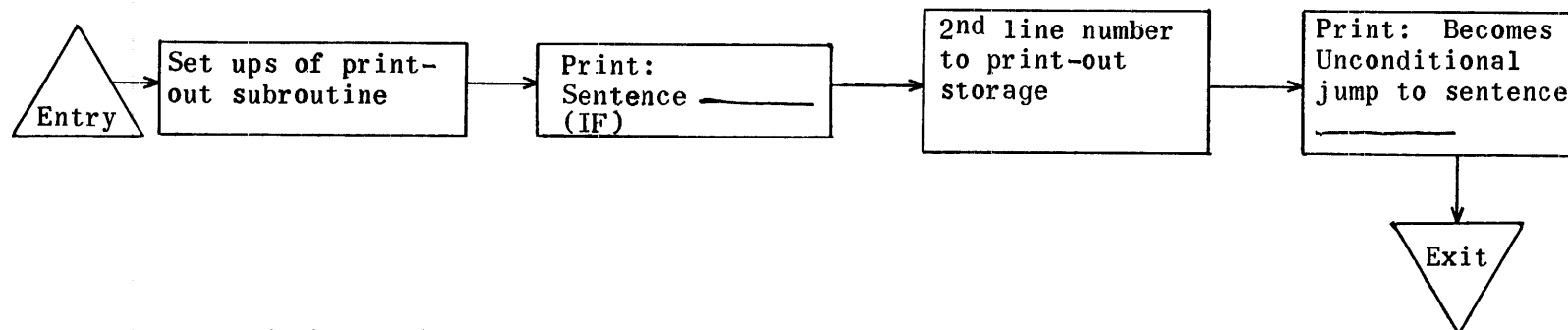
Sentence _____ (IF) Comparison Symbols Ambiguous



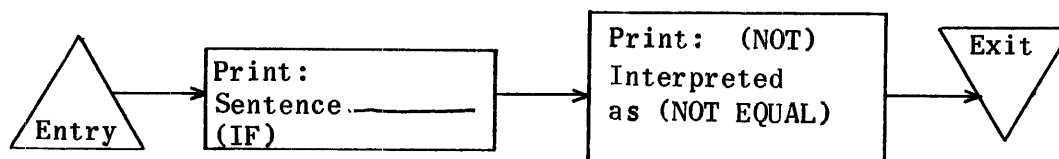
Sentence _____ (IF) Incorrectly Written _____



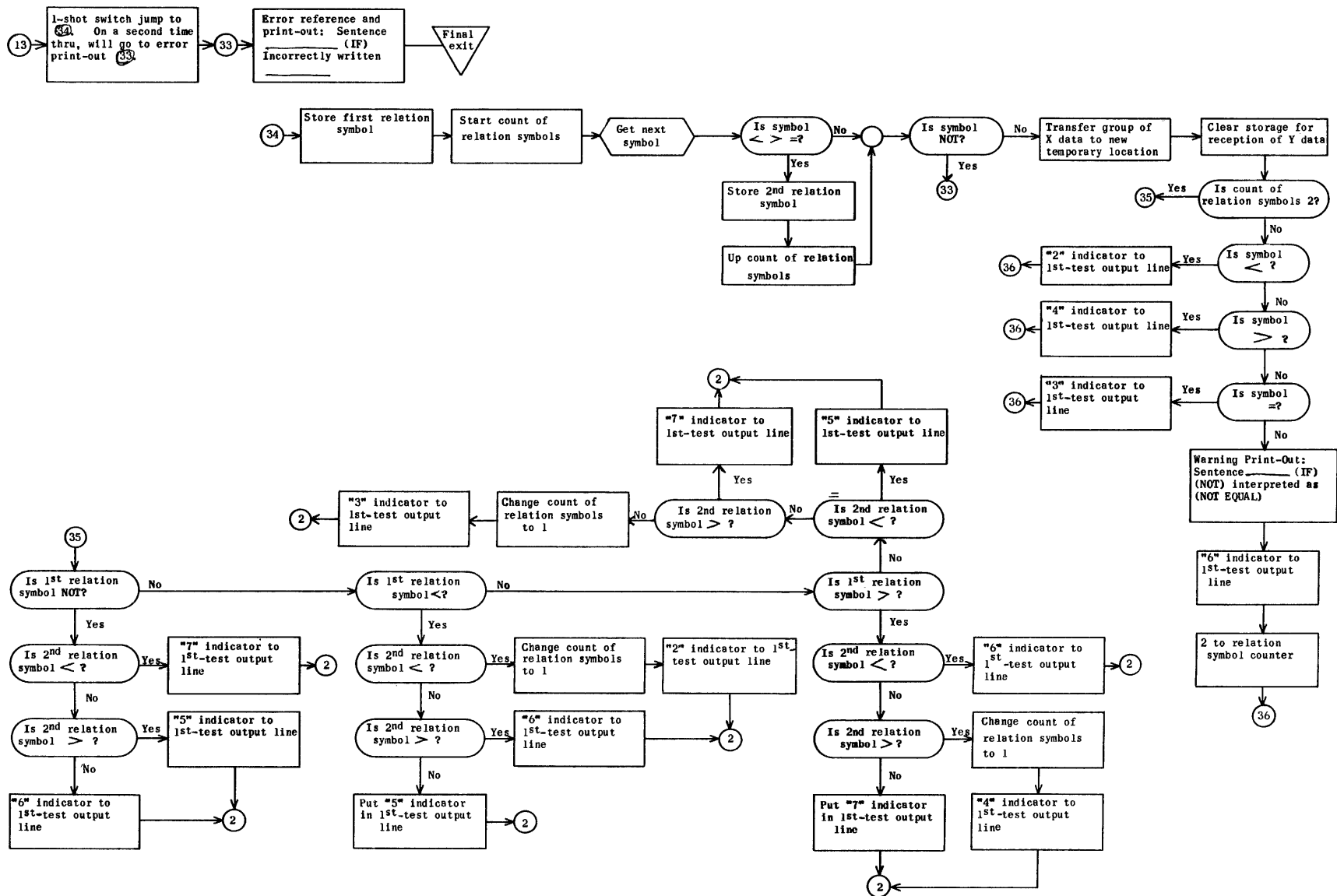
Sentence _____ (IF) Becomes Unconditional Jump to Sentence _____ (Warning only)

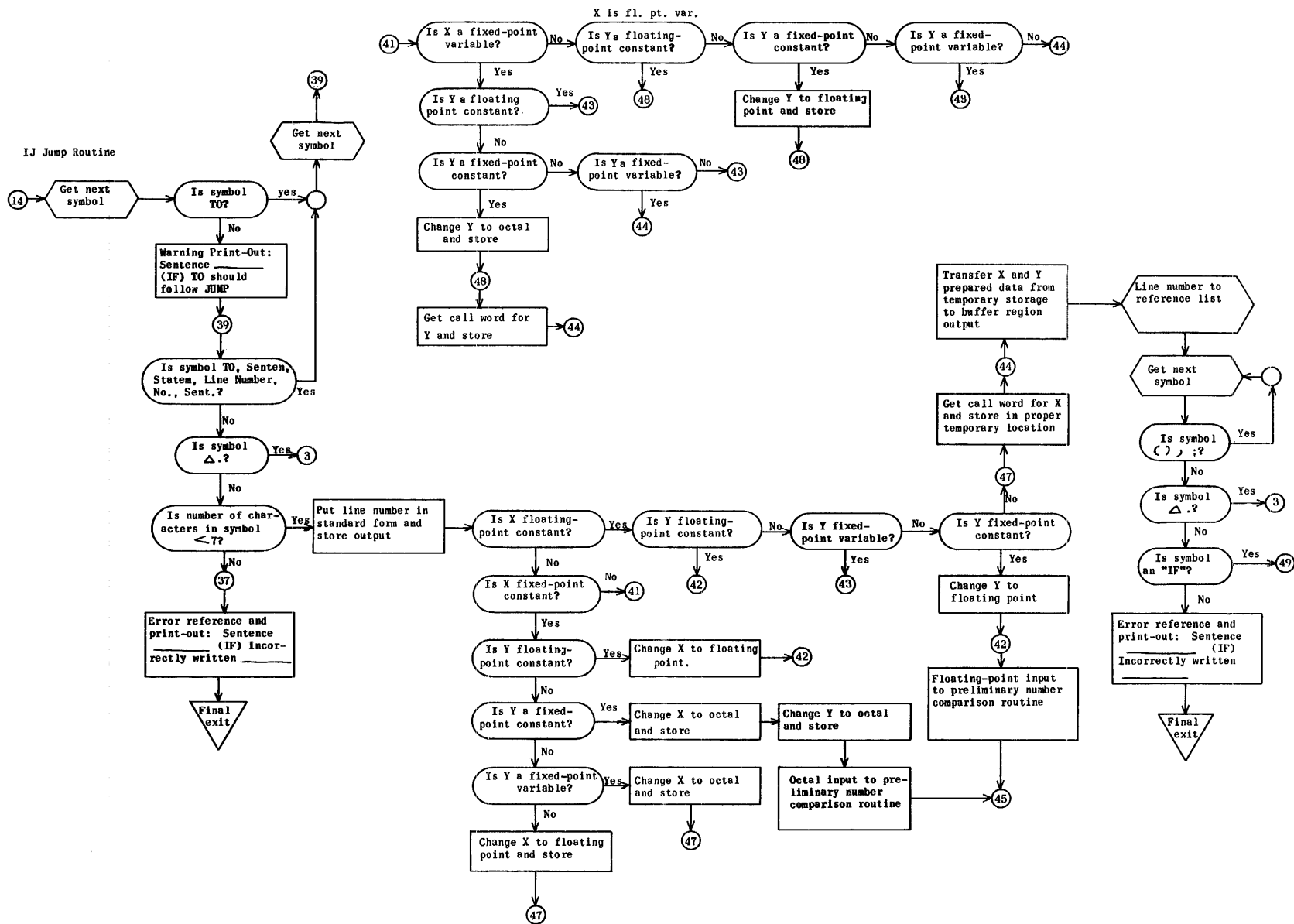


Sentence _____ (IF) (NOT) Interpreted as (NOT EQUAL)
(Warning only)

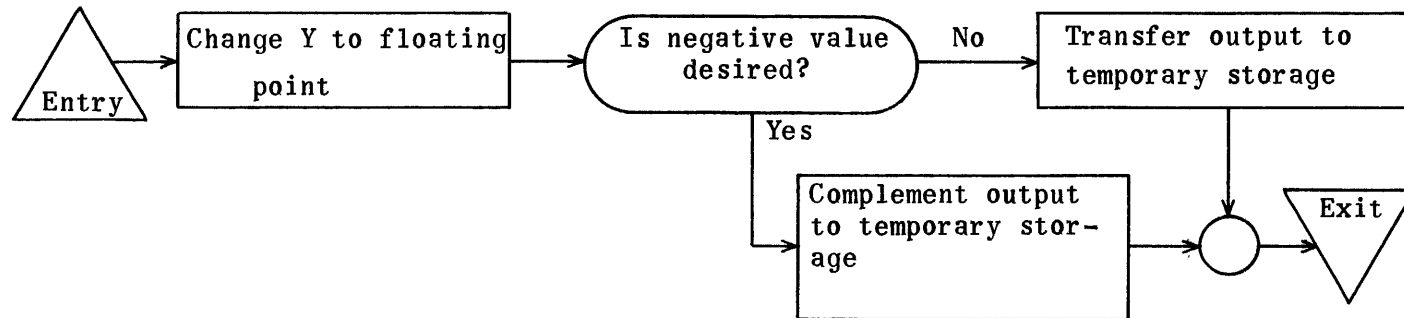


II Divider Routine

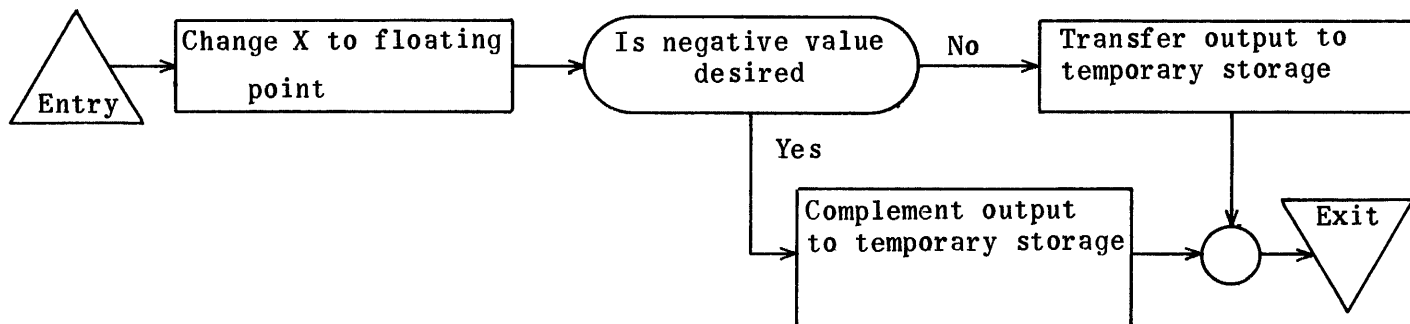




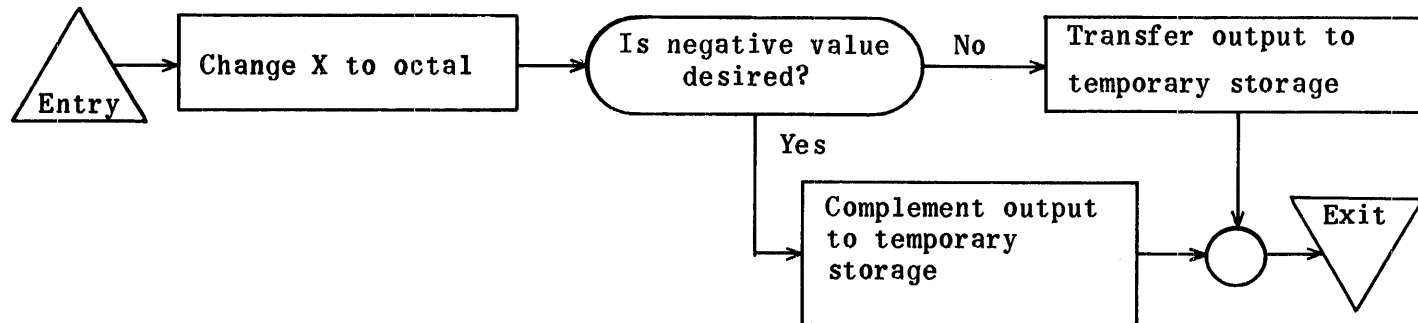
Subroutine to change Y to floating point and store



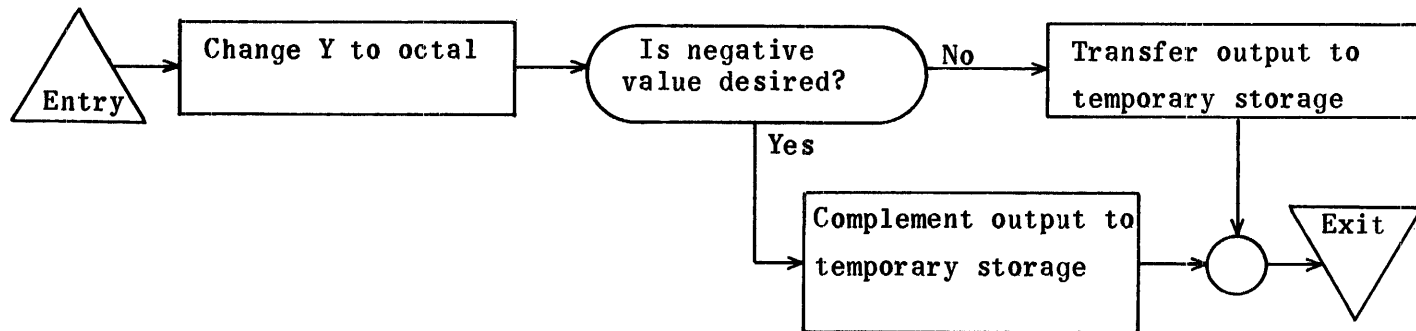
Subroutine to change X to floating point and store



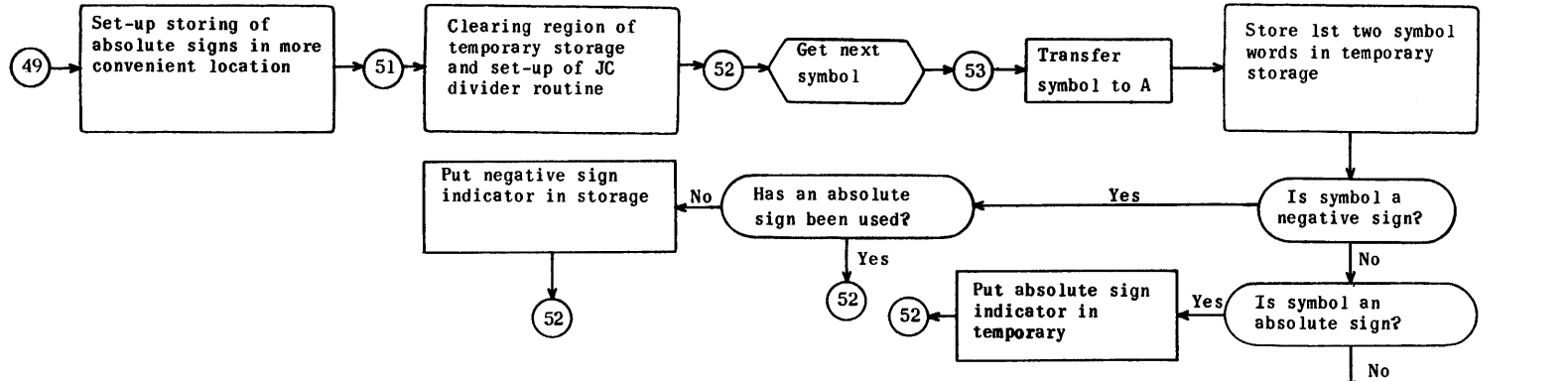
Subroutine to change X to octal and store



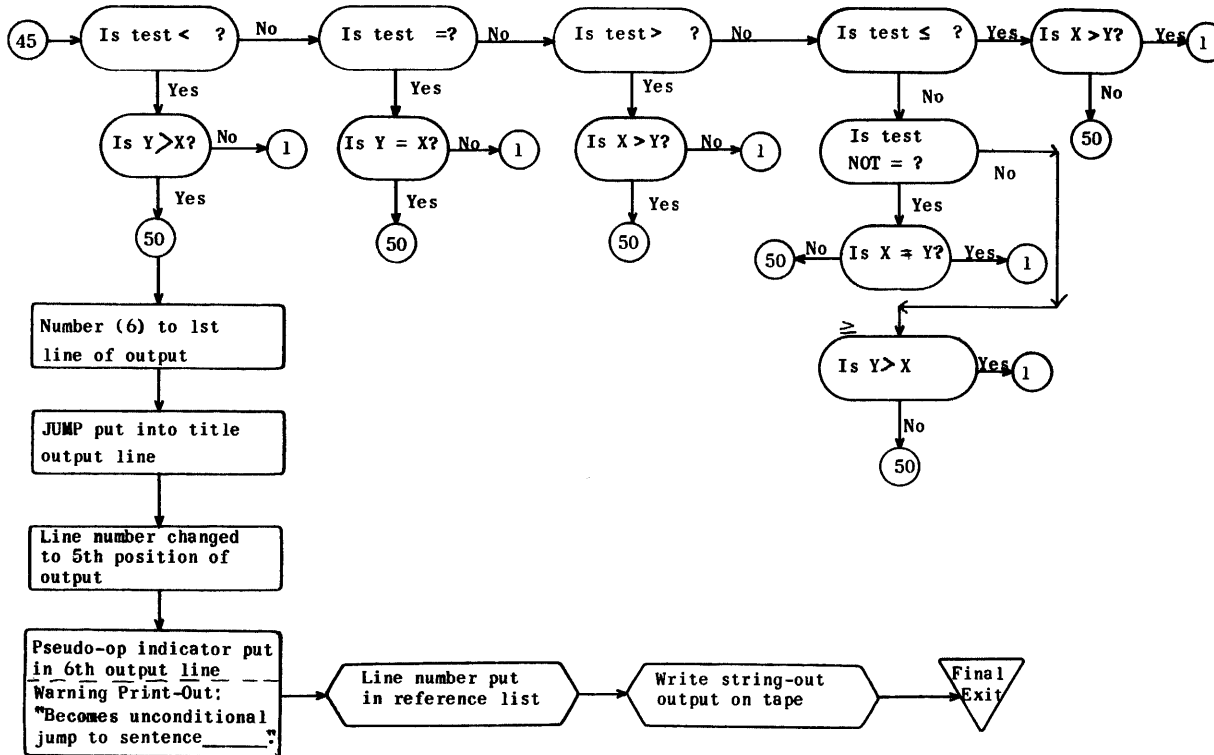
Subroutine to change Y to octal and store



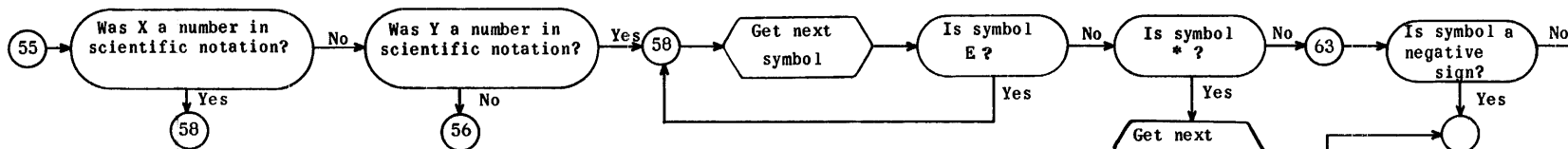
IU - Initial Control for Second or Third Clause



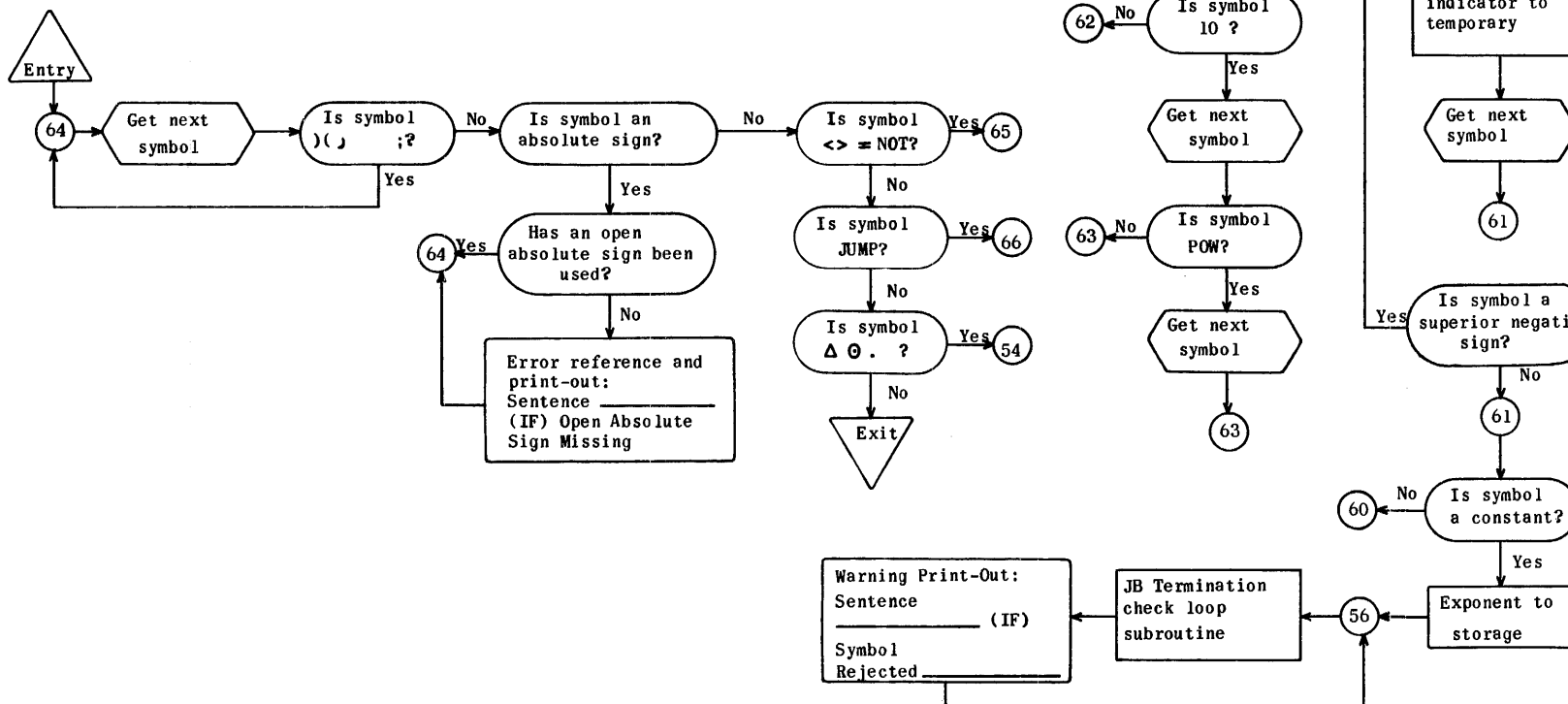
IT Preliminary Number Comparison Routine



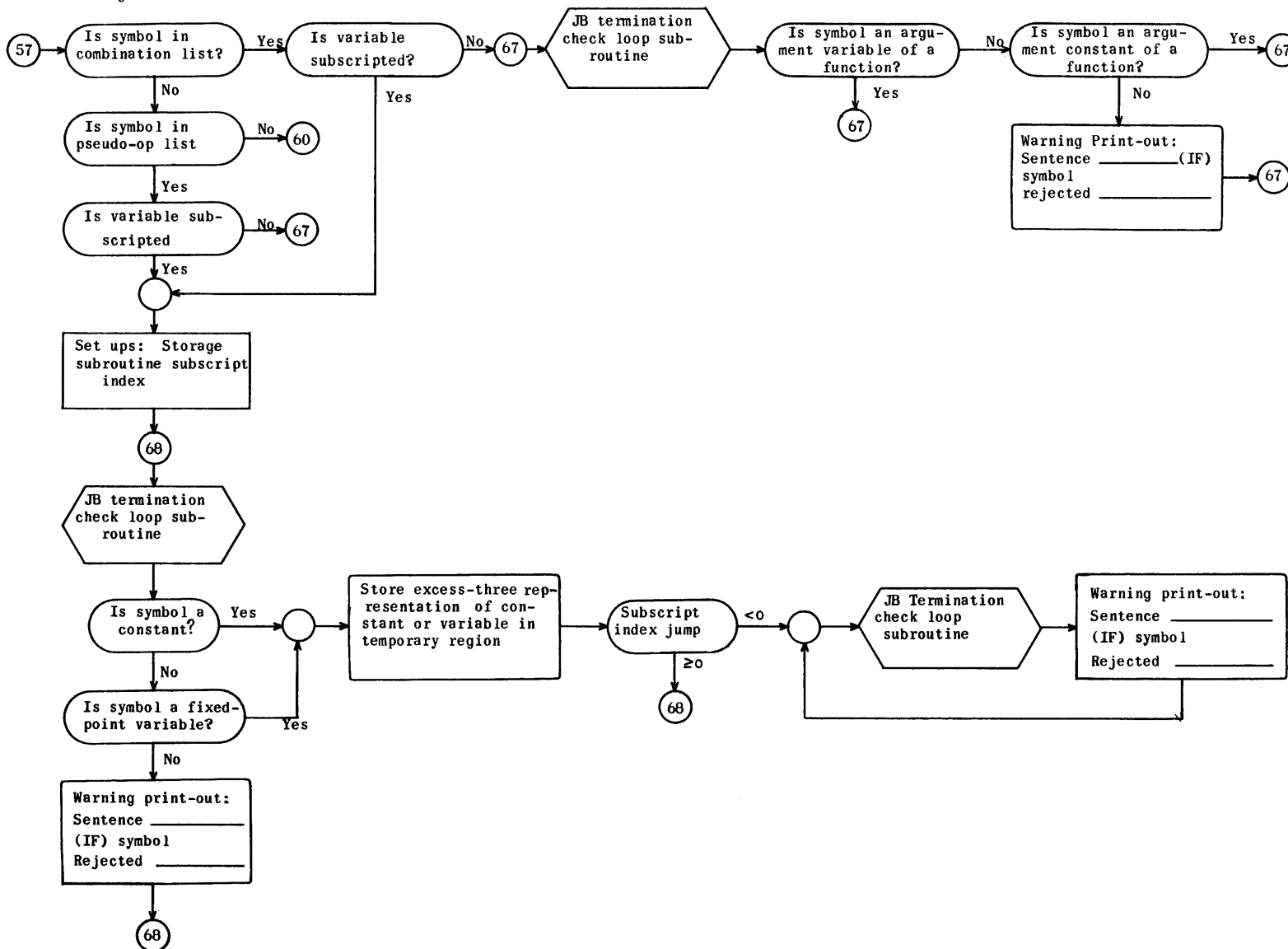
IY Constant and Fixed-Point Variable Routine for Second or Third Clause

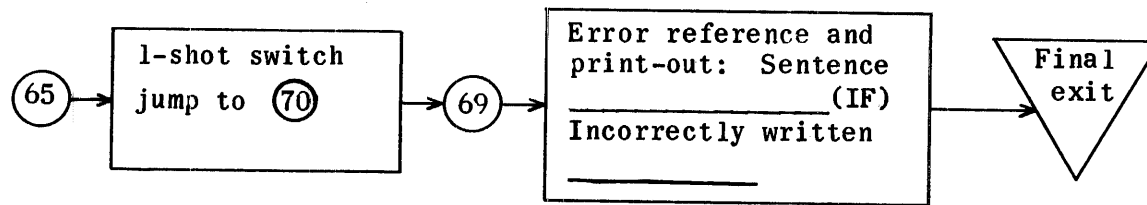


JB Termination Check Loop Subroutine

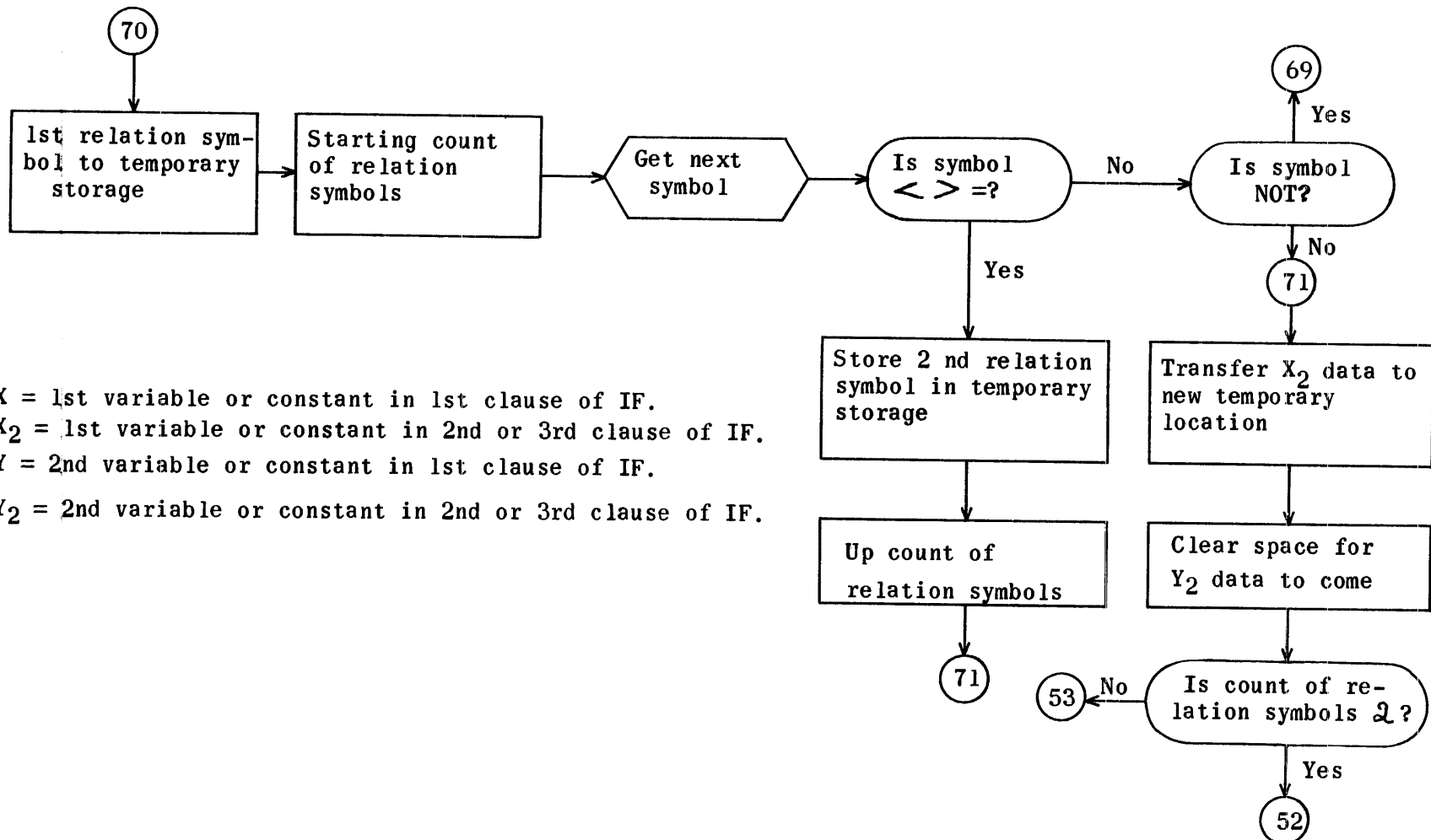


JB Floating-Point Variable Routine for Second or Third Clause

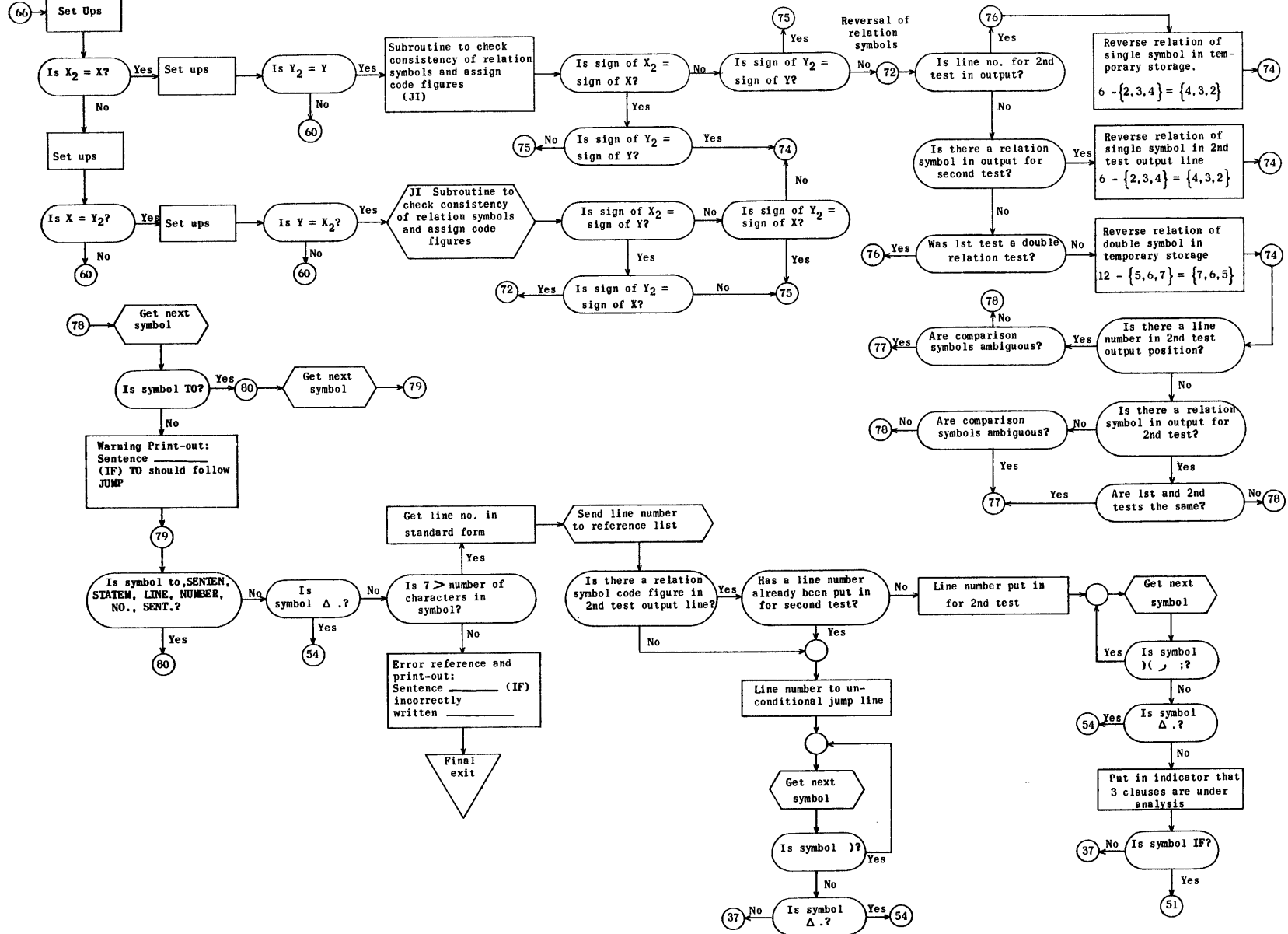




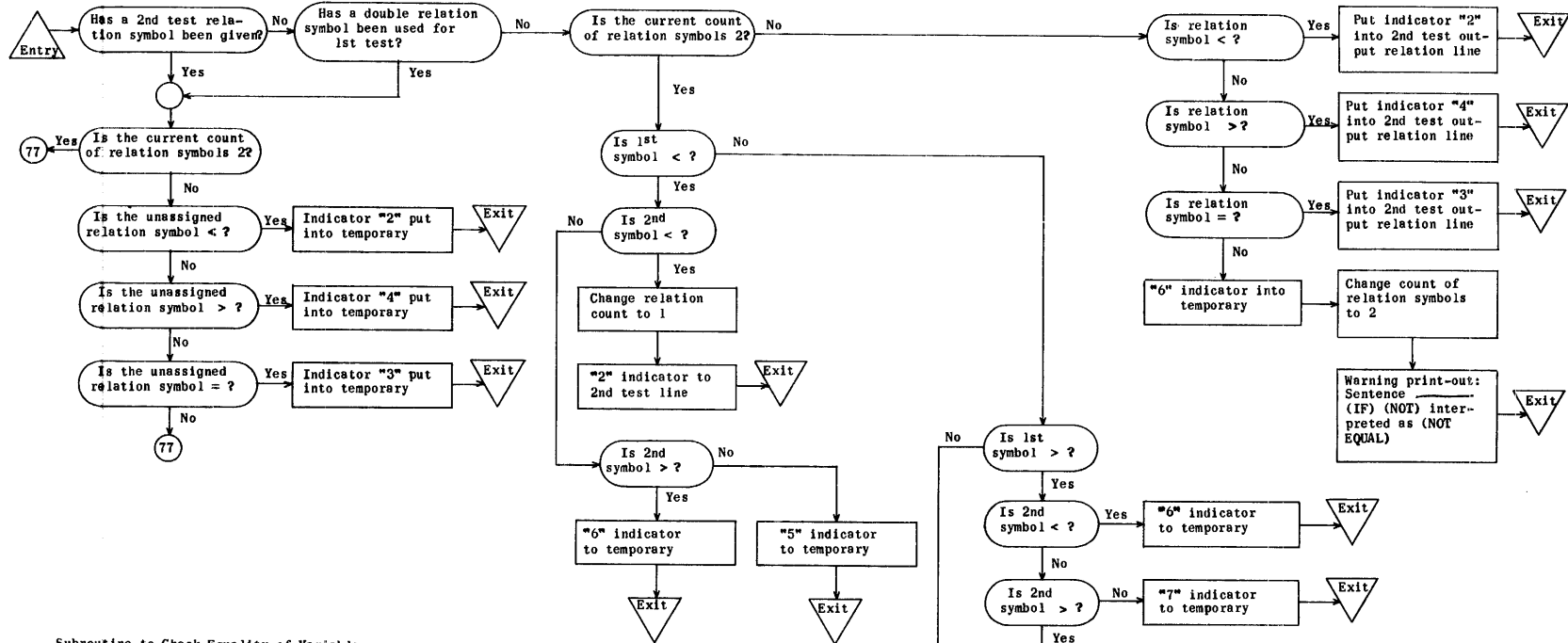
JC Relation Symbol Routines for Second or Third Clause



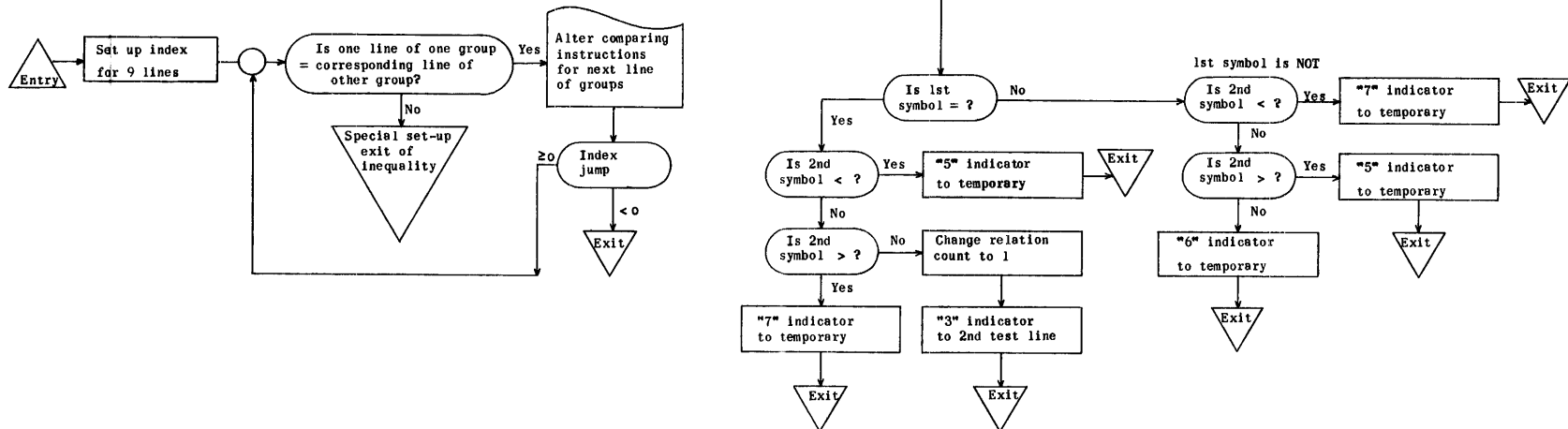
X = 1st variable or constant in 1st clause of IF.
 X₂ = 1st variable or constant in 2nd or 3rd clause of IF.
 Y = 2nd variable or constant in 1st clause of IF.
 Y₂ = 2nd variable or constant in 2nd or 3rd clause of IF.

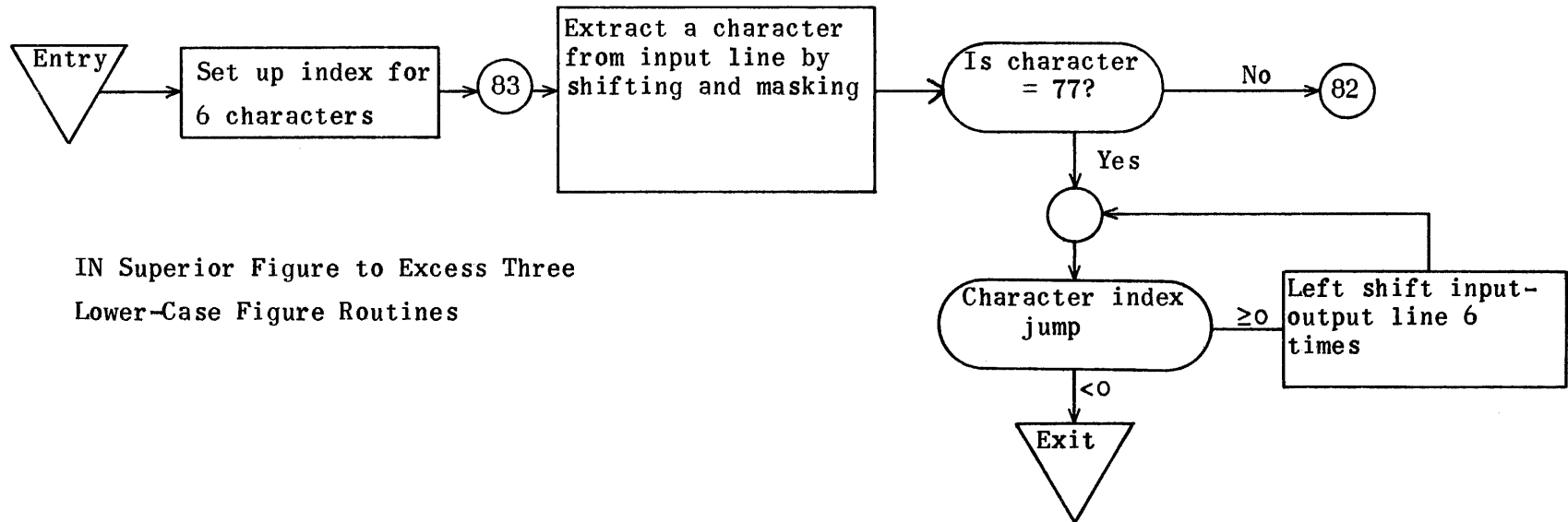


JI Subroutine to Check Consistency of Relation Symbols and Assign Code Figures

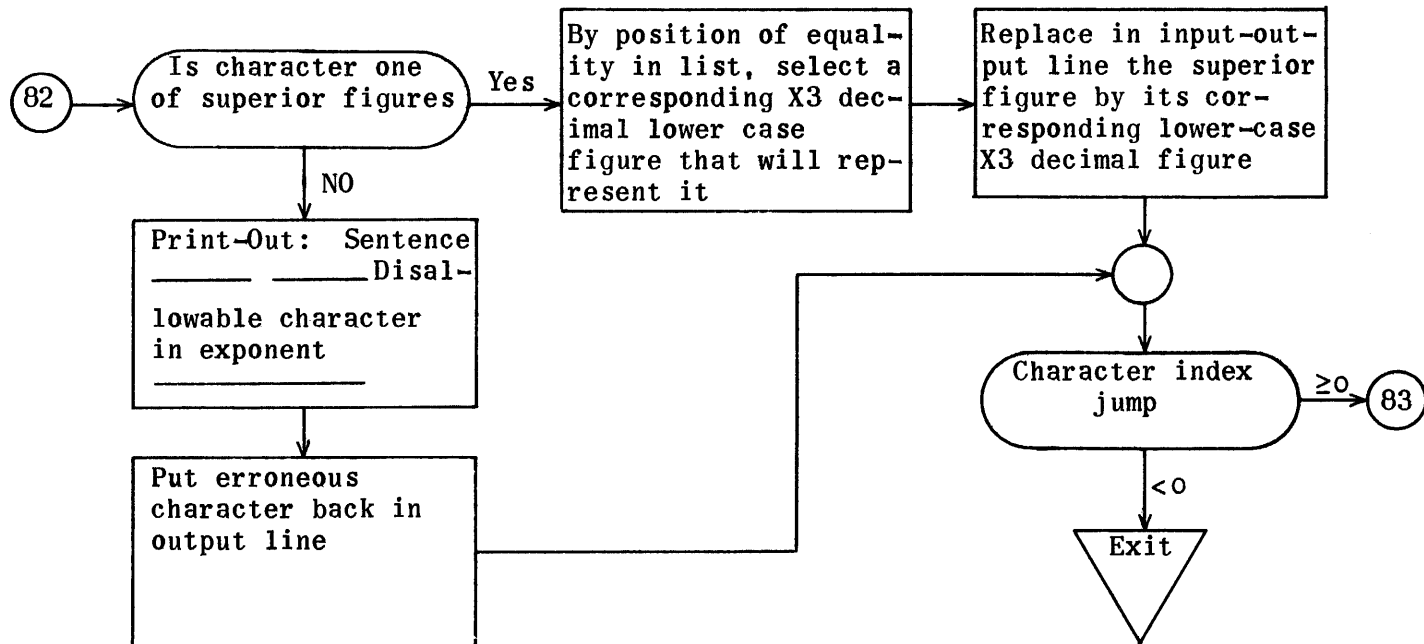


Subroutine to Check Equality of Variables or Constants in Separate Clauses

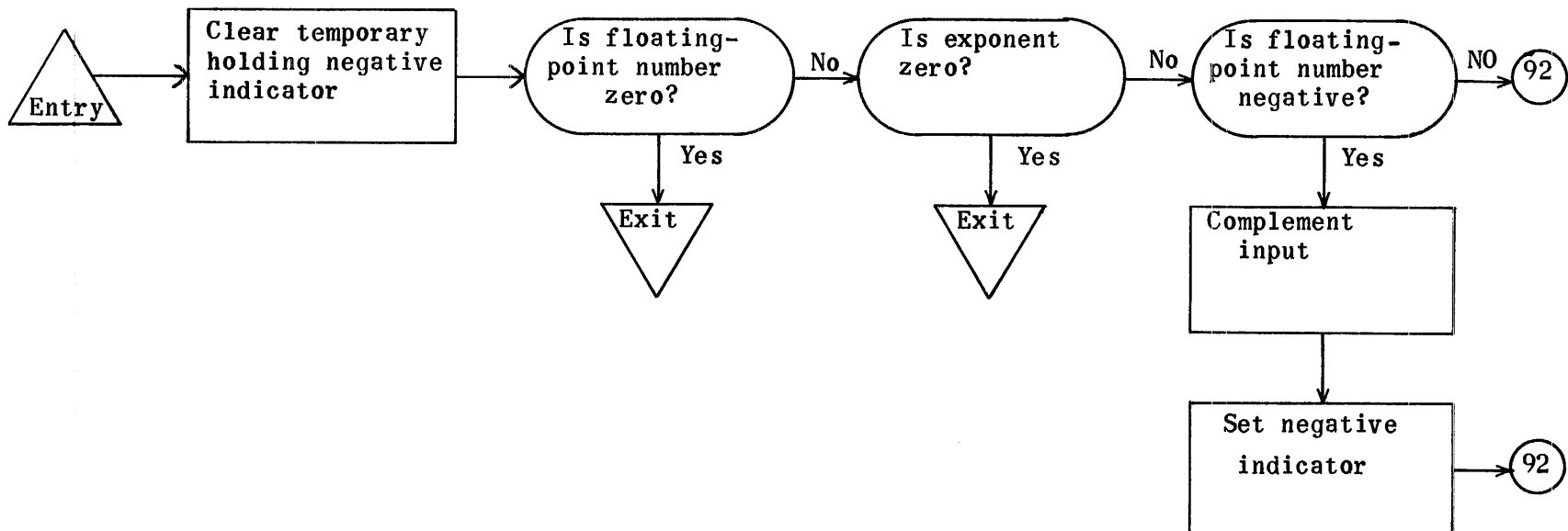


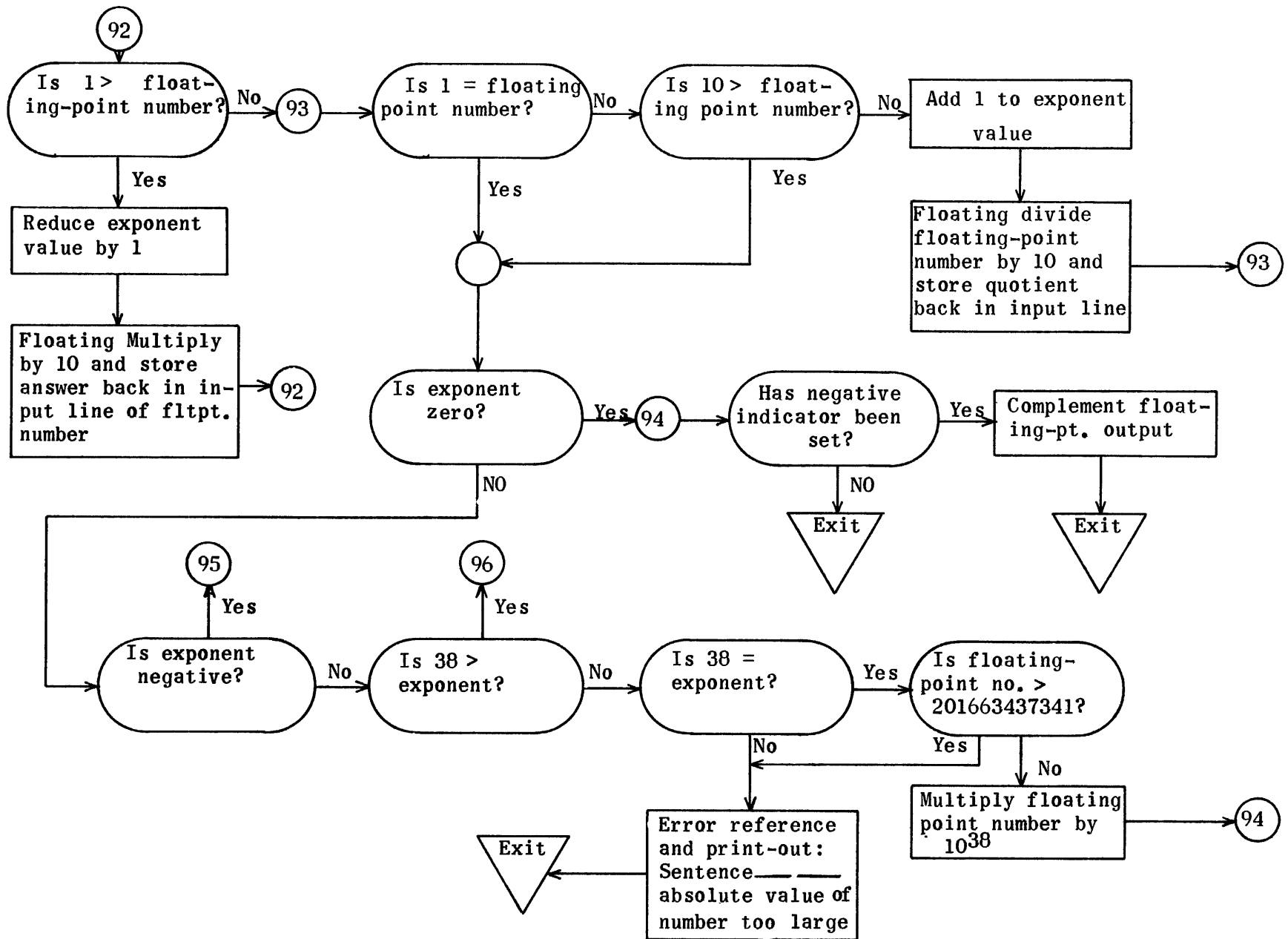


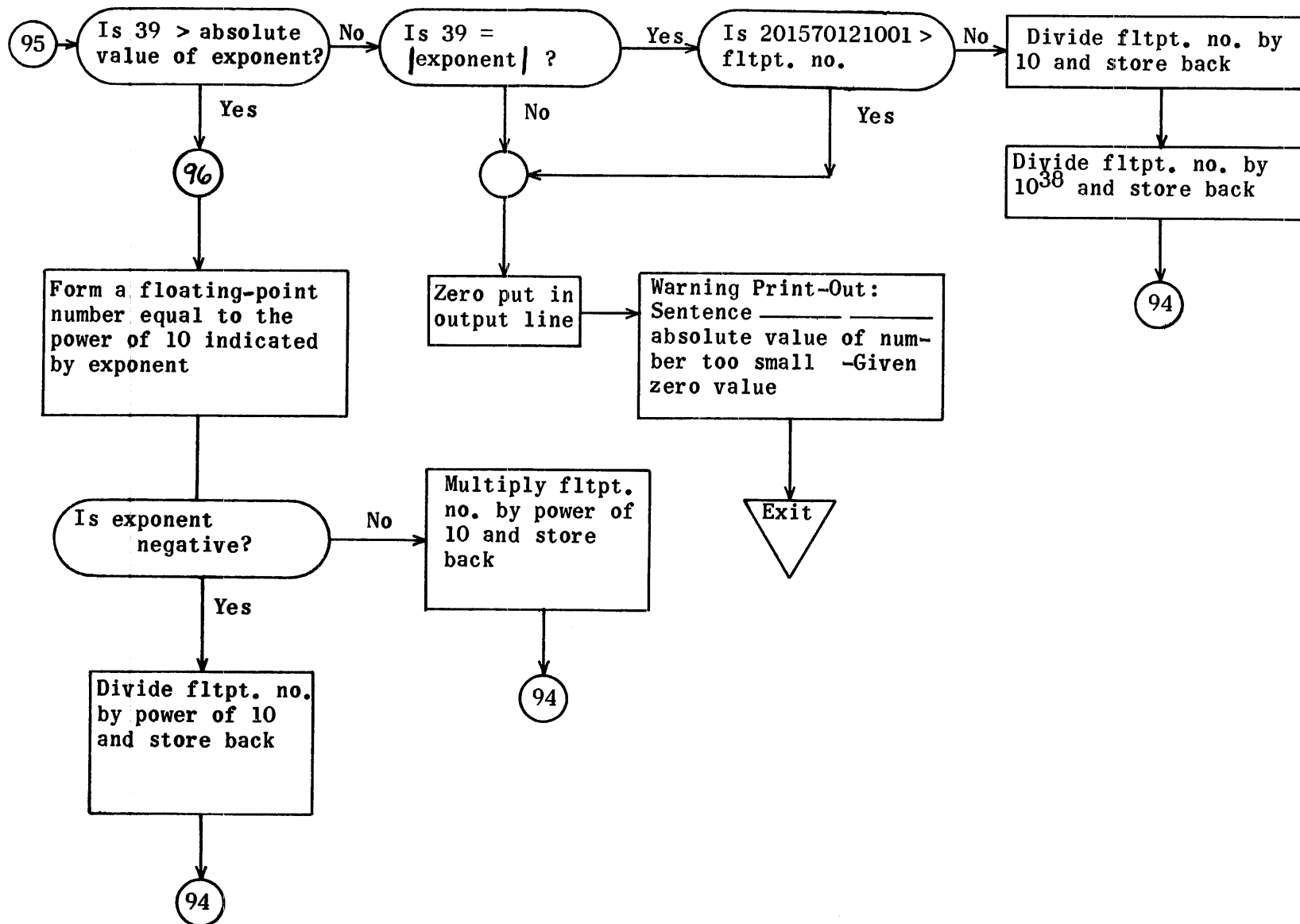
IN Superior Figure to Excess Three
Lower-Case Figure Routines



Flow Chart for IQ - Scientific Notation Evaluation Routine







IF STRING-OUT REGIONS

RE	SC4417	Two auxiliary print-out routines
RE	IE4430	Constant routine
RE	IF4565	Control routine
RE	IG4631	Fixed point variable routine
RE	IH4655	Floating point variable routine
RE	II4777	Relation symbol routine
RE	IJ5120	Jump routine
RE	IK5316	Termination space period routine
RE	IL5340	Floating point variable routine
RE	IM5503	Excess-Three Print-out storage
RE	IN5617	Superior to lower-case figure routine
RE	IO5647	Storage for "IN"
RE	IQ5676	Scientific notation check
RE	IR6025	Constants for IQ
RE	IS6037	Three temps. for IQ
RE	IT6042	Preliminary number comparison routine
RE	IU6105	2nd control routine
RE	IY6141	2nd constant and fixed-point variable routine
RE	JB6177	2nd floating-point variable routine
RE	JC6261	2nd relation symbol routine
RE	JF6305	2nd jump routine
RE	JI6470	Relation code assignment and consistency
RE	UF6614	Constant storage
RE	ID6702	Error print-out routines.

String-Out Subroutine regions are also needed
to assemble the IF tapes.

IF String-Out (Tape 1)

Auxiliary Print-Out Routines

	IA	SC		
0	TP	VN1	WB3	}
1	TP	UF2	WB5	
2	TP	WB10	WB6	
3	RJ	WA	WA12	
4	MJ	0	30000	}
5	TP	SC10	UP3	
6	RJ	UP2	UP	
7	MJ	0	30000	
10	40	SY2	1	
	CA	SC11		

Assists in warning that "if" sentence to "jump" change has been made.

Prints symbol occurring in sentence at point where analysis breaks down in print-out: Incorrectly written _____

Control Routine

	IA	IF		
0	MJ	0	CT	Exit
1	RP	10031	IF3	}
2	TP	UF7	VN4	
3	RP	10031	IF5	}
4	TP	UF7	VN71	
5	TP	UF30	VN	Number of lines (34) to of VN
6	TV	IF43	II	Setting up first line of relation routine.
7	TP	TS4	Q	}
10	QJ	IF11	IF12	
11	TP	UF27	VN33	Pseudo-op indicator to VN33
12	RJ	SY	SY1	Getting next symbol
13	TP	SY2	A	Symbol to A
14	TP	SY2	VN75	}
15	TP	SY3	VN76	
16	EJ	UF20	IF12	}
17	EJ	UF21	IF12	
20	EJ	UF2	IF12	
21	EJ	UF14	IK	Δ.
22	EJ	UF4	IF35	—
23	EJ	UF3	IF41	
24	EJ	UF17	IF12	(
25	TP	SY11	Q	}
26	QJ	IE	IF27	
27	TP	SY10	Q	}
30	QJ	IG	IF31	
31	TP	SY7	Q	}
32	QJ	IH	IF33	
33	RJ	ID16	ID13	
34	MJ	0	IF12	
35	TP	VN107	Q	}
36	QJ	IF12	IF37	
37	TP	UF27	VN106	
40	MJ	0	IF12	

Clearing lines to be used for output

Clearing temporary storage lines

Pseudo-op test

Pseudo-op indicator to VN33

Getting next symbol

Symbol to A

First 2 symbol output lines to temporary storage

Get next symbol.

Jump to space, period routine

Jump to negative sign subroutine

Jump to absolute sign subroutine

Get next symbol

Is symbol a constant? If so, jump to constant routine, ie

Is symbol a fixed-point variable. If so, jump to IG, fixed-point variable routine

Is symbol a floating-point variable? Jump to IH

Sentence -- (If) Symbol Rejected --

Puts negative sign indicator in proper storage if an absolute sign has not preceded it.

41	TP	UF27	VN107	Puts absolute sign indicator in proper storage space
42	MJ	O	IF12	
43	O	O	II3	
	CA	IF44		

Constant Routine

0	IA	IE			
	RJ	RB	RB1	Checks if figure has	{ 1 or fewer.'s 12 or less characters no letters
1	TP	II	A	}	Check if right of relation symbol
2	EJ	IE134	IE5		
3	TP	UF27	VN31		Constant indicator for X figure to output
4	MJ	O	IE6		
5	TP	UF27	VN32		Constant indicator for Y figure to output
6	TP	SY5	A	}	Is 7 > number of characters?
7	TJ	UF23	IE27		
10	TP	VN75	GG4	}	Conversion of figure to floating point
11	TP	VN76	GG5		
12	RJ	GG2	GG		
13	TP	GG3	VN71		Floating-point output to proper storage
14	TP	UF27	VN121		Floating-point indicator to storage
15	RJ	SY	SY1		Get next symbol
16	TP	SY2	A		
17	EJ	UF	IE46		Is symbol an e?
20	EJ	UF1	IE102		Is symbol a * ?
21	RJ	IE26	IE23		Jump to negative-value subroutine
22	MJ	O	IE117		
23	TP	VN106	Q	}	Is negative value desired?
24	QJ	IE25	IE26		
25	TN	VN71	VN71		Changing floating point figure to negative figure
26	MJ	O	30000		
27	TP	SY6	A		Are number of decimal points zero?
30	ZJ	IE10	IE31		
31	TP	VN75	VN72		Storing fixed-point X3 figure
32	RJ	SY	SY1		Getting next symbol
33	TP	SY2	A		
34	EJ	UF	IE45		Is e next symbol?
35	EJ	UF1	IE101		Is * next symbol?
36	MJ	O	IE117		Continuation beyond scientific-notation analysis
37	TP	VN72	GG4	}	Changing fixed-point X3 decimal figure to a floating-point number
40	TP	UF31	GG5		
41	RJ	GG2	GG		
42	TP	GG3	VN71		
43	TP	UF27	VN121		Floating-point indicator to storage
44	MJ	O	30000		
45	RJ	IE44	IE37		Changing fixed-point no. to a floating-point no.
46	RJ	SY	SY1		Getting next symbol
47	TP	SY2	A		

50	EJ	UF4	IE112	Is symbol a negative sign?
51	EJ	UF5	IE112	Is symbol a superior negative sign?
52	TP	SY11	Q	Is symbol a figure?
53	QJ	IE54	ID42	If not, print-out. Sentence -- Scientific notation incorrectly written
54	TP	SY2	VN100	Storing figure
55	TP	SY13	Q	Is symbol an exponent figure? if not, go directly to X3 decimal to octal conversion at IE63
56	QJ	IE57	IE63	
57	TP	SY2	IN2	Conversion of exponent to X3 decimal figure
60	RJ	IN	IN1	
61	TP	IN2	RS4	Output to RS 4 and jump to octal conversion routine
62	MJ	0	IE65	
63	RJ	RD	RD1	Check if symbol is a legitimate fixed-point figure
64	TP	SY2	RS4	Conversion of figure to octal
65	RJ	RS2	RS	
66	TP	VN77	Q	Check if sign of figure is negative
67	QJ	IE70	IE72	
70	TN	RS3	IQ3	Complementing figure and putting into scientific notation routine input
71	MJ	0	IE73	Figure to input of scientific notation routine
72	TP	RS3	IQ3	
73	TP	VN71	IQ2	
74	MJ	0	IE75	Dummy jump (replaces a discarded instruction)
75	RJ	IQ	IQ1	Check of scientific notation figures and converts to floating point
76	TP	IQ2	VN71	Output to storage
77	RJ	IE26	IE23	Changing output to negative if desired
100	MJ	0	IE115	Continuation beyond scientific notation
101	RJ	IE44	IE37	Changing fixed point to floating point no.
102	RJ	SY	SY1	Getting next symbol
103	TP	SY2	A	
104	EJ	UF22	IE106	Is next symbol a 10?
105	MJ	0	ID42	Print-out: Sentence -- Scientific notation incorrectly written
106	RJ	SY	SY1	Getting next symbol
107	TP	SY2	A	
110	EJ	UF25	IE46	Is symbol = POW?
111	MJ	0	IE50	If not, jump to other equality tests.
112	TP	UF27	VN77	Negative sign indicator storage for exponent
113	RJ	SY	SY1	Getting next symbol
114	MJ	0	IE52	
115	RJ	SY	SY1	Getting next symbol
116	TP	SY2	A	
117	EJ	UF3	IE130	Is symbol an absolute sign?
120	RP	20004	IE122	Is symbol) (, ; ?
121	EJ	UF16	IE115	
122	RP	20004	IE124	If symbol is < > = NOT jump to II relation symbol routine
123	EJ	UF10	II	

124	EJ	UF15	IJ	If symbol is JUMP, go to IJ jump routine
125	EJ	UF14	IK	Δ. exit
126	RJ	ID16	ID13	Print-out: Sentence -- Symbol Rejected --
127	MJ	0	IE115	
130	TP	VN107	Q	Has an open absolute sign been encountered?
131	QJ	IE115	IE132	If not, reference error routine and print-out: Sentence- -Open absolute sign missing.
132	RJ	ID56	ID52	
133	MJ	0	IE115	
134	RJ	II	III	
	CA	IE135		

Fixed-Point Variable Routine

Fixed-point	IA	IG		
variable	0	RJ	RH	Check if symbol is a valid variable
routine -	1	TP	UF27	Set fixed-point variable indicator
1st letter	2	TP	TS4	Is "if" sentence within Pseudo-Operation
i,j,k,l, or	3	QJ	IG4	
m	4	RJ	TS	Is symbol in Pseudo-Op list?
	5	MJ	0	No
	6	TP	TS3	Yes. Pseudo-Op call word to location
	7	MJ	0	
	10	RJ	TA	Is symbol in Combination List?
	11	MJ	0	No
	12	TP	TA4	Yes
	13	MJ	0	Exit for terminating characters before a divide or jump
	14	RJ	TK	Get next call word of 64, 65, 66 type
	15	AT	UF32	Adding in 64000 (fixed-point variable call-word base)
	16	TP	UF33	Building up file for addition of item to Combination List
	17	TP	SY2	
	20	TP	A	
	21	TP	UF7	Adding file to Combination List
	22	RJ	TE	
	23	MJ	0	Exit for terminating characters before a divide or jump
	CA	IG24		

Floating-Point Variable Routine

	IA	IH		
0	TV	IL136	IL40	Set-ups
1	TV	IL136	IL20	
2	TV	IL136	IL32	
3	TV	IL131	IL61	
4	TV	IL132	IL56	
5	TV	IL132	IL53	
6	TV	IL137	IL112	
7	TV	IL140	IL113	
10	RJ	RH	RH1	Check if valid variable.
11	TP	TS4	Q	Check if within pseudo-operation
12	QJ	IH13	IH61	

Within	13	RJ	TS	TS1	Is symbol in pseudo-op list?
Pseudo-	14	MJ	0	IH46	No
Op	15	TP	TS3	VN110	Yes. Call word to temporary location
	16	TP	UF34	A	Is call word > 76000?
	17	TJ	VN110	IH21	Exit subroutine
	20	MJ	0	IL64	Number of subscripts to (vn105) _v
Ps-Op	21	TP	UF37	Q	Subscript index set up
Sub. Var.	22	QT	VN110	VN105	630 RR to A _u
	23	LQ	VN105	36	630 (RR + no. subscripts), call word of modulus, to (vn105) _u
	24	RS	Q	UF6	Is subscript index zero?
	25	TP	Q	VN115	Call word of multiplier formed in (vn73) _u
	26	TP	UF40	Q	Multiplier call word to u of proper location
	27	QT	VN110	A	Handling subscript as constant & checking on variable.
	30	SA	UF41	17	Checking if fixed-point variable is in pseudo-op list and getting call word, if so
	31	TP	A	VN73	Checking if fixed-point variable is in combination List and, if not, adding it in. Call word to location.
	32	LQ	VN105	A17	
	33	AT	VN73	A	
	34	TU	A	VN105	
	35	TP	VN115	A	
	36	ZJ	IH37	IH41	
	37	RA	VN73	UF42	
	40	RJ	IL60	IL56	
	41	RJ	IL25	IL	
	42	RJ	IL33	IL26	
	43	RJ	IL52	IL34	
	44	IJ	VN115	IH36	
	45	MJ	0	IE115	
	46	TV	IL141	IL20	
	47	TV	IL141	IL40	
	50	TV	IL141	IL32	
	51	RJ	IH121	IH72	Check if symbol is in Comb. List and, if not, adding it in
	52	ZJ	IH53	IH54	Is index vn115 zero?
	53	RJ	IL130	IL112	Call word for multiplier to location
	54	RJ	IL25	IL	Constant or fixed-point variable subscript to location. Constant call wd to location.
	55	RJ	IL33	IL26	Search for call word of var. in pseudo-op list
	56	RJ	IL52	IL34	Search for call word of var. in Comb. List and adding it in if not found

Not within Pseudo-Op	57	IJ	VN115	IH52	Check if symbol is in Comb. List and, if not, adding it in Is subscript index zero? Call word for multiplier to location Handling subscripts as constant & checking on variable Checking if fixed-point variable is in Comb. List and, if not, adding it in. Call word to location.
	60	MJ	0	IE115	
	61	TV	IL142	IL20	
	62	TV	IL142	IL40	
	63	RJ	IH121	IH72	
	64	ZJ	IH65	IH66	
	65	RJ	IL130	IL112	
	66	RJ	IL25	IL	
	67	RJ	IL52	IL34	
	70	IJ	VN115	IH64	
	71	MJ	0	IE115	Check if symbol is in Combination List Not in Is in. Putting call word into loca- tion Check if symbol is a subscripted var- iable of 77 --- type
	72	RJ	TA	TA1	
	73	MJ	0	IH100	
	74	TP	TA4	VN110	
	75	TP	UF36	A	To get call word of 64, 65, 66 types Adding in 65000 (has to be fl. pt. var.) and storing
	76	TJ	VN110	IH110	
	77	MJ	0	IL64	
	100	RJ	TK	TK1	Adding file to Combination List
	101	AT	UF43	VN110	
	102	TP	UF33	TF	
	103	TP	SY2	TF1	
	104	TP	A	TF2	
	105	TP	UF7	TF3	
	106	RJ	TE	TE1	
	107	MJ	0	IE115	
	110	TP	UF44	Q	
	111	QT	TA5	VN105	
	112	ST	UF6	VN115	Number of subscripts to (vn105) _v Setting up index
	113	TP	UF45	Q	
	114	QT	TA5	A	Modulus to A _v
	115	LQ	A	25	
	116	RJ	GW	GW1	Call word of modulus to (vn105) _u
	117	TU	A	VN105	
	120	TP	VN115	A	
	121	MJ	0	30000	
		CA	IH122		

		IA	IL		
	0	RJ	SY	SY1	Getting next symbol
	1	TP	SY2	A	
	2	RP	20004	IL4 }	Is symbol) (, ; ?
	3	EJ	UF16	IL }	
	4	TP	SY11	Q }	Is symbol a constant?
	5	QJ	IL6	IL21 }	
	6	TP	SY13	Q	If symbol is a superior figure, error
	7	QJ	IJ12	IL10	ref. and print-out occurs: Sentence
					-- (If) Incorrectly Written --
Subscript	10	RJ	RD	RD1	Check for validity of fixed-point
a constant	11	RJ	IL63	IL61	constant
	12	TP	SY2	RS4 }	Put constant in temporary location
	13	RJ	RS2	RS }	Converting constant to octal
	14	TP	RS3	A }	
	15	RJ	GW	GW1 }	Getting call word for constant
	16	TU	UF35	IL53 }	
	17	RJ	IL55	IL53 }	Constant call word to temp. location
	20	MJ	0	[IH44]	Jump index check for subscripts of
					pseudo ops. IH57 or IH70 in v
					occasionally
	21	TP	SY10	Q }	Test if symbol is a fixed-point var-
					iable. If not, error print: Sentence
	22	QJ	IL23	IJ12 }	incorrectly written --
Subscript	23	RJ	RH	RH1	Check for validity of variable
a fixed-	24	RJ	IL63	IL61	Variable to temp. location
point	25	MJ	0	30000	
variable	26	RJ	TS	TS1	Check if variable is in pseudo-op
					list
	27	MJ	0	IL33	Not in list
Getting	30	TU	IL135	IL53	Is in list. Putting call word in
call word	31	RJ	IL55	IL53	location
for var.	32	MJ	0	[IH44]	Or IH 57 in v
in pseudo-	33	MJ	0	30000	
op list	34	RJ	TA	TA1	Check if fixed-point variable is in
					Combination List
	35	MJ	0	IL41	Not in list

Getting	36	TU	IL134	IL53	Is in List
call word	37	RJ	IL55	IL53	Call word to location
for var.	40	MJ	0	[IH44]	Jump to subscript index for pseudo-op.
in Comb.					(IH57 or IH70 in v occasionally)
List	41	RJ	TK	TK1	To get call word (next) of 64, 65, 66
Putting	42	AT	UF32	TF2	Adding in 64000
call word	43	TP	UF33	TF	
for var.	44	TP	SY2	TF1	
in Comb.	45	TP	UF7	TF3	
List and	46	RJ	TE	TE1	Adding variable and call word to
temp.	47	TP	TF2	Q	Combination List
location	50	TU	UF35	IL53	
	51	RJ	IL55	IL53	Putting call word in location
	52	MJ	0	30000	
	53	TV	[Q]	[VN111]	
	54	RA	IL53	UF6	Call word to location
	55	MJ	0	30000	
	56	TU	VN73	[VN111]	
	57	RA	IL56	UF6	Call word for multiplier to location
	60	MJ	0	30000	
	61	TP	SY2	[VN101]	
	62	RA	IL61	UF6	Storage of X3 for subscripts
	63	MJ	0	30000	
	64	TP	VN110	A	
Exit	65	TJ	UF46	IL67	Is it a dummy function ?
subroutine	66	TJ	UF47	IE115	Is it not a function?
	67	RJ	SY	SY1	Get next symbol
	70	TP	SY2	A	
	71	RP	20004	IL72	
	72	EJ	UF16	IL67	() , ; checks
	73	EJ	UF3	IL106	sign check
	74	RP	20004	IL76	
	75	EF	UF10	II	< > = NOT
	76	EJ	UF15	IJ	JUMP
	77	EJ	UF14	IK	△ .
	100	TP	SY7	Q	Test if symbol is a variable used as
	101	QJ	IL67	IL102	argument of single-valued function
	102	TP	SY11	Q	Test if symbol is a constant used as
	103	QJ	IL67	IL104	argument of single-valued function
	104	RJ	ID16	ID13	Error Print: Symbol Rejected --
	105	MJ	0	IL67	
Check for	106	TP	VN107	Q	
open ab-	107	QJ	IL67	IL110	Test if there has been an open abso-
olute	110	RJ	ID56	ID52	lute sign
sign	111	MJ	0	IL67	Error print: Open absolute sign
	112	RJ	IL112	[IL117]	missing
	113	RJ	IL113	[IL122]	
	114	TP	UF44	Q	1-shot switch
	115	QT	TA7	A	1-shot switch
	116	MJ	0	IL125	3rd Multiplier → A

117	TP	UF44	Q	}	1st Multiplier → A
120	QT	TA6	A		
121	MJ	0	IL125		
122	TP	UF45	Q	}	2nd Multiplier → A
123	QT	TA6	A		
124	LQ	A	25		
125	RJ	GW	GW1	}	Call word for multiplier to (VN73) _u
126	TU	A	VN73		
127	RJ	IL60	IL56		Call word to proper temporary location
130	MJ	0	30000		
131	0	0	VN101		
132	0	0	VN111		
133	0	VN73	0		
134	0	TA4	0		
135	0	TS3	0		Pseudo-op list routine location
136	0	0	IH44		
137	0	0	IL117		
140	0	0	IL122		
141	0	0	IH57		
142	0	0	IH70		
	CA	IL143			

Relation Symbol Routine

	IA	II		
0	RJ	II	II3	
1	RJ	ID40	ID34	Error Print: Sentence (IF) Incorrectly written --
2	MJ	0	IF	
3	TP	A	VN117	Store 1st relation symbol
4	TP	UF6	VN116	Start count of relation symbols
5	RJ	SY	SY1	Get next symbol
6	TP	SY2	A	
7	RP	20003	II13	} < > = tests
10	EJ	UF10	II11	
11	TP	A	VN120	Store 2nd relation symbol
12	RA	VN116	UF6	Up count of relation symbols
13	EJ	UF13	II1	Not?
14	RP	30031	II16	} Transfer block of X data to new temporary location
15	TP	VN71	VN40	
16	RP	10031	II20	} Clearing storage for reception of Y data
17	TP	UF7	VN71	
20	TP	VN65	A	} Is count of relation symbols equal to 2?
21	EJ	UF26	II41	

One	22	TP	VN66	A	
relation	23	EJ	UF10	II27	
symbol	24	EJ	UF11	II31	
	25	EJ	UF12	II33	
	26	MJ	0	II35	
	27	TP	UF26	VN4	
	30	MJ	0	IF13	
	31	TP	UF61	VN4	
	32	MJ	0	IF13	
	33	TP	UF60	VN4	
	34	MJ	0	IF13	
	35	RJ	ID121	ID116	
	36	TP	UF63	VN4	
	37	TP	UF26	VN65	
	40	MJ	0	IF13	
When there	41	TP	VN66	A	
are two	42	EJ	UF13	II46	
relation	43	EJ	UF10	II60	
symbols	44	EJ	UF11	II73	
	45	MJ	0	II106	
NOT	46	TP	VN67	A	
	47	EJ	UF10	II52	
	50	EJ	UF11	II54	
	51	MJ	0	II56	
	52	TP	UF64	VN4	
	53	MJ	0	IF12	
	54	TP	UF62	VN4	
	55	MJ	0	IF12	
	56	TP	UF63	VN4	
	57	MJ	0	IF12	
<	60	TP	VN67	A	
	61	EJ	UF10	II64	
	62	EJ	UF11	II67	
	63	MJ	0	II71	
	64	TP	UF6	VN65	
	65	TP	UF26	VN4	
	66	MJ	0	IF12	
	67	TP	UF63	VN4	
	70	MJ	0	IF12	
	71	TP	UF62	VN4	
	72	MJ	0	IF12	
>	73	TP	VN67	A	
	74	EJ	UF10	II77	
	75	EJ	UF11	II101	

< } Symbol tests
 > }
 = }
 NOT }
 < "2" indicator to output
 Return to if control routine to
 examine symbol not used .
 > "4" indicator to output

 = "3" indicator to output

 Sentence -- NOT interpreted as NOT
 EQUAL
 NOT = "6" indicator to output
 Putting 2 in relation-symbol counter

 NOT?
 < ?
 > ?
 =

 < ?
 > ?
 =
 NOT<indicator "7" to output line for
 1st test
 Return to if control routine to get
 next symbol
 NOT > indicator "5" to output line
 for 1st test

 NOT = indicator "6" to output line
 for 1st test

 <<?
 <>?
 < =
 <<

 <>or NOT = indicator "6" to output

 <=indicator "5" to output

 ><?
 >>?

	76	MJ	0	II104	> =
	77	TP	UF63	VN4	NOT = indicator "6" to output
	100	MJ	0	IF12	
	101	TP	UF6	VN65	>> Count of relation symbols reduced to 1
	102	TP	UF61	VN4	Indicator "4" to output
	103	MJ	0	IF12	
	104	TP	UF64	VN4	> = indicator "7" to output
	105	MJ	0	IF12	
=	106	TP	VN67	A	
	107	EJ	UF10	II112	= < ?
	110	EJ	UF11	II114	= > ?
	111	MJ	0	II116	= =
	112	TP	UF62	VN4	= < indicator "5" to output
	113	MJ	0	IF12	
	114	TP	UF64	VN4	= > indicator "7" to output
	115	MJ	0	IF12	
	116	TP	UF6	VN65	= = Count of relation symbols reduced to 1
	117	TP	UF60	VN4	Indicator "3" to output
	120	MJ	0	IF12	
		CA	II121		

Relation Symbol Code Assignment and Consistency Routine

		IA	JI		
	0	MJ	0	30000	
	1	TP	UF7	JI123	Clearing temporary
Relation	2	TP	VN6	A	} Has a 2nd-test relation symbol been supplied?
symbols	3	ZJ	J14	JI21	
in vn4 and	4	TP	VN142	A	} Is the count of relation symbols 2?
vn6	5	EJ	UF26	JI12	
	6	TP	VN143	A	
1 in	7	EJ	UF10	JI13	< ?
vn142	10	EJ	UF11	JI15	> ?
count	11	EJ	UF12	JI17	= ?
	12	MJ	0	ID103	Error Print: Comparison symbols ambiguous
	13	TP	UF26	JI123	< Indicator "2" to temporary
	14	MJ	0	JI	
	15	TP	UF61	JI123	> Indicator "4" to temporary
	16	MJ	0	JI	
	17	TP	UF60	JI123	= Indicator "3" to temporary
	20	MJ	0	JI	
	21	TP	VN65	A	} If 2 relation symbols in 1st part of if or in vn4, go to above
	22	EJ	UF26	JI4	
	23	TP	VN142	A	} Is the current count of relation symbols 2?
	24	EJ	UF26	JI44	
1 in vn4					
2 coming					
up					

	25	TP	VN143	A	
1 in vn4	26	EJ	UF10	JI32	< ?
1 coming	27	EJ	UF11	JI34	> ?
up to go	30	EJ	UF12	JI36	= ?
in vn6	31	MJ	0	JI40	NOT
	32	TP	UF26	VN6	< Indicator "2" to 2nd test output line
	33	MJ	0	JI	
	34	TP	UF61	VN6	> Indicator "4" to 2nd test output line
	35	MJ	0	JI	
	36	TP	UF60	VN6	= Indicator "3" to 2nd test output line
	37	MJ	0	JI	
	40	TP	UF63	JI123	"NOT" occurring alone was miscounted as one. Recounted as 2
	41	TP	UF26	VN142	
	42	RJ	ID121	ID116	Print: NOT interpreted as NOT EQUAL
	43	MJ	0	JI	
2 relation	44	TP	VN143	A	
symbols.	45	EJ	UF10	JI62	< ?
Nothing	46	EJ	UF11	JI110	> ?
in vn6,	47	EJ	UF12	JI75	= ?
1 in vn4	50	TP	VN144	A	NOT
	51	EJ	UF10	JI54	NOT < ?
	52	EJ	UF11	JI56	NOT > ?
	53	MJ	0	JI60	NOT =
	54	TP	UF64	JI123	NOT < Indicator "7" to temporary
	55	MJ	0	JI	
	56	TP	UF62	JI123	NOT > Indicator "5" to temporary
	57	MJ	0	JI	
	60	TP	UF63	JI123	NOT = Indicator "6" to temporary
	61	MJ	0	JI	
	62	TP	VN144	A	
	63	EJ	UF10	JI66	<< ?
	64	EJ	UF11	JI71	<> ?
	65	MJ	0	JI73	< =
1 in vn 4	66	TP	UF6	VN142	<< Correction of 2 count to 1 count
1 in vn 6	67	TP	UF26	VN6	Indicator "2" in 2nd test output line
	70	MJ	0	JI	
	71	TP	UF63	JI123	<> Indicator "6" to temporary
	72	MJ	0	JI	
	73	TP	UF62	JI123	< = Indicator "5" to temporary
	74	MJ	0	JI	
	75	TP	VN144	A	
	76	EJ	UF10	JI101	= < ?
	77	EJ	UF11	JI103	= > ?
	100	MJ	0	JI105	= =
	101	TP	UF62	JI123	= < Indicator "5" to temporary
	102	MJ	0	JI	
	103	TP	UF64	JI123	= > Indicator "7" to temporary

	104	MJ	0	J1		
1 in vn4	105	TP	UF6	VN142	=	=
1 in vn6	106	TP	UF60	VN6		} Correction of 2 count to 1 count. Indicator 3 to 2nd test output line
	107	MJ	0	J1		
	110	TP	VN144	A		
	111	EJ	UF10	J1114	>< ?	
	112	EJ	UF11	J1116	>> ?	
	113	MJ	0	J1121	> =	
	114	TP	UF63	J1123	<>	Indicator 6 to temporary
	115	MJ	0	J1		
1 in vn4	116	TP	UF6	VN142	>>	} Corr. of 2 count to 1 Indicator "4" to 2nd test output line
1 in vn6	117	TP	UF61	VN6		
	120	MJ	0	J1		
	121	TP	UF64	J1123	>=	Indicator "7" to temporary
	122	MJ	0	J1		
	123	0	0	0		
		CA	J1124			

Constant Storage

	IA	UF		
0	30	77777	77777	e
1	56	77777	77777	*
2	34	31777	77777	if
3	42	77777	77777	
4	02	77777	77777	-
5	00	77777	77777	✓
6	0	0	1	
7	0	0	0	
10	37	77777	77777	<
11	16	77777	77777	>
12	76	77777	77777	=
13	50	51667	77777	NOT
14	01	22777	77777	Δ .
15	44	67475	27777	JUMP
16	43	77777	77777	)
17	17	77777	77777	(
20	21	77777	77777	,
21	23	77777	77777	;
22	04	03777	77777	10
23	0	0	7	
24	0	0	14	12
25	52	51717	77777	POW
26	0	0	2	<
27	40	0	0	
30	0	0	34	
31	77	77777	77777	
32	0	0	64000	
33	0	3	3	
34	0	0	76000	

35	0	31000	0	
36	0	0	76777	
37	0	0	00700	
40	0	0	00077	
41	0	0	63000	
42	0	1	0	
43	0	0	65000	
44	0	0	77777	
45	0	77777	0	
46	0	0	62000	
47	0	0	66000	
50	0	0	11	9
51	66	51777	77777	TO
52	65	30506	63050	SENTEN
53	65	66246	63047	STATEM
54	46	34503	07777	LINE
55	50	67472	53054	NUMBER
56	50	51227	77777	NO.
57	65	30506	62277	SENT.
60	0	0	3	=
61	0	0	4	>
62	0	0	5	≤
63	0	0	6	NOT =
64	0	0	7	≥
65	0	0	10	
	CA	UF66		

If String-Out (Tape 2)

Error Print-Out Routines

	IA	ID		
0	TP	SY2	ID22	} Storing rejected symbol and referencing sentence number routine
1	RJ	WA	WA2	
2	MJ	0	30000	
3	RJ	WA	WA1	} Sentence -- (If) Inconsistent sign change
4	TP	ID7	UP3	
5	RJ	UP2	UP	
6	MJ	0	IF	} Error routine referenced
7	40	IM3	4	
10	40	ID17	4	
11	0	IM	3	}
12	0	0	0	
13	RJ	ID2	ID	
14	TP	ID10	UP3	} Sentence -- (If) Symbol Rejected --
15	RJ	UP2	UP	
16	MJ	0	30000	
17	65	73472	55146	} Warning only
20	01	54304	43026	
21	66	30270	10101	
22	0	0	0	}
23	TP	A	ID33	
24	AT	IM16	IM15	
25	RJ	WA	WA1	} Sentence -- (____) Disallowable character in exponent ____
26	TP	ID32	UP3	
27	RJ	UP2	UP	
30	TP	ID33	A	}
31	MJ	0	30000	
32	40	IM7	7	
33	0	0	0	} Error routine referenced
34	RJ	WA	WA1	
35	TP	ID41	UP3	
36	RJ	UP2	UP	} <u>Sentence</u> -- (If) Incorrectly Written
37	RJ	SC7	SC5	
40	MJ	0	30000	
41	40	IM17	4	} Error routine referenced
42	RJ	WA	WA2	
43	TP	ID51	UP3	
44	RJ	UP2	UP	} Sentence -- (If) Scientific notation Incorrectly written
45	TP	ID41	UP3	
46	RJ	UP2	UP	
47	RJ	UZ	UZ1	} Error routine referenced
50	MJ	0	IF	
51	40	IM23	4	
52	RJ	UZ	UZ1	}
53	RJ	WA	WA2	
54	TP	ID57	UP3	
55	RJ	UP2	UP	} Sentence -- (If) Open absolute sign missing
56	MJ	0	30000	
57	40	IM27	5	

60	RJ	WA	WA2	}	Sentence -- (If) "To" should follow "Jump"
61	TP	ID64	UP3		
62	RJ	UP2	UP	}	Warning only
63	MJ	0	30000		
64	40	IM34	4	}	
65	RJ	SC4	SC		
66	TP	VN4	IM47	}	Sentence -- (If) becomes unconditional Jump to Sentence _____
67	TP	ID72	UP3		
70	RJ	UP2	UP	}	Warning only
71	MJ	0	30000		
72	40	IM40	10	}	
73	RJ	WA	WA2		
74	TP	ID100	UP3	}	Sentence -- (If) Space period occurs before sufficient data given
75	RJ	UP2	UP		
76	RJ	UZ	UZ1	}	Reference error routine
77	MJ	0	30000		
100	40	IM50	11	}	
101	40	IM61	7		
102	40	IM70	5	}	Sentence -- (If) Comparison symbols ambiguous
103	RJ	UZ	UZ1		
104	RJ	WA	WA2	}	Error routine referenced. Terminate analysis of sentence
105	TP	ID102	UP3		
106	RJ	UP2	UP	}	Sentence -- (If) Set of variables differs From Initial Set
107	MJ	0	IF		
110	RJ	UZ	UZ1	}	Error routine referenced. Analysis of sentence terminated.
111	RJ	WA	WA2		
112	TP	ID101	UP3	}	Sentence -- (If) (NOT) interpreted as (NOT EQUAL)
113	RJ	UP2	UP		
114	MJ	0	IF	}	Warning only
115	40	IM75	6		
116	RJ	WA	WA2	}	Sentence -- (If) Fixed- and Floating-pt. Values are not comparable
117	TP	ID115	UP3		
120	RJ	UP2	UP	}	Error routine referenced. Analysis of sentence discontinued.
121	MJ	0	30000		
122	RJ	WA	WA2	}	
123	RJ	UZ	UZ1		
124	TP	ID127	UP3	}	
125	RJ	UP2	UP		
126	MJ	0	IF	}	
127	40	IM103	11		

CA ID130

Excess-Three Print-Out Storage

	IA	IM		
0	65	30506	63050	S E N T E N
1	26	30017	77777	C E Δ
2	0	0	0	
3	34	50265	15065	I N C O N S
4	34	65663	05066	I S T E N T

5	01	65343	25001	△	S	J	G	N	△
6	26	33245	03230	C	H	A	N	G	E
7	01	27346	52446	△	D	I	S	A	L
10	46	51712	42546	L	O	W	A	B	L
11	30	01263	32454	E	△	C	H	A	R
12	24	26663	05477	A	C	T	E	R	△
13	01	34500	13072	△	I	N	△	E	X
14	52	51503	05066	P	O	N	E	N	T
15	0	0	0						
16	77	77010	10100			△	△	△	
17	01	34502	65154	△	I	N	C	O	R
20	54	30266	64673	R	E	C	T	L	Y
21	01	71543	46666	△	W	R	I	T	T
22	30	50010	17777	E	N	△	△	△	△
23	01	65263	43050	△	S	C	I	E	N
24	66	34313	42601	T	I	F	I	C	△
25	50	51662	46634	N	O	T	A	T	I
26	51	50777	77777	O	N				
27	01	51523	05001	△	O	P	E	N	△
30	24	25655	14667	A	B	S	O	L	U
31	66	30016	53432	T	E	△	S	I	G
32	50	01473	46565	N	△	M	I	S	S
33	34	50327	77777	I	N	G			
34	01	01665	10165	△	△	T	O	△	S
35	33	51674	62701	H	O	U	L	D	△
36	31	51464	65171	F	O	L	L	O	W
37	01	44674	75277	△	J	U	M	P	△
40	01	25302	65147	△	B	E	C	O	M
41	30	65010	16750	E	S	△	△	U	N
42	26	51502	73466	C	O	N	D	I	T
43	34	51502	44601	I	O	N	A	L	△
44	44	67475	20166	J	U	M	P	△	T
45	51	01653	05066	O	△	S	E	N	T
46	30	50263	00177	E	N	C	E	△	
47	0	0	0						
50	01	65522	42630	△	S	P	A	C	E
51	01	01523	05434	△	△	P	E	R	I
52	51	27015	12626	O	D	△	O	C	C
53	67	54650	12530	U	R	S	△	B	E
54	31	51543	00165	F	O	R	E	△	S
55	67	31313	42634	U	F	F	I	C	I
56	30	50660	12724	E	N	T	△	D	A
57	66	24013	23470	T	A	△	G	I	V
60	30	50777	77777	E	N				
61	01	65306	60151	△	S	E	T	△	O
62	31	01702	45434	F	△	V	A	R	I
63	24	25463	06501	A	B	L	E	S	△
64	27	34313	13054	D	I	F	F	E	R
65	65	01315	45147	S	△	F	R	O	M
66	01	34503	46634	△	I	N	I	T	I
67	24	46016	53066	A	L	△	S	E	T
70	01	26514	75224	△	C	O	M	P	A

71	54	34655	15001	R	I	S	O	N	△
72	65	73472	55146	S	Y	M	B	O	L
73	65	01244	72534	S	△	A	M	B	I
74	32	67516	76522	G	U	O	U	S	.
75	01	17505	16643	△	(	N	O	T	)
76	01	34506	63054	△	I	N	T	E	R
77	52	54306	63027	P	R	E	T	E	D
100	01	24650	11750	△	A	S	△	(	N
101	51	66013	05367	O	T	△	E	Q	U
102	24	46437	77777	A	L	)			
103	31	34723	02776	F	I	X	E	D	-
104	01	24502	70131		A	N	D	△	F
105	46	51246	63450	L	O	A	T	I	N
106	32	76525	13450	G	-	P	O	I	N
107	66	01702	44667	T	△	V	A	L	U
110	30	65012	45430	E	S	△	A	R	E
111	01	50516	60126	△	N	O	T	△	C
112	51	47522	45424	O	M	P	A	R	A
113	25	46307	77777	B	L	E			
	CA	IM114							

Superior to Lower-Case Figure Routine

	IA	IN		
0	MJ	0	30000	Exit
1	MJ	0	IN3	Entry
2	O	0	0	Input-Output Line
3	TP	IO2	IO1	Setting up index for 6 characters
4	LQ	IN2	6	} Masking out 1st character to A
5	QT	IO	A	
6	EJ	IO	IN23	Is character equal to 77?
7	RP	20012	IN25	} Is character one of superior X3 decimal figures?
10	EJ	IO3	IN11	
11	SN	Q	17	-n, -(j-r)
12	SA	IN7	0	-n, -(j-r) + n, j = r in u position
13	TU	A	IN15	} (io15+r) → in15 _u
14	RA	IN15	IN27	
15	TP	30000	A	X3 decimal figure representation → A _v
16	TP	IO	Q	} Translated figure is incorporated into Input-Output line
17	QS	A	IN2	
20	IJ	IO1	IN4	Index of input exhaustion
21	MJ	0	IN	To exit
22	LQ	IN2	6	} When 1st character = 77, no further translation to X3 decimal is performed. The I-O line is merely returned to original position.
23	IJ	IO1	IN22	
24	MJ	0	IN	Exit
25	RJ	ID31	ID23	Sentence --- () Disallowable character in exponent _____.
26	MJ	0	IN16	
27	O	IO14	0	Parameter
	CA	IN30		

Storage for "IN"

	IA	IO		
0	0	0	77	
1	0	0	[5]	
2	0	0	5	
3	0	0	60	0
4	0	0	61	1
5	0	0	40	2
6	0	0	20	3
7	0	0	41	4
10	0	0	35	5
11	0	0	55	6
12	0	0	75	7
13	0	0	36	8
14	0	0	57	9
15	0	0	3	0
16	0	0	4	1
17	0	0	5	2
20	0	0	6	3
21	0	0	7	4
22	0	0	10	5
23	0	0	11	6
24	0	0	12	7
25	0	0	13	8
26	0	0	14	9
	CA	IO27		
	IA	IQ		
Scientific	0	MJ	0	30000
Notation	1	MJ	0	IQ4
Evaluation	2	0	0	0
Routine	3	0	0	0
	4	MJ	0	IQ107
	5	TP	IQ3	A
	6	ZJ	IQ7	IQ
	7	TP	IQ2	A
	10	SJ	IQ11	IQ14
	11	TM	A	IQ2
	12	TP	IR11	IS2
	13	TP	IQ2	A
	14	TJ	IR1	IQ24
	15	EJ	IR1	IQ30
	16	TJ	IR	IQ30
	17	RA	IQ3	IR2
	20	FD	IQ2	IR
	21	TP	Q	IQ2
	22	TP	Q	A
	23	MJ	0	IQ15
	24	RS	IQ3	IR2
	25	FM	IQ2	IR
	26	TP	Q	IQ2
	27	MJ	0	IQ13
				Exit
				Entry
				Input in fltpt. of no. to left of e or *.
				May be neg. or pos. Also output line
				Exponent of 10 in octal. May be negative
				or positive
				If exponent is zero, goes to exit
				Is floating point number negative?
				Changing fltpt. no. to positive and
				putting negative indicator into is2
				Is 1 > floating point input
				Is 1 = floating point input
				Is 10 > floating point input
				Loop to reduce floating-point
				number to one between 1 and 10
				such that $1 \leq \text{floating point no.} < 10$
				Loop to increase floating-point number
				such that $1 \leq \text{floating-point no.} < 10$

30	TP	IQ3	A	
31	ZJ	IQ32	IQ45	Is exponent zero?
32	SJ	IQ33	IQ51	Is exponent negative?
33	TM	A	A	} Is 39 > exponent
34	TJ	IR4	IQ54	
35	EJ	IR4	IQ37	
36	MJ	0	IQ101	
				Output to zero. Sentence ____ (____) absolute value of number too small - given zero value
37	TP	IQ2	A	} Is (IR7) > (IQ2)? Same printout and action as above.
40	TJ	IR7	IQ101	
41	FD	IQ2	IR	} Dividing IQ2 by 10 and storing in IQ2
42	TP	Q	IQ2	
43	FD	IQ2	IR10	} Dividing IQ2 by 10 ³⁸ and storing in IQ2
44	TP	Q	IQ2	
45	TP	IS2	Q	} Is negative value of floating point desired?
46	QJ	IQ47	IQ	
47	TN	IQ2	IQ2	Making floating-point output negative
50	MJ	0	IQ	
51	TJ	IR5	IQ54	Is 38 > exp.?
52	EJ	IR5	IQ71	Is 38 = exp.?
53	MJ	0	IQ75	Jump to error print-out section
54	ST	IR2	IS	Setting up index for multiplication
55	TP	IR	IS1	10 → IS1
56	IJ	IS	IQ60	Index jump
57	MJ	0	IQ63	
60	FM	IS1	IR	} Multiplying 10 to power desired in floating-point
61	TP	Q	IS1	
62	MJ	0	IQ56	
63	TP	IQ3	A	} Is exponent negative?
64	SJ	IQ65	IQ67	
65	FD	IQ2	IS1	Dividing fltpt. no. by power of 10 desired
66	MJ	0	IQ44	
67	FM	IQ2	IS1	Multiplying fltpt. no. by power of 10 desired
70	MJ	0	IQ44	
71	TP	IR6	A	Is (IQ2) > (IR6)? If so, too large.
72	TJ	IQ2	IQ75	
73	FM	IQ2	IR10	Multiplying by 10 ³⁸
74	MJ	0	IQ44	
75	RJ	WA	WA1	} Sentence ____ (____) absolute value of number too large
76	TP	IQ112	UP3	
77	RJ	UP2	UP	
100	MJ	0	IQ	Error routine referenced
101	TP	IR3	IQ2	} Sentence ____ (____) absolute value of number too small - given zero value
102	RJ	WA	WA2	
103	TP	IQ113	UP3	
104	RJ	UP2	UP	
105	TP	IQ114	UP3	Warning only
106	MJ	0	IQ77	
107	TP	IR3	IS2	Clearing is 2

110	TP	IQ2	A	}	If floating point input is zero, goes to exit
111	ZJ	IQ5	IQ		
112	40	IQ115	6		
113	40	IQ115	5		
114	40	IQ123	4		
115	01	01242	56551	△	△ A B S O
116	46	67663	00170	L	U T E △ V
117	24	46673	00151	A	L U E △ O
120	31	01506	74725	F	△ N U M B
121	30	54016	65151	E	R △ T O O
122	01	46245	43230	△	L A R G E
123	01	65472	44646	△	S M A L L
124	02	02323	47030	-	- G I V E
125	50	01743	05451	N	△ Z E R O
126	01	70244	66730	△	V A L U E
	CA	IQ127			

Constants for IQ

	IA	IR		
0	20	45000	0	10
1	20	14000	0	1
2	0	0	1	
3	0	0	0	
4	0	0	47	39
5	0	0	46	38
6	20	16634	37341	≅ 1.70141184
7	20	15701	21001	≅ 1.46936801
10	37	74547	32316	10 ³⁸
11	40	0	0	Indicator
	CA	IR12		

Temporary Storage

IS	0	Multiplication index
	1	Multiplication storage
	2	Negative answer indicator

If String-Out (Tape 3)

Jump Routine

	IA	IJ		
0	RJ	SY	SY1	
1	TP	SY2	A	
2	EJ	UF51	IJ14	TO?
3	RJ	ID63	ID60	Sentence -- (If) "To" should follow "JUMP"
4	TP	SY2	A	
5	RP	20007	IJ7	} TO, Sentence, Statement, line, number,
6	EJ	UF51	IJ14	
7	EJ	UF14	IK	Δ .
10	TP	SY5	A	} Are no. of chars. of what should be
11	TJ	UF23	IJ16	
				figure<7?
12	RJ	ID40	ID34	Sentence -- (If) Incorrectly Written --
13	MJ	0	IF	
14	RJ	SY	SY1	Get next symbol
15	MJ	0	IJ4	
16	TP	SY2	LN4	} Getting line number in proper form and
17	RJ	LN2	LN	
20	TP	LN3	VN5	} storing output
21	TP	VN70	Q	
22	QJ	IJ23	IJ53	} Is X floating-point constant?
23	TP	VN121	Q	
24	QJ	IJ25	IJ30	} Is Y floating-point constant?
25	TP	VN40	IT	
26	TP	VN71	IT1	} Sent to preliminary number comparison
27	MJ	0	IT2	
30	TP	VN74	Q	} Is Y fixed-point variable?
31	QJ	ID12a	IJ32	
32	TP	VN72	A	} Is Y fixed-point constant?
33	ZJ	IJ40	IJ34	
34	TP	VN40	A	} Y is a floating-point variable. Call
35	RJ	GW	GW1	
36	TP	Q	VN57	word for X constant to VN57
37	MJ	0	IJ160	} Jump to end
40	RJ	IJ52	IJ42	
41	MJ	0	IJ25	Y to floating-point
42	TP	VN72	GG4	Jump to comparison
43	TP	UF31	GG5	} Translation of Y fixed-point constant
44	RJ	GG2	GG	
45	TP	VN106	Q	} to floating-point and storage
46	QJ	IJ47	IJ51	
47	TN	GG3	VN71	} .
50	MJ	0	IJ52	
51	TP	GG3	VN71	} Is X fixed-point constant?
52	MJ	0	30000	
53	TP	VN41	A	
54	ZJ	IJ55	IJ131	

X is
floating-
point
constant

X is fixed- point constant	55	TP	VN121	Q	Is Y floating-point constant? X to floating-point Jump to comparison
	56	QJ	IJ57	IJ72	
	57	RJ	IJ71	IJ61	
	60	MJ	0	IJ25	
	61	TP	VN41	GG4	
	62	TP	UF31	GG5	
	63	RJ	GG2	GG	
	64	TP	VN55	Q	
	65	QJ	IJ66	IJ70	
	66	TN	GG3	VN40	
X is fixed- point constant	67	MJ	0	IJ71	Translation of X fixed-point constant to floating-point constant and storage
	70	TP	GG3	VN40	
	71	MJ	0	30000	
	72	TP	VN72	A	
	73	ZJ	IJ74	IJ122	
	74	RJ	IJ106	IJ77	
	75	RJ	IJ116	IJ107	
	76	MJ	0	IJ117	
	77	TP	VN41	RS4	
	100	RJ	RS2	RS	
	101	TP	VN55	Q	Translation of X excess-three decimal to octal and storage
	102	QJ	IJ103	IJ105	
	103	TN	RS3	VN41	
	104	MJ	0	IJ106	
	105	TP	RS3	VN41	
	106	MJ	0	30000	
	107	TP	VN72	RS4	
	110	RJ	RS2	RS	
	111	TP	VN106	Q	
	112	QJ	IJ113	IJ115	
	113	TN	RS3	VN72	Translation of Y excess-three decimal to octal and storage
	114	MJ	0	IJ116	
	115	TP	RS3	VN72	
	116	MJ	0	30000	
	117	TP	VN41	IT	
	120	TP	VN72	IT1	
	121	MJ	0	IT2	
	122	TP	VN74	Q	
	123	QJ	IJ124	IJ127	
	124	RJ	IJ106	IJ77	
X is fixed- pt. con- stant	125	TP	VN41	A	Is Y fixed-point variable? X to octal and storage Getting call word for X and storing X to floating point and storing Getting call word for X and storing Is X fixed-point variable? Is Y floating-point constant? Is Y fixed-point constant?
	126	MJ	0	IJ35	
	127	RJ	IJ71	IJ61	
	130	MJ	0	IJ34	
	131	TP	VN43	Q	
	132	QJ	IJ133	IJ146	
	133	TP	VN121	Q	
	134	QJ	ID122	IJ135	
	135	TP	VN72	A	
	136	ZJ	IJ137	IJ144	

X is fixed- point variable	137	RJ	IJ116	IJ107	Y to octal and storage
	140	TP	VN72	A	
	141	RJ	GW	GW1	Getting call word for Y and storing
	142	TP	Q	VN110	
	143	MJ	0	IJ160	
X is float- ing point varia- ble	144	TP	VN74	Q	
	145	QJ	IJ160	ID122	Is Y fixed-point variable?
	146	TP	VN121	Q	
	147	QJ	IJ156	IJ150	Is Y floating-point constant?
	150	TP	VN72	A	
	151	ZJ	IJ152	IJ154	Is Y fixed-point constant?
	152	RJ	IJ52	IJ42	Y to floating-point constant
	153	MJ	0	IJ156	
	154	TP	VN74	Q	
	155	QJ	ID122	IJ160	Is Y fixed-point variable?
	156	TP	VN71	A	
	157	MJ	0	IJ141	Getting call word for Y and storing
	160	RP	30010	IJ162	
	161	TP	VN54	VN11	
	162	RP	30010	IJ164	Transfer of prepared data to output region
	163	TP	VN105	VN21	
	164	TP	VN5	A	
	165	RJ	IX	IX1	
	166	RJ	SY	SY1	
	167	TP	SY2	A	
	170	RP	20004	IJ172	() , ; tests
	171	EJ	UF16	IJ166	
	172	EJ	UF14	IK	△ . check
	173	EJ	UF2	IU	"if" check
	174	RJ	ID40	ID34	Sentence -- (if) Incorrectly Written --
	175	MJ	0	IF	
		CA	IJ176		

Preliminary Number Comparison Routine

	IA	IT		
	0	0	0	1st no. X
	1	0	0	2nd no. Y
	2	TP	VN4	A
	3	EJ	UF26	IT11
	4	EJ	UF60	IT14
	5	EJ	UF61	IT17
	6	EJ	UF62	IT22
	7	EJ	UF63	IT25
	10	MJ	0	IT30
<	11	TP	IT	A
	12	TJ	IT1	IT32
	13	MJ	0	IF1
=	14	TP	IT	A
	15	EJ	IT1	IT32
	16	MJ	0	IF1
>	17	TP	IT1	A

Change "if" sentence to a "jump" sentence
Jump back to start of IF

	20	TJ	IT	IT32	
	21	MJ	0	IF1	
≤	22	TP	IT1	A	
	23	TJ	IT	IF1	
	24	MJ	0	IT32	
≠	25	TP	IT	A	
	26	EJ	IT1	IF1	
	27	MJ	0	IT32	
≥	30	TP	IT	A	
	31	TJ	IT1	IF1	
	32	TP	UF63	VN	Number (6) to 1st line of output
Changing	33	TP	UF15	VN2	JUMP put into title output line
IF to a	34	TP	VN5	VN4	Line number changed to 5th position of
Jump					output
Sentence	35	TP	VN33	VN5	Pseudo-op indicator to 6th output line
	36	RJ	ID71	ID65	Sentence -- (If) Becomes Unconditional
					Jump to Sentence --
	37	TP	VN4	A	} Line number to reference list
	40	RJ	IX	IX1	
	41	RJ	WT	WT1	} Write string-out on tape
	42	MJ	0	IF	
		CA	IT43		

Termination Space Period Routine

		IA	IK		
	0	TP	VN5	A	} Has a 1st number been filled in?
	1	ZJ	IK2	IK12	
	2	TP	VN4	A	} Has a 1st relation been filled in?
	3	ZJ	IK4	IK12	
	4	TP	VN14	A	} Is there a call word for X?
	5	ZJ	IK6	IK12	
	6	TP	VN24	A	} Is there a call word for Y?
	7	ZJ	IK10	IK12	
	10	RJ	WT	WT1	To write completed data on tape
	11	MJ	0	IF	Exit
	12	RJ	ID77	ID73	Sentence -- (If) Space period occurs
					before sufficient data given
	13	MJ	0	IF	
	14	TP	VN10	A	} Has a 3rd no. been filled in?
	15	ZJ	IK	JF141	
	16	TP	VN7	A	} Has a 2nd no. been filled in?
	17	ZJ	IK20	IK12	
	20	TP	VN6	A	} Has a 2nd relation been filled in?
	21	ZJ	IK	IK12	
		CA	IK22		

2nd Control Routine

		IA	IU		
Initial	0	TP	VN56	VN43	Abs. sign of X to more convient location
Control-	1	TP	VN107	VN74	Abs. sign of Y to more convient location
2nd or	2	RP	10015	IU4	Clearing region of temporary Storage
3rd time	3	TP	UF7	VN150	
thru	4	TV	IU33	JC	Setting up divider routine for 2nd or
					3rd time thru

5	RJ	SY	SY1	Get next symbol
6	TP	SY2	A	
7	TP	SY2	VN152}	Store 1st two symbol words in temporary
10	TP	SY3	VN153}	storage
11	EJ	UF4	IU25	-?
12	EJ	UF3	IU31	!?
13	EJ	UF17	IU5	(?
14	EJ	UF14	IK14	Δ. termination
15	TP	SY11	Q	Is symbol a constant?
16	QJ	IY	IU17}	
17	TP	SY10	Q	Is symbol a fixed-point variable?
20	QJ	IY33	IU21}	
21	TP	SY7	Q	Is symbol a floating-point variable?
22	QJ	JB	IU23}	
23	RJ	ID16	ID13	Sentence -- (If) Symbol Rejected --
24	MJ	0	IU5	
25	TP	VN151	Q	Negative sign if no previous absolute sign
26	QJ	IU5	IU27}	
27	TP	UF27	VN150}	
30	MJ	0	IU5}	
31	TP	UF27	VN151	Absolute sign indicator to temporary location
32	MJ	0	IU5	
33	0	0	JC3	
	CA	IU34		

2nd Constant and Fixed-Point Variable Routine

	IA	IY		
Constant	0	TP	VN47	A }
& Fix-Pt.	1	ZJ	IY4	IY2}
Var.	2	TP	VN100	A }
Routine -	3	ZJ	IY4	IY33}
2nd or	4	RJ	SY	SY1
3rd time	5	TP	SY2	A
Thru	6	EJ	UF	IY4
	7	EJ	UF1	IY21
	10	EJ	UF4	IY16
	11	EJ	UF5	IY16
	12	TP	SY11	Q }
	13	QJ	IY14	ID110}
	14	TP	SY2	VN155
	15	MJ	0	IY33
	16	TP	UF27	VN154
	17	RJ	SY	SY1
	20	MJ	0	IY12
	21	RJ	SY	SY1
	22	TP	SY2	A
	23	EJ	UF22	IY25
	24	MJ	0	ID42
	25	RJ	SY	SY1
	26	TP	SY2	A
	27	EJ	UF25	IY31

Did X have a number in scientific notation?
 Did Y have a number in scientific notation?
 Get next symbol
 e?
 *?
 -?
 -?
 Constant?
 Exponent to storage
 Negative sign indicator (40 0 0) to storage
 Get next symbol
 Get next symbol
 10?
 Get next symbol
 POW?

30	MJ	O	IY10	
31	RJ	SY	SY1	Get next symbol
32	MJ	O	IY10	
33	RJ	JB17	JB6	Termination check loop
34	RJ	ID16	ID13	
35	MJ	O	IY33	
	CA	IY36		

2nd Floating-Point Variable Routine

		IA	JB		
2nd or	0	RJ	TA	TA1	Is symbol in Comb. List?
3rd time	1	MJ	O	JB32	Not in
thru	2	TP	UF36	A	Is in
	3	TJ	TA4	JB37	Go to subscripted variable portion
Nonsub-	4	RJ	JB17	JB6	
script-	5	MJ	O	JB24	
ed var-	6	RJ	SY	SY1	Get next symbol
iable	7	TP	SY2	A	
	10	RP	20004	JB12	) (, ; tests
	11	EJ	UF16	JB6	
Termination	12	EJ	UF3	JB20	Abs. sign test
Loop	13	RP	20004	JB15	< > = NOT tests
	14	EJ	UF10	JC	
	15	EJ	UF15	JF	JUMP?
	16	EJ	UF14	IK14	△.
	17	MJ	O	30000	
	20	TP	VN151	Q	Has open absolute sign been given?
	21	QJ	JB6	JB22	
	22	RJ	ID56	ID52	"Sentence -- (If) Open absolute sign missing" Error routine reference and continuation of analysis
	23	MJ	O	JB6	
	24	TP	SY7	Q	Is symbol a variable? If either a variable or constant, it is
	25	QJ	JB4	JB26	
	26	TP	SY11	Q	Is symbol a constant? assumed to be redundant argument of a single valued function and no notice is taken of it
	27	QJ	JB4	JB30	
	30	RJ	ID16	ID13	Sentence -- If Symbol Rejected --
	31	MJ	O	JB4	
Not in	32	RJ	TS	TS1	Is symbol in pseudo-op list?
	33	MJ	O	ID110	Sentence -- (If) Set of variables differs from Initial Set
Is in	34	TP	UF34	A	Is variable subscripted?
	35	TJ	TS3	JB37	
	36	MJ	O	JB4	No
Sub-	37	TV	JB60	JB55	Setting up v of storage subroutine
scripted	40	TP	UF60	JB61	Setting up index
variable	41	RJ	JB17	JB6	Termination check
	42	TP	SY11	Q	Is symbol a constant?
	43	QJ	JB50	JB44	

	44	TP	SY10	Q	} Is symbol a fixed-point variable?
	45	QJ	JB50	JB46	
	46	RJ	ID16	ID13	
	47	MJ	0	JB41	
	50	RJ	JB57	JB55	Storing const. or fix.-pt. var. XS3 representation
	51	IJ	JB61	JB41	Index jump to start of loop
	52	RJ	JB17	JB6	} Termination check loop
	53	RJ	ID16	ID13	
	54	MJ	0	JB52	
	55	TP	SY2	[VN156]	
	56	RA	JB55	UF6	} Storage subroutine
	57	MJ	0	30000	
	60	0	0	VN156	
	61	0	0	0	
		CA	JB62		Index temporary
		IA	JC		
Relation	0	RJ	JC	JC3	} Sentence -- (If) Incorrectly Written --
Symbol	1	RJ	ID40	ID34	
Routine-	2	MJ	0	IF	
2nd or	3	TP	A	VN163	
3rd time	4	TP	UF6	VN162	1st relation symbol to storage
thru					Starting count in counter of relation symbols
	5	RJ	SY	SY1	Get next symbol
	6	TP	SY2	A	
	7	RP	20003	JC14	} <> = ?
	10	EJ	UF10	JC11	
	11	TP	A	VN164	2nd relation symbol to storage
	12	RA	VN162	UF6	Count of relation symbols increased
	13	MJ	0	JC15	
	14	EJ	UF13	JC1	Is 2nd symbol NOT?
	15	RP	30015	JC17	} Transferring data of X ₂ to VN130-144
	16	TP	VN150	VN130	
	17	RP	10015	JC21	} Clearing VN150-164 for use of Y ₂ data accumulation
	20	TP	UF7	VN150	
	21	TP	VN142	A	} Is 2 > no. of relation symbols, some value of SY2 hasn't yet been identified and stored
	22	TJ	UF26	IU6	
	23	MJ	0	IU5	Return to gather data on Y
		CA	JC24		
					2nd Jump Routine
		IA	JF		
Jump		TU	JF156	JF145	
Routine-	0	TU	JF160	JF146	
2nd or	1	TV	JF154	JF147	
3rd time	2	RJ	JF153	JF144	} Check if X ₂ = X
thru	3	TU	JF157	JF145	
	4	TU	JF161	JF146	
	5	TV	JF155	JF147	
	6	RJ	JF153	JF144	} Check if Y ₂ = Y
	7				

	10	RJ	JI	JI1	Checks consistency of relation symbols and assigns code figures
	11	TP	VN130	A	
	12	EJ	VN55	JF16	1st left sign = 2nd left sign $X_2(s)=X(s)$
	13	TP	VN150	A	1st left $\neq$ 2nd left $X_2(s) \neq X(s)$
	14	EJ	VN106	ID3	Inconsistent sign change $Y_2(s) = Y(s)$
	15	MJ	0	JF41	Double sign inequality. Relation reversal needed. $Y_2(s) \neq Y(s)$
	16	TP	VN150	A	$Y_2(s) = Y(s)$
	17	EJ	VN106	JF57	Sign of 2nd rt. = sign of 1st rt. Signs O.K. Reversal of relation not needed
	20	MJ	0	ID3	Inconsistent sign change. $Y_2 \neq Y$
	21	TU	JF156	JF145	
	22	TU	JF161	JF146	
	23	TV	JF155	JF147	
	24	RJ	JF153	JF144	Left = Right. Check if $X = Y_2$
	25	TU	JF157	JF145	
	26	TU	JF160	JF146	
	27	RJ	JF153	JF144	Right = Left. Check if $Y = X_2$
	30	RJ	JI	JI1	Checks consistency of relation symbols and assigns code figures
	31	TP	VN130	A	
	32	EJ	VN106	JF36	Sign follows group equality #1 $X_2(s) = Y(s)$
	33	TP	VN150	A	$X_2(s) \neq Y(s)$
	34	EJ	VN55	ID3	Inconsistent sign change $Y_2(s) = X(s)$
	35	MJ	0	JF57	$Y_2(s) \neq X(s)$ Double inequality of signs and reverse equality of no. series means relation reversal not needed
	36	TP	VN150	A	
	37	EJ	VN55	JF41	$Y_2(s) = X(s)$. Sign follows group reverse equality 2nd time. Relation reversal needed
	40	MJ	0	ID3	Inconsistent sign change. $Y_2(s) \neq X(s)$
Reversal of relation symbols due to changing of signs or altering of position of X & Y the 2nd or 3rd time thru	41	TP	VN7	A	Is line no. for 2nd test in output?
	42	ZJ	JF43	JF46	
	43	TP	UF63	A	
	44	ST	JI123	JI123	Reversing relation of single symbol in temp. storage $6-\{2,3,4\}=\{4,3,2\}$
	45	MJ	0	JF57	
	46	TP	VN6	A	
	47	ZJ	JF50	JF53	Is 2nd test line $\neq$ 0?
	50	TP	UF63	A	
	51	ST	VN6	VN6	
	52	MJ	0	JF57	Reversing relation of 2nd test line. $6-\{2,3,4\}=\{4,3,2\}$
	53	TP	VN65	A	
	54	EJ	UF26	JF43	
	55	TP	UF24	A	Was 1st test a double relation test?
	56	ST	JI123	JI123	
					$12 - \{5,6,7\} = \{7,6,5\}$

Check of consis- tency of relation symbols	57	TP	VN7	A	} Is line no. for 2nd test zero?
	60	ZJ	JF61	JF66	
	61	TP	UF50	A	
	62	ST	VN4	A	
	63	ST	VN6	A	
	64	EJ	J1123	JF76	
	65	MJ	0	ID103	
	66	TP	VN6	A	
	67	ZJ	JF70	JF72	
	70	EJ	VN4	ID103	
71	MJ	0	JF76	Is 2nd test line = 0?	
72	TP	UF50	A	} Is (VN6) = (VN4)?	
73	ST	VN4	A		
74	EJ	J1123	JF76		
75	MJ	0	ID103		
Loop to get line number	76	RJ	SY	SY1	[9 - (VN4)] should equal (J1123)
	77	TP	SY2	A	Comparison symbols ambiguous
	100	EJ	UF51	JF112	Start of procedure to get line no.
	101	RJ	ID63	ID60	Getting next symbol
	102	TP	SY2	A	Is symbol TO
	103	RP	20007	JF105	TO should follow Jump (Warning)
	104	EJ	UF51	JF112	Is symbol TO, SENTEN, STATEM, LINE, NUMBER, NO., SENT.?
	105	EJ	UF14	IK14	△. test
	106	TP	SY5	A	Is 7 > no. of chars.
	107	TJ	UF23	JF114	Incorrectly Written --
110	RJ	ID40	ID34	Exit	
111	MJ	0	IF	Get next symbol	
112	RJ	SY	SY1	} Getting line number in standard form	
113	MJ	0	JF102		
114	TP	SY2	LN4		
115	RJ	LN2	LN		
116	TP	LN3	A	} Sending line number to reference list	
117	RJ	IX	IX1		
120	TP	VN6	A		
121	ZJ	JF122	JF124		
122	TP	VN7	A	Is 2nd test line zero?	
123	ZJ	JF124	JF131	} Has a line number already been put in VN7?	
124	TP	LN3	VN10		
125	RJ	SY	SY1		
126	EJ	UF16	JF125		
127	EJ	UF14	IK14	) ?	
130	MJ	0	IJ12	△. routine of 2nd or 3rd time thru	
131	TP	LN3	VN7	Incorrectly Written _____	
132	RJ	SY	SY1	Line no. to VN7	
133	RP	20004	JF135	} (, ; ?	
134	EJ	UF16	JF132		
135	EJ	UF14	IK14		
136	TP	UF60	VN34		
137	EJ	UF2	IU2	△. test?	
140	MJ	0	IJ12	Putting indicator "3" in output to in- dicate there are to be 3 clauses	
Exit loop after line no. for VN7 is ob- tained. May be start of 3rd time thru if an "if" is recognized					Start of 3rd time thru with recognition of "IF"
					Incorrectly Written --

141	TP	VN34	A	}	Is this the third clause being analyzed?
142	EJ	UF60	IK12		
143	MJ	0	IK16		
144	TP	UF65	JF162		Index set up
145	TP	[VN43]	A	}	Check if storage of data for a constant or variable is equal to same data gathered for one on the first part of an "if" sentence
146	EJ	[VN131]	JF150		
147	MJ	0	JF21		
150	RA	JF145	UF42		
151	RA	JF146	UF42		
152	IJ	JF162	JF145		
153	MJ	0	30000		
154	0	0	JF21		
155	0	0	ID110		
156	0	VN43	0		X absolute
157	0	VN74	0		Y absolute
160	0	VN131	0		
161	0	VN151	0		
162	0	0	0		Index temporary storage
	CA	JF163			

VARY Translation Routine

The VARY translation routine builds from the input VARY sentence the required lists of symbols for generating the VARY coding and for providing the necessary connections so that the stated looping processes can be accomplished. These lists are called the VARY String-out, the VARY File, and the Variable List. In addition to these tasks, this routine makes appropriate checks, where possible, for the writing of VARY sentences containing non-ending loops. It also checks for compatibility of fixed- or floating-point variables and constants within a Modify Component of a VARY sentence.

String-Out Form for VARY Sentence

WLO	0	0	0	0	0	0	0	0	0	t	t	t	ttt = number of words in string-out
1	—	—	—	.	—	—	—	—	—	—	—	—	Sentence number in standard form
2	V	A	R	Y	7	7	7	7	7	7	7	7	XS-3 sentence identifier
3	0	0	0	0	0	0	0	s	s	s	s	s	sssss = sentence call word
4	0	0	0	0	0	0	0	0	0	0	y	y	YY = number of WITH words (max. 17 ₈)
5	—	—	—	.	—	—	—	—	—	—	—	—	Sentence number of first in range
6	—	—	—	.	—	—	—	—	—	—	—	—	Sentence number of last in range
7	—	—	—	.	—	—	—	—	—	—	—	—	Sentence number of transfer, if stated (zero otherwise)
10	r	0	0	0	0	0	0	0	0	0	0	0	r = 4 if WL7 is RESUME number (zero otherwise)
11	0	0	0	0	0	W	X	0	0	0	0	0	WX = indicator for tests of first variable
12	P	0	Call word ₀					Call word ₀					Variable. P=4 if call word ₀ is 63--- and floating. (zero otherwise)
13	q ₁	0	Call word ₁					Call word ₁					Initial value } q _i =4 if absolute value is specified (except constants)
14	q ₂	0	Call word ₂					Call word ₂					
15	q ₃	0	Call word ₃					Call word ₃					
16	0	0	0	0	0	W	X ₁	0	0	0	0	0	WX ₁ = indicator for tests of second variable
17	P ₁	0	CW ₀ ¹					CW ₀ ¹					Variable
20	q ₂ ¹	0	CW ₁ ¹					CW ₁ ¹					Initial value } P and q _i have same meaning as above
21	q ₃ ¹	0	CW ₂ ¹					CW ₂ ¹					
22	q ₄ ¹	0	CW ₃ ¹					CW ₃ ¹					
													Additional 5 word components as required

The WX indicators in the String-out provide to the VARY generator the information required so that the necessary tests, if any, may be built to guard against indefinite loops within the VARY cycle. These indicators may be explained by the modify component.

VARY α a(b)c ,

where α denotes the fixed or floating variable, and a, b, and c are fixed-or floating-point variables and/or constants.

Then, WX = 00 if a, b, and c are all constants. No test coding will be generated

WX = 10 if a and/or c are variable and b is a stated constant > 0 .

WX = 14, if a and/or c are variable and b is a stated constant < 0 .

WX = 20, if b is a variable, and a and c are constants such that $c-a > 0$.

WX = 21, if b is a variable, and a and c are constants such that $c-a < 0$.

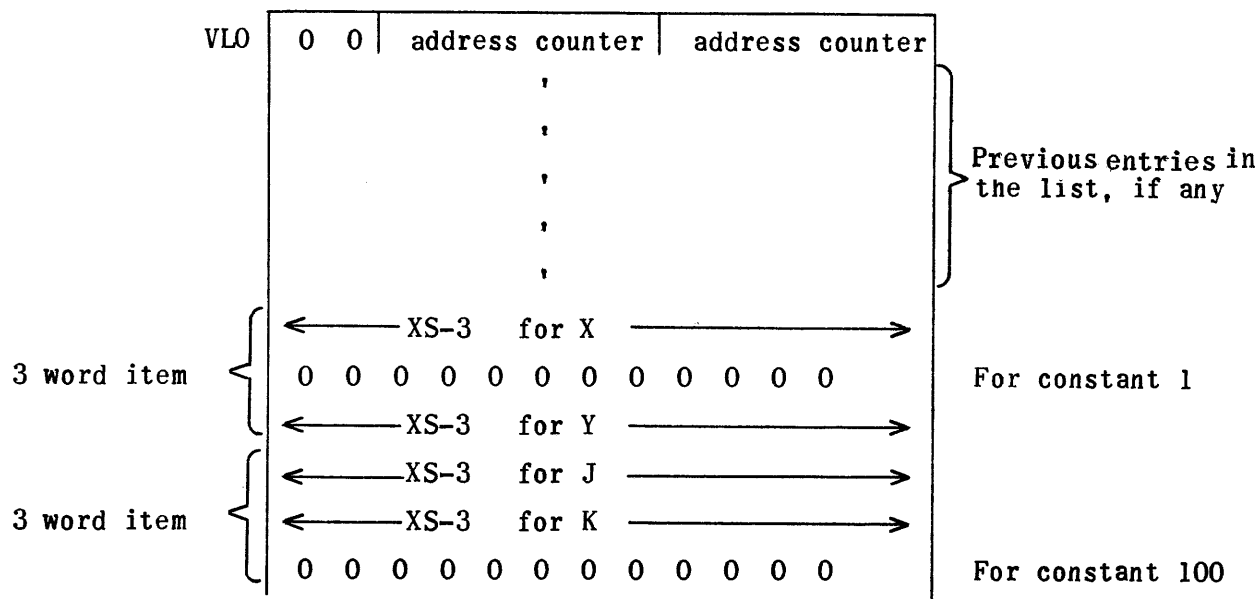
WX = 22, if b is a variable, and a and c are constants such that $c-a = 0$

WX = 30, if b is a variable, and a and/or c are variable.

Since the test coding that is built by the VARY generator for all cases except WX = 00 is only executed at the beginning of the looping process, none of the variables α , b, or c may be changed by other sentences within the VARY's range. In order to check that these variables are not altered by their appearance in other Modify components of this or other VARY sentences, or in COMPUTE sentences, or on the left of equations, the Variable List is built by the VARY translator. This list consists of three word entries for each modify component of VARY sentences. These entries remain in the list only until the translation process has progressed beyond the range of the VARY.

Example: VARY X 0(1)Y with J I(K)100 Sentences 10 thru 20 .

After the insertion of the 2 three-word items for this sentence, the Variable List would appear as follows:



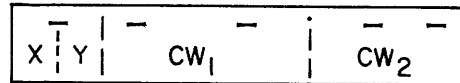
When a constant appears for b or c in the Modify component, the corresponding word in the Variable List is cleared.

The three-word items entered for the above sentence remain in the list for checking by the COMPUTE and Equation translators until sentence 20 has been processed. Then the VE routine of the Translation subroutines removes these items from the list.

The first entries into the VARY File are also made by the VARY translator. This file is used later, to:

1. Determine when items are to be removed from the Variable List.
2. Provide exits from VARY sentences when transfer components are not stated.
3. Establish the continuation of the VARY loop from the last sentence in its range.

A VARY File item consists of two words for each VARY sentence:



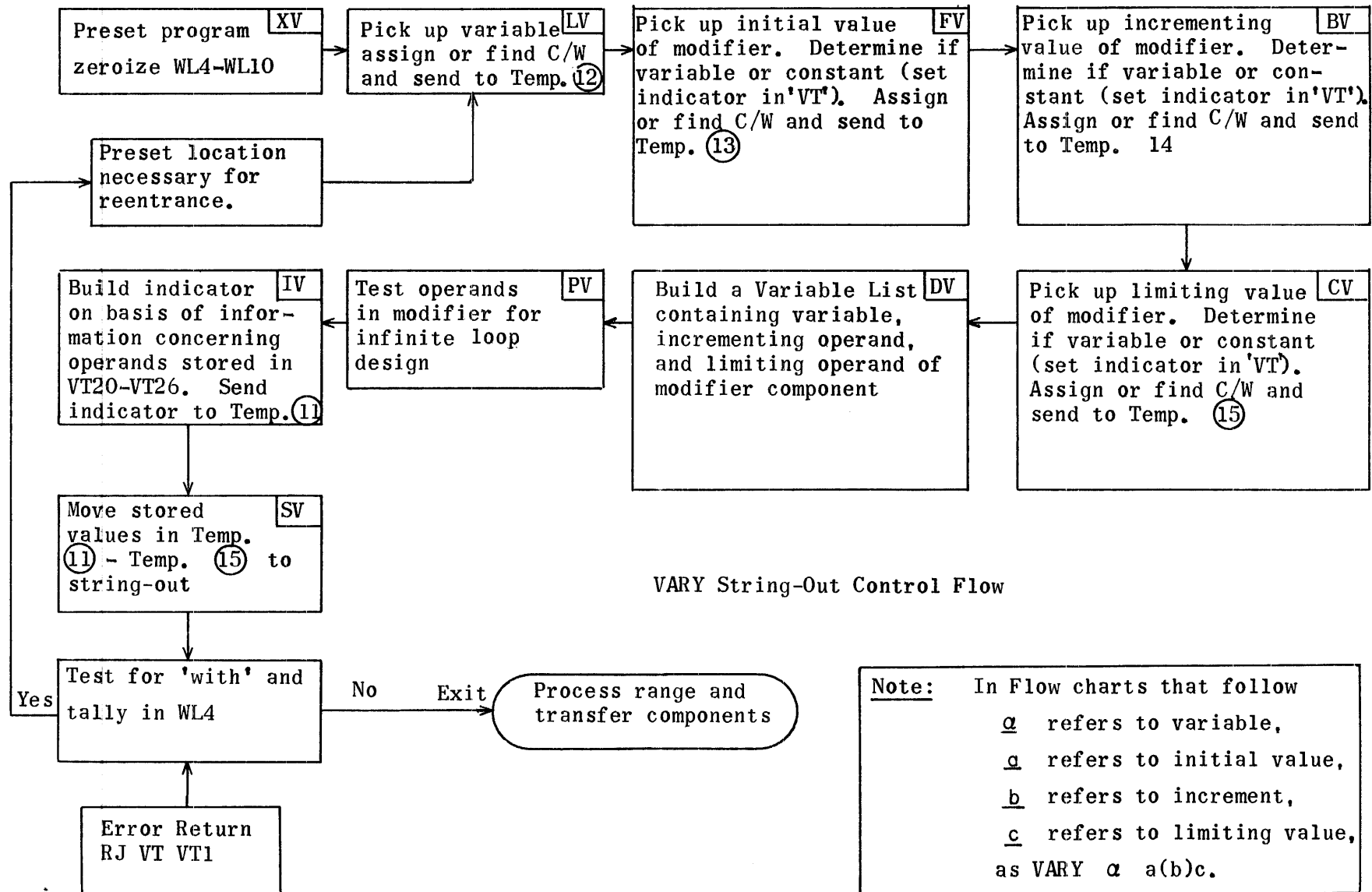
Sentence number of last sentence
 in range.
 CW₂ = Call word of VARY sentence.
 CW₁ = Call word of sentence to
 which exit is made.
 Y(4 bits) = number of WITH words.
 X(2 bits) = 10 if exit is normal
 transfer.
 = 11 if exit is RESUME
 of previous VARY.

All entries in the VARY File item except CW1 and X are made by the
 VARY translator. The item is completed later by the VE routine of the
 Translation Subroutines (see Section III, 3,a).

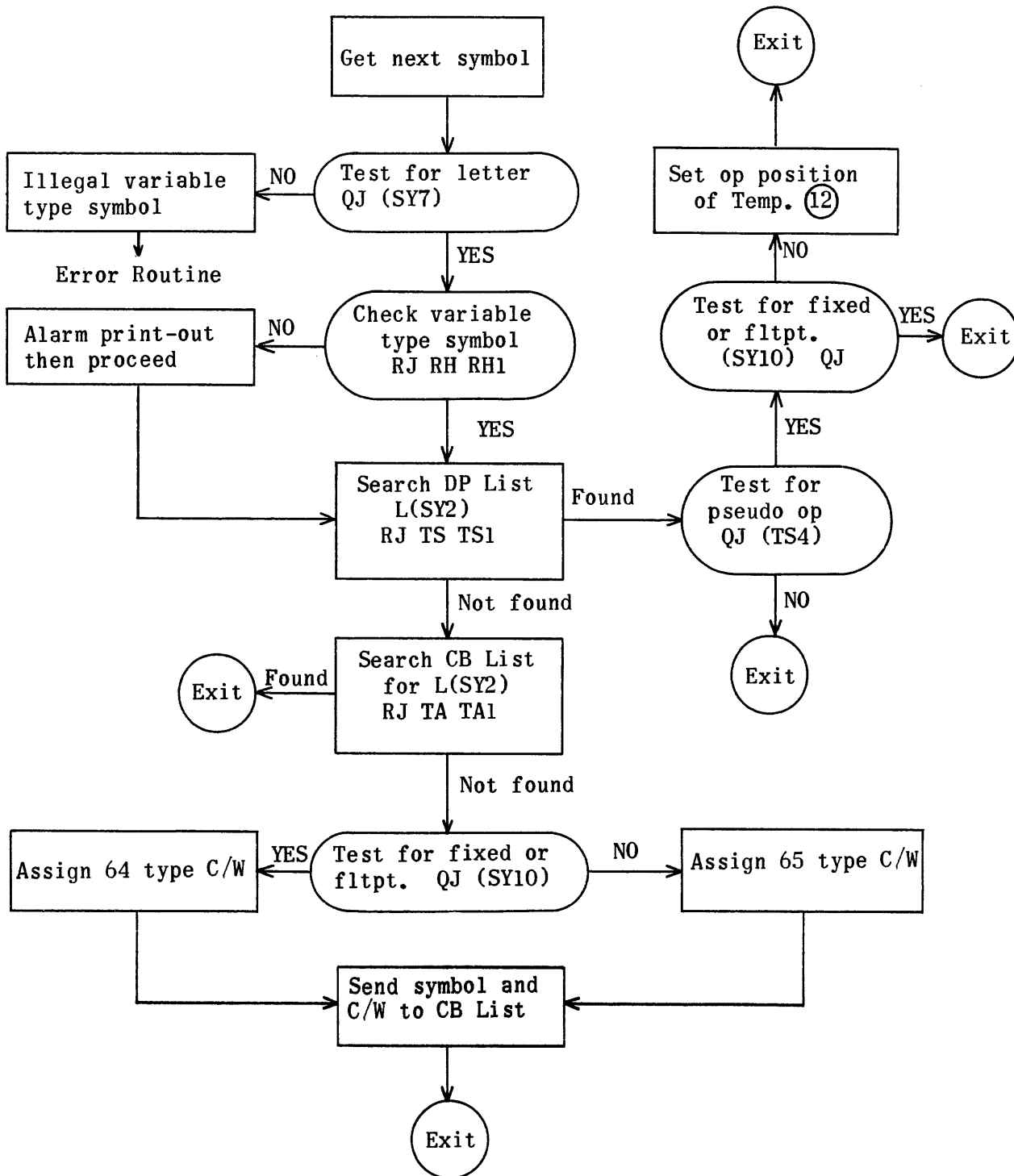
Error Print-Outs Used in VARY Translator

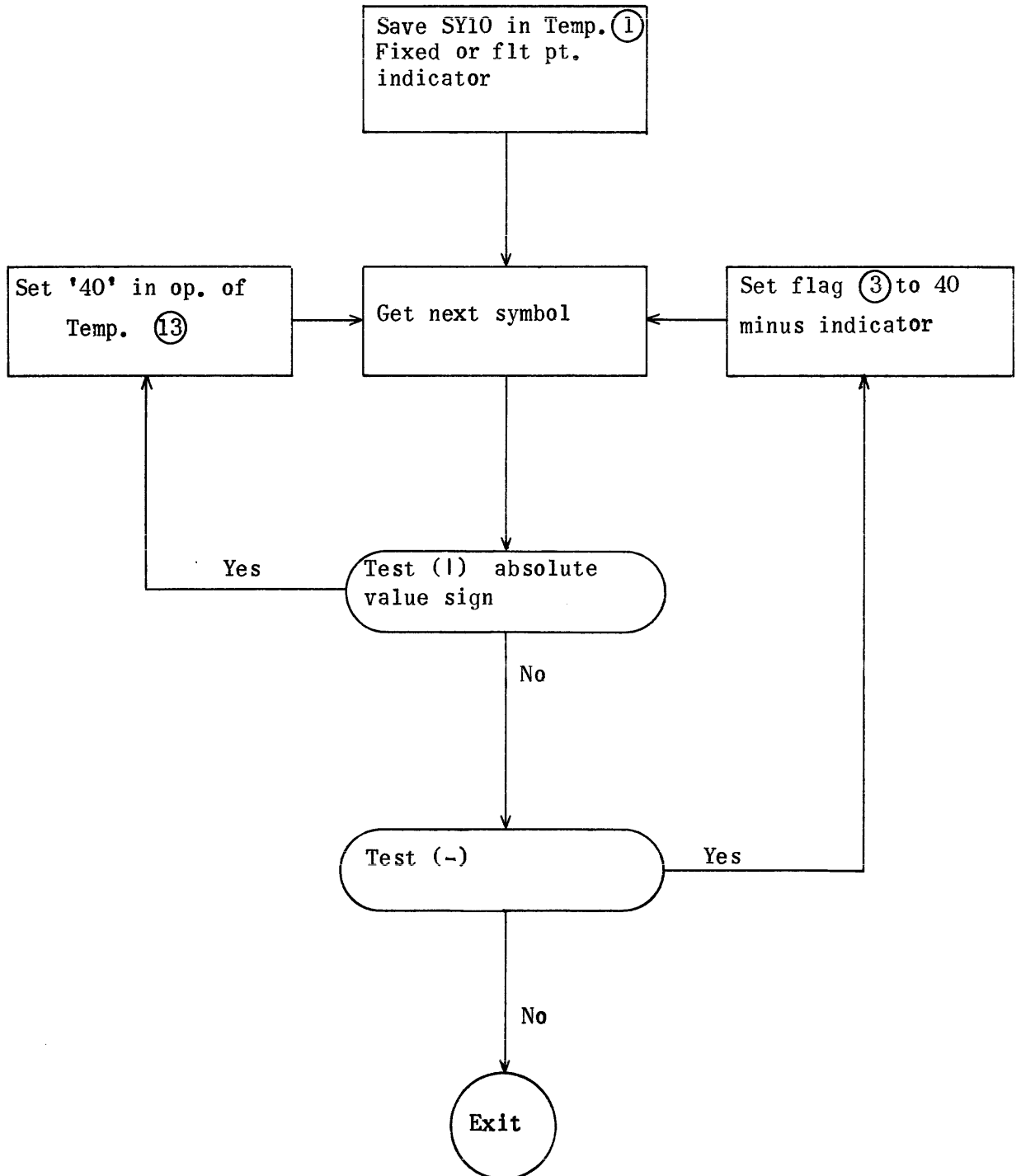
- ZA _____ is not a variable symbol.
- ZB Format error i.e., no parenthesis.
- ZC Inconsistent use of fixed or floating point operands.
- ZD Minus sign illegal prefix to variable.
- ZE Inconsistent use of absolute value signs.
- ZF _____type symbol is illegal in scientific notation format.
- ZG Illegal scientific notation format.
- ZH Change modifiers to eliminate infinite loop.
- ZI The number of modifier components exceed 15. Rest of sentence not checked.
- ZJ Misspelling or no referenced sentence.
- ZK No space period symbol.
- ZL Incomplete sentence i.e., no (range) component.
- ZM Variable _____ must not change values within VARY loop.
- ZN Error in transfer component.
- IN Disallowable character in exponent _____
- IQ { Absolute value of number too small - given zero value
 { Absolute value of number too large
- NV { Too many vary sentences in program.
 { VARY must not be last sentence in range of any VARY.
 { Range of inner VARY extends beyond range of outer VARY.

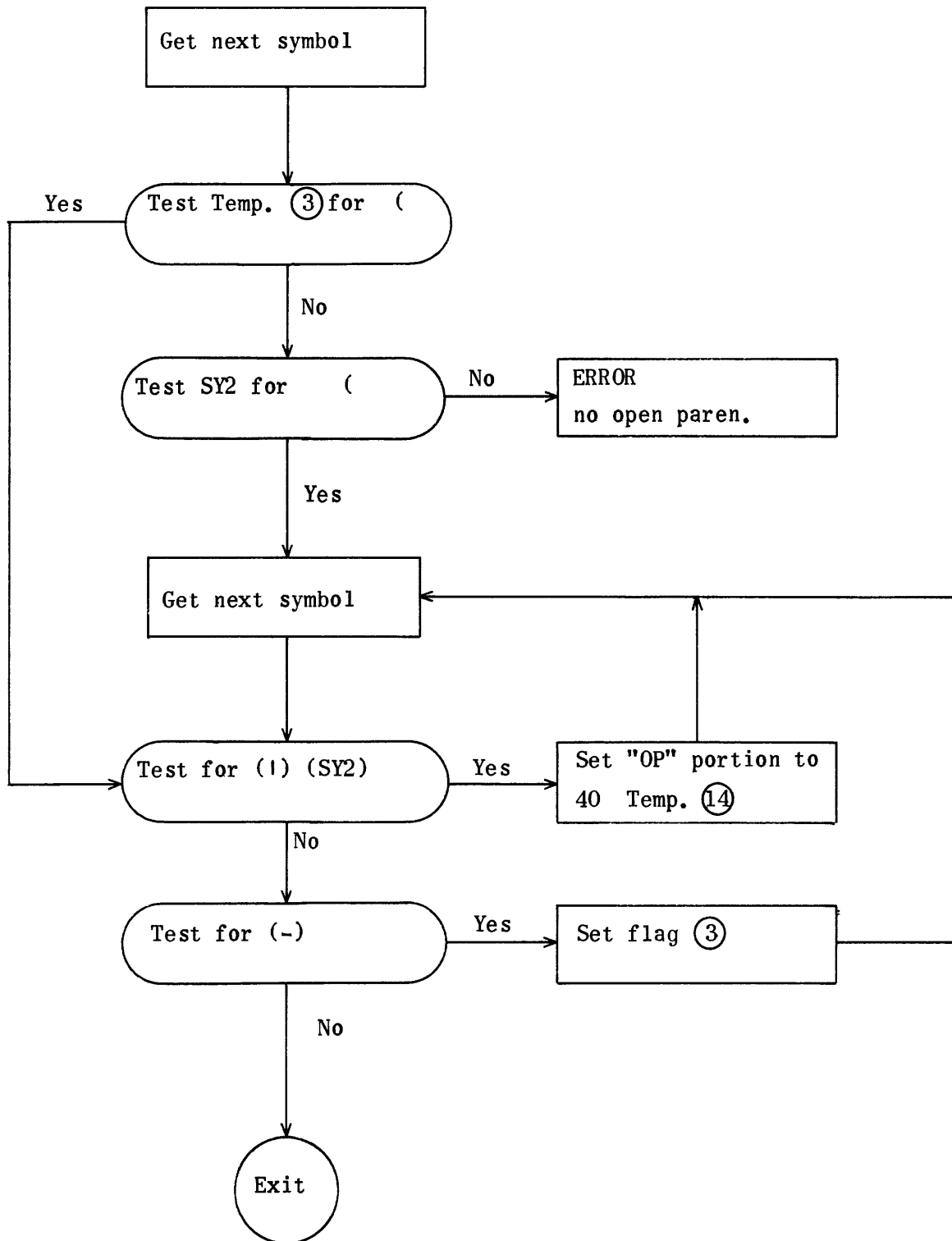
Vary Translator Flow Charts



α Region

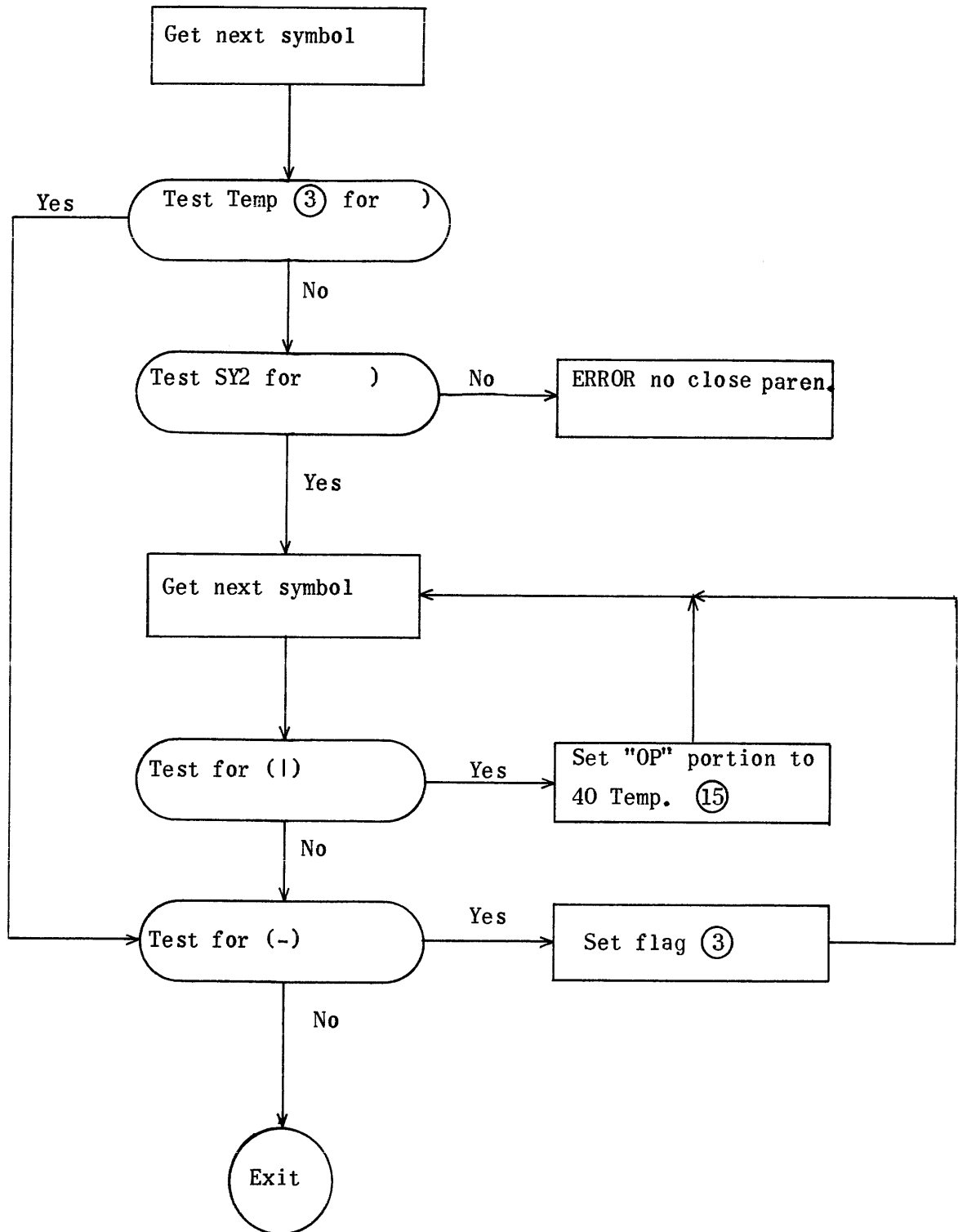


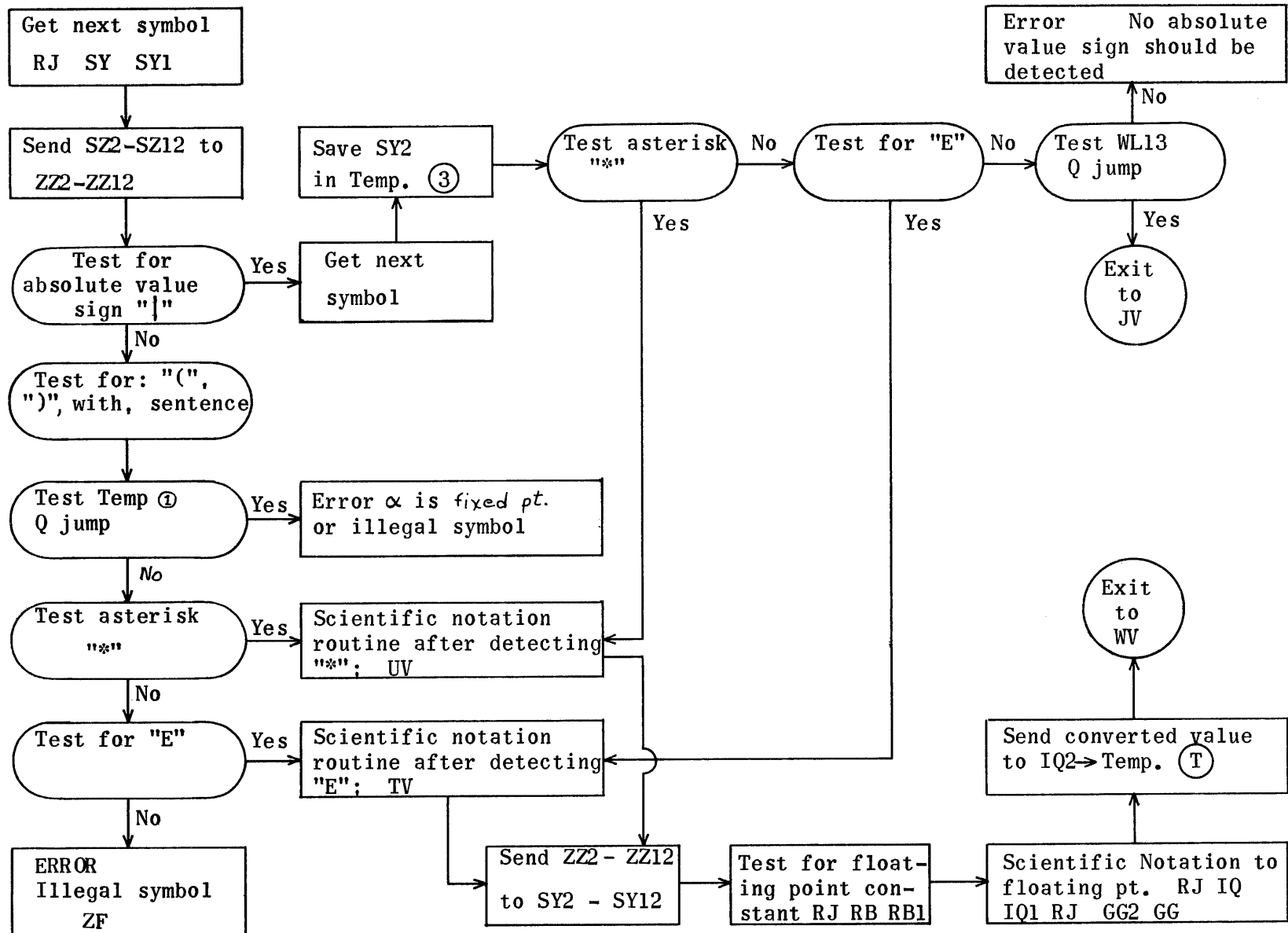




Bound Phase or Region 'C'

CV

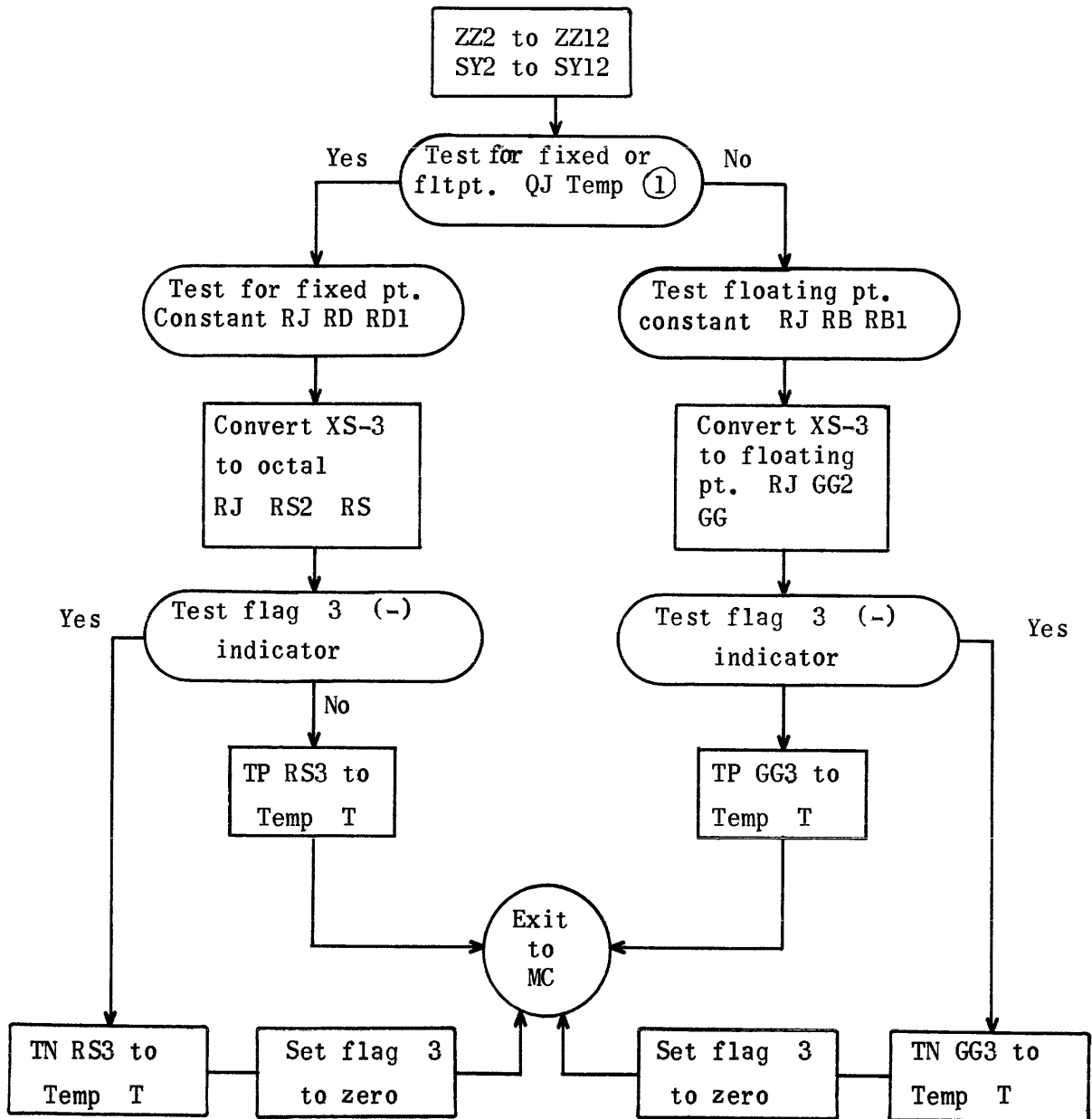


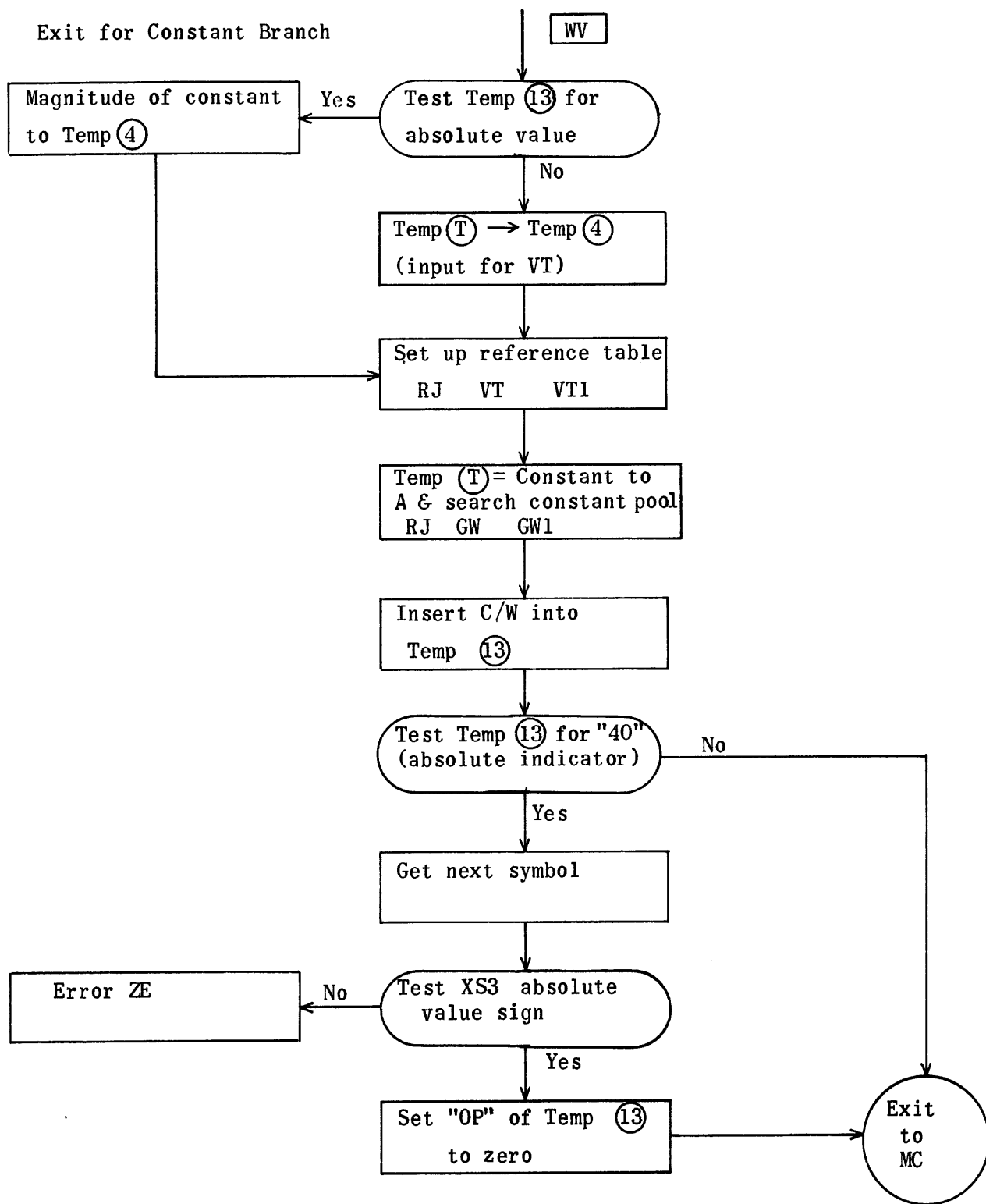


Vary Translator Flow Charts - HV-Constant Branch

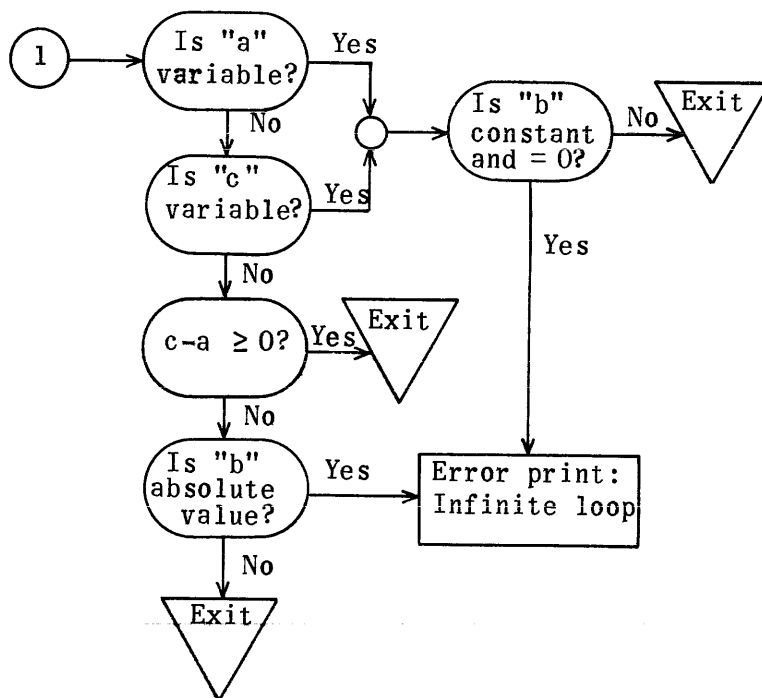
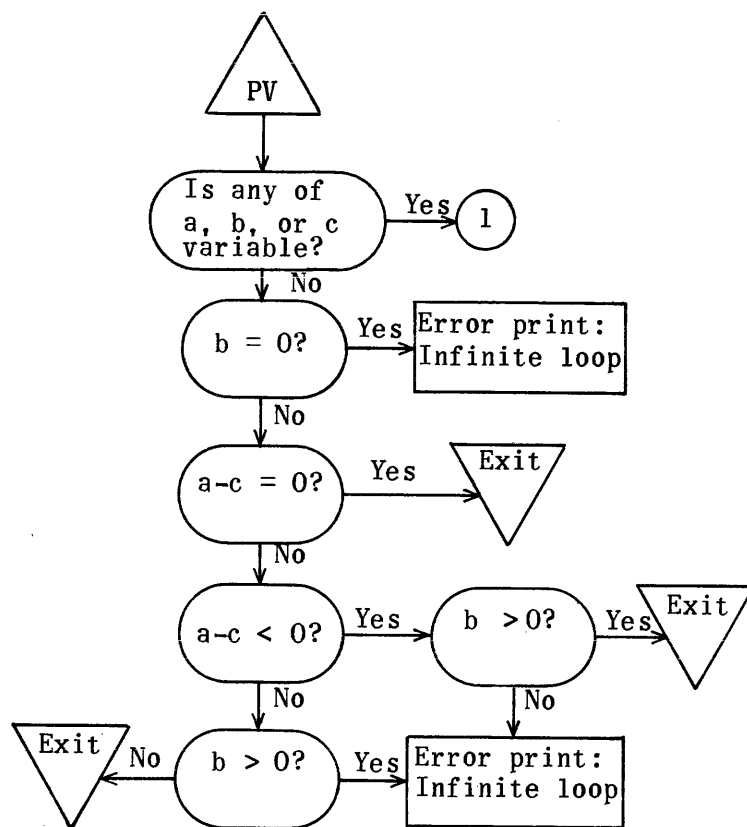
Fixed or floating point
constant branch

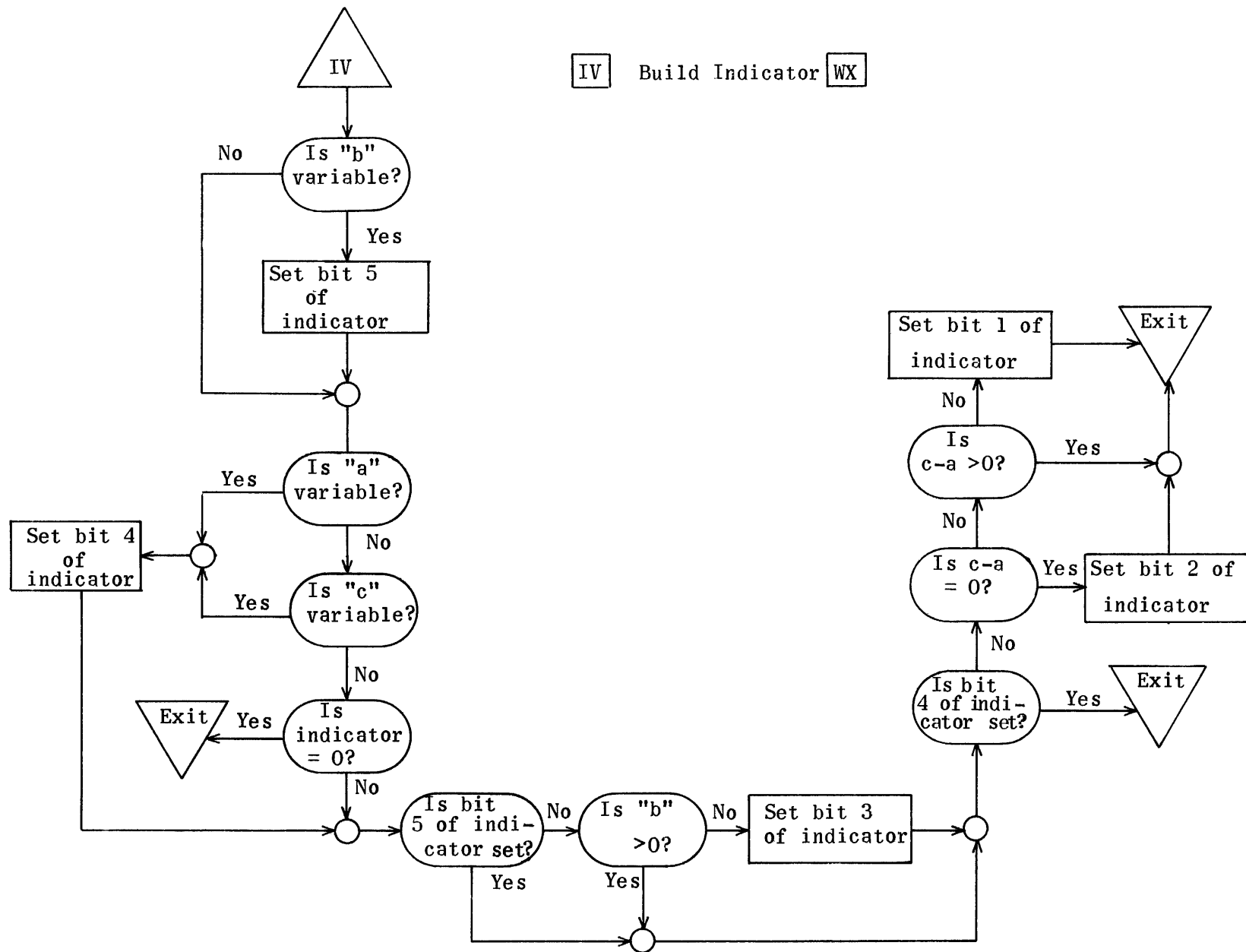
JV





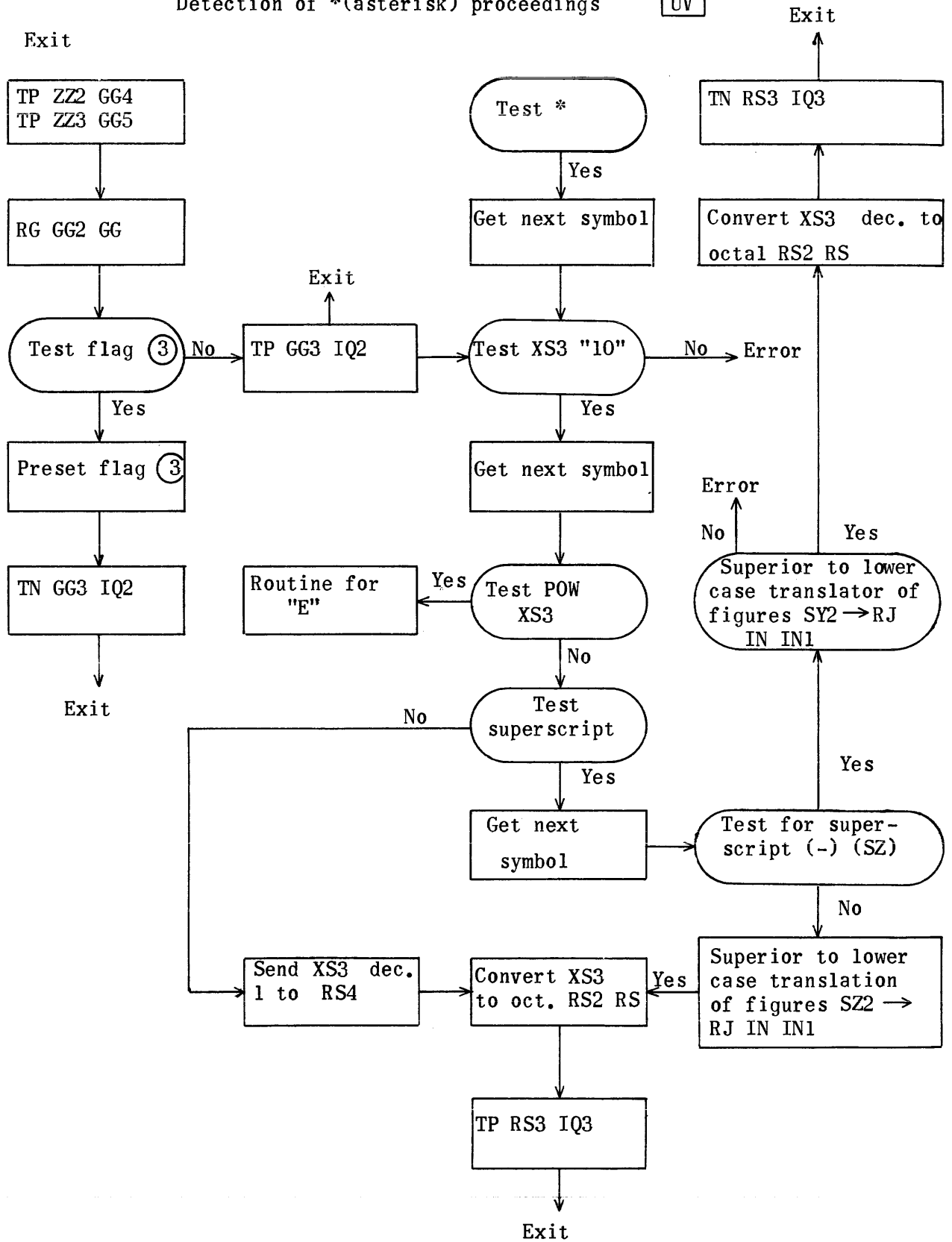
PV Preface for Indicator Region

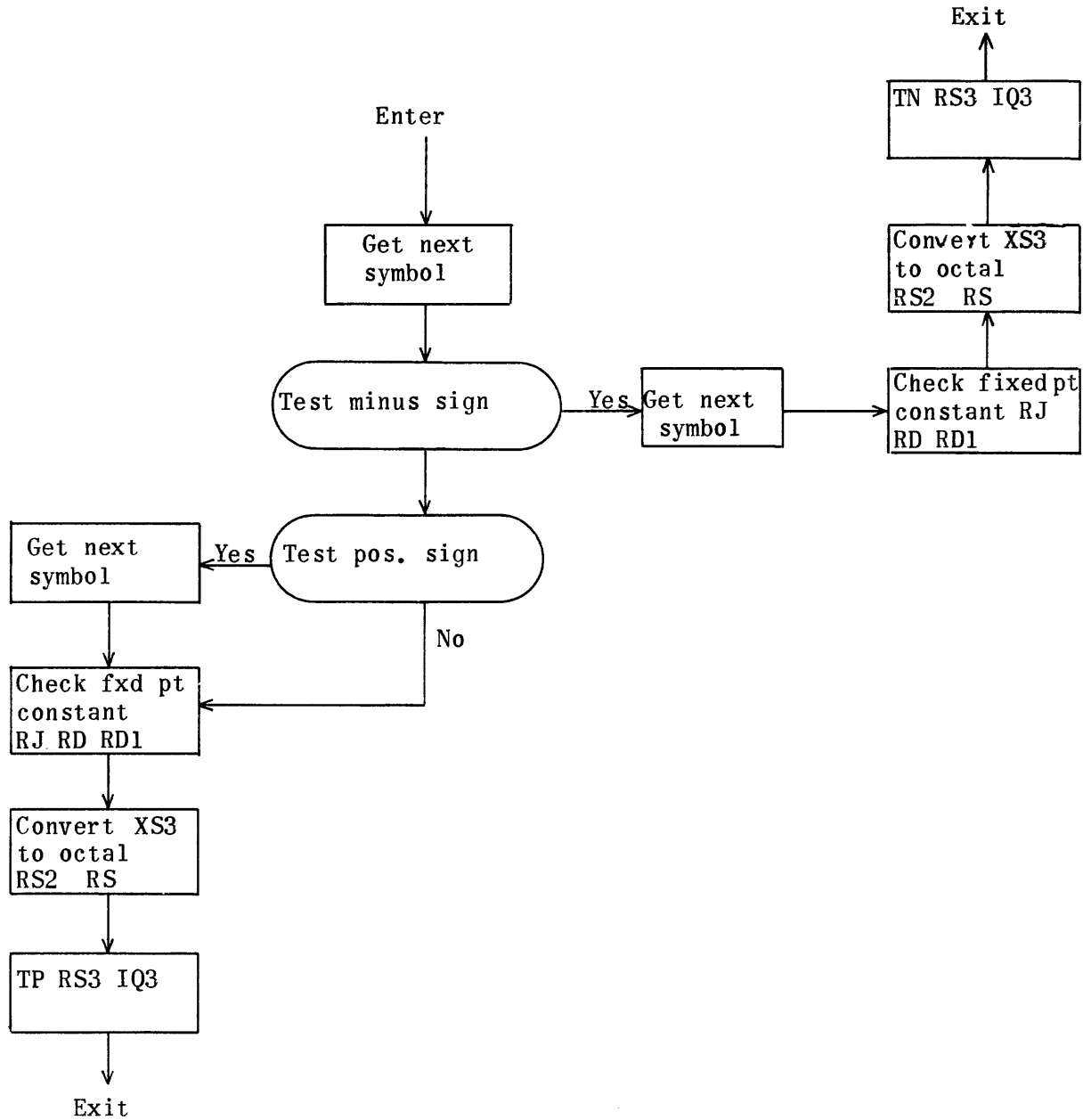


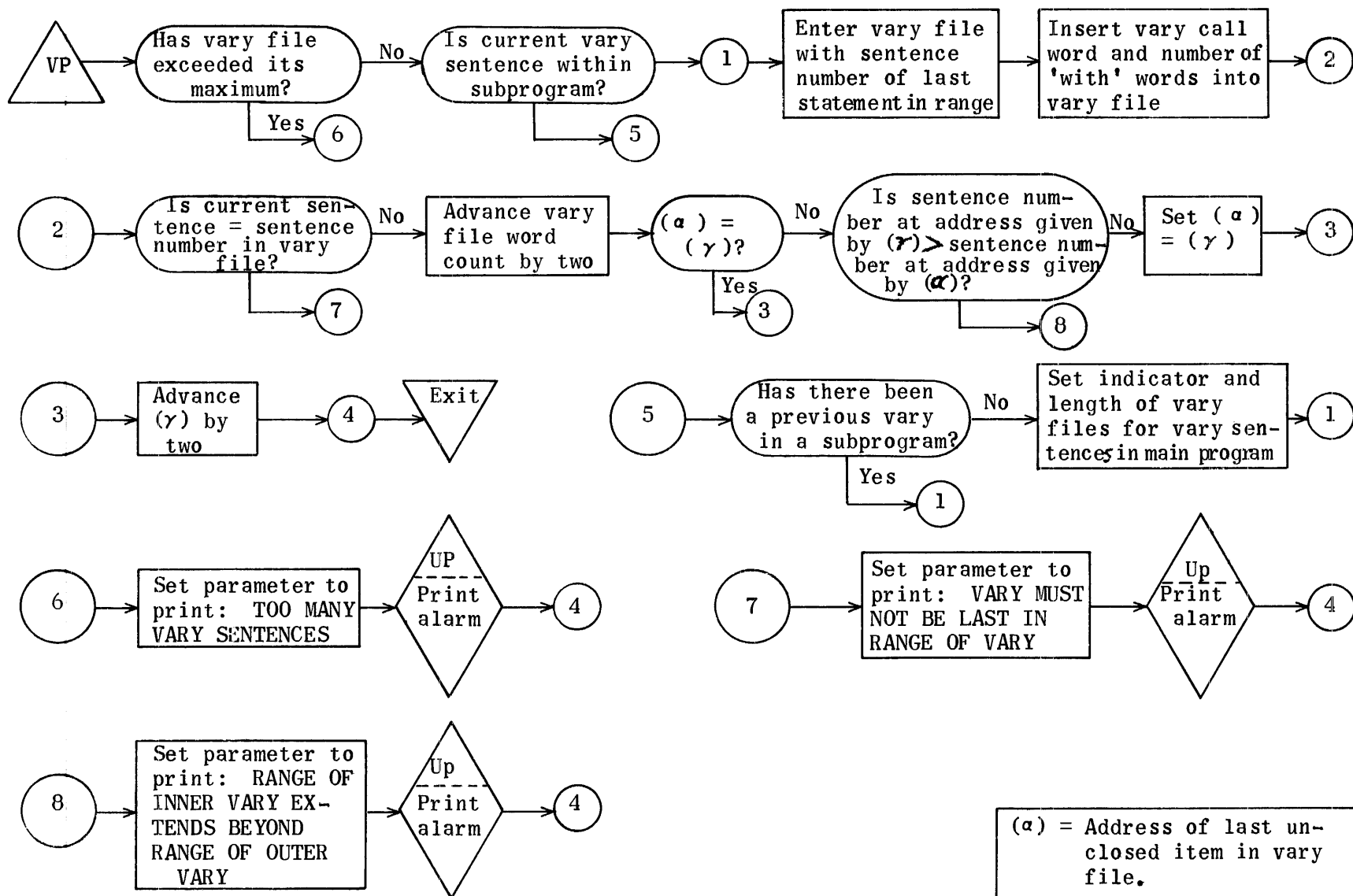


Detection of *(asterisk) proceedings

UV







Vary Translator Flow Charts - Build Vary File

Regions-VARY Translator

RE	MC4400
RE	LV4566
RE	VV4635
RE	FV4650
RE	BV4663
RE	CV4704
RE	GV4725
RE	HV5040
RE	JV5117
RE	WV5151
RE	DV5214
RE	PV5243
RE	IV5310
RE	SV5362
RE	VZ5417
RE	UV5433
RE	TV5465
RE	EV5506
RE	VT5521
RE	PL5553
RE	ID5613
RE	IM5743
RE	IN6057
RE	IO6107
RE	IQ6136
RE	IR6265
RE	ZA6277
RE	ZB6314
RE	ZC6330
RE	ZD6347
RE	ZE6364
RE	ZF6401
RE	ZG6423
RE	ZH6437
RE	ZI6455
RE	ZJ6500
RE	ZK6521
RE	ZL6533
RE	ZM6551
RE	VP6573
RE	VQ6633
RE	VS6647
RE	NV6655
RE	IS6711
RE	ZZ6714
RE	SC6726

RE	IF6727
RE	XV6730
RE	XU6735
RE	ZN6760

Translation Subroutine regions are also needed to assemble the VARY Translator tapes.

Also included, in the reco tapes of the VARY Translator are ID (130₈ lines in length), IM(114₈), IN(30₈), IO(27₈), IQ(127₈), IR(12₈), and IS (a set of 3 addresses used as temporaries). With one 2-line exception, this reco coding is identical to the regions similarly named in the IF String-Out where they are annotated and reproduced in this manual.

The 2-line exception is:

IA	ID3		
RJ	DV	DV1	Build Variable List File
MJ	0	MC40	Jump back to vary Translator Control
CA	ID5		

Master Control for Vary Translator

	IA	MC			
0	MJ	0	CT		
1	MJ	0	XV	Jump out for correction; then process	
2	RJ	VV	VV1	Test duplication of variable	
3	RJ	FV	FV1	First value region or 'a'	
4	TP	SY11	Q	Test for digit or decimal pt.	
5	QJ	MC6	MC12		
6	RJ	HV	HV1		
7	RJ	JV	JV1	Constant branch	a
10	RJ	WV	WV1		
11	MJ	0	MC13		
12	RJ	GV	GV1	Variable branch	
13	RA	PL10	PL2	Increase CTR+1 → CTR	
14	RJ	BV	BV1	Increment region or "b"	
15	TP	SY11	Q		
16	QJ	MC17	MC23		b
17	RJ	HV	HV1		
20	RJ	JV	JV1	Constant branch	
21	RJ	WV	WV1		
22	MJ	0	MC24		
23	RJ	GV	GV1	Variable branch	
24	RA	PL10	PL2		
25	RJ	CV	CV1	Bound phase or "c"	
26	TP	SY11	Q		c
27	QJ	MC30	MC34		
30	RJ	HV	HV1		
31	RJ	JV	JV1	Constant branch	
32	RJ	WV	WV1		
33	MJ	0	MC35		
34	RJ	GV	GV1	Variable branch	
35	RJ	DV	DV1	Build Variable File	
36	RJ	PV	PV1	Build preface region to indicator	
37	RJ	IV	IV1	Build indicator	
40	RJ	SV	SV1	Mover region	
41	TP	PL33	A	Possible WITH or SENTEN	
42	EJ	MC154	MC47	Test WITH	
43	EJ	MC152	ZL1	Test Δ. (Incomplete sentence)	
44	EJ	MC153	MC54	Test SENTEN	
45	RJ	SY	SY1	Get next symbol	
46	MJ	0	MC42	Go back & test for (WITH, Δ. , SENTEN)	
47	RA	WL4	PL1		
50	TJ	MC155	MC52	Test no. of WITH words ≤ 17 ₈	
51	MJ	0	ZI1		
52	RJ	VZ	VZ1	Presetting region	
53	MJ	0	MC1	Continue processing modifiers	
54	RJ	SY	SY1	Get next symbol which should be digit	
55	TP	SY11	Q	Is it digit? No → Error	
56	QJ	MC57	ZJ1		
57	TP	SY2	LN4		

60	RJ	LN2	LN	}	Put line number in proper form
61	TP	LN3	WL5		Line number → string-out
62	TP	LN3	A	}	
63	RJ	IX	IX1	}	Put line number in IZ list
64	RJ	SY	SY1		
65	EJ	MC146	MC70		Test THRU
66	EJ	MC147	MC70		Test THROUGH
67	MJ	O	MC157		Jump out to Test Δ .
70	RJ	SY	SY1		Look for sentence number
71	EJ	MC153	MC70		Test SENTENCE
72	TP	SY11	Q	}	
73	QJ	MC74	ZJ1	}	Test sentence number
74	TP	SY2	LN4	}	
75	RJ	LN2	LN	}	Put line number in proper form
76	MJ	O	MC163	}	→ TP LN3 WL6 insert
77	RJ	IX	IX1	}	Put line no. in IZ list
100	MJ	O	MC102		
101	TP	WL5	WL6		THRU - THROUGH, not detected
102	RJ	SY	SY1		
103	EJ	MC152	MC136		Test " Δ ."
104	EJ	MC151	MC122		Test "TO"
105	EJ	MC150	MC107		Test RESUME
106	MJ	O	XU10		Jump for correction
107	RJ	SY	SY1	}	
110	EJ	MC153	MC107	}	Test SENTENCE
111	TP	SY11	Q	}	
112	QJ	MC113	ZJ1	}	} Test for digit
113	TP	SY2	LN4	}	
114	RJ	LN2	LN	}	} Process line number
115	TP	LN3	WL7	}	Line number → string-out
116	TP	LN3	A	}	
117	RJ	IX	IX1	}	} Line number → list IZ
120	TP	PL4	WL10	}	
121	MJ	O	MC133		Jump to test for Δ .
122	RJ	SY	SY1		
123	EJ	MC153	MC122		Test SENTENCE
124	TP	SY11	Q	}	
125	QJ	MC126	ZJ1	}	Test for digit
126	TP	SY2	LN4	}	
127	RJ	LN2	LN	}	Process line number
130	TP	LN3	WL7	}	Line number → string-out
131	TP	LN3	A	}	
132	RJ	IX	IX1	}	Line number → list IZ
133	RJ	SY	SY1		
134	EJ	MC152	MC136		Test Δ .
135	MJ	O	ZK1		
136	RJ	VP	VP1		Build Vary File
137	RJ	VZ	VZ1		Presetting region
140	TV	SV11	PL30	}	WL _{n+1} → A
141	TP	PL30	A	}	WL _{n+1} - WL = n
142	ST	MC156	A	}	
143	AT	PL	WL	}	Put no. of words in string-out → WL

144	RJ	SS	SS1	
145	MJ	0	MC	
146	66	33546	77777	THRU
147	66	33545	16732	THROUGH
150	54	30656	74730	RESUME
151	66	51777	77777	TO
152	01	22777	77777	Δ .
153	65	30506	63050	SENTENCE
154	71	34663	37777	WITH
155	0	0	20	(20) ₈
156	0	0	WL	
157	EJ	MC152	MC161	} Test Δ .
160	MJ	0	MC101	
161	TP	WL5	WL6	
162	MJ	0	MC136	
163	TP	LN3	WL6	
164	TP	LN3	A	
165	MJ	0	MC77	
	CA	MC166		

Process α Region

	IA	LV	
0	MJ	0	30000
1	RJ	SY	SY1
2	TP	SY7	Q
3	QJ	LV4	ZA1
4	TP	SY2	VT20
5	RJ	RH	RH1
6	RJ	TS	TS1
7	MJ	0	LV23
10	TP	TS3	A
11	TJ	LV45	ZA1
12	TJ	LV46	LV14
13	MJ	0	ZA1
14	TP	TS3	SV22
15	TP	TS4	Q
16	QJ	LV17	LV
17	TP	SY10	Q
20	QJ	LV	LV21
21	RA	SV22	PL4
22	MJ	0	LV
23	RJ	TA	TA1
24	MJ	0	LV33
25	TP	TA4	A
26	TJ	LV45	ZA1
27	TJ	LV46	LV31
30	MJ	0	ZA1
31	TP	TA4	SV22
32	MJ	0	LV
33	TP	SY10	Q
34	QJ	LV35	LV40
35	RJ	TK	TK1
36	AT	PL14	SV22
37	MJ	0	LV42
40	RJ	TK	TK1
41	AT	PL15	SV22
42	TJ	LV45	ZA1
43	TJ	LV46	XU
44	MJ	0	ZA1
45	0	0	63000
46	0	0	67000
	CA	LV47	

		Test for 1st character, a letter
	$\alpha \rightarrow$ Temp α	
	Test variable type symbol	
	Search DP list	
	Not in list return	
	Test $62777 < CW \leq 66777$	
	C/W $\rightarrow$ Temp (12)	
	Test for Pseudo op	
	Test for fixed or fltpt.	
	Op. portion of Temp (12) to 40	
	Search Combination List	
	Not in list return	
	Test $62777 < C/W \leq 66777$	
	C/W of symbol in list $\rightarrow$ Temp (12)	
	Test for fixed or fltpt.	
	Sequence number $\rightarrow$ A	
	Fixed pt.; 64000 C/W $\rightarrow$ Temp (12)	
	Sequence number $\rightarrow$ A	
	65000 C/W $\rightarrow$ Temp (12)	
	Jump out for correction	

Search Variable List "VL" for Modifiers Previously Mentioned

	IA	VV		
0	MJ	0	30000	
1	SP	VL	0	$VL_{n+1} \rightarrow A$
2	SS	VV11	0	$-VL1 = n = \# \text{ words} - 1$
3	SA	VV12	0	Add J value
4	TU	A	VV6	
5	SP	VT20	0	Variable $\rightarrow A$
6	RP	[30000]	VV } ZM1 }	Search list VL
7	EJ	VL1	ZM1 }	
10	MJ	0	VV	
11	0	VL1	0	
12	0	20000	0	
	CA	VV13		

First Value "a" Region

	IA	FV		
0	MJ	0	30000	
1	TP	SY10	PL31	Set Temp ① indicator for fixed or floating point
2	RJ	SY	SY1	
3	EJ	PL16	FV5	Test (-)
4	MJ	0	FV7	
5	TP	PL4	PL23	$L(40,0,0) \rightarrow \text{flag } ③$
6	MJ	0	FV2	
7	EJ	PL13	FV11	Test ()
10	MJ	0	FV	Exit to Control (MC)
11	RA	SV23	PL4	Set Op. of Temp ⑬ to '40'
12	MJ	0	FV2	
	CA	FV13		

Increment Region "b"

	IA	BV		
0	MJ	0	30000	Exit
1	RJ	SY	SY1	
2	TP	PL33	A	} Temp ③ $\in (() \rightarrow A$
3	EJ	PL17	BV10	} Test (()
4	TP	SY2	A	} Test
5	EJ	PL17	BV7	} (()
6	MJ	0	ZB1	Error print and exit
7	RJ	SY	SY1	
10	TP	SY2	A	} Test
11	EJ	PL13	BV13	} ()
12	MJ	0	BV15	
13	RA	SV24	PL4	Set Op. of Temp ⑭ to '40'
14	MJ	0	BV7	
15	EJ	PL16	BV17	Test for (-)
16	MJ	0	BV	
17	TP	PL4	PL23	Set Op. of flag ③ to '40' minus ind.
20	MJ	0	BV7	
	CA	BV21		

Bound Phase or Region "c"

	IA	CV		
0	MJ	0	30000	
1	RJ	SY	SY1	Pick up)
2	TP	PL33	A	} Temp ③ $\in () \rightarrow A$
3	EJ	PL20	CV10	} Test ())
4	TP	SY2	A	} Test ())
5	EJ	PL20	CV7	}
6	MJ	0	ZB1	Error print and exit
7	RJ	SY	SY1	④
10	TP	SY2	A	} Test ()
11	EJ	PL13	CV13	}
12	MJ	0	CV15	
13	RA	SV25	PL4	Set Op. of Temp ⑮ to "40"
14	MJ	0	CV7	
15	EJ	PL16	CV17	Test for (-)
16	MJ	0	CV	
17	TP	PL4	PL23	Set Op. of flag ③ to "40" minus ind.
20	MJ	0	CV7	
	CA	CV21		

Variable Branch

0	IA	GV		
1	MJ	0	30000	
2	TP	PL4	PL22	Set flag ② for VT to '40'
3	TP	PL31	A	Temp ① = SY10 α → A
4	EJ	SY10	GV56	Test consistency of fixed or fltpt.
5	MJ	0	ZC1	Error print
6	RJ	TS	TS1	Search DP list
7	MJ	0	GV13	Not in list - go search CB
10	RA	GV10	PL10	
11	RA	[SV23]	TS3	C/W → Temp (13,14,15)
12	TP	GV46	GV10	Preset GV10
13	MJ	0	XU21	Go to build table
14	RJ	TA	TA1	Search CB list
15	MJ	0	GV21	Not in list - go assign C/W
16	RA	GV16	PL10	
17	RA	[SV23]	TA4	C/W → Temp (13,14,15)
20	TP	GV47	GV16	<u>Preset GV16</u>
21	MJ	0	XU21	Go build table
22	TP	PL31	Q	Temp ① = SY10 α → Q
23	QJ	GV23	GV34	Test for fixed or fltpt. variable
24	RA	GV27	PL10	Modifiers
25	RA	GV31	PL10	} Fxd. pt.
26	RJ	TK	TK1	Obtain proper sequence # → A
27	TP	A	PL30	A → Temp ①
30	RA	[SV23]	PL30	<u>Preset GV27</u>
31	RA	[SV23]	GV54	Temp (13,14,15) + 64000
32	TP	GV51	GV31	<u>Preset GV31</u>
33	MJ	0	GV44	Go build table
34	RA	GV40	PL10	} Modifiers
35	RA	GV42	PL10	
36	RJ	TK	TK1	
37	TP	A	PL30	
40	RA	[SV23]	PL30	
41	TP	GV50	GV40	<u>Preset GV40</u>
42	RA	SV23	GV55	Temp (13,14,15) + 65000
43	TP	GV52	GV42	Preset GV42
44	RJ	VT	XU13	Build table, but make correction first
45	MJ	0	GV76	
46	RA	SV23	TS3	} Presetters
47	RA	SV23	TA4	
50	RA	SV23	PL30	
51	RA	SV23	GV54	
52	RA	SV23	GV55	
53	0	0	0	
54	0	0	64000	
55	0	0	65000	
56	TP	PL23	Q	Flag ③ → Q (minus sign indicator)
57	QJ	GV60	GV62	
60	TP	PL	PL23	Reset flag ③ to 0

61	MJ	0	ZD1	Error print
62	TP	SY7	Q	Test
63	QJ	GV64	ZA1	Error for first character a letter
64	RJ	RH	RH1	Test for variable type symbol
65	MJ	0	GV5	
66	RA	GV67	PL10	
67	TP	[SV23]	Q	
70	TP	GV75	GV67	
71	QJ	GV72	GV	Test for opening absolute value sign
72	RJ	SY	SY1	Detected
73	EJ	PL13	GV	Test for closing absolute value sign
74	MJ	0	ZE1	Error print; inconsistent use of absolute
75	TP	SV23	Q	Presetter
76	RA	GV77	PL10	
77	TP	SV23	A	
100	TP	GV106	GV77	
101	MJ	0	GV107	
102	TJ	GV105	GV66	
103	MJ	0	ZA1	
104	0	0	63000	
105	0	0	67000	
106	TP	SV23	A	
107	TP	UV31	Q	L(0,77777,77777) → Q
110	QT	A	A	
111	TJ	GV104	ZA1	
112	MJ	0	GV102	
	CA	GV113		

Constant Branch

0	IA	HV		
1	MJ	0	[30000]	
2	RJ	SY	SY1	Senten: 1
3	TP	30011	HV4	
4	EJ	PL13	ZZ2	Save this information
5	MJ	0	HV17	Test () (absolute sign)
6	RJ	SY	SY1	No. () detected
7	MJ	0	HV45	(
10	EJ	PL6	TV1	Test (*) → set up routine
11	RA	HV12	PL10	Test (E) → set up routine
12	TP	[SV23]	Q	
13	TP	HV41	HV12	Preset HV12
14	QJ	HV16	HV15	
15	MJ	0	ZE1	Error print and exit
16	MJ	0	HV	Go to JV via MC
17	EJ	PL17	HV22	Test ((
20	EJ	PL20	HV22	Test ())
21	MJ	0	HV42	Jump out for correction
22	TP	SY2	PL33	((or)) → Temp ③
23	MJ	0	HV	Go to JV via MC
24	TP	PL31	Q	Temp ① → Q
25	QJ	HV26	HV55	Test fixed or floating point
26	MJ	0	ZC1	Error print and exit
27	EJ	PL7	UV1	Test (*) → set up routine
30	EJ	PL6	TV1	Test (E) → set up routine
31	MJ	0	ZF1	Error print and exit
32	RP	30011	HV34	
33	TP	ZZ2	SY2	
34	RJ	RB	RB1	
35	RJ	IQ	IQ1	
36	TP	IQ2	PL30	Send value to temp ①
37	RA	HV	PL1	Set up for return to WV
40	MJ	0	HV	
41	TP	SV23	Q	Presetter for HV12
42	EJ	MC154	HV22	Test WITH
43	EJ	MC153	HV22	Test SENTEN
44	MJ	0	HV50	
45	TP	SY2	PL33	
46	EJ	PL7	UV1	
47	MJ	0	HV10	
50	RA	HV51	PL10	No presetter
51	TP	[SV23]	Q	Test for previous mention of absolute value sign
52	TP	HV54	HV51	
53	QJ	ZE1	HV24	
54	TP	SV23	Q	
55	TP	SY2	A	
56	MJ	0	HV27	
	CA	HV57		

Fixed or Floating Point Constant Branch

0	IA	JV					
	MJ	0	30000				
1	RP	30011	JV3				
2	TP	ZZ2	SY2				
3	TP	PL31	Q	Fixed or fltpt. indicator → Q			
4	QJ	JV5	JV17				
5	RJ	RD	RD1	Fixed-pt. test			
6	TP	SY2	RS4	XS-3 to octal conversion			
7	RJ	RS2	RS				
10	TP	PL23	Q	Test flag ③ for (-) indication			
11	QJ	JV12	JV15				
12	TN	RS3	PL30	Negative value → Temp ①			
13	TP	PL	PL23	0 → flag ③			
14	MJ	0	JV	Exit to WV via MC			
15	TP	RS3	PL30	Positive value → Temp ①			
16	MJ	0	JV	Exit to WV via MC			
17	RJ	RB	RB1				
20	TP	SY2	GG4				
21	TP	SY3	GG5				
22	RJ	GG2	GG				
23	TP	PL23	Q				
24	QJ	JV25	JV30				
25	TN	GG3	PL30	Negative value → Temp ①			
26	TP	PL	PL23				
27	MJ	0	JV	Exit to WV via MC			
30	TP	GG3	PL30	Positive value → Temp ①			
31	MJ	0	JV	Exit to WV via MC			
	CA	JV32					

Fixed
Pt.

Floating
Pt.

Exit for Constant Branch

IA		WV	
0	MJ	0	30000
1	RA	WV2	PL10
2	TP	[SV23]	Q
3	TP	WV36	WV2
4	QJ	WV5	WV7
5	TM	PL30	PL34
6	MJ	0	WV10
7	TP	PL30	PL34
10	RJ	VT	VT1
11	TP	PL34	A
12	RJ	GW	GW1
13	RA	WV14	PL10
14	RA	[SV23]	Q
15	TP	WV37	WV14
16	TP	A	Q
17	QJ	WV20	WV
20	TP	WV35	Q
21	RA	WV22	PL10
22	QT	[SV23]	PL30
23	TP	WV40	WV22
24	TP	PL10	Q
25	LQ	Q	25
26	RA	WV27	Q
27	TP	PL	[SV23]
30	TP	WV41	WV27
31	RA	WV32	PL10
32	RA	[SV23]	PL30
33	TP	WV42	WV32
34	MJ	0	WV
35	37	77777	77777
36	TP	SV23	Q
37	RA	SV23	Q
40	QT	SV23	PL30
41	TP	PL	SV23
42	RA	SV23	PL30
	CA	WV43	

} Set up for (|) detection
 } Magnitude of constant → Temp ④
 } Constant → Temp ④
 } Set up reference table
 } Search and assign C/W from constant pool
 } C/W → Temp ⑬
 } Test for '40' indicator
 } L(3777....) → Q
 } Mask C/W → Temp ①
 } Shift counter PL10 to V
 } Clear cell (set Op. to 0)
 } C/W → Temp ⑬

Build Variable List

0	IA	DV		
	MJ	0	30000	
1	TV	VL	DV2	Counter $\rightarrow$ DV2 _v
2	TP	VT20	[30000]	$\alpha \rightarrow VL_n ; n \geq 1$
3	RA	VL	PL3	Increase counter VL by (0,1,1)
4	TP	VT25	Q	Test "b" for variable
5	QJ	DV12	DV6	
6	TV	VL	DV7	
7	TP	PL	[30000]	No variable. $\therefore 0 \rightarrow$ VL list
10	RA	VL	PL3	Increase Counter VL by (0,1,1) to test "c"
11	MJ	0	DV15	
12	TV	VL	DV13	
13	TP	VT22	[30000]	Insert variable in list
14	RA	VL	PL3	Increase counter VL by (0,1,1)
15	TP	VT26	Q	Test "c" for variable
16	QJ	DV23	DV17	
17	TV	VL	DV20	
20	TP	PL	[30000]	No variable. $\therefore 0 \rightarrow$ VL list
21	RA	VL	PL3	
22	MJ	0	DV	To preface region PV via MC
23	TV	VL	DV24	
24	TP	VT23	[30000]	Insert variable in list
25	RA	VL	PL3	
26	MJ	0	DV	Exit to preface region PV via MC
	CA	DV27		

Preface for Indicator Region

	IA	PV		
0	MJ	0	30000	
Test 1	TP	VT24	Q	$a' \rightarrow Q$
2.	2	QJ	PV21	PV3
	3	TP	VT25	Q
	4	QJ	PV21	PV5
	5	TP	VT26	Q
	6	QJ	PV21	PV7
	7	TP	VT22	A
	10	ZJ	PV11	ZH1
	11	TP	VT21	A
	12	ST	VT23	A
	13	ZJ	PV14	PV
	14	SJ	PV15	PV17
	15	TP	VT22	A
	16	SJ	ZH1	PV
	17	TP	VT22	A
	20	SJ	PV	ZH1
①	21	TP	VT24	Q
	22	QJ	PV35	PV23
	23	TP	VT26	Q
	24	QJ	PV35	PV25
	25	TP	VT23	A
	26	ST	VT21	A
	27	ZJ	PV30	PV
	30	SJ	PV31	PV33
	31	TP	SV24	Q
	32	QJ	ZH1	PV33
	33	MJ	0	PV
	34	0	0	0
Test 35	TP	VT25	Q	
4	36	QJ	PV	PV37
	37	TP	VT24	Q
	40	QJ	PV43	PV41
	41	TP	VT26	Q
	42	QJ	PV43	PV
	43	TP	VT22	A
	44	ZJ	PV	ZH1
	CA	PV45		

	$a' \rightarrow Q$
	PV21 = ①
	$b' \rightarrow Q$
	PV21 = ①
	$C' \rightarrow Q$
	PV21 = ①
	Test b = 0
	Error print infinite loop designed
	$a \rightarrow A$
	$a-c \rightarrow A$
	$0 \rightarrow A$
	Test b > 0; if not, error print,
	Infinite loop designed
	Test b < 0
	Error print, infinite loop designed
	$a' \rightarrow Q$
	Go to test 4
	$c' \rightarrow Q$
	$c' \rightarrow A$
	$c-a \rightarrow A$
	Test c-a > 0
	Test Temp ①④ ∈ b for absolute value
	Error print, infinite loop
	Test b', if variable, then exit
	Test b = 0. If so,
	error print, infinite loop designed

Build Indicator WX

	0	IA	IV		
	0	MJ	0	30000	
	1	TP	VT25	Q	} Test b'
	2	QJ	IV3	IV4	
	3	RA	SV21	IV47	1 → B5
	4	TP	VT24	Q	} Test a'
	5	QJ	IV6	IV10	
	6	RA	SV21	IV46	1 → B4
	7	MJ	0	IV12	
	10	TP	VT26	Q	} Test c'
	11	QJ	IV6	IV50	
Constant	12	TP	SV21	Q	} Test bit 5 = 1
	13	LQ	Q	20	
	14	QJ	IV22	IV15	} Test b > 0
	15	TP	IV42	A	
	16	TJ	VT22	IV22	} Test b < 0
	17	TP	VT22	Q	
	20	QJ	IV21	IV22	} 1 → B3
	21	RA	SV21	IV45	
	22	TP	SV21	Q	} Test bit 4 = 1
	23	LQ	Q	21	
	24	QJ	IV	IV25	} Text c-a = 0
	25	TP	VT23	A	
	26	ST	VT21	A	} Test c-a > 0
	27	ZJ	IV32	IV30	
	30	RA	SV21	IV44	1 → B2
	31	MJ	0	IV	
	32	TP	SV21	Q	} Test bit 4 = 1
	33	LQ	Q	21	
	34	QJ	IV	IV35	} Test c-a > 0
	35	TP	VT23	A	
	36	ST	VT21	A	} 1 → B1
	37	SJ	IV40	IV	
	40	RA	SV21	IV43	
	41	MJ	0	IV	
	42	0	0	0	
	43	0	1	0	
	44	0	2	0	
	45	0	4	0	
	46	0	10	0	
	47	0	20	0	
	50	TP	SV21	A	
	51	ZJ	IV12	IV	
		CA	IV52		

Mover Region

0	IA	SV		
0	MJ	0	30000	
1	TP	[SV22]	A	(OP, 0, C/W) → A
2	LA	A	17	(OP, 0, C/W) → (0, C/W, 0) → A
3	TU	A	[SV22]	(OP, C/W, C/W) → Temp (12)
4	RA	SV1	PL2	
5	RA	SV3	PL1	
6	IJ	SV16	SV1	Reset SV16
7	TP	SV26	SV1	
10	TP	SV27	SV3	
11	TP	[SV21]	[WL11]	Move Temp → WL
12	RA	SV11	PL3	
13	IJ	SV17	SV11	
14	TU	SV20	SV11	
15	MJ	0	SV30	
16	0	0	3	Index
17	0	0	4	Index
20	0	SV21	0	
21	0	0	0	Temp (11)
22	0	0	0	Temp (12)
23	0	0	0	Temp (13)
24	0	0	0	Temp (14)
25	0	0	0	Temp (15)
26	TP	SV22	A	
27	TU	A	SV22	
30	TP	SV33	SV16	Reset Index
31	TP	SV34	SV17	
32	MJ	0	SV	
33	0	0	3	
34	0	0	4	
	CA	SV35		

Presetter Region for Re-entry to LV

	IA	VZ		
0	MJ	0	30000	
1	TP	PL	PL10	0 → CTR
2	TP	PL	PL11	0 → CTR1
3	RP	10003	VZ5	0 → flags
4	TP	PL	PL21	
5	RP	10007	VZ7	0 → storage
6	TP	PL	VT20	
7	RP	10005	VZ11	0 → Temps ⑪ - ⑮
10	TP	PL	SV21	
11	RP	10006	VZ13	
12	TP	PL	PL30	
13	MJ	0	VZ	
	CA	VZ14		

Subroutine for Detection of Asterisk*

	IA	UV		
0	MJ	0	EV1	Jump to Exit
1	RJ	SY	SY1	
2	EJ	PL25	UV4	Test for XS-3 '10'
3	MJ	0	ZG1	Error print
4	RJ	SY	SY1	
5	EJ	PL5	TV1	Test for XS-3 POW
6	TP	SY13	Q	Test
7	QJ	UV10	UV21	for superscript
10	RJ	SY	SY1	
11	TP	SZ2	A	
12	EJ	UV31	UV21	Test for superscript (-)
13	TP	SZ2	IN2	
14	RJ	IN	IN1	
15	TP	IN2	RS4	No superscript (-)
16	RJ	RS2	RS	
17	TP	RS3	IQ3	
20	MJ	0	UV	
21	TP	SY2	IN2	
22	RJ	IN	IN1	
23	TP	IN2	RS4	
24	RJ	RS2	RS	
25	TN	TS3	IQ3	
26	MJ	0	UV	
27	TP	PL26	RS4	XS-3 (1) → RS4
30	MJ	0	UV16	
31	00	77777	77777	Superscript (-)
	CA	UV32		

Subroutine for Detection of "E"

	IA	TV		
0	MJ	0	EV1	Exit
1	RJ	SY	SY1	
2	EJ	PL16	TV13	Test (-)
3	EJ	PL36	TV5	Test (+)
4	MJ	0	TV6	
5	RJ	SY	SY1	
6	RJ	RD	RD1	Check fixed pt. constant
7	TP	SY2	RS4	Decimal
10	RJ	RS2	RS	} to octal conversion
11	TP	RS3	IQ3	
12	MJ	0	TV	
13	RJ	SY	SY1	
14	RJ	RD	RD1	Check fixed pt. constant
15	TP	SY2	RS4	Decimal
16	RJ	RS2	RS	} to octal conversion
17	TN	RS3	IQ3	
20	MJ	0	TV	
	CA	TV21		

Exit for Scientific Notation Set-up Routine

	IA	EV	
0	MJ	0	HV32
1	TP	ZZ2	GG4
2	TP	ZZ3	GG5
3	RJ	GG2	GG
4	TP	PL23	Q
5	QJ	EV6	EV11
6	TP	PL	PL23
7	TN	GG3	IQ2
10	MJ	0	EV
11	TP	GG3	IQ2
12	MJ	0	EV
	CA	EV13	

Test flag ③
Reset flag ③

Set Up Reference Table Used by VL, "WX", and Preface

	IA	VT		
0	MJ	0	[30000]	Exit
1	RA	PL11	PL1	Entrance. CTR1 + 1 → CTR1
2	TP	PL22	Q	Flag ② → Q
3	QJ	VT10	VT4	Test for variable (-)
4	RA	VT5	PL11	Constant VT20 = Beginning of table Preset VT5
5	TP	PL34	[VT20]	
6	TP	VT27	VT5	
7	MJ	0	VT16	Variable L(40,0,0) → Table Preset VT11
10	RA	VT11	PL11	
11	TP	PL4	[VT23]	
12	TP	VT30	VT11	Symbol → Table Preset VT14
13	RA	VT14	PL11	
14	TP	SY2	[VT20]	
15	TP	VT31	VT14	Preset VT14
16	TP	PL	PL22	Reset flag ② to 0
17	MJ	0	VT	
Table 20	0	0	0	α, the variable
21	0	0	0	a, the lower limit
22	0	0	0	b, the incrementing value
23	0	0	0	c, the upper limit
24	0	0	0	a' }
25	0	0	0	b' }
26	0	0	0	c' }
27	TP	PL34	VT20	(-) indicates Variable
30	TP	PL4	VT23	
31	TP	SY2	VT20	
	CA	VT32		

Constant Pool

	IA	PL		
0	0	0	0	
1	0	0	1	
2	0	1	0	
3	0	1	1	
4	40	0	0	
5	52	77777	77777	POW
6	30	77777	77777	E
7	56	77777	77777	*
10	0	0	0	CTR Slot for Temp 11 - Temp (15)
11	0	0	0	CTR1 Slot for (a) - (c) ; (a') - (c')
12	0	0	0	
13	42	77777	77777	
14	0	0	64000	
15	0	0	65000	
16	02	77777	77777	- lower case minus
17	17	77777	77777	(
20	43	77777	77777	)
21	0	0	0	Flag (1)
22	0	0	0	Flag (2)
23	0	0	0	Flag (3)
24	0	0	0	
25	04	03777	77777	X-S3 10
26	04	77777	77777	1
27	0	0	0	
30	0	0	0	Temp T
31	0	0	0	Temp (1)
32	0	0	0	Temp (2)
33	0	0	0	Temp (3)
34	0	0	0	Temp (4)
35	0	0	0	
36	63	77777	77777	+ lower case plus
37	77	77777	77777	
	CA	PL40		

	IA	XV	
0	RJ	XV	XV3
1	RJ	LV	LV1
2	MJ	0	MC2
3	RP	10005	XV1
4	TP	PL	WL4
	CA	XV5	

Zeroizes critical part
of Vary Stringout WL4 - WL10

	IA	XU		
0	TP	XU6	TF	} SY2 Put C/W and XS-3 symbol in CB list for variable α
1	TP	SY2	TF1	
2	TP	A	TF2	
3	TP	XU7	TF3	
4	RJ	TE	TE1	
5	MJ	0	LV	
6	0	3	3	
7	0	0	0	
10	RJ	SY	SY1	
11	EJ	MC152	ZN1	
12	MJ	0	MC104	
13	TP	XU17	XU5	} Put C/W and XS-3 symbol in CB list for a, b, and c
14	MJ	0	XU	
15	TP	XU20	XU5	
16	MJ	0	VT1	
17	MJ	0	XU15	
20	MJ	0	LV	
21	RJ	VT	VT1	
22	MJ	0	GV45	} Builds table
	CA	XU23		

Error Prints

	IA	ZA		
0	MJ	0	ID3	
1	RJ	WA	WA1	
2	TP	SY2	ZA6	
3	TP	ZA14	UP3	
4	RJ	UP2	UP	
5	MJ	0	ZA	
6	77	77777	77777	0 0 0 0 0 0
7	01	34650	15051	Δ I S Δ N 0
10	66	01240	17024	T Δ A Δ V A
11	54	34242	54630	R I A B L E
12	01	65734	72551	Δ S Y M B 0
13	46	22777	77777	L . 77 77 77 77
14	40	ZA6	6	
	CA	ZA15		

	IA	ZB		
0	MJ	0	ID3	
1	RJ	WA	WA1	
2	TP	ZB13	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZB	
5	31	51544	72466	F O R M A T
6	01	30545	45154	△ E R R O R
7	01	34223	02201	△ I . E . △
10	50	51015	22454	N O △ P A R
11	30	50663	33065	E N T H E S
12	34	65227	77777	I S .
13	40	ZB5	6	
	CA	ZB14		

	IA	ZC		
0	MJ	0	ID3	
1	RJ	WA	WA1	
2	TP	ZC16	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZC	
5	34	50265	15065	I N C O N S
6	34	65663	05066	I S T E N T
7	01	67653	00151	△ U S E △ O
10	31	01313	47230	F △ F I X E
11	27	01515	40131	D △ O R △ F
12	46	51246	63450	L O A T I N
13	32	01525	13450	G △ P O I N
14	66	01515	23054	T △ O P E R
15	24	50276	52277	A N D S .
16	40	ZC5	11	
	CA	ZC17		

	IA	ZD		
0	MJ	0	ID3	
1	RJ	WA	WA1	
2	TP	ZD14	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZD	
5	47	34506	76501	M I N U S △
6	65	34325	00134	S I G N △ I
7	46	46303	22446	L L E G A L
10	01	52543	03134	△ P R E F I
11	72	01665	10170	X △ T O △ V
12	24	54342	42546	A R I A B L
13	30	22777	77777	E .
14	40	ZD5	7	
	CA	ZD15		

	IA	ZE							
0	MJ	0	ID3						
1	RJ	WA	WA1						
2	TP	ZE14	UP3						
3	RJ	UP2	UP						
4	MJ	0	ZE						
5	34	50265	15065	I	N	C	O	N	S
6	34	65663	05066	I	S	T	E	N	T
7	01	67653	00151	△	U	S	E	△	O
10	31	01242	56551	F	△	A	B	S	O
11	46	67663	00170	L	U	T	E	△	V
12	24	46673	00165	A	L	U	E	△	S
13	34	32506	52277	I	G	N	S	.	
14	40	ZE5	7						
	CA	ZE15							
	IA	ZF							
0	MJ	0	ID3						
1	RJ	WA	WA1						
2	TP	SY2	ZF6						
3	TP	ZF21	UP3						
4	RJ	UP2	UP						
5	MJ	0	ZF						
6	77	77777	777777	0	0	0	0	0	0
7	01	66735	23001	△	T	Y	P	E	△
10	65	73472	55146	S	Y	M	B	O	L
11	01	34650	13446	△	I	S	△	I	L
12	46	30322	44601	L	E	G	A	L	△
13	34	50016	52634	I	N	△	S	C	I
14	30	50663	43134	E	N	T	I	F	I
15	26	01505	16624	C	△	N	O	T	A
16	66	34515	00101	T	I	O	N	△	△
17	01	01010	10131	△	△	△	△	△	F
20	51	54472	46622	O	R	M	A	T	.
21	40	ZF6	13						
	CA	ZF22							

	IA	ZG							
0	MJ	0	ID3						
1	RJ	WA	WA1						
2	TP	ZG13	UP3						
3	RJ	UP2	UP						
4	MJ	0	ZG						
5	34	46463	03224	I	L	L	E	G	A
6	46	01652	63430	L	△	S	C	I	E
7	50	66343	13426	N	T	I	F	I	C
10	01	50516	62466	△	N	O	T	A	T
11	34	51500	13151	I	O	N	△	F	O
12	54	47246	62277	R	M	A	T	.	
13	40	ZG5	6						
	CA	ZG14							

	IA	ZH							
0	MJ	0	ID3						
1	RJ	WA	WA1						
2	TP	ZH15	UP3						
3	RJ	UP2	UP						
4	MJ	0	ZH						
5	26	33245	03230	C	H	A	N	G	E
6	01	47512	73431	△	M	O	D	I	F
7	34	30546	50166	I	E	R	S	△	T
10	51	01304	63447	O	△	E	L	I	M
11	34	50246	63001	I	N	A	T	E	△
12	34	50313	45034	I	N	F	I	N	I
13	66	30014	65151	T	E	△	L	O	O
14	52	22777	77777	P	.				
15	40	ZH5	10						
	CA	ZH16							

	IA	ZI	
0	MJ	0	MC
1	RJ	WA	WA1
2	TP	ZI22	UP3
3	RJ	UP2	UP
4	MJ	0	ZI
5	66	33300	15067
6	47	25305	40151
7	31	01475	12734
10	31	34305	40126
11	51	47525	15030
12	50	66650	13072
13	26	30302	70104
14	10	22015	43065
15	66	01513	10165
16	30	50663	05026
17	30	01505	16601
20	26	33302	64530
21	27	22777	77777
22	40	Z15	15
	CA	ZI23	

T H E Δ N U
 M B E R Δ O
 F Δ M O D I
 F I E R Δ C
 O M P O N E
 N T S Δ E X
 C E E D Δ 1
 5 . Δ R E S
 T Δ O F Δ S
 E N T E N C
 E Δ N O T Δ
 C H E C K E
 D . 77 77 77 77

	IA	ZJ	
0	MJ	0	MC
1	RJ	WA	WA1
2	TP	ZJ20	UP3
3	RJ	UP2	UP
4	MJ	0	ZJ
5	47	34656	55230
6	46	46345	03201
7	51	54015	05101
10	54	30313	05430
11	50	26302	70165
12	30	50663	05026
13	30	22777	77777
14	01	51310	16530
15	50	66305	02630
16	01	50516	60126
17	33	30264	52722
20	40	ZJ5	7
	CA	ZJ21	

M I S S P E
 L L I N G Δ
 O R Δ N O Δ
 R E F E R E
 N C E D Δ S
 E N T E N C
 E .

Unused area

	IA	ZK		
0	MJ	0	WC	
1	RJ	WA	WA1	
2	TP	ZK11	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZK	
5	50	51016	55224	N O Δ S P A
6	26	30015	23054	C E Δ P E R
7	34	51270	16573	I O D Δ S Y
10	47	25514	62277	M B O L .
11	40	ZK5	4	
	CA	ZK12		

	IA	ZL		
0	MJ	0	MC	
1	RJ	WA	WA1	
2	TP	ZL15	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZL	
5	34	50265	14752	I N C O M P
6	46	30663	00165	L E T E Δ S
7	30	50663	05026	E N T E N C
10	30	01342	23022	E Δ I . E .
11	01	50510	11754	Δ N O Δ ' R
12	24	50323	04301	A N G E ' Δ
13	26	51475	25150	C O M P O N
14	30	50662	27777	E N T . 77 77
15	40	ZL5	10	
	CA	ZL16		

	IA	ZM		
0	MJ	0	ID3	
1	RJ	WA	WA1	
2	TP	VT20	ZM10	
3	TP	ZM21	UP3	
4	RJ	UP2	UP	
5	MJ	Q	ZM	
6	70	24543	42425	V A R I A B
7	46	30017	77777	L E Δ 77 77 77
10	01	01010	10101	Δ Δ Δ Δ Δ Δ
11	01	47676	56601	Δ M U S T Δ
12	50	51660	12633	N O T Δ C H
13	24	50323	00170	A N G E Δ V
14	24	46673	06501	A L U E S Δ
15	71	34663	33450	W I T H I N
16	01	70245	47301	Δ V A R Y Δ
17	77	77777	77777	77 77 77 77 77 77
20	46	51515	22277	L O O P .
21	40	ZM6	13	
	CA	ZM22		

	IA	ZN		
0	MJ	0	MC	
1	RJ	WA	WA1	
2	TP	ZN21	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZN	
5	30	54545	15401	E R R O R Δ
6	34	50016	65424	I N Δ T R A
7	50	65313	05401	N S F E R Δ
10	26	51475	25150	C O M P O N
11	30	50660	13422	E N T Δ I .
12	30	22014	73465	E . Δ M I S
13	65	52304	64630	S P E L L E
14	27	01715	15427	D Δ W O R D
15	01	51540	10134	Δ O R Δ Δ I
16	50	26515	45430	N C O R R E
17	26	66013	15154	C T Δ F O R
20	47	24662	27777	M A T . 77 77
21	40	ZN5	14	
	CA	ZN22		

Build VARY File

(Entered after processing sentence number of last statement in range)

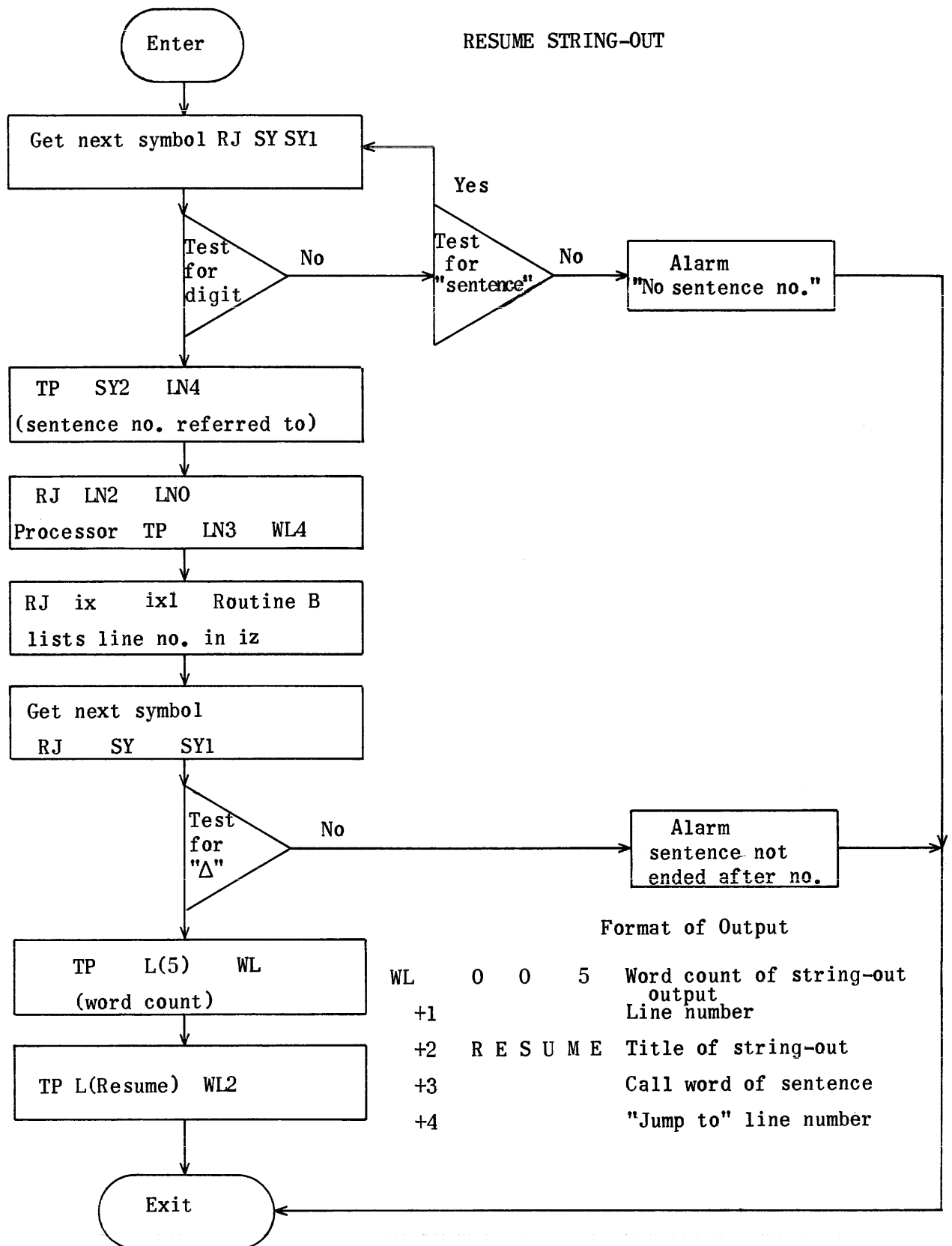
	IA	VP		
0	MJ	0	30000	
1	SP	VS5	0	} Has Vary File exceeded its maximum?
2	TJ	VF	VQ	
3	SP	WL3	0	} No
4	TJ	VS4	VP31	
① 5	TV	VI2	VP6	} Is 23000 > sentence call word?
6	TP	WL6	30000	
7	SP	VI2	0	} No, enter sentence number of last statement in range into Vary File
10	SA	VS	0	
11	TV	A	VP13	} Insert call word and no. of WITH words in next word of Vary File
12	SP	WL4	36	
13	AT	WL3	30000	
14	TU	VI	VP15	} Is current sentence number = sentence number in Vary File ?
15	SP	30000	0	
16	EJ	WL1	VQ4	
17	RA	VF	VS2	} Increase VF by two
20	SP	VI	0	
21	EJ	VI2	VP27	} (α) = (γ)?
22	TU	A	VP24	
23	TU	VI2	VP25	} Is sentence number at address given by (γ) > sentence number at address given by (α)?
24	SP	30000	0	
25	TJ	30000	VQ10	
26	TP	VI2	VI	} No, so set (α) = (γ)
③ 27	RA	VI2	VS1	
30	MJ	0	VP	} Advance (γ) by two
31	TP	VF	Q	
⑤ 32	QT	VS3	A	} Exit
33	ZJ	VP5	VP34	
34	LQ	Q	25	} Is indicator set? (i.e., has there been a 22---Vary before?)
35	QT	VS3	A	
36	TV	A	VF	
37	MJ	0	VP5	
	CA	VP40		

⑥		IA	VQ	
	0	RJ	WA	WA1
	1	TP	NV	UP3
	2	RJ	UP2	UP
⑦	3	MJ	0	VP
	4	RJ	WA	WA1
	5	TP	NV7	UP3
	6	RJ	UP2	UP
⑧	7	MJ	0	VP
	10	RJ	WA	WA1
	11	TP	NV21	UP3
	12	RJ	UP2	UP
	13	MJ	0	VP
		CA	VQ14	

	IA	VS	
0	0	1	1
1	0	2	2
2	0	2	0
3	0	0	07777
4	0	0	230000
5	0	20143	0
	CA	VS6	

144 = max. length of Vary File (excluding VF0)
(50 Vary statements in problem maximum)

	IA	NV							
0	40	NV1	6						
1	66	51510	14724	T	O	O	△	M	A
2	50	73017	02454	N	Y	△	V	A	R
3	73	01653	05066	Y	△	S	E	N	T
4	30	50263	06501	E	N	C	E	S	△
5	34	50015	25451	I	N	△	P	R	O
6	32	54244	72201	G	R	A	M	.	△
7	40	NV10	11						
10	70	24547	30147	V	A	R	Y	△	M
11	67	65660	15051	U	S	T	△	N	O
12	66	01253	00146	T	△	B	E	△	L
13	24	65660	16530	A	S	T	△	S	E
14	50	66305	02630	N	T	E	N	C	E
15	01	34500	15424	△	I	N	△	R	A
16	50	32300	15131	N	G	E	△	O	F
17	01	24507	30170	△	A	N	Y	△	V
20	24	54732	27777	A	R	Y	.		
21	40	NV22	12						
22	54	24503	23001	R	A	N	G	E	△
23	51	31013	45050	O	F	△	I	N	N
24	30	54017	02454	E	R	△	V	A	R
25	73	01307	26630	Y	△	E	X	T	E
26	50	27650	12530	N	D	S	△	B	E
27	73	51502	70154	Y	O	N	D	△	R
30	24	50323	00151	A	N	G	E	△	O
31	31	01516	76630	F	△	O	U	T	E
32	54	01010	10101	R	△	△	△	△	△
33	70	24547	32277	V	A	R	Y	.	
	CA	NV34							



RESUME STRING-OUT

RE RT4400
RE NA4431
RE NB4446

Translation subroutine regions are also needed to assemble this tape.

	IA	RT	
0	MJ	0	CT
1	RJ	SY	SY1
2	TP	SY11	Q
3	QJ	RT4	RT5
4	MJ	0	RT10
5	EJ	RT27	RT7
6	MJ	0	NA1
7	MJ	0	RT1
10	TP	SY2	LN4
11	RJ	LN2	LN
12	TP	LN3	WL4
13	RJ	IX	IX1
14	RJ	SY	SY1
15	TP	SY2	A
16	EJ	RT24	RT20
17	MJ	0	NB1
20	TP	RT26	WL
21	TP	RT25	WL2
22	RJ	SS	SS1
23	MJ	0	RT
24	01	22777	77777
25	54	30656	74730
26	0	0	5
27	65	30506	63050

Exit
Get next symbol
Is symbol a line number?
Is symbol SENTEN?
Error print-out: Sentence -- (Resume) Failed to Reference (VARY) SENTENCE
Getting line number in proper form and putting in output
Putting referenced line number in Referenced Line Number List
Is next symbol Δ . ?
Error print-out: Sentence -- (Resume) No Space Period Symbol
Word count to output
"RESUME" to title line of output
Get output block written on tape
Exit
 Δ .
R E S U M E
S E N T E N

CA RT30

	IA	NA	
0	MJ	0	RT
1	RJ	WA	WA1
2	TP	NA14	UP3
3	RJ	UP2	UP
4	MJ	0	NA
5	31	24344	63027
6	01	66510	15430
7	31	30543	05026

Print-Out and Error Reference: Sentence -- (Resume):
Print-Out: Failed to Reference (VARY) Sentence
Exit
F A I L E D
 Δ T O Δ R E
F E R E N C

10	30	01240	11770	E	△	A	△	(	V
11	24	54734	30165	A	R	Y	)	△	S
12	30	50663	05026	E	N	T	E	N	C
13	30	22777	77777	E.					
14	40	NA5	7						
	CA	NA15							

	IA	NB	
0	MJ	0	RT20
1	RJ	WA	WA1
2	TP	NB11	UP3 }
3	RJ	UP2	UP }
4	MJ	0	NB
5	50	51016	55224
6	26	30015	23054
7	34	51270	16573
10	47	25514	62277
11	40	NB5	4
	CA	NB12	

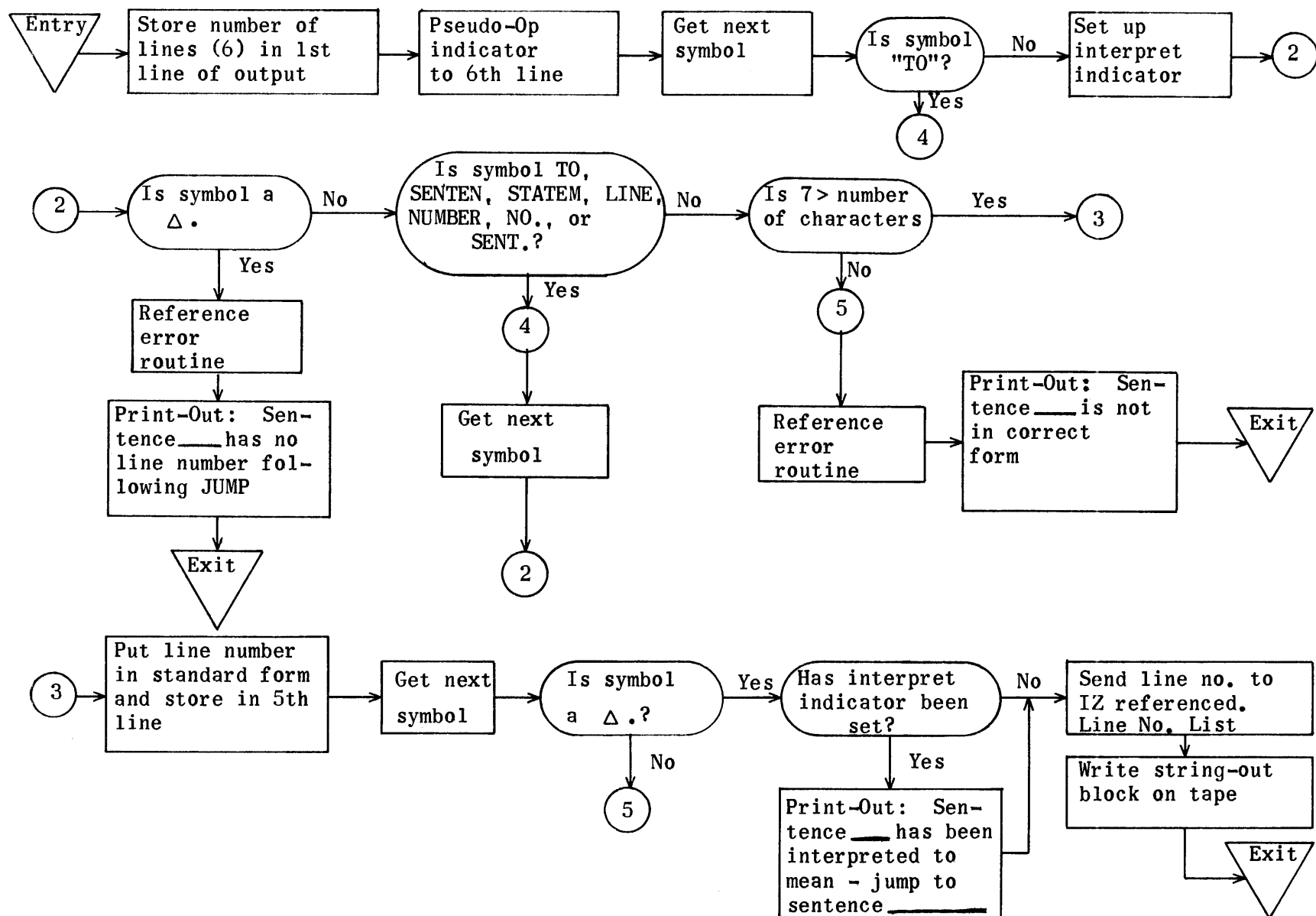
Error Reference and Print-Out: Sentence --
(Resume)

Print-Out: No Space Period Symbol

Exit

N	O	△	S	P	A
C	E	△	P	E	R
I	O	D	△	S	Y
M	B	O	L	.	

Flow Chart for Jump String-Out



JUMP STRING-OUT

Following JUMP in this instruction there should be the word TO. If this word does not appear in this sequence, the following print-out occurs after string-out has been completed: Sentence_____ Hasbeen interpreted To Mean -- Jump To Sentence_____. No reference is made to the error routine with this print-out.

Any combination of the symbols - TO, SENTEN, STATEM, LINE, NUMBER, NO., SENT. - may occur following Jump. The only essential requirement that the routine makes is that, if or when none of the above symbols appear, the symbol under surveillance must be the line number to which the jump is to be made. Of course, the symbols as shown above are merely the beginning in some instances of a more complete word. For example, STATEM is the first line output of the symbol STATEMENT. The routine only examines the first-line output of any symbol.

Each time one of the above symbols is recognized, a return is made to the "Get Next Symbol Routine" for the next symbol. Eventually the loop of recognition of these symbols must be broken by a symbol not in the group.

If a Δ . occurs before the line number has appeared, the error routine is referenced and the print-out is given: Sentence_____ Has No Line Number Following Jump.

If the loop of recognition of the 7 above-named symbols is broken by a symbol which exceeds 6 characters, the error routine is referenced, the string-out is terminated, and the print-out reads: Sentence_____ Is Not In Correct Form. The theory behind this step is that this non-recognized symbol should be the line number and, as such, have less than 7 characters.

If this assumed line number does have less than 7 characters, it is sent to the line-number routine to be put into standard form. See the Write-up on the line-number routine for what may happen to it here if it fails to meet the requirements of a line number.

The output from the line-number routine is put in the proper place in the output. See jump generation write-up for the format of the jumpstring-out output.

Now a final reference is made to the "Get Next Symbol Routine." If the next symbol is not a Δ . , the error routine is referenced, the string-out is terminated, and the print-out is given: Sentence_____Is Not in Correct Form.

If this symbol is a Δ . , a check is made if the sentence is within a pseudo op and the proper entry made in the output accordingly. Next the referenced line number is sent to the reference list by using the IX routine. Finally the completed string-out is transferred to tape by using the WT tape-write routine.

Jump String-Out Regions

RE	VN3507	} Subroutine regions used
RE	SY2466	
RE	LN2037	
RE	IX1552	
RE	WT3207	
RE	UP421	
RE	CT714	
RE	TS2762	
RE	UZ3067	
RE	SJ4700	
RE	NQ4730	
RE	JK4745	
RE	CN5020	

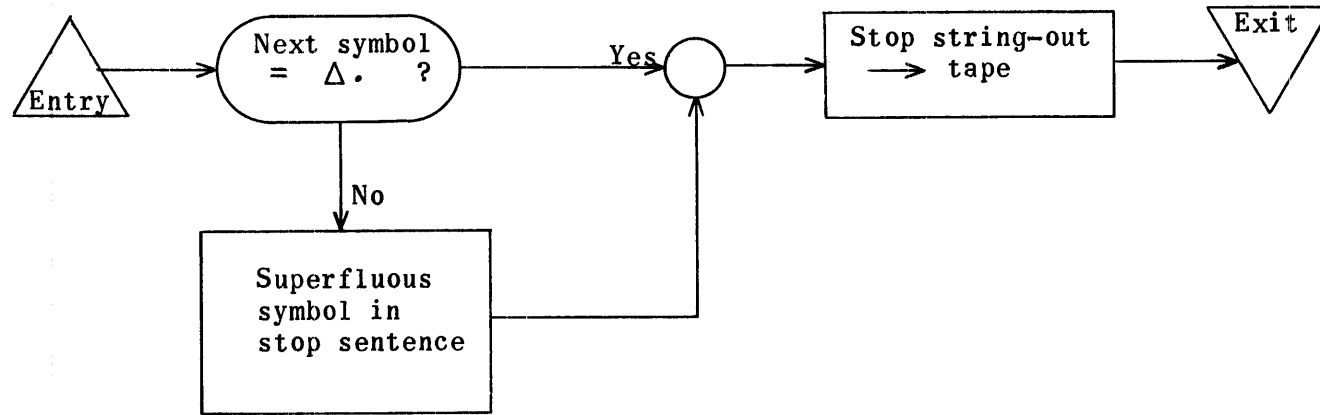
Jump String-Out

	IA	SJ		
0	MJ	0	CT	Exit
1	TP	CN	VN	Number of lines output (6) to 1st line of VN
2	TP	TS4	VN5	Pseudo-op indicator to VN5
3	TP	CN2	SJ27	Indicator 1 to SJ27
4	RJ	SY	SY1	Get next symbol
5	EJ	CN5	SJ10	Is symbol "T0"?
6	TP	CN3	SJ27	No. Clearing SJ27
7	RJ	NQ14	NQ6	Comparison subroutine
10	RJ	SY	SY1	Get next symbol
11	RJ	NQ14	NQ6	Comparison subroutine
12	MJ	0	SJ10	
13	TP	SY2	LN4	Getting line no. in standard form in VN4
14	RJ	LN2	LN	
15	TP	LN3	VN4	
16	RJ	SY	SY1	Get next symbol
17	EJ	CN1	SJ21	Is symbol Δ . ?
20	MJ	0	NQ	
21	TP	SJ27	A	If zero, without error, print: Sentence_____
22	ZJ	SJ23	NQ2	
23	TP	VN4	A	Line no. to ref. list
24	RJ	IX	IX1	
25	RJ	WT	WT1	Writing block on tape
26	MJ	0	SJ	
27	0	0	0	
	CA	SJ30		
	IA	NQ		
0	RJ	UZ	UZ1	Error reference
1	MJ	0	JK4	
2	TP	VN4	JK43	Line no. to print-out
3	MJ	0	JK10	
4	RJ	UZ	UZ1	Reference error routine
5	MJ	0	JK14	
6	EJ	CN1	NQ4	Is symbol a space period?
7	RP	20007	NQ11	Is symbol TO, SENTEN, STATEM, LINE NUMBER, NO., SENT.?
10	EJ	CN5	NQ14	
11	TP	SY5	A	
12	TJ	CN4	SJ13	Is 7 > no. of chars.?
13	MJ	0	NQ	
14	MJ	0	30000	
	CA	NQ15		
	IA	JK		
0	TP	VN1	JK26	Line no. to print-out space
1	TP	JK20	UP3	Sentence_____
2	RJ	UP2	UP	
3	MJ	0	30000	
4	RJ	JK3	JK	Sentence_____

5	TP	JK21	UP3	}	is not in correct form
6	RJ	UP2	UP		
7	MJ	0	SJ		
10	RJ	JK3	JK	}	Sentence_____ has been interpreted to mean- Jump to sentence_____
11	TP	JK22	UP3		
12	RJ	UP2	UP		
13	MJ	0	SJ23	}	Sentence _____ has no line number follow- ing jump
14	RJ	JK3	JK		
15	TP	JK23	UP3		
16	RJ	UP2	UP	}	
17	MJ	0	SJ		
20	0	JK24	3		
21	40	JK27	4		
22	40	JK33	12		
23	40	JK45	6		
24	65	30506	63050	}	Sentence Δ
25	26	30017	77777		
26	0	0	0		
27	01	34650	15051	}	is not in correct form
30	66	01345	00126		
31	51	54543	02666		
32	01	31515	44722	}	
33	01	33246	50125		
34	30	30500	13450		
35	66	30545	25430	}	Has been interpreted to mean _____ Jump to sentence_____ Δ .
36	66	30270	16651		
37	01	47302	45002		
40	02	44674	75201	}	
41	66	51016	53050		
42	66	30502	63001		
43	0	0	0	}	
44	01	22777	77777		
45	01	33246	50150		
46	51	01463	45030	}	Has no line number following jump
47	01	50674	72530		
50	54	01315	14646		
51	51	71345	03201	}	
52	44	67475	22277		
	CA	JK53			

	IA	CN		
0	0	0	6	Δ .
1	01	22777	77777	
2	0	0	1	
3	0	0	0	
4	0	0	7	
5	66	51777	77777	T 0
6	65	30506	63050	S E N T E N
7	65	66246	63047	S T A T E M
10	46	34503	07777	L I N E
11	50	67472	53054	N U M B E R
12	50	51227	77777	N O .
13	65	30506	62277	S E N T .
	CA	CN14		

Stop String-out Flow Chart



STOP STRING-OUT

RE SV4400

String-Out Subroutine regions are needed for assembly of this tape.

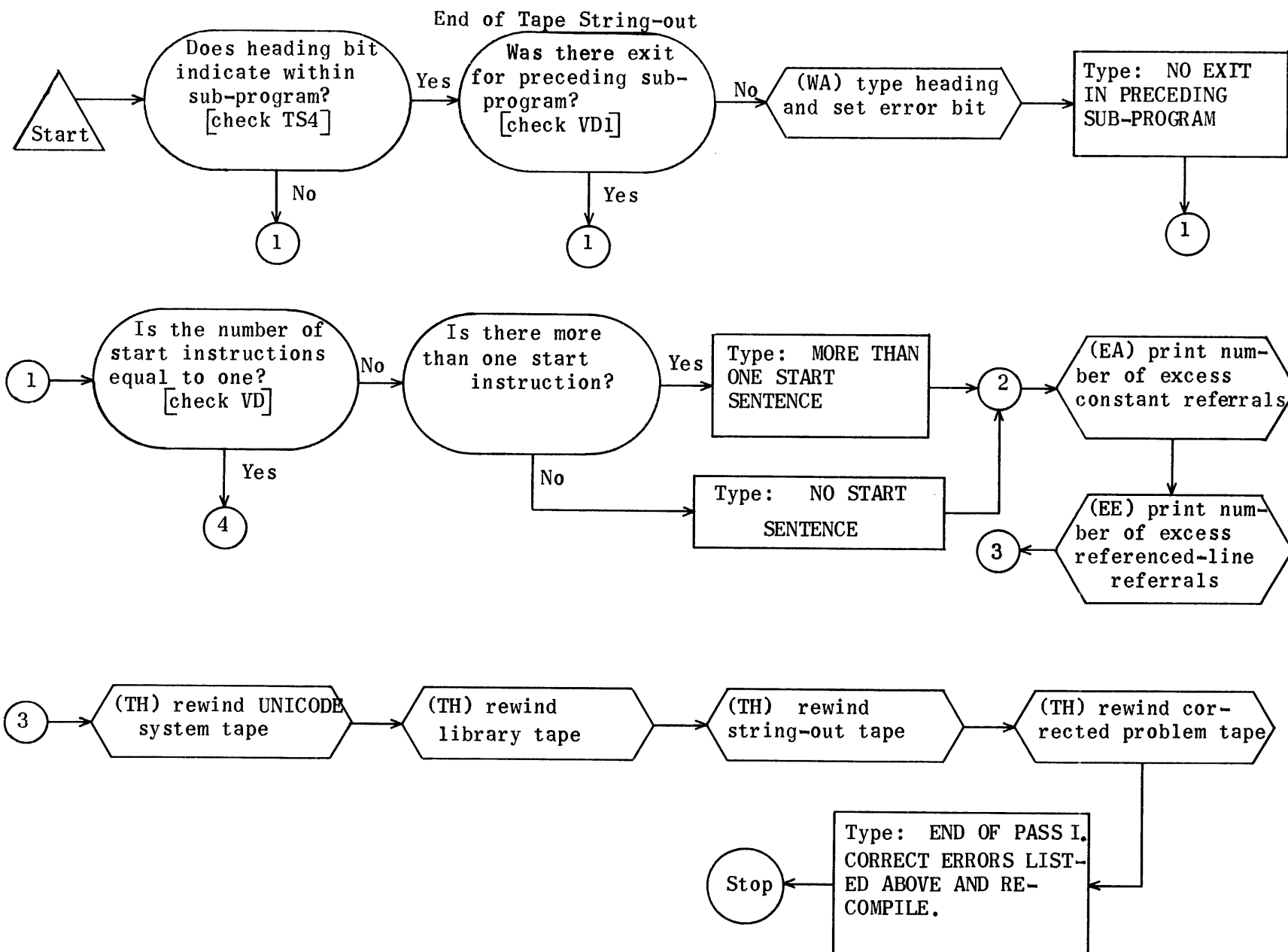
	IA	SV		
0	RJ	SY	SY1	Get next symbol
1	EJ	SV7	SV5	= Δ . ?
2	RJ	WA	WA1	No-error
3	TP	SV10	UP3	Print-out
4	RJ	UP2	UP	
5	RJ	SS	SS1	Yes - string-out $\rightarrow$ tape
6	MJ	0	CT	
7	01	22777	77777	
10	40	SV11	6	
11	65	67523	05431	S U P E R F
12	46	67516	76501	L U O U S Δ
13	65	73472	55146	S Y M B O L
14	01	34500	16566	Δ I N Δ S T
15	51	52016	53050	O P Δ S E N
16	66	30502	63022	T E N C E .
	CA	SV17		

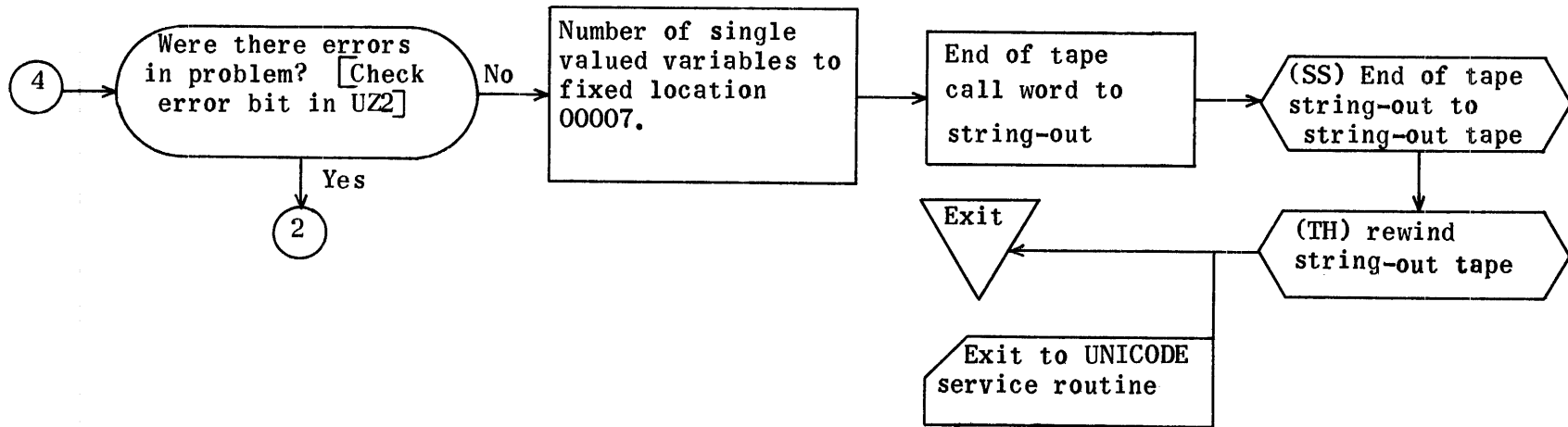
END OF TAPE STRING-OUT

The End of Tape string-out routine closes out the translation phase (Pass I) of the Unicode compilation run. If there was a sub-program preceding the "End of Tape" sentence in the input program, the routine checks to see if an "Exit" instruction was included in the sub-program. If not, the error bit is set and "No exit in preceding sub-program." is typed on the on-line Flexowriter. The routine also checks to see if there was exactly one "Start" sentence in the input program and if not, types "More than one start sentence" or "No start sentence." on the Flexowriter, whichever applies.

If there were previous errors in the program or if there was not exactly one "Start" sentence, the routine prints the number of excess constant referrals and the number of excess referenced-line referrals. The routine then rewinds the Unicode System Tape (Servo 1), the Library Tape (Servo 2), the String-out Tape (Servo 3 or 6) and the Corrected Problem Tape (Servo 5). The routine then types "End of Pass I. Correct errors listed above and recompile." on the on-line Flexowriter and stops.

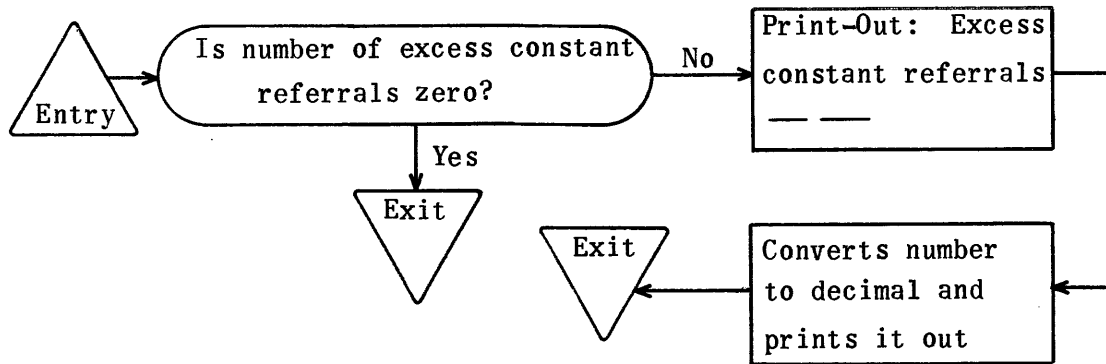
If there were no errors in the problem, the number of single valued variables is stored in fixed location 00007, the End of Tape callword (23000) is stored in the string-out, and the string-out is written on tape. The String-out Tape is then rewound and the routine exits to the Unicode Service Routine in preparation for the next phase of compilation.



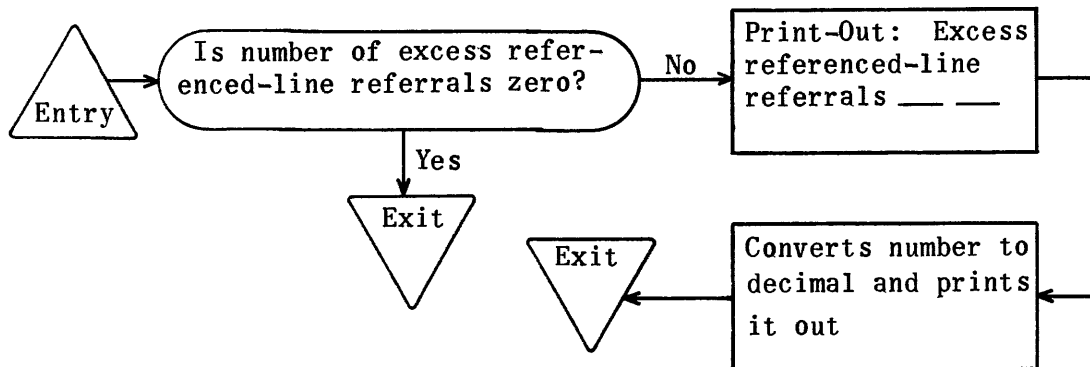


Flow Charts for Subroutines Supplied to End-of-Tape String-Out

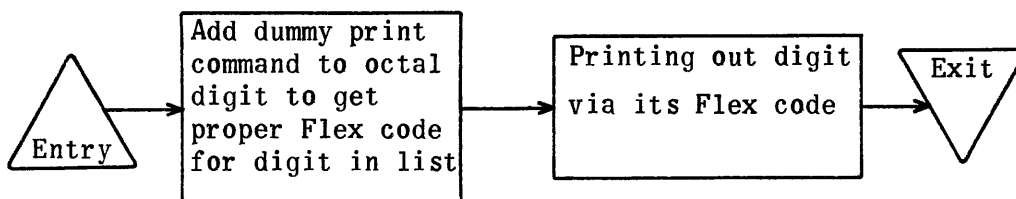
Print-Out of FX13, Number of Excess Constant Referrals



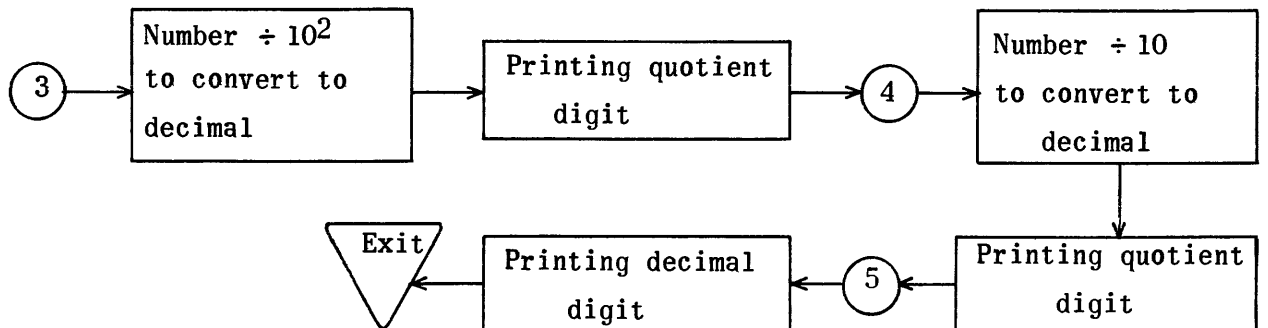
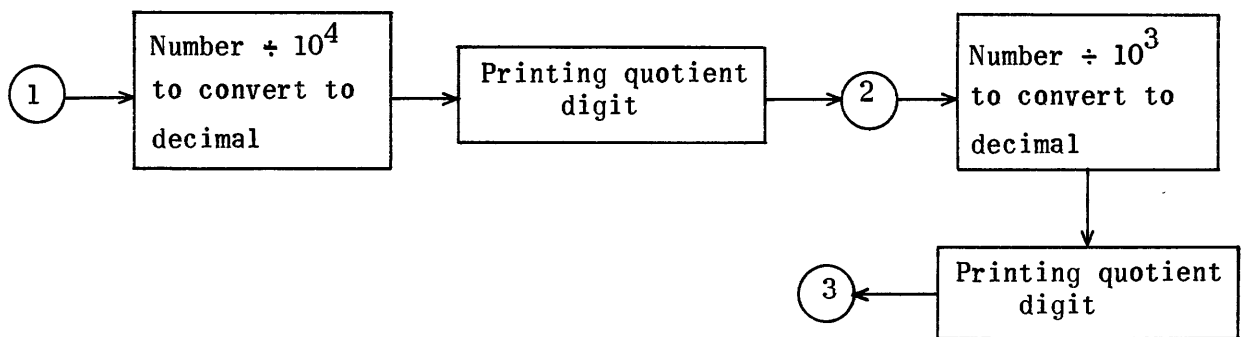
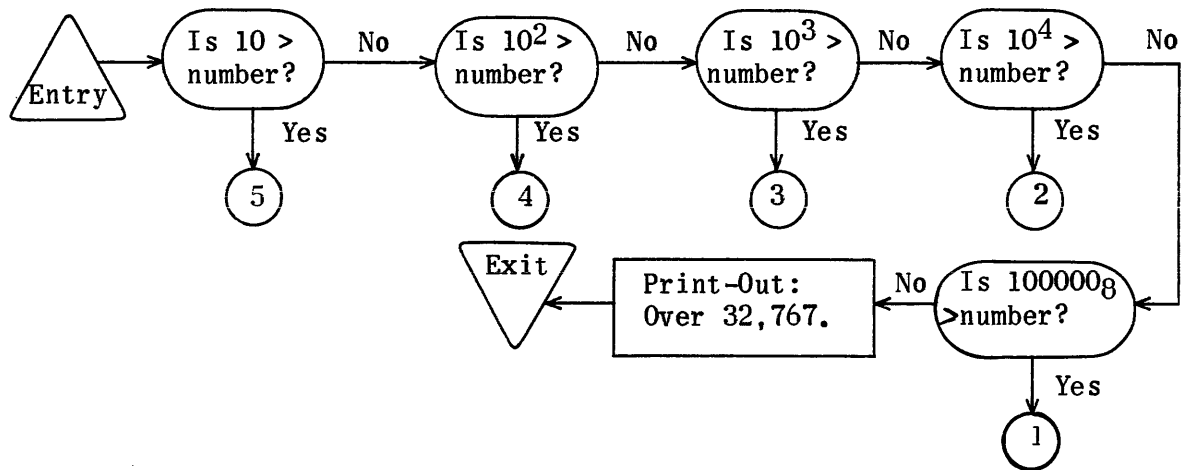
Print-Out of IX47, Number of Excess Referenced-Line Referrals



Single Digit Print-Out



Octal to Decimal Print-Out for Numbers $\leq 77777_8$ and ≥ 0



Regions for End of Tape String-Out
(String-out Subroutine Regions Also Required)

RE ZA77000
RE TV4400
RE TW4435
RE TX4447
RE TY4455
RE EA4502
RE EE4520
RE NU4537
RE ZU4555

	IA	TV		
0	MJ	0	ZA10	Exit → UNICODE serv. rtn. (read 1 blk UNICODE tape & jump to 1st word)
1	TP	TS4	Q	Heading bit → Q35
2	QJ	TV3	TV10	Within sub-program (pseudo operation)?
3	TP	VD1	Q	Exit bit → Q35
4	QJ	TV10	TV5	Was there exit inst. for preceding sub-program
5	RJ	WA	WA1	
6	TP	75552	UP3	Parameter → Uniprint
7	RJ	UP2	UP	Print: NO EXIT IN PRECEDING SUB-PROGRAM
① 10	TP	VD	A	#Start instructions → A
11	EJ	TX	TW	#Start inst. = 1?
12	ZJ	TV13	TV15	#Start inst. = 0? No → more than 1 start inst.
13	TP	TY	UP3	Alarm #1 parameter → Uniprint
14	MJ	0	TV16	
15	TP	TY6	UP3	Alarm #2 parameter → Uniprint
16	RJ	UP2	UP	Print alarm
② 17	RJ	EA	EA1	Print #excess constant referrals
20	RJ	EE	EE1	Print # excess referenced-line referrals
21	TP	TX2	TH3	
22	RJ	TH2	TH	Rewind UNICODE tape (servo #1)
23	TP	TX3	TH3	
24	RJ	TH2	TH	Rewind library tape (servo #2)
25	TP	TX4	A	Codeword to rewind stringout tape (servo #3) → A
26	AT	TN	TH3	Adv. servo no. by 3 → servo #6 if 7 servos
27	RJ	TH2	TH	Rewind stringout tape (servo #3 or #6)
30	TP	TX5	TH3	
31	RJ	TH2	TH	Rewind corrected problem tape (servo #5)
32	TP	TY12	UP3	
33	RJ	UP2	UP	Print: END OF PASS I. CORRECT ERRORS ETC.
34	MS	0	TV34	
	CA	TV35		

④

	IA	TW		
0	TP	UZ2	A	# errors in problem → A
1	ZJ	TV17	TW2	Were there errors in program?
2	TP	VB1	A	# single valued variables less 1 → A
3	AT	TX	7	# single valued variables → "v" of fixed loc. 7
4	TP	TX1	WL3	End of tape callword (23000) → string-out
5	RJ	SS	SS1	End of tape string-out → tape
6	TP	TX4	A	Codeword to rewind string-out tape (servo #3) → A
7	AT	TN	TH3	Adv. servo no. by 3 → Servo #6 if 7 servos
10	RJ	TH2	TH	Rewind string-out tape
11	MJ	0	TV	
	CA	TW12		

	IA	TX		
0	0	0	1	
1	0	0	23000	
2	10	1	0	Codeword to rewind UNICODE tape
3	10	2	0	Codeword to rewind library tape
4	10	3	0	Codeword to rewind string-out tape (5 servos)
5	10	5	0	Codeword to rewind corrected problem tape
	CA	TX6		

	IA	TY		
0	00	TY1	5	
1	47	51543	00166	M O R E Δ T
2	33	24500	15150	H A N Δ O N
3	30	01656	62454	E Δ S T A R
4	66	01653	05066	T Δ S E N T
5	30	50263	02277	E N C E . 77
6	00	TY7	3	
7	50	51016	56624	N O Δ S T A
10	54	66016	53050	R T Δ S E N
11	66	30502	63022	T E N C E .
12	00	TY13	12	
13	30	52070	15131	E N D Δ O F
14	01	52246	56501	Δ P A S S Δ
15	34	22010	12651	I . Δ Δ C O
16	54	54302	66601	R R E C T Δ
17	30	54545	15465	E R R O R S
20	01	46346	56630	Δ L I S T E
21	27	01242	55170	D Δ A B O V
22	30	01245	02701	E Δ A N D Δ
23	54	30265	14752	R E C O M P
24	34	46302	27777	I L E . 77 77
	CA	TY25		

Print-Out of FX13, Number of Excess Constant Referrals

	IA	EA		
0	MJ	0	30000	Exit
1	TP	FX13	A }	
2	ZJ	EA3	EA }	Is the number in FX13 zero?
3	TP	EA10	UP3 }	
4	RJ	UP2	UP }	Print-Out: EXCESS CONSTANT REFERRALS --
5	TP	FX13	A }	
6	RJ	ZU	ZU1 }	Converts number to decimal and prints it out
7	MJ	0	EA	
10	0	EA11	5	Parameter for print-out
11	30	72263	06565	E X C E S S
12	01	26515	06566	△ C O N S T
13	24	50660	15430	A N T △ R E
14	31	30545	42446	F E R R A L
15	65	02027	77777	S - -
	CA	EA16		

Print-Out of IX47, Number of Excess Referenced-Line Referrals

	IA	EE		
0	MJ	0	30000	Exit
1	TP	IX47	A }	
2	ZJ	EE3	EE }	Is the number zero?
3	TP	EE10	UP3 }	
4	RJ	UP2	UP }	Print-Out: EXCESS REFERENCED-LINE REFERRALS --
5	TP	IX47	A }	
6	RJ	ZU	ZU1 }	Converts number to decimal and prints it out
7	MJ	0	EE	
10	0	EE11	6	Parameter for Print-Out
11	30	72263	06565	E X C E S S
12	01	54303	13054	△ R E F E R
13	30	50263	02702	E N C E D -
14	46	34503	00154	L I N E △ R
15	30	31305	45424	E F E R R A
16	46	65020	27777	L S - -
	CA	EE17		

Input Digit in Av. Single Digit Print-Out Routine

	IA	NU		
0	AT	NU3	NU1	Entry. Adds dummy print command to octal digit & puts into next instruction
1	PR	0	0	Prints out decimal digit
2	MJ	0	30000	Exit
3	PR	0	NU4	Dummy Print Command
4	0	0	37	0
5	0	0	52	1
Flex	6	0	74	2
Codes	7	0	70	3

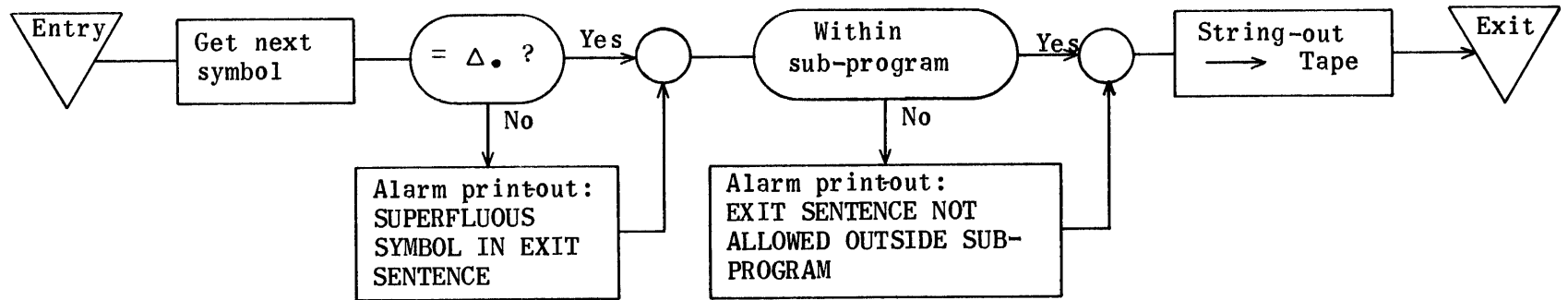
Flex Codes	10	{	0	0	64	4
	11		0	0	62	5
	12		0	0	66	6
	13		0	0	72	7
	14		0	0	60	8
	15		0	0	33	9

CA NU16

Input in A_v . Octal to Decimal Print-Out for Numbers ≤ 77777
 Print-Out for Numbers > 77777 is: OVER 32,767

	IA	ZU	
0	MJ	0 30000	Exit
1	TJ	ZU32 ZU21	Entry, Is 10 > no.?
2	TJ	ZU33 ZU17	Is $10^2 > \text{no.}$?
3	TJ	ZU34 ZU15	Is $10^3 > \text{no.}$?
4	TJ	ZU35 ZU13	Is $10^4 > \text{no.}$?
5	TJ	ZU41 ZU11	Is 100 000 ₈ > no.?
6	TP	ZU31 UP3	} Error print-out: OVER 32,767
7	RJ	UP2 UP	
10	MJ	0 ZU	Jump to exit
11	DV	ZU35 Q	$\div 10^4$ to convert to decimal
12	RJ	ZU30 ZU24	Printing out digit
13	DV	ZU34 Q	$\div 10^3$ to convert to decimal
14	RJ	ZU30 ZU24	Printing digit
15	DV	ZU33 Q	$\div 10^2$ to convert to decimal
16	RJ	ZU30 ZU24	Printing digit
17	DV	ZU32 Q	$\div 10$ to convert to decimal
20	RJ	ZU30 ZU24	Printing digit
21	RJ	NU2 NU	Printing digit
22	PR	0 ZU36	Printing carriage return
23	MJ	0 ZU	Jump to exit
24	TP	A ZU42	} Subroutine to facilitate continued conversion of octal number to decimal and print out decimal digit obtained
25	TP	Q A	
26	RJ	NU2 NU	
27	TP	ZU42 A	
30	MJ	0 30000	
31	40	ZU37 2	Parameter for print-out: OVER 32,767
32	0	0 12	10^2
33	0	0 144	10^3
34	0	0 1750	10^4
35	0	0 23420	10^5
36	0	0 45	Carriage return
37	51	70305 40106	0 V E R Δ 3
40	05	23121 11222	2 , 7 6 7
41	0	1 0	100,000 ₈ < 10^5
42	0	0 0	Temporary
	CA	ZU43	

Exit String-out Flow Chart



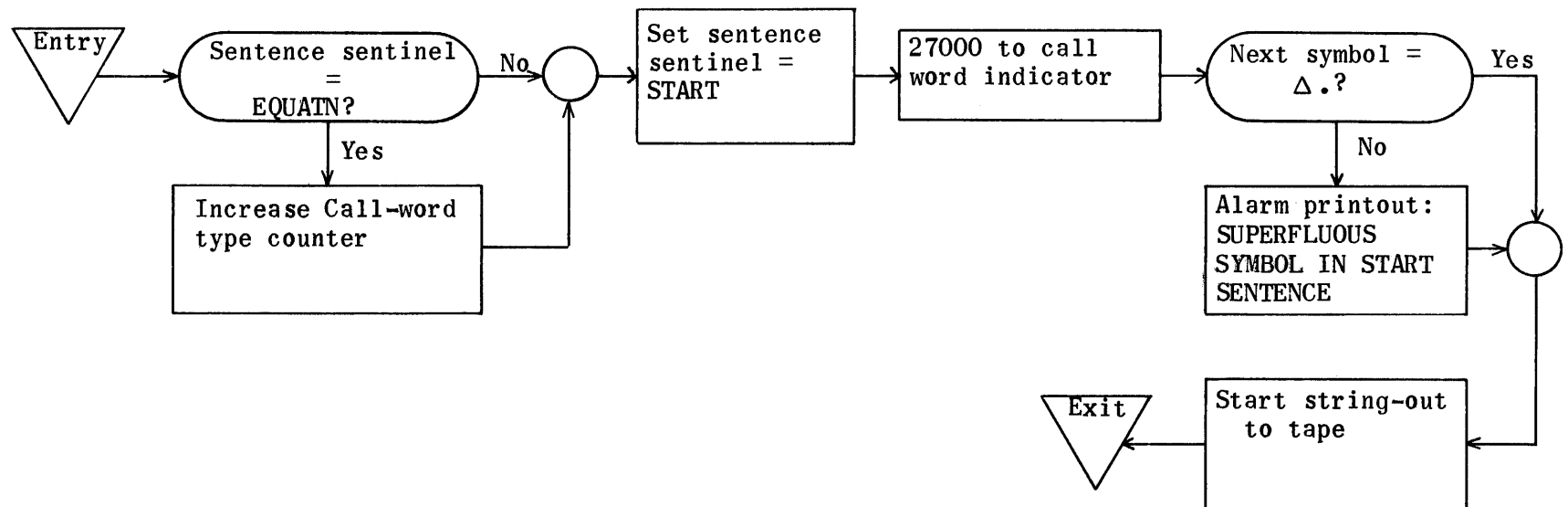
Exit String-Out

RE SW4400
RE SX4416

String-Out Subroutine regions also needed to assemble this tape

0	IA	SW		
0	RJ	SY	SY1	Get next symbol
1	EJ	SX	SW5	= Δ . ?
2	RJ	WA	WA1	No; get heading &
3	TP	SX1	UP3	Print out error sentence
4	RJ	UP2	UP	
5	TP	TS4	Q	Yes;
6	QJ	SW12	SW7	Op of TS4 = 40? (Within Subprogram?)
7	RJ	WA	WA1	No; get heading &
10	TP	SX12	UP3	Print out alarm sentence
11	RJ	UP2	UP	
12	TV	TS4	WL4	Yes, call word $\rightarrow$ WL4
13	RJ	SS	SS1	Output to tape
14	TP	SX11	VD1	Indicator to VD1 that an Exit has been found in the last pseudo operation
15	MJ	0	CT	Exit
	CA	SW16		
0	IA	SX		
0	01	22777	77777	Δ .
1	40	SX2	6	
2	65	67523	05431	S U P E R F
3	46	67516	76501	L U O U S Δ
4	65	73472	55146	S Y M B O L
5	01	34500	13072	Δ I N Δ E X
6	34	66016	53050	I T Δ S E N
7	66	30502	63022	T E N C E .
10	77	0	0	mask
11	40	0	0	
12	40	SX13	10	
13	30	72346	60165	E X I T Δ S
14	30	50663	05026	E N T E N C
15	30	01505	16601	E Δ N O T Δ
16	24	46465	17130	A L L O W E
17	27	01516	76665	D Δ O U T S
20	34	27300	16567	I D E Δ S U
21	25	01525	45132	B Δ P R O G
22	54	24472	27777	R A M .
	CA	SX23		

Start String-out Flow Chart



START STRING-OUT

RE SU4400

String-Out Subroutine regions are needed for assembly of this tape

IA	SU		
0	RA	VD	SU17 Increase VD by 1
1	SP	WL2	0 }
2	EJ	SU20	SU4 }
3	MJ	0	SU5 No; do not increase VB4
4	RJ	XJ	XJ1 Yes; increase VB4
5	TP	SU21	WL2 START → WL2
6	TP	SU22	A }
7	AT	VB4	WL3 Call word + (VB4) → WL3
10	RJ	SY	SY1 Get next symbol
11	EJ	SU23	SU15 = Δ . ?
12	RJ	WA	WA1 No; error
13	TP	SU24	UP3 Print-out
14	RJ	UP2	UP }
15	RJ	SS	SS1 String-out → tape
16	MJ	0	CT Exit
17	0	0	1
20	30	53672	46650 E Q U A T N
21	65	66245	46677 S T A R T Δ
22	0	0	27000 C/W
23	01	22777	77777 Δ .
24	40	SU25	7
25	65	67523	05431 S U P E R F
26	46	67516	76501 L U O U S Δ
27	65	73472	55146 S Y M B O L
30	01	34500	16566 Δ I N Δ S T
31	24	54660	16530 A R T Δ S E
32	50	66305	02630 N T E N C E
33	22	77777	77777 .
	CA	SU34	

EQUATION TRANSLATION ROUTINE

When an equation is encountered in a UNICODE Program the translation control transfers this routine from drum to core and jumps to it. The main functions of the routine are to make up a translation list to be used to generate machine code, assign call words to symbols, and detect and type out errors.

The translation list is made up by each translation routine as an output (Region WL). The equation translation routine is unique in one use of this list in that it keeps another list within the translation list. This is called the function dummies list and occupies locations WL4-WL23 or 16₁₀ locations. Hence there is room for 8 two-word items, the first word being the excess-three code for the symbol and the second word the dummy call word of this symbol. WLO-WL3 is the heading and the coded symbols start at WL24.

In an equation before START, the symbol is assigned a call word as follows:

- 1) If the symbol is in the Combination List, it already has a call word which determines the type of symbol with one exception. If the call word is 65xxx and the next symbol is an open parenthesis, the call word is changed to 66xxx and case 3 applies. If the symbol is that of a subscripted variable, the subscripts are assigned dummy call words and put in the list at WL4-WL23 and are handled similarly to the dummy arguments of a function.
- 2) If its first character is I, J, K, L or M, it is assigned a 64xxx call word and the equation is assumed to be fixed point.
- 3) If neither of the above cases applies, the next symbol is checked and if it is an open parenthesis the variable is assigned a 66xxx call word and a function mode is set. In this case all variables within the set of parentheses on the left are function dummies and put in the list at WL4-WL23. These dummies apply only to this particular equation.

After START no function call words are assigned by the equation translator but the pseudo operation heading translator assigns symbols dummy function call words, if equated to a real function by a COMPUTE. In an equation no arguments should be written with a function if the function appears on the

right or if it appears after START.

Operation symbols are determined to be fixed point if they appear in a fixed-point equation (determined by the first symbol) or if they appear when in the subscript mode; otherwise they are assumed to be floating point. The list of errors found in this paper suggest what cannot be written in an equation.

The following pages contain:

- 1) A list of call words or special codes assigned by the equation translator.
- 2) An example of the output for a function.
- 3) A list of error prints.
4. Explanations of some of the subroutines.

Special Codes Used by Equation Translation

Op	u	v	
0	0	10	Absolute value (open), floating and fixed
0	0	12	Absolute value (closed), floating
0	0	11	Absolute value (closed), fixed
0	0	20	+ floating
0	0	21	+ fixed
0	0	30	- floating binary
0	0	31	- fixed binary
0	0	32	- floating unary
0	0	33	- fixed unary
0	0	40	, ; comma or semicolon
0	0	50	= equals sign
0	0	60	* floating multiply
0	0	61	* fixed multiply
0	0	70	/ floating divide
0	0	71	/ fixed divide
0	0	100	POW
0	0	120	Δ . (space period)
0	1	0	(open parenthesis
0	2	0	) closed parenthesis
0	0	17100	Superscript = -1 (superscript 1 is ignored)
0	0	16000	Superscript = 1/2
0	0	16100	Superscript = -1/2
0	0	15000	Superscript = 2
0	0	15100	Superscript = -2
0	0	14000	Superscript = 3
0	0	14100	Superscript = -3
0	0	130xx	Superscript = +4 to +77 ₈ ($4 \leq xx \leq 77$)
0	0	131xx	Superscript = -4 to -77 ₈ ($4 \leq xx \leq 77$)
0	0	101	Integral POW > 63
0	0	25xxx	In equation for 66xxx, 65xxx or 64xxx symbol, before START
0	0	24xxx	Equation for 77xxx type symbol, before START.

EXAMPLE

Output of Equation Translators for an equation which occurs before START:

$$F(x_1, x_2(I)) = -|x_1|^3 + x_2(J) * S_2(I, L) / R \text{ POW SIN } Y \Delta.$$

	OP	u	v		
WL	0	0	55	# words in list	
WL1	01	06142	21001	Line number (excess 3) 39.5	} Heading
WL2	30	53672	46650	EQUATN (excess three)	
WL3	0	0	25xxx	Sentence call word (x=octal digit)	
WL4	72	04777	77777	x1 (excess three)	} The function dummy list for this equation.
WL5	0	0	62000	Call word of x1	
WL6	72	05777	77777	x2	
7	0	0	75001	Call word of x2	
10	34	77777	77777	I	
11	0	0	62003	Call word of I	
				} WL12-WL23=0	
WL24	0	0	66000	F	} The Translated Equation
	0	0	50	=	
	0	0	32	-	
	0	0	10		
	0	0	62000	x1	
	0	0	12		
	0	0	14000	Superscript 3	
	0	0	20	+	
	0	0	75001	x2	
	0	1	0	(	
	0	0	64xxx	J	
	0	2	0	)	
	0	0	60	*	
	0	0	77xxx	S2	
	0	1	0	(	
	0	0	62003	I	
	0	0	40	,	
	0	0	64xxx	L	
	0	2	0	)	
	0	0	70	/	
	0	0	65xxx	R	
	0	0	100	POW	
	0	0	5xxxx	SIN	
	0	0	65xxx	Y	
	0	0	120	Δ.	

Error Texts for Equations

- E1. Function symbol, ----, (sub program dummy) in fixed point equation.
- E2. Subscripted variable symbol, ----, (sub program'dummy) in fixed point equation.
- E3. Floating point variable, ----, (sub program dummy) used in fixed point equation.
- E4. Function symbol, ----, (sub program dummy) among subscripts of ----.
- E5. Subscripted variable symbol, ----, (sub program dummy) among subscripts of ----.
- E6. Floating point variable, ----, (sub program dummy) among subscripts of ----.
- E7. Fixed point variable, ----, (sub program dummy) in floating pt. equation.
- E8. Subscripted variable symbol, ----, among subscripts of ----.
- E9. Library routine, ----, among subscripts of ----.
- E10. Function, ----, among subscripts of ----.
- E11. Floating point variable, ----, among subscripts of ----.
- E12. Fixed point variable, ----, in floating point equation.
- E13. Library routine, ----, in fixed point equation.
- E14. Floating point symbol, in fixed point equation.
- E15. Function, ----, in fixed point equation.
- E16. Subscripted variable symbol, ----, in fixed point equation.
- E17. Library routine symbol, ----, with more than one argument, not followed by open parenthesis.
- E18. Subscripted variable symbol, ----, not followed by an open parenthesis.
- E19. Subscripted variable symbol, ----, (sub program dummy) not followed by an open parenthesis.
- E20. Subscripted variable symbol, ----, (function dummy) among subscripts of

*** END ***

- E21. Floating point variable, ----, (function dummy) among subscripts of ----.
- E22. Fixed point variable, ----, (function dummy) in floating point equation.
- E23. Subscripted variable symbol, ----, (function dummy) not followed by an open parenthesis.
- E24. More than one separate equation for ----.
- E25. Superfluous symbols on left.
- E26. Function, ----, on left, not followed by an open parenthesis. Rest of this sentence not checked.
- E27. Library routine symbol, ----, is first symbol of sentence. Rest of sentence not checked.
- E28. An Equation for ---- in the range of a VARY sentence in which ---- appears.
- E29. Illegal symbol (----) for left of equation.
- E30. More than one subscript on ----, an argument of the function ----.
- E31. Library routine symbol, ----, on left, among arguments of the function ----.
- E32. Superscript symbol, ----, among subscripts of ----.
- E33. Superscript symbol, ----, in fixed point equation.
- E34. More than four superscript symbols in sequence.
- E35. POW operation symbol among subscripts of ----.
- E36. POW operation symbol in fixed point equation.
- E37. Number of library routine operands (by comma count) not equal to number listed for this routine.
- E38. Interlocking parenthesis and absolute value signs. (|) |
- E39. Closed parenthesis appears with no corresponding open.

- E40. Number of subscripts on ---- (by comma count) is not equal to number obtained from dimension sentence.
- E41. Open parenthesis among subscripts of ----.
- E42. Illegal symbol, ----, for right of equation.
- E43. Incorrect use of comma.
- E44. Number of equals signs not equal to one.
- E45. Some open parentheses not closed.
- E46. Number of open absolute value signs not equal to number of closed.
- E47. Number of open parentheses on left not equal to number of closed.
- E48. Superfluous arguments on function ----.
- E49. Within arguments of more than 7 library routines. Arguments of ---- not checked.
- E50. Too many dummy arguments on function ----. Rest of this sentence not checked.
- E51. Pseudo operation symbol, ----, on right.
- E52. Incorrect symbol sequence. sym sym
- E53. Closed absolute value appears with no corresponding open.
- E54. Open parenthesis, on left, among subscripts of ----.
- E55. Incorrect use of open parenthesis on left among arguments of ----.
- E56. Incorrect use of comma among arguments of ----.
- E57. More than 29 unclosed open parentheses and/or absolute value signs.
- E58. Constant illegal on left, before start.

SYMBOL PAIR CHECKER

Almost every pair of symbols is checked by this routine. The bit in the array on the following page corresponding to the symbol pair consisting of the last two symbols picked up is checked. If it is a 1, the pair is illegal and the error is printed. The left symbol is picked up from SZ2 and the right from SY2.

After checking a pair the routine sets the right symbol as the left symbol for the next check. Hence, before entering the routine, the new right symbol must be set as an input.

The left symbol is indicated by an address in PC2 and the right symbol by a shift count in PC3. The addresses range from PD2 to PD22 and the shifts from 0 to 20. Notice from the way that the array is set up that to change a shift count to an address it is only necessary to add the shift count to address PD2. This is the way a right symbol becomes the next left symbol.

If for some reason it is desired that no error be printed out for a symbol, the shift count is set to 20 and the symbol pairs with any other legally.

The codes for the error print of this routine are with those of the translation subroutines. It is called F18 and is in region FI.

Meaning of Region PD Used by Symbol Pair Checker (PC)

<u>Left Symbol</u>	<u>Right Symbol</u>										<u>Ad- dress</u>	<u>Octal</u>
	Doesn't matter (open)	Δ .	=	LIB	POW	Up/ Up-	Up Const. * or /	+ or -	, or ;	FTN	Sub. Var. VAR. or Const.) or (closed)	
(	0 0	1 1 0	1 1 1	1 1 1	1 1 0	1 0 0	0 1 0	0 1 0	0 1 0	0 1 0	PD2 =	67642
) or (closed)	0 1	0 0 1	0 1 0	0 0 0	0 0 0	0 1 1	1 0 1	1 0 1	1 0 1	1 0 1	PD3 =	112035
VAR. or Const.	0 1	0 0 1	0 1 0	0 0 0	0 0 0	0 1 1	1 0 1	1 0 1	1 0 1	1 0 1	PD4 =	112035
Sub. Var.	0 1	0 0 1	0 1 0	0 0 0	0 0 0	0 1 1	1 0 0	1 0 0	1 0 0	1 0 0	PD5 =	112034
FTN	0 1	0 0 1	0 1 0	0 0 0	0 0 0	0 1 1	1 0 0	1 0 0	1 0 0	1 0 0	PD6 =	112034
, or ;	0 0	1 1 0	1 1 1	1 1 1	1 1 0	1 0 0	0 1 0	0 1 0	0 1 0	0 1 0	PD7 =	67642
+ or -	0 0	1 0 0	1 1 1	1 1 1	1 1 1	1 0 0	0 1 0	0 1 0	0 1 0	0 1 0	PD10 =	47742
* or /	0 0	1 1 0	1 1 1	1 1 1	1 1 0	1 0 0	0 1 0	0 1 0	0 1 0	0 1 0	PD11 =	67642
Up Const.	0 1	0 0 1	0 0 1	1 0 0	0 0 0	0 1 1	1 0 1	1 0 1	1 0 1	1 0 1	PD12 =	111435
Up -	0 1	1 0 1	1 1 1	1 1 1	0 1 1	1 1 1	1 1 1	1 1 1	1 1 1	1 1 1	PD13 =	157377
Up /	0 1	1 1 1	1 1 1	1 1 1	0 1 1	1 1 1	1 1 1	1 1 1	1 1 1	1 1 1	PD14 =	177377
POW	0 0	1 0 0	1 1 1	1 1 1	1 1 0	1 0 0	0 1 0	0 1 0	0 1 0	0 1 0	PD15 =	47642
LIB	0 0	1 1 0	1 1 1	1 1 1	1 1 0	1 0 0	0 1 0	0 1 0	0 1 0	0 1 0	PD16 =	67642
=	0 0	1 1 0	1 1 1	1 1 1	1 1 0	1 0 0	0 1 0	0 1 0	0 1 0	0 1 0	PD17 =	67642
Δ .	0 1	1 1 1	1 1 1	1 1 1	1 1 1	1 1 1	1 1 1	1 1 1	1 1 1	1 1 1	PD20 =	177777
(open)	0 0	1 0 0	1 1 1	1 1 1	1 1 0	1 0 0	0 1 0	0 1 0	0 1 0	0 1 0	PD21 =	47642
Doesn't matter	0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	PD22 =	0

To see if a symbol pair is legal, find the left symbol in the column on the left and the right symbol in the row at the top. If the box of intersection of the row and column of this symbol pair contains a 1, the pair is illegal. If it contains a zero, the pair is legal.

0 = legal symbol pair

1 = illegal symbol pair

Get Call Word from Dummy Function List

This routine searches addresses WL4 - WL23 for the symbol in SY2. If the symbol is found it is put in TU2 and its call word is in TU3 in the v address. Reference the routine as follows:

RJ	TU	TU1
MJ	0	(Not in list)
MJ	0	(In list)

Send Call Word to Dummy Function List

The XS3 code for the symbol is sent to TP2 and the dummy call word to TP3 then

RJ TP TP1

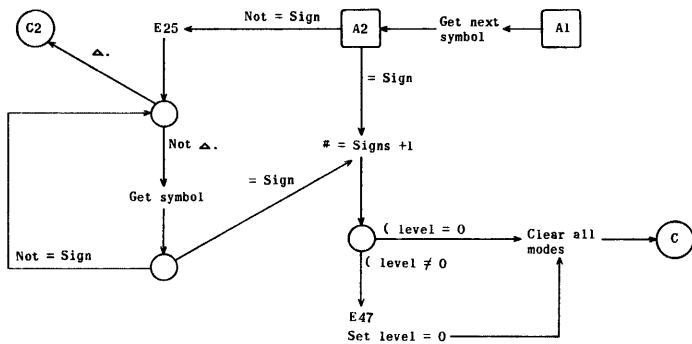
sends the two-word file to the list WL4 - WL23.

Delete Library or Function Arguments, or Subscripts

Sometimes an error is encountered and the arguments of a function or library routine or subscripts of a variable should be ignored without checking so the number of errors doesn't become excessive.

To use the routine

RJ	FV	FV1
MJ	0	(If Δ . is encountered)
MJ	0	(after deletion)



○ ——— decision
□ ——— referenced point

CT.- Trans. Control

A, A1, A2 - decision on 1st
page

etc.

CB - Combination list

Eq. - Equation

Flt pt.- Floating point

(- open parenthesis

or parenthesis

) - closed parenthesis

E27 - Error number 27
(See list of errors for text)

Var. - variable

X - octal digit

△ - Space code

Δ.- Space period

Ftn. - Function

args = arguments

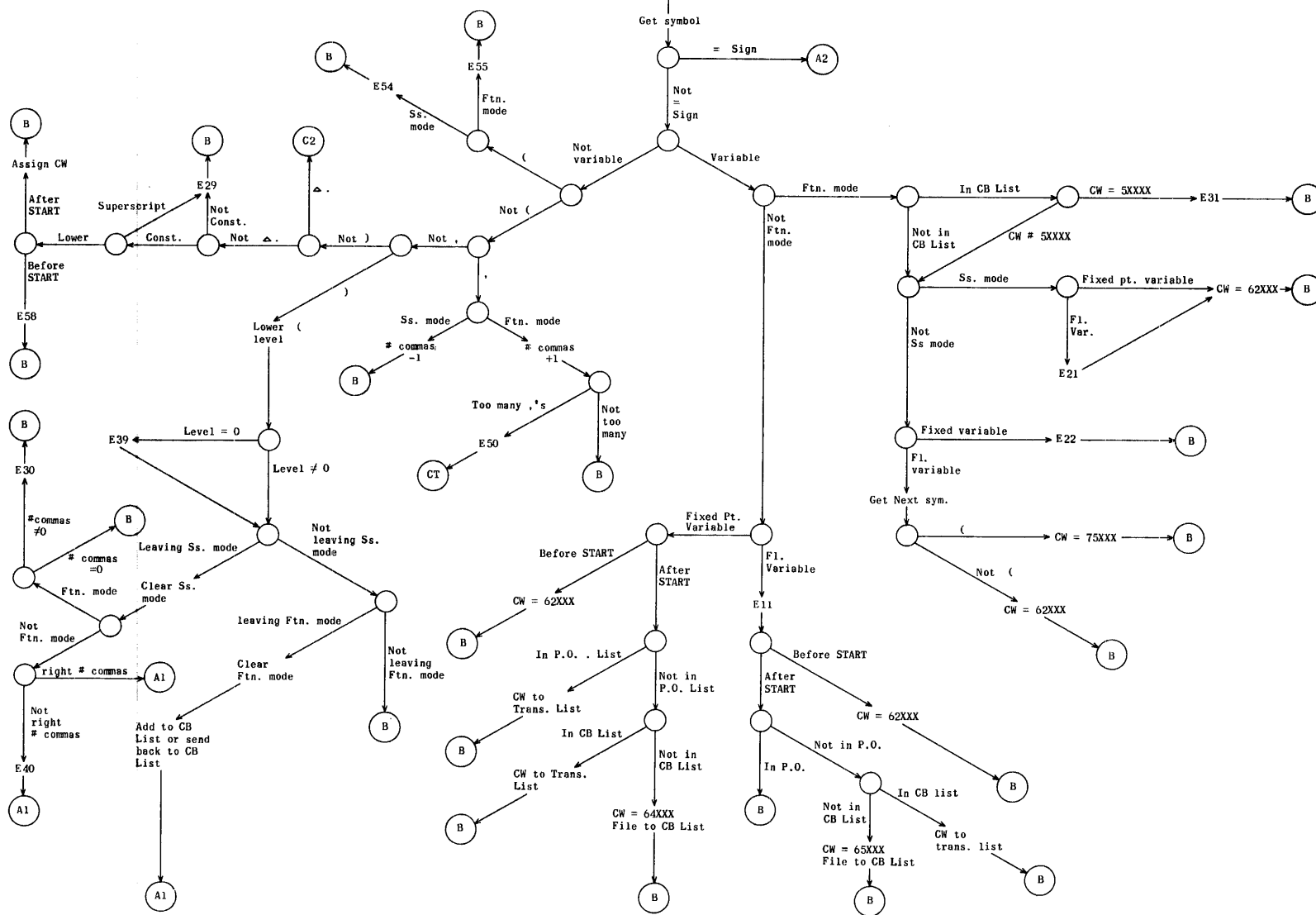
S₂ = Subscriber

B.02 **residue operation**

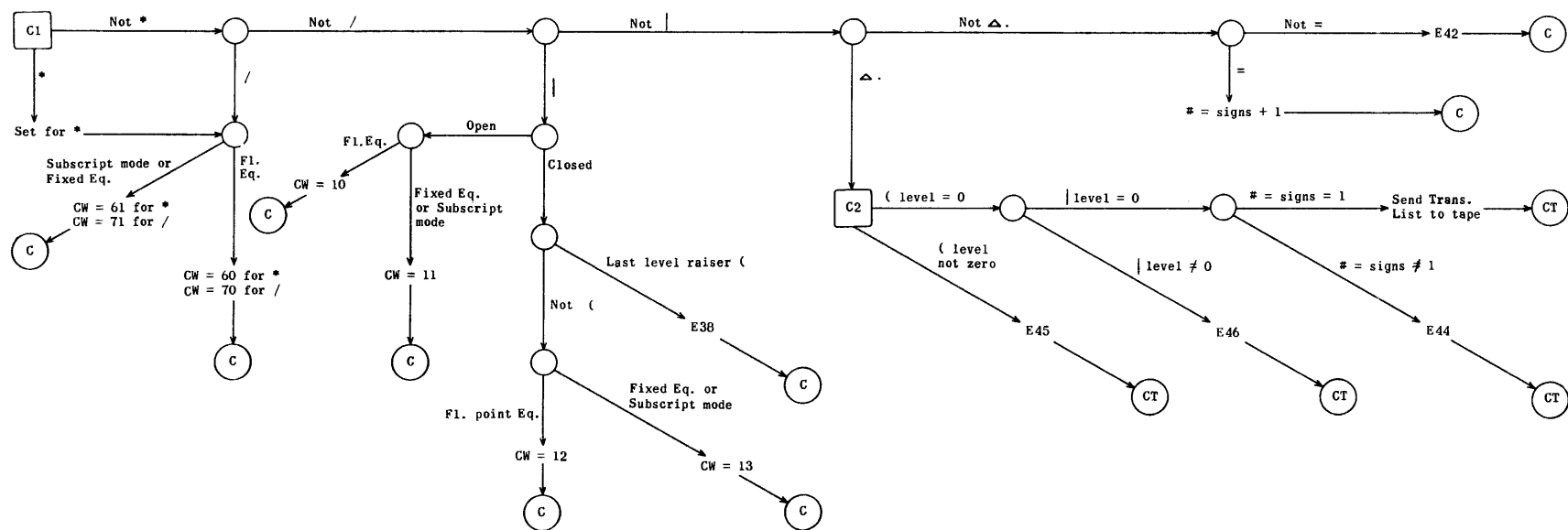
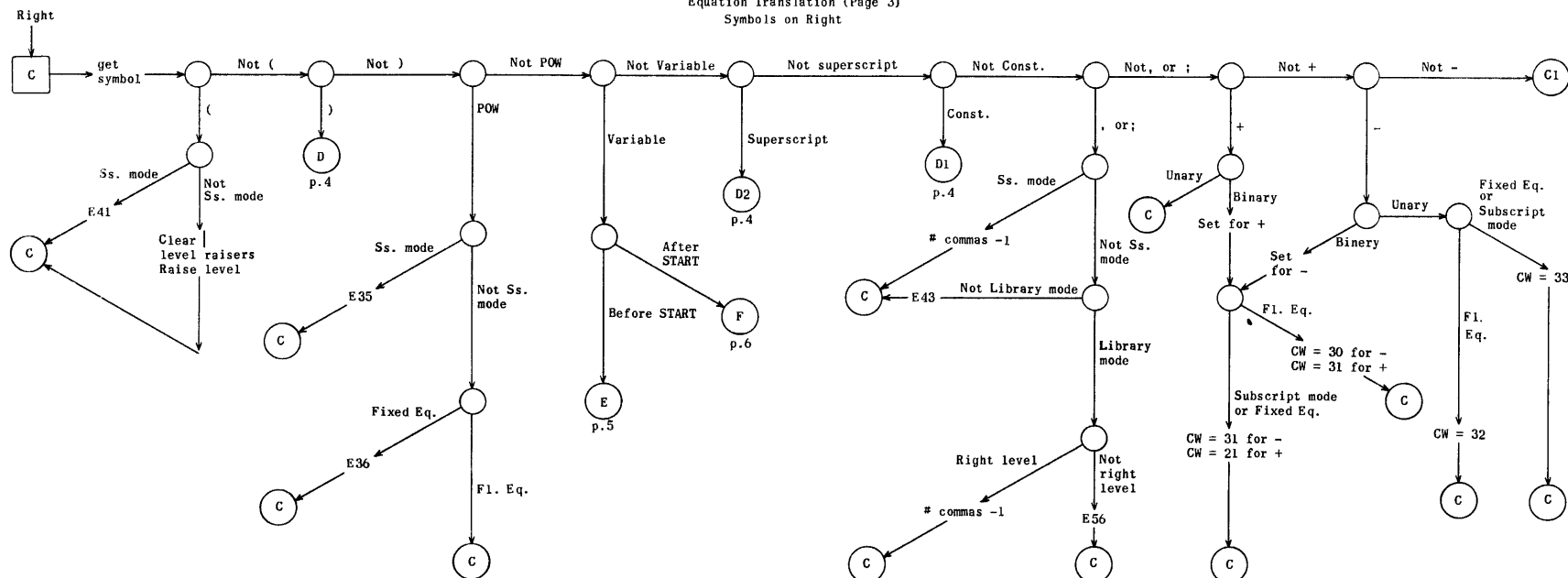
1. *Chlorophyll a* and *Chlorophyll b* were determined by the method of Lichtenthaler and Whistler (1973).

1

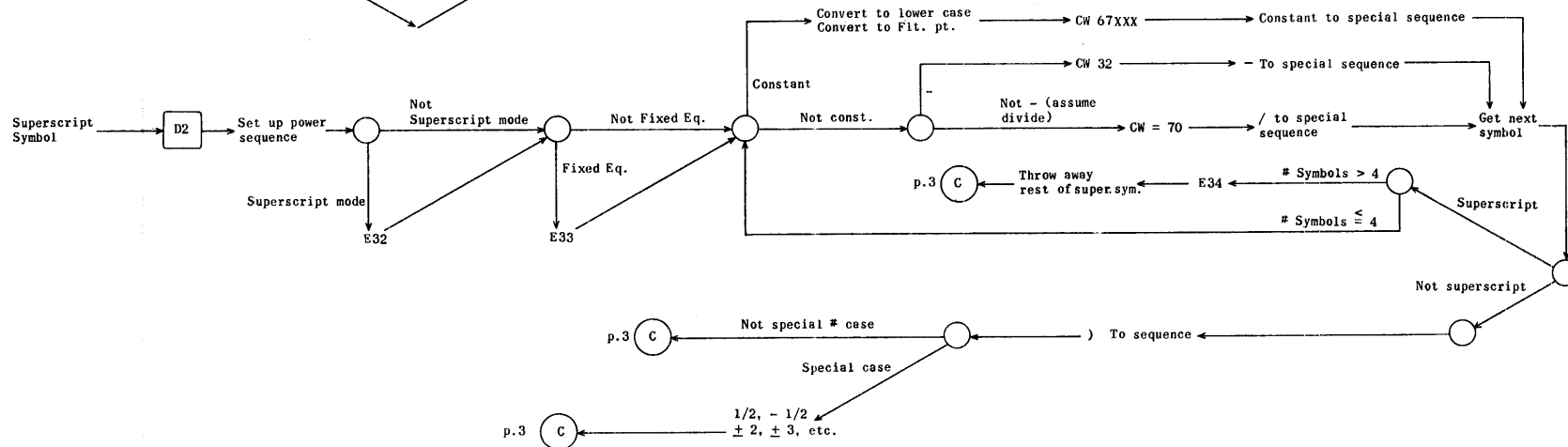
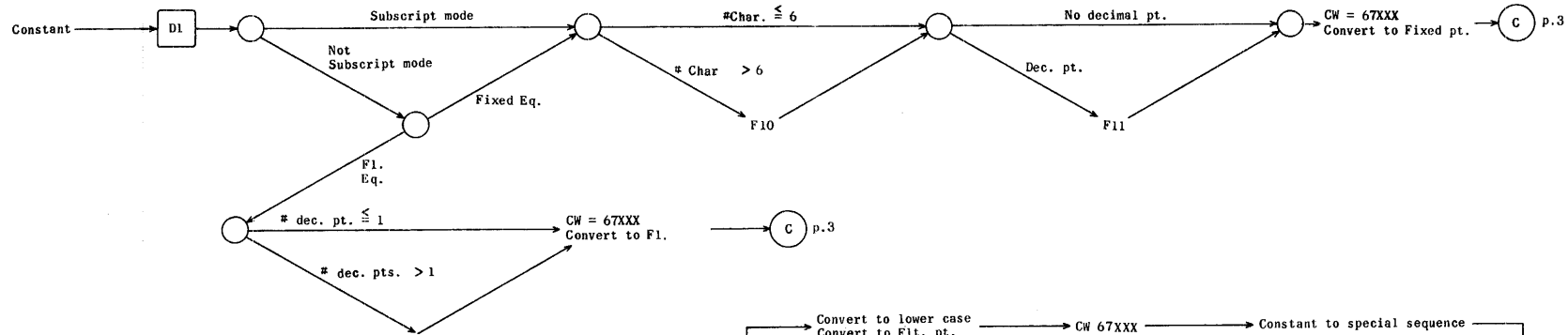
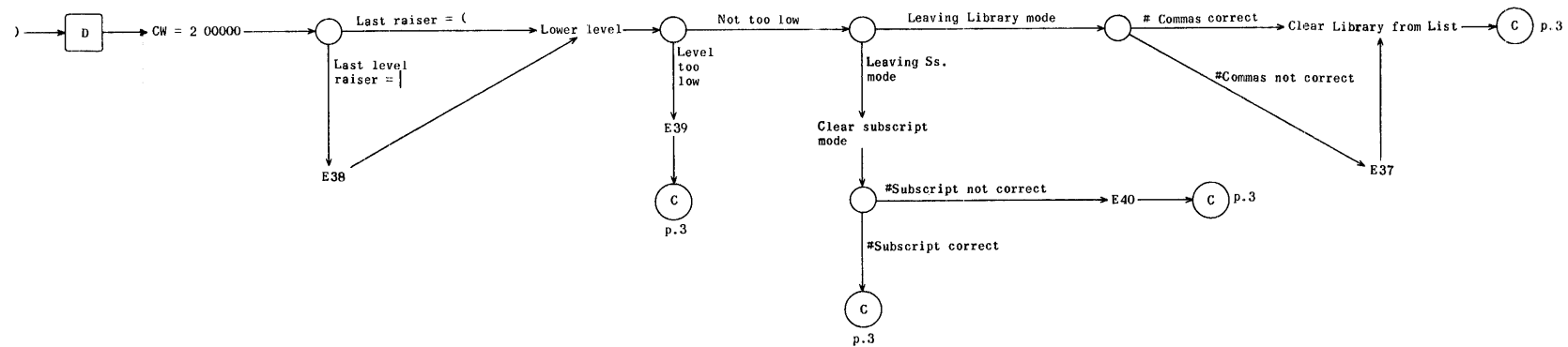
pow = power

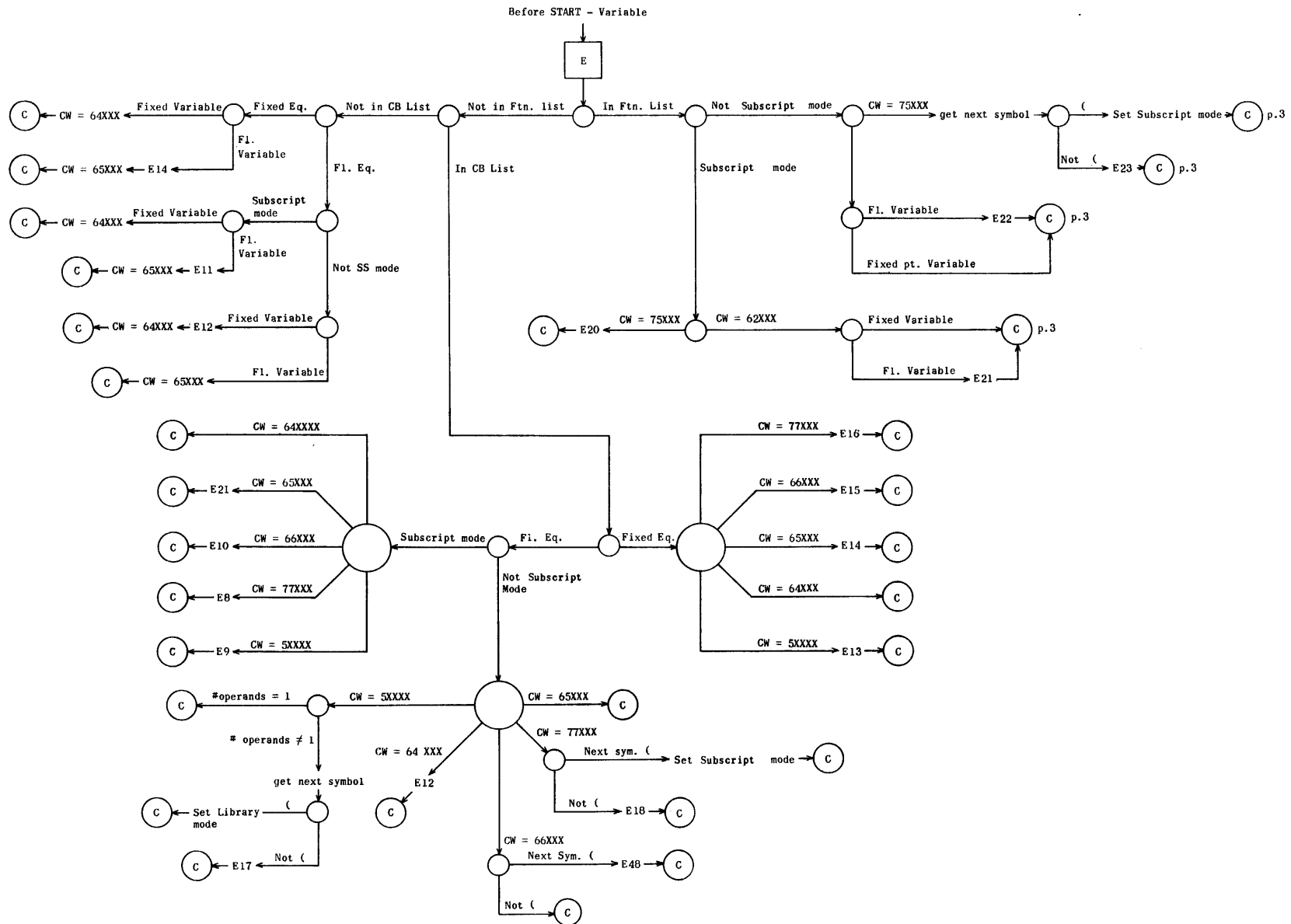


Equation Translation (Page 3)
Symbols on Right



Equation Translation (Page 4)
Symbols on Right (Cont.)







Equation Translation Routine

RE	FR04000		Setups	
RE	CA04027	}		
RE	CG04067			
RE	CH04127			
RE	CI04167		Constants	
RE	CK04227			
RE	DA04243		Left	
RE	FJ04260			
RE	FK04271	}		
RE	FL04303			
RE	FM04313			
RE	FN04323		Right	
RE	FP04337			
RE	FU04343			
RE	FV04361		Subroutine	
RE	HCO4405			
RE	HF04412	}		
RE	HG04417			
RE	HH04432			
RE	HK04444			
RE	HLO4455			
RE	HMO4474			
RE	HNO4506			
RE	HQ04515			
RE	HR04524			
RE	HS04535			
RE	HU04551			
RE	HV04564			
RE	HW04623			
RE	HX04645			
RE	HY04666			
RE	HZ04673			
RE	MA04702			
RE	MB04717			
RE	MC04724			
RE	MDO4753			
RE	ME05001			Right
RE	MG05032			
RE	MH05046			
RE	MI05067			
RE	MJ05211			
RE	MK05220			
RE	ML05236			
RE	MM05253			
RE	MN05300			
RE	MP05310			
RE	MQ05315			
RE	MS05337			
RE	MT05343			

RE	MU05372	}	Right
RE	MV05404		
RE	MW05410		
RE	MX05412		
RE	MY05415		
RE	MZ05431		
RE	NA05434		
RE	NB05447		
RE	NC05461		
RE	ND05471		
RE	NE05502		
RE	NF05511		
RE	NG05532		
RE	NH05551		
RE	NI05576		
RE	NJ05621		
RE	NK05626		
RE	NL05645		
RE	NN05652		
RE	NP05670		
RE	NS05676		
RE	NV05715		
RE	NW05733		
RE	NX05740		
RE	NY05745		
RE	NZ05753		
RE	PC05762		
RE	PD06006		
RE	TP06031		
RE	TU06057		
RE	WC06077		
RE	WD06104		
RE	XC06115		
RE	XD06123		
RE	XE06152		
RE	XF06205		
RE	XG06220		
RE	XH06237		
RE	XI06251		
RE	XL06260		
RE	XQ06276		
RE	XR06317		
RE	XT06344		
RE	XV06362		
RE	XW06400		
RE	XY06405		
RE	XZ06432		
RE	YB06444		
RE	YC06511		
RE	YD06532		
RE	YE06537		
RE	YF06541		

RE	YG06560	}	Left
RE	YH06571		
RE	YI06603		
RE	YJ06611		
RE	YK06623		
RE	YL06644		
RE	YM06673		
RE	YN06705		
RE	YP06723		
RE	YQ06731		
RE	YR06745		
RE	YS06751		
RE	YT06765		
RE	YU07003		
RE	YV07016	△.	
RE	YW07054	}	Left
RE	YX07075		
RE	YY07115		
RE	YZ07135		
RE	ZA07151		
RE	ZB07163		
RE	ZC07212		
RE	ZD07226		
RE	ZE07262		
RE	ZF07271		
RE	ZG07300		
RE	ZH07313		
RE	ZI07330		
RE	ZJ07341		
RE	ZK07346		
RE	ZL07353	}	Right
RE	ZM07364		
RE	ZN07416		
RE	ZQ07434		
RE	ZR07452		
RE	ZT07464		
RE	ZV07510		
RE	ZW07525		
RE	ZX07556		
RE	ZY07571		
RE	ZZ07615	}	Left
RE	KX07620		
RE	KY07645	}	Variables
RE	VA07663		
RE	VC07725	}	Error Text Constants
RE	EF73047		
RE	EG73106		
RE	EH73146		
RE	EI73212		
RE	EJ73251		
RE	EK73300		
RE	EL73334		

RE	EM73362	} Error Text Constants
RE	EN73410	
RE	EP73452	
RE	ER73515	
RE	EX73556	
RE	GA73611	
RE	GB73643	
RE	GC73706	
RE	GD73737	
RE	GE74000	
RE	GF74027	
RE	GH74052	
RE	GI74113	
RE	GJ74151	
RE	GK74202	
RE	GL74225	
RE	GM74255	
RE	GP74306	
RE	GQ74343	
RE	GV74375	
RE	FQ74444	
RE	GY74461	
RE	GZ74510	

Equation Translation - Setups and Subroutines

Region	Name
FR	Setups
TU	Get CW from dummy Ftn. list
TP	Send CW to dummy Ftn. list
XL	Lower (level on left
FV	Delete Lib., Ss., Ftn.
WC	Constants & variables
WD	Superfluous args. error
PC	Symbol pair checker
PD	Constants
MI	Lower upper case XS3 constant

Setups

	IA	FR		
0	MJ	0	CT	Exit
1	RP	10042	FR3	} Clear VA
2	TP	CA27	VA	
3	RP	10040	FR5	} Clear VC
4	TP	CA27	VC	
5	TP	CH15	VA15	= 0 2 2
6	TP	CH31	VA23	= 0 0 100
7	TP	CH22	VA24	= 0 1 0
10	TP	CI31	VA40	= 0 0 7
11	TP	CA3	VA1	= 0 0 1
12	TP	CA3	VA2	= 0 0 1
13	TP	CK1	VC10	= 0 VC10 VC10
14	TP	CK2	VC20	= 0 VC20 VC20
15	TP	CK3	VC30	= 0 VC30 VC30
16	TP	CK4	EW3	EW3 = 0 WL23 WL23
17	TP	CK5	TP4	TP4 = 0 WL4 WL4
20	TP	CA27	XL2	Clear print ind.
21	TP	VD	A	} After ↓ Before → FR24
22	ZJ	FR23	FR24	
23	TP	WB11	WL2	WL2 = EQUATI
24	TP	CK6	VB3	VB3 = -1
25	RP	10020	YB	} Clear dummy list → A
26	TP	CA27	WL4	
	CA	FR27		

Get CW from Dummy Function List

	IA	TU0				RJ	TU	TU1
						MJ	0	Not in list
						MJ	0	In list
0	MJ	0	(30000)	Exit				
1	MJ	0	TU4	Start				
2	0	0	0	XS3				
3	0	0	0	CW				
4	TP	SY2	A	Sym. → A				
5	RP	20020	TU0	Not in list → TU0 (Exit)				
6	EJ	WL4	TU7	In list				
7	SP	TU15	0					
10	SS	Q	17	r → A _u				
11	AT	TU16	TU14	WL3 + r → TU14				
12	RA	TU0	TU17	Set for in list				
13	RP	30002	TU0	File → output → Exit				
14	0	0	0					
15	0	0	20020	JN				
16	TP	WL3	TU2					
17	0	0	1					
	CA	TU20						

Send Dummy CW to Function List

	IA	TP			
0	MJ	0	(30000)	Exit	
1	MJ	0	TP6	Start	
2	0	0	0	XS3	
3	0	0	(0)	CW	
4	0	(WL4)	(WL4)	Add. of next file in dummy list	
5	0	WL24	WL24	Limit add.	
6	TP	TP4	A		
7	TJ	TP5	TP20	O.K. → TP20 No ↓	
10	TP	VA37	Q		
11	QJ	TP12	TP13	Not in CB List ↓ in → TP13	
12	RJ	TE	TE1	Add to list	
13	RJ	WA	WA1		
14	TP	GQ	UP3		
15	TP	VA10	GQ10	E50	
16	RJ	UP2	UP		
17	MJ	0	FR	→ Exit	
20	TV	TP4	TP22	Set address	
21	RP	30002	TP23		
22	TP	TP2	(30000)	File → list	
23	RA	TP4	TP25	Modify	
24	MJ	0	TP	Exit	
25	0	2	2		
	CA	TP26			

Lower (Level on Left

	IA	XL		
0	MJ	0	(30000)	Exit
1	MJ	0	XL3	Start
2	0	0	0	Ind.
3	RS	VA1	CA3	Lower (level
4	TJ	CA3	XL6	0 K ↓ < 1 → XL6
5	MJ	0	XL	Exit
6	TP	CA3	VA1	Set level = 1
7	TP	XL2	Q	Pre. point → exit
10	QJ	XL	XL11	No ↓
11	RJ	WA	WA1	
12	TP	GI	UP3	E39
13	RJ	UP2	UP	
14	TP	CA24	XL2	Set print
15	MJ	0	XL	Exit
	CA	XL16		

Delete LIB, SS, FTN

Y RJ FV FV1
Y+1 MJ 0 (Δ . exit)
Y+2 Normal exit

	IA	FV		
0	MJ	0	(30000)	Exit
1	TP	WC1	WC	Level = 1
2	RJ	SY	SY1	Sym $\rightarrow$ A
3	EJ	WC2	FV7	($\rightarrow$ FV7
4	EJ	WC3	FV11	) $\rightarrow$ FV11
5	EJ	WC4	FV17	Δ . $\rightarrow$ FV17
6	MJ	0	FV2	Return
7	RA	WC	WC1	Raise level
10	MJ	0	FV2	Return
11	RS	WC	WC1	Lower level
12	ZJ	FV2	FV13	Level zero $\downarrow$ no $\rightarrow$ return
13	RA	FV	WC1	Increase exit
14	TP	CA5	A	Set PC with)
15	AT	PD1	PC2	}
16	MJ	0	FV	
17	TP	CA36	A	}
20	AT	PD1	PC2	
21	RA	VA1	WC	Raise level
22	RJ	DA	DA1	Check level
23	MJ	0	FV	$\rightarrow$ Exit
	CA	FV24		

	IA	WC		
0	0	0	0	Level
1	0	0	1	Const.
2	17	77777	77777	(
3	43	77777	77777	)
4	01	22777	77777	Δ .
	CA	WC5		

Superfluous Argument Error

	IA	WD		
0	RJ	WA	WA1	}
1	TP	SZ2	GM7	
2	TP	GM	UP3	
3	RJ	UP2	UP	
4	TP	GP	UP3	}
5	RJ	UP2	UP	
6	RJ	FV	FV1	"Args. not checked"
7	MJ	0	YV	Delete FTN
				Δ . $\rightarrow$ (C2)
10	MJ	0	ZM1	$\rightarrow$ (C)
	CA	WD11		

Pair Checker PC

	IA	PC		
0	MJ	0	(30000)	Exit
1	MJ	0	PC4	Start
2	0	PD22	PD22	Address (left symbol)
3	0	20	20	Shift count (right symbol)
4	TU	PC2	PC5	} Code word → Q
5	TP	(30000)Q		
6	TV	PC3	PC7	} Shift bit → A
7	SP	PD	(30000)	
10	QT	A	A	Ind. → A
11	ZJ	PC12	PC21	OK → PC21 No ↓
12	RJ	WA	WA1	} E52
13	RP	30003	PC15	
14	TP	SZ2	FQ6	
15	RP	30003	PC17	
16	TP	SY2	FQ12	} Right sym.
17	TP	FQ	UP3	
20	RJ	UP2	UP	
21	SP	PD1	0	
22	AT	PC3	PC2	} Set for next check
23	MJ	0	PC	
	CA	PC24		Exit

	IA	PD		
0	0	0	1	Shift bit
1	0	PD2	PD2	1st address of code words
2	0	0	67642	(
3	0	1	12035	) or (closed)
4	0	1	12035	Var. or const.
5	0	1	12034	Sub. Var.
6	0	1	12034	FTN
7	0	0	67642	, or ;
10	0	0	47742	+ or -
11	0	0	67642	* or /
12	0	1	11435	Up const.
13	0	1	57377	Up -
14	0	1	77377	Up /
15	0	0	47642	POW
16	0	0	67642	LIB
17	0	0	67642	=
20	0	1	77777	Δ.
21	0	0	47642	(open)
22	0	0	0	"Doesn't matter"
	CA	PD23		

Conversion of Upper Case XS-3 Constant to Lower Case

	IA	MI		
0	MJ	0	(30000)	
1	MJ	0	MI6	
2	0	0	0	} Output data
3	0	0	0	
4	0	0	0	} Input data
5	0	0	0	
6	TP	MI121	MI104	Zeroize temp.
7	TP	MI121	MI105	Zeroize temp.
10	TP	MI4	MI46	Store input
11	TP	MI5	MI47	Store input
12	TP	MI103	MI110	Set Index
13	TP	MI106	MI107	Set Index
14	TP	MI111	MI25	Set store command
15	LQ	MI46	00006	} Convert and Store
16	QT	MI50	A	
17	RP	20013	MI25	
20	EJ	MI51	MI21	
21	TP	MI101	A	
22	ST	Q	MI117	
23	LA	MI117	00017	
24	RA	MI25	MI117	
25	TP	MI64	Q	
26	QT	MI50	MI45	
27	LQ	MI104	00006	} Reset for second word
30	RA	MI104	MI45	
31	IJ	MI107	MI14	
32	RA	MI30	MI102	
33	RA	MI27	MI102	
34	RA	MI15	MI102	} Preset for second set of input data
35	IJ	MI110	MI13	
36	TP	MI104	MI2	
37	TP	MI105	MI3	
40	TP	MI111	MI25	
41	TP	MI112	MI15	
42	TP	MI113	MI30	
43	TP	MI114	MI27	
44	MJ	0	MI	
45	0	0	0	
46	0	0	0	
47	0	0	0	
50	0	0	00077	
51	0	0	00060	Upper Case ↓
52	0	0	00061	
53	0	0	00040	
54	0	0	00020	
55	0	0	00041	
56	0	0	00035	
57	0	0	00055	

60	0	0	00075
61	0	0	00036
62	0	0	00057
63	0	0	00062
64	0	0	00077
65	0	0	00003
66	0	0	00004
67	0	0	00005
70	0	0	00006
71	0	0	00007
72	0	0	00010
73	0	0	00011
74	0	0	00012
75	0	0	00013
76	0	0	00014
77	0	0	00022
100	0	0	00077
101	0	0	20013
102	0	00001	0
103	0	0	00001
104	0	0	0
105	0	0	0
106	0	0	00005
107	0	0	0
110	0	0	0
111	TP	MI64	Q
112	LQ	MI46	00006
113	RA	MI104	MI45
114	LQ	MI104	00006
115	PR	0	MI120
116	MS	0	40000
117	0	00007	0
120	0	0	00015
121	0	0	0
	CA	MI122	

Lower Case

Lower Case

Parameter Error

Constants

	IA	CA0			
0	04	0	0		
1	06	0	0		
2	0	0	77777		
3	0	0	1		
4	0	0	0	(	
5	0	1	1	) or (closed)	
6	0	2	2	Variable or const.	
7	0	3	3	Ss. Var.	
10	0	4	4	FTN	
11	0	5	5	, or ;	
12	0	6	6	+ or -	
13	0	7	7	* or /	
14	0	10	10	Upper Const.	
15	0	11	11	Upper -	
16	0	12	12	Upper /	
17	0	13	13	POW	
20	0	14	14	LIB	
21	0	15	15	BY or =	
22	0	16	16	Δ .	
23	0	17	17	(open)	
24	40	0	0		
25	0	0	64000		
26	0	3	3		
27	0	0	0	Zero	
30	17	77777	77777	(77 77 77 77 77	
31	0	0	65000		
32	0	0	777		
33	0	0	66000		
34	06	0	3		
35	0	0	62000		
36	0	20	20	O.K. for P.C.	
37	0	0	01000		
	CA	CA40			

for PC

Constants (cont.)

	IA	CG		
0	11	11111	11111	Fltpt.
1	22	22222	22222	Fixed pt.
2	33	33333	33333	Ftn.
3	44	44444	44444	Sub. var.
4	21	77777	77777	, 77 77 77 77 77
5	23	77777	77777	; 77 77 77 77 77
6	43	77777	77777	) 77 77 77 77 77
7	76	77777	77777	= <u> </u> →
10	25	73777	77777	BY <u> </u> →
11	01	22777	77777	Δ. <u> </u> →
12	00	77777	77777	- 77 77 77 77 77
13	15	77777	77777	/ 77 77 77 77 77
14	42	77777	77777	<u> </u> →
15	0	0	50	=
16	0	0	120	Δ.
17	0	0	77000	
20	0	0	70	Mask
21	0	0	61000	
22	0	0	63000	
23	0	0	76000	
24	0	1	0	(
25	0	0	00700	Mask
26	0	VL1	VL1	
27	0	07777	0	Mask
30	0	0	70000	
31	0	0	40000	
32	0	0	50000	
33	02	0	0	
34	0	2	0	)
35	0	70000	0	
36	0	0	40	,
37	0	10000	0	
	CA	CG40		

Constants (cont.)

	IA	CHO		
0	0	CH35	0	Fltpt. /
1	0	CI11	0	Fixed pt. /
2	0	CI13	0	Fltpt. *
3	0	CI12	0	Fixed pt. *
4	0	0	75000	
5	52	51717	77777	PO W 77 77 77
6	63	77777	77777	+ 77 77 77 77 77
7	02	77777	77777	- $\longrightarrow$
10	56	77777	77777	* $\longrightarrow$
11	64	77777	77777	/ $\longrightarrow$
12	0	0	12	(closed) floating
13	0	0	13	(closed) fixed
14	0	0	10	(open) fl. and fixed
15	0	2	2	
16	0	0	5	
17	0	0	VA24	Base add. for sequence
20	0	0	VA31	for comparison
21	TP	CG34	0	
22	0	1	0	
23	0	1	1	
24	0	0	3	
25	0	0	2	
26	0	CIO	0	
27	0	VA32	0	
30	0	0	16100	
31	0	0	100	
32	0	0	77	
33	0	0	16000	
34	0	CI1	0	
35	0	0	70	/
36	0	0	32	-
37	0	0	00100	To set VC3 & POW
	CA	CH40		

Constants (cont.)

	IA	CIO	
0	00	77777	77777 -
1	20	14000	0 1
2	15	77777	77777 /
3	20	24000	0 2
4	20	26000	0 3
5	0	0	13000 CW for 4 to 63 & -4 to -63
6	0	0	15000 2 & -2
7	0	0	14000 3 & -3
10	0	0	17100 -1
11	0	0	71 Fixed /
12	0	0	61 Fixed *
13	0	0	60 Fl. *
14	0	0	60000
15	0	0	10000
16	0	0	33 Fixed unary -
17	0	0	32
20	0	0	31
21	0	0	30
22	0	0	21 Fixed +
23	0	0	20 Fl. +
24	0	3	3
25	0	0	75000
26	30	0	0
27	0	0	0 Base add. of print list
30	0	20000	0
31	0	0	7 Mask
32	0	VC20	VC20 } Limits on LIB modes
33	0	VC30	VC30 }
34	0	07000	0
35	0	CI20	0 Fixed -
36	0	CI21	0 Fl. -
37	0	CI22	0 Fixed +
	CA	CI40	

Constants (cont.)

	IA	CKO		
0	0	CI23	0	F1. +
1	0	VC10	VC10	
2	0	VC20	VC20	
3	0	VC30	VC30	
4	0	WL23	WL23	Set EW3
5	0	WL4	WL4	
6	77	77777	77776	
7	0	0	25000	64,65,66xxx separate Eq.
10	0	0	24000	77xxx separate Eq.
11	0	0	WL	
12	0	0	67000	Const. CW
13	0	30000	0	Const.
	CA	CK14		

Variables for Equation Translation

	IA	VA0		
0	0	0	0	Print ind.
1	0	0	1	(level bit
2	0	0	1	1 level bit
3	0	0	0	# (level raisers in sequence
4	0	0	0	# 1 level raisers in sequence
5	0	0	0	Ss. mode level
6	0	0	0	XS3 of Ss. variable
7	0	0	0	Ftn. mode level
10	0	0	0	XS3 of Ftn.
11	0	0	0	Level of subscripts for FTN.
12	0	0	0	Lib. mode
13	0	0	0	# commas for Lib.
14	0	0	0	# of '='s
15	0	2	2	# of words in up. c. sequence for string-out
16	0	0	0	Fixed 40 - floating 00
17	0	0	0	# commas, SV mode (count)
20	0	0	0	# commas, FTN mode (count)
21	0	0	0	# commas, Lib. mode (count)
22	0	0	0	# upper case symbols in sequence
23	0	0	100	POW (
24	0	1	0	Upper case sequence
25	0	0	0	for string-out
26	0	0	0	(Call words)
27	0	0	0	
30	0	0	0	
31	0	0	0	
32	0	0	0	Upper case sequence
33	0	0	0	for special call word
34	0	0	0	comparison (XS3)
35	0	0	0	
36	0	0	0	# Ss. -1 (from dimension list)
37	0	0	0	not in CB List bit
40	0	0	7	Ftn. mode format mask
41	0	(0)	0	Address in CB List
	CA	VA42		

Variables (cont.)

	IA	VC		
0	0	0	0	Any decimal points in sequence
1	0	0	0	Index
2	0	0	0	Constant CW temp.
3	0	0	0	Add in to constant CW for -
4	0	0	0	Save const. CW
5	0	0	0	Ind. print for "more than 29 ('s etc."
6	0	0	0	
7	0	0	0	
10	0	VC10	VC10	Address of Lib. level
11	0	0	0	} Level of Lib.
12	0	0	0	
13	0	0	0	
14	0	0	0	
15	0	0	0	
16	0	0	0	
17	0	0	0	
20	0	VC20	VC20	Address of # of commas for Lib.
21	0	0	0	} # commas for Lib.
22	0	0	0	
23	0	0	0	
24	0	0	0	
25	0	0	0	
26	0	0	0	
27	0	0	0	
30	0	VC30	VC30	} XS3 of Lib.
31	0	0	0	
32	0	0	0	
33	0	0	0	
34	0	0	0	
35	0	0	0	
36	0	0	0	
37	0	0	0	
	CA	VC40		

Equation Translation Left

Section A - First Symbol, Before START

Section B - First Symbol, After START

Section C - Not First Symbol

Region	Section	Region	Section
YB	A		
YH	A	ZE	C
YK	A	ZD	C
YG	A	ZI	C
YF	A	ZJ	C
YJ	A	ZK	C
YE	A	ZL	C
YD	A	DA	C
YI	A	KX	C
YC	A	KY	C
YM	A		
YN	A		
YW	B		
YS	B	Letters on left of coding sheets are connectors (They are also on the flow charts) i.e., (B), (A₂), etc.	
YR	B		
YQ	B		
YP	B		
YU	B		
YT	B		
		Connector	Section
YX	C	(A)	A
ZA	C		
ZC	C	(A ₁)	A
ZB	C		
YZ	C	(A ₂)	A
YY	C		
YL	C	(B)	C
ZF	C		
ZG	C		
ZH	C		

Left-Before START-1st Symbol

(A)	0	IA	YB		
		TP	VD	A	} After → YW
	1	ZJ	YW	YB2	
					} Before ↓
	2	RJ	TA	TA1	
					} Not in CB List → YC
	3	MJ	0	YC	
					} In ↓
	4	TP	TA4	Q	
					} Lib. → YD
	5	QT	CG30	A	
					} No ↓
	6	EJ	CG32	YD	
					} CW → string
	7	TP	TA4	EW2	
	10	RJ	EW	EW1	
					} CW = 77XXX → YB17
	11	TP	TA4	Q	
					} No ↓
	12	QT	CG17	A	
					} 25XXX CW → WL3
	13	EJ	CG17	YB17	
					} 24XXX CW → WL3
	14	QT	CA32	A	
					} No prev. Eq. → YI
	15	AT	CK7	WL3	
					} Prev. Eq. ↓
	16	MJ	0	YB21	
					} E24
	17	QT	CA32	A	
					} CW → A
	20	AT	CK10	WL3	
					} 64XXX → YE
	21	TP	TA5	Q	
					} 65XXX → YH
	22	QT	CA1	A	
					} 66XXX → YG
	23	ZJ	YB24	YI	
					} Sym. → A 77XXX ↓
	24	RJ	WA	WA1	
					} → YJ No ↓
	25	TP	SY2	EX31	
					} E18
	26	TP	EX22	UP3	
					} Sym. → A
	27	RJ	UP2	UP	
					} → (A1)
	30	TP	TA4	Q	
					} 64XXX → YE
	31	QT	CG17	A	
					} 65XXX → YH
	32	EJ	CA25	YE	
					} 66XXX → YG
	33	EJ	CA31	YH	
					} Sym. → A 77XXX ↓
	34	EJ	CA33	YG	
					} → YJ No ↓
	35	RJ	SY	SY1	
					} E18
	36	EJ	CA30	YJ	
					} Sym. → A
	37	RJ	WA	WA1	
					} → (A1)
	40	TP	SZ2	EN30	
					} 64XXX → YE
	41	TP	EN22	UP3	
					} 65XXX → YH
	42	RJ	UP2	UP	
					} 66XXX → YG
	43	TP	SY2	A	
					} Sym. → A
	44	MJ	0	YH10	
					} → (A1)
		CA	YB45		

End of Left Symbols

(A2) {	0	IA	YH		
	1	RJ	SY	SY1	
	2	EJ	CG7	YK	= → YK
	3	RJ	WA	WA1 }	
	4	TP	GA	UP3 }	E25
	5	RJ	UP2	UP }	
(A1) {	6	TP	SY2	A }	
	7	EJ	CG11	YV	Δ. → (C2)
	10	RJ	SY	SY1	Sym → A
	11	EJ	CG7	YK	= → YK
		MJ	0	YH6	
		CA	YH12		

0	IA	YK	CA3	# = 's + 1
1	RA	VA14	A	
2	TP	VA1	YK7	level zero → YK7 no ↓
3	EJ	CA3	WA1 }	
4	RJ	WA	UP3 }	E47
5	TP	GL14	UP }	
6	RJ	UP2	VA1 }	set to (level zero
7	TP	CA3	YK11 }	
10	RP	10011	VA3 }	Clear all modes
11	TP	CA27	YK13 }	
12	RP	10004	VA17 }	
13	TP	CA27	VA36 }	Clear #SS
14	TP	CA21	A }	set left of PC with =
15	AT	PD1	PC2 }	
16	TP	CG15	EW2 }	CW → list
17	RJ	EW	EW1 }	
20	MJ	0	ZM1	→ (C)
	CA	YK21		

66XXX

	IA	YG		
0	RJ	SY	SY1	Sym → A
1	EJ	CA30	YG7	(→ YG7 no ↓
2	RJ	WA	WA1	
3	TP	SZ2	GA11	E26
4	TP	GA6	UP3	
5	RJ	UP2	UP	
6	MJ	0	FR	→ Exit to trans. control
7	LQ	TA31	25	
10	MJ	0	YF4	
	CA	YG11		

65XXX

	IA	YF		
0	RJ	SY	SY1	Sym → A
1	EJ	CA30	YF3	(→ YF3 no ↓
2	MJ	0	YH1	→ (A2)
3	RA	TA4	CA37	CW = 66XXX
4	TP	TA31	VA41	Save address in VA41
5	RA	VA41	CA3	
6	TP	VA1	VA7	Set Ftn. mode
7	TP	SZ2	VA10	XS3 of Ftn.
10	RA	VA1	CA3	Raise (level
11	TP	PD1	PC2	Set PC
12	TP	CA26	TF	# words = 3
13	TP	CA34	TF3	Set format
14	TP	CG20	VA40	Set mask
15	RP	30002	YX1	Set up to return to
16	TP	TA3	TF1	CB list then → (B)
	CA	YF17		

77XXX (

	IA	YJ		
0	TP	VA1	VA5	Set Ss. mode
1	TP	TA5	Q	#Ss. → VA 36
2	QT	CA2	VA36	
3	RA	VA1	CA3	Raise level
4	TP	PD1	PC2	Set PC
5	TP	CG24	EW2	CW → list
6	RJ	EW	EW1	
7	RS	VA36	CA3	#Ss. -1
10	TP	SZ2	VA6	XS3 of Ss. variable
11	MJ	0	YX1	→ (B)
	CA	YJ12		

64XXX

	IA	YE		
0	TP	CA24	VA16	Set fixed equation
1	MJ	0	YH	→ (A2)
	CA	YE2		

Lib. = 5XXXX

	IA	YD		
0	RJ	WA	WA1	
1	TP	SY2	GB5	E27
2	TP	GB	UP3	
3	RJ	UP2	UP	
4	MJ	0	FR	→ trans. control
	CA	YD5		

No previous Equation

	IA	YI	
0	RA	TA5	CA1
1	RJ	TD	TD1
2	TP	TA4	Q }
3	QT	CG17	A }
4	EJ	CA31	YF
5	MJ	O	YB30
	CA	YI6	

Set eq.
Return to list

CW → A
65XXX → YF no ↓

Not in CB List

	IA	YC	
0	RJ	RH	RH1
1	RJ	TK	TK1
2	AT	CK7	WL3
3	TP	SY10	Q }
4	QJ	YM	YC5 }
5	RJ	SY	SY1
6	EJ	CA30	YN
7	TP	VB1	A }
10	AT	CA31	EW2 }
11	TP	A	TF2 }
12	RJ	EW	EW1 }
13	TP	CA26	TF }
14	TP	SZ2	TF1 }
15	TP	CA1	TF3 }
16	RJ	TE	TE1 }
17	TP	SY2	A
20	MJ	O	YH1
	CA	YC21	

Check Var. symbol
Increase CW.
25XXX CW → WL3
IJKLM → YM
No ↓
Sym → A
(→ YN no ↓

CW = 65XXX
CW → list
File → CB list

Sym → A
→ (A2)

Not in CB List 1st Letter IJKLM

	IA	YM		
0	TP	VB1	A	} 64XXX
1	AT	CA25	EW2	
2	TP	A	TF2	
3	RJ	EW0	EW1	CW → list
4	TP	CA24	VA16	Set fixed equation
5	TP	SY2	TF1	Sym. → TF1
6	TP	CA26	TF0	} CW → CB List
7	TP	CA1	TF3	
10	RJ	TE0	TE1	
11	MJ	0	YH	→ (A2)
	CA	YM12		

NOT IJKLM Next Sym. (

	IA	YN		
0	TP	VB1	A	} 66XXX
1	AT	CA33	EW2	
2	TP	A	TF2	
3	RJ	EW0	EW1	CW → list
4	TP	SZ2	TF1	XS3 of Ftn.
5	TP	SZ2	VA10	} Store format
6	TP	CA34	TF3	
7	TP	CA26	TF0	
10	TP	VA1	VA7	Set Ftn.
11	RA	VA1	CA3	Increase level
12	TP	PD1	PC2	Set P. C.
13	TP	CA24	VA37	Set "not in CB List" bit
14	TP	CG20	VA40	Set mask
15	MJ	0	YX1	→ (B)
	CA	YN16		

Left After Start 1st symbol

	IA	YW		
0	RJ	TS	TS1 }	Not in P. Op. list → YP
1	MJ	0	YP }	In ↓
2	TP	TS3	EW2 }	CW → list
3	RJ	EW	EW1 }	
4	RJ	KY	KY1 }	Check Vary List
5	TP	TS3	Q }	XX000 → A
6	QT	CG17	A }	
7	EJ	CG21	YQ	61XXX → YQ
10	EJ	CG22	YR	63XXX → YR
11	RJ	SY	SY1	76YXX ↓ Sym → A
12	EJ	CA30	YS	(→ YS no ↓
13	RJ	WA	WA1 }	
14	TP	SZ2	EP6 }	E19
15	TP	EP	UP3 }	
16	RJ	UP2	UP }	
17	TP	SY2	A	Sym → A
20	MJ	0	YH1	→ (A2)
	CA	YW21		

	IA	YS0		
0	TP	CG24	EW2 }	(→ string
1	RJ	EW0	EW1 }	
2	TP	SZ2	VA6 }	Set Ss. mode
3	TP	VA1	VA5 }	
4	RA	VA1	CA3 }	Raise level
5	TP	TS3	Q }	
6	QT	CG25	Q }	Store # Ss.
7	LQ	Q	36 }	
10	TP	Q	VA36 }	
11	TP	PD1	PC2	Set P.C.
12	RS	VA36	CA3	Ss. - 1
13	MJ	0	YX1	→ (B)
	CA	YS14		

63XXX ↓

	IA	YR0		
0	TP	SY10	Q }	1st letter IJKLM → YR2
1	QJ	YR2	YH }	No → (B)
2	TP	CA24	VA16	Set fixed eq.
3	MJ	0	YH	→ (B)
	CA	YR4		

61XXX

	IA	YQ		
0	RJ	SY	SY1	Sym → A
1	EJ	CA30	YQ3	(→ YQ3 no ↓
2	MJ	0	YH1	→ (A2)
3	RJ	WA	WA1 }	
4	TP	SZ2	GM7 }	E48
5	TP	GM	UP3 }	
6	RJ	UP2	UP }	
7	TP	GP	UP3 }	"Args. not checked"
10	RJ	UP2	UP }	
11	RJ	FV	FV1	Delete Ftn.
12	MJ	0	YV	Δ. → (C2)
13	MJ	0	YH	→ (A2)
	CA	YQ14		

Not in P.Op. List-After START-1st Sym

	IA	YP		
0	RJ	TA	TA1 }	Not in CB List → YT
1	MJ	0	YT }	In ↓
2	TP	TA4	EW2 }	CW → list
3	RJ	EW	EW1 }	
4	RJ	KY	KY1	Check Vary List
5	MJ	0	YU	
	CA	YP6		

Not in Vary List

	IA	YU		
0	TP	TA4	Q }	
1	QT	CG30	A }	X0000 → A
2	EJ	CG32	YD	5XXXX → YD
3	QT	CG17	A	XX000 → A
4	EJ	CA25	YE	64XXX → YE
5	EJ	CA31	YH	65XXX → YH
6	EJ	CA33	YQ	66XXX → YQ
7	TP	CG33	Q	77XXX ↓
10	QS	CG33	TA5	Set Equation bit
11	RJ	TD	TD1	Return to CB List
12	MJ	0	YB35	
	CA	YU13		

Not in CB List

	IA	YT		
0	RJ	TK	TK1	Increase CW counter
1	TP	SY10	Q }	1st letter IJKLM → YT3
2	QJ	YT3	YT14 }	No → YT14
3	AT	CA25	EW2 }	CW = 64XXX
4	TP	CA24	VA16 }	Set fixed Equation
5	TP	A	TF2 }	
6	TP	CA26	TF }	Build file
7	TP	SY2	TF1 }	
10	TP	CG33	TF3 }	
11	RJ	EW	EW1	CW → list
12	RJ	TE	TE1	File → CB List
13	MJ	0	YH	→ A2
14	AT	CA31	EW2	CW = 65XXX
15	MJ	0	YT5	
	CA	YT16		

Left - Not 1st symbol

(B)	{	0	IA	YX	PC1	Pair Check
		1	RJ	PC	SY1	Sym $\rightarrow$ A
		2	RJ	SY	YK	= $\rightarrow$ YK
		3	EJ	CG7	Q	
		4	TP	SY7	YX5 }	Var. $\rightarrow$ YL no $\downarrow$
		5	QJ	YL	YY	($\rightarrow$ YY
		6	EJ	CA30	YZ }	
		7	EJ	CG4	YZ }	, or; $\rightarrow$ YZ
		10	EJ	CG5	ZA	) $\rightarrow$ ZA
		11	EJ	CG6	KX	Δ . $\rightarrow$ KX $\rightarrow$ (C2)
		12	MJ	0	GC4	
		13	TP	SY2	WA1	
		14	RJ	WA	UP3 }	E29
		15	TP	GC	UP	
		16	RJ	UP2	PC3	Set PC
		17	TP	CA36	YX	$\rightarrow$ (B)
			MJ	0		
			CA	YX20		

Left, not 1st sym.

	IA	ZA		
0	TP	CA5	PC3	Set PC
1	RJ	XL	XL1	Level checker
2	TP	VA1	A	No ↓
3	EJ	VA5	ZB	Leaving Ss. mode → ZB
4	EJ	VA7	ZC	Leaving Ftn. mode → ZC
5	TP	VA7	A	In Ftn. mode → YX
6	ZJ	YX	ZA7	No ↓
7	TP	CG34	EW2	CW → list
10	RJ	EW	EW1	
11	MJ	0	YX	→ (B)
	CA	ZA12		

) Leaving Ftn. mode

	IA	ZC		
0	TP	CA27	VA7	Clear Ftn. mode
1	RJ	PC	PC1	→ PC
2	TP	CG35	Q	# commas → list
3	QS	VA20	TF3	
4	TP	VA37	Q	"Not in CB List" bit set ↓
5	QJ	ZC6	ZC11	Not set → ZC11
6	RJ	TE	TE1	Add to CB List
7	TP	CA27	VA37	Clear "not" bit
10	MJ	0	YH	→ (A2)
11	TV	VA41	ZC13	Send back to CB List
12	RP	30003	YH	→ (A2)
13	TP	TF1	(30000)	
	CA	ZC14		

Left)-leaving Ss. mode

	IA	ZB		
0	RJ	PC	PC1	Do PC
1	TP	CA27	VA5	Clear Ss. mode
2	TP	VA7	A	
3	ZJ	ZB4	ZB14	Ftn. mode ↓ no → ZB14
4	TP	VA36	A	# commas = 0 → (B)
5	ZJ	ZB6	YX1	No ↓
6	RJ	WA	WA1	
7	TP	VA6	GC20	
10	TP	GC12	UP3	E30
11	TP	VA10	GC27	
12	RJ	UP2	UP	
13	MJ	0	YX1	→ (B)
14	TP	CG34	EW2	CW → list
15	RJ	EW	EW1	
16	TP	VA36	A	Right # commas → ZB24
17	ZJ	ZB20	ZB24	No ↓
20	RJ	WA	WA1	
21	TP	VA6	GI17	E40
22	TP	GI12	UP3	
23	RJ	UP2	UP	
24	TP	CA27	VA17	Clear # Ss.
25	TP	CA27	VA36	
26	MJ	0	YH	→ (A2)
	CA	ZB27		

Left , or ;

	IA	YZ0		
0	TP	CA11	PC3	Set PC
1	TP	VA5	A	
2	ZJ	YZ3	YZ7	Ss. mode ↓ no → YZ7
3	RS	VA36	CA3	Commas - 1
4	TP	CG36	EW2	C.W. → list
5	RJ	EW0	EW1	
6	MJ	0	YX	→ (B)
7	RA	VA20	CG37	Commas + 1
10	TP	CK13	A	Too many operands → TP10
11	TJ	VA20	TP10	No ↓
12	LQ	VA40	3	Shift mask
13	MJ	0	YX	→ (B)
	CA	YZ14		

Left (

	IA	YY		
0	TP	CA4	PC3	Set PC
1	RA	VA1	CA3	Increase level
2	TP	VA7	A	
3	ZJ	YY4	YY11	Ftn. mode ↓ Ss. mode → YY11
4	RJ	WA	WA1	
5	TP	VA10	GV45	E55
6	TP	GV31	UP3	
7	RJ	UP2	UP	
10	MJ	0	YX	→ (B)
11	RJ	WA	WA1	
12	TP	VA6	GV27	E54
13	TP	GV15	UP3	
14	RJ	UP2	UP	
15	TP	CG24	EW2	CW → list
16	RJ	EW	EW1	
17	MJ	0	YX	→ (B)
	CA	YY20		

Left-not First Symbol-Var.

	IA	YLO		
0	RJ	RH0	RH1	Check symbol
1	TP	CA6	PC3	Set PC
2	TP	VA7	A	Not Ftn. mode → ZD
3	ZJ	YL4	ZD	Ftn. mode ↓
4	RJ	TA0	TA1	Not in CB List → YL11
5	MJ	0	YL11	In ↓
6	TP	TA4	Q	
7	QT	CG30	A	Lib. → ZE
10	EJ	CG32	ZE	No ↓
11	TP	VA5	A	
12	ZJ	YL13	ZF	Ss. mode ↓ no → ZF
13	TP	SY10	Q	
14	QJ	YL22	YL15	Not IJKLM ↓ yes → YL22
15	RJ	WA	WA1	
16	TP	SY2	ER6	
17	TP	VA6	ER20	E21
20	TP	ER	UP3	
21	RJ	UP2	UP	
22	TP	SY2	TP2	
23	RA	VB3	CA3	Assign dummy CW
24	AT	CA35	TP3	
25	RJ	TP	TP1	CW → dummy list
26	MJ	0	YX	→ (B)
	CA	YL27		

Var.-Ftn. mode-not Ss. mode

	IA	ZF0		
0	TP	SY10	Q	} IJKLM ↓ no → ZG0
1	QJ	ZF2	ZG0	
2	RJ	WA	WA1	} E22
3	TP	ER22	UP3	
4	TP	SY2	ER27	
5	RJ	UP2	UP	} Return
6	MJ	0	YL22	
	CA	ZF7		

Not IJKLM

	IA	ZG		
0	RJ	PC	PC1	PC
1	RJ	SY	SY1	Symbol → A
2	EJ	CA30	ZH	(→ ZH no ↓
3	RA	VB3	CA3	} 62XXX CW → dummy list
4	AT	CA35	TP3	
5	TP	SZ2	TP2	
6	RJ	TP0	TP1	} Fltpt. → format
7	TP	VA40	Q	
10	QS	CG	TF3	
11	TP	SY2	A	Sym → A
12	MJ	0	YX2	→ (B)
	CA	ZG13		

Ftn. mode-Var. (

	IA	ZH0		
0	RA	VB3	CA3	} 75XXX CW → dummy list
1	AT	CH4	TP3	
2	TP	SZ2	TP2	
3	RJ	TP0	TP1	
4	RA	VB3	CA3	Leave space for Ss.
5	TP	PD1	PC2	Set PC
6	TP	VA40	Q	Ss. var → format
7	QS	CG3	TF3	} Set Ss. mode
10	TP	VA1	VA5	
11	TP	SZ2	VA6	
12	TP	CA27	VA36	# Ss. - 1 = 0
13	RA	VA1	CA3	Raise level
14	MJ	0	YX1	→ (B)
	CA	ZH15		

Lib. 5XXXX

	IA	ZE0		
0	RJ	WA	WA1	} E31
1	TP	SY2	GD5	
2	TP	VA10	GD17	
3	TP	GD	UP3	
4	RJ	UP2	UP	
5	TP	CA36	PC3	Set PC "doesn't matter"
6	MJ	0	YX	→ (B)
	CA	ZE7		

Left - Not 1st symbol
Not Ftn. mode
Var.

	IA	ZD		
0	TP	SY10	Q	IJKLM → ZI
1	QJ	ZI	ZD2	no ↓
2	RJ	WA	WA1	
3	TP	SY2	EK6	
4	TP	VA6	EK16	E11
5	TP	EK	UP3	
6	RJ	UP2	UP	
7	TP	VD	A	Before → ZI2
10	ZJ	ZD11	ZI2	After ↓
11	RJ	TS	TS1	Not in P.Op. list → ZD16
12	MJ	0	ZD16	In ↓
13	TP	TS3	EW2	CW → list
14	RJ	EW	EW1	
15	MJ	0	YX	→ (B)
16	RJ	TA	TA1	Not in CB list — ZD23
17	MJ	0	ZD23	In ↓
20	TP	TA4	EW2	CW → string
21	RJ	EW	EW1	
22	MJ	0	YX	→ (B)
23	RJ	TK	TK1	Increase CW
24	AT	CA31	EW2	Assign 65XXX CW
25	TP	A	TF2	
26	TP	CA26	TF	File → CB list
27	TP	SY2	TF1	
30	TP	CA27	TF3	
31	RJ	TE	TE1	
32	RJ	EW	EW1	CW → list
33	MJ	0	YX	→ (B)
	CA	ZD34		

1st Letter IJKLM

	IA	ZI		
0	TP	VD	A }	Before START ↓
1	ZJ	ZJ	ZI2 }	No → ZJ
2	RA	VB3	CA3 }	
3	AT	CA35	TP3 }	
4	TP	A	EW2 }	Dummy CW = 62XXX → list
5	TP	SY2	TP2 }	
6	RJ	TP0	TP1 }	
7	RJ	EW0	EW1 }	CW → list
10	MJ	0	YX }	→ (B)
	CA	ZI11		

After START

	IA	ZJ0		
0	RJ	TS0	TS1 }	
1	MJ	0	ZK }	In P.Op. list ↓ no → ZK
2	TP	TS3	EW2 }	CW → list
3	RJ	EW	EW1 }	
4	MJ	0	YX }	→ (B)
	CA	ZJ5		

Not in P.Op. list

	IA	ZK0		
0	RJ	TA0	TA1 }	Not in CB List → ZL0
1	MJ	0	ZL }	In ↓
2	TP	TA4	EW2 }	CW → list
3	RJ	EW	EW1 }	
4	MJ	0	YX }	→ (B)
	CA	ZK5		

Not in CB List

	IA	ZL		
0	RJ	TK	TK1 }	Assign 64XXX CW
1	AT	CA25	EW2 }	
2	TP	A	TF2 }	
3	TP	CA26	TF }	New file → CB list
4	TP	SY2	TF1 }	
5	TP	CA27	TF3 }	
6	RJ	TE	TE1 }	
7	RJ	EW	EW1 }	CW → list
10	MJ	0	YX }	→ (B)
	CA	ZL11		

Level Checking Sub.

	IA	DA		
0	MJ	0	(30000)	Exit
1	MJ	0	DA3	Start
2	0	0	40	32 ₁₀
3	TP	VA1	A	}
4	AT	VA2	A	
5	TJ	DA2	DA	level + (level → A
6	TP	VC5	Q	O.K. → exit no ↓
7	QJ	DA	DA10	Prev. print → exit
10	RJ	WA	WA1	No ↓
11	TP	GY12	UP3	}
12	RJ	UP2	UP	
13	TP	CA24	VC5	E57
14	MJ	0	DA	Set ind.
	CA	DA15		Exit

Inserted in region YX with jump
at YX11 = MJ 0 KX

	IA	KX		
0	EJ	CG11	YV	Δ . → (C2)
1	TP	SY11	Q	}
2	QJ	KX3	YX12	
3	TP	SY13	Q	}
4	QJ	YX12	KX5	
5	TP	VD	A	}
6	ZJ	KX7	KX20	
7	RJ	RD	RD1	}
10	TP	SY2	RS4	
11	RJ	RS2	RS	}
12	TP	RS3	A	
13	RJ	GW	GW1	}
14	TP	Q	EW2	
15	RJ	EW	EW1	}
16	TP	CA6	PC3	
17	MJ	0	YX	Set PC
20	RJ	WA	WA1	→ (B)
21	TP	GZ	UP3	E58
22	RJ	UP2	UP	}
23	TP	CA36	PC3	
24	MJ	0	YX	Set PC with doesn't matter.
	CA	KX25		→ (B)

Check Vary List

	IA	KY		
0	MJ	0	(30000)	
1	TP	VL	A	} Set up N of RP
2	ST	CG26	A	
3	TP	CG27	Q	
4	QS	A	KY6	
5	TP	SY2	A	Sym → A
6	RP	20000	KY	Not in Vary List → exit
7	EJ	VL1	KY10	In ↓
10	RJ	WA	WA1	} E28
11	TP	SY2	GB26	
12	TP	SY2	GB40	
13	TP	GB22	UP3	
14	RJ	UP2	UP	
15	MJ	0	KY	Exit
	CA	KY16		

Equation Translation
Right

Section D - Switch = | / * - + , ; constant
 Section E - Superscript symbols
 Section G - POW) (
 Section H - VAR Before START
 Section I - VAR After START, In pseudo Op. list
 Section F - VAR After START, Not in pseudo Op. list
 Section F - also Δ .

Region	Section	Region	Section	Region	Section
ZM	D			HL	H
MB	D	ZR	G	XQ	H
MD	D	NY	G	HS	H
ME	D	ZQ	G	HR	H
MA	D	FN	G	HU	H
ZZ	D	FM	G	HQ	H
ZY	D	FL	G		
MY	D	FP	G		
ZX	D	ZN	G	XD	I
MZ	D			NA	I
ZW	D			HZ	I
ZV	D	ZT	H	HY	I
XC	D	XG	H	HX	I
		XF	H	HW	I
		XH	H	ND	I
MC	E	XI	H	NE	I
MH	E	XE	H	NC	I
MJ	E	XW	H	NB	I
ML	E	XV	H		
MN	E	XT	H	HV	F
MM	E	XR	H	NL	F
MQ	E	HG	H	NJ	F
MG	E	HH	H	NI	F
MP	E	HF	H	NH	F
MK	E	HC	H	NS	F
MU	E	XZ	H	NP	F
MT	E	HK	H	NK	F
MS	E	XY	H	NG	F
MV	E	HN	H	NX	F
MX	E	HM	H		
MW	E			NW	F
				NV	F
				NN	F
				FU	F
				NF	F
				FK	F
				FJ	F
				NZ	F
				YV	F

Sections of connectors;

Connector	Section
(C)	D
(C2)	F
(D1)	D
(D2)	E
(E)	H
(F)	I

Equation Translation Right

C		IA	ZM		
	0	RJ	PC	PC1	Pair check
	1	RJ	SY	SY1	Symbol $\rightarrow$ A
	2	EJ	CA30	ZN	($\rightarrow$ ZN
	3	EJ	CG6	ZQ	) $\rightarrow$ ZQ
	4	EJ	CH5	ZR	POW $\rightarrow$ ZR
	5	TP	SY7	Q	Var. $\rightarrow$ ZT no $\downarrow$
	6	QJ	ZT	ZM7	} Superscript $\rightarrow$ D2 no $\downarrow$
	7	TP	SY13	Q	
	10	QJ	MC	ZM11	
	11	TP	SY11	Q	} Constant $\rightarrow$ ZV
	12	QJ	ZV	ZM13	
	13	EJ	CG4	ZW	
	14	EJ	CG5	ZW	} , or ; $\rightarrow$ ZW
	15	EJ	CH6	ZX	
	16	EJ	CH7	ZY	
	17	EJ	CH10	ZZ	+ $\rightarrow$ ZX
	20	EJ	CH11	MA	- $\rightarrow$ ZY
	21	EJ	CG14	MD	* $\rightarrow$ ZZ
	22	EJ	CG11	YV	/ $\rightarrow$ MA
	23	EJ	CG7	MB	$\rightarrow$ MD
	24	RJ	WA	WA1	Δ . $\rightarrow$ C2
	25	TP	SY2	GJ16	= $\rightarrow$ MB
	26	TP	GJ12	UP3	} E42
	27	RJ	UP2	UP	
	30	TP	CA36	PC3	
	31	MJ	0	ZM	Set PC
		CA	ZM32		C

Right =

	IA	MB		
0	RA	VA14	CA3	# = 's + 1
1	TP	CA21	PC3	Set PC
2	TP	CG15	EW2	CW $\rightarrow$ list
3	RT	EW0	EW1	$\rightarrow$ C
4	MJ	0	ZM	
	CA	MB5		

			Right	Absolute Value	
					open
	IA	MD			
0	TP	SZ2	A		
1	EJ	CG7	MD17	=	
2	RP	20005	MD6	+ POW	
3	EJ	CH5	MD17	- * / → MD17	
4	0	0	0		
5	0	0	0		
6	EJ	CA30	MD17	(	
7	EJ	CG4	MD17	,	
10	EJ	CG5	MD17	;	
11	TP	EW2	Q		
12	QT	CG30	A		
13	EJ	CG32	MD17	Lib. → open	
14	TP	Q	A		
15	EJ	CH14	MD17	° → open	
16	MJ	0	ME	Closed → ME	
17	TP	CA23	PC3	Set PC with open	
20	TP	CH14	EW2		
21	RJ	EW	EW1	CW → string	
22	TP	CA27	VA3	Clear (level raisers	
23	RA	VA4	CA3	level raisers + 1	
24	RA	VA2	CA3	Raise level	
25	MJ	0	ZM	→ (C)	
	CA	MD26			

Absolute Value
| (Closed)

	IA	ME		
0	TP	CA5	PC3	Set PC
1	RS	VA2	CA3	Lower level
2	TJ	CA3	ME4	Level too low → ME4
3	MJ	0	ME10	OK → ME10
4	TP	CA3	VA2	Set level =
5	RJ	WA	WA1	E53
6	TP	GV	UP3	
7	RJ	UP2	UP	
10	TP	VA3	A	(Raisers = 0 → ME15 No ↓
11	ZJ	ME12	ME15	
12	RJ	WA	WA1	E38
13	TP	GH24	UP3	
14	RJ	UP2	UP	
15	TP	VA4	A	1 > Level raisers → ME20 No ↓
16	TJ	CA3	ME20	
17	RS	VA4	CA3	Raisers -
20	TP	VA5	A	Ss. mode ↓ no → ME25 Fixed CW → list
21	ZJ	ME22	ME25	
22	TP	CH13	EW2	→ (C)
23	RJ	EW	EW1	
24	MJ	0	ZM	Fixed Eq. → ME22 No ↓
25	TP	VA16	Q	
26	QJ	ME22	ME27	Floating CW → list
27	TP	CH12	EW2	
30	MJ	0	ME23	
	CA	ME31		

Right
/

	IA	MA0		
0	TU	CH0	MA12 }	Set for /
1	TU	CH1	MA5 }	
2	TP	CA13	PC3	Set P.C.
3	TP	VA5	A }	
4	ZJ	MA5	MA10 }	Ss. mode ↓ no → MA10
5	TP	(30000)	EW2 }	Fixed CW → list
6	RJ	EW0	EW1 }	
7	MJ	0	ZM	→ (C)
10	TP	VA16	Q }	Fixed eq. → MA5
11	QJ	MA5	MA12 }	No ↓
12	TP	(30000)	EW2 }	Fl. C.W. → list
13	RJ	EW0	EW1 }	
14	MJ	0	ZM	→ (C)
	CA	MA15		

Right
*

	IA	ZZ0		
0	TU	CH2	MA12	Fltpt. *
1	TU	CH3	MA5	Fixed Pt. *
2	MJ	0	MA2	→ / sequence
	CA	ZZ3		

Right
- (minus)

	IA	ZY		
0	TP	CA12	PC3	Set P.C.
1	TP	EW2	Q	Last CW → Q
2	QT	CG30	A	
3	EJ	CG30	MY	} Var. Const. → MY0 Spec. CW
4	EJ	CI14	MY	
5	EJ	CI15	MY	
6	TP	Q	A	
7	EJ	CH12	MY	} (closed) → MY0
10	EJ	CH13	MY	
11	EJ	CG34	MY	
Unary - 12	TP	VA5	A	) → MY0
13	ZJ	ZY16	ZY14	Ss. mode → ZY16
14	TP	VA16	Q	No ↓
15	QJ	ZY16	ZY21	Fixed eq. ↓ no → ZY21 Fixed unary - → string
16	TP	CI16	EW2	
17	RJ	EW0	EW1	
20	MJ	0	ZM	→ (C)
21	TP	CI17	EW2	Floating unary - → string
22	RJ	EW0	EW1	
23	MJ	0	ZM	→ (C)
	CA	ZY24		

Binary - (minus)
or
+ (plus)

	IA	MY0		
0	TU	CI35	MY6	} Set for -
1	TU	CI36	MY11	
2	TP	VA5	A	} Ss. mode → MY6 No ↓
3	ZJ	MY6	MY4	
4	TP	VA16	Q	} Fixed eq. ↓ no → MY11 Fixed $\bar{+}$ → string
5	QJ	MY6	MY11	
6	TP	(30000)	EW2	} Floating $\bar{+}$ → string
7	RJ	EW0	EW1	
10	MJ	0	ZM	→ (C)
11	TP	(30000)	EW2	} Floating $\bar{+}$ → string
12	RJ	EW0	EW1	
13	MJ	0	ZM	→ (C)
	CA	MY14		

Right
+

	IA	ZX0		
0	TP	CA12	PC3	Set PC
1	TP	EW2	Q	Last CW → Q
2	QT	CG30	A	
3	EJ	CG30	MZ0	Var., Const. or Spec. CW → MZ0
4	EJ	CI14	MZ0	
5	EJ	CI15	MZ0	
6	TP	Q	A	
7	EJ	CH12	MZ0	(closed) → MZ0
10	EJ	CH13	MZ0	
11	EJ	CG34	MZ0	) → MZ0
Unary +12	MJ	0	ZM	→ (C)
	CA	ZX13		

Binary +

	IA	MZ0		
0	TU	CI37	MY6	Set for +
1	TU	CK0	MY11	
2	MJ	0	MY2	
	CA	MZ3		

Right , or ;

	IA	ZW		
0	TP	CA11	PC3	Set PC
1	TP	VA5	A	Ss. mode ↓ no → ZW7
2	ZJ	ZW3	ZW7	
3	RS	VA36	CA3	
4	TP	CG36	EW2	# commas - 1
5	RJ	EW	EW1	CW → list
6	MJ	0	ZM	→ (C)
7	TP	VC10	A	
10	EJ	CK1	ZW25	Lib. mode ↓ no → ZW25
11	TU	VC10	ZW12	
12	TP	(30000)	A	Right level → ZW22
13	AT	CA3	A	
14	EJ	VA1	ZW22	No ↓
15	RJ	WA	WA1	
16	TU	VC30	ZW17	E56
17	TP	(30000)	GY10	
20	TP	GY	UP3	
21	RJ	UP2	UP	# commas - 1
22	TU	VC20	ZW23	
23	RS	(30000)	CA3	
24	MJ	0	ZW4	E43
25	RJ	WA	WA1	
26	TP	GJ24	UP3	
27	RJ	UP2	UP	
30	MJ	0	ZW4	
	CA	ZW31		

1st Superscript Symbol

D2		IA	MC		
	0	RP	10010	MC2	} Clear variables
	1	TP	CA27	VC	
	2	TP	CH31	VA23	} Set POW sequence
	3	TP	CH22	VA24	
	4	TP	CA3	VA22	Set count = 1
	5	TP	CH15	VA15	
	6	TP	VA5	A	} Set seq.
	7	ZJ	MC10	MC17	
	10	RJ	WA	WA1	} Ss. mode ↓ no → MC17
	11	TP	VA6	GD37	
	12	TP	SY2	GD26	} E32
	13	TP	SY3	GD27	
	14	TP	GD21	UP3	
	15	RJ	UP2	UP	
	16	MJ	0	MK	} Fixed eq. ↓
	17	TP	VA16	Q	
	20	QJ	MC21	MK	
	21	RJ	WA	WA1	
	22	TP	SY2	GE5	} E33
	23	TP	SY3	GE6	
	24	TP	GE	UP3	
	25	RJ	UP2	UP	
	26	MJ	0	MK	
		CA	MC27		

Sequence Loop

		IA	MH		
0	RJ	PC	PC1	P.C.	
1	RJ	SY	SY1	Sym. → A	
2	TP	SY13	Q		
3	QJ	MH4	MJ	Superscript ↓ no → MJ	
4	RA	VA22	CA3	Count symbols	
5	TJ	CH16	MK	5 > # sym. → MK no ↓	
6	RJ	WA	WA1	} E34	
7	TP	GE16	UP3		
10	RJ	UP2	UP		
11	RJ	SY	SY1		
12	TP	SY13	Q	Sym. → A	} ↷
13	QJ	MH11	MH14	Superscript	
14	TP	CA36	A	No ↓	
15	AT	PD1	PC2	Set PC with "doesn't matter"	
16	TP	CG34	EW2		
17	TP	SY2	A	CW =)	
20	MJ	0	ZM2	Sym. → A	
	CA	MH21		→ (C)	

Sym $\equiv$ 4

	IA	MJ		Not superscript	↓	
0	TP	CH17	A	}) → Sequence		
1	AT	CH21	A			
2	AT	VA22	A			
3	AT	CA3	MJ4			
4	0	30000	30000	} TP CG34 VA+		
5	TP	VC	Q		Dec. pt. in seq. → ML	no → MM
6	QJ	ML	MM			
	CA	MJ7				

Dec. pt. in seq., i.e., not spec. case ↓

	IA	ML			
0	SP	VA22	17	} # sym in seq. → A → VA15	
1	SA	VA22	0		
2	AT	CA26	VA15		
3	AT	EW3	A	Add. → A	
4	TJ	CJ10	MN	Fit in list → MN	no ↓
5	TP	EW4	Q	} Prev. print → ML12 no ↓	
6	QJ	ML12	ML7		
7	RJ	WA	WA1		
10	TP	FI	UP3		F16
11	RJ	UP2	UP		
12	TP	CJ10	EW3	Set limit	
13	TP	CA24	EW4	Set ind.	
14	MJ	0	MH16		
	CA	ML15			

Seq. Fits in List

	IA	MN			
0	TV	EW3	MN5	} Fix address	
1	RA	MN5	CA3		
2	TP	CG27	Q	} Set n of repeat	
3	QS	VA15	MN4		
4	RP	30000	MN6	} Seq. → list	
5	TP	VA23	(30000)		Set address.
6	RA	EW3	VA15		
7	MJ	0	MH16		
	CA	MN10			

Possible Special Case

	IA	MMO		
0	TP	VA32	A	- → MM4
1	EJ	CI0	MM4	
2	TP	CA27	VC3	Clear VC3
3	MJ	0	MM5	
4	TP	CH37	VC3	Set VC3 for -
5	TP	VA22	A	# symbols = 3 → MG0
6	EJ	CH24	MG	No ↓
7	EJ	CA3	MP	= 1 → MP0
10	EJ	CH25	MQ	= 2 → MQ0 4 ↓
11	TU	CH26	MM14	Set loop addresses
12	TU	CH27	MM15	
13	TP	CH24	VC1	Set index
14	TP	(30000)	A	- 1/2 → A
15	EJ	(30000)	MM17	= → MM17 ≠ → ML0
16	MJ	0	ML0	
17	RA	MM14	CH22	Modify
20	RA	MM15	CH22	
21	LJ	VC1	MM14	Return
22	TP	CH30	EW2	16100 → CW of - 1/2 → list
23	RJ	EW0	EW1	
24	MJ	0	MH17	→ (C)
	CA	MM25		

Two Symbols in Sequence

	IA	MQ		
0	TP	VA32	A	- → MQ3
1	EJ	CI	MQ3	
2	MJ	0	ML	No → ML
3	TP	VA33	A	
4	EJ	CI1	MX	Next sym. = 1 → MX
5	EJ	CI3	MS	= 2 → MS
6	EJ	CI4	MV	= 3 → MV no
7	TP	SZ11	Q	
10	QJ	MQ11	ML	Constant ↓ No → ML
11	TP	MI2	RS4	
12	RJ	RS2	RS	Convert to octal
13	TP	CH32	A	# > 77 → MW
14	TJ	RS3	MW	no ↓
15	SP	RS3	0	130xx
16	SA	VC3	0	131xx } → list
17	AT	CI5	EW2	
20	RJ	EW	EW1	
21	MJ	0	MH16	
	CA	MQ22		

symbols in sequence = 3

	IA	MG		
0	TU	CH34	MG3	} Set loop addresses
1	TU	CH27	MG4	
2	TP	CH25	VC1	Set index = 2
3	TP	(30000)	A	1/2 → A
4	EJ	(30000)	MG6	= Special case → MG6
5	MJ	0	ML	≠ → ML
6	RA	MG3	CH22	} Modify addresses
7	RA	MG4	CH22	
10	IJ	VC1	MG3	
11	TP	CH33	EW2	} CW → list
12	RJ	EW	EW1	
13	MJ	0	MH17	→ (C)
	CA	MG14		

symbols = 1

	IA	MP		
0	TP	VA32	A	
1	EJ	CI1	MH16	= 1 → (C)
2	EJ	CI3	MS	= 2 → MS
3	EJ	CI4	MV	= 3 → MV
4	MJ	0	MQ7	→ MQ7
	CA	MP5		

Superscript Symbol

	IA	MK0			
0	TP	SY11	Q	}	Constant → MT0
1	QJ	MT0	MK2	}	no ↓
2	TP	SY2	A	}	- → MU
3	EJ	CG12	MU	}	No (assume /) ↓
4	TP	CA16	PC3		Set PC
5	TP	VA22	A	}	
6	AT	CH17	A	}	/ → sequence
7	TV	A	MK10	}	
10	TP	CH35	(30000)	}	
11	TP	VA22	A	}	
12	AT	CH20	A	}	/ → special sequence
13	TV	A	MK14	}	
14	TP	SY2	(30000)	}	
15	MJ	0	MH		→ get next symbol of sequence
	CA	MK16			

Superscript -

	IA	MU0			
0	TP	CA15	PC3		Set PC
1	TP	VA22	A	}	
2	AT	CH17	A	}	- → sequence
3	TV	A	MU4	}	
4	TP	CH36	(30000)	}	
5	TP	VA22	A	}	
6	AT	CH20	A	}	- → special sequence
7	TV	A	MU10	}	
10	TP	SY2	(30000)	}	
11	MJ	0	MH		→ return
	CA	MU12			

Superscript Up Constant

0	IA	MT		
1	TP	CA14	PC3	Set PC
2	RJ	RB	RB1	Check constant
3	TP	SY2	MI4	Lower constant
4	TP	SY3	MI5	
5	RJ	MI	MI1	
6	TP	MI2	GG4	Convert to floating point
7	TP	MI3	GG5	
10	RJ	GG2	GG	
11	TP	GG3	A	Assign CW
12	RJ	GW	GW1	Store CW
13	TP	Q	VC2	
14	TP	VA22	A	
15	AT	CH17	A	CW → sequence
16	TV	A	MT16	
17	TP	VC2	(30000)	
20	TP	VA22	A	Fltpt. const. → Special sequence
21	AT	CH20	A	
22	TV	A	MT22	
23	TP	GG3	(30000)	Pt. in sequence ↓ no → MH
24	TP	SY6	A	
25	ZJ	MT25	MH	
26	TP	CA24	VC	Set ind.
	MJ	0	MH	→return
	CA	MT27		

Constant = 2

0	IA	MS		
1	TP	VC3	A	
2	AT	CI6	EW2	CW → list
3	RJ	EW	EW1	
	MJ	0	MH17	
	CA	MS4		

Constant = 3

	IA	MV		
0	TP	VC3	A	
1	AT	CI7	EW2	CW → list
2	RJ	EW	EW1	
3	MJ	0	MH17	→ (C)
	CA	MV4		

Constant = 1

	IA	MX		
0	TP	CI10	EW2 }	CW → list
1	RJ	EW	EW1 }	
2	MJ	0	MH17	→ (C)
	CA	MX3		

| Const. | > 77

	IA	MW		
0	RA	VA23	CA3	Change POW to 101
1	MJ	0	ML	
	CA	MW2		

Right POW

	IA	ZR		
0	TP	CH37	EW2 }	CW → string
1	RJ	EW0	EW1 }	
2	TP	CA17	PC3	Set PC
3	TP	VA5	A }	
4	ZJ	ZR5	NY }	Ss. mode ↓ no → NY
5	RJ	WA	WA1 }	
6	TP	VA6	GF10 }	
7	TP	GF	UP3 }	E35
10	RJ	UP2	UP }	
11	MJ	0	ZM	→ (C)
	CA	ZR12		

Not Ss. mode

	IA	NY		
0	TP	VA16	Q }	Floating pt. eq. → ZM
1	QJ	NY2	ZM }	No ↓
2	RJ	WA	WA1 }	
3	TP	GF12	UP3 }	E36
4	RJ	UP2	UP }	
5	MJ	0	ZM	→ (C)
	CA	NY6		

Right)

	IA	ZQ		
0	TP	CA5	PC3	Set PC
1	TP	CG34	EW2 }	CW → string
2	RJ	EW0	EW1 }	
3	TP	VA4	A }	Last raiser → FP
4	ZJ	FP	ZQ5 }	no ↓
5	TP	VA3	A	(raisers = 0 → ZQ10
6	ZJ	ZQ7	ZQ10	No ↓
7	RS	VA3	CA3	(raisers - 1
10	RS	VA1	CA3	Lower level
11	TJ	CA3	FLO	Too low → FLO
12	EJ	VA5	FM	Leaving Ss. mode → FMO
13	TU	VC10	ZQ14 }	Leaving Lib. → FNO
14	EJ	(30000)	FN }	
15	MJ	0	ZM	→ (C)
	CA	ZQ16		

Leaving Lib.

	IA	FN		
0	TU	VC20	FN1	} # commas correct → FN10
1	TP	(30000)	A	
2	ZJ	FN3	FN10	} No ↓
3	RJ	WA	WA1	
4	TU	VC30	FN5	} E37
5	TP	(30000)	GH13	
6	TP	GH	UP3	
7	RJ	UP2	UP	} Take Lib. off list
10	RS	VC10	CH23	
11	RS	VC20	CH23	
12	RS	VC30	CH23	} → (C)
13	MJ	0	ZM	
	CA	FN14		

Leaving Ss. mode

	IA	FM		
0	TP	CA27	VA5	} Clear Ss. mode
1	TP	VA36	A	
2	ZJ	FM3	ZM	} # Ss. correct → (C)
3	RJ	WA	WA1	
4	TP	VA6	GI17	} No ↓
5	TP	GI12	UP3	
6	RJ	UP2	UP	
7	MJ	0	ZM	} E40
	CA	FM10		
				} → (C)

Level too Low

	IA	FL		
0	TP	CA3	VA1	} Set level = 0
1	TP	VA	Q	
2	QJ	ZM	FL3	} Previous point → ZM
3	RJ	WA	WA1	
4	TP	GI	UP3	} No ↓
5	RJ	UP2	UP	
6	TP	CA24	VA	} E39
7	MJ	0	ZM	
	CA	FL10		} Set point
				} → (C)

Interlocking (and |

	IA	FP		
0	RJ	WA	WA1	} E38 →) Section
1	TP	GH24	UP3	
2	RJ	UP2	UP	
3	MJ	0	ZQ10	
	CA	FP4		

Right (

	IA	ZN0		
0	TP	CA4	PC3	Set PC
1	TP	CG24	EW2	CW → string
2	RJ	EW	EW1	} Not Ss. mode → ZN11 Ss. mode ↓
3	TP	VA5	A	
4	ZJ	ZN5	ZN11	
5	RJ	WA	WA1	} E41
6	TP	VA6	GJ10	
7	TP	GJ	UP3	
10	RJ	UP2	UP	} Clear level raisers Raise level → (C)
11	TP	CA27	VA4	
12	RA	VA3	CA3	
13	RA	VA1	CA3	
14	MJ	0	ZM	
	CA	ZN15		

Right-Constant

D1		IA	ZV0			
	0	TP	CA6	PC3	Set PC	
	1	TP	VA5	A	Ss. mode → ZV5	
	2	ZJ	ZV5	ZV3	No ↓	
	3	TP	VA16	Q		
	4	QJ	ZV5	XC0	Fixed eq. ↓ no → XC0	
	5	RJ	RD	RD1	Check const.	
	6	TP	SY2	RS4	Convert to fixed point	
	7	RJ	RS2	RS0		
	10	TP	RS3	A	Assign CW	
	11	RJ	GW0	GW1		
	12	TP	Q	EW2	CW → string-out	
	13	RJ	EW0	EW1		
	14	MJ	0	ZM	New CW → list } crr. #1	
		CA	ZV15		→ (C)	

Floating Point Constant

		IA	XC0			
0	RJ	RB0	RB1	Check const.		
1	TP	SY2	GG4			
2	TP	SY3	GG5	Convert to fltpt.		
3	RJ	GG2	GG0			
4	TP	GG3	A	Const. → A		
5	MJ	0	ZV11			
	CA	XC6				

Right - Var. - Before

(E)	0	IA	ZT		
	1	TP	VD	A	} Before → ZT2
	2	ZJ	XD	ZT2	
	3	RJ	TU	TU1	} After → XD = (F)
	4	MJ	0	XE	
	5	TP	TU3	EW2	} Not in Ftn. list → XE
	6	RJ	EW	EW1	
	7	TP	VA5	A	} In ↓
	10	ZJ	ZT10	XF	
	11	TP	TU3	Q	} CW → string
	12	QT	CG17	A	
	13	EJ	CI25	XG	} Ss. mode ↓ no → XF
	14	TP	SY10	Q	
	15	QJ	ZT22	ZT15	} CW → A
	16	RJ	WA	WA1	
	17	TP	SY2	ER6	} 75xxx → XG no ↓
	20	TP	VA6	ER20	
	21	RJ	UP2	UP	} Assume 62xxx
	22	TP	CA6	PC3	
	23	MJ	0	ZM	} IJKLM → ZT22 no ↓
		CA	ZT24		
					} E21
					} Set PC
					} → (C)

Right - In Ftn. list Ss. mode 75xxx

	0	IA	XG		
	1	TP	CA7	PC3	} → PC
	2	RJ	PC	PC1	
	3	RJ	WA	WA1	} E20
	4	TP	SY2	EP30	
	5	TP	VA6	EP41	} UP3
	6	TP	EP22	UP	
	7	RJ	UP2	UP	} Sym → A
	10	RJ	SY	SY1	
	11	EJ	CA30	XG12	} (→ XG12 no ↓
	12	MJ	0	ZM3	
	13	TP	GP17	UP3	} → (C)
	14	RJ	UP2	UP	
	15	RJ	FV	FV1	} "Ss. not checked"
	16	MJ	0	YV	
		MJ	0	ZM1	} delete Ss.
		CA	XG17		
					} Δ . → (C2)
					} → (C)

Not Ss. mode-in Ftn. list

	IA	XF		
0	TP	TU3	Q	xx000 → A
1	QT	CG17	A	
2	EJ	CI25	XH	75xxx → XH no ↓
3	TP	SY10	Q	
4	QJ	XF5	XF11	IJKLM ↓ no → XF11
5	RJ	WA	WA1	
6	TP	SY2	ER27	E22
7	TP	ER22	UP3	
10	RJ	UP2	UP	
11	TP	CA6	PC3	Set PC
12	MJ	0	ZM	→ (C)
	CA	XF13		

75xxx

	IA	XH		
0	TP	CA7	PC3	Do PC
1	RJ	PC	PC1	
2	RJ	SY	SY1	Sym → A
3	EJ	CA30	XI	(→ XI no ↓
4	RJ	WA	WA1	
5	TP	SZ2	EX6	E23
6	TP	EX	UP3	
7	RJ	UP2	UP	
10	TP	SY2	A	Sym → A
11	MJ	0	ZM3	→ (C)
	CA	XH12		

75xxx (

	IA	XI		
0	TP	CA4	PC3	Set PC
1	TP	VA1	VA5	Set Ss. mode
2	TP	SZ2	VA6	
3	TP	CA27	VA36	Clear # commas
4	TP	CG24	EW2	CW → list
5	RJ	EW	EW1	
6	MJ	0	ZN11	→ (→ (C)
	CA	XI7		

Right-Before-Var.
Not in Ftn. list

	IA	XE			
0	RJ	TA	TA1	}	
1	MJ	O	XQ	}	Not in CB List → XQ in ↓
2	TP	TA4	EW2	}	CW → list
3	RJ	EW	EW1	}	
4	TP	CA6	PC3		Set PC
5	TP	VA16	Q	}	Not fixed eq. → XR
6	QJ	XE7	XR	}	Fixed ↓
7	TP	TA4	Q		
10	QT	CG17	A		
11	EJ	CG17	XT		77xxx → XT
12	EJ	CA33	XV		66xxx → XV
13	EJ	CA31	XW		65xxx → XW
14	EJ	CA25	ZM		64xxx → (C)
15	TP	CA20	PC3	}	5xxxx ↓
16	RJ	PC	PC1	}	Do PC
17	RJ	WA	WA1	}	E13
20	TP	SY2	EL4	}	
21	TP	EL	UP3	}	
22	RJ	UP2	UP	}	
23	RJ	SY	SY1		Sym → A
24	EJ	CA30	XE26	(	→ XE26
25	MJ	O	ZM3		No → (C)
26	TP	GP	UP3	}	"Args not checked"
27	RJ	UP2	UP	}	
30	RJ	FV	FV1		Delete LIB
31	MJ	O	YV		Δ . → (C2)
32	MJ	O	ZM1		→ (C)
	CA	XE33			

65xxx

	IA	XW		
0	RJ	WA	WA1	}
1	TP	SY2	EL17	}
2	TP	EL12	UP3	}
3	RJ	UP2	UP	}
4	MJ	O	ZM	
	CA	XW5		

E14

→ (C)

66xxx

	IA	XV		
0	TP	CA10	PC3	} Do PC
1	RJ	PC	PC1	
2	RJ	WA	WA1	
3	TP	SY2	EM3	} E15
4	TP	EM	UP3	
5	RJ	UP2	UP	
6	RJ	SY	SY1	Sym $\rightarrow$ A
7	EJ	CA30	XV11	($\rightarrow$ XV11 no $\downarrow$
10	MJ	0	ZM3	$\rightarrow$ (C)
11	TP	GP	UP3	"Args not checked"
12	RJ	UP2	UP	} Delete Ftn.
13	RJ	FV	FV1	
14	MJ	0	YV	
15	MJ	0	ZM1	Δ . $\rightarrow$ (C2)
	CA	XV16		$\rightarrow$ (C)

77xxx

	IA	XT		
0	TP	CA7	PC3	} Do PC
1	RJ	PC	PC1	
2	RJ	WA	WA1	
3	TP	SY2	EM17	} E16
4	TP	EM11	UP3	
5	RJ	UP2	UP	
6	RJ	SY	SY1	Sym $\rightarrow$ A
7	EJ	CA30	XT11	($\rightarrow$ XT11
10	MJ	0	ZM3	No $\rightarrow$ (C)
11	TP	GP17	UP3	} "Ss. not checked"
12	RJ	UP2	UP	
13	RJ	FV	FV1	
14	MJ	0	YV	Delete Ss.
15	MJ	0	ZM1	Δ . $\rightarrow$ (C2)
	CA	XT16		$\rightarrow$ C

Var.
Before - In CB List
Floating Point Equation

	IA	XR			
0	TP	VA5	A	}	Ss. mode —XY
1	ZJ	XY	XR2		No
2	TP	TA4	Q	}	
3	QT	CG17	A		xx000 → A
4	EJ	CG17	XZ		77xxx → XZ
5	EJ	CA33	HC		66xxx → HC
6	EJ	CA31	ZM		65xxx → (C)
7	EJ	CA25	HF		64xxx → HF
10	TP	CA20	PC3	}	5xxxx ↓ pair check
11	RJ	PC	PC1		
12	TP	TA4	Q	}	# operands = 1 → (C)
13	QT	CI31	VA13		
14	TJ	CH25	ZM1		No ↓
15	RJ	SY	SY1		Sym → A
16	EJ	CA30	HG		(→ HG no ↓
17	TP	SZ2	EN5	}	
20	RJ	WA	WA1		
21	TP	EN	UP3		E17
22	RJ	UP2	UP		
23	TP	SY2	A		Sym → A
24	MJ	0	ZM2		→ (C)
	CA	XR25			

Lib. (

	IA	HG			
0	TP	VA1	VA12	}	Set Lib.
1	RA	VC20	CH23		Room for Lib. → HH
2	TJ	CI33	HH		No ↓
3	RS	VC20	CH23		Add. - 1
4	RJ	WA	WA1	}	
5	TP	SZ2	GM25		
6	TP	GM11	UP3		E49
7	RJ	UP2	UP		
10	RJ	FV	FV1		Delete Lib. args
11	MJ	0	YV		Δ . → C2
12	MJ	0	ZM1		→ (C)
	CA	HG13			

Room for Lib.

	IA	HH		
0	TV	A	HH2	} # commas → list
1	TP	VA13	A	
2	ST	CA3	(30000)	
3	RA	VC10	CH23	} Lib. level → list
4	TV	A	HH5	
5	TP	VA12	(30000)	
6	RA	VC30	CH23	} XS3 of Lib. → list
7	TV	A	HH10	
10	TP	SZ2	(30000)	
11	MJ	0	ZN	→ (section
	CA	HH12		

64xxx

	IA	HF		
0	RJ	WA	WA1	} E12
1	TP	SY2	EK25	
2	TP	EK20	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZM	→ (C)
	CA	HF5		

Floating Pt. Eq. Right

66xxx ↓

	IA	HC		
0	TP	CA10	PC3	} Pair check
1	RJ	PC	PC1	
2	RJ	SY	SY1	Sym → A
3	EJ	CA30	WD	(→ WD no ↓
4	MJ	0	ZM3	→ (C)
	CA	HC5		

77xxx ↓

	IA	XZ		
0	TP	CA7	PC3 }	Pair check
1	RJ	PC	PC1 }	
2	RJ	SY	SY1	Sym → A
3	EJ	CA30	HK	(→ HK no ↓
4	RJ	WA	WA1 }	
5	TP	SZ2	EN30 }	E18
6	TP	EN22	UP3 }	
7	RJ	UP2	UP }	
10	TP	SY2	A	Sym → A
11	MJ	0	ZM3	→ (C)
	CA	XZ12		

77xxx (

	IA	HK		
0	TP	VA1	VA5 }	Set Ss. mode
1	TP	TA5	Q }	
2	QT	CA2	A }	Ss. - 1 → VA36
3	ST	CA3	VA36 }	
4	TP	SZ2	VA6	Store XS3 of variable
5	TP	CA4	PC3	Set PC
6	TP	CG24	EW2 }	(→ list
7	RJ	EW	EW1 }	
10	MJ	0	ZN11	→ (→ (C)
	CA	HK11		

Ss. mode - Right - In CB List

	IA	XY		
0	TP	TA4	Q	}
1	QT	CG17	A	
2	EJ	CG17	HL	xx000 → A
3	EJ	CA33	HM	77xxx → HL
4	EJ	CA31	HN	66xxx → HM
5	EJ	CA25	ZM	65xxx → HN
6	TP	CA20	PC3	64xxx → (C)
7	RJ	PC	PC1	5xxxx ↓
10	RJ	WA	WA1	Do pair check
11	TP	SY2	EJ4	}
12	TP	VA6	EJ13	
13	TP	EJ	UP3	
14	RJ	UP2	UP	
15	RJ	SY	SY1	Sym → A
16	EJ	CA30	XY20	(→ XY20 no ↓
17	MJ	0	ZM3	→ (C)
20	TP	GP25	UP3	}
21	RJ	UP2	UP	
22	RJ	FV	FV1	"Args not checked"
23	MJ	0	YV	Delete Lib.
24	MJ	0	ZM1	→ (C2)
	CA	XY25		→ (C)

65xxx

	IA	HN	
0	TP	CA6	PC3
1	RJ	WA	WA1
2	TP	SY2	ER6
3	TP	VA6	ER20
4	TP	ER	UP3
5	RJ	UP2	UP
6	MJ	0	ZM
	CA	HN7	

Set PC

E21

→ (C)

66xxx

	IA	HM		
0	TP	CA10	PC3	} Do PC
1	RJ	PC	PC1	
2	RJ	WA	WA1	} E10
3	TP	SY2	EJ20	
4	TP	VA6	EJ25	
5	TP	EJ15	UP3	
6	RJ	UP2	UP	
7	RJ	SY	SY1	Sym → A
10	EJ	CA30	WD	(→WD no ↓
11	MJ	0	ZM3	→ (C)
	CA	HM12		

77xxx

	IA	HL		
0	TP	CA7	PC3	} Do PC
1	RJ	PC	PC1	
2	RJ	WA	WA1	} E8
3	TP	SY2	EI26	
4	TP	VA6	EI35	
5	TP	EI20	UP3	
6	RJ	UP2	UP	
7	RJ	SY	SY1	Sym → A
10	EJ	CA30	HL12	(→HL12 no ↓
11	MJ	0	ZM3	→ (C)
12	TP	GP17	UP3	} "Ss. not checked"
13	RJ	UP2	UP	
14	RJ	FV	FV1	Delete Ss.
15	MJ	0	YV	Δ. → (C2)
16	MJ	0	ZM1	→ (C)
	CA	HL17		

Variable Right - Before Not in Ftn. or CB Lists

	IA	XQ			
0	RJ	RH	RH1	}	Check symbol Set to add to CB List
1	TP	CA27	TF3		
2	TP	CA26	TF		
3	TP	SY2	TF1	}	Fixed → HQ No ↓ Not Ss. → HR Ss. ↓ Not LJKLM → HS Yes ↓ CW = 64xxx
4	TP	VA16	Q		
5	QJ	HQ	XQ6		
6	TP	VA5	A	}	CW → list
7	ZJ	XQ10	HR		
10	TP	SY10	Q		
11	QJ	XQ12	HS	}	File → CB List Set PC → (C)
12	RJ	TK	TK1		
13	AT	CA25	TF2		
14	TP	A	EW2	}	
15	RJ	EW	EW1		
16	RJ	TE	TE1		
17	TP	CA6	PC3		
20	MJ	0	ZM		
	CA	XQ21			

Not LJKLM

	IA	HS			
0	RJ	WA	WA1	}	E //
1	TP	SY2	EK6		
2	TP	VA6	EK16		
3	TP	EK	UP3	}	Assign 65xxx CW
4	RJ	UP2	UP		
5	RJ	TK	TK1		
6	AT	CA31	EW2	}	CW → list file → CB List Set PC → (C)
7	TP	A	TF2		
10	RJ	EW	EW1		
11	RJ	TE	TE1		
12	TP	CA6	PC3		
13	MJ	0	ZM		
	CA	HS14			

Not Ss. Mode

	IA	HR		
0	TP	CA6	PC3	Set PC
1	TP	SY10	Q	IJKLM → HU
2	QJ	HU	HR3	No ↓
3	RJ	TK	TK1	65xxx CW
4	AT	CA31	TF2	
5	TP	A	EW2	CW → list
6	RJ	EW	EW1	
7	RJ	TE	TE1	File → CB List
10	MJ	0	ZM	→ (C)
	CA	HR11		

IJKLM

	IA	HU		
0	RJ	WA	WA1	
1	TP	SY2	ER25	E12
2	TP	EK20	UP3	
3	RJ	UP2	UP	
4	RJ	TK	TK1	Assign 64xxx CW
5	AT	CA25	TF2	
6	TP	A	EW2	CW → list
7	RJ	EW	EW1	
10	RJ	TE	TE1	File → CB List
11	TP	CA6	PC3	Set PC
12	MJ	0	ZM	→ (C)
	CA	HU13		

Fixed Equation

	IA	HQ		
0	TP	SY10	Q	IJKLM → XQ11
1	QJ	XQ11	HQ2	No ↓
2	RJ	WA	WA1	
3	TP	SY2	EL17	E14
4	TP	EL12	UP3	
5	RJ	UP2	UP	
6	MJ	0	HS5	→ 65xxx CW
	CA	HQ7		

Right-After-Var. In P.Op. List

F		IA	XD		
	0	RJ	TS	TS1	} Not in P.Op. list → HV
	1	MJ	0	HV	
	2	TP	TS3	EW2	} In ↓
	3	RJ	EW	EW1	
	4	TP	CA6	PC3	} CW → string
	5	TP	VA16	Q	
	6	QJ	HW	XD7	} Set PC
	7	TP	VA5	A	
	10	ZJ	HX	XD11	} Fixed eq. → HW
	11	TP	TS3	Q	
	12	QT	CG17	A	} No ↓
	13	EJ	CG21	HY	
	14	EJ	CG22	HZ	} Ss. mode → HX
	15	TP	CA7	PC3	
	16	RJ	PC	PC1	} No ↓
	17	RJ	SY	SY1	
	20	EJ	CA30	NA	} xx000 → A
	21	RJ	WA	WA1	
	22	TP	SZ2	EP6	} 61xxx → HY
	23	TP	EP	UP3	
	24	RJ	UP2	UP	} 63xxx → HZ
	25	TP	SY2	A	
	26	MJ	0	ZM3	} 76Yxx ↓
		CA	XD27		

Sym → A
→ (C)

76Yxx (

		IA	NA		
0	TP	TS3	Q	} #Ss. → Q	
1	QT	CG25	Q		
2	LQ	Q	36	} # commas → VA36	
3	TP	Q	A		
4	ST	CA3	VA36	} Set Ss. mode	
5	TP	VA1	VA5		
6	TP	SZ2	VA6	} (CW → list	
7	TP	CG24	EW2		
10	RJ	EW	EW1	} Set PC	
11	TP	CA4	PC3		
12	MJ	0	ZN11	} → (C)	
	CA	NA13			

63xxx

0	IA	HZ	Q	}	IJKLM ↓ no → (92)
1	TP	SY10	ZM		
2	QJ	HZ2	WA1	}	E7
3	RJ	WA	EI5		
4	TP	SY2	UP3		
5	RJ	EI	UP		
6	MJ	UP2	ZM	}	→ (C)
	CA	0			
		HZ7			

61xxx

0	IA	HY	PC3	}	Do PC
1	TP	CA10	PC1		
2	RJ	PC	SY1	}	Sym → A
3	RJ	SY	WD		
4	EJ	CA30	ZM3		
	MJ	0			
	CA	HY5			(→ WD no ↓
					→ (C)

SS Mode

0	IA	HX	Q	}	xx000 → A
1	TP	TS3	A		
2	QT	CG17	NB	}	61xxx → NB
3	EJ	CG21	NC		
4	EJ	CG22	PC3		
5	TP	CA7	PC1		
6	RJ	PC	WA1	}	76Yxx ↓
7	RJ	WA	EH6		
10	TP	SY2	EH20		
11	TP	VA6	UP3		
12	RJ	EH	UP	}	Do PC
13	RJ	UP2	SY1		
14	RJ	SY	HX16		
15	RJ	CA30	ZM3		
16	EJ	0	FV1	}	E5
17	MJ	FV	YV		
18	MJ	0	ZM1		
20	MJ	0			
	CA	HX21			Sym → A
					(→ HX16
					→ (C)
					Delete SS
					Δ → (C2)
					→ (C)

Fixed Equation

	IA	HW			
0	TP	TS3	Q	}	xx000 → A
1	QT	CG17	A	}	
2	EJ	CG21	ND		61xxx → ND
3	EJ	CG22	NE		63xxx → NE
4	TP	CA7	PC3	}	76Yxx ↓
5	RJ	PC	PC1	}	Do PC
6	RJ	WA	WA1	}	
7	TP	SY2	EF24	}	E2
10	TP	EF16	UP3	}	
11	RJ	UP2	UP	}	
12	RJ	SY	SY1		Sym → A
13	EJ	CA30	HW15		(→ HW15
14	MJ	0	ZM3		→ (C)
15	TP	GP10	UP3	}	
16	RJ	UP2	UP	}	"Ss. not checked"
17	RJ	FV	FV1		Delete Ss.
20	MJ	0	YV		Δ → (C2)
21	MJ	0	ZM1		→ (C)
	CA	HW22			

61xxx

	IA	ND			
0	TP	CA10	PC3	}	
1	RJ	PC	PC1	}	Do PC
2	RJ	WA	WA1	}	
3	TP	SY2	EF4	}	
4	TP	EF	UP3	}	E1
5	RJ	UP2	UP	}	
6	RJ	SY	SY1		Sym → A
7	EJ	CA30	WD		(→ WD no ↓
10	MJ	0	ZM3		→ (C)
	CA	ND11			

63xxx

	IA	NE		
0	TP	SY10	Q	IJKLM → (C)
1	QJ	ZM	NE2	no ↓
2	RJ	WA	WA1	
3	TP	SY2	EG6	E3
4	TP	EG	UP3	
5	RJ	UP2	UP	
6	MJ	0	ZM	→ (C)
	CA	NE7		

SS Mode 63xxx

	IA	NC		
0	TP	SY10	Q	IJKLM → (C)
1	QJ	ZM	NC2	No ↓
2	RJ	WA	WA1	
3	TP	SY2	EH30	
4	TP	VA6	EH42	E6
5	TP	EH22	UP3	
6	RJ	UP2	UP	
7	MJ	0	ZM	→ (C)
	CA	NC10		

61xxx

	IA	NB		
0	TP	CA10	PC3	
1	RJ	PC	PC1	Do PC
2	RJ	WA	WA1	
3	TP	SY2	EG24	
4	TP	VA6	EG36	E4
5	TP	EG20	UP3	
6	RJ	UP2	UP	
7	RJ	SY	SY1	Sym → A
10	EJ	CA30	WD	(→ WD
11	MJ	0	ZM3	→ (C)
	CA	NB12		

Right, After, Var. Not in Pseudo Op. List

0	IA	HV			
1	RJ	TA	TA1	}	
2	MJ	O	NF	}	Not in CB List → NF in ↓
3	TP	TA4	Q	}	4xxxx → FU
4	QT	CG30	A	}	
5	EJ	CG31	FU	}	No ↓
6	TP	TA4	EW2	}	
7	RJ	EW	EW1	}	CW → list
10	TP	CA6	PC3	}	Set PC
11	TP	VA16	Q	}	Fixed → NG
12	QJ	NG	HV12	}	Floating ↓
13	TP	VA5	A	}	Ss. mode → NH
14	ZJ	NH	HV14	}	No ↓
15	TP	TA4	Q	}	
16	QT	CG17	A	}	CW → A
17	EJ	CG17	NI		77xxx → NI
20	EJ	CA25	NJ		64xxx → NJ
21	EJ	CA31	ZM		65xxx → (C)
22	EJ	CA33	NL		66xxx → NL
23	TP	CA20	PC3	}	5xxxx ↓
24	RJ	PC	PC1	}	Do PC
25	TP	TA4	Q	}	CW → A → VA13
26	QT	CI31	VA13	}	
27	TJ	CH25	ZM1		# operands = 1 → (C) no ↓
30	RJ	SY	SY1		Sym. → A
31	EJ	CA30	HG		(→ HG no ↓
32	RJ	WA	WA1	}	
33	TP	SZ2	EN5	}	E17
34	TP	EN	UP3	}	
35	RJ	UP2	UP	}	
36	TP	SY2	A		Sym. → A
	MJ	O	ZM3		→ (C)
	CA	HV37			

66xxx

0	IA	NL		
1	TP	CA10	PC3	}
2	RJ	PC	PC1	}
3	RJ	SY	SY1	
4	EJ	CA30	WD	
	MJ	O	ZM3	
	CA	NL5		

Pair check

Sym → A

(→ WD

→ (C)

64xxx

	IA	NJ		
0	RJ	WA	WA1	
1	TP	SY2	EK25	
2	TP	EK20	UP3	E12
3	RJ	UP2	UP	
4	MJ	0	ZM	→ (C)
	CA	NJ5		

77xxx

	IA	NI		
0	TP	CA7	PC3	Do PC
1	RJ	PC	PC1	
2	RJ	SY	SY1	Sym. → A
3	EJ	CA30	NI12	(→ NI12 no ↓
4	RJ	WA	WA1	
5	TP	SZ2	EN30	E18
6	TP	EN22	UP3	
7	RJ	UP2	UP	
10	TP	SY2	A	Sym → A
11	MJ	0	ZM3	→ (C)
12	TP	VA1	VA5	Set Ss. mode
13	TP	SZ2	VA6	
14	TP	TA5	Q	
15	QT	CA2	A	Ss. - 1 → storage
16	ST	CA3	VA36	
17	TP	CG24	EW2	(→ list
20	RJ	EW	EW1	
21	TP	CA4	PC3	Set PC
22	MJ	0	ZN11	→ (section → (C)
	CA	NI23		

Ss. Mode

0	IA	NH		
1	TP	TA4	Q	} xx000 → A
2	QT	CG17	A	
3	EJ	CG17	NK	77xxx → NK
4	EJ	CA25	ZM	64xxx → (C)
5	EJ	CA31	NP	65xxx → NP
6	EJ	CA33	NS	66xxx → NS
7	TP	CA20	PC3	} 5xxxx ↓
10	RJ	PC	PC1	
11	RJ	WA	WA1	Do PC
12	TP	SY2	EJ4	} E9
13	TP	VA6	EJ13	
14	RJ	EJ	UP3	
15	RJ	UP2	UP	
16	RJ	SY	SY1	Sym. → A
17	EJ	CA30	NH20	(→ NH20 no ↓
20	TP	GP25	UP3	→ (C)
21	RJ	UP2	UP	"Args not checked"
22	RJ	FV	FV1	Delete Lib.
23	MJ	0	YV	Δ → YV
24	MJ	0	ZM1	→ (C)
	CA	NH25		

66xxx

0	IA	NS		
1	TP	CA10	PC3	} Do pair check
2	RJ	PC	PC1	
3	RJ	WA	WA1	} E10
4	TP	SY2	EJ20	
5	TP	VA6	EJ25	
6	TP	EJ15	UP3	
7	RJ	UP2	UP	
10	RJ	SY	SY1	Sym → A
11	EJ	CA30	NS12	(→ NS12 no ↓
12	MJ	0	ZM3	→ (C)
13	TP	GP	UP3	"Args not checked"
14	RJ	UP2	UP	Delete arguments
15	RJ	FV	FV1	Δ, → (C2)
16	MJ	0	YV	→ (C)
	MJ	0	ZM1	
	CA	NS17		

65xxx

	IA	NP		
0	RJ	WA	WA1	
1	TP	SY2	EK6	
2	TP	VA6	EK16	E11
3	TP	EK	UP3	
4	RJ	UP2	UP	
5	MJ	0	ZM	→ (C)
	CA	NP6		

77xxx

	IA	NK		
0	TP	CA7	PC3	
1	RJ	PC	PC1	Do PC
2	RJ	WA	WA1	
3	TP	SY2	EI26	
4	TP	VA6	EI35	E8
5	TP	EI20	UP3	
6	RJ	UP2	UP	
7	RJ	SY	SY1	Sym → A
10	EJ	CA30	NK12	(→ NK12 no ↓
11	MJ	0	ZM3	→ (C)
12	TP	GP17	UP3	
13	RJ	UP2	UP	"Ss. not checked"
14	RJ	FV	FV1	Delete Ss.
15	MJ	0	YV	Δ. → (C2)
16	MJ	0	ZM1	→ (C)
	CA	NK17		

Fixed Equation

	IA	NG			
0	TP	TA4	Q }	xx000 →	A
1	QT	CG17	A }		
2	EJ	CG17	NN	77xxx →	NN
3	EJ	CA33	NV	66xxx →	NV
4	EJ	CA31	NW	65xxx →	NW
5	EJ	CA25	ZM	64xxx →	Ⓒ
6	TP	CA20	PC3 }	5xxxx	
7	RJ	PC	PC1 }	Do PC	
10	RJ	WA	WA1		
11	TP	SY2	EL4		
12	TP	EL	UP3	E13	
13	RJ	UP2	UP		
14	RJ	SY	SY1	Sym →	A
15	EJ	CA30	NX	(→ NX	no ↓
16	MJ	0	ZM3	→	Ⓒ
	CA	NG17			

5xxxx (

	IA	NX			
0	TP	GP	UP3 }	"Args not checked"	
1	RJ	UP2	UP }		
2	RJ	FV	FV1	Delete Lib.	
3	MJ	0	YV	Δ . →	Ⓒ2
4	MJ	0	ZM1	→	Ⓒ
	CA	NX5			

65xxx

	IA	NW			
0	RJ	WA	WA1 }		
1	TP	SY2	EL17 }	E14	
2	TP	EL12	UP3 }		
3	RJ	UP2	UP }		
4	MJ	0	ZM	→	Ⓒ
	CA	NW5			

66xxx

	IA	NV		
0	TP	CA10	PC3	} Do PC
1	RJ	PC	PC1	
2	RJ	WA	WA1	} E15
3	TP	SY2	EM3	
4	TP	EM	UP3	
5	RJ	UP2	UP	
6	RJ	SY	SY1	Sym → A
7	EJ	CA30	NV11	(→ NV11
10	MJ	0	ZM3	→ (C)
11	TP	GP	UP3	} "Args not checked"
12	RJ	UP2	UP	
13	RJ	FV	FV1	Delete args
14	MJ	0	YV	→ (C2)
15	MJ	0	ZM1	→ (C)
	CA	NV16		

77xxx

	IA	NN		
0	TP	CA7	PC3	} Do PC
1	RJ	PC	PC1	
2	RJ	WA	WA1	} E16
3	TP	SY2	EM17	
4	TP	EM11	UP3	
5	RJ	UP2	UP	
6	RJ	SY	SY1	Sym. → A
7	EJ	CA30	NN11	(→ NN11 no ↓
10	MJ	0	ZM3	→ (C)
11	TP	GP17	UP3	} "Ss. not checked"
12	RJ	UP2	UP	
13	RJ	FV	FV1	Delete Ss.
14	MJ	0	YV	Δ. → (C2)
15	MJ	0	ZM1	→ (C)
	CA	NN16		

Right, After, Not in P.Op. list

4xxxx

	IA	FU		
0	TP	CA6	PC3	} Do PC
1	RJ	PC	PC1	
2	RJ	WA	WA1	} E51
3	TP	SY2	GQ27	
4	TP	GQ21	UP3	
5	RJ	UP2	UP	
6	RJ	SY	SY1	Sym. → A
7	EJ	CA30	FU11	(→ FU11 no ↓
10	MJ	0	ZM3	→ Ⓒ
11	TP	GP26	UP3	} "Operands not checked"
12	RJ	UP2	UP	
13	RJ	FV	FV1	Delete operands
14	MJ	0	YV	Δ. → Ⓒ2
15	MJ	0	ZM1	→ Ⓒ
	CA	FU16		

Var. - Right - After Not in P.Op. List
Not in CB List

	IA	NF		
0	RJ	RH	RH1	} Check Var. sym. Set to add to CB List
1	TP	CA26	TF	
2	TP	SY2	TF1	
3	TP	CA27	TF3	
4	TP	CA6	PC3	Set PC
5	TP	VA16	Q	Fixed eq. → NZ
6	QJ	NZ	NF7	No ↓
7	TP	VA5	A	Ss. mode → FJ
10	ZJ	FJ	NF11	No ↓
11	TP	SY10	Q	IJKLM → FK
12	QJ	FK	NF13	No ↓
13	RJ	TK	TK1	Increase CW
14	AT	CA31	EW2	65xxx CW → list
15	RJ	EW	EW1	
16	TP	EW2	TF2	→ CB List
17	RJ	TE	TE1	
20	MJ	0	ZM	→ Ⓒ
	CA	NF21		

Not SS mode - IJKLM

	IA	FK		
0	RJ	WA	WA1	
1	TP	SY2	EK25	
2	TP	EK20	UP3	E12
3	RJ	UP2	UP	
4	RJ	TK	TK1	Increase CW counter
5	AT	CA25	EW2	64xxx CW → list
6	TP	A	TF2	
7	RJ	EW	EW1	
10	RJ	TE	TE1	CW → CB list
11	MJ	0	ZM	→ (C)
	CA	FK12		

Ss. Mode

	IA	FJ		
0	TP	SY10	Q	IJKLM → FK4
1	QJ	FK4	FJ2	No ↓
2	RJ	WA	WA1	
3	TP	SY2	EK6	E11
4	TP	EK	UP3	
5	RJ	UP2	UP	
6	MJ	0	NF13	
7	TP	VA6	EK16	
10	MJ	0	FJ2	
	CA	FJ11		

Fixed Equation

	IA	NZ		
0	TP	SY10	Q	IJKLM — FK4
1	QJ	FK4	NZ2	No ↓
2	RJ	WA	WA1	
3	TP	SY2	EL17	E14
4	TP	EL12	UP3	
5	RJ	UP2	UP	
6	MJ	0	NF13	
	CA	NZ7		

△.

(C2)

0	IA	YV		
1	TP	CG16	EW2	}
2	RJ	EW	EW1	
3	TP	VA7	A	}
4	ZJ	YV4	YV7	
5	TP	VA37	Q	}
6	QJ	YV6	YV7	
7	RJ	TE	TE1	
10	TP	VA14	A	}
11	EJ	CA3	YV14	
12	RJ	WA	WA1	}
13	TP	GK	UP3	
14	RJ	UP2	UP	}
15	TP	VA1	A	
16	EJ	CA3	YV21	}
17	RJ	WA	WA1	
20	TP	GK10	UP3	}
21	RJ	UP2	UP	
22	TP	VA2	A	}
23	EJ	CA3	YV26	
24	RJ	WA	WA1	}
25	TP	GL	UP3	
26	RJ	UP2	UP	}
27	TP	CA22	PC3	
30	RJ	PC	PC1	}
31	TP	EW3	Q	
32	QT	CA2	A	}
33	ST	CK11	A	
34	AT	CA3	WL	}
35	RJ	SS	SS1	
	MJ	0	FR	
	CA	YV36		

C.W. → list
Ftn. ↓ no → YV7
Not in CB List ↓ in → YV7
Add to list
= 's = 1 → YV14 no ↓
E44
On (level zero → YV21
No ↓
E45
On level zero → YV26
No ↓
E46
→ PC
Word count → WL
Send S.O. to tape
Exit → Control

Equation Translation Error Prints

Region	Error Numbers
EF	E1,E2
EG	E3,E4
EH	E5,E6
EI	E7,E8
EJ	E9,E10
EK	E11,E12
EL	E13,E14
EM	E15,E16
EN	E17,E18
EP	E19,E20
ER	E21,E22
EX	E23,E24
GA	E25,E26
GB	E27,E28
GC	E29,E30
GD	E31,E32
GE	E33,E34
GF	E35,E36
GH	E37,E38
GI	E39,E40
GJ	E41,E42,E43
GK	E44,E45
GL	E46,E47
GM	E48,E49
GP	Suffixes of error prints
GQ	E50,E51
FQ	E52
GV	E53,E54,E55
GY	E56,E57
GZ	E58

Error Prints for Equation Translation

	IA	EF			
	0	40	EF1	15	
E1	1	31	67502	66634	F U N C T I
	2	51	50016	57347	O N Δ S Y M
	3	25	51462	10177	B O L , Δ 77
	4	0	0	0	Sym.
	5	01	21011	76567	Δ , Δ (S U
	6	25	01525	45132	B Δ P R O G
	7	54	24470	12767	R A M Δ D U
	10	47	47734	30134	M M Y) Δ I
	11	50	01010	10101	N Δ Δ Δ Δ Δ
	12	01	31347	23027	Δ F I X E D
	13	01	52513	45066	Δ P O I N T
	14	01	30536	72466	Δ E Q U A T
	15	34	51502	27777	I O N . 77 77
E2	16	40	EF17	20	
	17	65	67256	52654	S U B S C R
	20	34	52663	02701	I P T E D Δ
	21	70	24543	42425	V A R I A B
	22	46	30016	57347	L E Δ S Y M
	23	25	51462	10177	B O L , Δ 77
	24	0	0	0	Sym.
	25	01	21011	76567	Δ , Δ (S U
	26	25	01010	10101	B Δ Δ Δ Δ Δ
	27	01	01010	10101	Δ Δ Δ Δ Δ Δ
	30	01	52545	13254	Δ P R O G R
	31	24	47012	76747	A M Δ D U M
	32	47	73430	13450	M Y) Δ I N
	33	01	31347	23027	Δ F I X E D
	34	01	52513	45066	Δ P O I N T
	35	01	30536	72466	Δ E Q U A T
	36	34	51502	27777	I O N . 77 77
	CA		EF37		

E3	0	LA	EG	17	
	1	40	EG1	46634	F L O A T I
	2	31	46512	25134	N G Δ P O I
	3	50	32015	02454	N T Δ V A R
	4	50	66017	63021	I A B L E .
	5	34	24254	77777	Δ 77 77 77 77 77
	6	01	77777	0	Var.
	7	0	0	76567	Δ , Δ (S U
	10	01	21011	45132	B Δ P R O G
	11	25	01525	10101	R A M Δ Δ Δ
	12	54	24470	10127	Δ Δ Δ Δ Δ D
	13	01	01010	34301	U M M Y) Δ
	14	67	47477	13472	I N Δ F I X
	15	34	50013	25034	E D Δ P O I
	16	30	27015	05367	N T Δ E Q U
	17	50	66013	15022	A T I O N .
E4	20	24	66345	17	
	21	40	EG21	66634	F U N C T I
	22	31	67502	57347	O N Δ S Y M
	23	51	50016	10177	B O L , Δ 77
	24	25	51462	0	Sym.
	25	0	0	76567	Δ , Δ (S U
	26	01	21011	45132	B Δ P R O G
	27	25	01525	12767	R A M Δ D U
	30	54	24470	30101	M M Y) Δ Δ
	31	47	47734	10101	Δ Δ Δ Δ Δ Δ
	32	01	01010	15032	Δ A M O N G
	33	01	24475	56526	Δ S U B S C
	34	01	65672	66501	R I P T S Δ
	35	54	34526	77777	O F Δ 77 77 77
	36	51	31017	0	Sym.
	37	0	0	77777	. 77 77 77 77 77
		22	77777		
		CA	EG40		

E5	0	IA	EH	21	
	1	40	EH1	52654	S U B S C R
	2	65	67256	02701	I P T E D Δ
	3	34	52663	42425	V A R I A B
	4	70	24543	57347	L E Δ S Y M
	5	46	30016	10177	B O L , Δ 77
	6	25	51462	0	Sym.
	7	0	0	76567	Δ , Δ (S U
	10	01	21011	10101	B Δ Δ Δ Δ Δ
	11	25	01010	10101	Δ Δ Δ Δ Δ Δ
	12	01	01010	13254	Δ P R O G R
	13	01	52545	76747	A M Δ D U M
	14	24	47012	12447	M Y) Δ A M
	15	47	73430	16567	O N G Δ S U
	16	51	50320	43452	B S C R I P
	17	25	65265	13101	T S Δ O F Δ
	20	66	65015	0	Sym.
	21	0	0	77777	. 77 77 77 77 77
E6	22	22	77777	21	
	23	40	EH23	46634	F L O A T I
	24	31	46512	25134	N G Δ P O I
	25	50	32015	02454	N T Δ V A R
	26	50	66017	63021	I A B L E ,
	27	34	24254	77777	Δ 77 77 77 77 77
	30	01	77777	0	Var.
	31	0	0	76567	Δ , Δ (S U
	32	01	21011	45132	B Δ P R O G
	33	25	01525	10101	R A M Δ Δ Δ
	34	54	24470	10127	Δ Δ Δ Δ Δ D
	35	01	01010	34301	U M M Y) Δ
	36	67	47477	03201	A M O N G Δ
	37	24	47515	52654	S U B S C R
	40	65	67256	50151	I P T S Δ O
	41	34	52666	77777	F Δ 77 77 77 77
	42	31	01777	0	Sym.
	43	0	0	77777	. 77 77 77 77 77
	43	22	77777		
		CA	EH44		

E7	0	IA	EI	17	
	1	40	EI1	02701	F I X E D Δ
	2	31	34723	06601	P O I N T Δ
	3	52	51345	42425	V A R I A B
	4	70	24543	17777	L E , Δ 77 77
	5	46	30210	0	Var.
	6	0	0	76567	Δ , Δ (S U
	7	01	21011	45132	B Δ P R O G
	10	25	01525	10101	R A M Δ Δ Δ
	11	54	24470	10101	Δ Δ Δ Δ Δ Δ
	12	01	01010	74747	Δ Δ Δ D U M M
	13	01	01276	45001	Y) Δ I N Δ
	14	73	43013	46634	F L O A T I
	15	31	46512	25134	N G Δ P O I
	16	50	32015	05367	N T Δ E Q U
	17	50	66013	15022	A T I O N .
E8	20	24	66345	16	
	21	40	EI21	52654	S U B S C R
	22	65	67256	02701	I P T E D Δ
	23	34	52663	42425	V A R I A B
	24	70	24543	57347	L E Δ S Y M
	25	46	30016	10177	B O L , Δ 77
	26	25	51462	0	Var.
	27	0	0	44751	Δ , Δ A M O
	30	01	21012	10101	N G Δ Δ Δ Δ
	31	50	32010	10101	Δ Δ Δ Δ Δ Δ
	32	01	01010	56526	Δ S U B S C
	33	01	65672	66501	R I P T S Δ
	34	54	34526	77777	O F Δ 77 77 77
	35	51	31017	0	Var.
	36	0	0	77777	. 77 77 77 77 77
		22	77777		
		CA	EI37		

E9	0	IA	EJ	14	
	1	40	EJ1	46766	L I B R A R
	2	46	34255	16766	Y Δ R O U T
	3	73	01545	10177	I N E , Δ 77
	4	34	50302	0	Sym.
	5	0	0	44751	Δ , Δ A M O
	6	01	21012	56725	N G Δ S U B
	7	50	32016	45266	S C R I P T
	10	65	26543	10101	S Δ O F Δ Δ
	11	65	01513	10101	Δ Δ Δ Δ Δ Δ
	12	01	01010	77777	Δ Δ Δ 77 77 77
	13	01	01017	0	Sym.
	14	0	0	77777	. 77 77 77 77 77
E10	15	22	77777	11	
	16	40	EJ16	66634	F U N C T I
	17	31	67502	17777	O N , Δ 77 77
	20	51	50210	0	Sym.
	21	0	0	44751	Δ , Δ A M O
	22	01	21012	56725	N G Δ S U B
	23	50	32016	45266	S C R I P T
	24	65	26543	10177	S Δ O F Δ 77
	25	65	01513	0	Sym.
	26	0	0	77777	. 77 77 77 77 77
		22	77777		
		CA	EJ27		

E11	0	IA	EK	17	
	1	40	EK1	46634	F L O A T I
	2	31	46512	25134	N G Δ P O I
	3	50	32015	02454	N T Δ V A R
	4	50	66017	63021	I A B L E ,
	5	34	24254	77777	Δ 77 77 77 77 77
	6	01	77777	0	Var.
	7	0	0	44751	Δ , Δ A M O
	10	01	21012	10101	N G Δ Δ Δ Δ
	11	50	32010	10101	Δ Δ Δ Δ Δ Δ
	12	01	01010	10165	Δ Δ Δ Δ Δ S
	13	01	01010	65434	U B S C R I
	14	67	25652	15131	P T S Δ O F
	15	52	66650	77777	Δ 77 77 77 77 77
	16	01	77777	0	Sym.
	17	0	0	77777	. 77 77 77 77 77
E12	20	22	77777	13	
	21	40	EK21	02701	F I X E D Δ
	22	31	34723	06601	P O I N T Δ
	23	52	51345	42425	V A R I A B
	24	70	24543	17777	L E , Δ 77 77
	25	46	30210	0	Var.
	26	0	0	45001	Δ , Δ I N Δ
	27	01	21013	46634	F L O A T I
	30	31	46512	25134	N G Δ P O I
	31	50	32015	10101	N T Δ Δ Δ Δ
	32	50	66010	36724	Δ Δ E Q U A
	33	01	01305	02277	T I O N . 77
		66	34515		
		CA	EK34		

E13	0	IA	EL	11	
	1	40	EL1	42454	L I B R A R
	2	46	34255	16766	Y Δ R O U T
	3	73	01545	10177	I N E , Δ 77
	4	34	50302	0	Sym.
	5	0	0	00131	, Δ I N Δ F
	6	21	01345	70152	I X E D Δ P
	7	34	72302	60130	O I N T Δ E
	10	51	34506	63451	Q U A T I O
	11	53	67246	77777	N . 77 77 77 77
E14	12	50	22777	13	F L O A T I
	13	40	EL13	46634	N G Δ P O I
	14	31	46512	25134	N T Δ S Y M
	15	50	32015	57347	B O L , Δ 77
	16	50	66016	10177	Sym.
	17	25	51462	0	Δ , Δ I N Δ
	20	0	0	45001	F I X E D Δ
	21	01	21013	02701	P O I N T Δ
	22	31	34723	06601	Δ Δ Δ Δ Δ Δ
	23	52	51345	10101	Δ E Q U A T
	24	01	01010	72466	I O N . 77 77
	25	01	30536	27777	
	26	34	51502		
		CA	EL26		

		IA	EM		
E15	0	40	EM1	10	
	1	31	67502	66634	F U N C T I
	2	51	50210	17777	O N , Δ 77 77
	3	0	0	0	Sym.
	4	01	21013	45001	Δ , Δ I N Δ
	5	31	34723	02701	F I X E D Δ
	6	52	51345	06601	P O I N T Δ
	7	30	53672	46634	E Q U A T I
	10	51	50227	77777	O N . 77 77 77
E16	11	40	EM12	14	
	12	65	67256	52654	S U B S C R
	13	34	52663	02701	I P T E D Δ
	14	70	24543	42425	V A R I A B
	15	46	30016	57347	L E Δ S Y M
	16	25	51462	10177	B O L , Δ 77
	17	0	0	0	Sym.
	20	01	21013	45001	Δ , Δ I N Δ
	21	31	34723	02701	F I X E D Δ
	22	01	01010	10101	Δ Δ Δ Δ Δ Δ
	23	01	52513	45066	Δ P O I N T
	24	01	30536	72466	Δ E Q U A T
	25	34	51502	27777	I O N . 77 77
		CA	EM26		

E17	0	IA	EN	21	
	1	40	EN1	42454	L I B R A R
	2	46	34255	16766	Y Δ R O U T
	3	73	01545	16573	I N E Δ S Y
	4	34	50300	62101	M B O L , Δ
	5	47	24514	0	Sym.
	6	0	0	13466	Δ , Δ W I T
	7	01	21017	15430	H Δ M O R E
	10	33	01475	45001	Δ T H A N Δ
	11	01	66332	10101	Δ Δ Δ Δ Δ Δ
	12	01	01010	12454	O N E Δ A R
	13	51	50300	05066	G U M E N T
	14	32	67473	16601	, Δ N O T Δ
	15	21	01505	65171	F O L L O W
	16	31	51464	57301	E D Δ B Y Δ
	17	30	27012	00152	O P E N Δ P
	20	51	52305	06633	A R E N T H
	21	24	54305	52277	E S I S . 77
E18	22	30	65346	17	S U B S C R
	23	40	EN23	52654	I P T E D Δ
	24	65	67256	02701	V A R I A B
	25	34	52663	42425	L E Δ S Y M
	26	70	24543	57347	B O L , Δ 77
	27	46	30016	10177	Sym.
	30	25	51462	0	Δ , Δ N O T
	31	0	0	05166	Δ Δ Δ Δ Δ Δ
	32	01	21015	10101	Δ Δ Δ Δ Δ Δ
	33	01	01010	10101	Δ Δ Δ Δ Δ Δ
	34	01	01010	64651	Δ F O L L O
	35	01	31514	12573	W E D Δ B Y
	36	71	30270	15152	Δ A N Δ O P
	37	01	24500	22454	E N Δ P A R
	40	30	50015	33065	E N T H E S
	41	30	50663	77777	I S . 77 77 77
		34	65227		
		CA	EN42		

E19	0	IA	EP	21	
	1	40	EP1	52654	S U B S C R
	2	65	67256	02701	I P T E D Δ
	3	34	52663	42425	V A R I A B
	4	70	24543	57347	L E Δ S Y M
	5	46	30016	10177	B O L , Δ 77
	6	25	51462	0	Sym.
	7	0	0	56725	, Δ (S U B
	10	21	01176	13254	Δ P R O G R
	11	01	52545	10101	A M Δ Δ Δ Δ
	12	24	47010	74773	Δ D U M M Y
	13	01	27674	16601	) Δ N O T Δ
	14	43	01505	65171	F O L L O W
	15	31	51464	57301	E D Δ B Y Δ
	16	30	27012	15230	A N Δ O P E
	17	24	50015	45430	N Δ P A R E
	20	50	01522	06534	N T H E S I
	21	50	66333	06534	N T H E S I
	21	65	22777	77777	S . 77 77 77 77
E20	22	40	EP23	20	
	23	65	67256	62654	S U B S C R
	24	34	52663	02701	I P T E D Δ
	25	70	24543	42425	V A R I A B
	26	46	30016	57347	L E Δ S Y M
	27	25	51462	10177	B O L , Δ 77
	30	0	0	0	Sym.
	31	01	21011	73167	Δ , Δ (F U
	32	50	26663	45150	N C T I O N
	33	01	01010	10101	Δ Δ Δ Δ Δ Δ
	34	01	27674	74773	Δ D U M M Y
	35	43	01244	75150	) Δ A M O N
	36	32	01656	72565	G Δ S U B S
	37	26	54345	26665	C R I P T S
	40	01	51310	17777	Δ O F Δ 77 77
	41	0	0	0	Sym.
	42	22	77777	77777	. 77 77 77 77 77
		CA	EP43		

E21	0	IA	ER	21	
	1	40	ER1	46634	F L O A T I
	2	31	46512	25134	N G Δ P O I
	3	50	32015	02454	N T Δ V A R
	4	50	66017	63021	I A B L E ,
	5	34	24254	77777	Δ 77 77 77 77 77
	6	01	77777	0	Sym.
	7	0	0	73167	Δ , Δ (F U
	10	01	21011	45150	N C T I O N
	11	50	26663	10101	Δ Δ Δ Δ Δ Δ
	12	01	01010	10127	Δ Δ Δ Δ Δ D
	13	01	01010	34301	U M M Y) Δ
	14	67	47477	03201	A M O N G Δ
	15	24	47515	52654	S U B S C R
	16	65	67256	50151	I P T S Δ O
	17	34	52666	77777	F Δ 77 77 77 77
	20	31	01777	0	Sym.
	21	0	0	77777	. 77 77 77 77 77
E22	22	22	77777	16	
	23	40	ER23	02701	F I X E D Δ
	24	31	34723	06601	P O I N T Δ
	25	52	51345	42425	V A R I A B
	26	70	24543	17777	L E , Δ 77 77
	27	46	30210	0	Sym.
	30	0	0	73167	Δ , Δ (F U
	31	01	21011	45150	N C T I O N
	32	50	26663	74773	Δ D U M M Y
	33	01	27674	10101	) Δ Δ Δ Δ Δ
	34	43	01010	00131	Δ Δ I N Δ F
	35	01	01345	63450	L O A T I N
	36	46	51246	13450	G Δ P O I N
	37	32	01525	36724	T Δ E Q U A
	40	66	01305	02277	T I O N . 77
		66	34515		
		CA	ER41		

		IA	EX		
E23	0	40	EX1	21	
	1	65	67256	52654	S U B S C R
	2	34	52663	02701	I P T E D Δ
	3	70	24543	42425	V A R I A B
	4	46	30016	57347	L E Δ S Y M
	5	25	51462	10177	B O L , Δ 77
	6	0	0	0	Sym.
	7	01	21011	73167	Δ , Δ (F U
	10	50	26663	45150	N C T I O N
	11	01	01010	10101	Δ Δ Δ Δ Δ Δ
	12	01	27674	74773	Δ D U M M Y
	13	43	01505	16601	) Δ N O T Δ
	14	31	51464	65171	F O L L O W
	15	30	27012	57301	E D Δ B Y Δ
	16	24	50015	15230	A N Δ O P E
	17	30	01522	45430	N Δ P A R E
	20	50	66333	06534	N T H E S I
	21	65	22777	77777	S . 77 77 77 77
E24	22	40	EX23	10	
	23	47	51543	00166	M O R E Δ T
	24	33	24500	15150	H A N Δ O N
	25	30	01653	05224	E Δ S E P A
	26	54	24663	00130	R A T E Δ E
	27	53	67246	63451	Q U A T I O
	30	50	01315	15401	N Δ F O R Δ
	31	0	0	0	Sym.
	32	22	77777	77777	. 77 77 77 77 77
		CA	EX33		

E25	0	IA	GA	5	
	1	40	GA1	05431	S U P E R F
	2	65	67523	76501	L U O U S Δ
	3	46	67516	55146	S Y M B O L
	4	65	73472	00146	S Δ O N Δ L
	5	65	01515	27777	E F T . 77 77
E26	6	30	31662	23	
	7	40	GA7	66634	F U N C T I
	10	31	67502	17777	O N , Δ 77 77
	11	51	50210	0	Sym.
	12	0	0	15001	Δ , Δ O N Δ
	13	01	21015	62101	L E F T , Δ
	14	46	30316	13151	N O T Δ F O
	15	50	51660	13027	L L O W E D
	16	46	46517	12450	Δ B Y Δ A N
	17	01	25730	10101	Δ Δ Δ Δ Δ Δ
	20	01	01010	23050	Δ Δ O P E N
	21	01	01515	43050	Δ P A R E N
	22	01	52245	53465	T H E S I S
	23	66	33306	43065	. Δ Δ R E S
	24	22	01015	10166	T Δ O F Δ T
	25	66	01513	16530	H I S Δ S E
	26	33	34650	02630	N T E N C E
	27	50	66305	60126	Δ N O T Δ C
	30	01	50516	53027	H E C K E D
	31	33	30264	77777	. 77 77 77 77 77
		22	77777		
		CA	GA32		

E27	0	IA	GB	21	
	1	40	GB1	42454	L I B R A R
	2	46	34255	16766	Y Δ R O U T
	3	73	01545	16573	I N E Δ S Y
	4	34	50300	62101	M B O L , Δ
	5	47	25514	0	Sym.
	6	0	0	0	Δ , Δ I S Δ
	7	01	21013	46501	F I R S T Δ
	10	31	34546	55146	S Y M B O L
	11	65	73472	10101	Δ Δ Δ Δ Δ Δ
	12	01	01010	53050	O F Δ S E N
	13	51	31016	63022	T E N C E .
	14	66	30502	06566	Δ Δ R E S T
	15	01	01543	16633	Δ O F Δ T H
	16	01	51310	53050	I S Δ S E N
	17	34	65016	63001	T E N C E Δ
	20	66	30502	12633	N O T Δ C H
	21	50	51660	02722	E C K E D .
	22	30	26453	20	
E28	22	40	GB23	05367	A N Δ E Q U
	23	24	50013	15001	A T I O N Δ
	24	24	66345	17777	F O R Δ 77 77
	25	31	51540	0	Sym.
	26	0	0	0	Δ I N Δ T H
	27	01	34500	16633	E Δ R A N G
	30	30	01542	10124	E Δ O F Δ A
	31	30	01513	10101	Δ V A R Y Δ
	32	01	70245	05066	Δ Δ Δ Δ Δ Δ
	33	01	01010	00134	Δ Δ S E N T
	34	01	01653	33426	E N C E Δ I
	35	30	50263	77777	N Δ W H I C
	36	50	01713	0	H Δ 77 77 77 77
	37	33	01777	23024	Symbol
	40	0	0	77777	Δ A P P E A
	41	01	24525		R S . 77 77 77
	42	54	65227		
		CA	GB43		

E29	0	IA	GC	11	
	1	40	GC1	03224	I L L E G A
	2	34	46463	34725	L Δ S Y M B
	3	46	01657	70101	0 L Δ (Δ Δ
	4	51	46011	0	Sym.
	5	0	0	13151	Δ Δ) Δ F O
	6	01	01170	03166	R Δ L E F T
	7	54	01463	13053	Δ 0 F Δ E Q
	10	01	51310	45150	U A T I O N
	11	67	24663	77777	. 77 77 77 77 77
E30	12	22	77777	16	
	13	40	GC13	00166	M O R E Δ T
	14	47	51543	15150	H A N Δ O N
	15	33	24500	72565	E Δ S U B S
	16	30	01656	26601	C R I P T Δ
	17	26	54345	77777	0 N Δ 77 77 77
	20	51	50017	0	Sym.
	21	0	0	00124	, Δ A N Δ A
	22	21	01245	73050	R G U M E N
	23	54	32674	10101	T Δ O F Δ Δ
	24	66	01513	63330	Δ Δ Δ T H E
	25	01	01016	02666	Δ F U N C T
	26	01	31675	17777	I O N Δ 77 77
	27	34	51500	0	Sym.
	30	0	0	77777	. 77 77 77 77 77
		22	77777		
		CA	GC31		

E31	0	IA	GD	20	
	1	40	GD1	42454	L I B R A R
	2	46	34255	16766	Y Δ R O U T
	3	73	01545	16573	I N E Δ S Y
	4	34	50300	62101	M B O L , Δ
	5	47	25514	0	Sym.
	6	0	0	15001	Δ , Δ O N Δ
	7	01	21015	62101	L E F T , Δ
	10	46	30316	03201	A M O N G Δ
	11	24	47515	10101	Δ Δ Δ Δ Δ Δ
	12	01	01010	74730	A R G U M E
	13	24	54326	15131	N T S Δ O F
	14	50	66650	00131	Δ T H E Δ F
	15	01	66333	63451	U N C T I O
	16	67	50266	77777	N Δ 77 77 77 77
	17	50	01777	0	Sym.
	20	0	0	77777	. 77 77 77 77 77
E32	21	22	77777	17	S U P E R S
	22	40	GD22	05465	C R I P T Δ
	23	65	67523	26601	S Y M B O L
	24	26	54345	55146	, Δ 77 77 77 77
	25	65	73472	77777	Sym.
	26	21	01777	0	Δ , Δ A M O
	27	0	0	0	N G Δ Δ Δ Δ
	30	0	0	44751	Δ Δ Δ Δ Δ Δ
	31	01	21012	10101	Δ Δ Δ Δ S U
	32	50	32010	10101	B S C R I P
	33	01	01010	10101	T S Δ O F Δ
	34	01	01010	16567	Sym.
	35	01	01010	43452	. 77 77 77 77 77
	36	25	65265	13101	
	37	66	65015	0	
	40	0	0	77777	
		22	77777	77777	
		CA	GD41		

E33	0	IA	GE	15	
	1	40	GE1	05465	S U P E R S
	2	65	67523	26601	C R I P T Δ
	3	26	54345	55146	S Y M B O L
	4	65	73472	77777	, Δ 77 77 77 77
	5	21	01777	0	Sym.
	6	0	0	0	
	7	0	0	0	
	10	01	21013	45001	Δ Δ Δ I N Δ
	11	31	34723	02701	F I X E D Δ
	12	52	51345	06601	P O I N T Δ
	13	01	01010	10101	Δ Δ Δ Δ Δ Δ
	14	01	01010	13053	Δ Δ Δ Δ E Q
	15	67	24663	45150	U A T I O N
	16	22	77777	77777	. 77 77 77 77 77
E34	17	40	GE17	10	
	18	47	51543	00166	M O R E Δ T
	19	33	24500	10701	H A N Δ 4 Δ
	20	65	67523	05465	S U P E R S
	21	26	54345	26601	C R I P T Δ
	22	65	73472	55146	S Y M B O L
	23	65	01345	00165	S Δ I N Δ S
	24	30	53673	05026	E Q U E N C
	25	30	22777	77777	E . 77 77 77 77
	26	CA	GE27		

		IA	GF		
E35	0	40	GF1	11	
	1	52	51710	15152	P O W Δ O P
	2	30	54246	63451	E R A T I O
	3	50	01657	34725	N Δ S Y M B
	4	51	46012	44751	O L Δ A M O
	5	50	32016	56725	N G Δ S U B
	6	65	26543	45266	S C R I P T
	7	65	01513	10177	S Δ O F Δ 77
	10	0	0	0	Sym.
	11	22	77777	77777	. 77 77 77 77 77
E36	12	40	GF13	10	Δ Δ Δ Δ Δ Δ
	13	52	51710	15152	P O W Δ O P
	14	30	54246	63451	E R A T I O
	15	50	01657	34725	N Δ S Y M B
	16	51	46013	45001	O L Δ I N Δ
	17	31	34723	02701	F I X E D Δ
	20	52	51345	06601	P O I N T Δ
	21	30	53672	46634	E Q U A T I
	22	51	50227	77777	O N . 77 77 77
		CA	GF23		

E37	0	IA	GH	23	
	1	40	GH1	53054	N U M B E R
	2	50	67472	15152	△ O F △ O P
	3	01	51310	02765	E R A N D S
	4	30	54245	30126	△ (B Y △ C
	5	01	17257	40126	O M M A △ C
	6	51	47472	64301	O U N T) △
	7	51	67506	14634	F O R △ L I
	10	31	51540	47301	B R A R Y △
	11	25	54245	76634	△ R O U T I
	12	01	54516	77777	N E △ 77 77 77
	13	50	30017	0	Sym.
	14	0	0	15051	△ I S △ N O
	15	01	34650	36724	T △ E Q U A
	16	66	01305	10150	L △ T O △ N
	17	46	01665	05401	U M B E R △
	20	67	47253	63027	L I S T E D
	21	46	34656	40166	△ F O R △ T
	22	01	31515	15451	H I S △ R O
	23	33	34650	03022	U T I N E .
E38	24	67	66345	14	
	25	40	GH25	05446	I N T E R L
	26	34	50663	45032	O C K I N G
	27	51	26453	43050	△ P A R E N
	30	01	52245	53065	T H E S E S
	31	66	33306	70124	△ A N D △ A
	32	01	24502	66766	B S O L U T
	33	25	65514	44667	E △ V A L U
	34	30	01702	10101	E △ △ △ △ △
	35	30	01010	25065	△ S I G N S
	36	01	65343	70101	. △ △ (△ △
	37	22	01011	30101	△ △) △ △
	40	42	01014	77777	77 77 77 77 77
		42	77777		
		CA	GH41		

		IA	GI		
E39	0	40	GI1	11	
	1	26	46516	53027	C L O S E D
	2	01	52245	43050	△ P A R E N
	3	66	33306	53465	T H E S I S
	4	01	24525	23024	△ A P P E A
	5	54	65017	13466	R S △ W I T
	6	33	01505	10126	H △ N O △ C
	7	51	54543	06552	O R R E S P
	10	51	50273	45032	O N D I N G
	11	01	51523	05022	△ O P E N .
E40	12	40	GI13	23	
	13	50	67472	53054	N U M B E R
	14	01	51310	16567	△ O F △ S U
	15	25	65265	43452	B S C R I P
	16	66	65015	15001	T S △ O N △
	17	0	0	0	Sym.
	20	01	17257	30126	△ (B Y △ C
	21	51	47472	40126	O M M A △ C
	22	51	67506	64301	O U N T) △
	23	01	01010	10101	△ △ △ △ △ △
	24	34	65015	05166	I S △ N O T
	25	01	30536	72446	△ E Q U A L
	26	01	66510	15067	△ T O △ N U
	27	47	25305	40151	M B E R △ O
	30	25	66243	45030	B T A I N E
	31	27	01315	45147	D △ F R O M
	32	01	27344	73050	△ D I M E N
	33	65	34515	00165	S I O N △ S
	34	30	50663	05026	E N T E N C
	35	30	22777	77777	E . 77 77 77 77
		CA	GI36		

E41	0	IA	GJ		
	1	40	GJ1	11	
	2	51	52305	00152	O P E N Δ P
	3	24	54305	06633	A R E N T H
	4	30	65346	50124	E S I S Δ A
	5	47	51503	20165	M O N G Δ S
	6	67	25652	65434	U B S C R I
	7	52	66650	15131	P T S Δ O F
	10	01	77777	77777	Δ 77 77 77 77 77
	11	0	0	0	Sym.
E42	12	22	77777	77777	. 77 77 77 77 77
	13	40	GJ13	11	
	14	34	46463	03224	I L L E G A
	15	46	01657	34725	L Δ S Y M B
	16	51	46210	17777	O L , Δ 77 77
	17	0	0	0	Sym.
	20	01	21013	15154	Δ , Δ F O R
	21	01	54343	23366	Δ R I G H T
	22	01	51310	13053	Δ O F Δ E Q
	23	67	24663	45150	U A T I O N
E43	24	22	77777	77777	. 77 77 77 77 77
	25	40	GJ25	4	
	26	34	50265	15454	I N C O R R
	27	30	26660	16765	E C T Δ U S
	30	30	01513	10126	E Δ O F Δ C
		51	47472	42277	O M M A . 77
		CA	GJ31		

E44	0	IA	GK		
	1	40	GK1	7	
	2	50	67472	53054	N U M B E R
	3	01	51310	13053	Δ O F Δ E Q
	4	67	24466	50165	U A L S Δ S
	5	34	32506	50150	I G N S Δ N
	6	51	66013	05367	O T Δ E Q U
	7	24	46016	65101	A L Δ T O Δ
	10	51	50302	27777	O N E . 77 77
E45	11	40	GK11	6	
	12	65	51473	00151	S O M E Δ O
	13	52	30500	15224	P E N Δ P A
	14	54	30506	63330	R E N T H E
	15	65	30650	15051	S E S Δ N O
	16	66	01264	65165	T Δ C L O S
		30	27227	77777	E D . 77 77 77
		CA	GK17		

		IA	GL							
E46	0	40	GL1	13						
	1	50	67472	53054	N	U	M	B	E	R
	2	01	51310	15152	Δ	O	F	Δ	O	P
	3	30	50012	42565	E	N	Δ	A	B	S
	4	51	46676	63001	O	L	U	T	E	Δ
	5	70	24466	73001	V	A	L	U	E	Δ
	6	65	34325	06501	S	I	G	N	S	Δ
	7	50	51660	13053	N	O	T	Δ	E	Q
	10	67	24460	16651	U	A	L	Δ	T	O
	11	01	50674	72530	Δ	N	U	M	B	E
	12	54	01513	10126	R	Δ	O	F	Δ	C
	13	46	51653	02722	L	O	S	E	D	.
E47	14	40	GL15	13						
	15	50	67472	53054	N	U	M	B	E	R
	16	01	51310	15152	Δ	O	F	Δ	O	P
	17	30	50015	22454	E	N	Δ	P	A	R
	20	30	50663	33065	E	N	T	H	E	S
	21	30	65210	15150	E	S	,	Δ	O	N
	22	01	46303	16621	Δ	L	E	F	T	,
	23	01	50516	60130	Δ	N	O	T	Δ	E
	24	53	67244	60166	Q	U	A	L	Δ	T
	25	51	50674	72530	O	N	U	M	B	E
	26	54	01513	10126	R	Δ	O	F	Δ	C
	27	46	51653	02722	L	O	S	E	D	.
		CA	GL30							

		IA	GM		
E48	0	40	GM1	10	
	1	65	67523	05431	S U P E R F
	2	46	67516	76501	L U O U S Δ
	3	24	54326	74730	A R G U M E
	4	50	66650	15150	N T S Δ O N
	5	01	31675	02666	Δ F U N C T
	6	34	51500	17777	I O N Δ 77 77
	7	0	0	0	Sym.
	10	22	77777	77777	. 77 77 77 77 77
E49	11	40	GM12	17	
	12	71	34663	33450	W I T H I N
	13	01	24543	26747	Δ A R G U M
	14	30	50666	50151	E N T S Δ O
	15	31	01475	15430	F Δ M O R E
	16	01	66332	45001	Δ T H A N Δ
	17	12	01463	42554	7 Δ L I B R
	20	24	54730	15451	A R Y Δ R O
	21	67	66345	03065	U T I N E S
	22	22	24543	26747	. A R G U M
	23	30	50666	50151	E N T S Δ O
	24	31	01777	77777	F Δ 77 77 77 77
	25	0	0	0	Sym.
	26	01	50516	60126	Δ N O T Δ C
	27	33	30264	53027	H E C K E D
	30	22	77777	77777	. 77 77 77 77 77
		CA	GM31		

"Args, Operands, Ss. Not Checked"

		IA	GP						
After	0	40	GP1	7	Δ	Δ	Δ	Δ	Δ
E10, E48,	1	01	01010	10101	Δ	Δ	Δ	Δ	Δ
E13, E15,	2	01	01010	10101	Δ	Δ	Δ	Δ	Δ
	3	01	01010	10101	Δ	Δ	Δ	Δ	Δ
	4	01	01245	43267	Δ	Δ	A	R	G U
	5	47	30506	66501	M	E	N	T	S Δ
	6	50	51660	12633	N	O	T	Δ	C H
	7	30	26453	02722	E	C	K	E	D .
After	10	40	GP11	6					
E2, E5	11	01	01656	72565	Δ	Δ	S	U	B S
	12	26	54345	26665	C	R	I	P	T S
	13	01	50516	60101	Δ	N	O	T	Δ Δ
	14	01	01010	10101	Δ	Δ	Δ	Δ	Δ Δ
	15	26	33302	64530	C	H	E	C	K E
	16	27	22777	77777	D	.	77	77	77 77
After	17	40	GP20	5					
E20,E8,	20	01	01656	72565	Δ	Δ	S	U	B S
E16	21	26	54345	26665	C	R	I	P	T S
	22	01	50516	60126	Δ	N	O	T	Δ C
	23	33	30264	53027	H	E	C	K	E D
After	24	22	77777	77777	.	77	77	77	77 77
E9	25	40	GP4	4					
E51	26	40	GP27	6					
	27	01	01010	10101	Δ	Δ	Δ	Δ	Δ Δ
	30	01	01010	10151	Δ	Δ	Δ	Δ	Δ 0
	31	52	30542	45027	P	E	R	A	N D
	32	65	01505	16601	S	Δ	N	O	T Δ
	33	26	33302	64430	C	H	E	C	K E
	34	27	22777	77777	D	.	77	77	77 77
		CA	GP35						

E50	0	IA	GQ	20	
	1	40	GQ1	14724	T O O Δ M A
	2	66	51510	76747	N Y Δ D U M
	3	50	73012	45432	M Y Δ A R G
	4	47	73012	06665	U M E N T S
	5	67	47305	13167	Δ O N Δ F U
	6	01	51500	45150	N C T I O N
	7	50	26663	77777	Δ 77 77 77 77 77
	10	01	77777	0	Sym.
	11	0	0	10101	. Δ Δ Δ Δ Δ
	12	22	01010	10154	Δ Δ Δ Δ Δ R
	13	01	01010	15131	E S T Δ O F
	14	30	65660	46501	Δ T H I S Δ
	15	01	66333	63050	S E N T E N
	16	65	30506	05166	C E Δ N O T
	17	26	30015	02645	Δ C H E C K
	20	01	26333	77777	E D . 77 77 77
E51	21	30	27227	10	
	22	40	GQ22	72751	P S E U D O
	23	52	65306	05424	Δ O P E R A
	24	01	51523	00165	T I O N Δ S
	25	66	34515	14621	Y M B O L ,
	26	73	47255	77777	Δ 77 77 77 77 77
	27	01	77777	0	Sym.
	30	0	0	15001	Δ , Δ O N Δ
	31	01	21015	36622	R I G H T .
		54	34323		
		CA	GQ32		

E52	0	IA	FQ	14	
	1	40	FQ1	15454	I N C O R R
	2	34	50265	16573	E C T Δ S Y
	3	30	26660	60165	M B O L Δ S
	4	47	25514	05026	E Q U E N C
	5	30	53673	10101	E . Δ Δ Δ Δ
	6	0	0	0	1st Sym.
	7	0	0	0	
	10	0	0	0	
	11	0	0	0	
	12	01	01777	77777	Δ Δ 77 77 77 77
	13	0	0	0	2nd Sym.
	14	0	0	0	
		0	0	0	
		CA	FQ15		

E53	0	IA	GV	14	
	1	40	GV1	53027	C L O S E D
	2	26	46516	55146	△ A B S O L
	3	01	24256	17024	U T E △ V A
	4	67	66300	12452	L U E △ A P
	5	46	67300	46501	P E A R S △
	6	52	30245	30150	W I T H △ N
	7	71	34663	10101	O △ △ △ △ △
	10	51	01010	10101	△ △ △ △ △ △
	11	01	01010	12651	△ △ △ △ C O
	12	01	01010	55251	R R E S P O
	13	54	54306	03201	N D I N G △
	14	50	27345	02277	O P E N . 77
E54	15	51	52305	00152	△ △ △ △ △ △
	16	40	GV16	13	O P E N △ P
	17	51	52305	06633	A R E N T H
	20	24	54305	52101	E S I S , △
	21	30	65346	63031	O N △ L E F
	22	51	50014	44751	T , △ A M O
	23	66	21012	56725	N G △ S U B
	24	50	32016	45266	S C R I P T
	25	65	26543	10101	S △ O F △ △
	26	65	01513	10101	△ △ 77 77 77 77
	27	01	01777	77777	Sym.
	27	0	0	0	. 77 77 77 77 77
	30	22	77777	77777	
E55	31	40	GV32	15	I N C O R R
	32	34	50265	15454	E C T △ U S
	33	30	26660	16765	E △ O F △ O
	34	30	01513	10151	P E N △ P A
	35	52	30500	15224	R E N T H E
	36	54	30506	63330	S I S △ O N
	37	65	34650	15150	△ L E F T △
	40	01	46303	16601	A M O N G △
	41	24	47515	03201	△ △ A R G U
	42	01	01245	43267	M E N T S △
	43	47	30506	66501	O F △ 77 77 77
	44	51	31017	77777	Sym.
	45	0	0	0	. 77 77 77 77 77
	46	22	77777	77777	
		CA	GV47		

E56	0	IA	GY	11	
	1	40	GY1	15454	I N C O R R
	2	34	50265	16765	E C T Δ U S
	3	30	26660	10126	E Δ O F Δ C
	4	30	01513	40124	O M M A Δ A
	5	51	47472	20124	M O N G Δ A
	6	47	51503	73050	R G U M E N
	7	54	32674	13101	T S Δ O F Δ
	10	66	65015	0	Sym.
	11	0	0	77777	. 77 77 77 77 77
E57	12	22	GY13	14	
	13	40	51543	00166	M O R E Δ T
	14	47	24500	10514	H A N Δ 2 9
	15	33	67502	15152	Δ U N C L O
	16	01	30270	22454	S E D Δ O P
	17	65	50015	33065	E N Δ P A R
	20	30	50663	45027	E N T H E S
	21	30	65012	10101	E S Δ A N D
	22	64	51540	56551	/ O R Δ Δ Δ
	23	01	01242	00170	Δ Δ A B S O
	24	46	67663	00165	L U T E Δ V
	25	24	32506	52227	A L U E Δ S
	26	34	GY27		I G N S . 77
		CA			

E58	0	IA	GZ	7	
	1	40	GZ1	56624	C O N S T A
	2	26	66013	60151	N T Δ I L L
	3	50	01463	03151	E G A L Δ O
	4	30	01253	56624	N Δ L E F T
	5	50	66227	77777	, Δ B E F O
	6	21	CA		R E Δ S T A
	7	54			R T . 77 77 77
		CA			

Pseudo Operation Heading Translator

This routine provides the necessary input and output references required by the sentences of the sub program, and also furnishes to the Pseudo Operation Heading generator and to the Exit sentence the necessary linkage so that control is transferred properly during the Object Program.

The input and output references are set by building the DP List containing the dummy variables and their associated dummy call words. These dummy variables as used by the sentences of the sub program are assigned call words as follows:

76x-- subscripted variable (x = number of subscripts).

63--- single valued fixed or floating variable.

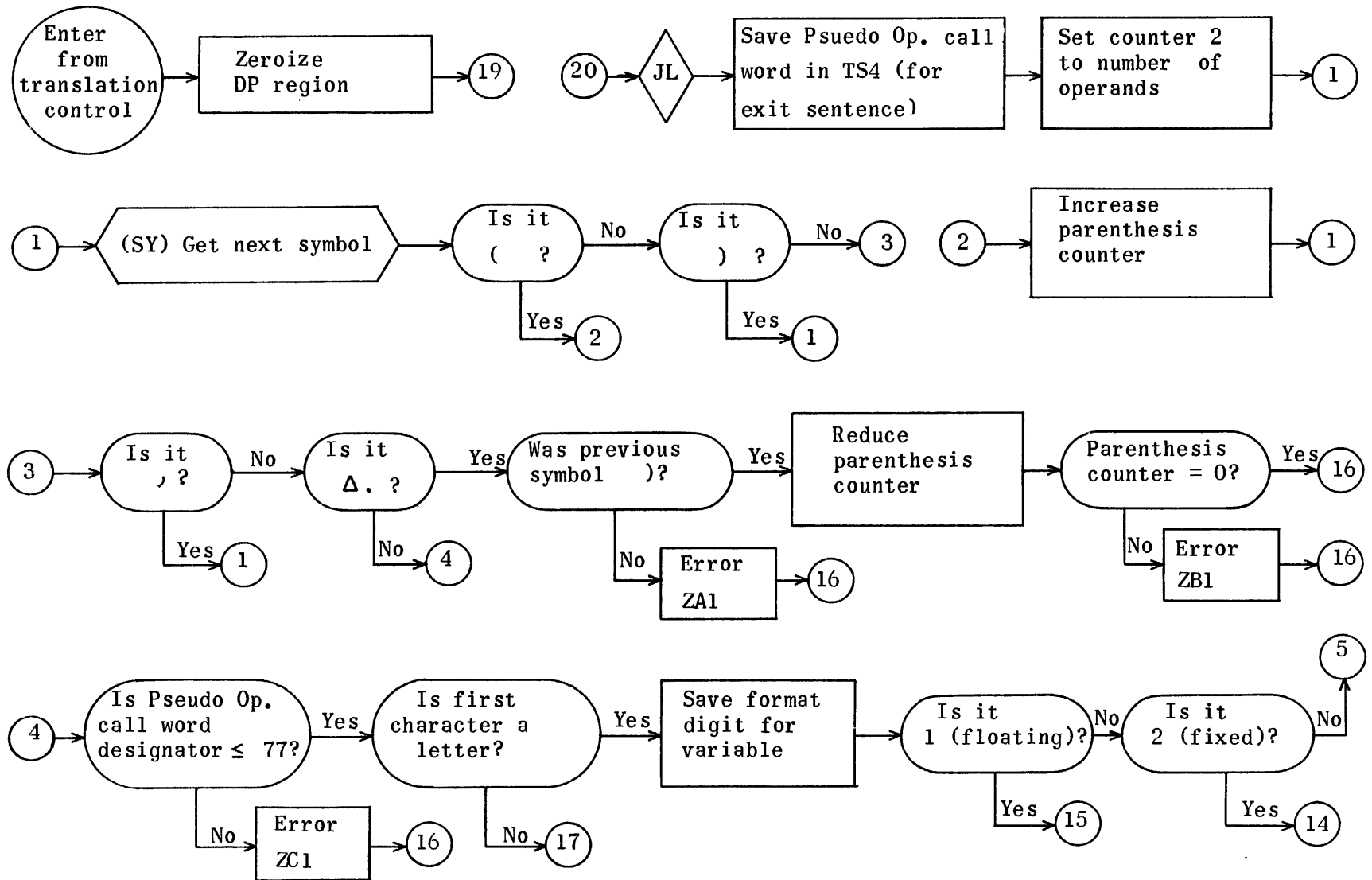
61--- functions.

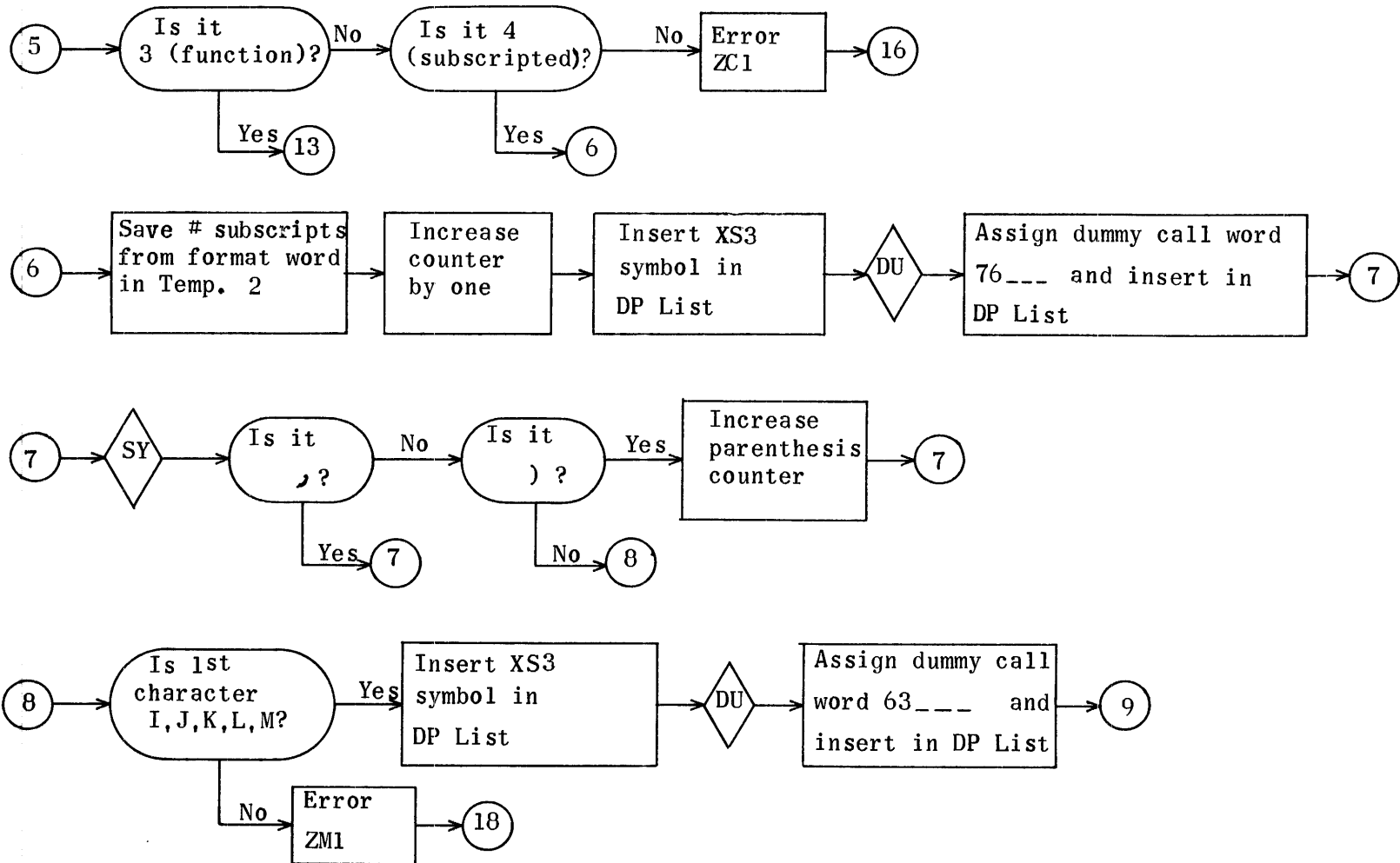
These call words represent the addresses from which input values or the locations of output values are to be obtained during the Object Program. The addresses represented by these call words lie in the Sub Program Input List, a part of the Termination Buffer.

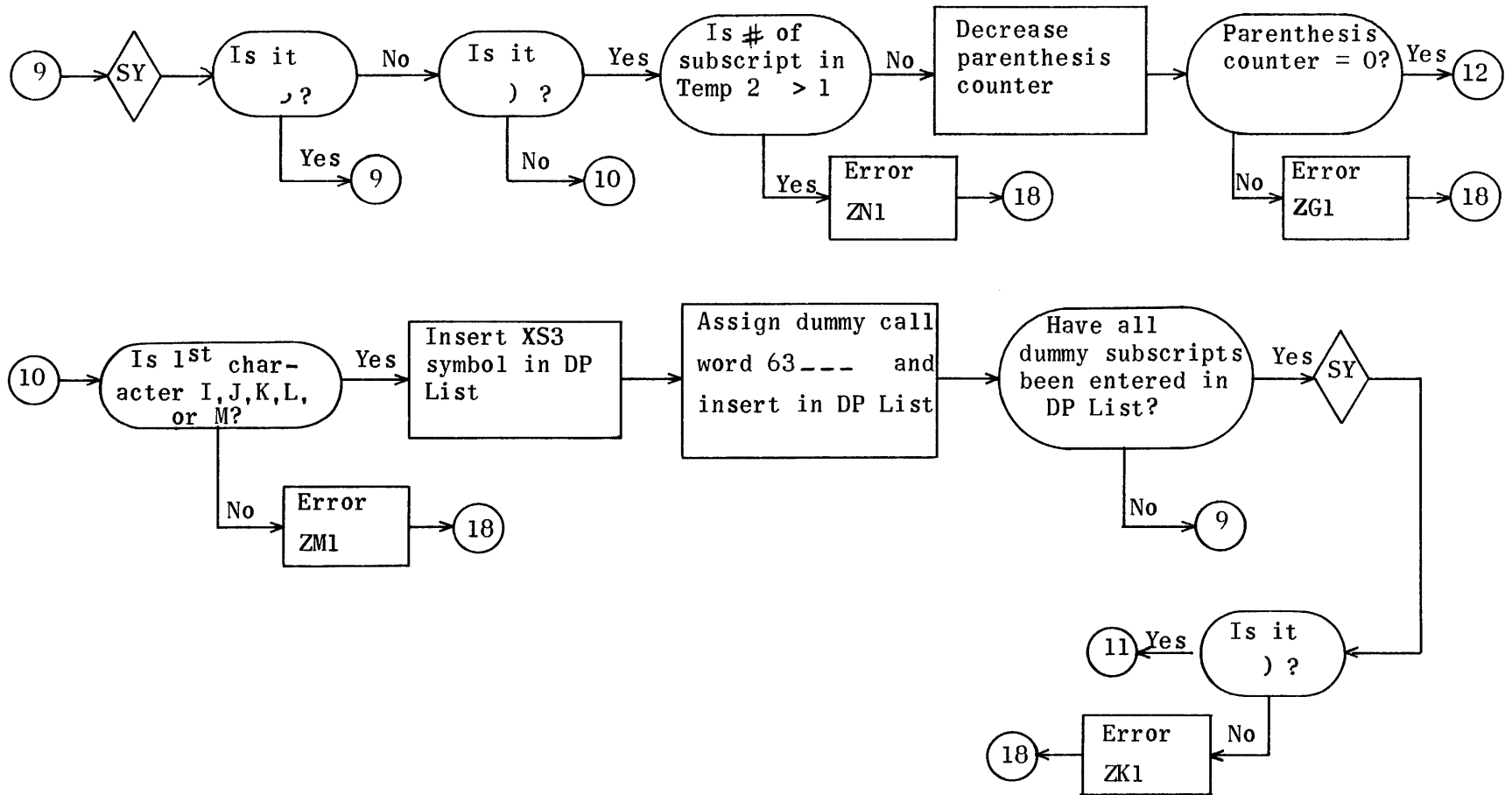
The 22--- type call word of the next sentence of the sub program will be used by the Heading Sentence generator to form the jump instruction which, during running, transfers control into the sub program after being referenced by the COMPUTE sentence.

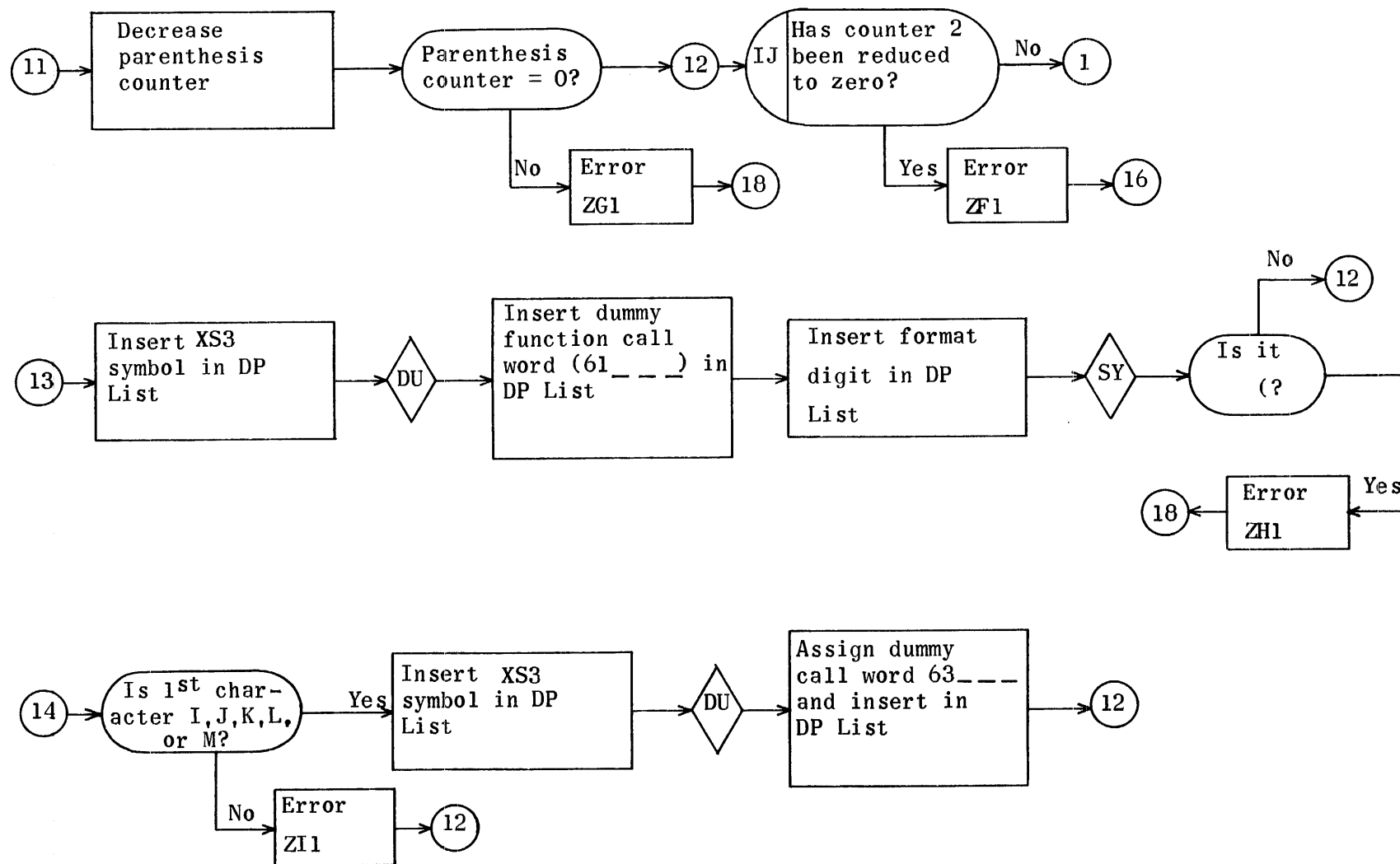
A check is also made by this routine to insure that any previous sub programs have EXIT sentences.

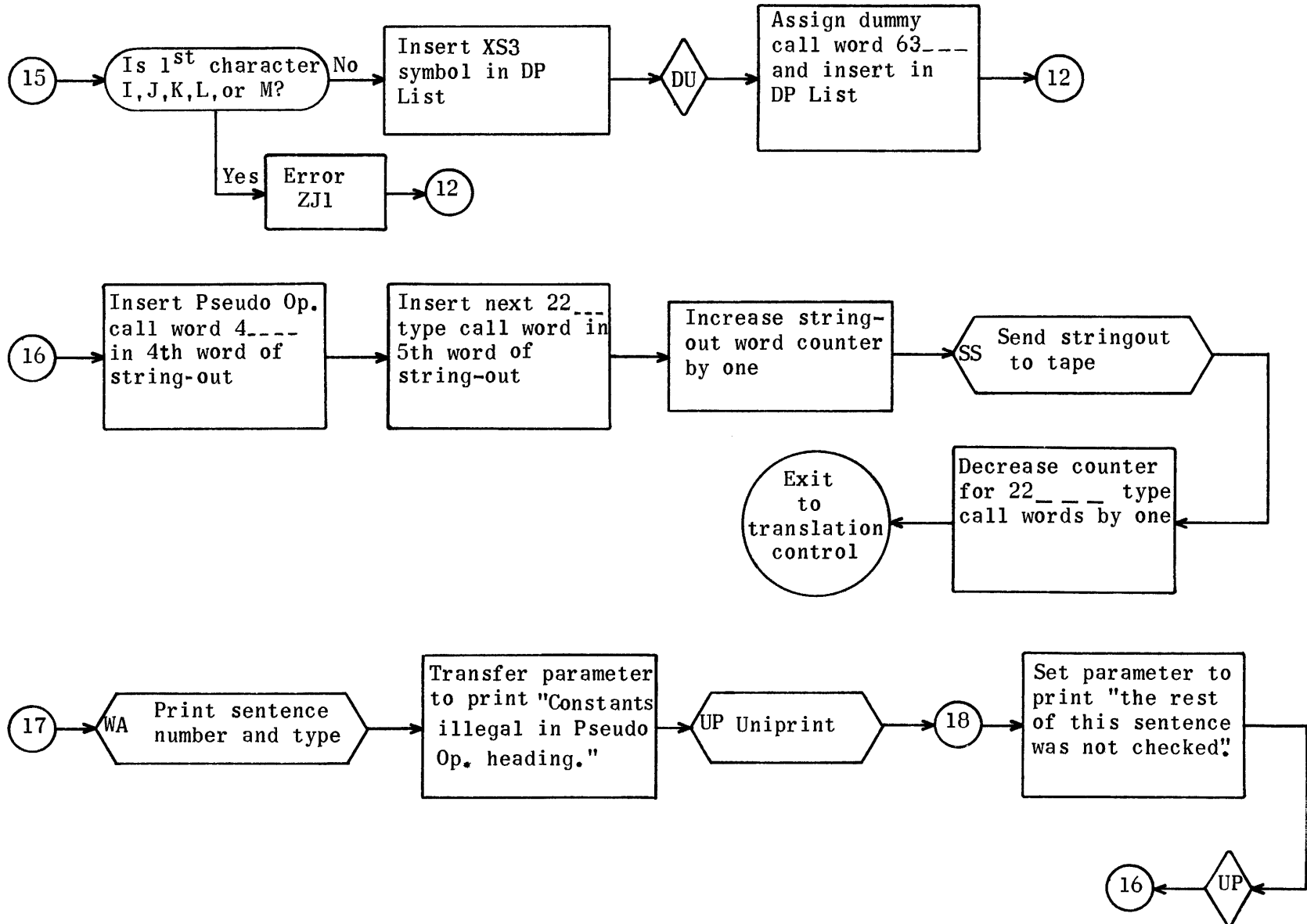
922

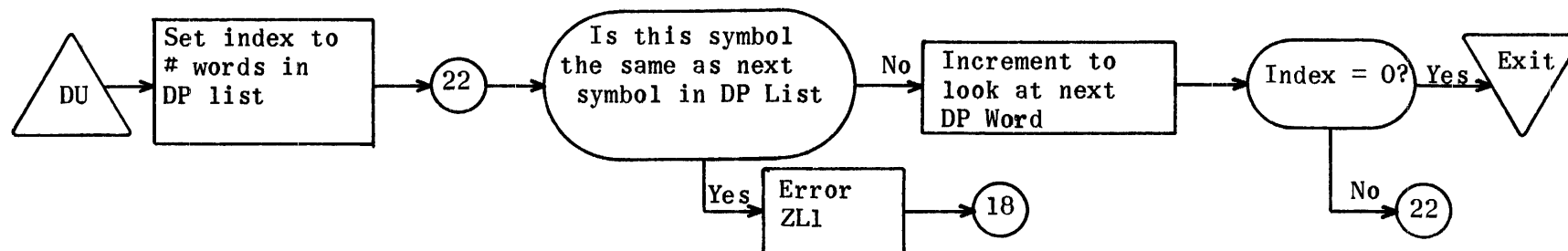
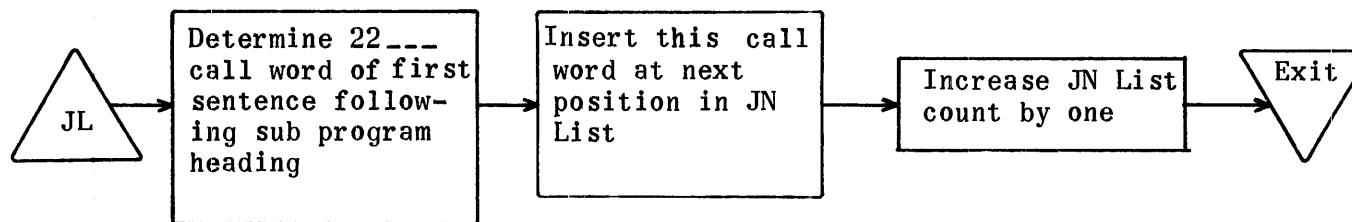
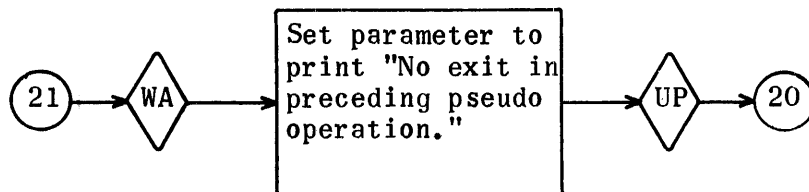
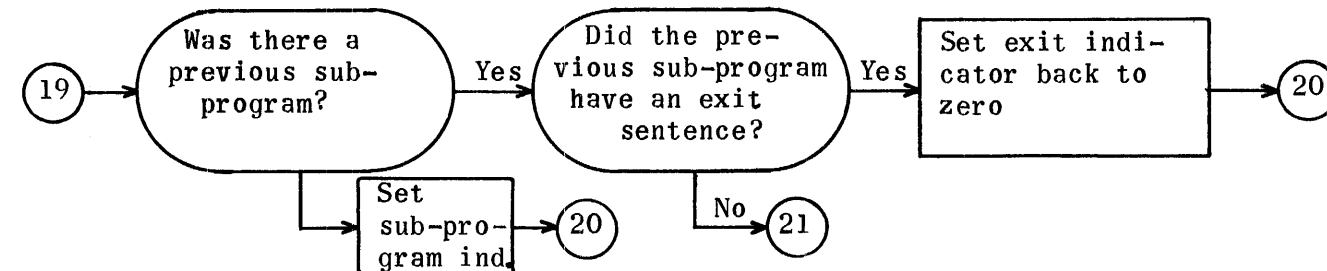












List of Error Print-Outs

Program Entrance	Print-Out "Sentence_____ (SUB)" precedes all of these.
ZA1	No final closing parenthesis.
ZB1 } and ZG1 }	The number of open parentheses is not equal to the number of close parentheses.
ZC1	Machine or internal error.
ZF1	Failed to exit. Probably no space period symbol.
ZH1	Parenthetical expression not allowed for the functional variable in the Pseudo Op Heading statement.
ZI1	Fixed point operands do not begin with legal characters i.e. (I, J, K, L or M) in the Pseudo Op Heading.
ZJ1	Floating point operands in the Pseudo Op Heading begin with illegal characters, i.e. (I, J, K, L or M).
ZK1	No closing parenthesis or too many subscripts.
ZL1	Duplicated symbols are not allowed in Pseudo Op Heading.
ZM1	Erroneous subscript found.
ZN1	At least one subscript missing on subscripted variable.
ZO1	The rest of this sentence was not checked.
XN32	No exit in preceding Pseudo Operation.
XN3	Constants illegal in Pseudo Op Heading.

Regions for Pseudo Operation Heading Translator

RE	PU4400
RE	PV4471
RE	PX4630
RE	PY4667
RE	PZ4714
RE	PW4741
RE	PE4747
RE	DU4761
RE	JL5006
RE	ZA5023
RE	ZB5036
RE	ZC5062
RE	ZF5075
RE	ZG5113
RE	ZH5122
RE	ZI5153
RE	ZJ5204
RE	ZK5234
RE	ZL5252
RE	ZM5273
RE	ZN5306
RE	ZO5327
RE	XN5355

There is one conflict between the string-out subroutine regions and the above regions (each uses region ZO). However, if the above regions are read in first and the string-out subroutine regions second, the Pseudo-Op Heading tape will assemble properly.

Master Control Region for Pseudo-OP Heading Translator.

		IA	PU		
	0	MJ	0	PW	
	1	RP	10132	XN17	} Zeroize
	2	TP	PU44	DP	
(20)	3	RJ	JL	JL1	
					Region DP (90 slots)
	4	TP	TA4	Q	Reference routine for building Pseudo
	5	TV	TA4	TS4	OP 2nd lines.
	6	QT	PU45	PU46	(TA4) = no. of operands
(1)	7	RJ	SY	SY1	Send 40--- CW to TS4
	10	EJ	PU47	PU12	Mask off no. of operands and set CTR2
	11	MJ	0	XN	Test "("
(2)	12	RA	PU50	PU51	Jump to test ")"
	13	MJ	0	PU7	Parentheses Counter + 1 → Paren. Ctr.
(3)	14	EJ	PU52	PU7	Test ", "
	15	EJ	PU53	PU17	Test "Δ. "
	16	MJ	0	PU26	
	17	TP	SZ2	A	} Test previous symbol for ")".
	20	EJ	PU54	PU22	
	21	MJ	0	ZA1	
	22	RS	PU50	PU51	Error! No close parenthesis
	23	ZJ	ZB1	PU	Parenthesis Counter - 1 → Paren. Ctr.
					A ≠ 0 ; Indicates opening and closing
					parens not equal.
	24	MJ	0	PU65	
	25	ZJ	ZC1	PU	
(4)	26	TP	PU55	A	} Test CTRL (assigning CW
	27	TJ	PU63	PU65	
	30	MJ	0	ZC1	
	31	TP	TA5	Q	sequence numbers) ≤ (77) ₈
	32	QT	PU56	PU57	Error!
	33	RA	PU31	PU70	Mask and save format digit
					1, 2, 3, or 4
					Modifies address for picking up next
					operand.
	34	TP	PU57	A	Format digit 1, 2, 3 or 4 → A
	35	EJ	PU51	PZ1	Test for '1'
	36	EJ	PU60	PY1	Test for '2'
	37	EJ	PU61	PX1	Test for '3' (Function)
	40	EJ	PU62	PV1	Test for '4' (subscripted variable)
(12)	41	MJ	0	ZC1	Error! Format digit not 1, 2, 3 or 4.
	42	IJ	PU46	PU7	Initial no. operands -1 → PU46
	43	MJ	0	ZF1	Error! Failed to exit.
	44	0	0	0	
	45	0	0	77	Mask
	46	0	0	0	
	47	17	77777	77777	(
	50	0	0	0	
	51	0	0	1	
	52	21	77777	77777	Δ.
	53	01	22777	77777	
	54	43	77777	77777	)

Master Control Region for Pseudo-OP Heading Translator (continued)

55	0	0	0	
56	0	0	7	Mask
57	0	0	0	
60	0	0	2	
61	0	0	3	
62	0	0	4	
63	0	0	100	
64	77	77777	77776	-1
65	TP	SY7	Q	} Test first character for letter.
66	QJ	PU31	XN3	
67	MJ	0	PU25	
70	0	1	0	
	CA	PU71		

Sub Region for Subscripted Variable

⑥	0	IA	PV	PU42	Exit
	1	MJ	0	A	Mask 'op' portion of format word and save in Temp. 2 = PV112
	2	TP	PU31	PV110	
	3	RS	A	PV4	
	4	TU	A	Q	
	5	TP	[0]	PV112	
	6	QT	PV111	PV113	CTR + 1 → CTR) (set up proper address) Insert XS-3 symbol in DP List Reset PV10) (Test duplication of symbol) Previously saved 'op' → A
	7	RA	PU64	PU64	
	10	RA	PV10	[DP]	
	11	TP	SY2	PV10	
	12	TP	PV114	DU1	
	13	RJ	DU	A	Shift op to v of A _R AR + 76 - 00 → A _R 76*N*00 + CTR1 → Temp. 5 = PV116 Previously saved 'op' → A Shift 'op' to A 0-5 Jump out for correction Setting up index which determines when to begin exit
	14	TP	PV112	A	
	15	LT	14	A	
	16	RA	A	PV115	
	17	AT	PU55	PV116	
	20	TP	PV112	A	Insert '76' N-- CW in DP
	21	LT	6	PV117	
	22	MJ	0	PV130	
	23	AT	PU55	PU55	
	24	TP	PV120	PV121	
	25	RA	PV121	PV117	Test Test ', ' Jump to correction ① Parenthesis counter +1 → Paren. Ctr
	26	RA	PU64	PV113	
	27	RA	PV27	PU64	
	30	TP	PV116	[DP]	
	31	TP	PV122	PV27	
⑦	32	RJ	SY	SY1	Insert XS-3 symbol in DP list
	33	EJ	PV123	PV31	
	34	EJ	PV124	PV35	
	35	MJ	0	PV102	
	36	RA	PV125	PV113	
	37	MJ	0	PV31	'63000' → A '63000' + CTR1 → A Save CW in Temp. 5 = PV116 CTR1 + 1 → CTR1 CTR + 1 → CTR
	40	RA	PU64	PV113	
	41	RA	PV41	PU64	
	42	TP	SY2	[DP]	
	43	TP	PV114	PV41	
⑨	44	RJ	DU	DU1	Send CW to DP List
	45	TP	PV126	A	
	46	RA	A	PU55	
	47	TP	A	PV116	
	50	RA	PU55	PV113	
	51	RA	PU64	PV113	Test ', ' Jump to correction ② Jump to correction ③
	52	RA	PV52	PU64	
	53	TP	PV116	[DP]	
	54	TP	PV122	PV52	
	55	RJ	SY	SY1	
	56	EJ	PV123	PV54	
	57	EJ	PV127	PV104	
	58	MJ	0	PV133	
	60	RA	PV61	PU64	

Sub Region for Subscripted Variable (continued)

	61	TP	SY2	[DP]	
	62	TP	PV114	PV61	Insert XS-3 symbol in DP list
	63	RJ	DU	DU1	
	64	TP	PV126	A	'63000' → A
	65	AT	PU55	PV116	'63000' + CTR1 → A
	66	RA	PU55	PV113	CTR1+1 → CTR1
	67	RA	PU64	PV113	CTR+1 → CTR
	70	RA	PV71	PU64	Send CW to DP List
	71	TP	PV116	[DP]	
	72	TP	PV122	PV71	
	73	LJ	PV121	PV54	
	74	RJ	SY	SY1	
	75	EJ	PV127	PV77	Test ')'
	76	MJ	0	ZK1	Error. No closing parenthesis
(11)	77	RS	PV125	PV113	Parenthesis counter -1 → Paren. Ctr.
	100	ZJ	ZG1	PV	Test for same no. of open and close parens.
(8)	101	0	0	0	
	102	TP	SY10	Q	① Test for subscript
	103	QJ	PV37	ZM1	
	104	TP	PV107	A	② If op > 01 then wrong Exit was used and at least one subscript will be missing from DP List
	105	TJ	PV112	ZN1	
	106	MJ	0	PV77	
	107	01	0	0	
	110	0	1	0	
	111	77	0	0	Mask
	112	0	0	0	Temp. 2
	113	0	0	1	
	114	TP	SY2	DP	Presetter for XS-3 → DP
	115	0	0	76000	CW
	116	0	0	0	Temp. 5
	117	0	0	0	Temp. 3
	120	77	77777	77775	-2
	121	0	0	0	Index
	122	TP	PV116	DP	Presetter for CW → DP
	123	21	77777	77777	,
	124	17	77777	77777	(
	125	0	0	0	Parenthesis Counter
	126	0	0	63000	CW
	127	43	77777	77777	)
(10)	130	TP	PV117	A	Setting up an index for the no. of passes thru last loop
	131	RA	A	PV113	
	132	MJ	0	PV22	
	133	TP	SY10	Q	
	134	QJ	PV135	ZM1	③ Test for subscript I, J, K, L or M
	135	RA	PU64	PV113	
	136	MJ	0	PV60	
		CA	PV137		

Sub Region for Filing Functions in DP

	IA	PX	PU42	
13	0	MJ	0	PU42
	1	RA	PU64	PX27
	2	RA	PX3	PU64
	3	TP	SY2	[DP]
	4	TP	PX32	PX3
	5	RJ	DU	DU1
	6	RA	PU64	PX27
	7	TP	PX36	A
	10	AT	PU55	PX35
	11	RA	PX12	PU64
	12	TP	PX35	[DP]
	13	TP	PX33	PX12
	14	RA	PU55	PU60
	15	TP	PU31	A
	16	RS	A	PX30
	17	TU	A	PX22
	20	RA	PU64	PX27
	21	RA	PX22	PU64
	22	TP	[30000]	[DP]
	23	TP	PX34	PX22
	24	RJ	SY	SY1
	25	EJ	PX31	ZH1
	26	MJ	0	PX
	27	0	0	1
	30	0	1	0
	31	17	77777	77777
	32	TP	SY2	DP
	33	TP	PX35	DP
	34	TP	30000	DP
	35	0	0	0
	36	0	0	61000
		CA	PX37	

				Increase CTR by 1
				Send XS-3 symbol DP
				Preset PX3
				Increase CTR by 1
				C W → A
				C W to Temp. ⑥ after being sequenced
				Temp. ⑥ → DP _n
				Preset PX12
				CTR1+2 → CTR1 →
				Set up address to obtain
				correct format digit
				CTR+1 → CTR
				Send format digit(1, 2, 3 or 4) to DP
				List
				Preset PX22
				Test '('
				(
				Preset PX3
				Preset PX12
				Preset PX22
				Temp. ⑥

Sub Region for FIXED POINT OPERAND

(14)	0	IA	PY		
	1	MJ	0	PU42	
	2	TP	SY10	Q	} Testing
	3	QJ	PY3	ZI1	
	4	RA	PU64	PY22	for I, J, K, L or M
	5	RA	PY5	PU64	CTR+1 → CTR
	6	TP	SY2	[DP]	
	7	TP	PY23	PY5	Insert XS-3 symbol in DP
	10	RJ	DU	DU1	Preset PY5
	11	TP	PY21	A	CW → A
	12	AT	PU55	PY20	Temp. (7) CW plus sequence no.
	13	RA	PU55	PY22	CTR1+1 → CTR1
	14	RA	PU64	PY22	CTR+1 → CTR
	15	RA	PY15	PU64	
	16	TP	PY20	[DP]	Insert 63---type call word in DP List
	17	TP	PY24	PY15	Preset PY15
	20	MJ	0	PY	
	21	0	0	0	Temp. (7)
	22	0	0	63000	CW for FIXED POINT CONSTANT
	23	0	0	1	
	24	TP	SY2	DP	Presetter PY5
		TP	PY20	DP	Presetter PY15
		CA	PY25		

Sub Region for FLOATING POINT OPERAND

⑮	0	IA	PZ		
	1	MJ	0	PU42	
	2	TP	SY10	Q	} Testing for I, J, K, L or M CTR+1 → CTR
	3	QJ	ZJ1	PZ3	
	4	RA	PU64	PZ22	
	5	RA	PZ5	PU64	
	6	TP	SY2	[DP]	Insert XS-3 symbol
	7	TP	PZ23	PZ5	Preset PZ5
	8	RJ	DU	DU1	
	9	TP	PZ21	A	
	10	AT	PU55	PZ20	Temp. ⑧ - CW plus sequence no.
	11	RA	PU55	PZ22	CTR1+1 → CTR1
	12	RA	PU64	PZ22	CTR+1 → CTR
	13	RA	PZ15	PU64	
	14	TP	PZ20	[DP]	Insert 63--- type call word in DP List
	15	TP	PZ24	PZ15	Preset PZ15
	16	MJ	0	PZ	
	17	0	0	0	Temp. ⑧
	18	0	0	63000	CW for FLOATING POINT
	19	0	0	1	
	20	TP	SY2	DP	
	21	TP	PZ20	DP	
	22	CA	PZ25		

	0	IA	PW		
	1	TP	PW4	PU46	} CTR2 CTR1 CTR
	2	TP	PW4	PU55	
	3	TP	PW5	PU64	
	4	MJ	0	PE1	
	5	0	0	0	
	6	77	77777	77776	
	7	CA	PW6		

①⑥

	IA	PE	
0	MJ	0	CT
1	TP	TA4	WL3
2	RA	PE10	VB4
3	TP	A	WL4
4	RA	WL	PE11
5	RJ	SS	SS1
6	RS	VB4	PE11
7	MJ	0	PE
10	0	0	22000
11	0	0	1
	CA	PE12	

Pseudo Op CW → WL3
 22000 + VB4 = CW of 1st sentence
 after heading
 CW → WL4
 Word counter + 1 → Word ctr.
 Send string out to tape

①⑧

	IA	Z0	
0	MJ	0	PU
1	MJ	0	Z02
2	TP	Z025	UP3
3	RJ	UP2	UP
4	MJ	0	Z0
5	01	01010	10101
6	01	01010	10101
7	01	01010	10101
10	01	01010	10101
11	01	01010	10101
12	01	01010	10101
13	01	01010	10101
14	01	01010	10101
15	01	01010	10101
16	66	33300	15430
17	65	66015	13101
20	66	33346	50165
21	30	50663	05026
22	30	01712	46501
23	50	51660	12633
24	30	26453	02722
25	40	Z05	20
	CA	Z026	

Causes carriage return
 before print out

T	H	E	△	R	E
S	T	△	O	F	△
T	H	I	S	△	S
E	N	T	E	N	C
E	△	W	A	S	△
N	O	T	△	C	H
E	C	K	E	D	.

Test Duplication of Symbol

(22)

	IA	DU		
0	MJ	0	[30000]	
1	RA	DU20	PU64	Index + Ctr. = CTR. -1
2	TP	[DP]	A	
3	EJ	SY2	DU5	
4	MJ	0	DU15	
5	TP	DU24	A	0 → A
6	TJ	DU20	DU12	Index > 0 indicates error
7	TP	DU23	DU2	Exit routine
10	TP	DU22	DU20	
11	MJ	0	DU	
12	TP	DU23	DU2	Error routine
13	TP	DU22	DU20	
14	MJ	0	ZL1	
15	RA	DU2	DU21	Modify DP address
16	IJ	DU20	DU2	
17	MJ	0	DU7	Jump to Exit routine
20	77	77777	77776	Index
21	0	1	0	
22	77	77777	77776	Presetter for Index
23	TP	DP	A	
24	0	0	0	
	CA	DU25		

Routine to Make Up List JN of 2nd Lines of Pseudo Op's

	IA	JL		
0	MJ	0	30000	Exit
1	TP	JL12	Q	Mask to Q
2	QT	JN	A	JN = 0 20000 0 initially
3	LQ	A	25	Shift to V position number of pseudo ops.
4	AT	JL14	A	Set-up to get proper loading position for next call word in list.
5	TV	A	JL7	
6	SP	VB4	17	Securing and loading call word in list.
7	AT	JL13	[30000]	
10	RA	JN	PU70	Increasing list length count
11	MJ	0	JL	Exit
12	0	07777	0	Mask
13	0	22000	0	Base call word of pseudo ops
14	0	0	JN1	Base address of 1st call word in list.
	CA	JL15		

Error Print-Outs

	IA	ZA		
0	MJ	0	PU	
1	RJ	WA	WA1	
2	TP	ZA12	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZA	
5	50	51013	13450	
6	24	46012	64651	
7	65	34503	20152	
10	24	54305	06633	
11	30	65346	52277	
12	40	ZA5	5	
	CA	ZA13		

Prints: SENTENCE ____ (SUB)
Gives print-out indicated
by parameter.

N O Δ F I N
A L Δ C L O
S I N G Δ P
A R E N T H
E S I S . Δ

	IA	ZB		
0	MJ	0	PU	
1	RJ	WA	WA1	
2	TP	ZB23	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZB	
5	66	33300	15067	
6	47	25305	40151	
7	31	01515	23050	
10	01	52245	43050	
11	66	33306	53065	
12	01	34650	15051	
13	66	01305	36724	
14	46	01665	10166	
15	33	30010	10101	
16	01	01506	74725	
17	30	54015	13101	
20	26	46516	53001	
21	52	24543	05066	
22	33	30653	06522	
23	40	ZB5	16	
	CA	ZB24		

T H E Δ N U
M B E R Δ O
F Δ O P E N
Δ P A R E N
T H E S E S
Δ I S Δ N O
T Δ E Q U A
L Δ T O Δ T
H E Δ Δ Δ
Δ Δ N U M B
E R Δ O F Δ
C L O S E Δ
P A R E N T
H E S E S .

	IA	ZC		
0	MJ	0	PU	
1	RJ	WA	WA1	
2	TP	ZC12	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZC	
5	47	24263	33450	M A C H I N
6	30	01515	40134	E Δ O R Δ I
7	50	66305	45024	N T E R N A
10	46	01305	45451	L Δ E R R O
11	54	22777	77777	R . 77 77 77 77
12	40	ZC5	5	
	CA	ZC13		

	IA	ZF		
0	MJ	0	PU	
1	RJ	WA	WA1	
2	TP	ZF15	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZF	
5	31	24344	63027	F A I L E D
6	01	66510	13072	Δ T O Δ E X
7	34	66220	15254	I T . Δ P R
10	51	25242	54673	O B A B L Y
11	01	50510	16552	Δ N O Δ S P
12	24	26300	15230	A C E Δ P E
13	54	34512	60165	R I O D Δ S
14	73	47255	14622	Y M B O L .
15	40	ZF5	10	
	CA	ZF16		

	IA	ZG	
0	MJ	0	Z01
1	RJ	WA	WA1
2	TP	ZB23	UP3
3	RJ	UP2	UP
4	TP	ZG6	PV102
5	MJ	0	ZG
6	0	0	0
	CA	ZG7	

	IA	ZH	
0	MJ	0	Z01
1	RJ	WA	WA1
2	TP	ZH30	UP3
3	RJ	UP2	UP
4	MJ	0	ZH
5	52	24543	05066
6	33	30663	42624
7	46	01307	25254
10	30	65653	45150
11	65	01505	16601
12	24	46465	17130
13	27	01315	15401
14	66	33300	10101
15	01	01010	10101
16	01	01316	75026
17	66	34515	02446
20	01	70245	43424
21	25	46300	13450
22	01	66333	00152
23	65	30672	75101
24	51	52013	33024
25	27	34503	20165
26	66	24663	04730
27	50	66227	77777
30	40	ZH5	23
	CA	ZH31	

P	A	R	E	N	T
H	E	T	I	C	A
L	Δ	E	X	P	R
E	S	S	I	O	N
S	Δ	N	O	T	Δ
A	L	L	O	W	E
D	Δ	F	O	R	Δ
T	H	E	Δ	Δ	Δ
Δ	Δ	Δ	Δ	Δ	Δ
Δ	Δ	F	U	N	C
T	I	O	N	A	L
Δ	V	A	R	I	A
B	L	E	Δ	I	N
Δ	T	H	E	Δ	P
S	E	U	D	O	Δ
O	P	Δ	H	E	A
D	I	N	G	Δ	S
T	A	T	E	M	E
N	T	.	77	77	77

	IA	ZI		
0	MJ	0	PU42	
1	RJ	WA	WA1	
2	TP	ZI30	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZI	
5	31	34723	02701	F I X E D Δ
6	52	51345	06601	P O I N T Δ
7	51	52305	42450	O P E R A N
10	27	65012	75101	D S Δ D O Δ
11	50	51660	12530	N O T Δ B E
12	32	34500	17134	G I N Δ W I
13	66	33014	63032	T H Δ L E G
14	24	46010	10101	A L Δ Δ Δ Δ
15	01	01010	10101	Δ Δ Δ Δ Δ Δ
16	01	01263	32454	Δ Δ C H A R
17	24	26663	05465	A C T E R S
20	01	34223	02201	Δ I . E . Δ
21	17	34214	42145	(I , J , K
22	21	46015	15401	, L Δ O R Δ
23	47	43013	45001	M) Δ I N Δ
24	66	33300	15265	T H E Δ P S
25	30	67275	10151	E U D O Δ O
26	52	01333	02427	P Δ H E A D
27	34	50322	27777	I N G . 77 77
30	40	ZI5	23	
	CA	ZI31		

	IA	ZJ	
0	MJ	0	PU42
1	RJ	WA	WA1
2	TP	ZJ37	UP3
3	RJ	UP2	UP
4	MJ	0	ZJ
5	31	46512	46634
6	50	32015	25134
7	50	66015	15230
10	54	24502	76501
11	34	50016	63330
12	01	52653	06727
13	51	02515	20133
14	30	24273	45032
15	01	01010	10101
16	01	01253	03234
17	54	01713	46633
20	01	34464	63032
21	24	46012	63324
22	54	24266	63054
23	65	01342	23022
24	01	17342	14421
25	45	21460	15154
26	21	47432	27777
27	40	ZJ5	22
	CA	ZJ30	

F L O A T I
 N G Δ P O I
 N T Δ O P E
 R A N D S Δ
 I N Δ T H E
 Δ P S E U D
 O - O P Δ H
 E A D I N G
 Δ Δ Δ Δ Δ Δ
 Δ Δ B E G I
 N Δ W I T H
 Δ I L L E G
 A L Δ C H A
 R A C T E R
 S Δ I . E .
 Δ (I , J ,
 K , L Δ O R
 , M) . 77 77

	IA	ZK		
0	MJ	0	Z01	
1	RJ	WA	WA1	
2	TP	ZK15	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZK	
5	50	51012	64651	N O Δ C L O
6	65	34503	20152	S I N G Δ P
7	24	54305	06633	A R E N T H
10	30	65346	50151	E S I S Δ O
11	54	01665	15101	R Δ T O O Δ
12	47	24507	30165	M A N Y Δ S
13	67	25652	65434	U B S C R I
14	52	66652	27777	P T S . 77 77
15	40	ZK5	10	
	CA	ZK16		

	IA	ZL		
0	MJ	0	Z01	
1	RJ	WA	WA1	
2	TP	ZL20	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZL	
5	27	67524	63426	D U P L I C
6	24	66302	70165	A T E D Δ S
7	73	47255	14665	Y M B O L S
10	01	24543	00150	Δ A R E Δ N
11	51	66012	44646	O T A L L
12	51	71302	70134	O W E D Δ I
13	50	01526	53067	N Δ P S E U
14	27	51015	15201	D O Δ O P Δ
15	01	01010	10101	Δ Δ Δ Δ Δ Δ
16	01	01333	02427	Δ Δ H E A D
17	34	50322	27777	I N G . 77 77
20	40	ZL5	13	
	CA	ZL21		

	IA	ZM		
0	MJ	0	Z01	
1	RJ	WA	WA1	
2	TP	ZM12	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZM	
5	30	54545	15030	E R R O N E
6	51	67650	16567	O U S Δ S U
7	25	65265	43452	B S C R I P
10	66	01315	16750	T Δ F O U N
11	27	22777	77777	D . 77 77 77 77
12	40	ZM5	5	
	CA	ZM13		

	IA	ZN		
0	MJ	0	Z01	
1	RJ	WA	WA1	
2	TP	ZN20	UP3	
3	RJ	UP2	UP	
4	MJ	0	ZN	
5	24	66014	63024	A T Δ L E A
6	65	66015	15030	S T Δ O N E
7	01	65672	56526	Δ S U B S C
10	54	34526	60147	R I P T Δ M
11	34	65653	45032	I S S I N G
12	01	51500	16567	Δ O N Δ S U
13	25	65265	43452	B S C R I P
14	66	30270	10101	T E D Δ Δ Δ
15	01	01010	10101	Δ Δ Δ Δ Δ Δ
16	01	01702	45434	Δ Δ V A R I
17	24	25463	02277	A B L E . 77
20	40	ZN5	13	
	CA	ZN21		

	IA	XN		
0	EJ	PU54	PU7	} Test closing parenthesis
1	MJ	0	PU14	
2	MJ	0	Z01	
3	RJ	WA	WA1	} Print region
4	TP	XN7	UP3	
5	RJ	UP2	UP	
6	MJ	0	XN2	
7	40	XN10	7	
10	26	51506	56624	C O N S T A
11	50	66650	13446	N T S Δ I L
12	46	30322	44601	L E G A L Δ
13	34	50015	26530	I N Δ P S E
14	67	27510	15152	U D O Δ O P
15	01	33302	42734	Δ H E A D I
16	50	32227	77777	N G .
17	TP	TS4	Q	} Test for heading bit
20	QJ	XN21	XN25	
21	TP	VD1	Q	} Test for Exit Bit
22	QJ	XN23	XN32	
23	TP	XN30	VD1	Zeroize VD1
24	MJ	0	PU3	Set heading bit
25	RA	TS4	XN27	
26	MJ	0	PU3	
27	40	0	0	
30	0	0	0	
31	MJ	0	PU3	} Error Print region
32	RJ	WA	WA1	
33	TP	XN45	UP3	
34	RJ	UP2	UP	
35	MJ	0	XN31	
36	50	51013	07234	N O Δ E X I
37	66	01345	00152	T Δ I N Δ P
40	54	30263	02734	R E C E D I
41	50	32015	26530	N G Δ P S E
42	67	27510	15152	U D O Δ O P
43	30	54246	63451	E R A T I O
44	50	22777	77777	N .
45	40	75543	7	
	CA	XN46		

IV. GENERATION PHASE

IV. GENERATION PHASE

1. Generation Setup and Drum Loader

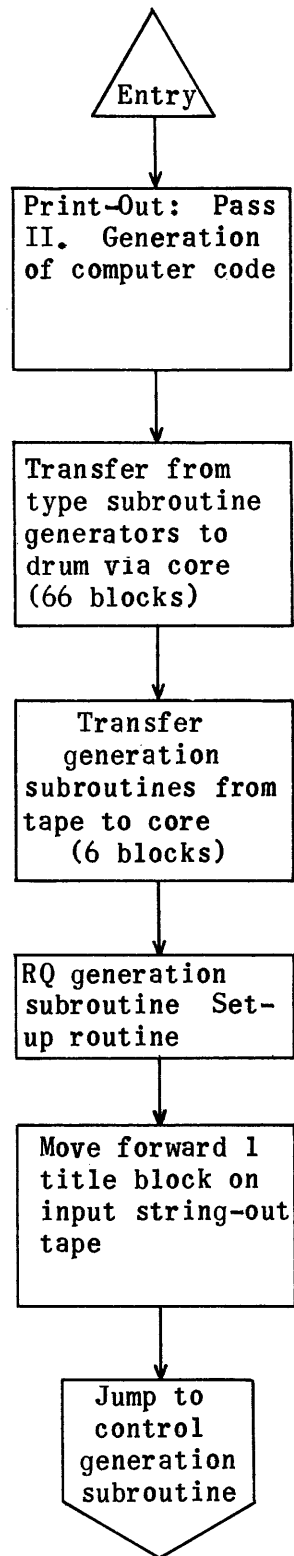
After the initial print-out: Pass II. Generation of Computer Code, this routine transfers from tape to core to drum the two 27-block sections of the Subroutine Generators. Next, it transfers from tape to core the 6 blocks of Generation Subroutines.

RQ routine is then referenced to set up some block and line counters, put the proper parameter for 5 or 7 Uniservos into Op Control routine, and put two title blocks on both tape 5 holding Op File I and the tape that is to hold the generated coding. This tape will be either on Uniservo 4 or 7, depending on whether 5 or 7 Uniservos are used. In addition, on the 22nd line of the title block for the generated coding, the RQ routine puts the contents of 13, which is a count of the number of blocks of the corrected problem on tape 5.

The tape holding the string-out input, either on Uniservo 3 or 6, is now moved forward one block to bypass the title. Then an exit jump is made to the beginning address of CG, the Control Generation subroutine.

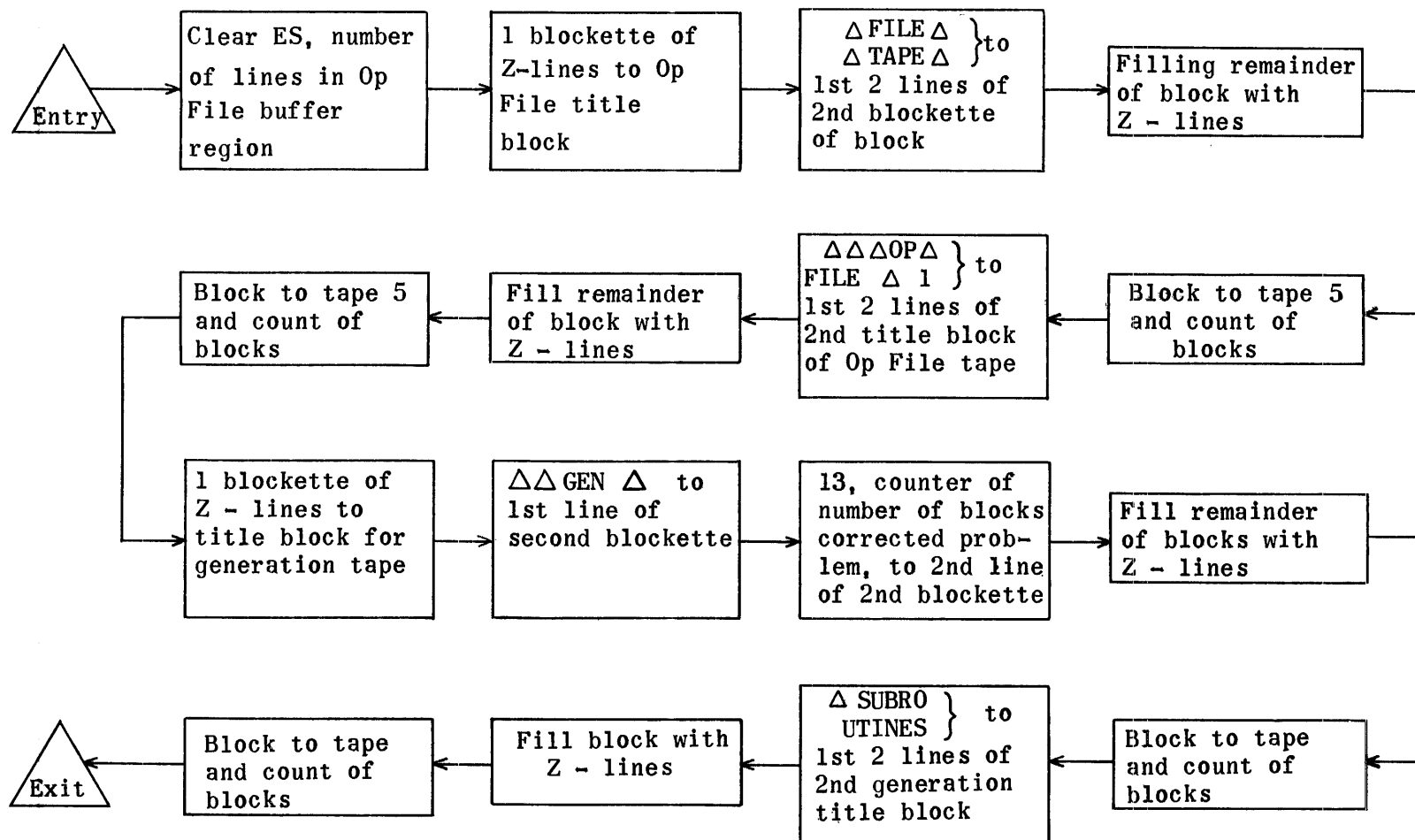
Also included in the same block as the Generation Set-Up is a Drum Loader routine which can be used to help update or correct the UNICODE Master Tape. First a flex or biocatal copy of a revised Generator is loaded into the core. A look at the annotated coding of this Drum Loader will show the numerical PAK setting to the left of a section of coding labeled for the specific Generator. A start at this address will transfer the Generator to its proper place in the drum. In this connection, the separate write-up on the System Tape Package should be consulted to ensure that the other necessary steps are taken to update the Master Tape correctly.

Generation Set-Up Routines - Flow Charts



Generation Set-Up Routines - Flow Charts

RQ Routine To Set Up Op Control Routines and Write Title Blocks For
Op File I and Generated Routines on Tapes



Generation Setup and Drum Loader Regions

RE DA7230
RE RQ7256
RE BF7325
RE BG6250
RE BH6250
RE CJ50212
RE CK56462
RE CD3300
RE CE3300
RE CN600
RE EX7350

Generation Subroutine regions are also needed to assemble this tape.

Generation Setup and Drum Loader

Print-Out:					
Pass II. Generation of Computer Code	0	IA	DA		
	1	RJ	BF	BF1	
	2	TP	DA21	GT3	
	3	RJ	GT2	GT	
To get subroutine generators on drum via core from tape	4	RP	BG30000	DA5	
	5	TP	BR	CJ	
	6	TP	DA22	GT3	
	7	RJ	GT2	GT	
	10	RP	BH30000	DA11	
	11	TP	BR	CK	
	12	TP	DA23	GT3	To get generation subroutines from tape to core
	13	RJ	GT2	GT	
	14	RJ	RQ	RQ1	To operate generation Op-Control set up routine
	15	TP	TN	A	Move forward tape {3} 1 block. This servo holds the string-outs plus 1-block title
	16	AT	DA24	GT3	
	17	RJ	GT2	GT	
	20	MJ	0	DA20	Dummy jump instruction (replaces discarded "12" clearing instruction)
	21	MJ	0	CG	Jump to exit
	22	50	CD1	BR	Parameters for generators in 2 sect.
	23	50	CE1	BR	
	24	50	CN1	BR	
	25	30	103	0	Parameter for generation subroutines
					Parameter for moving forward tape 3 1 block
					Excess storage of a zero formerly used in instruction 17
		CA	DA26		
Routine to Set Up Op Control	0	IA	RQ		
Tape Write	1	MJ	0	30000	Exit
Routine and Write Title	2	TP	GP7	ES	Clears 2 storage locations, es = no. lines in Op File buffer region, np es 5 = no. of blocks of Op Files
	3	TP	GP7	ES5	
Blocks for Op File and Gen. Routines on Tapes	4	RP	10024	RQ5	Puts 20 ₈ lines of Z's into np
	5	TP	GP2	NP	
	6	TP	GP13	NP24	# File # } into np
	7	TP	GP14	NP25	
	10	RP	10142	RQ11	Filling remainder of np with Z's
	11	TP	GP2	NP26	
Op File Setup	12	TP	RC	GT3	Parameter → generalized tape handler
	13	RJ	GT2	GT	Writing block on tape via tape hdlr.
	14	RA	ES5	GP10	Count of blocks
	15	TP	GP	NP	Δ Δ Δ O P Δ } to np
	16	TP	GP1	NP1	
	17	RP	10166	RQ20	F I L E Δ 1 } Filling rest of np with Z's
	20	TP	GP2	NP2	
	21	TP	RC	GT3	Using tape handler
		RJ	GT2	GT	

Gen.Routines
Setup

22	RA	ES5	GP10	Count of blocks
23	TP	RC2	A	} Putting proper parameter in RC4 de- pending on whether TN = 0 or 0 3 0. The latter is for 7 servos.
24	AT	TN	RC4	
25	TP	GP7	ES6	Clearing counter for no. blocks of generated subroutines
26	RP	10024	RQ30	} 20 lines of Z's to GN, buffer region of subroutines used in writing on tape
27	TP	GP2	GN	
30	TP	GP15	GN24	△△Gen△to GN
31	TP	13	GN25	Count of no. blocks of corrected problem on tape 5
32	RP	10142	RQ34	} Filling GN with lines of Z's
33	TP	GP2	GN26	
34	TP	RC4	GT3	} Writing block on tape
35	RJ	GT2	GT	
36	RA	ES6	GP10	Count of blocks
37	TP	GP11	GN	} △SUBRO } to GN UTINES }
40	TP	GP12	GN1	
41	RP	10166	RQ43	} Filling GN with lines of Z's
42	TP	GP2	GN2	
43	TP	RC4	GT3	} Writing block on tape and counting blocks
44	RJ	GT2	GT	
45	RA	ES6	GP10	
46	MJ	0	RQ	Jump to exit
	CA	RQ47		

	IA	BF		
0	MJ	0	30000	
1	TP	BF4	UP3	} Gives print-out:△△△△△ Pass II. Generation of Computer Code
2	RJ	UP2	UP	
3	MJ	0	BF	
4	0	BF5	10	Parameter
5	01	01010	10101	△ ——— △
6	52	24656	50134	P A S S I
7	34	22010	10101	I . △△△△
10	01	32305	03054	△ G E N E R
11	24	66345	15001	A T I O N △
12	51	31012	65147	O F △ C O M
13	52	67663	05401	P U T E R △
14	26	51273	07777	C O D E
	CA	BF15		

Routine used to load Generators from Core to Drum (used only in assembling initially the UNICODE generators; operated by console manipulation). A changed generator is first read into the core. Then a start with PAK set to the numbered address shown at left will transfer the generator to its place in the drum.

	IA	EX		
7350	RP	DR30000	EX44	Start
	TP	KB	CJ	
7352	RP	DM30000	EX44	Jump
	TP	KB	EA	
7354	RP	DK30000	EX44	If
	TP	KB	EB	
7356	RP	DO30000	EX44	Print
	TP	KB	EC	
7360	RP	DY30000	EX44	Compute
	TP	KB	ED	
7362	RP	DU30000	EX44	Vary
	TP	KB	EF	
7364	RP	DQ30000	EX44	Resume
	TP	KB	EY	
7366	RP	DW30000	EX44	Exit
	TP	KB	EH	
7370	RP	DT30000	EX44	Type
	TP	KB	EI	
7372	RP	DN30000	EX44	List
	TP	KB	EJ	
7374	RP	DP30000	EX44	Read
	TP	KB	EK	
7376	RP	DS30000	EX44	Stop
	TP	KB	EL	
7400	RP	DZ30000	EX44	Dimens.
	TP	KB	EM	
7402	RP	DX30000	EX44	Pseudo-Op Heading
	TP	KB	EN	
7404	RP	DV30000	EX44	End of Tape
	TP	KB	EO	
7406	RP	DH30000	EX44	Eq. 1 Listing
	TP	KB	ET	
7410	RP	DI30000	EX44	Eq. 2 Redundancy
	TP	KB	EU	
7412	RP	DJ30000	EX44	Eq. 3 Generator
	TP	KB	EV	
	MS	O	EX2	
	CA	EX45		

2. GENERATION SUBROUTINES

2. Generation Subroutine Regions

	RE	TN20	Temporary to indicate whether 5 or 7 servos (set in string-out)
	RE	GT21	} Tape handler
	RE	TH21	
	RE	UP421	
	RE	UQ443	} Line-number processing routine
	RE	US453	
	RE	UW513	
	RE	EP537	} Machine Error Routine
	RE	BR537	
	RE	BP564	
	RE	BQ632	} Sentence number print-out during an error print-out
	RE	WA653	
	RE	WB677	
	RE	GP717	} Op routine to write output on tape and RG routine used as adjunct to VARY
	RE	RC735	
	RE	ES742	
	RE	RG755	} Op routine to write output on tape and RG routine used as adjunct to VARY
	RE	OP1047	
	RE	XP1126	
	RE	CW1211	Call word routine
	RE	LW1250	Routine to get call word of referenced line number from list
	RE	HI1306	Routine to put call word in referenced line-number list
	RE	KI1336	Illegal line jump check routine
	RE	LS1465	Library list routine
	RE	CG1530	} Control generation routine
	RE	CH1642	
	RE	VX1670	
	RE	VE2044	Excess three to Flex code
	RE	NP2052	Op File buffer
	RE	WL2242	} Input buffer (250 ₈ lines maximum)
	RE	BK2242	
	RE	KB2512	
	RE	GN5360	Region where any generator operates
	RE	CI144	Generated routines output buffer
	RE	CI144	Library List Threshold (keeps max. of library routines referenced at 99)
Size of Generators on Drum	RE	DH612	Eq. 1, Generation
	RE	DI2604	Eq. 2, Redundancy
	RE	DJ2615	Eq. 3, Generator
	RE	DK763	If
	RE	DM33	Jump
	RE	DN461	List
	RE	DO107	Print
	RE	DP430	Read
	RE	DQ174	Resume
	RE	DR30	Start
	RE	DS425	Stop
	RE	DT560	Type
	RE	DU766	Vary

	RE DV502	End of tape
	RE DW34	Exit
	RE DX24	Pseudo-Op Heading
	RE DY551	Compute
	RE DZ53	Dimension
	RE ZZ7230 }	Generation Set-Up Block
	RE DA7230 }	
	RE ZA77000	Region of UNICODE Service Routines
	RE FC40001	Excess Three to Flex-Code List
	RE CB40101	Combination List
	RE DL40102	Dimension List
	RE CL46101	Constant List
	RE VF47101	Vary File List
	RE IZ47246	Referenced Line-number List
	RE JN47722	List of call words of 2nd-line numbers of sub-programs
	RE RW50023	Rewind List of call words of tape numbers referenced
	RE LN50046	Library List of call words of library routines referenced
Initial addresses of gener- ators stored on drum	RE CJ50212	Start
	RE EA50242	Jump
	RE EB50275	If
	RE EC51260	Print
	RE ED51367	Compute
	RE EF52140	Vary
	RE EY53126	Resume
	RE EH53322	Exit
	RE EI53356	Type
	RE EJ54136	List
	RE EK54617	Read
	RE EL55247	Stop
	RE EM55674	Dimension
	RE EN55747	Pseudo-Op Heading
	RE EO55773	End of Tape
	RE ET56475	Equation 1, Generation Sorting
	RE EU57307	Equation 2, Redundancy Check
	RE EV62113	Equation 3, Generator
These routines are part of End of Tape and are refer- enced from Control Generation	RE IG2675	Routine to put Constant List on tape 5
	RE UG2713	Routine to put Dimension List on tape 5
	RE EG2732	Rewind tape routine
	RE BE2750	Gives generation termination print-out
	RE BU2774	Sends excess-three symbol list to tape 5

Unless otherwise stated in the coding, the above regions are sufficient for all the Generation Subroutines included in this section.

Control Generation Routine

As with the Generation Set-Up Routine, the indicator TN is used to determine on which Uniservo the string-out input to generation has been written. It may either be on Uniservo 3 or 6 depending on whether 5 or 7 Uniservos have been used.

The title block having been passed by, the first line of any block on this tape beginning a new sentence string-out contains the number of lines in the string-out. If this number is greater than 170₈, two blocks of input for that sentence are read into the input buffer. The buffer can only preserve a maximum of 250₈ significant words. Any excess lines above 250₈ are later overlaid by the transfer of a generator into the core. Alarms in the Translation or String-Out Phase have already given adequate protection against possible exceeding of this string-out buffer maximum so there is no further check made against this contingency in generation.

The line number of the sentence being analyzed is compared with the numbers in the referenced line-number list IZ by means of routine HI and, if found, its call word added to the list as needed.

The content of BK2, which holds the name of the sentence under analysis, is compared with a list of sentence titles to determine what type of sentence is being analyzed. Depending on which equality is found, a jump is made to a corresponding section of the routine which brings in the proper generator from the drum, makes a referencing return jump to it, and then jumps back to the beginning entry of Control Generation to get the next sentence input. The return jump to the generator causes the generation of the desired sentence coding, followed by the writing of it on tape and the storing of its Op File in a buffer for later writing on tape.

If none of a list of 13 sentence types is recognized, a comparison is made to see if End of Tape instruction has been encountered. If yes, the End of Tape Generator is brought into the core from the drum. Control of the End of Tape generation is retained in the Control Generation from CG77-CG110 where a succession of return jumps does a variety of things explained more thoroughly in the write-up on the End of Tape Generator. The exit from this succession is a jump to ZA10, the address of an overall UNICODE control service routine in the drum.

If the End of Tape sentence is not recognized as suggested in the preceding paragraph, the call word of the sentence is checked to see if it is greater than 40000. If yes, the sentence is recognized as a Pseudo-Op Heading and the generator for this type is brought into the core. Before referencing this generator, the line number of the sentence is put in the proper place in the prelude of the routine which is to be generated. Following this extra remedial step, the generator is referenced similarly as with other generators.

If the call word of the sentence proposed above is not greater than 40000, the sentence is determined by this system of elimination to be the only remaining type not considered - namely, an equation.

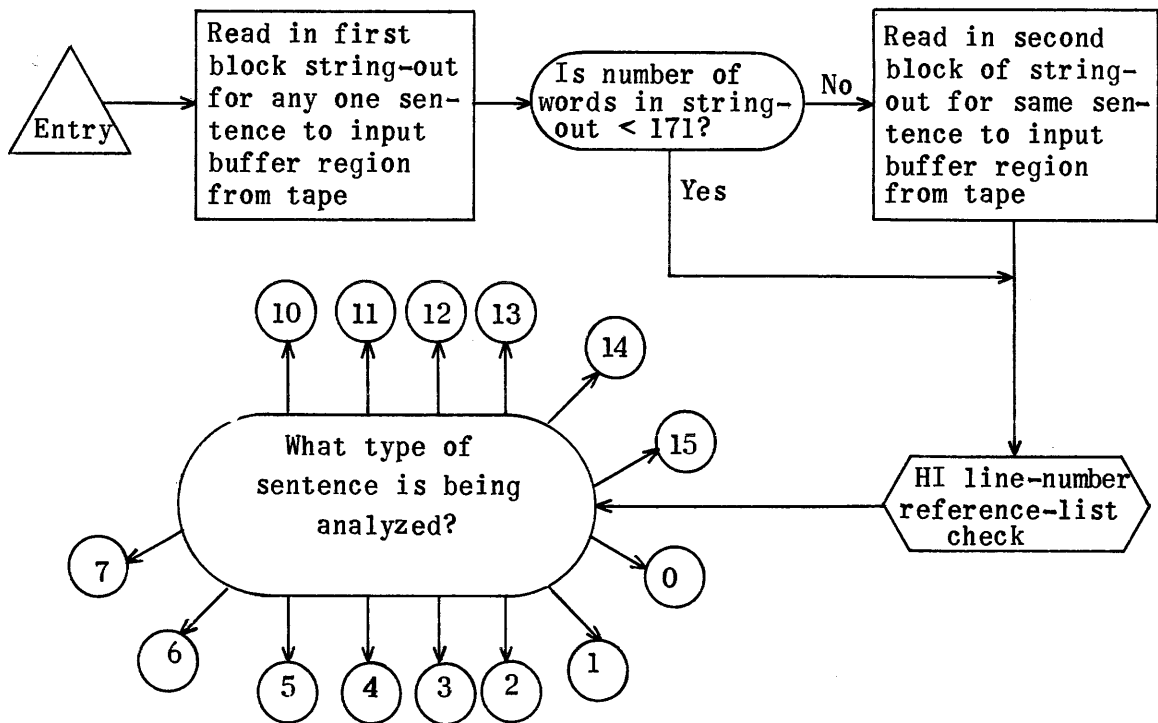
This type of sentence requires special handling which is explained at greater length in another write-up. Briefly an equation sorting routine is transferred from drum to core and then referenced by Control; next, an equation redundancy routine is transferred from drum to core and similarly referenced; finally, the last equation generator is brought into the core and referenced to complete the handling of the sentence. Termination is as usual by getting the next sentence input.

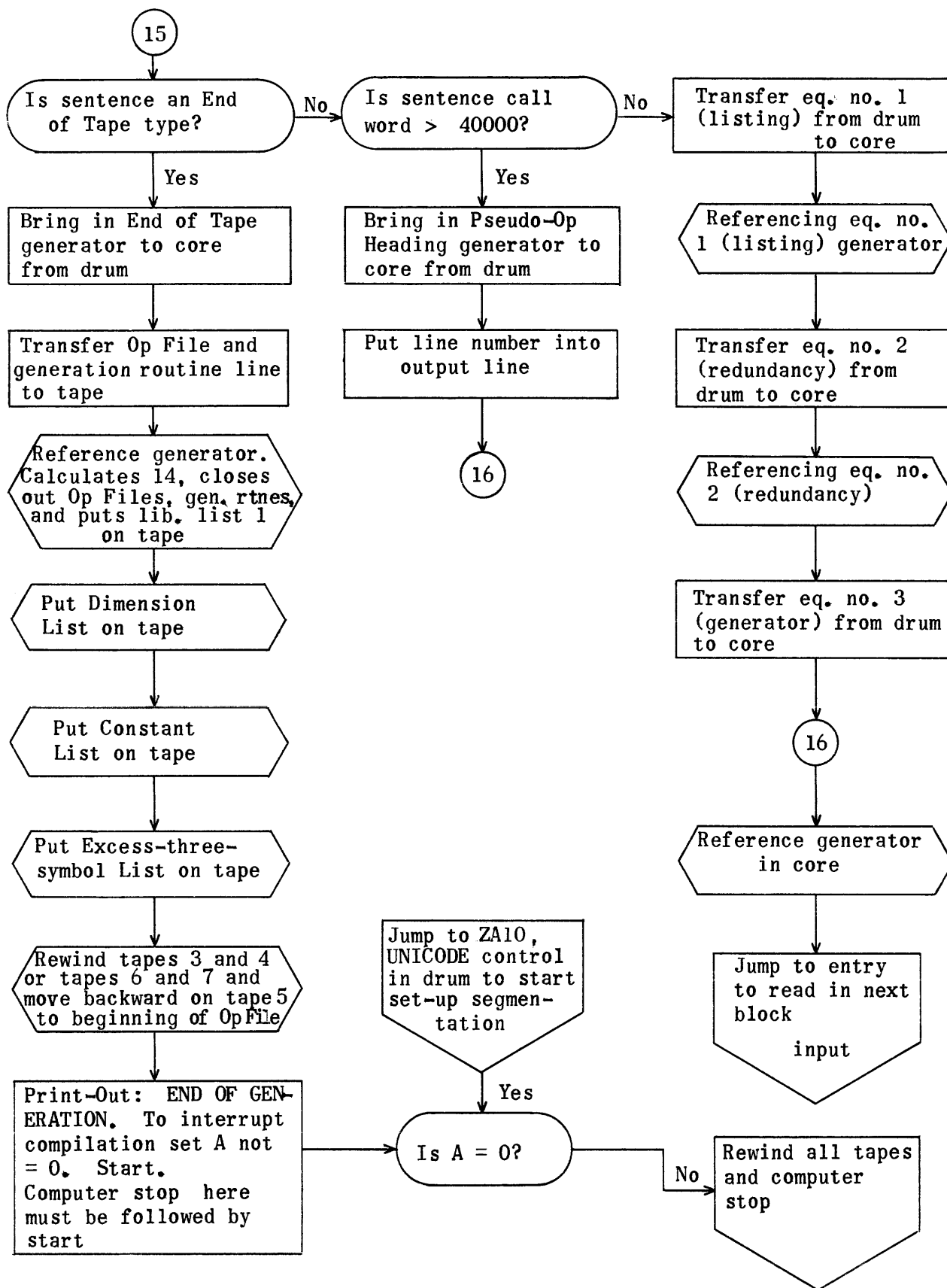
The only final termination to generation lies in the recognition and handling of an End of Tape sentence.

All generators or portions of generators are transferred to the core at the same starting address, 2512. This first line is an exit line, the entry line for the main referencing of a generator being the Reco equivalent of 2513.

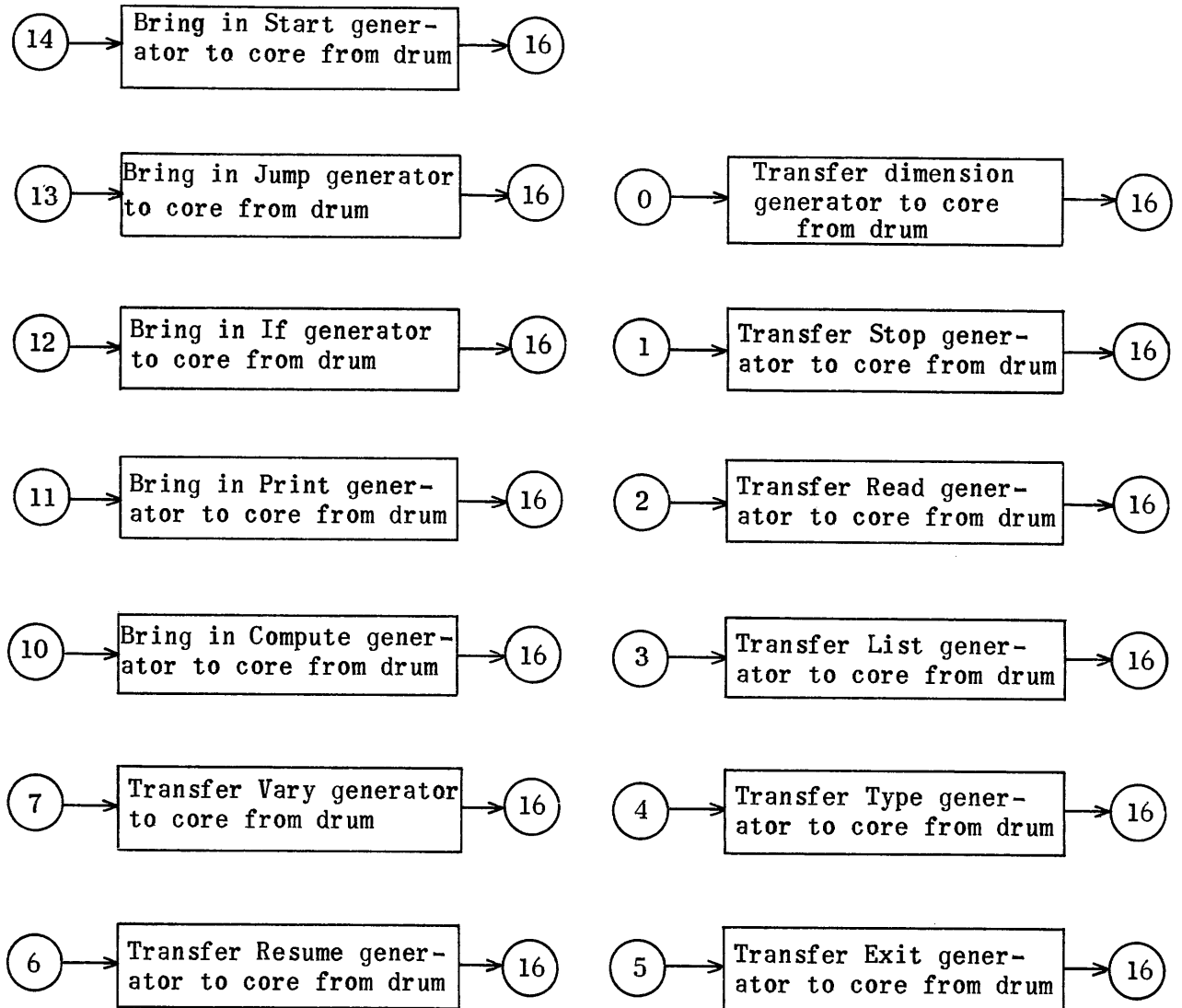
The generators are packed on the drum starting at 50212. Information concerning their assumed length on the drum and their initial addresses there is obtained by examination of the annotated copy of Generation Subroutine regions preceding the Reco coding of the Generation Phase.

Flow Chart for Control Generation Routine





Flow Chart for Control Generation Routine (cont.)



CONTROL GENERATION ROUTINE

	IA	CG			
0	TP	TN	A	} Bringing in 1st block (for any one sentence) from tape { 3 } to bk, buffer input region	
1	AT	CH2	GT3		
2	RJ	GT2	GT		
3	TP	BK	A	} Is input < 171?	
4	TJ	CH	CG10		
5	TP	TN	A	} Bringing in 2nd block for same sentence to input region from tape { 3 } 6	
6	AT	CH3	GT3		
7	RJ	GT2	GT		
10	RJ	HI	HI1	Check if line no. is in ref. list and, if so, giving it a call word	
11	TP	BK2	A	Name of sentence to A 20014 13 12	
12	RP	20015	CG21	} Determining if sentence is start, jump, if 11 10 7 6 5 4 print, compute, vary, resume, exit, type, 3 2 1 0 list, read, stop, or dimens. and going to proper transfer command accordingly.	
13	EJ	CH4	CG14		
14	SN	Q	0		
15	SA	CG34	1		
16	AT	CG35	A		
17	TV	A	CG20		
20	MJ	0	30000		
21	EJ	CH21	CG77	Is sentence an End of Tape?	
22	TP	CH1	A	} Is call word > 40000? If so, it is a Pseudo-Op Heading	
23	TJ	BK3	CG75		
Sentence is an equation	24	RP	DH30000	CG26	} Translation for equation
	25	TP	ET	KB	
	26	RJ	KB	KB1	
	27	RP	DI30000	CG31	} Redundancy check for equation
	30	TP	EU	KB	
	31	RJ	KB	KB1	
	32	RP	DJ30000	CG41	} Transferring equation generator to core
	33	TP	EV	KB	
	34	0	0	20015	
	35	0	0	CG41	} Unused words
36	0	0	0		
37	0	0	0		
40	TP	BK1	2531	} Putting line number into proper output line for Pseudo-Op Heading Routine to correct oversight in that routine	
41	RJ	KB	KB1	Standard return jump to any generator	
42	MJ	0	CG	Return to get next sentence	
43	RP	DR30000	CG41	} Transfers Start generator from drum to core and then jumps to reference the generator	
44	TP	CJ	KB		
45	RP	DM30000	CG41	Same with Jump	
46	TP	EA	KB	} Same with If	
47	RP	DK30000	CG41		
50	TP	EB	KB		
51	RP	DO30000	CG41	} Same with Print	
52	TP	EC	KB		

53	RP	DY30000	CG41	} Same with Compute
54	TP	ED	KB	
55	RP	DU30000	CG41	} Same with Vary
56	TP	EF	KB	
57	RP	DQ30000	CG41	} Same with Resume
60	TP	EY	KB	
61	RP	DW30000	CG41	} Same with Exit
62	TP	EH	KB	
63	RP	DT30000	CG41	} Same with Type
64	TP	EI	KB	
65	RP	DN30000	CG41	} Same with List
66	TP	EJ	KB	
67	RP	DP30000	CG41	} Same with Read
70	TP	EK	KB	
71	RP	DS30000	CG41	} Same with Stop
72	TP	EL	KB	
73	RP	DZ30000	CG41	} Same with Dimension
74	TP	EM	KB	
75	RP	DX30000	CG40	} Brings in Pseudo-Op Heading Routine, jumps to CG40 to perform 1st function of routine before referencing the generator proper
76	TP	EN	KB	
77	RP	DV30000	CG101	} Brings in End of Tape generator
100	TP	EO	KB	
101	TP	CH25	OP1	} Sends End of Tape data to tape
102	RJ	OP	OP2	
103	RJ	KB	KB1	Ref. generator; Calculates 14, closes out Op File, gen. rtnes, and puts list 1 on tape Puts Dimension List on tape Puts Constant List on tape Getting Excess-three symbol List Rewind 3 and 4 or 6 and 7 and move backward on 5 to beginning of Op File
104	RJ	UG	UG1	
105	RJ	IG	IG1	
106	RJ	BU	BU1	
107	RJ	EG	EG1	
110	RJ	BE	BE1	Termination of gen. print-out
111	MJ	O	ZA10	Exit to UNICODE control in drum
	CA	CG112		
	IA	CH		
0	0	0	171	
1	0	0	40000	
2	50	103	BK	Parameter to read 1st block of any sentence string-out to input
3	50	103	BK170	Parameter to read 2nd block of same sentence string-out to input
4	65	66245	46677	START
5	44	67475	27777	JUMP
6	34	31777	77777	IF
7	52	54345	6677	PRINT
10	26	51475	26766	COMPUT
11	70	24547	37777	VARY
12	54	30656	74730	RESUME
13	30	72346	67777	EXIT
14	66	73523	7777	TYPE
15	46	34656	67777	LIST

16	54	30242	77777	READ
17	65	66515	27777	STOP
20	27	34473	5065	DIMENS
21	30	50277	77777	END
22	0	23000	2	} Op File setup for End of Tape
23	0	0	0	
24	0	23000	0	Gen. setup for End of Tape
25	0	CH24	CH22	Parameter for referencing op routine for End of Tape
	CA	CH26		

COMPUTER ERROR ROUTINE

This routine is used when a compilation inconsistency occurs in the Generation Phase or later. It prints the following:

ALARM△XX△△Compilation△Inconsistency△(Possible Computer Error).
Recompile.

where XX, a decimal number from 1 - 20, is determined by where the routine is entered. The regular entrances to the routine are BR1 to BR24. If the entrance is BR1, XX = 1; if the entrance is BR10, XX = 8; if the entrance is BR24, XX = 20; etc. The routine should not be entered with a return jump.

After the print-out, all tapes are rewound, and the computer stops with PAK set at the UNICODE service entrance of compilation.

A rewind of all tapes with a computer stop is secured without a print-out by entering the routine at BQ6.

Eleven alarms (entries BR1-BR13) have been assigned at different portions of the UNICODE coding. Explanations of these alarms follow:

COMPILER INCONSISTENCY OR COMPUTER ERROR ALARMS

<u>Print-out</u>	<u>Entry</u>	<u>Description</u>
ALARM 1	BR 1	Subscripted variable symbol or call word is not found in the Dimension List.
ALARM 2	BR 2	Referenced sentence number is not found in the Reference (IZ) List.
ALARM 3	BR 3	Sentence number of last sentence in the range of a VARY is not found in the VARY File (VARY generator).
ALARM 4	BR 4	Initial address of Operand List > Current address (Equation Generator).
ALARM 5	BR 5	Call word is not found in Directory I (Segmentation Phase).
ALARM 6	BR 6	Op File III and Directory 4 are inconsistent. Flagged call word is not found in Op File III (Allocation Phase).
ALARM 7	BR 7	Tape on Servo 2 positioned incorrectly.
ALARM 8	BR 10	Tape on Servo 3 (or 6) positioned incorrectly.
ALARM 9	BR 11	Tape on Servo 4 (or 7) positioned incorrectly.
ALARM 10	BR 12	Tape on Servo 5 positioned incorrectly.
ALARM 11	BR 13	Call word within routine is not found in Op File III (Processor).

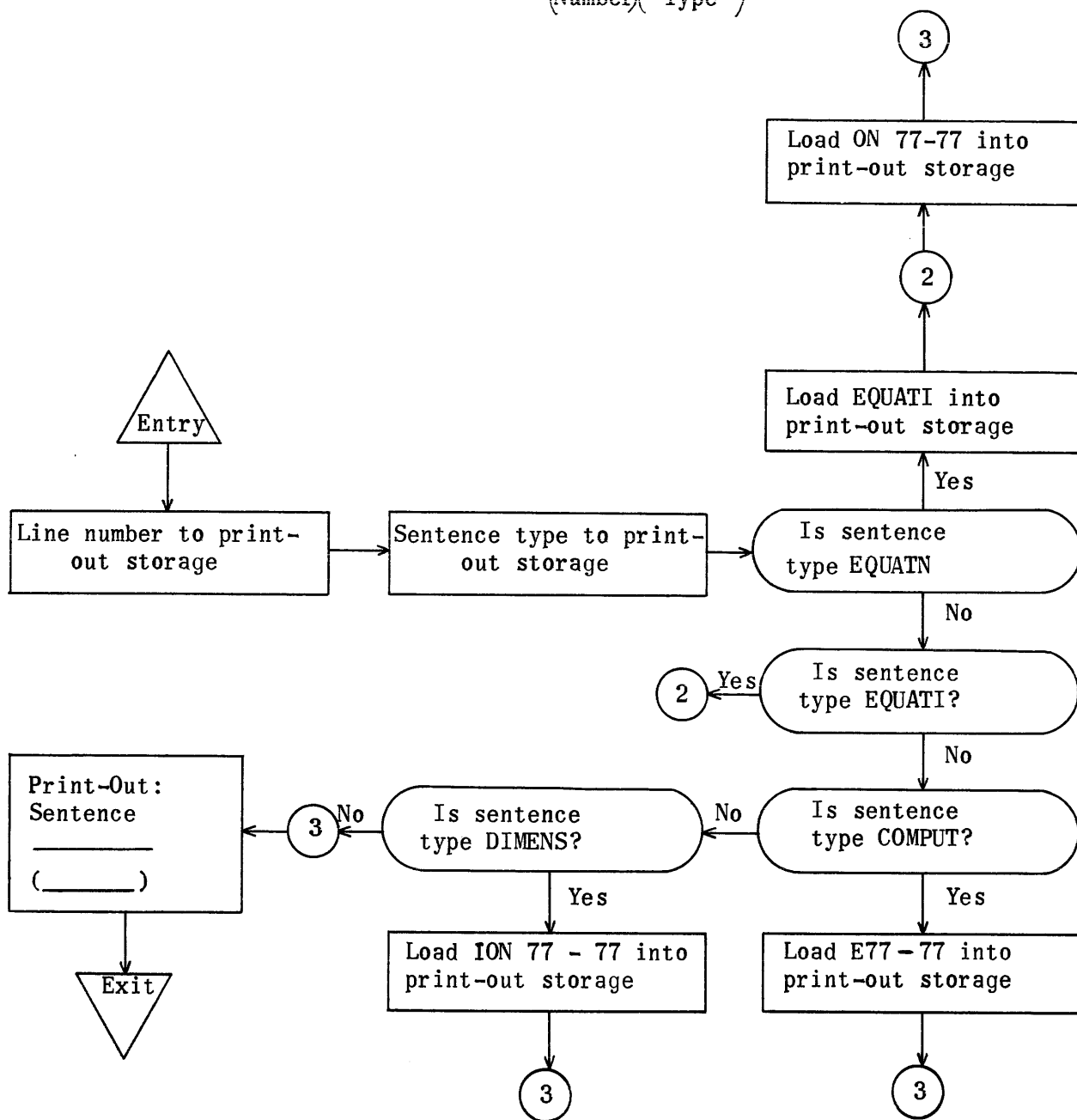
Machine Error Print

	IA	BR		
0	0	0	BR1	Used entries
1	RJ	BP	BQ	
2	RJ	BP	BQ	
3	RJ	BP	BQ	
4	RJ	BP	BQ	
5	RJ	BP	BQ	
6	RJ	BP	BQ	
7	RJ	BP	BQ	
10	RJ	BP	BQ	
11	RJ	BP	BQ	
12	RJ	BP	BQ	Unused entries
13	RJ	BP	BQ	
14	RJ	BP	BQ	
15	RJ	BP	BQ	
16	RJ	BP	BQ	
17	RJ	BP	BQ	
20	RJ	BP	BQ	
21	RJ	BP	BQ	
22	RJ	BP	BQ	
23	RJ	BP	BQ	
24	RJ	BP	BQ	
	CA	BR25		
	IA	BQ		
0	TP	BP	A	Manipulation to get proper alarm number into print-out coding
1	SS	BR	17	
2	AT	BP1	BQ3	
3	0	30000	30000	
4	TP	BP26	UP3	Print-out
5	RJ	UP2	UP	
6	TP	BQ17	BQ20	Entry for rewind and stop only. Sets index for 7 Uniservos
7	MJ	10000	BQ11	Bypasses next instruction if MJ1 is set
10	TP	BQ16	BQ20	Sets index of rewinding tapes for 5 Uniservos
11	TP	BP44	GT3	"Rewind parameter" to tape handler
12	RJ	GT2	GT	Rewind Uniservo via tape handler subroutine
13	RA	GT3	BP45	Increasing tape handler parameter to rewind next tape
14	IJ	BQ20	BQ12	Jump back to rewind remaining tapes
15	MS	0	ZA	Exit and stop
16	0	0	4	Index Constant (Used by Read Generator also)
17	0	0	6	Index Constant for 7 Uniservos
20	0	0	0	Holds index for rewinding Uniservos
	CA	BQ21		

0	IA 0	BP 0	30000	Set by entry "return jump" lines and used to compute alarm no.
1	TP	BP1	BP30	Dummy instruction used to make up instruction BQ3
2	04	77220	10126	1. $\Delta \Delta C$
3	05	77220	10126	2. $\Delta \Delta C$
4	06	77220	10126	3. $\Delta \Delta C$
5	07	77220	10126	4. $\Delta \Delta C$
6	10	77220	10126	5. $\Delta \Delta C$
7	11	77220	10126	6. $\Delta \Delta C$
10	12	77220	10126	7. $\Delta \Delta C$
11	13	77220	10126	8. $\Delta \Delta C$
12	14	77220	10126	9. $\Delta \Delta C$
13	04	03220	10126	10. $\Delta \Delta C$
14	04	04220	10126	11. $\Delta \Delta C$
15	04	05220	10126	12. $\Delta \Delta C$
16	04	06220	10126	13. $\Delta \Delta C$
17	04	07220	10126	14. $\Delta \Delta C$
20	04	10220	10126	15. $\Delta \Delta C$
21	04	11220	10126	16. $\Delta \Delta C$
22	04	12220	10126	17. $\Delta \Delta C$
23	04	13220	10126	18. $\Delta \Delta C$
24	04	14220	10126	19. $\Delta \Delta C$
25	05	03220	10126	20. $\Delta \Delta C$
26	0	BP27	15	Parameter for print-out
27	24	46245	44701	ALARM Δ
30	0	0	0	Filled in by alarm number computation coding
31	51	47523	44624	O M P I L A
32	66	34515	00134	T I O N Δ I
33	50	26515	06534	N C O N S I
34	65	66305	02673	S T E N C Y
35	01	17525	16565	Δ (P O S S
36	34	25463	00126	I B L E C
37	51	47526	76630	O M P U T E
40	54	01305	45451	R Δ E R R O
41	54	43220	10154	R) . $\Delta \Delta$ R
42	30	26514	75234	E C O M P I
43	46	30227	77777	L E .
44	10	1	0	Rewind parameter to tape handler
45	0	1	0	Constant (Used in Read Generator also)
	CA	BP46		

Flow Chart for WA, WB "Sentence ____ (____)" Print-Out - Generation

(Line \ Sentence
Number \ Type)



WA Subroutine to Print-Out "Sentence (Line Number) (Sentence Type)"
without Error Referencing

	IA	WA		
0	MJ	0	30000	Exit
1	MJ	0	WA2	Start
2	TP	WL1	WB3	Line number to print-out storage
3	TP	WL2	A	} Sentence type to print-out storage and A
4	TP	A	WB5	
5	TP	WB10	WB6	
6	EJ	WB15	WA15	Is type EQUATN?
7	EJ	WB11	WA16	Is type EQUATI?
10	EJ	WB16	WA20	Is type COMPUT?
11	EJ	WB17	WA22	Is type DIMENS?
12	TP	WB	UP3	} Print-Out: Sentence _____(_____)
13	RJ	UP2	UP	
14	MJ	0	WA	
15	TP	WB11	WB5	} Load {EQUATI ON77—77} into print-out storage
16	TP	WB12	WB6	
17	MJ	0	WA12	
20	TP	WB13	WB6	Load E77—77 into print-out storage
21	MJ	0	WA12	
22	TP	WB14	WB6	Load ION77777 into print-out storage
23	MJ	0	WA12	
	CA	WA24		

	IA	WB		
0	0	WB1	7	Parameter for print-out
1	65	30506	63050	S E N T E N
2	26	30010	17777	C E Δ Δ 77 77
3	0	0	0	Line number
4	01	01011	77777	Δ Δ Δ (77 77
5	0	0	0	} Sentence name
6	0	0	0	
7	43	01017	77777	) Δ Δ 77 77 77
10	77	77777	77777	Filler
11	30	53672	46634	EQUATI
12	51	50777	77777	ON 77—77
13	30	77777	77777	E 77—77
14	34	51507	77777	ION 77 - 77
15	30	53672	46650	EQUATN
16	26	51475	26766	COMPUT
17	27	34473	05065	DIMENS
	CA	WB20		

ROUTINE TO CONTROL TRANSFER OF OP FILE 1 ITEMS
AND GENERATED SUBROUTINES TO TAPE

To use Op Control Routine, a parameter word must be sent to Op₁ prior to the entry instruction RJ Op Op₂. u of Op₁ must contain the address of the first line of the generated subroutine; v, the address of the first line of the Op File 1 entry for the subroutine. The function of the routine is to transfer the generated subroutines and Op File 1 items to Uniservos. Z lines are added to fill up the last block of each generated subroutine. 170₈ lines constitute a block.

The actual writing on tape units is done by means of the generalized tape handler routine. The proper parameter words and entries to the tape handler routine are supplied in this Op Control Routine.

The entry instruction of this routine is a return jump to RG subroutine which adds a call word to the Op File and alters the exit line of the generated routine if the sentence generated is the last sentence in a VARY loop.

Op File 1 items are stored in a buffer region, NP, and accumulated until they fill a block, whence they are transferred to tape. Not taken care of in this routine is the final unfilled block of Op File 1 items. The coding necessary to fill this final Op File 1 block with Z's and transfer it to tape is contained in the End of Tape instruction.

The routine handles data stored either in the core or drum. If subroutine data is in the drum and it exceeds a block, it is transferred a block at a time into region GN in the core and thence onto tape. Subroutine data groups less than a block are transferred from drum to core in one set of repeat instructions. Op File 1 data groups in the drum are transferred to NP in the core just as if they were located in the core.

Subroutine data groups in the core are handled somewhat faster. All complete blocks of a subroutine are handled as a single unit in their referral to the generalized tape handler. The last partial block of a subroutine is then transferred to region GN, where the proper number of Z lines is added to make a complete block. This region GN of 170₈ lines is needed only during Op Control operation. Between referrals to the routine it may be used for other

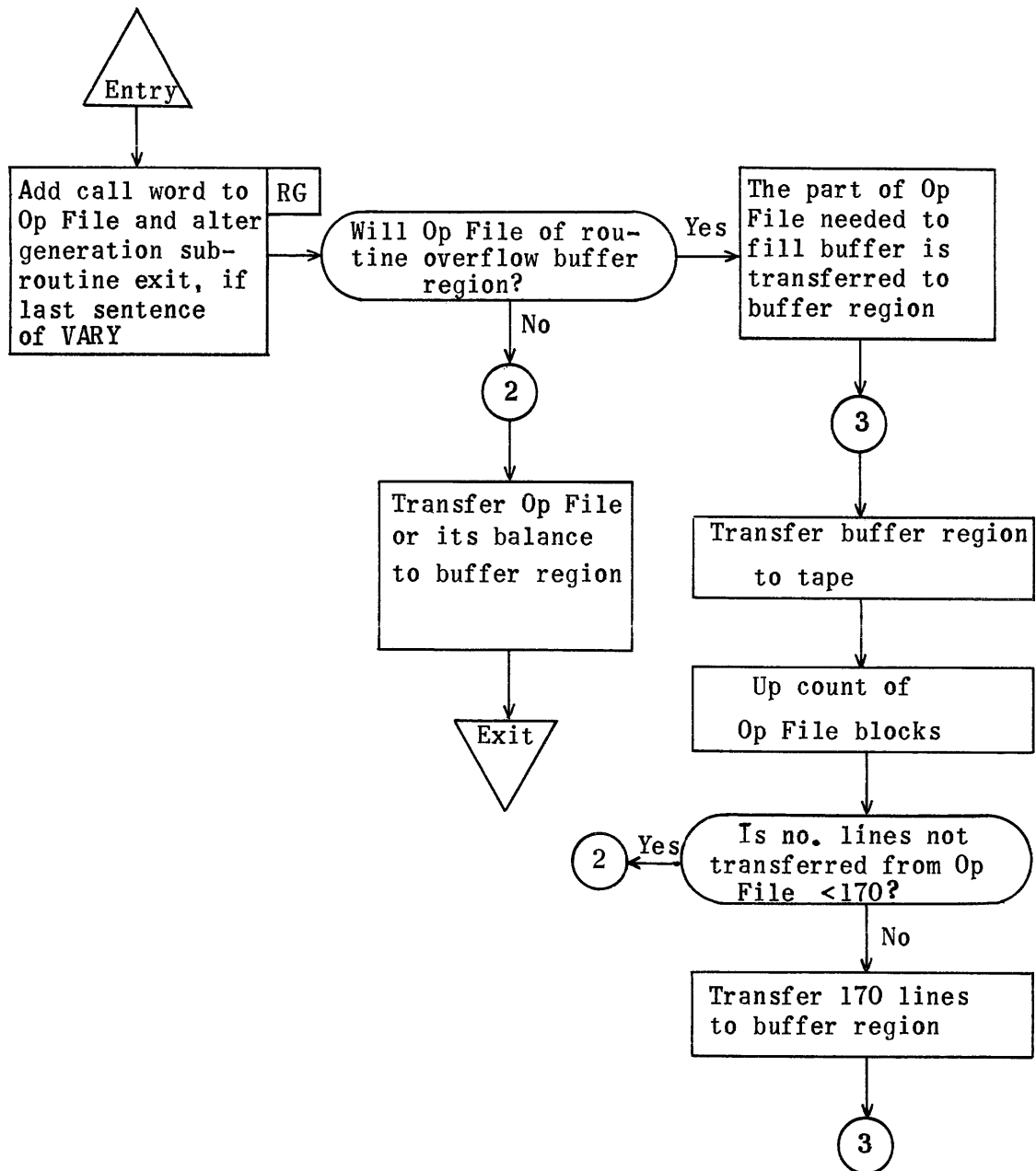
temporary purposes. In this respect it differs from region NP which is reserved exclusively for Op File 1 item storage during generation.

At the beginning of generation instruction RJ RQ RQ1 causes the proper starting blocks to be written on tape, and sets up the proper parameters for the generalized tape handler, depending upon whether 5 or 7 Uniservos are used. On Uniservo for generated subroutines, GEN TAPE is written on the 21st and 22nd lines of the first block, SUBROUTINES on the 1st and 2nd lines of the 2nd block. On Uniservo for Op File items, FILE TAPE and OP FILE 1 are similarly put on the first two blocks, with Z - fillers on the balance of the block lines. The counts of blocks of subroutines and Op File 1 items include these starting blocks.

The v portion of the first line of a subroutine program must contain the number of lines of prelude and routine. Similarly, the v portion of the first line of an Op File 1 item should contain the number of lines in the item. These figures are used in this routine.

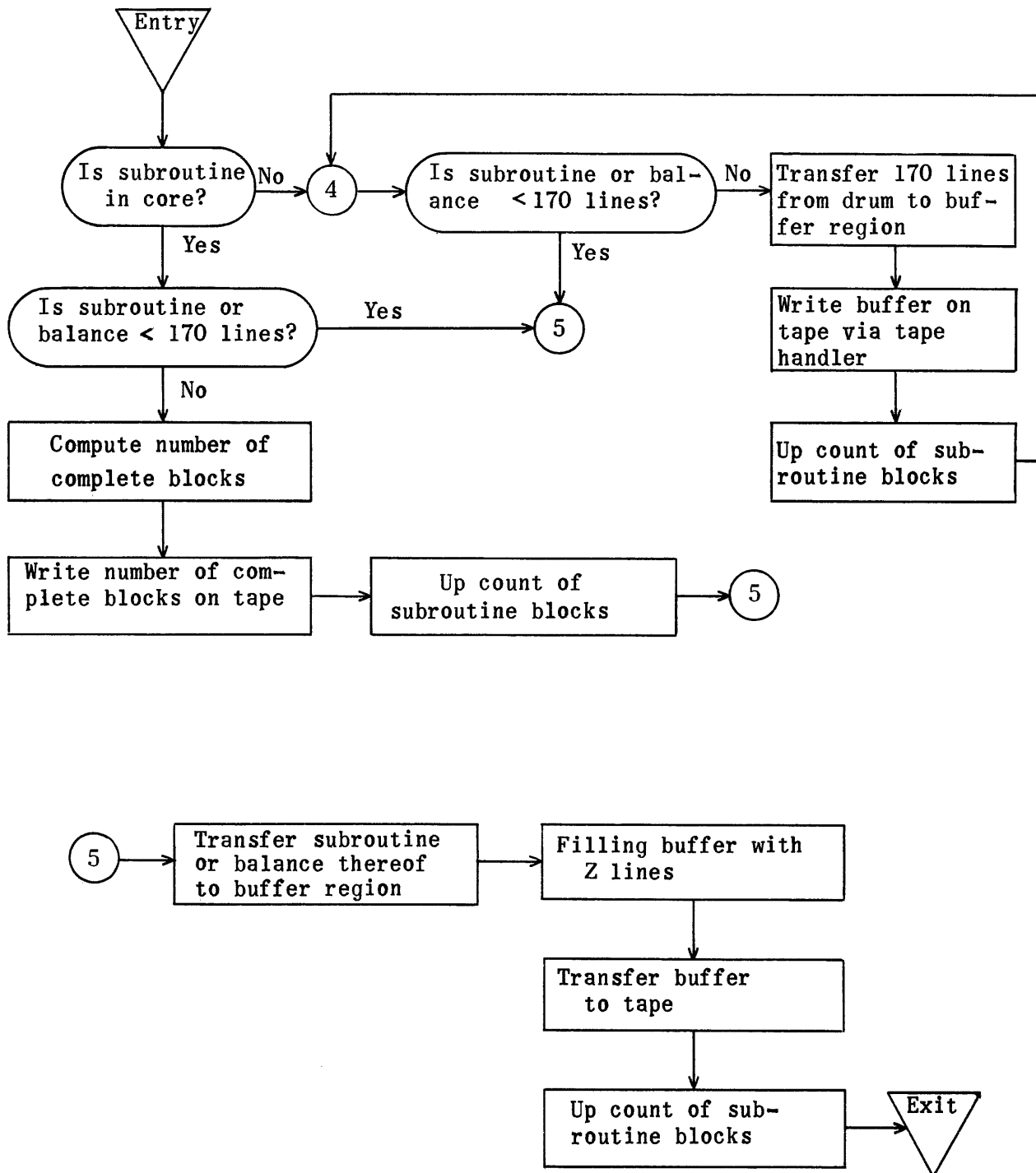
ES, the counter for the number of lines in NP, is used in the End of Tape routine to transfer the final Op File 1 block to tape. ES5, the number of blocks of Op File 1 written on tape, and ES6, the number of blocks of subroutines written on tape, are also used in subsequent routines.

Flow Charts for Op Control Routines - Generation
Op Routine To Store Op Files and Write Buffer Loads on Tape



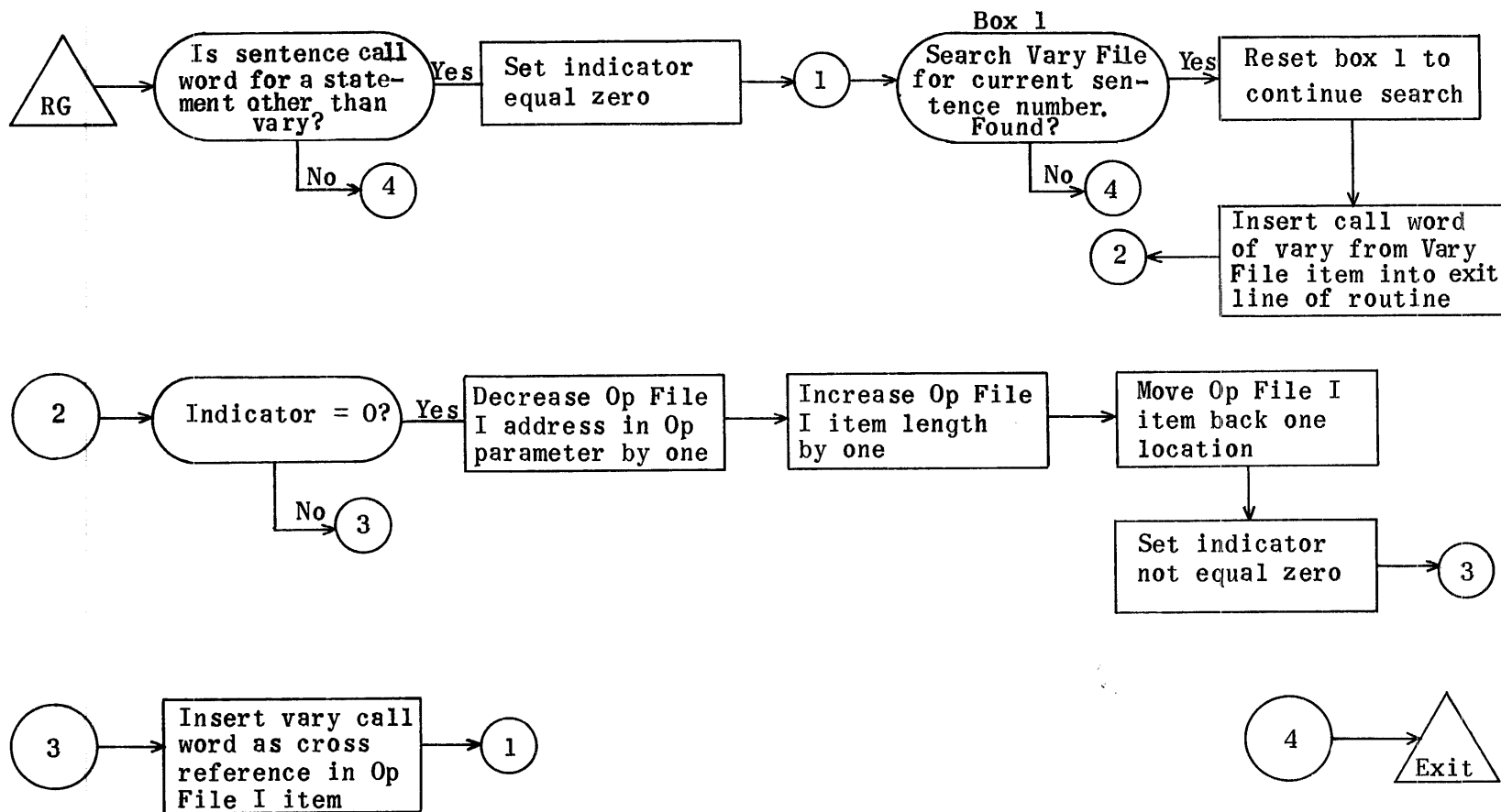
Flow Charts For Op Control Routines - Generation - (Cont.)

XP Routine To Write Generated Routines on Tape



Flow Charts for Op Control Routines - Cont.

RG Routine to Insert Vary Call Word in Exit Line and in Op File I Item of Last Statement in a Vary Range



OP CONTROL SUBROUTINE

Routine to	IA	OP			
Store Op	0	MJ	0	30000	Exit
Files and	1	O	30000	30000	u = address of 1st line of generated sub-
Write Op					routine. v = address of 1st line of Op File
Files on	2	RJ	RG	RG1	VARY subroutine alters exit lines of routine
Tape					being written on tape
	3	TP	OP1	Q	Address of 1st line of Op File → ES11
	4	QT	GP3	ES11	
	5	SP	OP1	17	Number of lines in Op File → ES1
	6	TU	A	OP10	
	7	TP	GP3	Q	
	10	QT	30000	ES1	ES + ES1 → ES2
	11	TP	ES	ES2	
	12	RA	ES2	ES1	Does ES2 exceed 170?
	13	TJ	GP5	OP44	
	14	TP	GP5	A	Number of lines left to fill NP → ES4
	15	ST	ES	ES4	
	16	SA	GP6	17	Sets up u of repeat command Op 25 so that
	17	TU	A	OP25	
					proper number of lines needed to fill NP
					are transferred to it
	20	TP	ES	ES3	Sets up v of Op 26 such that Op File items are
	21	RA	ES3	RC1	
	22	TV	ES3	OP26	transferred to correct part of NP
	23	SP	ES11	17	
	24	TU	A	OP26	Sets up u of Op 26 to correct beginning add.
	25	RP	30000	OP27	
	26	TP	30000	30000	Transfers Op File items to fill up NP
	27	TP	RC	GT3	
	30	RJ	GT2	GT	Writes completed NP onto a block of tape
	31	RA	ES5	GP10	
	32	TP	GP7	ES	Count of Op File blocks
	33	RA	ES11	ES4	Clears ES storage
					ES11 + ES4 → ES11, the next address of Op
					File item to be transferred
	34	RS	ES1	ES4	ES1 - ES4 → ES1, number of lines left in
					Op File item to be transferred
	35	TJ	GP5	OP44	Is 170 > no. of lines left in Op File item?
	36	SP	ES11	17	Setting up u of Op 64 to proper address
	37	TU	A	OP41	
	40	RP	30170	OP42	Transferring 170 Op File items to NP
	41	TP	30000	NP	
	42	TP	GP5	ES4	170 → ES4
	43	MJ	0	OP27	Number lines in NP → ES3
	44	TP	ES	ES3	
	45	TP	A	ES	Number of lines due to be in NP → ES
	46	TP	ES1	A	
	47	SA	GP6	17	Sets up u of repeat to proper value
	50	TU	A	OP55	
	51	RA	ES3	RC1	Sets up v of Op 56 to correct address
	52	TV	ES3	OP56	

1 block of
170 lines
to buffer
region NP
for trans-
fer to
tape

Last
partial
block to
NP

	53	SP	ES11	17	}	Sets up u of Op 56 to correct address
	54	TU	A	OP56		
	55	RP	30000	XP	}	Transfers last quantity of Op File items to NP, leaving NP as an unfilled partial buffer region block.
	56	TP	30000	30000		
		CA	OP57			
Routine to Write Generated Routines on Tape		IA	XP			
	0	TU	OP1	XP2	}	Number of lines of subroutine → ES12
	1	TP	GP3	Q		
	2	QT	30000	ES12	}	Address of 1st line of subroutines → ES10
	3	LQ	OP1	25		
	4	QT	GP3	ES10	}	If 30000 > address, subroutine is in core. Otherwise, assumed to be in drum
	5	TJ	GP6	XP22		
Drum	6	TP	ES12	A	}	Test to see if subroutine has fewer than 170 lines left
	7	TJ	GP5	XP40		
	10	SP	ES10	17	}	Set-up u of XP13 to right address
	11	TU	A	XP13		
	12	RP	30170	XP14	}	Transfer 170 lines of subroutines to GN
	13	TP	30000	GN		
	14	TP	RC4	GT3	}	Write block on tape
	15	RJ	GT2	GT		
	16	RA	ES6	GP10		Count of blocks
	17	RS	ES12	GP5		Reduce ES12 to no. of lines left in subrtne.
Core	20	RA	ES10	GP5		Increase ES10 to address of next line of subroutine to be transferred
	21	MJ	0	XP6		
	22	TP	ES12	A	}	Test for less than 170 lines in subroutine
	23	TJ	GP5	XP40		
	24	DV	GP5	ES7		Number of blocks in subroutine → ES7
	25	TP	A	ES12		Remainder or number lines in last partial block to ES12
	26	TP	ES7	A	}	Setting up parameter with correct number of blocks to be written on tape
	27	ST	GP10	A		
	30	LA	A	25	}	Setting up correct referencing address of parameter
	31	AT	RC4	GT3		
Partial Block	32	TV	ES10	GT3		Writing blocks on tape
	33	RJ	GT2	GT		Count of blocks
	34	RA	ES6	ES7		Calculating number of lines written on tape
	35	MP	ES7	GP5		Correcting ES10 address to that of next line to be transferred
	36	AT	ES10	ES10		
	37	TP	ES12	A	}	Test to see if any lines left to be written of subroutine
	40	ZJ	XP41	OP		
	41	SA	GP6	17	}	Setting up u of repeat to proper value
	42	TU	A	XP45		
	43	SP	ES10	17	}	Setting up correct referencing address of next line of subroutine in XP46
	44	TU	A	XP46		
	45	RP	30000	XP47	}	Putting remaining lines of subroutine in GN
	46	TP	30000	GN		

47	TP	GP5	A	} Setting up u of XP55 such that remainder of GN is filled with right number of lines of Z's
50	SS	ES12	17	
51	TU	A	XP55	
52	RA	XP55	GP4	
53	TV	ES12	XP56	} Setting up v of XP56 so that Z lines start at right place in GN
54	RA	XP56	RC3	
55	RP	10000	XP57	} Filling up GN with Z's
56	TP	GP2	30000	
57	TP	RC4	GT3	Parameter word → generalized tape handler
60	RJ	GT2	GT	To tape handler
61	RA	ES6	GP10	Count of blocks
62	MJ	0	OP	Exit
	CA	XP63		

	IA	RC		
0	71	00105	NP	Parameter word to write 1 block on tape 5 from NP
1	0	0	NP	Address of 1st line of Op File storage
2	71	00104	GN	Tape 4 parameter write from GN
3	0	0	GN	Address of 1st line of storage used for a partial block of a gen. routine
4	0	0	0	Parameter temporary used for writing from GN to either tape 4 or tape 7

CA RC5

	IA	GP		
0	01	01015	15201	△ △ △ 0 P △
1	31	34463	00104	F I L E △ 1
2	74	74747	47474	Line of Z's
3	0	0	77777	Mask
4	0	10000	0	
5	0	0	170	
6	0	0	30000	
7	0	0	0	
10	0	0	1	
11	01	65672	55451	△ S U B R 0
12	67	66345	03065	U T I N E S
13	01	31344	63001	△ F I L E △
14	01	66245	23001	△ T A P E △
15	01	01323	05001	△ △ G E N △
	CA	GP16		

Op Control Subroutine-Sequential Uses
of Temporary Storage (ES)

- 0 Number of lines in NP, buffer region in which Op Files are accumulated
 for writing on tape.
- 1 { Number of lines in current Op File being stored.
 { Number of lines remaining to be stored from current Op File.
- 2 Used to accumulate ES + ES1.
- 3 { ES $\rightarrow$ ES3
 { (ES3 + NP) $\rightarrow$ ES3, giving the address in NP to which next line is to
 { be transferred.
- 4 Number of lines put in NP from current Op File the last time a group of
 lines was transferred.
- 5 Number of blocks of Op File 1 written on tape.
- 6 Number of blocks of subroutines written on tape.
- 7 Number of whole blocks of subroutines to be written on tape.
- 10 Address of next line of subroutine to be written on tape.
- 11 Address of next line of Op File to be transferred to NP.
- 12 { Number of lines of subroutines.
 { Number of lines of subroutine in last block.

OP CONTROL SUBROUTINE - RG ROUTINE

To Insert Vary Call Word in Exit Line of Last Statement of
Range and in the Op File I Item

④	IA	RG		
	0	MJ	0	30000 Exit
	1	TU	OP1	RG3 Set address of routine into RG3
	2	TP	RG62	Q u-mask → Q
	3	QT	30000	A Routine CW → A
	4	TJ	RG63	RG7 27000 > CW?
	5	TJ	RG64	RG12 No; 30000 > CW?
	6	MJ	0	RG No; so out
	7	TJ	RG65	RG 22000 > CW? If so, out
	10	TJ	RG66	RG12 No; 23000 > CW?
	11	MJ	0	RG No; so out
①	12	TP	RG62	RG71 Set QJ indicator bit = 0
	13	TU	VF	RG16 CW = 22 --- or 27 ---
	14	TU	RG70	RG17 Set EJ initially
	15	TP	BK1	A Sentence number → A
	16	RP	30000	RG } Check for matching sentence number in
	17	EJ	30000	RG20 } Vary File
	20	SP	RG16	0 jn → AR _u
	21	LQ	Q	17 jn - r → Q _u
	22	TU	Q	RG16 Set RP for Continuing Search
	23	SS	Q	0 r → AR _u
	24	SA	RG17	0 Add r to Starting Point of Previous Search
	25	TU	A	RG17 Reset EJ to continue search at next word
②	26	TU	A	RG33 Set address of Vary CW in transfer command
	27	TU	A	RG56 Set Vary CW address in shift command
	30	LQ	OP1	Q25 Address of routine → Q _v
	31	TV	Q	RG33 Set routine address in transfer command
	32	RA	RG33	RG67 Set to address of exit line
	33	TV	30000	30000 Transfer Vary CW to exit line of routine
	34	TP	RG71	Q } Determine if this Op File I item has been
	35	QJ	RG56	RG36 } changed before
	36	SP	OP1	17 No, so enter Op File address in A _u
	37	TU	A	RG43 } Enter Op File I address in commands
	40	TU	A	RG50 }
	41	TU	A	RG52 }
	42	TU	A	RG54 }
	43	SP	30000	17 Enter # words of item in A _u
	44	SA	RG64	0 Add j = 3
	45	TU	A	RG53 Set into RP
	46	RS	OP1	GP10 Decrease Op File I address in parameter
	47	TV	A	RG54 Set transfer instruction to new address of
				Op File I item
	50	SA	30000	0 } Set address for insertion of new cross
	51	TV	A	RG60 } reference
	52	RA	30000	GP10 Increase # words in Op File I item by one

③

53	RP	30000	RG55	} Move Op File I item up one location
54	TP	30000	30000	
55	TP	RG	RG71	Set QJ indicator bit = 1
56	SP	30000	17	Vary CW $\rightarrow$ A_{Ru}
57	TP	RG62	Q	u-mask $\rightarrow$ Q
60	QT	A	30000	Enter Vary CW as Cross Reference in Op File I
61	MJ	0	RG15	Back to continue search
62	0	77777	0	
63	0	27000	0	
64	0	30000	0	
65	0	22000	0	
66	0	23000	0	
67	0	0	6	
70	0	VF1	0	
71	0	0	0	
72	CA	RG72		

CONSTANT CALL WORD ROUTINE FOR GENERATION

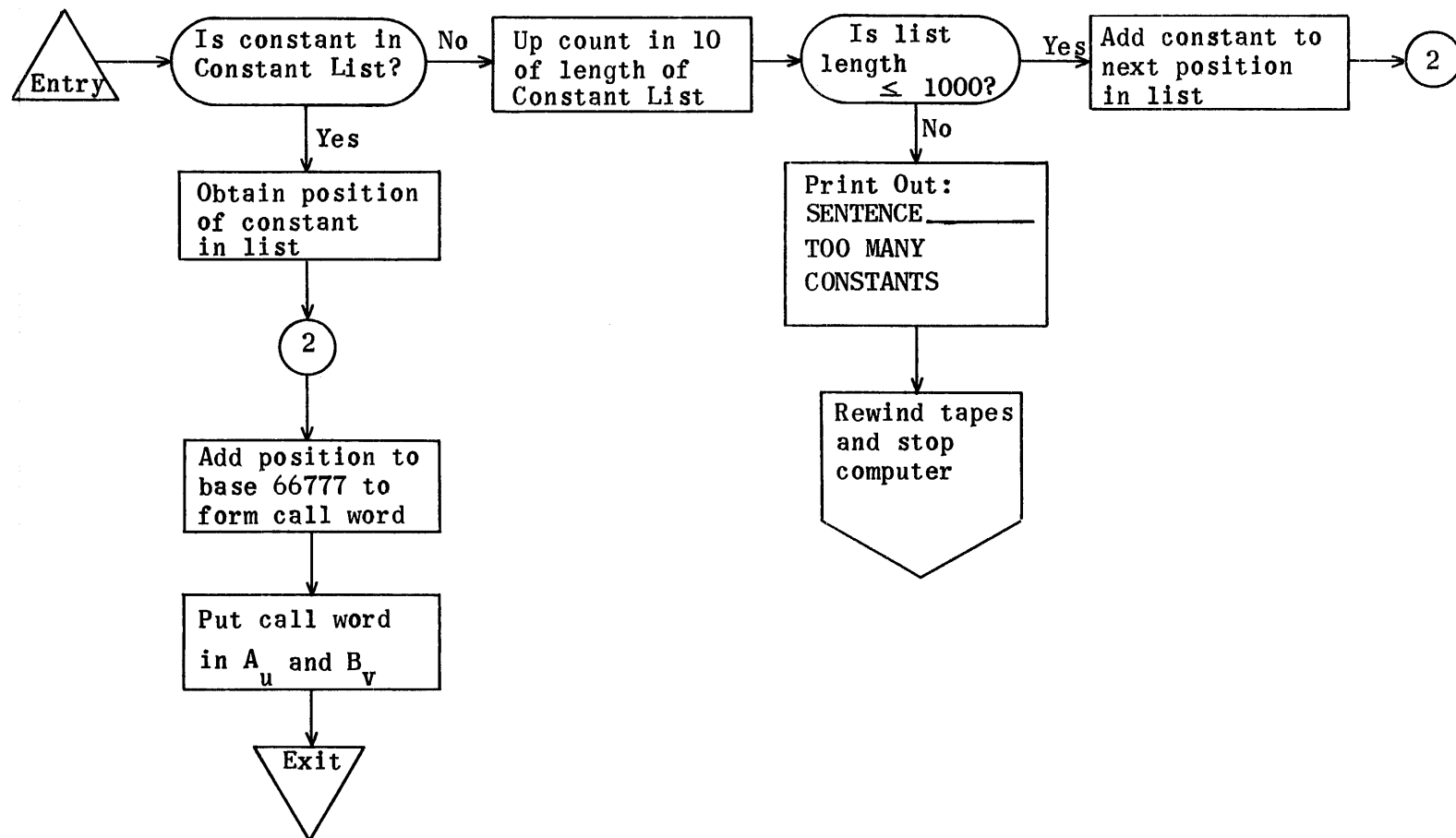
At the start of string-out 00 20000 00000 is put in fixed address 10, which is a counter register for the number of constants. Both u and v of 10 increase with each added constant. If n of 0 2, n in 10 exceeds 1000_8 , the computer stops with the alarm print-out -- (Sentence _____) Too Many Constants.

Input constant goes to A and the instruction RJ CW CW1 activates the routine. The call word output goes to Q_v and A_u .

The routine occupies 36_8 lines in region CW. Region CL, the list of constants, may occupy up to 1000_8 lines in the maximum-size problem. Needed for the alarm print-out is the UP print-out subroutine.

If a constant is already in list CL, entry to the routine gives the call word by determination of its position. If a constant is not in the list, it is added to it and given the next call-word-number assignment.

Flow Chart - CW Call Word Routine of Constants - Generation



Call Word-Generation

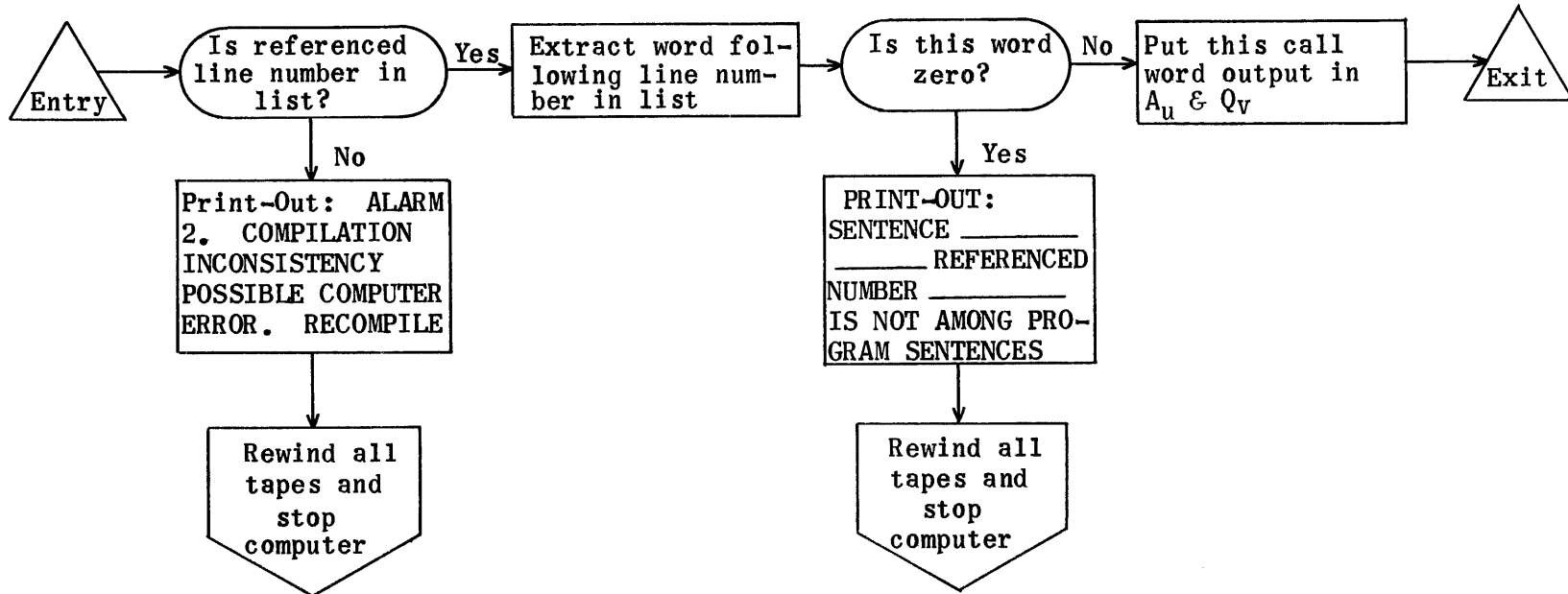
	IA	CW		
	0 MJ	0	30000	Exit
	1 TU	10	CW2	Sets up u of repeat by using Constant List count in 10
	2 RP	30000	CW7	} Checks to see if constant is in CL, Con. List
	3 EJ	CL	CW4	
	4 SN	Q	17	
	5 SA	10	0	$[-j, -(n-r)] \quad (A_R)_u$
	6 MJ	0	CW22	$[-j, r-n] + [j, \bar{n}]^R_u = r \rightarrow (A_R)_u$
When constant is not in list	7 TP	A	Q	Constant $\rightarrow q$
	10 TV	10	CW20	$n \rightarrow v$ of CW17
	11 RA	10	CW27	Counter 10 increased by 1
	12 TJ	CW30	CW17	Is Call List ≤ 1000 ?
	13 RJ	WA	WA2	Gets sentence number print-out
	14 TP	CW26	UP3	} Sends parameter to error print-out routine and gets print-out: Too Many Constants
	15 RJ	UP2	UP	
Adding Constant to list	16 MJ	0	BQ6	Jump to rewind tapes and computer stop rtne.
	17 RA	CW20	CW36	$(n + CL) \rightarrow v$ of CW20
	20 TP	Q	30000	Constant added to next position in list
	21 SP	10	17	$n \rightarrow (A_R)_u$
	22 AT	CW31	Q	$(r + 66777) \rightarrow q_u$. Call word in q_u
	23 QT	CW35	Q	Extraneous material in q destroyed via mask and QT. Call word formed in A_u
	24 LQ	Q	25	Call word put in Q_v
	25 MJ	0	CW	Jump to exit
	26 40	CW32	3	Parameter word used in print-out
	27 0	1	1	Used also as a constant in LS routine and in read generator
	30 0	21001	1001	Threshold check on size of CL
	31 0	66777	0	Base number from which call words are determined by adding to position in list CL
	32 66	51510	14724	T O O Δ M A
	33 50	73012	65150	N Y Δ C O N
	34 65	66245	6665	S T A N T S
	35 0	77777	0	Mask
	36 0	0	CL	Address of 1st line of Constant List
	CA	CW37		

Routine D to Get Call Word of Referenced Line
Number from List IZ - Generation

To use this routine, the referenced line number is put in A. Then instruction RJ LW LW1 will put the call-word output in A_u and Q_v . If a line number is not found in list IZ, the computer will stop with the error print-out: ALARM 2. COMPILATION INCONSISTENCY (POSSIBLE COMPUTER ERROR). RECOMPILE.

If the call word of a line number is not found, the computer will stop with the error print-out: SENTENCE _____ (_____) REFERENCED NUMBER _____ IS NOT A PROGRAM SENTENCE.

Flow Chart - LW Generator Subroutine to Get Call Word of Referenced
Line Number From Referenced-Line-Number List IZ



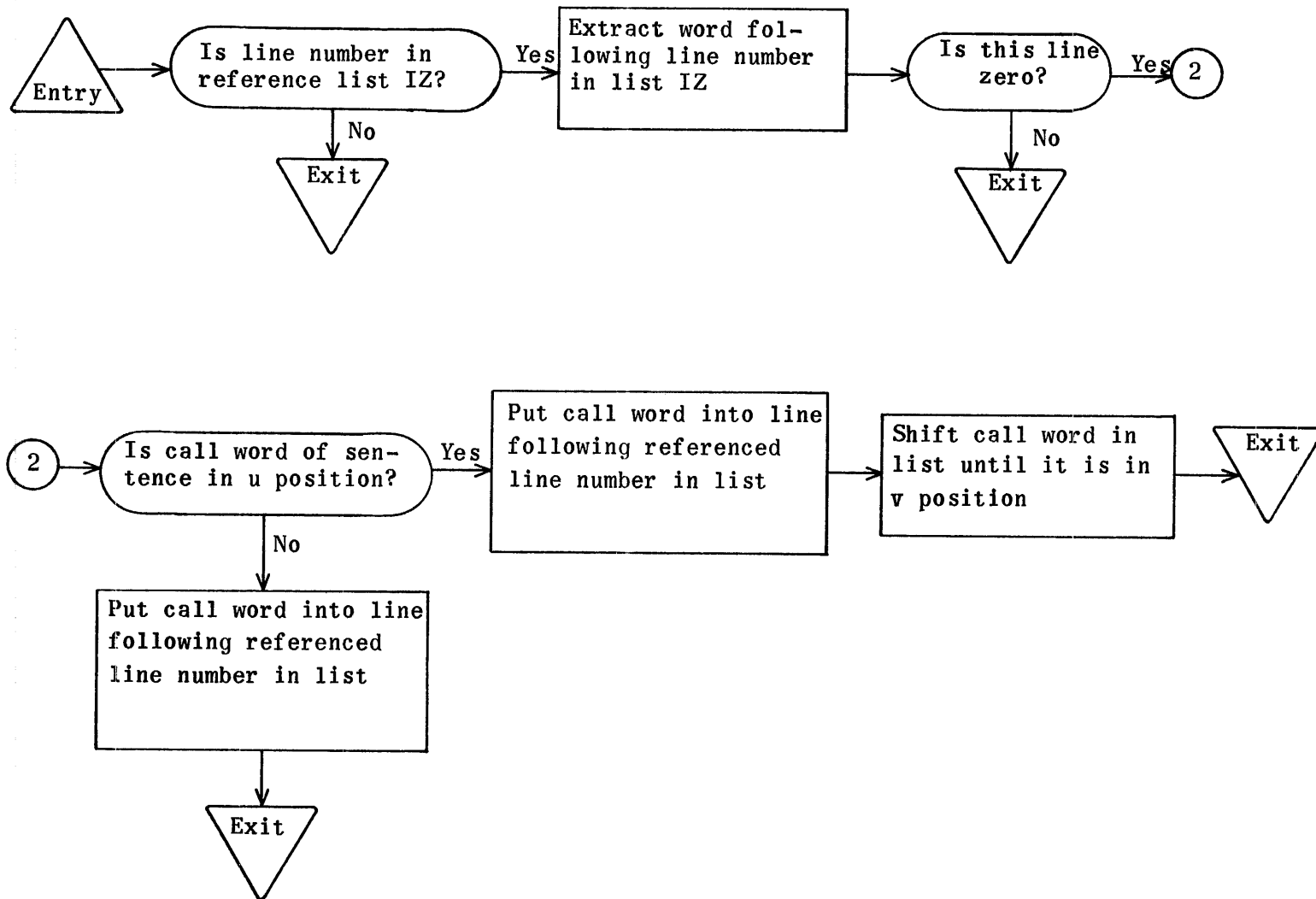
Generation--LW Routine to Get Call Word of Referenced
Line No. from List IZ--Routine D

	IA	LW		
0	MJ	0	30000	Exit
1	TP	A	LW27	Entry.Storing line no.for possible print-out
2	TU	11	LW3	Setting up repeat via Reference List Counter
3	RP	30000	BR2	} Search for line number in Line No. Call
4	EJ	IZ	LW5	
5	SN	Q	17	Word List
6	SA	11	0	$- [j, (n-r)] = -j, (r-n)$
7	SA	LW23	0	$[-j, (r-n)] + [j,n] = r$
10	TU	A	LW11	$r + IZ \rightarrow A_u$
11	TP	30000	A	Setting up u of next instruction
12	ZJ	LW13	LW16	Call word of line no. $\rightarrow A_v$
13	TP	A	Q	Test if call word is there
14	LA	A	17	Call word $\rightarrow Q_v$
15	MJ	0	LW	Call word $\rightarrow A_u$
16	RJ	WA	WA2	Sentence no. Print-Out
17	TP	LW22	UP3	} Referenced No. -- Is not a Program Sentence.
20	MJ	0	CW15	
				Jumps to computer stop after rewind tapes
				via CW routine portion
21	07	77777	77777	Used as a mask in LS routine
22	40	LW24	11	Parameter for Print-Out
23	0	IZ	0	Address of 1st line of line no. list
24	54	30313	05430	R E F E R E
25	50	26302	70150	N C E D Δ N
26	67	47253	05401	U M B E R Δ
27	0	0	0	
30	01	34650	15051	Δ I S Δ N O
31	66	01240	15254	T Δ A Δ P R
32	51	32542	44701	O G R A M Δ
33	65	30506	63050	S E N T E N
34	26	30227	77777	C E .
35	65	77777	77777	Unused
	CA	LW36		

Line Number Check Routine C -- Generation

Instruction RJ HI HI1 activates this routine without input. The routine picks out the line number of the sentence under surveillance from buffer region BK and checks to see if this number has previously been put in Line Number Reference List IZ. If an equality is found, a check is made to see if the call word of the line number has been put in the address following the line number in IZ. If not, the call word of the line number is obtained from the buffer region and inserted in the v part of this location.

Flow Chart for Routine C Generation - HI Reference-List Line-Number Checking Routine



Reference-List Line-Number Checking Routine C-- Generation

	IA	HI		
	0 MJ	0	30000	Exit
	1 TU	11	HI3	Sets up u of repeat by counter 11
	2 TP	BK1	A	Line no. to A
	3 RP	30000	HI	} Is line no. in reference list IZ?
	4 EJ	IZ	HI5	
	5 SN	Q	17	-j, (r-n)
	6 SA	11	0	j, n+[-j, (r-n)] = r
	7 SA	HI26	0	r + IZ → A _u
	10 TU	A	HI12	Sets up instruction to get call word of line number to A
	11 LQ	HI12	Q25	Puts address in Q _v of line that should have call word
	12 TP	30000	A	
	13 ZJ	HI	HI14	Exit if a call word is there
	14 TP	BK3	A	Call word of sentence to A
	15 TJ	HI27	HI23	Is [0 1 0] > (BK3). If so, call word is in v
	16 TV	Q	HI17	Sets up right ref.-list address in v of next instruction
Call word of sentence is in u position	17 TU	BK3	30000	Call word to line following line no. in ref. list
	20 TU	HI12	HI21	Sets up same ref.-list address in next instruction
	21 LQ	30000	25	Transfers call word to v position
	22 MJ	0	HI	
	23 TV	Q	HI24	Puts address into v of next instruction
Call word in BK3 is in v position	24 TP	BK3	30000	Call word to right line in ref. list
	25 MJ	0	HI	
	26 0	IZ	0	
	27 0	1	0	
	CA	HI30		

KI Illegal Line Jump Check Routine -- Generation

After a call word has been obtained for a line number, it should be put in A_u (the rest of A must be cleared) for this test. (The regular call word output of LW gives the call word as desired in A_u .) In Q should be a pseudo-operation indicator. 40 0 0 indicates that a sentence is within a pseudo op. Zero indicates being outside a pseudo Op. Now instruction RJ KI KI1 will activate this routine.

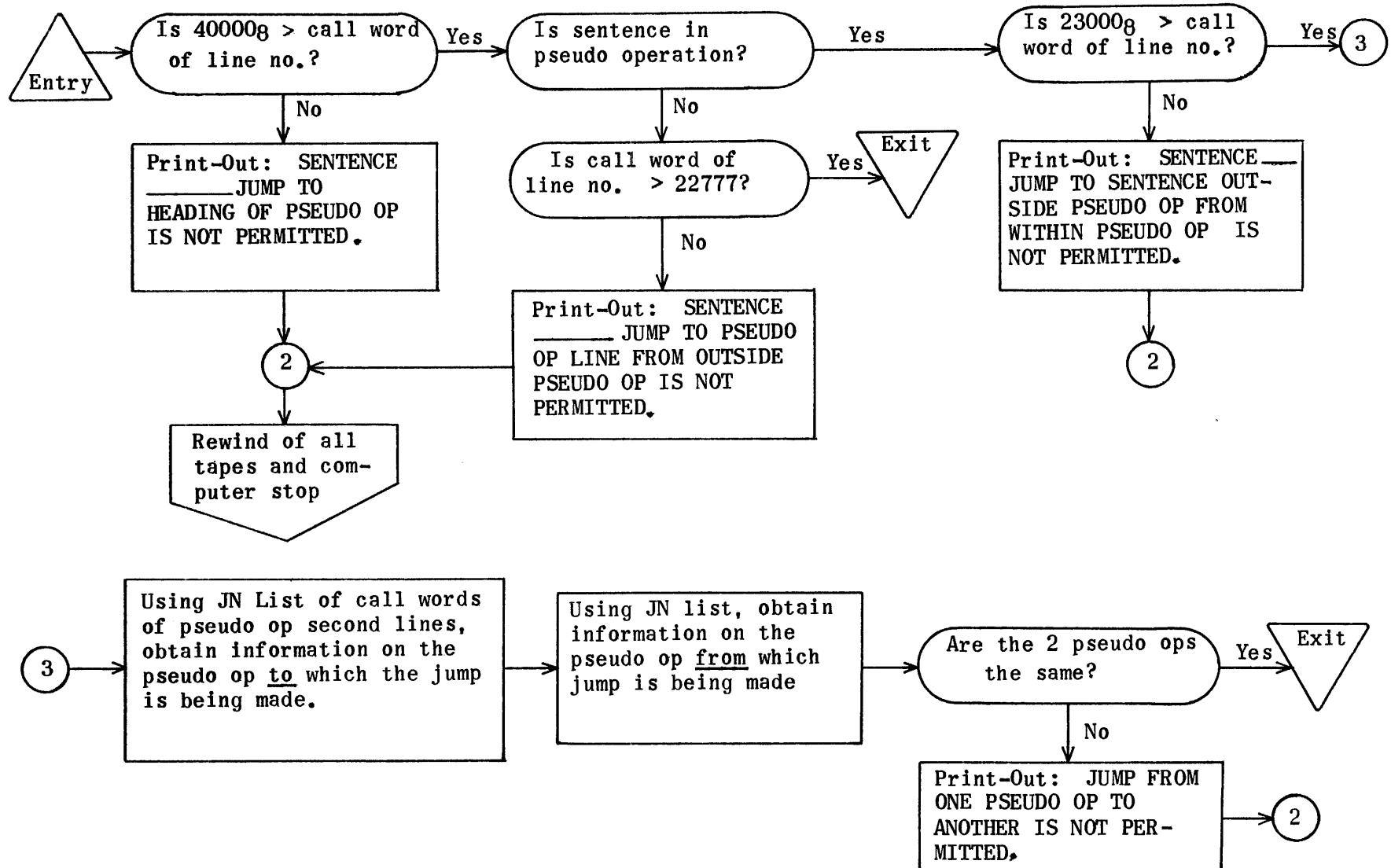
Four error print-outs occur for attempted illegal jumps. All are prefixed by SENTENCE_____

- 1) JUMP TO SENTENCE OUTSIDE PSEUDO OP FROM WITHIN PSEUDO OP IS NOT PERMITTED.
- 2) JUMP TO PSEUDO OP LINE FROM OUTSIDE PSEUDO OP IS NOT PERMITTED.
- 3) JUMP FROM ONE PSEUDO OP TO ANOTHER IS NOT PERMITTED.
- 4) JUMP TO HEADING OF PSEUDO OP IS NOT PERMITTED.

The computer stops after each of these print-outs and all tapes are rewound.

The call word comparisons within the routine are based on the convention that all sentences within a pseudo op, except the first, have a call word less than 23000 and all sentences outside a pseudo op have a call word greater than 22777. The call word of the first line or heading of a pseudo op is 40000 plus. Only the compute instruction may reference this line. A list JN of 2nd line pseudo-op call words is used to check illegal jumps from one pseudo op to another.

KI Illegal Line Jump Check Routine - Flow Chart
 Line No. Call Word Input in A_u -- Pseudo-Op Indicator in Q



KI Illegal Line Jump Check Routine--Generation

Line No. Call Word Input in A_u
Pseudo-Op Indicator in Q

	IA	KI		
	0 MJ	0	30000	Exit
	1 TP	BK1	KI57	Line No. to Print-Out Storage
	2 TJ	KI54	KI10	Is 40000 > call word of line no.?
	3 TP	KI46	UP3	Sentence _____ ΔJump Δ
	4 RJ	UP2	UP	
	5 TP	KI51	UP3	To heading of pseudo-op is not permitted
	6 RJ	UP2	UP	
	7 MJ	0	BQ6	Jump to rewind of servos and computer stop
Call word < 40000 within pseudo-op	10 QJ	KI11	KI14	Is sentence within pseudo-op?
	11 TJ	KI52	KI23	Is 23000 > call word of line no.?
	12 TP	KI45	UP3	Sentence _____ Jump to sentence outside pseudo op from within pseudo op is not permitted
	13 MJ	0	KI6	
Outside pseudo-op	14 TP	A	Q	Line no. call word Q _u
	15 TP	KI53	A	Is line no. call word > 22777?
	16 TJ	Q	KI	
	17 TP	KI46	UP3	Sentence _____ Jump
	20 RJ	UP2	UP	
Within pseudo-op	21 TP	KI47	UP3	To pseudo op sentence from outside pseudo op is not permitted
	22 MJ	0	KI6	
	23 TU	JN	KI24	Setting up u of next instruction for com- parison of line no. call word with list JN of pseudo op 2nd lines
	24 RP	20000	KI41	Purpose of comparison is to find via Q to which pseudo op jump is being made
	25 TJ	JN1	KI26	
	26 TP	Q	KI126	
	27 TU	JN	KI31	Setting up u of instruction to number within JN list plus 20000
	30 SP	BK3	17	Getting call word of sentence to u of A
	31 RP	20000	KI43	Determining via Q in which pseudo op we are operating
	32 TJ	JN1	KI33	
	33 TP	KI126	A	If pseudo op in which we are operating is the same as pseudo op to which we are jumping, Q and KI126 will be equal
	34 EJ	Q	KI	
	35 TP	KI46	UP3	Sentence _____ Jump
	36 RJ	UP2	UP	
	37 TP	KI50	UP3	From 1 pseudo op to another is not permitted
	40 MJ	0	KI6	
	41 TP	JN	KI126	
	42 MJ	0	KI27	
	43 TP	JN	Q	
	44 MJ	0	KI33	
	45 0	KI55	20	Parameter for Sentence _____ Jump to Sentence outside pseudo op from within pseudo op is not permitted

46	0	KI55	4	Parameter for Sentence_____Jump
47	40	KI75	12	Parameter for ___ to pseudo op sentence from
				outside pseudo op is not permitted
50	40	KI107	10	Parameter for ___ from one pseudo op to
				another is not permitted
51	40	KI117	7	Parameter for ___ to heading of pseudo op is
				not permitted
52	0	23000	0	
53	0	22777	0	
54	0	40000	0	
55	65	30506	63050	S E N T E N
56	26	30017	77777	C E Δ
57	0	0	0	
60	01	44674	75201	Δ J U M P Δ
61	66	51016	53050	T O Δ S E N
62	66	30502	63001	T E N C E Δ
63	51	67666	53427	O U T S I D
64	30	01526	53067	E Δ P S E U
65	27	51015	15201	D O Δ O P Δ
66	31	54514	70171	F R O M Δ W
67	34	66333	45001	I T H I N Δ
70	52	65306	72751	P S E U D O
71	01	51520	13465	Δ O P Δ I S
72	01	50516	60101	Δ N O T Δ Δ
73	52	30544	73466	P E R M I T
74	66	30272	27777	T E D .
75	66	51015	26530	T O Δ P S E
76	67	27510	15152	U D O Δ O P
77	01	46345	03001	Δ L I N E Δ
100	31	54514	70151	F R O M Δ O
101	67	66653	42730	U T S I D E
102	01	52653	06727	Δ P S E U D
103	51	01515	20134	O Δ O P Δ I
104	65	01505	16601	S Δ N O T Δ
105	52	30544	73466	P E R M I T
106	66	30272	27777	T E D .
107	01	31545	14701	Δ F R O M Δ
110	51	50300	15265	O N E Δ P S
111	30	67275	10151	E U D O Δ O
112	52	01665	10124	P Δ T O Δ A
113	50	51663	33054	N O T H E R
114	01	34650	15051	Δ I S Δ N O
115	66	01523	05447	T Δ P E R M
116	34	66663	02722	I T T E D .
117	66	51013	33024	T O Δ H E A
120	27	34503	20151	D I N G Δ O
121	31	01526	53067	F Δ P S E U
122	27	51015	15201	D O Δ O P Δ
123	34	65015	05166	I S Δ N O T
124	01	52305	44734	Δ P E R M I
125	66	66302	72277	T T E D .
126	0	0	0	
	CA	KI127		

LS Library List Routine - Generation

The list LN built up on the drum starting at 50046 during Generation is a list of call words of library routines referenced.

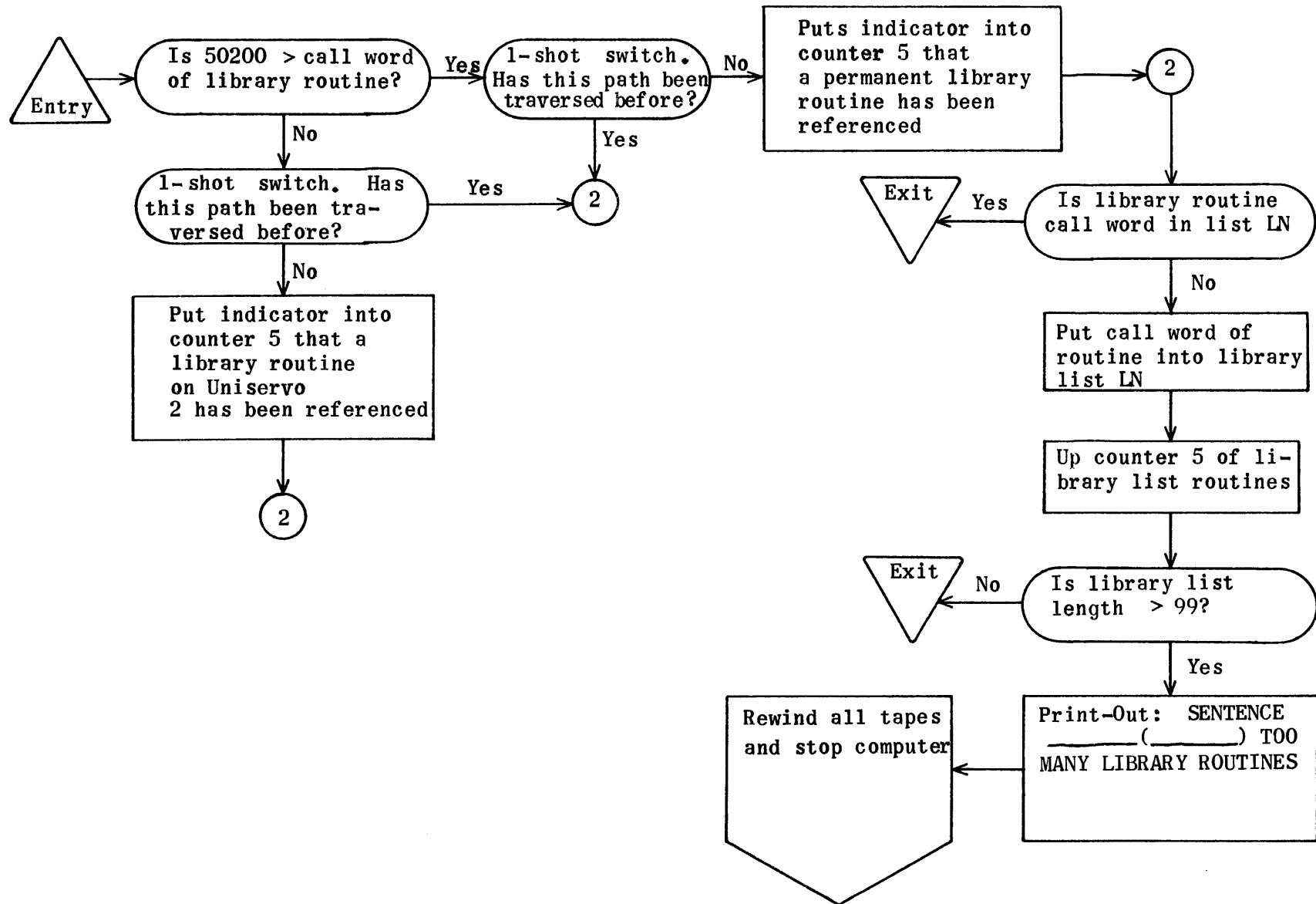
Each call word is stored in the u position. Call words less than 50200 designate Permanent Library Routines. Call words ≥ 50200 designate library routines which have been put on the library tape on Uniservo 2 by the Librarian.

LS Library List Routine distinguishes between the two types of library routines referenced and by means of two one-shot switches puts an indicator into counter 5 the first time each of the types is used. If a Permanent Library Routine has been referenced, 40 is put into the operation portion of counter 5. If a library routine from Uniservo 2 has been referenced, 20 is put into the operation portion of counter 5. Of course, when both types have been referenced, 60 is in the operation portion of 5.

The input to the routine is the call word of the library routine in A_u . A search is made of the list to see if that call word has previously been put in. If it is already there, a quick exit occurs. If it is not there, it is put in the list at the next empty space and counter 5 is increased by 00 00001 00001 to w0 2000n n where w is the indicator mentioned above and n is the cumulative number of call words in the list. Before the exit, a check is made to see if n is less than 100. If it is not, the following print-out occurs: SENTENCE _____ (_____) TOO MANY LIBRARY ROUTINES.

The blanks in this print-out are filled in, respectively, by a line number and the sentence type. Of course, if this improbable print-out occurs, all the tapes are rewound and the computer stops.

Flow Chart for Generation LS Library List Routine



Library List Routine--Generation

Input Call Word in A_n

	IA	LS		
0	MJ	0	30000	Exit
1	TP	A	Q	Call Word → q
2	TJ	LS40	LS7	Is 50200 > call word? If so, a Permanent Library Routine is being referenced
3	RJ	LS3	LS5	Call word is ≥ 50200
4	MJ	0	LS12	
5	CC	5	LS42	Puts indicator that library routine from Uniservo 2 is being referenced into 5
6	MJ	0	LS12	
7	RJ	LS7	LS11	
10	MJ	0	LS12	
11	CC	5	LS41	Puts indicator that a Permanent Library Routine has been referenced into 5
12	TP	Q	A	Call Word → A
13	TU	5	LS14	Sets up u of repeat so that library list may be compared
14	RP	20000	LS16	} If library routine is already in list, go to exit
15	EJ	LN	LS	
16	TV	5	LS20	} Putting library routine call word in list LN when it has not been found there
17	RA	LS20	LS37	
20	TP	Q	30000	
21	RA	5	CW27	Adding 0 1 1 to counter 5
22	TP	LW21	Q	Putting mask 07 77777 77777 into q
23	QT	5	A	Masking indicator bits out of 5 for transfer to A as 0 2000n n
24	TJ	LS36	LS	Is 00 20100 00100 > A. Thus n, number of routines, must be < 100 ₁₀
25	RJ	WA	WA2	} Print-Out: SENTENCE _____ (_____) TOO MANY LIBRARY ROUTINES. Rewinds tapes and stops computer
26	TP	LS30	UP3	
27	MJ	0	CW15	
30	40	LS31	5	Parameter for print-out
31	66	51510	14724	T O O Δ M A
32	50	73014	63425	N Y Δ L I B
33	54	24547	30154	R A R Y Δ R
34	51	67663	45030	O U T I N E
35	65	77777	77777	S
36	0	CI20000	CI	Threshold for number of library routines. CI = 100 ₁₀
37	0	0	LN	Address of 1st line of library list
40	0	50200	0	Call word of 1st non-permanent library routine
41	40	0	0	Indicator that a Permanent Library Routine has been referenced
42	20	0	0	Indicator that Uniservo 2 Library Routine has been referenced
	CA	LS43		

Routine for Conversion of Excess-Three to Flex Code

Three regions -- VX (154₈ addresses), FC (100₈ addresses), and VE (6 addresses) -- comprise the total 262₈ needed for this routine.

The parameter input line, VX4, holds the address of the first line of input in the u portion. In the v portion is given the number of words of input. If every other line starting with the first address is to be selected as input, 40 is put in the operation code of VX4. If every line following the first address is to be used, 00 is put in the operation code.

In the u portion of VX3, the desired address of the first line of output should be supplied. V of VX3 must be zero before entering routine. The routine when operated by instruction RJ VX2 VX will supply in the v portion of VX3 the number of words of output. The Flex code output is packed from the left with the necessary added shift-up or shift-down Flex codes.

Every excess-three character from 00 to 76 has a corresponding Flex code or set of Flex codes assigned to it as shown on the accompanying table. 77, though assigned Flex code 77, is not stored in the output.

Because of the addition of shift-up and shift-down Flex codes and the fact that some excess-three characters are represented by more than one Flex code, greater storage space must be allowed for the output than is needed for the input.

In the table the first digit of each Flex code representation is not stored in the output. It is used only to distinguish between lower-case and capital letters. A 4 in the first position indicates lower-case; 0 indicates capital.

A basic assumption in this routine is that the characters on the Flexowriter are in the standardized form shown in Table I on page 15 of PX38. (Input and Output Systems - Univac Scientific). Any changes made on the Flexowriter keyboard from this standard will cause erroneous results in the print and type instructions of UNICODE.

The keyboard of the Unityper is assumed to be in accordance with the changes proposed on the two pages following the code table. See Chapter 8, Tape Preparation, of the UNICODE Manual for an explanation of a system of bypassing this requirement of an altered keyboard.

Excess Three to Flex Code Table

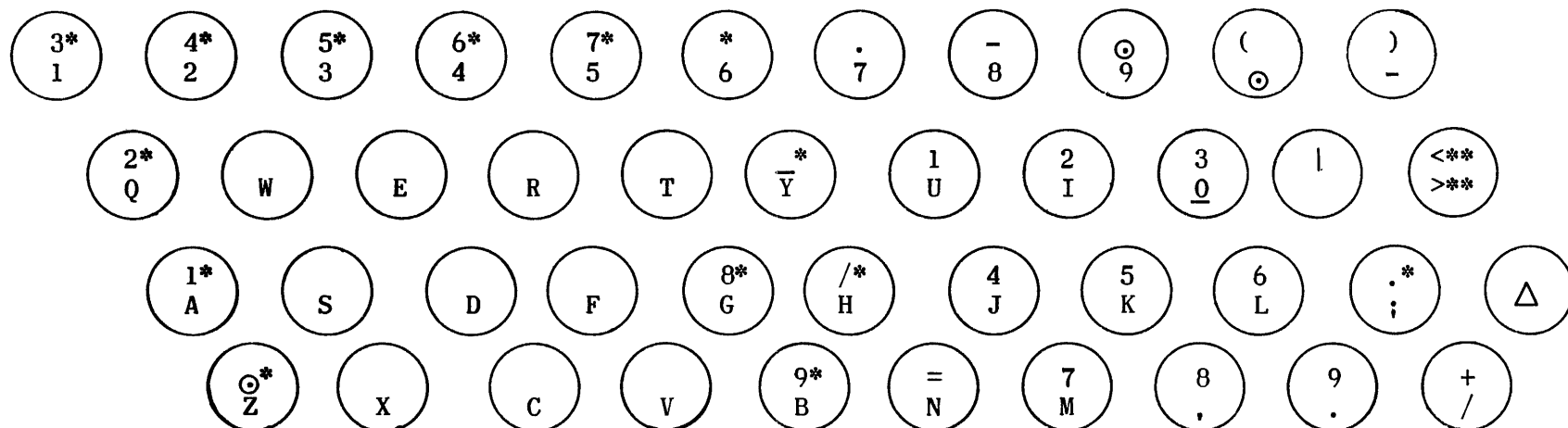
<u>Excess Three Character Where Different</u>	<u>Excess Three Code</u>	<u>Flex Code</u>	<u>Char- acter</u>
	0	056	-
	1	004	Space
	2	456	—
	3	437	0
	4	452	1
	5	474	2
	6	470	3
	7	464	4
	10	462	5
	11	466	6
	12	472	7
	13	460	8
	14	433	9
/	15	054	/
>	16	413 01 12	gtr
	17	046	(
	20	070	3
	21	446	,
	22	442	.
;	23	446	,
	24	030	A
	25	023	B
	26	016	C
	27	022	D
	30	020	E
	31	026	F
	32	013	G
	33	005	H
	34	014	I
	35	062	5
	36	060	8
<	37	411 24 12	lsr

Excess Three to Flex Code Table (Cont.)

<u>Excess Three Character Where Different</u>	<u>Excess Three Code</u>	<u>Flex Code</u>	<u>Char- acter</u>
	40	074	2
	41	064	4
	42	450	
	43	042	)
	44	032	J
	45	036	K
	46	011	L
	47	007	M
	50	006	N
	51	003	O
	52	015	P
	53	035	Q
	54	012	R
	55	066	6
*	56	427 61 54	*
	57	033	9
	60	037	0
	61	052	1
	62	044	.
	63	454	+
	64	054	/
	65	024	S
	66	001	T
	67	034	U
	70	017	V
	71	031	W
	72	027	X
	73	025	Y
	74	021	Z
	75	072	7
	76	444	=
	77	077	Ignore

Proposed Unityper Changes

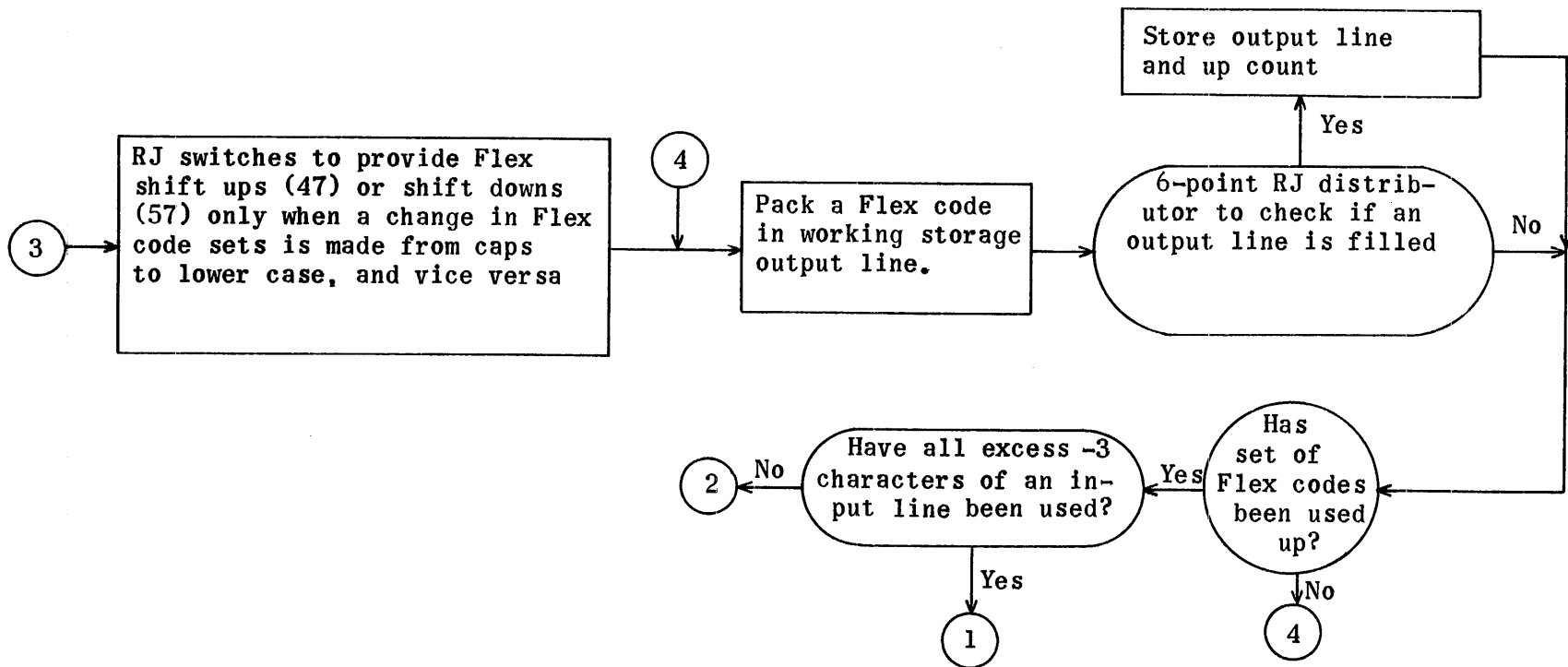
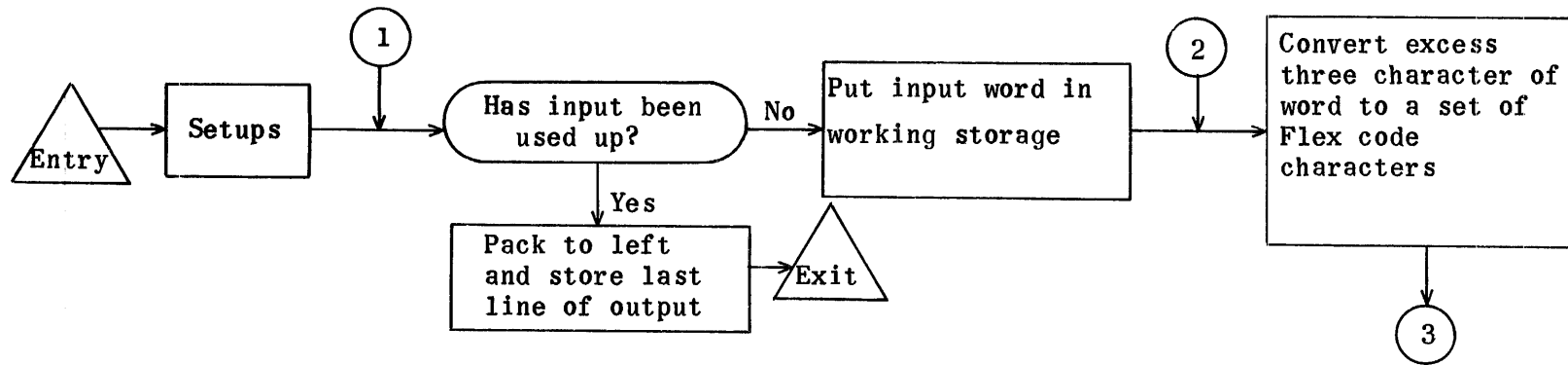
PRESENT			PROPOSED ADDITIONS	
<u>PULSE CODE</u>	<u>NAME</u>	<u>SYMBOL</u>	<u>NAME</u>	<u>SYMBOL</u>
00 0000	Ignore	$\bar{1}$	Superscript minus	-
00 1101	Apostrophe	'	Superscript slash	/
00 1110	Ampersand	&	Greater than	>
01 0000	Carriage return	$\bar{r}$	Superscript three	3
01 1101	Number	#	Superscript five	5
01 1110	Cent	$\bar{c}$	Superscript eight	8
01 1111	At one	@	Less than	<
10 0000	Tab	$\bar{t}$	Superscript two	2
10 0001	Quotation	"	Superscript four	4
10 1101	Dollar	\$	Superscript six	6
10 1111	Question	?	Superscript nine	9
11 0000	Sigma	Σ	Superscript zero	⊙
11 0001	Beta	β	Superscript one	1
11 0010	Colon	:	Superscript decimal point	.
11 1101	Per cent	%	Superscript Seven	7



	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	$\overline{-}^*$ $\overline{1}$	Δ	-	$\odot$	1	2	3	4	5	6	7	8	9	$\overline{/}^*$ $\overline{\epsilon}$	$\overline{>}^{***}$ $\overline{\epsilon}$	(
01	3^* $\overline{1}$	,	.	;	A	B	C	D	E	F	G	H	I	5^* $\overline{\#}$	8^* $\overline{\phi}$	$<^{***}$ $\overline{}$
10	2^* $\overline{1}$	4^* $\overline{''}$		)	J	K	L	M	N	$\underline{O}$	P	Q	R	6^* $\overline{\$}$	$*$	9^* $\overline{?}$
11	0^* $\overline{\Sigma}$	1^* $\overline{\beta}$	$\cdot^*$ $\overline{:}$	+	/	S	T	U	V	W	X	Y	Z	7^* $\overline{\%}$	=	$\overline{}$

* Superscript symbols
 ** New symbols

Excess Three To Flex Code Subroutine Flow Chart



Excess-Three to Flex Code Subroutine

		Must be set at zero before using subroutine
VX3 =	Address of 1st line of output	No. of words of output
VX4 =	00 if every line, 40 if every other line	Address of 1st line of input No. of words of input

IA	VX		
0 MJ	0	VX5	Entry
1 RJ	0	0	
2 MJ	0	30000	Exit
3 0	0	0	
4 0	0	0	
5 TP	VX152	VE5	Set up input line character index
6 RP	10005	VX10	Set up n-point distributor
7 TV	VX141	VX65	to beginning values
10 TV	VX140	VX72	
11 TV	VX136	VX47	Set up to insure that the output will start
12 TV	VX137	VX54	with either a shift up or shift down
13 TP	VX150	VE	47 57
14 TP	VX150	VE1	Clear 2 addresses of working storage
15 TP	VX4	Q	Parameter VX4 → q
16 QJ	VX17	VX21	Test to see if 40 is in operation code of q
17 TP	VX153	VE2	[0 2 0] → VE2 to enable input address to be incremented so every other line is taken
20 MJ	0	VX22	
21 TP	VX147	VE2	[0 1 0] → VE2
22 TV	VX4	VE	Number of words of input → VE
23 TU	VX4	VX31	Address of 1st input line → u of VX31
24 TU	VX3	VE3	Address of 1st output line → v of VE3
25 LQ	VE3	25	
26 TV	VE3	VX102	Address of 1st output line → v of VX102
27 IJ	VE	VX31	Index jump for number words of input
30 MJ	0	VX121	
31 TP	30000	VE4	Line of input → VE4
32 LQ	VE4	17	Initial shift of input line so 1st character will be in proper position
33 LQ	VE4	6	Puts character XY to be masked out in right side of u of VE4
34 TP	VX142	A	[SP fc 0] + [0 xy 0] → [SP fc + xy 0]
35 QA	VX143	VX36	→ VX36
36 0	0	0	
37 EJ	VX151	VX115	If Flex code picked up is a 77, storing part of routine is bypassed
40 LA	A	44	A _R → A _L

41	SJ	VX42	VX45	Test if character is lower case. Negative means lower case
42	LA	A	3	} Left shift and split subtract to eliminate 4 from Flex code representation
43	SS	A	0	
44	MJ	0	VX47	Jump to switch
45	LA	A	3	Left shift to properly position Flex code at far left of AL
46	MJ	0	VX54	Jump to switch
47	RJ	VX54	VX55	Entry Switch to provide shift ups and shift downs in output only when a change is made from lower case to caps or vice versa
50	RJ	VX54	VX52	
51	MJ	0	VX73	↑ Exit, No action.
52	SA	VX145	0	
53	MJ	0	VX60	↑ Exit.
54	RJ	VX47	VX50	Entry, ↑
55	RJ	VX47	VX57	
56	MJ	0	VX73	↓ Exit - no action.
57	SA	VX144	0	↓ Exit.
60	LQ	VE1	6	} When Switch has provided a 47 or 57 to be inserted in VE1, these three steps pack the Flex code into the output line
61	SA	VE1	0	
62	TP	A	VE1	
63	SS	A	0	Clears AR
64	RJ	VX64	VX65	} N-Point Distributor used to determine when each output line has received its six Flex code characters
65	RJ	VX64	VX101	
66	RJ	VX64	VX101	
67	RJ	VX64	VX101	
70	RJ	VX64	VX101	
71	RJ	VX64	VX101	
72	MJ	0	VX102	
73	LA	A	6	} A no-action exit from the Switch causes subroutine to transfer the Flex code character to the output, following which AR is cleared, and a jump is made to the n-point distributor
74	LQ	VE1	6	
75	SA	VE1	0	
76	TP	A	VE1	
77	SS	A	0	
100	MJ	0	VX64	
101	ZJ	VX73	VX115	When a = 0?, an excess-three character has had its Flex code set used up
102	TP	VE1	30000	Completed output line is stored
103	SA	VX102	0	} v of instruction VX102 is increased by 1
104	SA	VX146	0	
105	TP	A	VX102	
106	SS	A	0	Clears AR
107	TP	VX150	VE1	Output line is cleared
110	SA	VX3	0	} Number of words of output is increased by 1
111	SA	VX146	0	
112	TP	A	VX3	
113	SS	A	0	
114	MJ	0	VX101	
115	IJ	VE5	VX33	Index jump to determine when a line of input has been exhausted
116	TP	VX152	VE5	Index is set up for next input line

117	RA	VX31	VE2	VX31 is increased to take next line of input
120	MJ	0	VX27	Jump back to input-word index
121	TP	VE1	A	} Test to see if output line is empty
122	ZJ	VX123	VX2	
123	TV	VX134	VX72	} Setting up n-point distributor to handle last line of output
124	RP	10005	VX126	
125	TV	VX135	VX65	
126	LQ	VE1	6	} Loop of left shifting until elements in VE1 are in leftmost position
127	MJ	0	VX64	
130	TP	VX102	VX131	} Putting last line into output
131	0	0	0	
132	RA	VX3	VX146	Completing count of output lines
133	MJ	0	VX2	Handling final line of output
134	0	0	VX130	
135	0	0	VX126	
136	0	0	VX55	
137	0	0	VX50	
140	0	0	VX102	
141	0	0	VX101	
142	SP	FC	0	
143	0	77	0	
144	0	0	57	
145	0	0	47	
146	0	0	1	
147	0	1	0	
150	0	0	0	
151	07	70000	0	
152	0	0	5	
153	0	2	0	Constant (used also by Read Generator)
	CA	VX154		

Region FC of Flex codes is not shown here because it has previously been given in the section explaining the tables used in the translation phase.

EXCESS-THREE TO FLEX CODE USES OF WORKING STORAGE VE

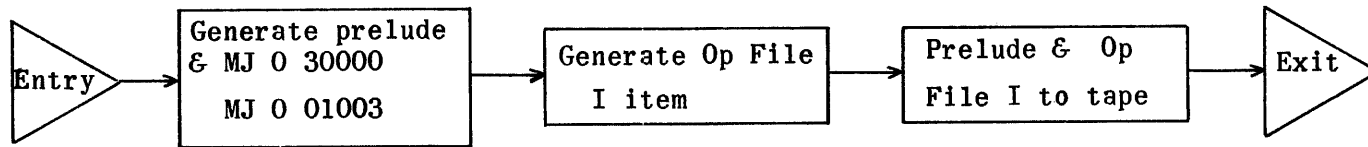
- 0 Index for number of words in input.
- 1 For assembling output characters.
- 2 To increment address from which input lines are taken.
- 3 To assemble first address of output.
- 4 To store line of excess-three characters while working on it.
- 5 Character index per line of input.

3. GENERATORS

3. GENERATORS

The generators are described in the order in which they are loaded on the drum. See Generation Subroutine Regions in the preceding section for this order.

Start Generator Flow Chart



START GENERATOR SUBROUTINE

RE PT2512
RE PL5360
RE FL2542

RE OP1047 Generation subroutine
RE WL2242 Input to generator

	IA	PT		
0	MJ	0	30000	Exit
1	RP	30010	PT3	} Generate Prelude format
2	TP	PT16	PL	
3	SP	WL3	17	} Sentence call word to prelude
4	TU	A	PL	
5	TP	WL1	PL5	Standard line no.
6	RP	30002	PT10	} Generate Op File I format
7	TP	PT26	FL	
10	SP	WL3	17	} Sentence call word to Op File I
11	TU	A	FL	
12	TP	PT15	OP1	} Parameters to Op (→ tape)
13	RJ	OP	OP2	
14	MJ	0	PT	Exit
15	0	PL	FL	} Parameters
16	0	0	10	
17	0	0	2	} Prelude format
20	0	0	0	
21	0	0	0	
22	0	0	0	
23	0	0	0	
24	MJ	0	30000	
25	MJ	0	01003	} Op File I format
26	0	[0]	2	
27	0	0	2	
	CA	PT30		

Jump Generation Routine

The string-out input to the Jump Generator consists of 6 lines. The first four conform to the standard format: number of words in string-out in v part of first line, line number in second line, title in third line, and call word of sentence in v part of fourth line. The fifth line contains the line number of the line to which the jump is to be executed. The sixth line contains 40 0 0 if the sentence is within a pseudo op; if not, it is cleared.

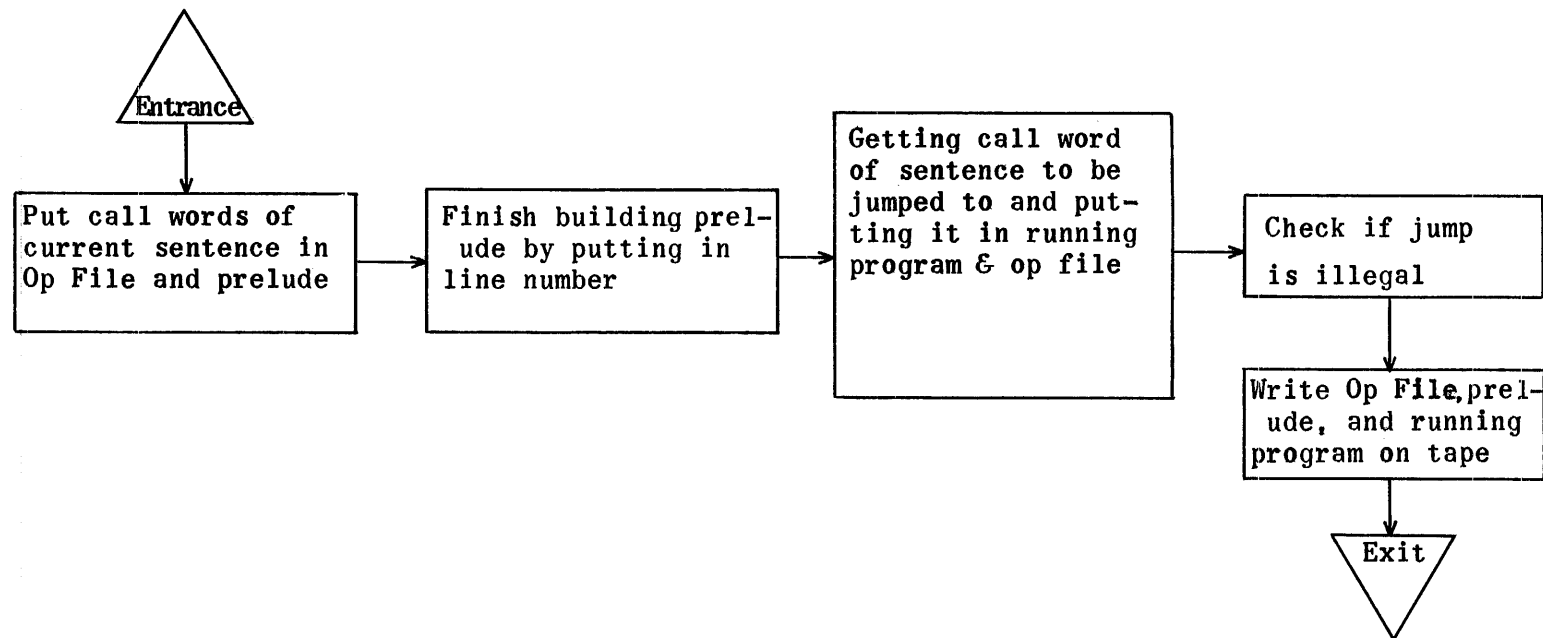
The Jump Generator builds up the Op File, prelude, and running program from this data. The first line of the generated running program is a dummy line, MJ 0 30000, put in to conform to the standard format of all generated running programs. In the v part of the second running program line is put the call word of the sentence to which the jump is to be made. Following this line is the ten line, 10 0 1, put in to ensure that the jump will be made to the second line of running program generated for the sentence. The first line of any running program has been standardized as the exit line of that segment of coding.

Also put in the Op File is the call word of the line number to which the jump is to be made.

A separate routine, KI, is used to determine if the jump is permitted. Jumps may be made within a pseudo operation only to other sentences that are also within the same pseudo op. Jumps may not be made from outside a pseudo op to sentences within a pseudo op. Jumps may not be made to the first line or initial heading of a pseudo op.

As with other generation routines, the final instruction is the reference to the op routine which stores the Op File and writes the generated routine on tape.

Jump Generation Flow Chart



Jump Generation

RE JU2512

Generation Subroutine regions are also needed to assemble
this tape

	IA	JU		
0	MJ	0	30000	
1	SP	BK3	17	} Call word of sentence to 1st line of Op File in u position
2	TU	A	JU17	
3	TU	A	JU22	Call word of sentence to 1st line of prelude in u position
4	TP	BK1	JU27	Line no. to prelude
5	TP	BK4	A	} Getting call word of line number to be jumped to from ref. list
6	RJ	LW	LW1	
7	TV	Q	JU31	Call word to v of running prog. jump line
10	TU	A	JU21	Call word to Op File cross reference
11	TP	BK5	Q	} Puts pseudo op indicator into Q Check to see if jump is permitted
12	RJ	KI	KI1	
13	TP	JU16	OP1	} Writes Op File & prelude & running program on tape
14	RJ	OP	OP2	
15	MJ	0	JU	Exit
16	0	JU22	JU17	Parameter for writing Op File & generated routine on tape
17	0	30000	3	} Op File "dummy" lines
20	0	0	2	
21	0	30000	0	
22	0	30000	11	
23	0	0	3	} Prelude "dummy" lines
24	0	0	0	
25	0	0	0	
26	0	0	0	
27	0	0	0	
30	MJ	0	30000	} Running program "dummy" lines
31	MJ	0	30000	
32	10	0	1	
	CA	JU33		

If Generation

The "if" string-out output VN - VN33, as described in the write-up for same, becomes the input BK - BK33 of this routine. KB, the control routine, generates the coding needed for obtaining the variables or constants to be compared. If X or Y is a subscripted variable, a relative constant call word (10000 or 10001) is assigned to it in the coding. The regular call word of X or Y is stored following the generated routine. The processor later replaces the 10000 call word with the address of this stored call word. The stored call word in turn is replaced by the address of the first line of the array of the variable. This obviates the necessity of the routine keeping track of where the call word of the variable has been put and putting this address relative to 1000 in wherever a reference is made to the variable.

When X or Y is a pseudo op subscripted variable, this indirect method is no longer necessary. Because the pseudo op call word is a call word of a call word, the address is obtained by using it directly.

KB uses KC routine to generate the coding needed for getting the right address of an item in an array of values of a subscripted variable. The number of subscripts of a subscripted variable must never be zero. The call word of the first line of an array is obtained by using the subscripts but giving them zero value. Operation of KC is based upon the following formula:

$$1 [X(I,J,K,L)] = 1 [X(0,0,0,0)] + [I \cdot M_I + J \cdot M_J + K \cdot M_K + L] \text{ Modulus } Z$$

Z, the modulus, is the product of the number of values assignable to each subscript in the dimension statement. The subscripts vary from right to left. $M_K = L_D$ where L_D means the number of values retained for L as shown in dimension statement, viz: DIM X(I_D, J_D, K_D, L_D). $M_J = K_D \cdot L_D$. $M_I = J_D \cdot K_D \cdot L_D$. $Z = I_D \cdot J_D \cdot K_D \cdot L_D$. $1 [X(0,0,0,0)]$ means the address of the variable whose subscripts are zero. Thus, it is the address of the first line in an array. It is located by use of the relative constants 10000 and 10001, as explained above.

Thus the program divides the sum of the products and the value of L by the modulus Z. The remainder in A is added to the address represented by

1 [X(0,0,0,0)] to give the final address of 1 [X(I,J,K,L)]. This address is transferred to A_u where it is used in a TU instruction to set up a TP instruction that will put the variable into a temporary storage location, 60000 for X and 60001 for Y. Following instructions give X and Y absolute or negative values, if desired. The processor changes 60000 and 60001 to assigned addresses.

Coding for the tests needed is now generated. The threshold jumps that make the tests go to manual jumps within the coding. This is done to facilitate segmentation. In the v addresses of these manual jumps is put the call word of the line number to which the main jump is to be made. Following the manual jumps, "10 lines" are put to ensure that the jump to coding of a sentence goes to the second line of the coding, the first line being an exit.

Subroutine KD generates the test coding for the situation in which three line numbers of possible jumps have been supplied in the string-out input. Subroutine KF takes care of the case in which there are two line numbers and one test which divides X and Y into two exhaustive categories. Also taken care of in this routine is the situation in which two tests are requested but no third line number is given for a final jump in case both fail. In the latter instance, of course, the jump is made back to the beginning address, 1000, of the coding. From here, a jump is made to the second line of the coding for the subsequent sentence.

Subroutine KK takes care of the case when just one test is made in an "if" sentence. In addition, a segment of it handles the details of termination and references the op routine to get the Op File and generated routine written on tape.

KH consists of a group of subroutines which assist in all the functions described above. The Op File is stored and built up here. In its initial form KH57 holds 0 30000 2 in the first line of Op File. This line is not preset at the start of a use of the Generation Routine. Hence, it becomes necessary to transfer the generation coding afresh either from drum or tape prior to each use of it. The v of this instruction is added to as lines are filled in the Op File. One subroutine of KH keeps a count of the line of the running program relative to 1000 as it loads output region GL. This count, called

100x on "if" generation annotated coding, is used to compute jump addresses and to calculate total lines at end of "if" generation.

Print-outs that may occur during "if" generation are those which accompany use of KI, the illegal line number check routine. (See separate write-up on KI.) In a KH subroutine, the call word of the line number is obtained from LW, sent to KI for checking, and then put in the generated routine. (See reference list write-up for print-out that may occur in LW.)

KA holds constants used and KE holds dummy instructions that are modified and assembled to make up the generated routine.

Attached is an explanation of the use of temporary storage (addresses GL100 - 112), a descriptive format of first eight lines of GL output, and samples of generated coding.

Initial Lines of Output

GL					
Prelude	0	0	u	v	Call word of sentence to u of GL. $v = 100x - 1000 + 6$. Thus v is no. lines generated routine incl. "10 lines" and prelude.
	1	0	0	v	$v = 100x - 1000$, number of lines subject to address modification.
	2	0	20000	0	The 2 in 3rd digit from left is a count of temporaries used. Count of relative constants is in 4th digit from right.
	3	0	0	0	Number of inputs is zero.
	4	0	0	0	Number of outputs is zero
Exit line of running program	5	Line Number			Line number of sentence
	6	MJ	0	v	$v = 100x - \text{number of "10" lines} + 3$. This gives a jump to 2nd line of coding of subsequent sent.
	7				

Temporary Storage for If Generation

GL				
100				Where each new line of running program is assembled before insertion in output GL.
101	0	0	100x	Ordinal number of running program line relative to 1000 ("10 lines" are included.) Always has next line to be used.
102	0	0	v	Used in computing jump lines.
103	0	0	v	Number of relative constants used. (10000 and/or 10001)
104	0	0	v	Call word of subscripted variable equated to 10000.
105	0	0	v	Call word of subscripted variable equated to 10001.
106	0	0	v	Number of "10 lines"
107	0	u	v	(BK15)
110	0	u	v	(BK16)
111	0	u	v	(BK17)
112	0	0	v	(BK20)

Holds, alternatively, the values in BK15 - 20 on X and BK25 - 30 on Y when these are subscripted variables.

IF GENERATION - SAMPLE CODING

*Example 1

NOT in Pseudo-Op List

< First test

> Second test

For both X and Y: Sign complement desired

Absolute value desired

X and Y subscripted variables with 4 subscripts each

GL					
	0	0	u	50	u = Call word of "IF" sentence line no.
	1	0	0	42	
	2	0	20000	02000	
	3	0	0	0	
	4	0	0	0	
	5	Line Number			
1000	6	MJ	0	1042	
1001	7	SP	u	0	u = Call word of L subscript u = Call word of M_K = K multiplier v = Call word of K subscript u = Call word of M_J v = Call word of J u = Call word of M_I v = Call word of I u = Call word of Z, modulus u = Call word of Z, modulus
1002	10	MA	u	v	
1003	11	MA	u	v	
1004	12	MA	u	v	
1005	13	TJ	u	1007	{ Braced portion is coding to obtain X in proper form in 60000.
1006	14	DV	u	Q	
1007	15	SA	10000	17	
1010	16	TU	A	1011	
1011	17	TP	30000	60000	Coding to put Y in proper form in 60001. Explanations of u and v are same as for X above.
1012	20	TM	60000	60000	
1013	21	TN	60000	60000	
1014	22	SP	u	0	
1015	23	MA	u	v	X → A Is X < Y? Is X > Y?
1016	24	MA	u	v	
1017	25	MA	u	v	
1020	26	TJ	u	1022	
1021	27	DV	u	Q	
1022	30	SA	10001	17	
1023	31	TU	A	1024	
1024	32	TP	30000	60001	
1025	33	TM	60001	60001	
1026	34	TN	60001	60001	
1027	35	TP	60000	A	
1030	36	TJ	60001	1033	
1031	37	EJ	60001	1034	

*The sentence from which this program was developed read as follows: If $-|X(i,j,k,l)| < -|Y(i,j,k,l)|$ jump to sentence 42, if $-|X(i,j,k,l)| > -|Y(i,j,k,l)|$ jump to sentence 54, if $-|X(i,j,k,l)| = -|Y(i,j,k,l)|$ jump to sentence 4Δ.

1032	40	MJ	0	v	v = Call word of line number of 2nd test. X > Y.
	41	10	0	1	
1033	42	MJ	0	v	v = Call word of line number of 1st test. X < Y.
	43	10	0	1	
1034	44	MJ	0	v	v = Call word of 3rd line number. X = Y.
	45	10	0	1	
1035	46	0	0	v	10000 equated to 1035 by processor. v = Call
					word of 1st line of X array.
1036	47	0	0	v	10001 equated to 1036 by processor. v = Call
					word of 1st line of Y array.
1037					Temporary storage. 60000 equated to this address
					by processor.
1040					Temporary storage. 60001 equated to this address
					by processor.
1041					1st line of program of next sentence (exit).
1042					2nd line of program of next sentence (entrance).

Op File

KH

57	0	u	7	u is call word of "if" line number.
60	0	0	41	100x - 1000 - "10 lines" + 2 temporaries.
61	0	u	0	u is call word of X.
62	0	u	0	u is call word of Y
63	0	u	0	u is call word of 2nd test line number.
64	0	u	0	u is call word of 1st test line number.
65	0	u	0	u is call word of 3rd line number.

The count kept in GL101 of the ordinal program number line relative to 1000 is not the same as that shown above since it includes "10 lines". Proper deductions are made for this inclusion in computing the totals near the end of the program.

*Example 2

In Pseudo Op List with a Pseudo Op subscripted variable of 3 subscripts for X
& 2 for Y

< = Single test

Only one line number (one for above test)

Sign Complement of X desired

GL					
	0	0	u	33	u = Call word of "If" sentence line number
	1	0	0	25	
	2	0	20000	0	
	3	0	0	0	
	4	0	0	0	
	5	Line Number			
1000	6	MJ	0	1027	X > Y
1001	7	SP	u	0	} u = Call word of K subscript u = Call word of M _J v = Call word of J u = Call word of M _I v = Call word of I
1002	10	MA	u	v	
1003	11	MA	u	v	
1004	12	TJ	u	1006	
1005	13	DV	u	Q	u = Call word of Z, modulus
1006	14	SA	u	17	u = Call word of Z, modulus
1007	15	TU	A	1010	u = Pseudo op call word of call word of X
1010	16	TP	30000	60000	} —This puts X into 60000
1011	17	TN	60000	60000	
1012	20	SP	u	0	} u = Call word of J subscript u = Call word of M _I v = Call word of I u = Call word of Z, modulus
1013	21	MA	u	v	
1014	22	TJ	u	1016	
1015	23	DV	u	Q	
1016	24	SA	u	17	} u = Pseudo op call word of call word of Y
1017	25	TU	A	1020	
1020	26	TP	30000	60001	—This puts Y into 60001
1021	27	TP	60001	A	
1022	30	TJ	60000	1000	Is X > Y?
1023	31	MJ	0	v	v = Call word of line no. of 1st test. X ≤ Y.
	32	10	0	1	
1024					Temp storage 60000 equated to this address by processor.
1025					Temp storage 60001 equated to this address by processor.

Op File

KH

57	0	u	3	u = Call word of "If" sentence line number.
60	0	0	26	Running prog. length including 2 temporaries
61	0	u	0	u = Call word of line number of test

*If - X(i,j,k) < = Y(i,j) jump to sentence 37Δ.

*Example 3

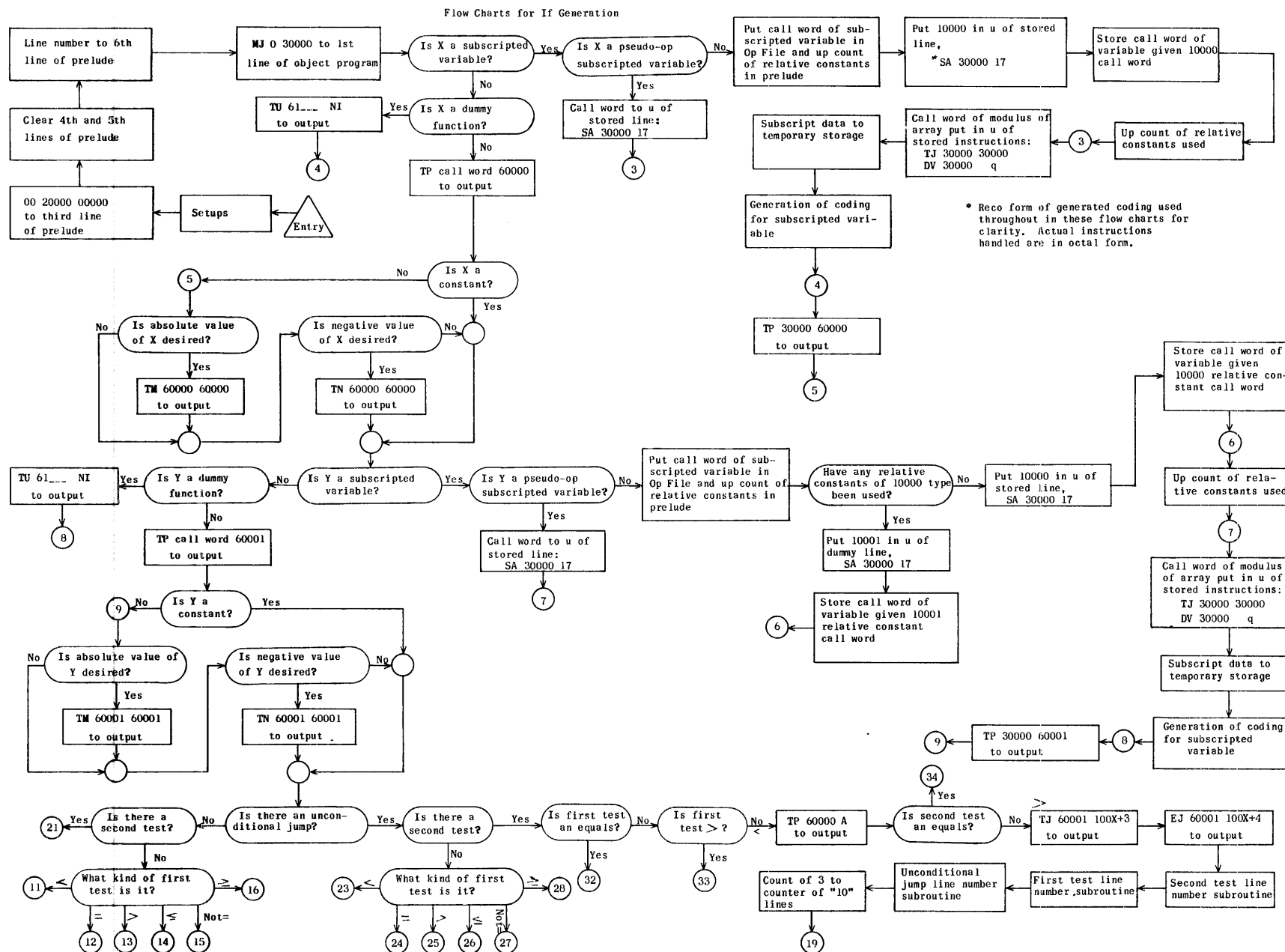
< First test
 > Second test
 No 3rd line number
 X and Y are not subscripted variables.

GL					
	0	0	u	20	u = Call word of "if" sentence line number
	1			12	
	2	0	20000	0	
	3	0	0	0	
	4	0	0	0	
	5	Line Number			
1000	6	MJ	0	1013	X = Y
1001	7	TP	u	60000	u = Call word of X
1002	10	TP	u	60001	u = Call word of Y
1003	11	TP	60000	A	
1004	12	TJ	60001	1007	Is X < Y?
1005	13	EJ	60001	1000	Is X > Y?
1006	14	MJ	0	v	v = Call word of line number of 2nd test. X > Y.
	15	10	0	1	
1007	16	MJ	0	v	v = Call word of line number of 1st test. X < Y.
	17	10	0	1	
1010					Temporary (60000)
1011					Temporary (60001)
1012					1st line of next sentence coding
1013					2nd line of generated program of next sentence

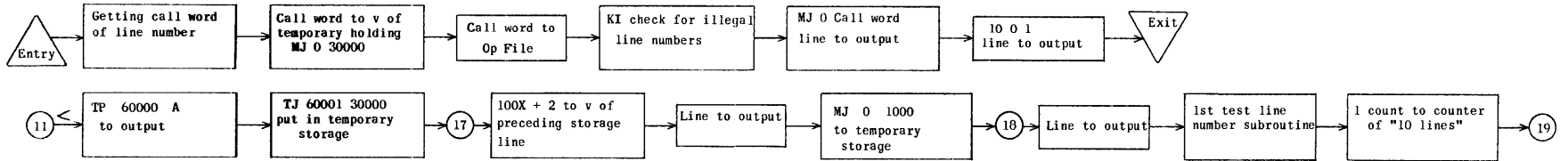
Op File

KH				
57	0	u	4	u = Call word of "if" sentence line number
60	0	0	12	
61	0	u	0	u = Call word of 2nd test line number
62	0	u	0	u = Call word of 1st test line number

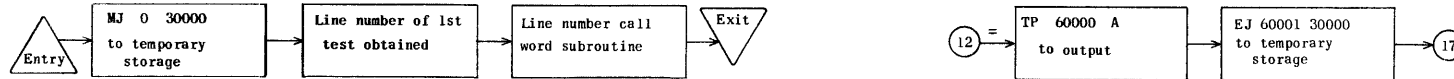
*If X < Y jump to sentence 67, if X > Y jump to sentence 33Δ.



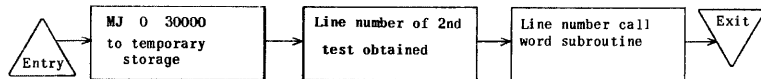
Line Number Call Word Subroutine



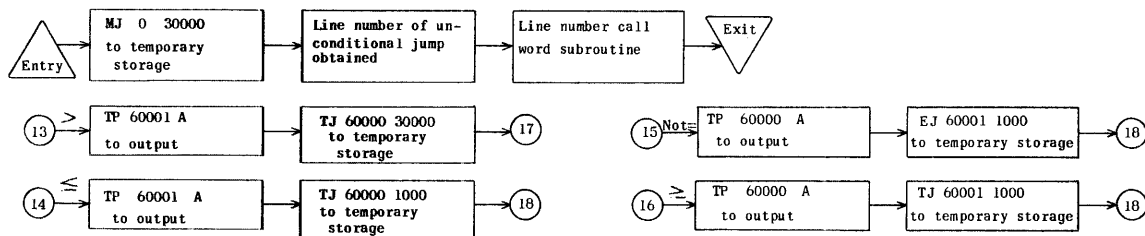
First Test Line Number Subroutine



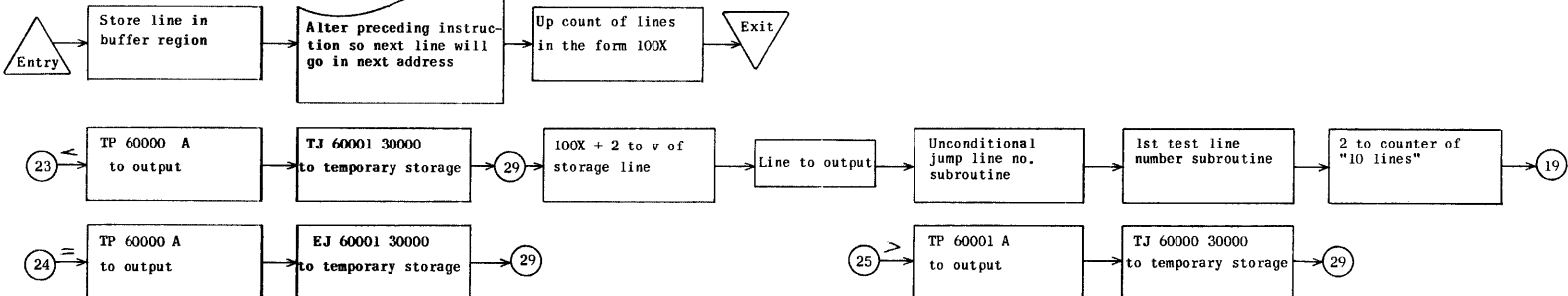
Second Test Line Number Subroutine



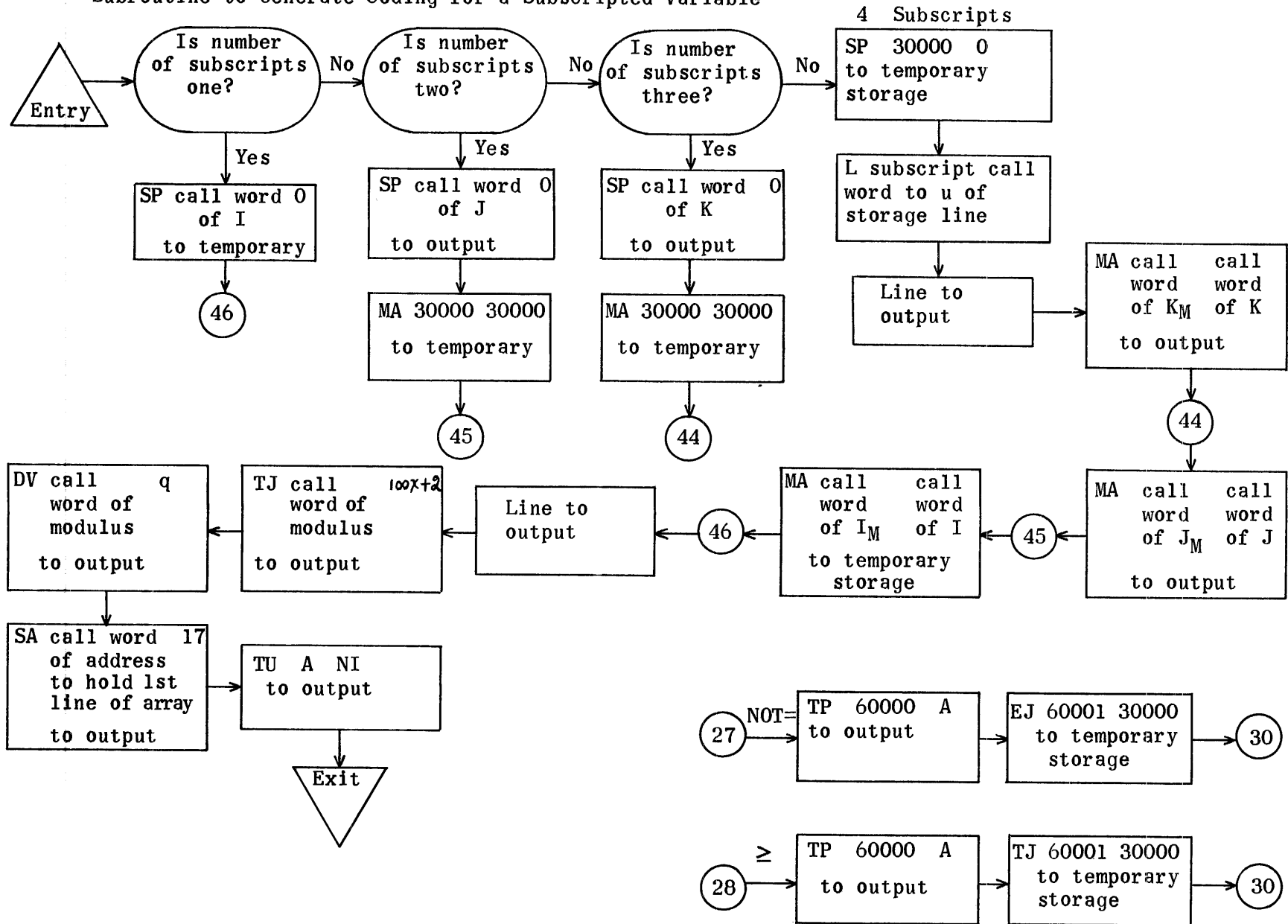
Unconditional Jump Line Number Subroutine



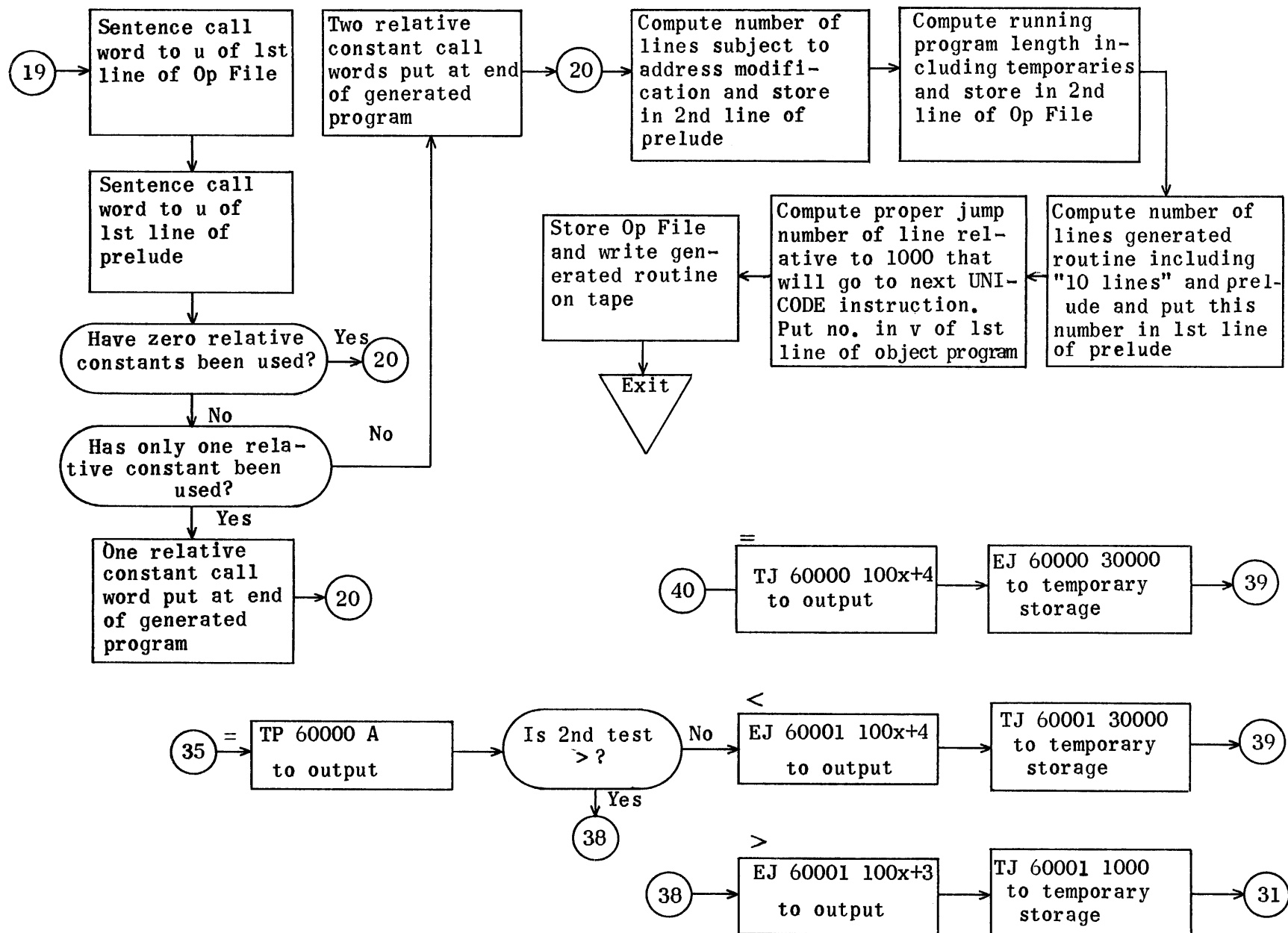
Subroutine to Put Line in Output

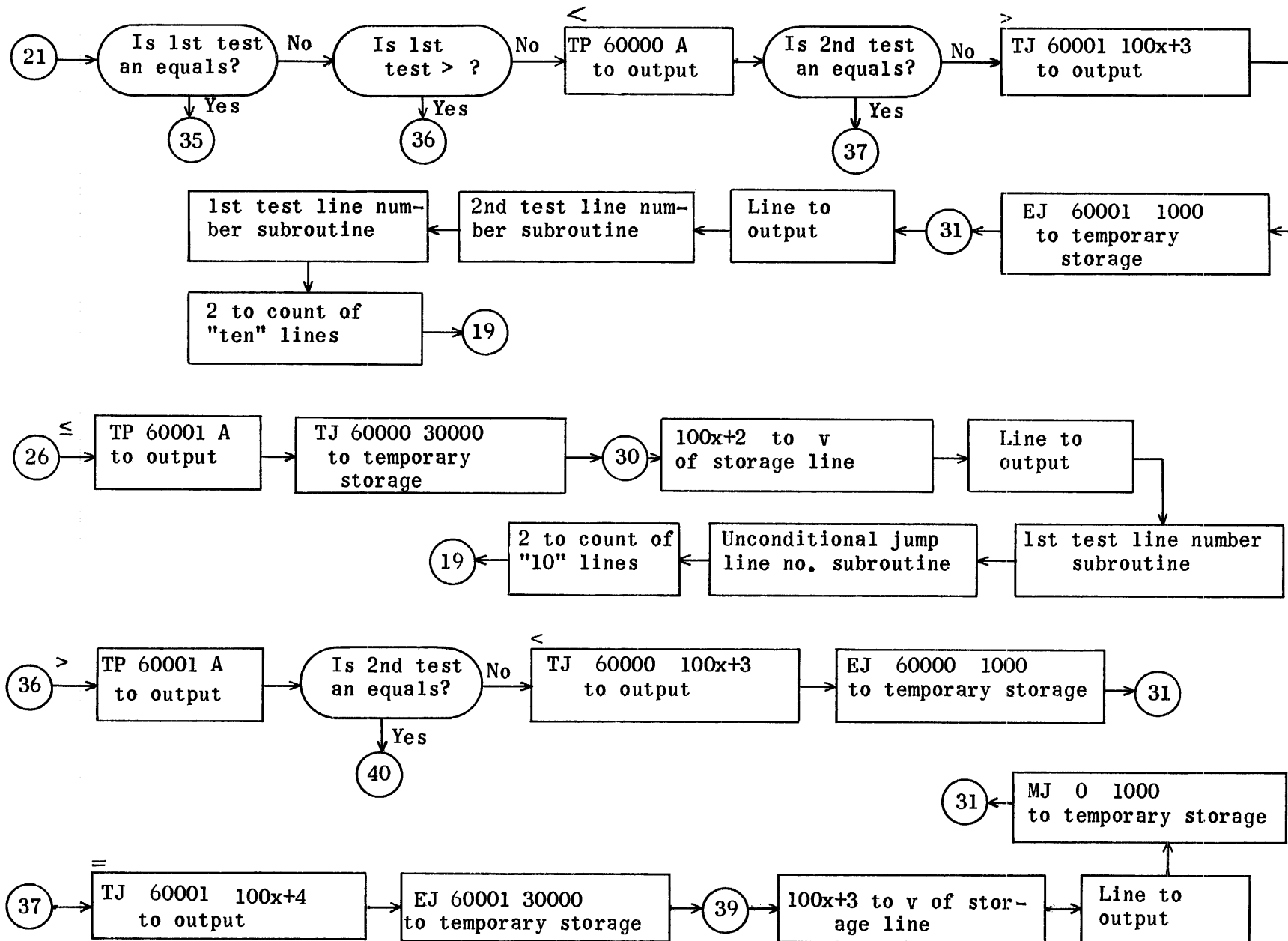


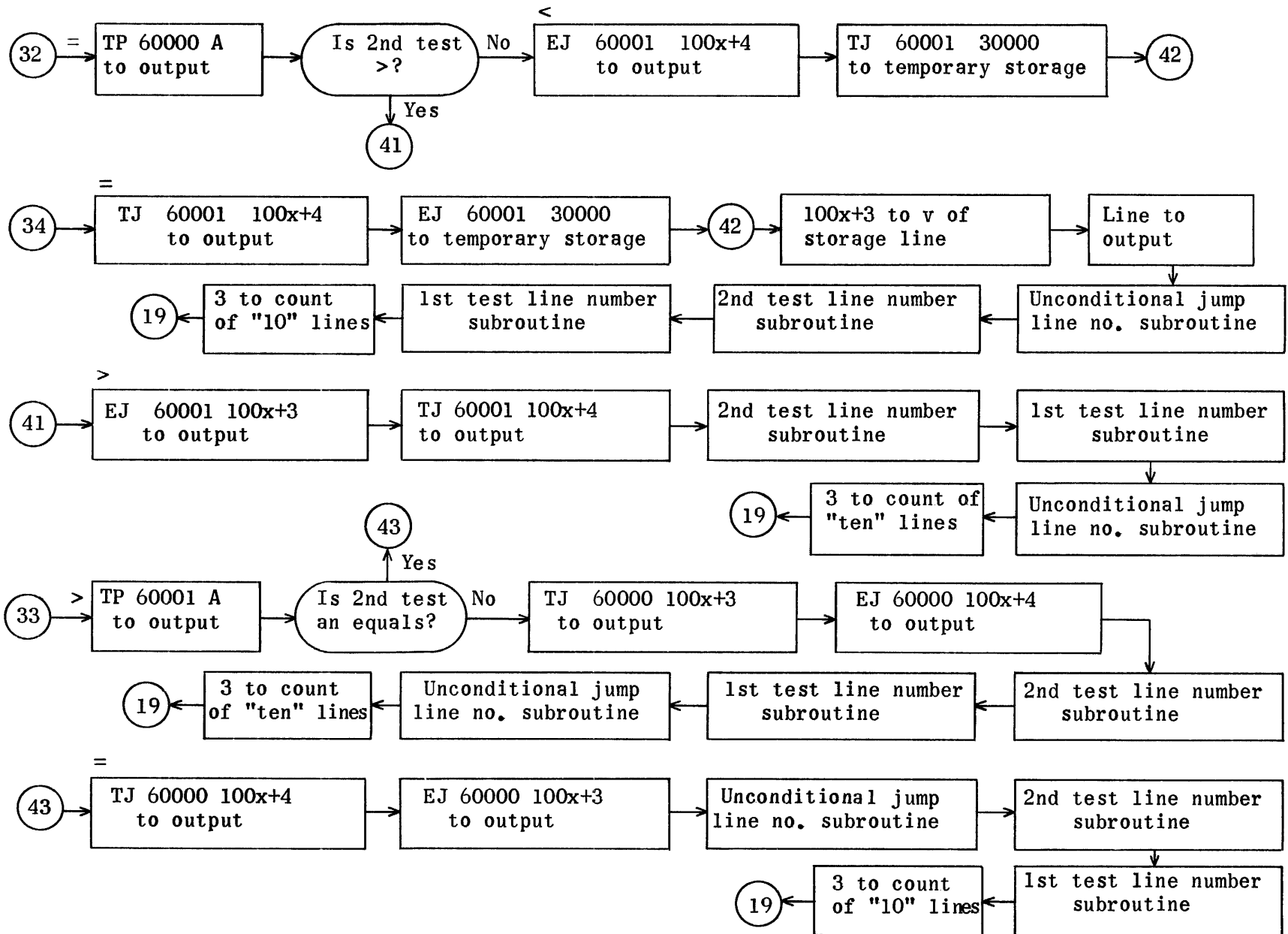
Subroutine to Generate Coding for a Subscripted Variable



Termination Subroutine







IF Generation Regions

RE	KB2512
RE	KA2644
RE	KC2660
RE	KD2737
RE	KE3070
RE	KF3123
RE	KH3265
RE	KK3353
RE	SA3457
RE	KR3462

Generation Subroutine regions are also needed to assemble this tape.

If Generation

KD - Unconditional jump with second test
 KB - Start- Control
 KF - Unconditional Jump and no second test. KF53 - no unconditional jump and a second test.
 KK - No unconditional jump and no second test. KK47-Termination loop.
 KC - Subscripted variable coding
 KH and KR - Subsidiary routines
 KA - Constants
 KE - Dummy instructions

Unconditional	IA	KD			
jump with	0	TP	BK7	A	} Is there a second test?
second	1	ZJ	KD2	KF	
test	2	TP	BK4	A	} Is first test an =?
	3	EJ	KA4	KD41	
<1st test	4	EJ	KA5	KD75	Is first test > ?
	5	TP	KE10	GL100	} TP 60000 A to output
	6	RJ	KH40	KH35	
	7	TP	BK6	A	} Is 2nd test an =?
>2nd test	10	EJ	KA4	KD25	
	11	TP	KE11	GL100	TJ 60001 30000
	12	RJ	KH24	KH21	To put proper jump (+3) in v of above
	13	RJ	KH40	KH35	Line to output
	14	RA	GL102	KA	Adding 1 to jump storage line in GL102
	15	TP	KE14	GL100	EJ 60001 30000
	16	TV	GL102	GL100	Jump (100X + 4) to v of above
	17	RJ	KH40	KH35	Line to output
	20	RJ	KH3	KH	Line no. of 2nd test to output plus "10" line
	21	RJ	KH7	KH4	1st test line no. + "10" line to output
	22	RJ	KH13	KH10	Unconditional jump line number + 10 line to output
	23	TP	KA4	GL106	3 to GL106 as count of number of 10 lines
= 2nd test	24	MJ	0	KK47	Jump to termination loop
	25	TP	KE11	GL100	TJ 60001 30000
	26	RJ	KH30	KH25	100X + 4 to v of above instruction
	27	RJ	KH40	KH35	Line to output
	30	RS	GL102	KA	100X + 3 now in GL102
	31	TP	KE14	GL100	EJ 60001 30000
	32	TV	GL102	GL100	100X + 3 to v of above
	33	RJ	KH40	KH35	Line to output
	34	RJ	KH13	KH10	MJ 0 v followed by 10 line to output.
					v = unconditional jump line no.
	35	RJ	KH3	KH	MJ 0 v followed by 10 0 1. 2nd test line no. = v
	36	RJ	KH7	KH4	MJ 0 v (first test) with 10 0 1 to output
	37	TP	KA4	GL106	Number of 10 lines (3) to storage
= 1st test	40	MJ	0	KK47	Jump to termination loop
	41	TP	KE10	GL100	} TP 60000 A to output
	42	RJ	KH40	KH35	
	43	TP	BK6	A	} Is 2nd test >
	44	EJ	KA5	KD61	

<2nd test	45	TP	KE14	GL100	EJ 60001 30000
	46	RJ	KH30	KH25	100X + 4 to v of above
	47	RJ	KH40	KH35	Line to output
	50	RS	GL102	KA	100X + 4 - 1
	51	TP	KE11	GL100	TJ 60001 30000
	52	MJ	0	KD32	
	53	TP	KE32	GL100	TU A 30000
	54	TU	A	GL100	Puts 61 --- Call word into u of above instruction
	55	RJ	KR4	KR1	Puts address of next instruction into v of above instruction and sends line to output. [TU 61---NI]
	56	MJ	0	30000	
	57	0	0	0	
>2nd test	60	0	0	0	
	61	TP	KE14	GL100	EJ 60001 30000
	62	RJ	KH24	KH21	100X + 3 to v of above instruction
	63	RJ	KH40	KH35	Line to output
	64	RA	GL102	KA	100X + 3 + 1
	65	TP	KE11	GL100	TJ 60001 30000
	66	TV	GL102	GL100	100X + 4 to v of above
	67	RJ	KH40	KH35	Line to output
	70	RJ	KH3	KH	2nd test
	71	RJ	KH7	KH4	1st test
	72	RJ	KH13	KH10	Unconditional test
	73	TP	KA4	GL106	3 to count of ten lines
>1st test	74	MJ	0	KK47	To termination loop
	75	TP	KE15	GL100	} TP 60001 A to output
	76	RJ	KH40	KH35	
	77	TP	BK6	A	
<2nd test	100	EJ	KA4	KD115	Is 2nd test =?
	101	TP	KE16	GL100	TJ 60000 30000
	102	RJ	KH24	KH21	100X + 3 to v of above
	103	RJ	KH40	KH35	Line to output
	104	RA	GL102	KA	100X + 3 + 1
	105	TP	KE24	GL100	EJ 60000 30000
	106	TV	GL102	GL100	100X + 4 to v of above
	107	RJ	KH40	KH35	Line to output
	110	RJ	KH3	KH	2nd test
	111	RJ	KH7	KH4	1st test
	112	RJ	KH13	KH10	Unconditional test
	113	TP	KA4	GL106	3 to count of "10" lines
= 2nd test	114	MJ	0	KK47	To termination loop
	115	TP	KE16	GL100	TJ 60000 30000
	116	RJ	KH30	KH25	100X + 4 to v of above
	117	RJ	KH40	KH35	Line to output
	120	RS	GL102	KA	100X + 4 - 1
	121	TP	KE24	GL100	EJ 60000 30000
	122	TV	GL102	GL100	100X + 3 to v of above
	123	RJ	KH40	KH35	Line to output
	124	RJ	KH13	KH10	Unconditional test
	125	RJ	KH3	KH	2nd test
	126	RJ	KH7	KH4	1st test

	127	TP	KA4	GL106	3 to count of "10" lines
	130	MJ	O	KK47	To termination loop
		CA	KD131		
Start-		IA	KB		
Control	0	MJ	O	30000	Exit
	1	TV	KK101	KH35	GL6 to v of KH35
	2	TV	KK102	KH41	KH6 to v of KH41
	3	TP	KA1	GL103	Clearing GL103, the counter of relative constants used
	4	TP	KE12	GL2	0 20000 0 to output
	5	TP	KA1	GL3	} Clear GL3, GL4
	6	TP	KA1	GL4	
	7	TP	BK1	GL5	Line no. to GL5
	10	TP	KA2	GL101	0 0 1000 to GL101
	11	TP	KE	GL100	} MJ 0 30000 to GL6, first line of Object Program
	12	RJ	KH40	KH35	
Generation	13	TP	BK11	A	} If BK11 isn't zero, X is a subscripted variable
coding for	14	ZJ	KB15	KR5	
X	15	SP	BK14	17	} If 77000 > call word, it is a pseudo-op sub-scripted variable
	16	TJ	KA10	KB37	
	17	RJ	KH16	KH14	Puts call word of sub. var. in Op File & ups count of relative cons. used
	20	TU	KA11	KE31	10000 to u of SA 30000 17
	21	TP	BK14	GL104	Storing call word of variable given 10000 call word
	22	RA	GL103	KA	Count of relative const. used increased by 1
	23	TP	KA7	Q	0 77777 0 mask to q
	24	QT	BK11	A	Call word of modulus to A _n
	25	TU	A	KE27	} Modulus call word to u of { TJ 30000 30000
	26	TU	A	KE30	
	27	RP	30004	KB31	} Subscript data to GL107-12
	30	TP	BK15	GL107	
	31	TP	KA6	Q	Mask of 0 0 77777 to q
	32	QT	BK11	A	Number of subscripts to A
	33	RJ	KC	KC1	Generation of coding for subscripted variable
	34	TP	KE1	GL100	} TP 30000 60000 to GL100 and to generated output coding
	35	RJ	KH40	KH35	
	36	MJ	O	KB47	
	37	TU	A	KE31	Call word to u of instruction (SA 30000 17) for pseudo op
Non-	40	MJ	O	KB23	
subscrip-	41	TP	KE1	GL100	TP 30000 60000 to GL100
ted X	42	SP	BK14	17	} Call word to u of above
	43	TU	A	GL100	
	44	RJ	KH40	KH35	Line to generated coding output
	45	TP	BK31	Q	} Is X a constant?
	46	QJ	KB57	KB47	
	47	TP	BK13	Q	} Is absolute value desired?
	50	QJ	KB51	KB53	
	51	TP	KE3	GL100	} TM 60000 60000 to output
	52	RJ	KH40	KH35	
	53	TP	BK12	Q	} Is negative value desired?
	54	QJ	KB55	KB57	

Generation coding for Y	55	TP	KE4	GL100	} TN 60000 60000 to output
	56	RJ	KH40	KH35	
	57	TP	BK21	A	} Is Y a subscripted variable?
	60	ZJ	KB61	KR10	
	61	SP	BK24	17	} If 77000>call word, it is a pseudo op sub. variable
	62	TJ	KA10	KB110	
	63	RJ	KH16	KH14	} Puts call word of sub. var. in Op File and ups count of rel. cons. used
	64	TP	GL103	A	
	65	ZJ	KB66	KB71	} Have any relative constants of 10000 type been used?
	66	TU	KA12	KE31	
Non-sub- scripted Y	67	TP	BK24	GL105	10001 to u of SA 30000 17
	70	MJ	O	KB73	Storing call word of variable given 10001 call word
	71	TU	KA11	KE31	10000 to u of SA 30000 17
	72	TP	BK24	GL104	Storing call word of variable given 10000 call word
	73	RA	GL103	KA	Count of relative constants increased
	74	TP	KA7	Q	0 77777 0 to q
	75	QT	BK21	A	} Modulus call word to u of { TJ 30000 30000 DV 30000 q
	76	TU	A	KE27	
	77	TU	A	KE30	
	100	RP	30004	KB102	} Subscript data of Y to temporary storage
	101	TP	BK25	GL107	
	102	TP	KA6	Q	0 0 77777 to q
	103	QT	BK21	A	Number of subscripts to A
	104	RJ	KC	KC1	Gen. of coding for subscripted variable
	105	TP	KE2	GL100	} TP 30000 60001 to output
	106	RJ	KH40	KH35	
	107	MJ	O	KB120	} Call word to u of SA 30000 17 for pseudo op
	110	TU	A	KE31	
	111	MJ	O	KB74	} TP 30000 60001 to output line
	112	TP	KE2	GL100	
	113	SP	BK24	17	} Call word to u of above line
	114	TU	A	GL100	
	115	RJ	KH40	KH35	Line to output
	116	TP	BK32	Q	} Is Y a constant?
	117	QJ	KB130	KB120	
	120	TP	BK23	Q	} Is absolute value desired?
	121	QJ	KB122	KB124	
	122	TP	KE5	GL100	} TM 60001 60001 to output
	123	RJ	KH40	KH35	
	124	TP	BK22	Q	} Is negative value desired?
	125	QJ	KB126	KB130	
	126	TP	KE6	GL100	} TN 60001 60001 to output
	127	RJ	KH40	KH35	
	130	TP	BK10	A	} Is there an unconditional jump?
	131	ZJ	KD	KF53	
		CA	KB132		

Uncon-	IA	KF		
ditional	0 RJ	SA2	SA	} Puts MJ Z 0 in A where Z is code number for relation test
Jump and	1 TP	KE7	A	
no 2nd	2 RP	30005	KF47 ≥ ?	} Determining what first test is
Test	3 TJ	KF4	KF4	
	4 MJ	2	KF11 < ?	
	5 MJ	3	KF22 = ?	
	6 MJ	4	KF26 > ?	
	7 MJ	5	KF32 ≤ ?	
	10 MJ	6	KF43	
			NOT = ?	
< 1st test	11 TP	KE10	GL100	} TP 60000 A to output
	12 RJ	KH40	KH35	
	13 TP	KE11	GL100	TJ 60001 30000
	14 RJ	KH34	KH31	100X + 2 to v of above
	15 RJ	KH40	KH35	Line to output
	16 RJ	KH13	KH10	Unconditional test
	17 RJ	KH7	KH4	1st test
	20 TP	KA3	GL106	2 to count of "10" lines
	21 MJ	0	KK47	To termination loop
= (1st)	22 TP	KE10	GL100	} TP 60000 A to output
	23 RJ	KH40	KH35	
	24 TP	KE14	GL100	EJ 60001 30000
	25 MJ	0	KF14	
> 1st	26 TP	KE15	GL100	} TP 60001 A to output
	27 RJ	KH40	KH35	
	30 TP	KE16	GL100	TJ 60000 30000
≤ 1st	31 MJ	0	KF14	
	32 TP	KE15	GL100	} TP 60001 A to output
	33 RJ	KH40	KH35	
	34 TP	KE16	GL100	TJ 60000 30000
	35 RJ	KH34	KH31	100X + 2 to v of above
	36 RJ	KH40	KH35	Line to output
	37 RJ	KH	KH4	1st test
	40 RJ	KH13	KH10	Unconditional jump
	41 TP	KA3	GL106	2 to count of "10" lines
	42 MJ	0	KK47	To termination loop
NOT =	43 TP	KE10	GL100	} TP 60000 A to output
(1st)	44 RJ	KH40	KH35	
	45 TP	KE14	GL100	EJ 60001 30000
≥ 1st	46 MJ	0	KF35	
	47 TP	KE10	GL100	} TP 60000 A to output
	50 RJ	KH40	KH35	
	51 TP	KE11	GL100	TJ 60001 30000
	52 MJ	0	KF35	
No uncon-	53 TP	BK7	A	} Is there a second test?
ditional	54 ZJ	KF55	KK	
jump and	55 TP	BK4	A	} Is 1st test =?
a second	56 EJ	KA4	KF106	
test	57 EJ	KA5	KF124	Is 1st test > ?
< (1st test)	60 TP	KE10	GL100	} TP 60000 A to output
	61 RJ	KH40	KH35	
	62 TP	BK6	A	} Is 2nd test =?
	63 EJ	KA4	KF75	

2nd test >	64	TP	KE11	GL100	TJ 60001 30000
	65	RJ	KH24	KH21	100X + 3 to v of above
	66	RJ	KH40	KH35	Line to output
	67	TP	KE21	GL100	} EJ 60001 1000 to output
	70	RJ	KH40	KH35	
	71	RJ	KH3	KH	2nd test setup
	72	RJ	KH7	KH4	1st test setup
	73	TP	KA3	GL106	2 to count of "ten" lines
2nd test =	74	MJ	0	KK47	To termination
	75	TP	KE11	GL100	TJ 60001 30000
	76	RJ	KH30	KH25	100X + 4 to v of above
	77	RJ	KH40	KH35	Line to output
	100	TP	KE14	GL100	EJ 60001 30000
	101	RS	GL102	KA	100X + 4 - 1
	102	TV	GL102	GL100	100X + 3 to v of above instruction
	103	RJ	KH40	KH35	Line to output
	104	TP	KE20	GL100	MJ 0 1000
= (1st test)	105	MJ	0	KF70	} TP 60000 A to output
	106	TP	KE10	GL100	
	107	RJ	KH40	KH35	} Is 2nd test > ?
	110	TP	BK6	A	
< 2nd test	111	EJ	KA5	KF117	} EJ 60001 30000
	112	TP	KE14	GL100	
	113	RJ	KH30	KH25	100X + 4 to v of above
	114	RJ	KH40	KH35	Line to output
	115	TP	KE11	GL100	TJ 60001 30000
> 2nd test	116	MJ	0	KF101	} EJ 60001 30000
	117	TP	KE14	GL100	
	120	RJ	KH24	KH21	100X + 3 to v of above
	121	RJ	KH40	KH35	Line to output
	122	TP	KE22	GL100	TJ 60001 1000
> 1st test	123	MJ	0	KF70	} TP 60001 A to output
	124	TP	KE15	GL100	
	125	RJ	KH40	KH35	} Is 2nd test = ?
	126	TP	BK6	A	
< 2nd test	127	EJ	KA4	KF135	} TJ 60000 30000
	130	TP	KE16	GL100	
	131	RJ	KH24	KH21	100X + 3 to v of above
	132	RJ	KH40	KH35	Line to output
	133	TP	KE23	GL100	EJ 60000 1000
	134	MJ	0	KF70	} TJ 60000 30000
= 2nd test	135	TP	KE16	GL100	
	136	RJ	KH30	KH25	100X + 4 to v of above
	137	RJ	KH40	KH35	Line to output
	140	TP	KE24	GL100	EJ 60000 30000
	141	MJ	0	KF101	
		CA	KF142		
No Unconditional Jump & no Second Test	0	IA	KK		
	1	RJ	SA2	SA	} Puts MJ Z 0 in A where Z is code number for relation test
	2	TP	KE7	A	
	3	RP	30005	KK43 ≥ ?	
	3	TJ	KK4	KK44	

	4	MJ	2	KK11 < ?	} Determination of type of 1st test
	5	MJ	3	KK23 = ?	
	6	MJ	4	KK27 > ?	
	7	MJ	5	KK33 ≤ ?	
<1st test	10	MJ	6	KK37 NOT =?	
	11	TP	KE10	GL100	} TP 60000 A to output
	12	RJ	KH40	KH35	
	13	TP	KE11	GL100	TJ 60001 30000
	14	RJ	KH34	KH31	100X + 2 to v of above
	15	RJ	KH40	KH35	Line to output
	16	TP	KE20	GL100	} MJ 0 1000 to output
	17	RJ	KH40	KH35	
	20	RJ	KH7	KH4	1st test set-up
	21	TP	KA	GL106	1 to count of "10" lines
= 1st test	22	MJ	0	KK47	To termination loop
	23	TP	KE10	GL100	} TP 60000 A to output
	24	RJ	KH40	KH35	
	25	TP	KE14	GL100	EJ 60001 30000
>	26	MJ	0	KK14	} TP 60001 A to output
	27	TP	KE15	GL100	
	30	RJ	KH40	KH35	TJ 60000 30000
	31	TP	KE16	GL100	} TP 60001 A to output
≤	32	MJ	0	KK14	
	33	TP	KE15	GL100	TJ 60000 1000
	34	RJ	KH40	KH35	} TP 60001 A to output
	35	TP	KE13	GL100	
NOT =	36	MJ	0	KK17	TJ 60000 1000
	37	TP	KE10	GL100	} TP 60000 A to output
	40	RJ	KH40	KH35	
	41	TP	KE21	GL100	EJ 60001 1000
≥	42	MJ	0	KK17	} TP 60000 A to output
	43	TP	KE10	GL100	
	44	RJ	KH40	KH35	TJ 60001 1000
	45	TP	KE22	GL100	} TP 60000 A to output
	46	MJ	0	KK17	
Termination Loop	47	SP	BK3	17	} Sentence call word to u of 1st line of Op File
	50	TU	A	KH57	
	51	TP	A	GL	Call word to u of 1st line of prelude
	52	TP	GL103	A	} Is number of relative const. equal to zero?
	53	ZJ	KK54	KK64	
	54	EJ	KA	KK62	Is number of rel. cons. equal to 1?
No. of rel. cons. = 2	55	TP	GL104	GL100	} Relative constant call words put at end of generated program. Numbers were equated to 10000 and 10001 in coding
	56	RJ	KH40	KH35	
	57	TP	GL105	GL100	
	60	RJ	KH40	KH35	
	61	MJ	0	KK64	} Call word of no.equated to 10000 put at end
	62	TP	GL104	GL100	
	63	RJ	KH40	KH35	} Ordinal 100X program line to A
	64	TP	GL101	A	
	65	ST	KA2	GL1	Subtracting 1000 and putting in GL1
	66	TV	A	GL	100X-1000 to GL _v
	67	ST	GL106	A	} 100X-1000-number of "ten" lines + 2 to 2nd line of Op File
	70	AT	KA3	KH60	

	71	RA	GL	KA13	100X - 1000 + 6 to v of prelude 1st line
	72	TP	GL101	A	
	73	ST	GL106	A	100X - number of "ten" lines + 3 to v of
	74	AT	KA4	A	1st line of running program
	75	TV	A	GL6	
	76	TP	KK103	OP1	
	77	RJ	OP	OP2	Writing Op File and gen. routines on tape
	100	MJ	0	KB	
	101	0	0	GL6	
	102	0	0	KH61	
	103	0	GL	KH57	
		CA	KK104		
		IA	KC		
Sub-	0	MJ	0	30000	
scripted	1	EJ	KA	KC53	Are no. of subscripts 1?
Variable	2	EJ	KA3	KC45	Are no. of subscripts 2?
Coding	3	EJ	KA4	KC37	Are no. of subscripts 3?
Number of	4	TP	KE25	GL100	SP 30000 0
subscripts	5	SP	GL112	17	Call word of L subscript to above line. Sub-
are 4	6	TU	A	GL100	scripts assumed to be I,J,K,L
	7	RJ	KH40	KH35	Line to output
	10	TP	KE26	GL100	MA 30000 30000
	11	TU	GL111	GL100	$K_M \rightarrow$ u of above line
	12	TV	GL111	GL100	$K \rightarrow$ v of above
	13	RJ	KH40	KH35	Line to output
	14	TU	GL110	GL100	J_M to u of MA 30000 30000
	15	TV	GL110	GL100	J to v of above
	16	RJ	KH40	KH35	Line to output
	17	TU	GL107	GL100	I_M to u of MA 30000 30000
	20	TV	GL107	GL100	I to v of above
	21	RJ	KH40	KH35	Line to output
	22	TP	KE27	GL100	TJ Mod. 30000
	23	TV	GL101	GL100	
	24	RA	GL100	KA3	100X + 2 to v of above
	25	RJ	KH40	KH35	Line to output
	26	TP	KE30	GL100	
	27	RJ	KH40	KH35	DV Mod. q to output
	30	TP	KE31	GL100	SA CW 17 to output. CW is call word of
	31	RJ	KH40	KH35	address of line that will hold address of
					1st line of array
	32	RJ	KR4	KR	TU A NI to output. NI means Next Instr.
	33	MJ	0	KC	
	34	0	62000	0	Stored constant used to identify dummy
					pseudo op functions
	35	RJ	KD56	KD53	TU 61--- NI to output
3 sub-	36	MJ	0	KB105	Return to finish handling variable Y
scripts	37	TP	KE25	GL100	SP 30000 0
	40	SP	GL111	17	
	41	TU	A	GL100	K subscript call word to u of above
	42	RJ	KH40	KH35	Line to output

	43	TP	KE26	GL100	MA 30000 30000 to output line
2 sub- scripts	44	MJ	0	KC14	
	45	TP	KE25	GL100	SP 30000 0
	46	SP	GL110	17	J subscript call word to u of above
	47	TU	A	GL100	
	50	RJ	KH40	KH35	Line to output
	51	TP	KE26	GL100	MA 30000 30000 to output line
1 sub- script	52	MJ	0	KC17	
	53	TP	KE25	GL100	SP 30000 0
	54	SP	GL107	17	I call word to u of above
	55	TU	A	GL100	
	56	MJ	0	KC21	
			CA	KC57	

Subsidiary Subroutines

2nd test	TA	KH			
line no.	0	TP	KE	GL100	MJ 0 30000 to output line
subroutine	1	TP	BK7	A	Line number of 2nd test to A
	2	RJ	KH56	KH45	Getting call word for line number, putting it in Op File, putting it in output and putting 10 line in output
	3	MJ	0	30000	
1st test	4	TP	KE	GL100	MJ 0 30000
	5	TP	BK5	A	Line number 1st test to A
subroutine	6	RJ	KH56	KH45	Line number call word subroutine
Uncon-	7	MJ	0	30000	
ditional	10	TP	KE	GL100	MJ 0 30000
jump line	11	TP	BK10	A	Line no. of unconditional jump to A
no. sub-	12	RJ	KH56	KH45	Line number call word subroutine
routine	13	MJ	0	30000	
Subscripted	14	RJ	KH44	KH41	Call word sent to Op File
var. call	15	RA	GL2	KA2	Ups count of relative constants used in prelude
word to					
Op. File	16	MJ	0	30000	
	17	RJ	KD56	KD53	TU 61 --- NI to output
	20	MJ	0	KB34	Return to finish handling variable X
+3 Jump setter	21	TP	GL101	GL102	(1000 + count of lines) to GL102
	22	RA	GL102	KA4	+ 3
	23	TV	GL102	GL100	v of GL100 set to 100X + 3
	24	MJ	0	30000	
+4 Jump setter	25	TP	GL101	GL102	100X + 4 to v of output line
	26	RA	GL102	KA5	
	27	TV	GL102	GL100	
	30	MJ	0	30000	
+ 2 Jump setter	31	TP	GL101	GL102	100X + 2 to v of output line
	32	RA	GL102	KA3	
	33	TV	GL102	GL100	
	34	MJ	0	30000	
Subroutine	35	TP	GL100	[GL6]	Inserting output line in proper place in generated coding
to put					
line in	36	RA	KH35	KA	Upping v of KH35 by 1
output	37	RA	GL101	KA	Count of 1000 plus number of lines
	40	MJ	0	30000	

Loads Op File for call word Line number call word subroutine	41	TP	A	[KH61]	Loads Op File with call word
	42	RA	KH41	KA	Ups v of above by 1
	43	RA	KH57	KA	Increases count of Op File lines by 1
	44	MJ	0	30000	
	45	RJ	LW	LW1	Getting call word of line number
	46	TV	Q	GL100	Call word to v of output line
	47	RJ	KH44	KH41	Call word to Op File
	50	SP	Q	17	Call word of line no. to Au
	51	TP	BK33	Q	Puts Pseudo-Op indicator in Q
	52	RJ	KI	KI1	Check for illegal line numbers
	53	RJ	KH40	KH35	Line to output
	54	TP	KE17	GL100	10 0 1 to output
	55	RJ	KH40	KH35	
	56	MJ	0	30000	
Op File build-up space	57	0	30000	2	
	60	0	0	0	
	61	0	0	0	
	62	0	0	0	
	63	0	0	0	
	64	0	0	0	
	65	0	0	0	
		CA	KH66		
		IA	KA		
		0	0	1	
Constants	1	0	0	0	
	2	0	0	1000	
	3	0	0	2	
	4	0	0	3	
	5	0	0	4	
	6	0	0	77777	
	7	0	77777	0	
	10	0	77000	0	
	11	0	10000	0	
	12	0	10001	0	
	13	0	0	6	
		CA	KA14		
		IA	KE		
Dummy Instruc- tions for Use in Building Generated Routine	0	MJ	0	30000	
	1	TP	30000	60000	
	2	TP	30000	60001	
	3	TM	60000	60000	
	4	TN	60000	60000	
	5	TM	60001	60001	
	6	TN	60001	60001	
	7	MJ	0	0	
	10	TP	60000	A	
	11	TJ	60001	30000	
	12	0	20000	0	
	13	TJ	60000	1000	
	14	EJ	60001	30000	
	15	TP	60001	A	

	IA	SA		
0	SP	BK4	17	} Puts 1st test code relation number in A_u and then in $KE7_u$.
1	TU	A	KE7	
2	MJ	0	30000	
	CA	SA3		

Subroutine
to put <
TU-A-NI
in output

Print Generation Routine

It is assumed that everything following PRINT# in this instruction is to be printed. Thus, parentheses are included in the print-out. Since the printed data is generally self-explanatory comments following a print instruction are assumed redundant. If they are included, the routine will include them in the printed output.

It is further assumed that any one Print instruction is no longer than 6 Unityper lines. Material that would fill more space than this should be divided into as many instructions as needed to bring each within the proper limit.

The first line of the string-out contains the number of lines in the string-out. In the second line is the line number of the instruction. The third contains the title, PRINT. The v portion of the fourth line holds the call word assigned to the instruction. The fifth and succeeding lines contain the excess-three characters which will be converted to Flex code during generation.

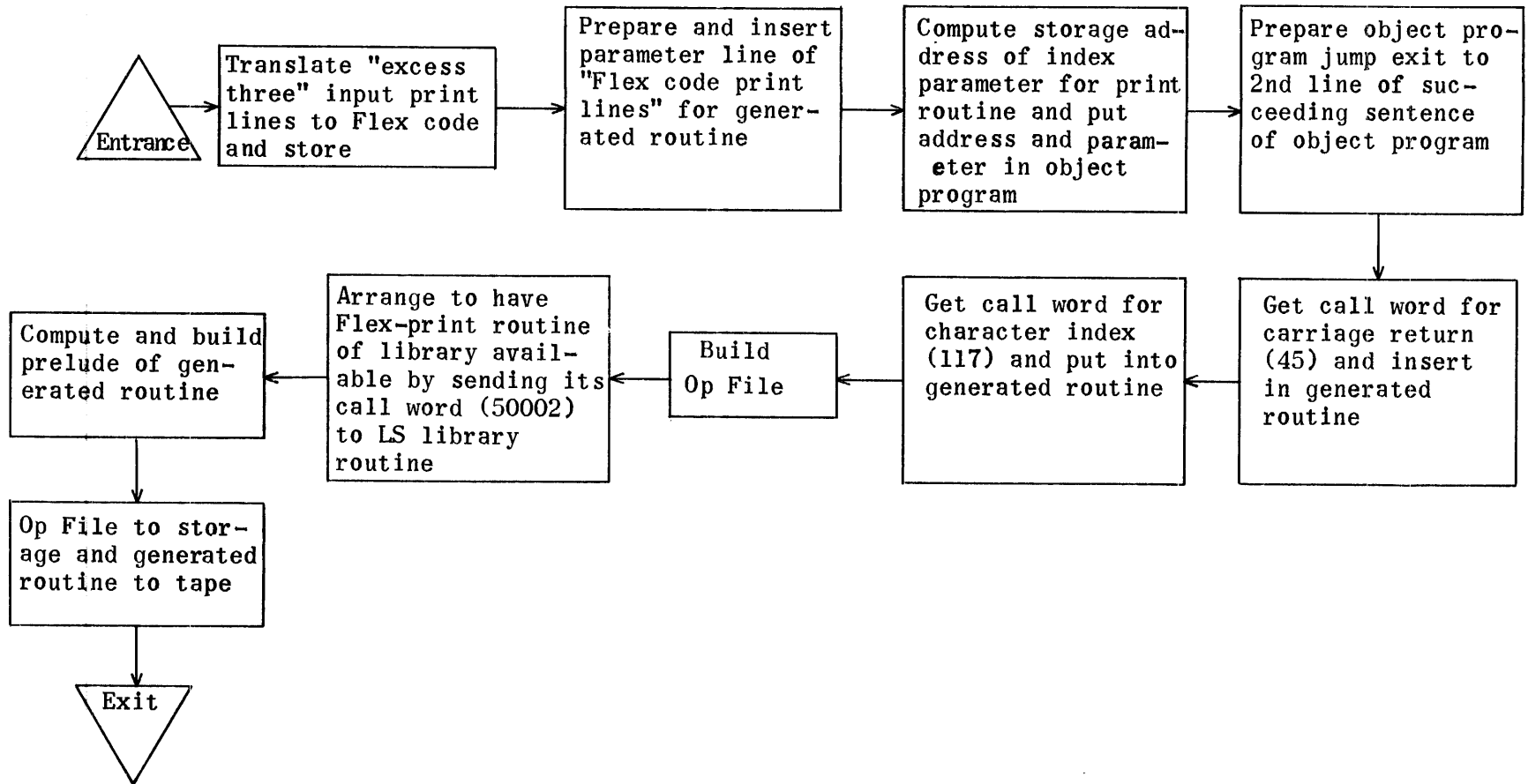
Excess-three codes are converted by the routine to Flex codes which are stored for later printing during the Object Program run. A library routine is referenced during this run to effect the printing. Because the library routine to print Flex code directly is much shorter than a library routine that could print directly from excess three, this procedure of converting from excess-three to Flex code during generation was adopted. Space saving during the Object Program run was considered of paramount importance.

GL, the region in which the print subroutine is stored before writing on tape, has to have at its disposal a variable address length that will hold the 22₈-address print subroutine and prelude plus the indeterminate group of Flex codes. The Flex codes, because of the addition of shift-up and shift-down codes and substitution of more than one Flex code for a corresponding excess-three character, will likely take up more space than the group of excess-three lines in BK. GL cannot exceed 520₈ lines and in most cases will be far less.

The size of the Op File 1 item for PRINT is 3 addresses. The generated subroutine is 22₈ + number of words of Flex code. The number of words in the running program is 12₈ + number of words of Flex code.

After the instruction in BK, the buffer region, has been identified during generation as PRINT, the control generation routine by instruction RJ UR UR1 activates the print generator. The print routine generates the coding needed for the print subroutine and gets it written on the proper tape. Also stored in NP for later writing on tape is the Op File 1 item.

Flow Chart of Print Generation Routine



Print Generation

Regions

RE	UR2512	Print generator
RE	GL5360	Storage for generated subroutine plus prelude

Generation Subroutine Regions are also needed to assemble this tape.

Print Generation

	IA	UR		
	0	MJ	0	30000
	1	TP	BK	A
	2	RS	A	UR75
	3	TV	A	UR50
	4	TP	UR50	VX4
	5	TP	UR51	VX3
	6	RJ	VX2	VX
	7	TP	VX3	GL20
	10	TU	UR76	GL20
	11	TV	GL20	UR106
	12	RA	UR106	UR77
	13	TV	UR106	UR65
	14	TV	UR106	UR63
	15	RA	UR63	UR100
	16	SP	UR106	17
	17	AT	UR106	GL21
	20	TP	UR101	A
	21	RJ	CW	CW1
	22	TV	Q	UR64
	23	TP	UR102	A
	24	RJ	CW	CW1
	25	TU	A	UR65
	26	SP	BK3	17
	27	TU	A	UR52
	30	TU	A	UR55
	31	TV	GL20	UR53
	32	RA	UR53	UR103
	33	TP	UR54	A
	34	RJ	LS	LS1
	35	TP	UR104	UR56
	36	TV	GL20	UR57
	37	TP	BK1	UR62
	40	TV	GL20	UR55
	41	RA	UR55	UR105
	42	RP	30020	UR44
	43	TP	UR55	GL
	44	TP	UR47	OP1
	45	RJ	OP	OP2
	46	MJ	0	UR
	47	0	GL	UR52
	50	0	BK4	0
	51	0	GL22	0
Op File	52	0	30000	3
l Item	53	0	0	30000
	54	0	50002	0
				Exit
				Entry.No. of lines of input → a
				No. of words of excess 3 to parameter line
				Parameter words to X3 to Flex code routine
				Jump to X3 to Flex code routine
				Output parameter line from X3 to Flex code
				Changing u to address relative to 1000
				Computing and putting temporary storage address in running program where needed
				Preparing jump exit of running program to 2nd line of succeeding subroutine
				Temp storage address in u and v sent to region accumulating subroutine, later to be used as a parameter in print tag rtne.
				Getting call word for carriage return Flex code
				Inserting call word for carriage return in program
				Getting call word for index 117
				Call word to running program
				Call word of statement to 1st line of Op File
				Computing & putting no. of lines in running program, including temporaries, into Op File
				Putting call word for print tag rtne. into LN, so this routine will be available in the library
				No. address modification lines to prelude
				Number unmodifiable constants to prelude
				Line no. to prelude
				Computing & putting proper number in v of prelude 1st line
				Prelude & 1st line of running program to assembly region
				Transferring Op File & subroutine to Op File from where it will go on tape later
				Exit to 1st line
				Parameter line for Op control routine
				Parameter lines for X3 to Flex code
				u = call word of print statement
				v = no. lines, running program, including temporaries
				Call word for print tag routine

GL	0	{	55	0	30000	30000	u = Call word of print statement. v = #
	1		56	0	0	30000	lines prelude & routine
	2		57	0	0	0	v = number lines subject to address modi-
Pre-	3		60	0	0	0	fication
lude	4		61	0	0	0	v = number of unmodifiable constants
	5		62	0	0	0	
	6		63	MJ	0	30000	Line number
	7		64	PR	0	30000	Exit - 1st line of running program
	10		65	TP	30000	30000	v = 1 (45)
Run-	11		66	TP	01007	50002	u = 1 (117) v = temp. storage address
ning	12		67	10	0	3	
	13		70	TP	01010	50002	Parameter lines to Print Tag Routine
Pro-	14		71	10	0	4	
gram	15		72	RJ	50002	50002	
	16		73	10	2	0	Jump to print tag routine
	17		74	MJ	0	1000	
	20		75	0	0	4	Jump back to first line of running program
	21		76	0	1011	0	
			77	0	0	1011	
			100	0	0	2	
			101	0	0	45	
			102	0	0	117	
			103	0	0	12	
			104	0	0	14	
			105	0	0	22	
			106	0	0	30000	
			CA	UR107			

Compute Generation Routine

This routine forms a code that sets up data in the proper location for the equation coding, followed by a return jump to this part. All the actual computational coding is done by the equation generation routine.

The string-out input to the Compute Generator contains the call words of the variables or constants in the same sequence in which they appear in the input sentences. Subscripted variables have multipliers and moduli saved in addition to their call words. Thus one subscripted variable occupies 2, 3 or 4 locations, depending on how many subscripts it has, and the subscript call words follow thereafter. Functions within pseudo operations have only the call word of the function in the string-out and no argument call words, even though the arguments may be stated in the input sentence. At the end of every string-out is a line of zeros followed by 01 227-7 (the XS3 representation of Δ). When a compute sentence contains several terms separated by "and's" in the input sentence, the string-out has a zero line inserted between the last call word of one term and the first call word of the following term.

The Compute Generator builds up Operation File I, prelude, running program and file of relative constants from this data. The first line of the generated coding - the exit line - is an MJ to the line that follows second after the last generated line. The last line of the running code is an MJ to the exit line. After this follows a list of generated relative constants.

For terms that are not inside parentheses in the input sentence (that is, for the first string-out call word and those following zero lines) the routine generates:

- 1) In most cases (call-word types 66,65,64,4)

RJ	25---	25---	or	RJ	4---	4----
10	00000	00001		10	00000	00001

- 2) For subscripted variables (call-word type 77)

TP	64---	62000	}	1,2,3 or 4 of those lines depending on how many subscripts the subscripted variable has.
TP	64---	62001		
TP	64---	62002		
TP	64---	62003		
RJ	24---	24---		
10	00000	0001		

3) And for dummy functions (call-word type 61)

	TU	Call word + 1	α
	TV	Call word + 1	α
	RA	α	Call word of constant 1
α	RJ	[30000]	[30000]

The terms that are inside parentheses in the input sentence precede the code of the RJ line and ten line of their respective symbol. Their handling splits into two groups: those that are within or are input parameters to a pseudo operation, and those that are not.

To set up input parameters for reference to a pseudo operation the routine generates:

4) For functions (call-word type 66)

β	00	Call word	Call word	} in the location for relative constants
$\beta + 1$	00	25---	25---	
	TP	β	61---	} in the running code
	TP	$\beta + 1$	61---	
	00	25---	00000	in Operation File I

5) For subscripted variables (call-word type 77)

γ	00	Call word	Call word	relative constant
	TP	γ	76X---	running code
	00	77---	00000	OP File I
	TP	Call word of 63---	(1)	} with 1,2,3 or 4 subscripts No. (4), (1) & (4), (1) & (2) & (4) or (1) & (2) & (3) & (4) of these rows appear in running code.
		1st multi-plier		
	TP	Call word of 63---	(2)	
		2nd multi-plier		
	TP	Call word of 63---	(3)	
		3rd multi-plier		
	TP	Call word of 63---	(4)	
		modulus		

6) For all others (call-word types 64, 65 or 67)

TP	Call word	63---	running code
----	-----------	-------	--------------

For terms within parentheses of a function or a subscripted variable but not within a pseudo operation the routine generates:

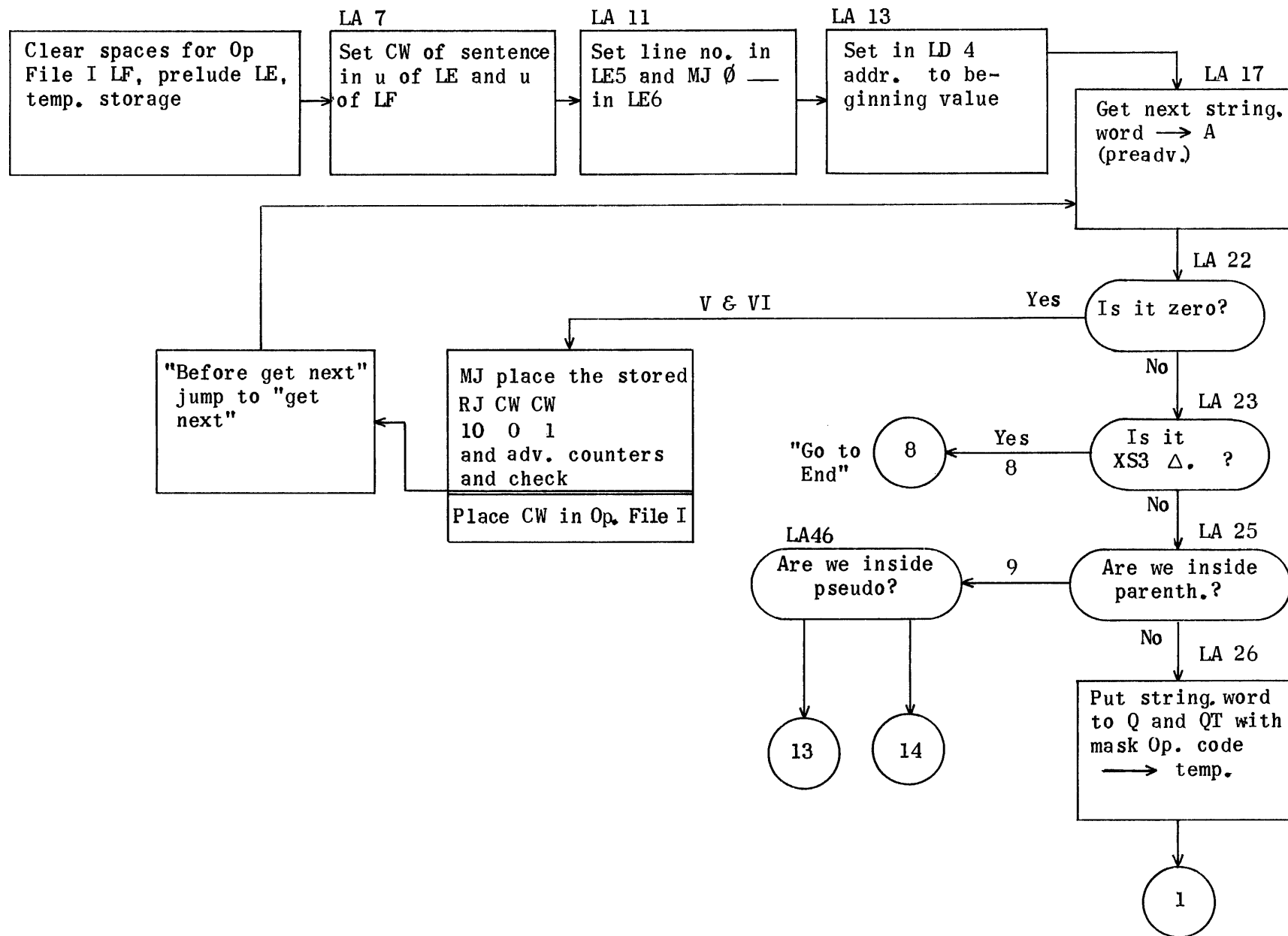
7) For subscripted variables (call-word types 76 or 77)

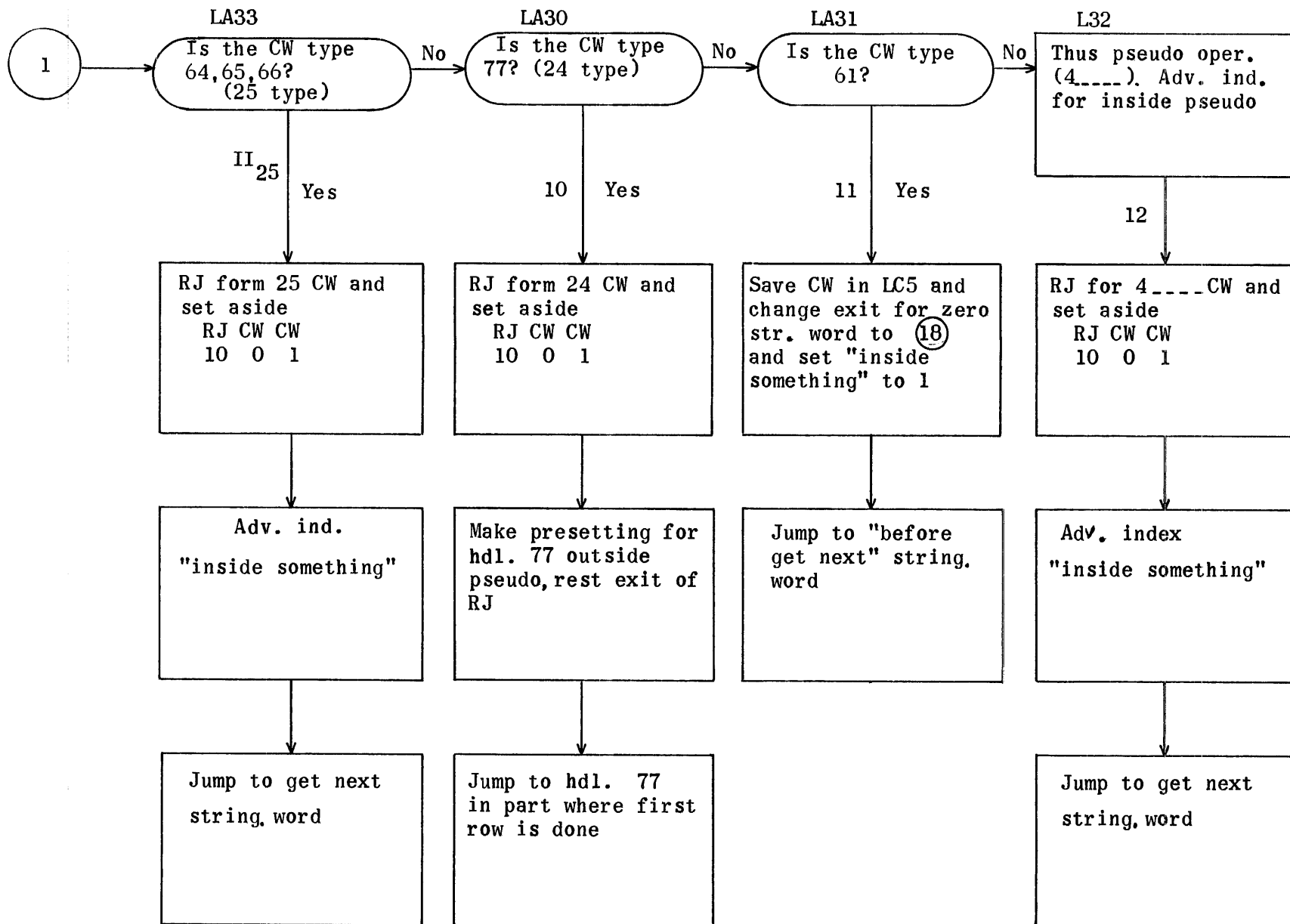
δ	00	Call word	Call word	relative constants
	TP	δ	75---	running code
	TP	modulus	62---	
(for 77 only)	00	Call word	00000	OP File I (this does not apply for Call-word type 76)

8) For all others (Call-word types 63,64,65 or 67)

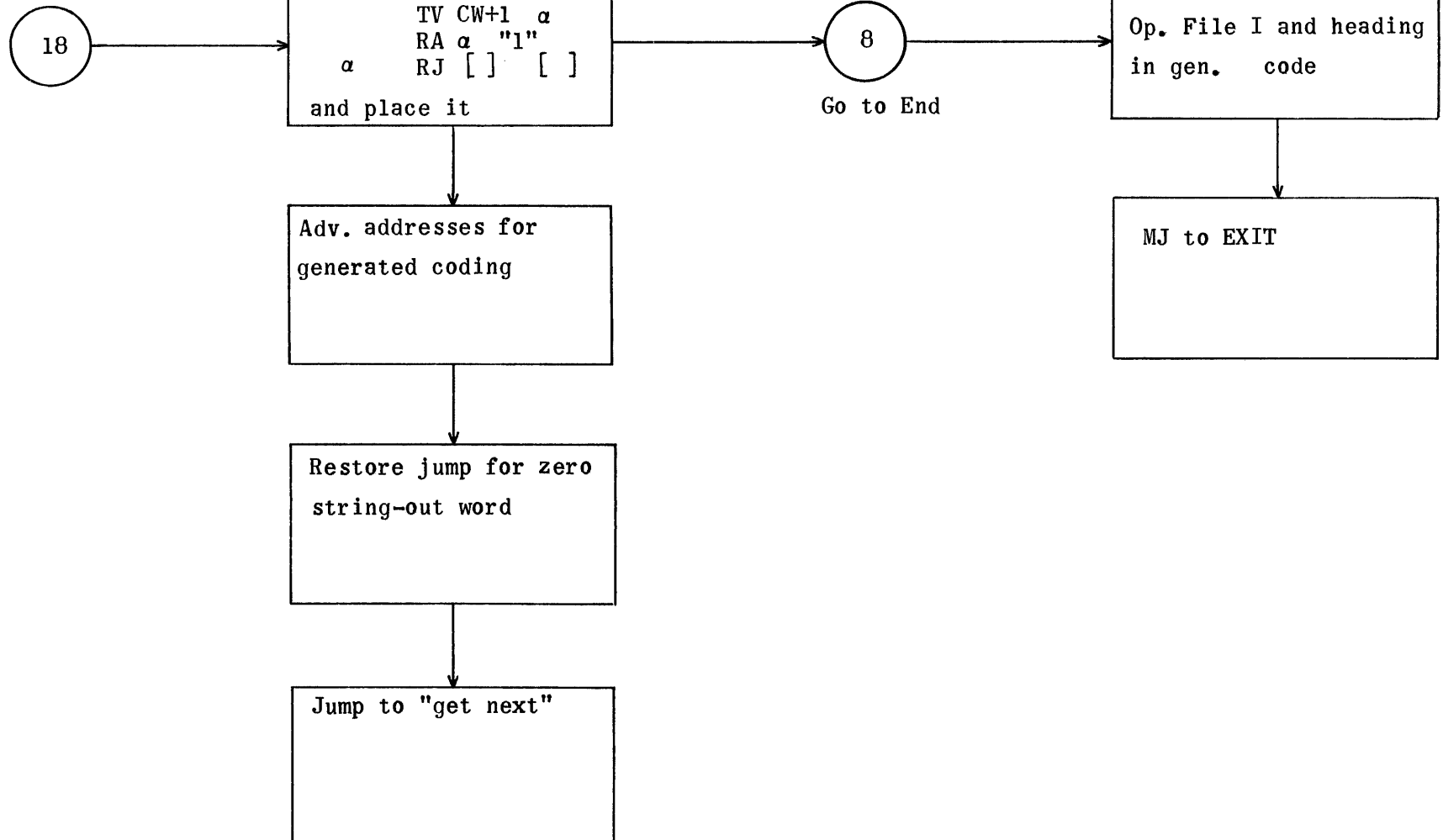
TP	Call word	62---	running code
----	-----------	-------	--------------

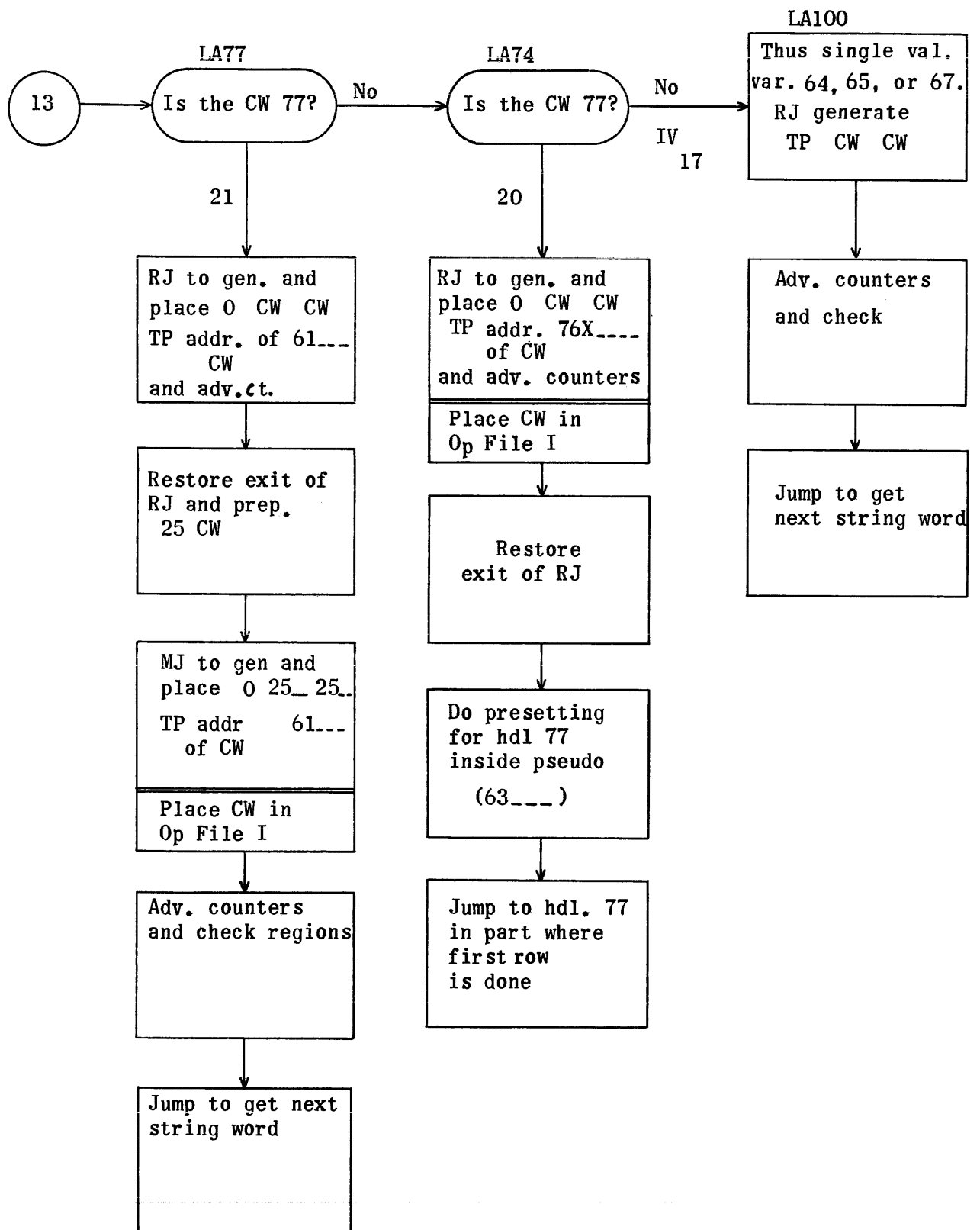
Flow Chart of Compute Generation Routine

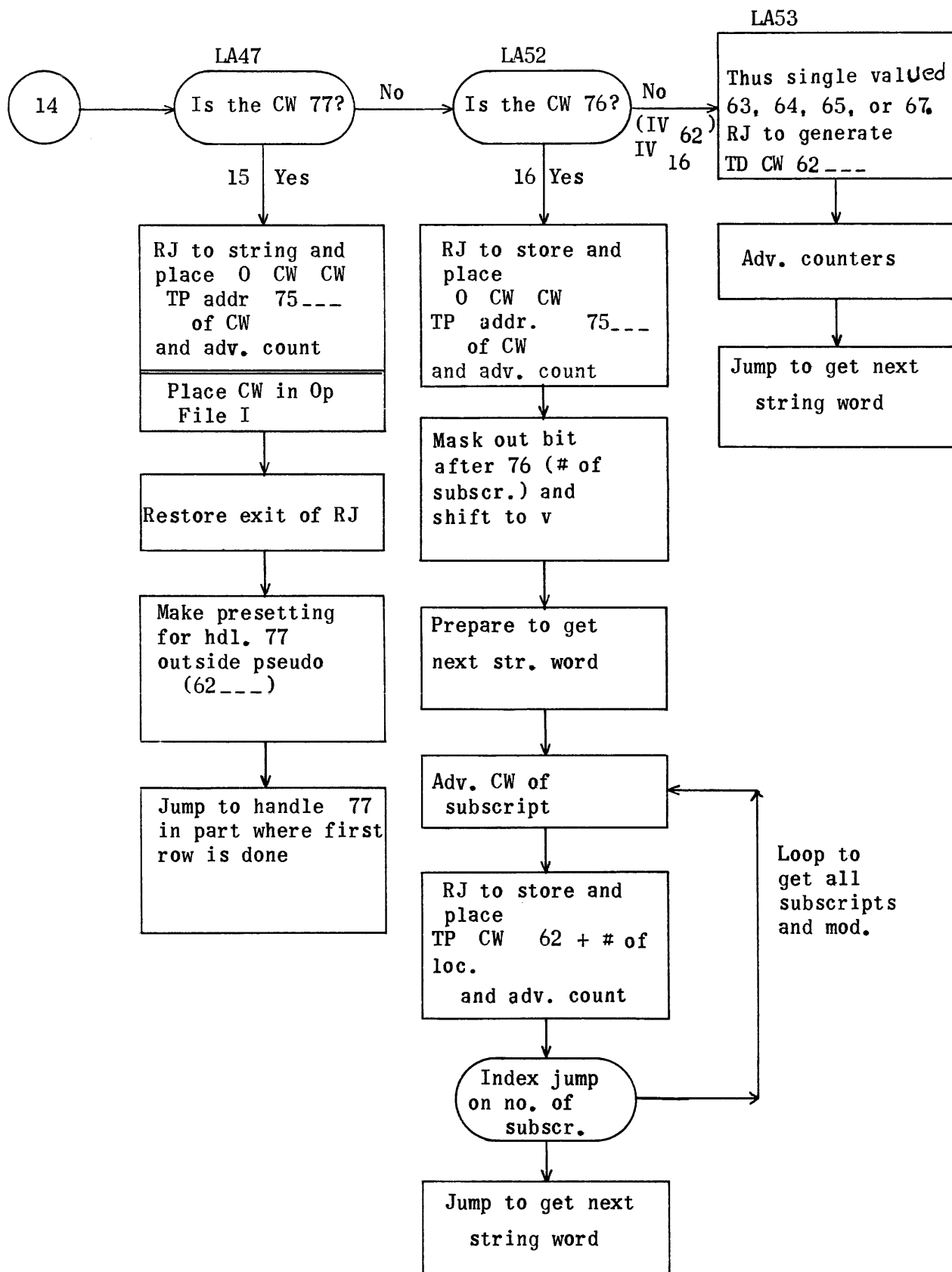




Changed exit of zero string. word
LA 113







Subroutines

I Generate and place
 0 CW CW
 TP addr. [00000 + no. of locat.
 of CW ↑
 first 2 or 3 bits
 filled by the dif-
 ferent entries
 adv. addresses, check
 exceeded regions and jump
 to get next stringout word

III Handle 77 CW cases with dif-
 ferent entries for inside
 and outside pseudo operation,
 and jump to get next string-
 out word

V When zero stringout word,
 place RJ CW CW (was put
 10 0 1
 aside by II) and clear both
 "insides" and adv. resp.
 clear counters and go to
 get next stringout word

II Generate and put aside
 RJ CW CW
 10 0 1
 Advance index "inside some-
 thing" and jump to get next
 string. word

IV Generate and place TP CW
 62___ + no. of location
 Restore subrout. and advance
 counters with exceeded reg.
 check. Only used in RJ
 by III and (16)

VI Check whether CW is already
 in OP File I. When not,
 place it there and adv.
 with exceed. region check
 Only used in RJ by V

Compute Generator Regions

RE	LA2512	Main program	
RE	LB2710	Subroutines	
RE	LC3117	Constants	
RE	LJ3172	Checks for exceeded regions	
RE	LK3213	Alarm print	
RE	LO3217	Alarm (> region LF)	
RE	LS3230	Alarm (> 1777)	
RE	LT3241	Alarm (> region LE)	
RE	LV3251	Alarm (> region LI)	
RE	LP2000	Constant for last addr. + 1 of exit	
RE	LQ5360	Constant for last addr. + 1 of region LF	
RE	LR6777	Constant for last addr. + 1 of region LE	
RE	LU7777	Constant for last addr. + 1 of region	
RE	LD3263	Temporaries	LI should always
RE	LE5360	Generated coding	} follow LE so that in the end when adding the generated con- stants to LE the re- gion never can be ex- ceeded
RE	LI7000	Storage for constants	
RE	LF3300	Op File I	
RE	CW1211	} Generation subroutines used	
RE	BK2242		
RE	OP1047		
RE	WA653		
RE	UP421		
RE	BQ632		
RE	WB677		

	IA	LA		
	0 MJ	0	30000	Exit
	1 RP	10014	LA3	} Clear temporaries
	2 TP	LC	LD	
	3 RP	10005	LA6	} Clear space for prelude
	4 TP	LC	LE	
	5 10	0	1	Const. used by LA14
3	6 SP	BK3	17	} Place sent. CW in prelude and Op File I
	7 TP	A	LE	
	10 TP	A	LF	} Place XS3 code of line number in Op File I
4	11 TP	BK1	LE5	
	12 TP	LC20	LE6	Prepare exit line in prelude
	13 TV	LC6	LD3	} Set starting value in temporaries
	14 TP	LA5	LD14	
	15 TP	LC2	LD5	
	16 TP	LC22	LD4	
5	17 RA	LA20	LC1	} (Pre-advanced) bring next string-out word
6	20 TP	BK3	LD6	
	21 TP	LD6	A	in storage LD6 and to A
7	22 EJ	LC	LB144	Is it zero?
	23 EJ	LC7	LB173	Is it Δ .?
	24 TP	LC	A	} Are we inside something?
	25 TJ	LD	LA46	
	26 TP	LD6	Q	Put string-out word $\rightarrow$ Q
	27 QT	LC23	LD7	Mask out operation code into LD7 and A
	30 EJ	LC23	LA40	CW type 77?
	31 EJ	LC17	LA42	CW type 61?
	32 TJ	LC15	LA35	CW type 4?
	33 RJ	LB36	LB42	Left only CW type 64, 65, 66; jump to set aside RJ CW CW
			10 0 1	
12	34 MJ	0	LA17	Go to get next string-out word
	35 TP	LC2	LD1	Set index "inside pseudo operation"
	36 RJ	LB36	LB32	Jump to set aside RJ CW CW
			10 0 1	} CW4
	37 MJ	0	LA17	Go to get next string-out word
10	40 RJ	LB36	LB37	Jump to set aside RJ CW CW
			10 0 1	CW77
11	41 MJ	0	LB176	Jump to handle 77 CW
	42 TP	LD6	LD13	Save CW 61---in LD13 v address
	43 TV	LC34	LA22	Change exit for zero string-out word to
			LA113	} CW61
	44 TP	LC2	LD	Set "inside something" to 1
	45 MJ	0	LA17	Jump to get next string-out word
9	46 TJ	LD1	LA74	Are we inside pseudo? Yes $\rightarrow$ LA74
14	47 TP	LD6	Q	In parent. but not inside pseudo:
	50 QT	LC23	LD7	mask out Operation Code
	51 EJ	LC23	LA55	CW type 77?
	52 EJ	LC10	LA60	CW type 76?
	53 RJ	LB133	LB116	Left only type 63, 64, 65 or 67: go to
				generating with 62...
	54 MJ	0	LA17	

15	55	RJ	LB14	LB21	RJ to gener. and placing 0 CW CW TP Addr. 75--- of CW	} CW Type 77 (not inside pseudo)
	56	RJ	LB143	LB134	RJ to taking care of Op File I	
	57	MJ	0	LB112	Jump to handle subscr. var. case	
16	60	RJ	LB133	LB167	RJ to gener. and placing 0 CW CW TP Addr. 75--- of CW	} CW type 76 (no-- never!-- inside Pseudo- Op)
	61	TP	LC21	Q	} Mask out bit # of subscripts # of subscr. in v of LD 10	
	62	QT	LD6	LD10		
	63	LQ	LD10	36	} Prepare to get next string- out word	} Subscr. -1 (zero jump needed since index by 1 too high)
	64	TP	LC46	Q		
	65	QT	LD6	A		
17	66	AT	LC15	LD6	Bring CW-1 of first sub- script in storage LD6	} Subscr. -1 (zero jump needed since index by 1 too high)
	67	MJ	0	LA70	Space filling jump (free)	
19	70	RA	LD6	LC2	Up date CW	
	71	RJ	LB133	LB116	Jump to generate and place TP [CW] 62...+ # of locat.	} Subscr. -1 (zero jump needed since index by 1 too high)
	72	IJ	LD10	LA73	IJ on # of subscr.	
	73	ZJ	LA70	LA17	Jump to get next string. word	
13	74	TP	LD6	Q	} Operat. code → A and LD7	} In parenthe- ses inside pseudo op.
	75	QT	LC23	LD7		
	76	EJ	LC23	LA102	CW type 77?	
	77	EJ	LC12	LA105	CW type 66?	} CW type 77 in- side pseudo
	100	RJ	LB133	LB160	Left only CW type 64, 65, or 67	
	101	MJ	0	LA17	Go to get next string. word	
20	102	RJ	LB14	LB23	RJ to gener. and place 0 CW CW TP Addr. 76X... of CW	} CW type 66 inside pseudo
	103	RJ	LB143	LB134	RJ to take care of Op File 1	
	104	MJ	0	LB114	Jump to handle subscr. var. case	
21	105	RJ	LB14	LB15	RJ to gener. and place 0 CW CW TP Addr. 61... of CW	} CW type 66 inside pseudo
	106	TP	LC23	Q	} Put mask for CW code — Q Change CW in LD6 to sent. CW25...	
	107	QS	LC14	LD6		
	110	RJ	LB14	LB15	RJ to gener. and place 0 CW CW TP Addr. 61... of CW	

18

111	RJ	LB143	LB134	RJ to take care of Op File I	
112	MJ	0	LA17	Jump to get reset string- out word	
113	RA	LD13	LC2	Adv. CW by 1	
114	SP	LD13	17	Move CW + 1 to u addr.	
115	TU	A	LC30	} Place CW in u of $\begin{matrix} TU \\ TV \end{matrix} \begin{bmatrix} \\ \end{bmatrix} \begin{bmatrix} \\ \end{bmatrix}$	
116	TU	A	LC31		
117	TP	LC24	A		
120	AT	LD5	A	Put "1003" in v of A } form excl. 10 lines "next addr.+ 2"	
121	TV	A	LC30	} Add # of addr. in gener. coding excl. 10 lines	Case
122	TV	A	LC31		CW
123	MJ	0	LA167		61...
124	RP	30004	LA126	} Place "next addr. + 2" (excl. 10 lines) in v of $\begin{matrix} TU \\ TV \end{matrix} \begin{bmatrix} \\ \end{bmatrix} \begin{bmatrix} \\ \end{bmatrix}$	
125	TP	LC30	30000		
126	TV	LC36	LA22		
127	RA	LD5	LC5	} Place 4 rows of generated coding	
130	RJ	LJ12	LJ7		
131	RP	10003	LA133		
132	TP	LC	LD	} Restore exit of case zero string. word	
133	MJ	0	LA17		
134	TV	LD3	LE		
135	RS	LE	LC37	} Adv. counters resp. clear # of locat. rel. 100 and both insides	
136	TV	LE	LE1		
137	RS	LE1	LC43		
140	RS	LD4	LC22	} Fill v of LE (LE7 +...) - LE	
141	TV	LD4	LE2		
142	LQ	LE2	11		
143	RS	LB140	LC42	} Fill v of LE1 (LE7 +...) - LE - 6 (6 is prelude coding)	
144	TV	A	LF		
145	RA	LE6	LD5		
146	RA	LE6	LC2	} Fill v of LE2	
147	ST	LC20	LF1		
150	RS	LF1	LC2		
151	SP	LE2	6	} Fill v of LF	
152	AT	LA162	LA162		
153	TV	LE	LA163		
154	RA	LA163	LC37	} Fill v of LE6 (second addr. of next routine)	
155	LQ	LE2	Q33		
156	RA	LE	Q		
157	RA	LE1	Q	} Fill v of LF1 (LE6-LC20)	
160	RA	LF1	Q		
161	RA	LE6	Q		
162	RP	30000	LJ	} Prepare moving of generated constants	
163	TP	LI	30000		
164	TP	LC44	OP1		
165	RJ	OP	OP2	} Add constants to # of generated addresses	
166	MJ	0	LA		

22
23

24
8

Go to
end

167	SP	LC30	17	Come from LA123	} Patch for generating RA addr. 1
170	TU	A	LC32		
171	TP	LC2	A	} Get a CW for const. 1	
172	RJ	CW	CW1		
173	TV	Q	LC32		
174	TV	LD3	LA125		
175	MJ	0	LA124	Jump back to coding	}
	CA	LA176			

		IA	LB		
I	0	SP	LD6	17	} Form $\emptyset$ CW CW in next storage addr. save CW in u in LC36 (used by LB135 in VI)
	1	AT	LD6	LI	
	2	TU	A	LC36	
	3	MJ	0	LJ15	
	4	TU	LD4	LC26	} Generate TP 01... CW + # of locat. and place it for gener. const. 0 CW CW for gener. code TP 01...CW + #loc.
	5	RA	LC26	LD2	
	6	TV	LD3	LB7	
	7	TP	LC26	30000	
	10	RA	LD2	LC2	} Advance counters
	11	RJ	LJ12	LJ11	
	12	RA	LD4	LC27	
	13	RA	LD5	LC2	
I exit	14	MJ	0	30000	
I ₆₁	15	TV	LC17	LC26	Put 61000 in v of LC26
	16	MJ	0	LB	
I ₆₂	17	TV	LC16	LC26	Put 62000 in v of LC26
	20	MJ	0	LB	
I ₇₅	21	TV	LC11	LC26	Put 75000 in v of LC26
	22	MJ	0	LB	
I _{76_x}	23	TU	LA20	LB25	} Get next string-out word (in advance, not yet officially) into LD7
	24	RA	LB25	LC1	
	25	TP	30000	LD7	
	26	SP	LD7	6	Shift # of subscripts 9 ₁₀ bits to the left
	27	TV	A	LC26	} Put 76X000 in v of LC26
	30	RA	LC26	LC10	
	31	MJ	0	LB	
II	32	SP	LD6	17	} Form RJ CW CW } and set it aside 10 $\emptyset$ 1
	33	AT	LD6	A	
	34	AT	LC35	LD13	
	35	TP	LC2	LD	Set index "inside something" to 1
II exit	36	MJ	0	30000	Only used in RJ
II ₂₄	37	TP	LC23	Q	} Make 24...CW before going to II
	40	QS	LC13	LD6	
	41	MJ	0	LB32	
II ₂₅	42	TP	LC23	Q	} Make 25...CW before going to II
	43	QS	LC14	LD6	
	44	MJ	0	LB32	
III	45	RP	10004	LB47	} Clear temp. storage Handl. of subscr. var.
	46	TP	LC	LD7	
	47	RJ	LB111	LB106	Get next string-out word $\rightarrow$ A
	50	TU	A	LD10	Modulus $\rightarrow$ u of LD10

	51	TV	A	LD7	} # of subscr. -1 → v of LD7
	52	RS	LD7	LC2	
	53	SP	LD10	71	
	54	RJ	CW	CW1	} Get const. CW for mod.
	55	TP	A	LD10	
	56	IJ	LD7	LB60	CW for mod. → u of LD10
	57	MJ	0	LB103	First IJ on # of subscripts
	60	RJ	LB111	LB106	Get next string-out word → A
	61	TU	A	LD12	Store multiplier 2 in LD12(u)
	62	TV	A	LD11	Store multiplier 1 in LD11(v)
	63	TP	LD11	A	} Get CW for multiplier 1
	64	RJ	CW	CW1	
	65	TU	A	LC26	} Generate and place TP [CW mult. 1] ^{62...} _{63...# of}
	66	RJ	LB133	LB121	
	67	IJ	LD7	LB71	for multipl. 1
	70	MJ	0	LB103	Second IJ on # of subscr.
	71	SP	LD12	71	} Get const. CW for mult. 2
	72	RJ	CW	CW1	
	73	TU	A	LC26	} Generate and place TP [CW mult. 2] ^{62...} _{63...# of}
	74	RJ	LB133	LB121	
	75	IJ	LD7	LB77	Third IJ on # of subscr.
	76	MJ	0	LB103	locat.
	77	RJ	LB111	LB106	Get next string-out word → A
	100	RJ	CW	CW1	Get const. CW for mult. 3 (this string
					word has data only in v)
	101	TU	A	LC26	} Generate and place TP [CW mult. 3] ^{62...# of} _{63...# of}
	102	RJ	LB133	LB121	
	103	TU	LD10	LC26	for multipl. 1
	104	RJ	LB133	LB121	} Generate and place TP [CW modulus] ^{62...# of} _{63...# of}
	105	MJ	0	LA17	
	106	RA	LA20	LC1	Go to get next string-out word
	107	TU	LA20	LB110	} Get next string-out word to A
	110	TP	30000	A	
	111	MJ	0	30000	(used as subrout. in III)
III ₆₂	112	TV	LC40	LB121	Set jump to IV ₆₂ in IV
	113	MJ	0	LB45	
III ₆₃	114	TV	LC41	LB121	Set jump to IV ₆₃ in IV
	115	MJ	0	LB45	
IV ₁₆	116	SP	LD6	17	} Prepare IV ₁₆
	117	TU	A	LC26	
	120	MJ	0	LB122	Generate and place TP CW
IV	121	MJ	0	30000	Jump to right entry
IV ₆₂	122	TV	LC16	LC26	Place 62...in LC26
	123	MJ	0	LB125	
IV ₆₃	124	TV	LC15	LC26	Place 63...in LC26
	125	RA	LC26	LD2	Add # of locat. to ⁶² ₆₃ ... in LC26 v address
	126	TV	LD3	LB127	Place "next addr. for gen. coding" in NI
	127	TP	LC26	30000	Place one line of generated coding
	130	RA	LD2	LC2	} Adv. counters and check exceeded region
	131	RJ	LJ12	LJ11	
	132	RA	LD5	LC2	
IV	133	MJ	0	30000	
VI exit	134	TP	LC25	Q	Mask CW at hand → A in u address

	135	QT	LC36	A	}	
	136	RP	20000	LB140	}	
	137	EJ	LF2	LB143	}	Compare whether already in Op File 1
	140	TP	A	LF2		When not yet in Op File 1, put CW in it (TP because space is not cleared before!)
	141	MJ	0	LJ3	}	Advance addresses (1 in v)
	142	RA	LB136	LC1	}	Advance addresses (1 in u)
VI	exit	143	MJ	0		Used only in RJ
V		144	TV	LD3		
		145	RP	30002		
		146	TP	LD13		Place the 2 rows that have { RJ CW CW been set away { 10 0 1
		147	RA	LD5		
		150	RJ	LJ12		Adv. counters
		151	RP	10003		
		152	TP	LC		Clear all "inside's" and counter for number of addr. in array
		153	TU	LD13		Put CW in u of LC36
		154	RJ	LB143		RJ to take care of Op File 1
		155	MJ	0		Jump to get next string-out word
I ₆₃		156	TV	LC15		
		157	MJ	0		Prepare I ₆₃
IV ₁₇		160	SP	LD6		
		161	TU	A		Prepare IV ₁₇
		162	MJ	0		
		163	TP	LA34		Shortcut (skip everything up to next zero word)
		164	RJ	LA22		
		165	TP	LC45		Restore LA23
		166	MJ	0		Jump to handle zero word situation
		167	SP	LD6		
		170	TU	A		
		171	TV	LC11		Place MJ in end
		172	MJ	0		
		173	TP	LC20		
		174	RJ	LB133		
		175	MJ	0		
		176	TP	LC46		Mask for last 3 bits, come from LA41
		177	RA	LA20		Advance to next string-out word
		200	TU	LA20		
		201	QT	30000		Mask out # of subscripts
		202	EJ	LC2		Is it 1?
		203	TJ	LC5		Is it < 4?
		204	RA	LA20		It is 4 : advance once more
		205	RA	LA20		It is 2 or 3: advance once more
		206	MJ	0		It is 1 : go to get next string-out word
			CA	LB207		

Constants

	IA	LC	
0	0	0	0
1	0	1	0
2	0	0	1
3	0	0	2
4	0	0	3
5	0	0	4
6	0	0	LE7
7	01	22777	77777
10	0	0	76000
11	0	0	75000
12	0	0	66000
13	0	0	24000
14	0	0	25000
15	0	0	63000
16	0	0	62000
17	0	0	61000
20	MJ	0	01000
21	0	0	00700
22	0	10000	10000
23	0	0	77000
24	0	0	1003
25	0	77777	0
26	TP	30000	30000
27	0	1	1
30	TU	30000	30000
31	TV	30000	30000
32	RA	30000	30000
33	RJ	30000	30000
34	0	0	LA113
35	RJ	0	0
36	0	30000	LB144
37	0	0	LE
40	TU	0	LB122
41	TV	0	LB124
42	TP	A	LF
43	0	0	6
44	0	LE	LF
45	EJ	LC7	LB173
46	0	0	7
47	TP	A	LQ
50	0	0	LR
51	MJ	0	LP
52	AT	LD6	LU
	CA	LC53	

XS3 "△."

These four rows belong together
(the 30000 must not be messed up)

This instr.-const. must have zero's !!!

For exceeded region checks

Exceeded Region Tests

	IA	LJ		
0	TJ	LC51	LA164	} Gen. constants
1	TU	LS	LK1	
2	MJ	0	LK	Alarm
3	RA	LB140	LC2	} Region LE
4	TJ	LC47	LB142	
5	TU	LT	LK1	
6	MJ	0	LK	Alarm
7	RA	LD3	LC3	} Region LF
10	RA	LD3	LC2	
11	RA	LD3	LC2	
12	TJ	LC50	30000	
13	TU	LO	LK1	
14	MJ	0	LK	Alarm
15	RA	LB1	LC2	} Region LI
16	TJ	LC52	LB4	
17	TU	LV	LK1	
20	MJ	0	LK	Alarm
	CA	LJ21		

Alarm Entrance

	IA	LK	
0	RJ	WA	WA2
1	TP	30000	UP3
2	RJ	UP2	UP
3	MJ	0	BQ6
	CA	LK4	

Alarm 1

	IA	LS		
0	0	LS1	0	
1	40	LS2	7	
2	32	30503	05424	Generated code
3	66	30270	12651	exceeds 1777
4	27	30013	07226	addresses.
5	30	30276	50104	
6	12	12120	12427	
7	27	54306	56530	
10	65	22777	77777	
	CA	LS11		

Alarm 2

	IA	LT		
0	0	LT1	0	
1	40	LT2	6	
2	54	30323	45150	Region for Op File 1
3	01	31515	40151	is exceeded.
4	52	01313	44630	
5	01	04013	46501	
6	30	72263	03027	
7	30	27227	77777	
	CA	LT10		

Alarm 3

	IA	LO		
0	0	LO1	0	
1	40	LO2	7	
2	54	30323	45150	Region for generated
3	01	31515	40132	coding is exceeded.
4	30	50305	42466	
5	30	27012	65127	
6	34	50320	13465	
7	01	30722	63030	
10	27	30272	27777	
	CA	LO11		

Alarm 4

	IA	LV	
0	0	LV1	0
1	40	LV2	10
2	54	30323	45150
3	01	31515	40132
4	30	50305	42466
5	30	27012	65150
6	65	66245	06665
7	01	34650	13072
10	26	30302	73027
11	22	77777	77777
	CA	LV12	

Region for generated
constants is exceeded.

Temporaries				Compute Generate	
LD	0			Index "inside something"	}
	1			Index " pseudo oper.	
	2		[]	Running # adv. by IV(+ 1), 18(= 0), I(+ 1)	}
	3		[]	Addr. of gen. code adv. by	
				V(+ 2), IV(+ 1), 18(+ 4), I(+ 1)	
	4	∅	10000 10000	CW of gener. const. adv. by	
				I(+ 1 in u and v)	}
	5	∅	∅ 1	adv. by V(+ 1)	
				IV(+ 1), 18(+ 4), I(+ 1)	}
	6			Next string-out word	
	7			Working space	}
	10			Working space	
	11			Working space	
	12			Working space	
	13			Storage space for {RJ CW CW	}
				10 ∅ 1	
	14	10	∅ 1		

Generated Code:

LE	0	∅	sent.CW	∅	}
	1				
	2				
	3				
	4				
	5	Line #	XS3 code		}
	6	MJ	∅ 01000	Set by LA12 and LA160	
	:				

OP File I

LF	0	∅	sent.CW	∅	}
	1				
	2				
	3				
	4				

Generated Constants

LI	0
	1
	.
	.
	.

Vary Generation Routine

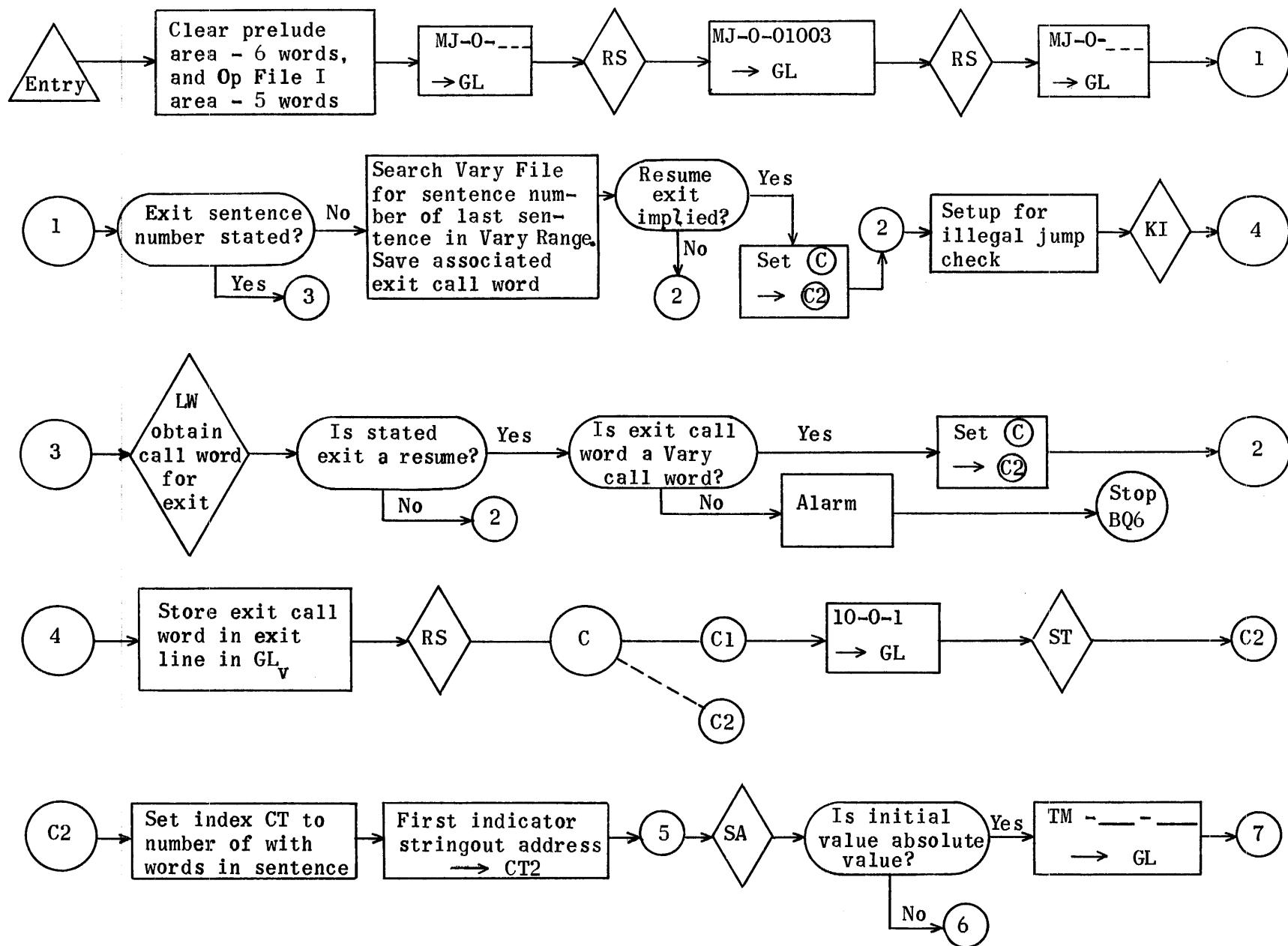
The Vary generator uses the string-out and the Vary File as built during the Translation Phase to prepare the relative coding necessary to perform the functions of the input Vary sentence. This generator also provides an Op File I item to be used by the Segmentation Phase. The relative coding is later modified by the Processor according to the information furnished by the Allocator.

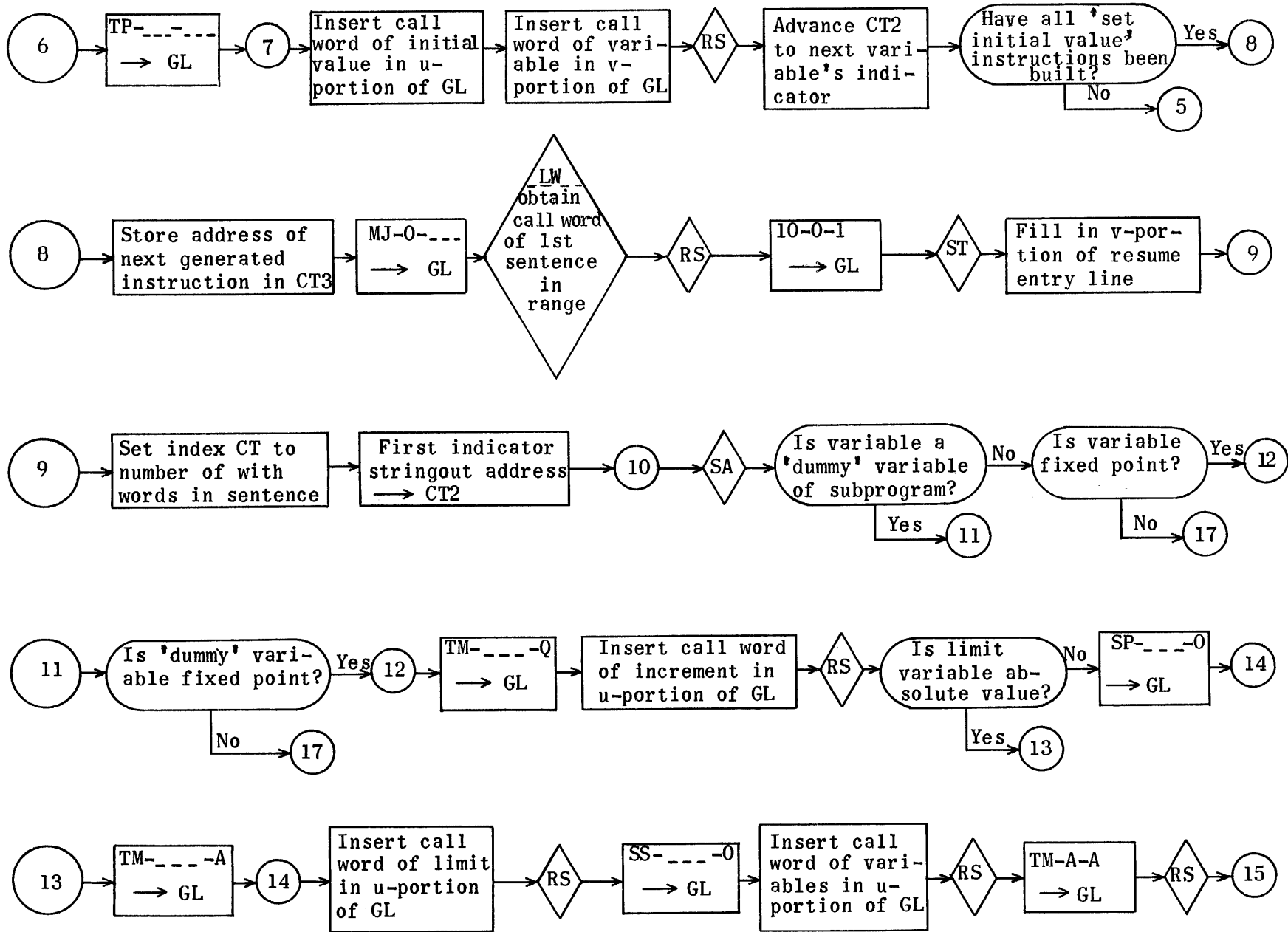
The generated Vary coding may be broken down into the sequence of functions performed by the Vary sentence.

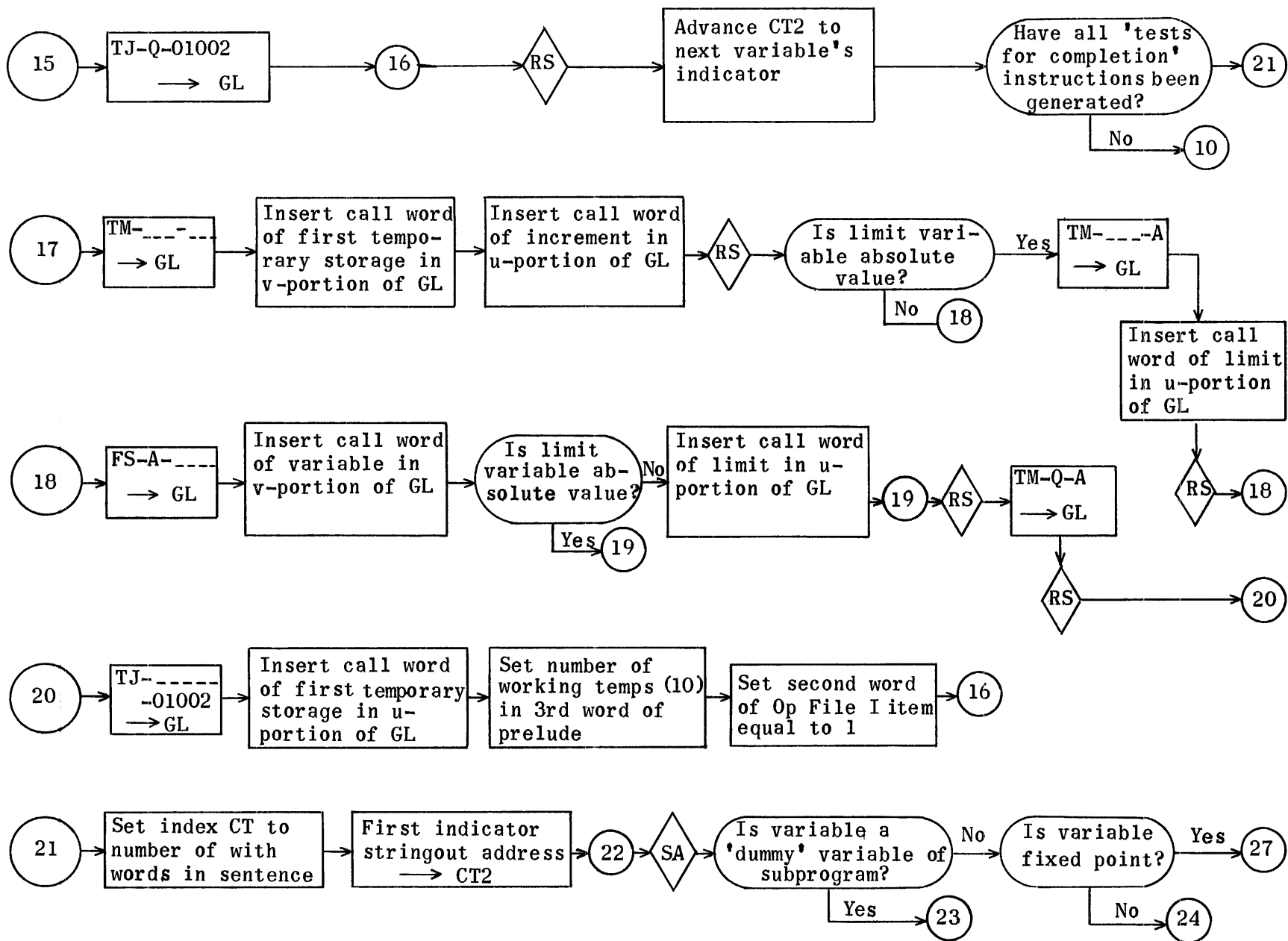
- | | | |
|---|--|---|
| ① | MJ - 00000 -[4] | Resume entry |
| | MJ - 00000 -[② or ⑤] | Normal entry |
| | MJ - 00000 -[] | Exit as specified in string-out
or Vary File |
| ② | { Set initial value(s) } | |
| ③ | { Jump to first statement
in range } | |
| ④ | { Tests for completion of
Vary. If finished, to ① } | |
| | { Modify variable(s) } | |
| | { Back to ③ } | |
| ⑤ | { Tests for indefinite
loop, if required } | |
| | { Back to ② } | |

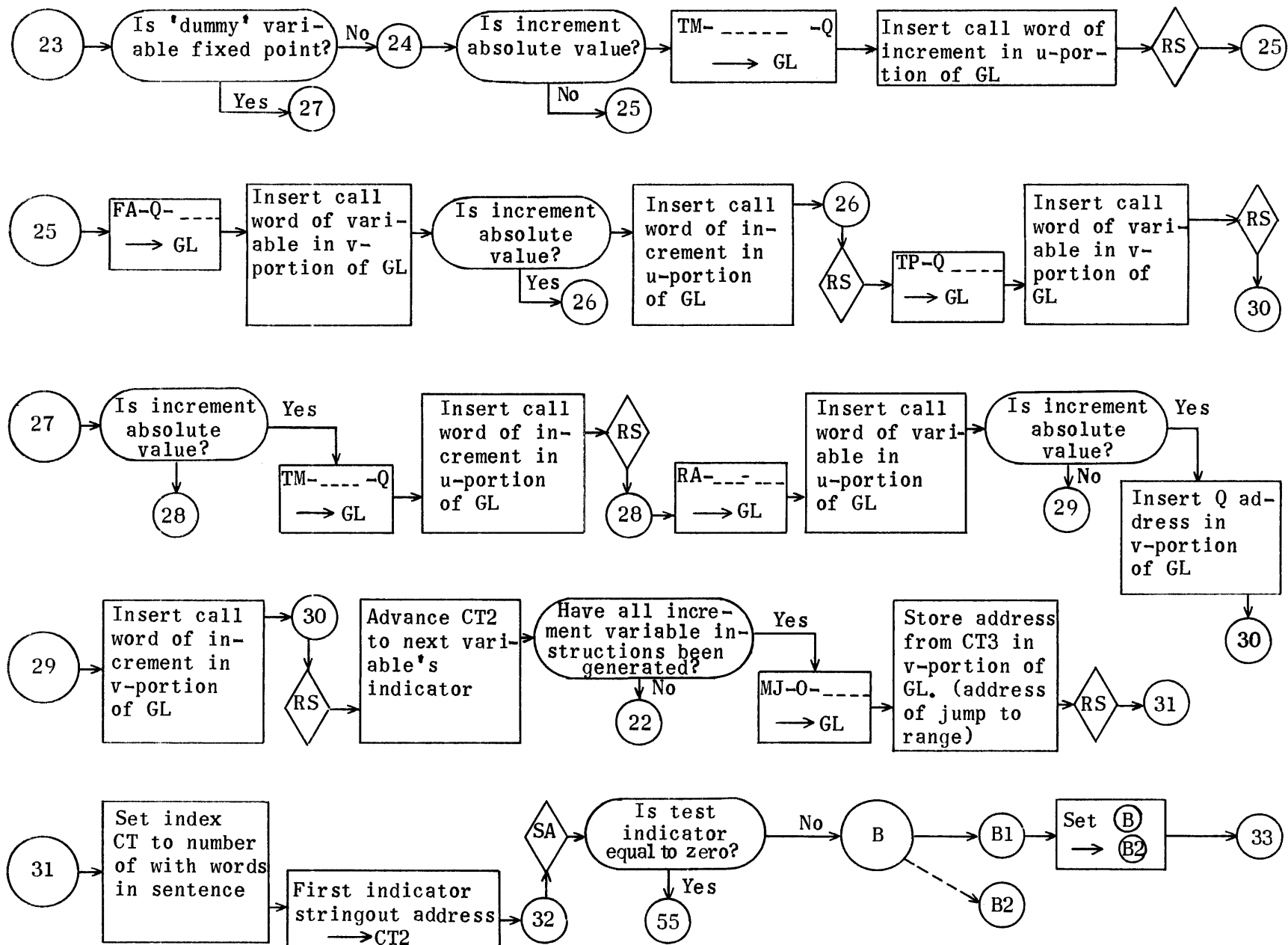
The call word from the u-portion of the Vary File item is filled into the Exit instruction by the Vary generator when the transfer component is implied rather than stated.

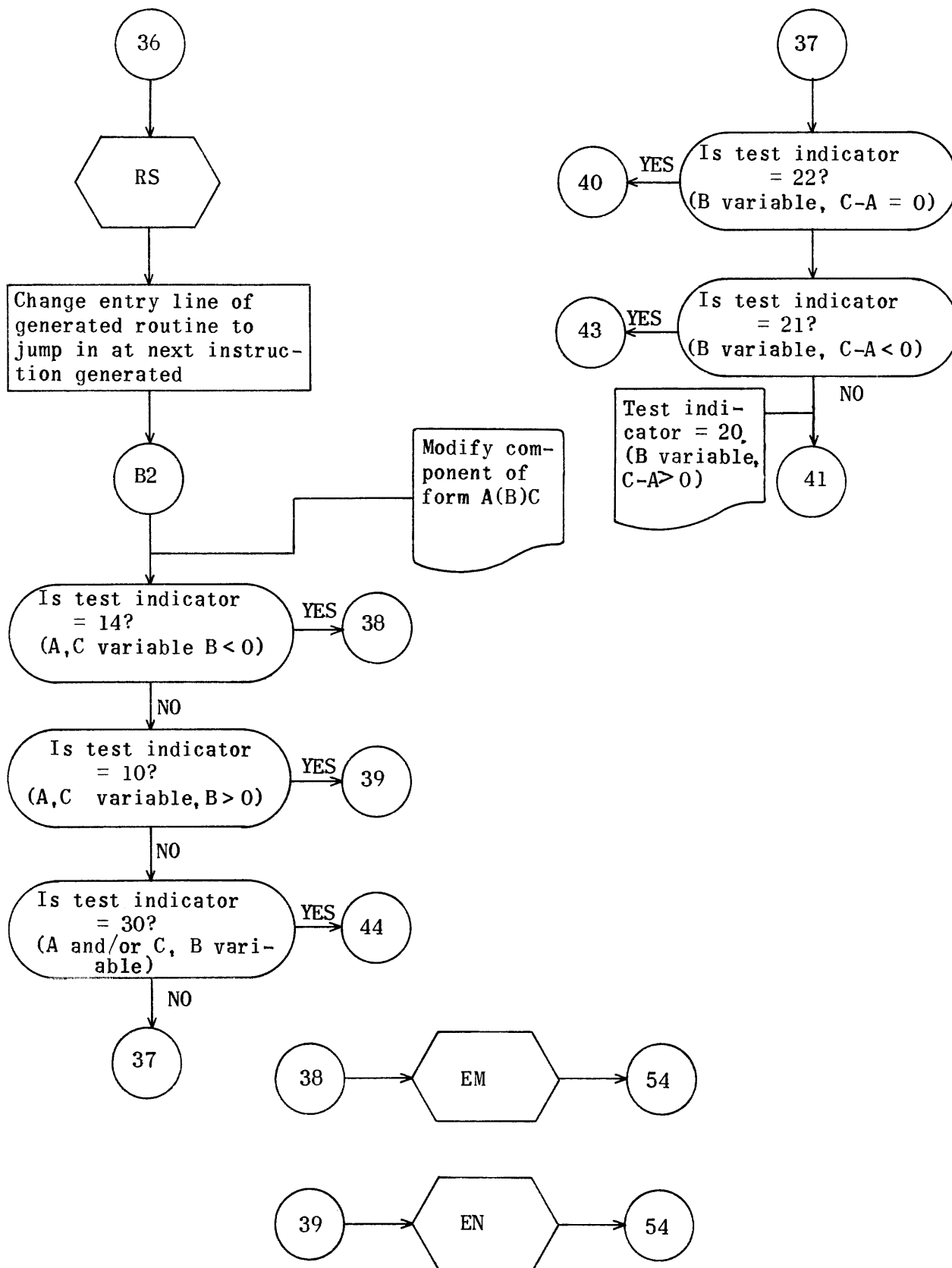
The return to the Vary coding from the last sentence in the Vary's range is accomplished by the insertion of the Vary call word (from the v-portion of the Vary File item) into the Exit line of the generated coding for the last sentence in the range. This insertion is effected by the RG routine of the Generation subroutines. The Exit line of the last sentence in the Vary range will return to the Vary coding at the Resume Entry line.

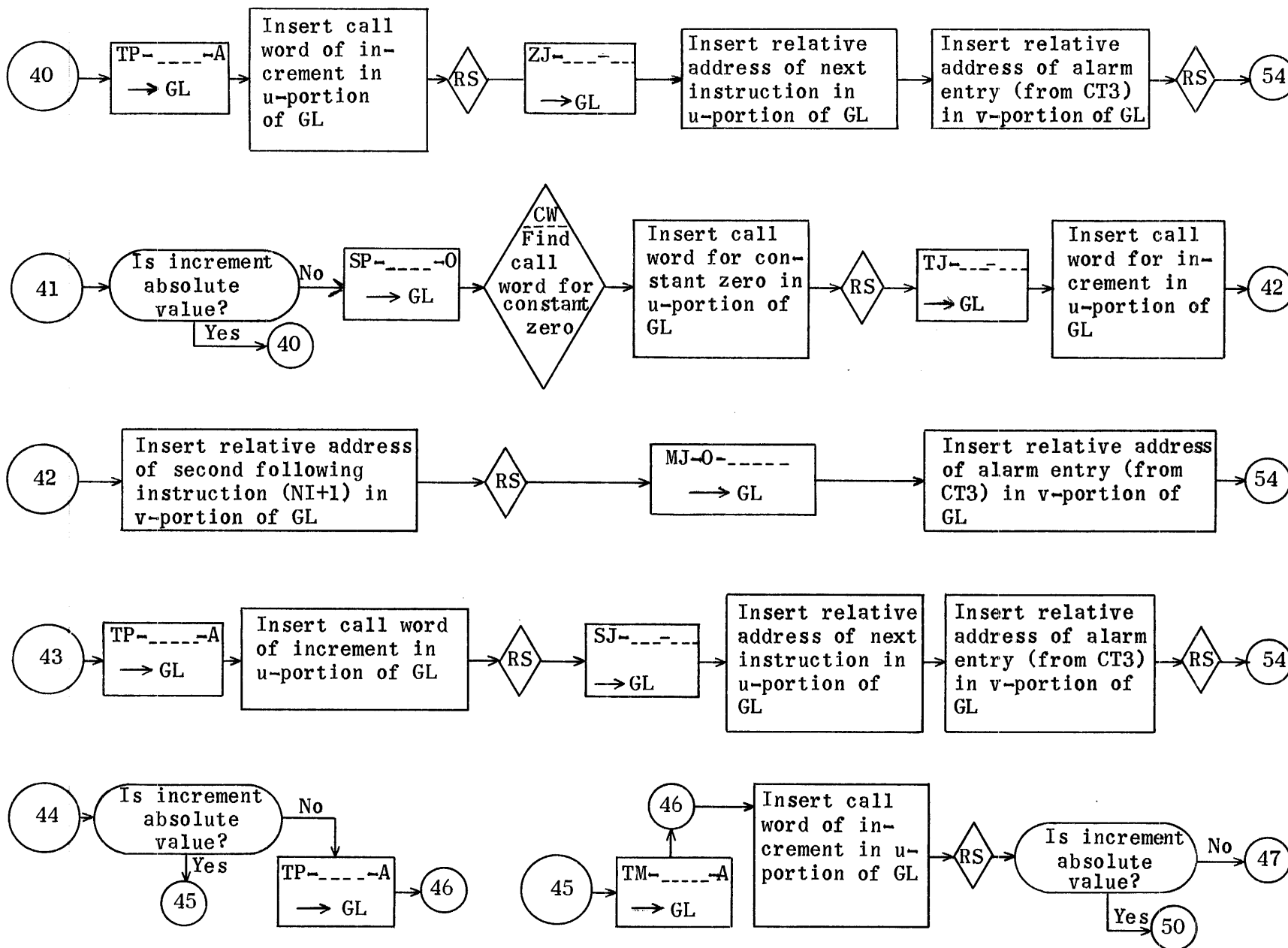


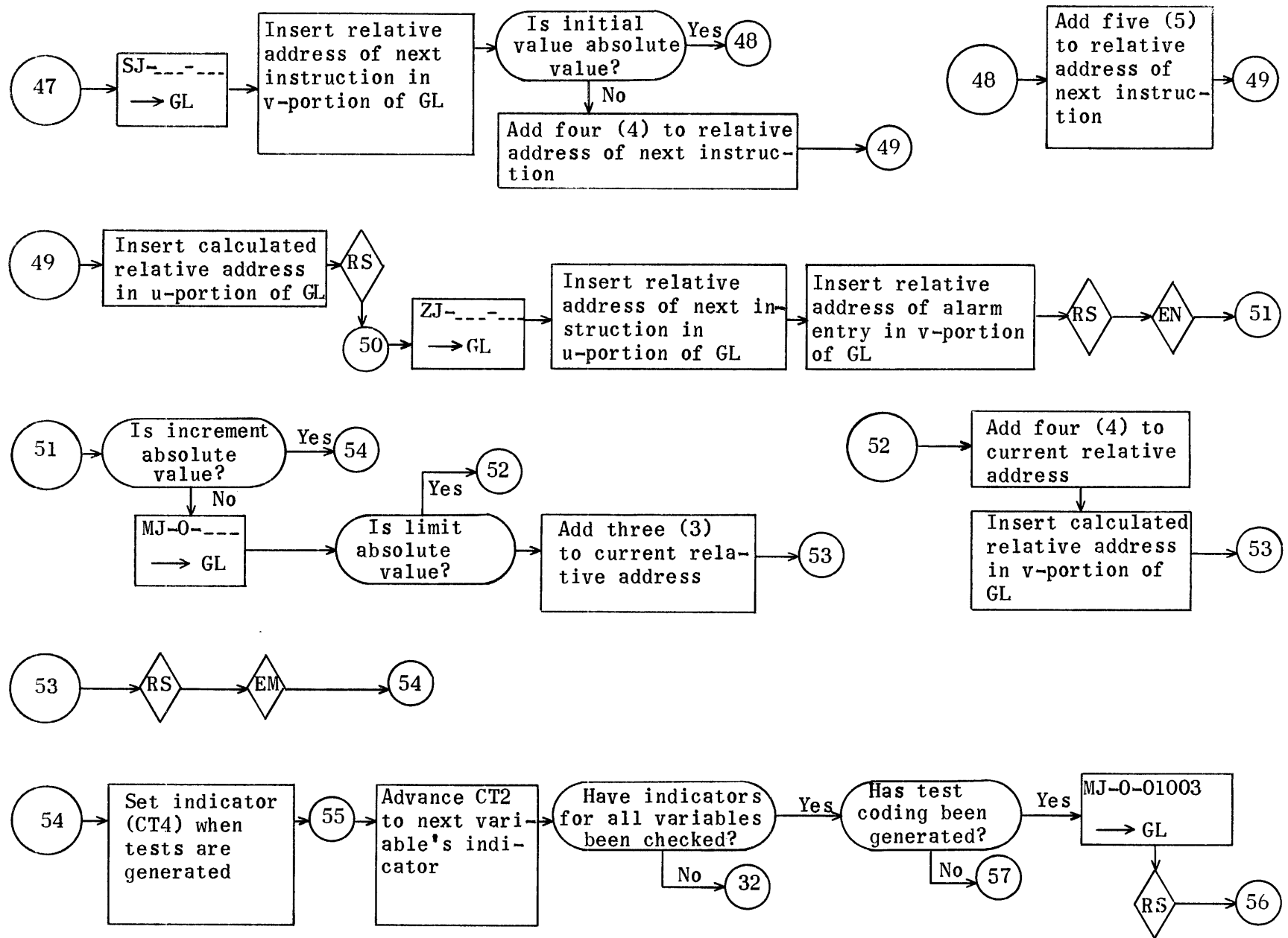


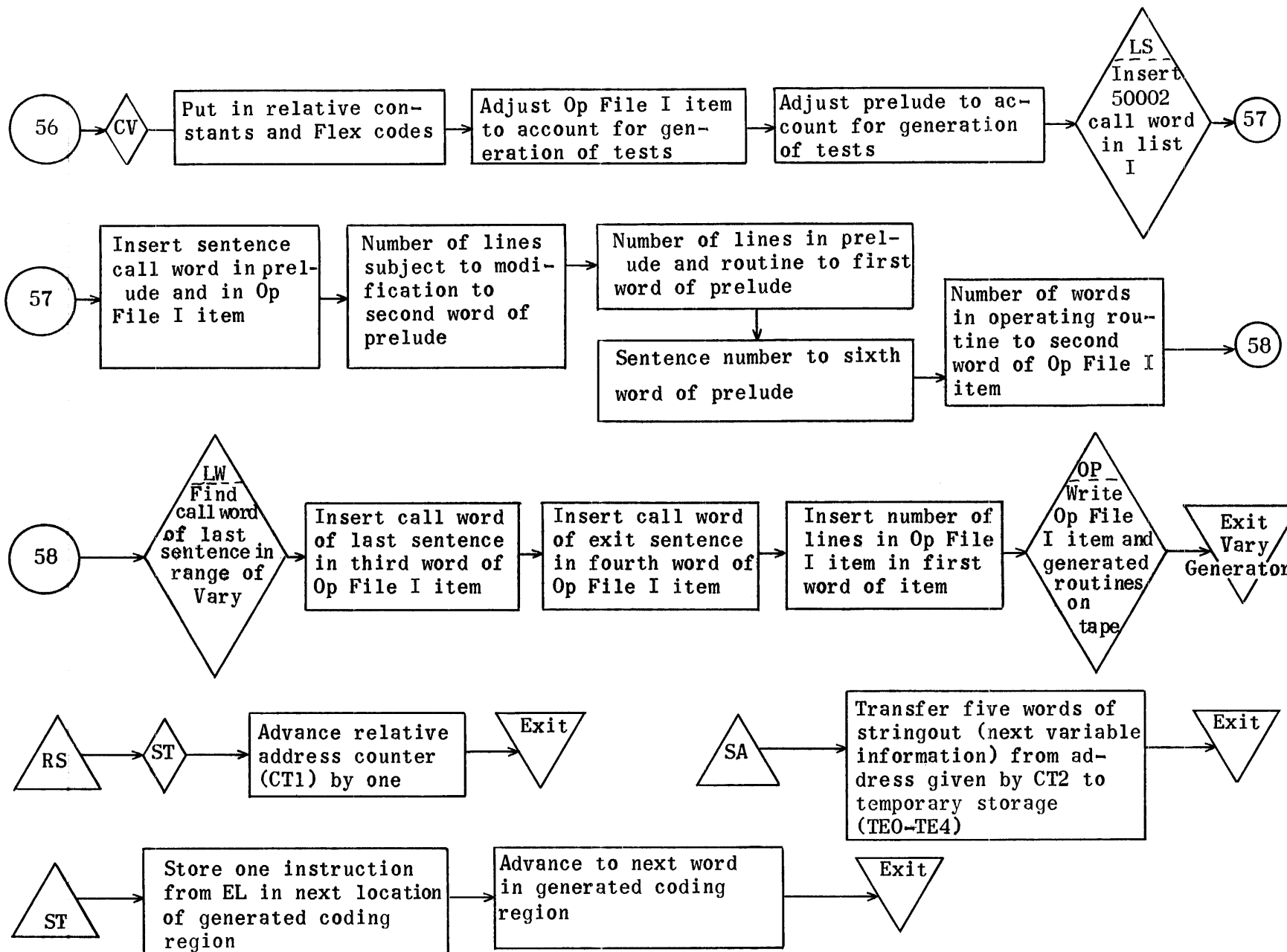


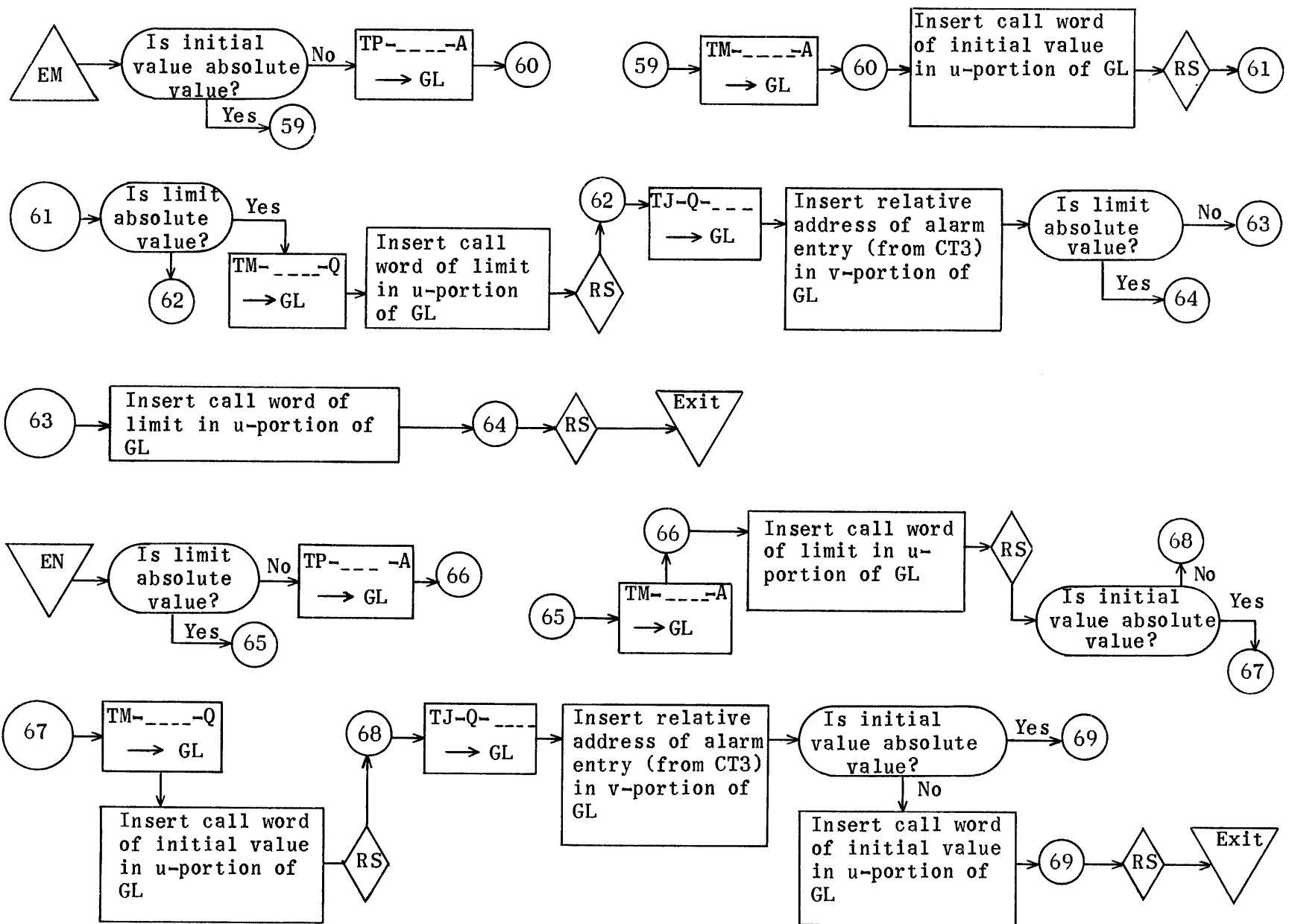


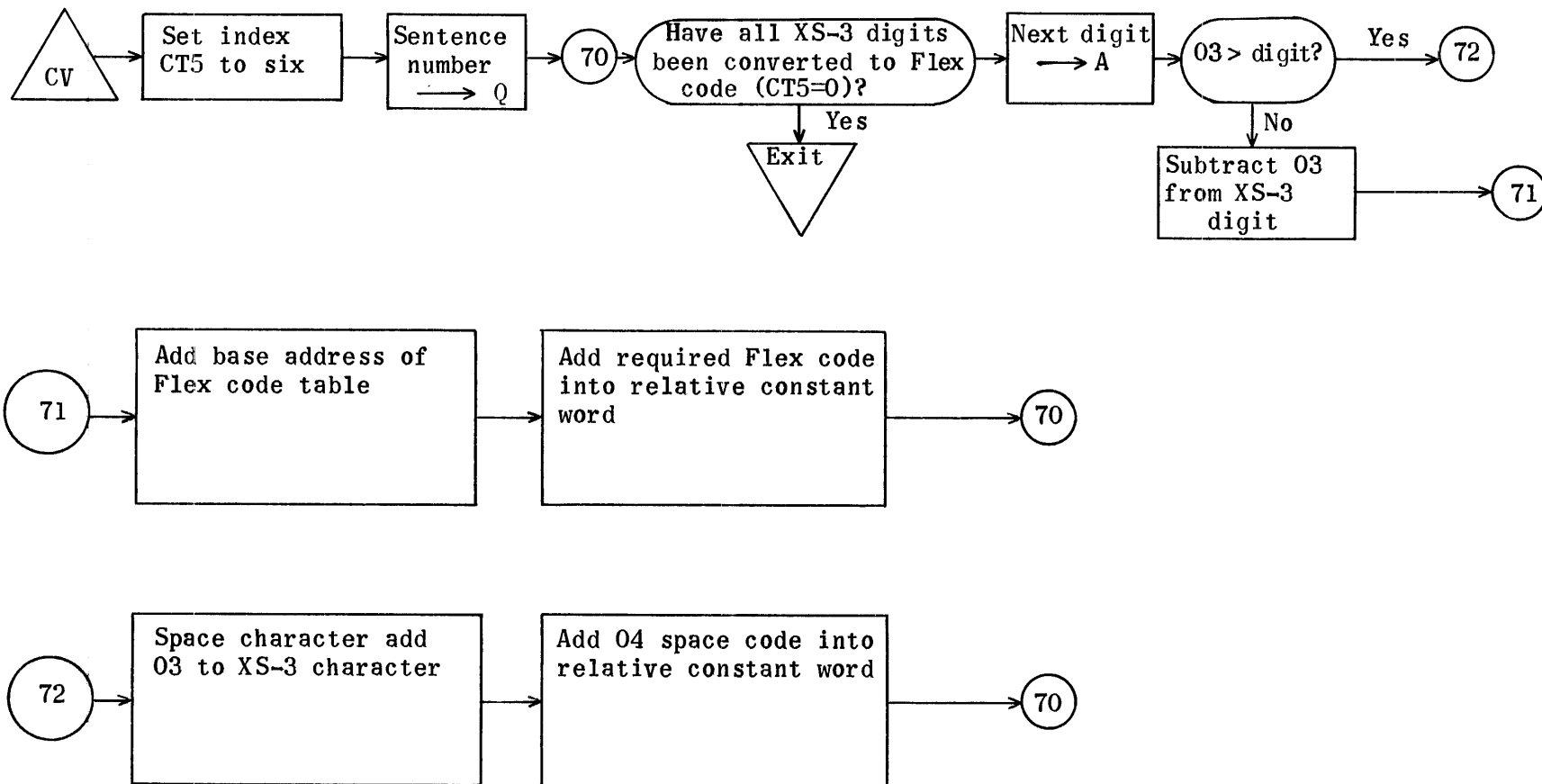












To convert XS-3 sentence number to Flex code, insert in relative constants

Vary Generator Regions

RE	VY2512	(1)
RE	IN2513	(4)
RE	VA2517	(50)
RE	VE2567	(40)
RE	CE2627	(21)
RE	VB2650	(23)
RE	FX2673	(24)
RE	FL2717	(27)
RE	VC2746	(12)
RE	FG2760	(16)
RE	FD2776	(22)
RE	VT3020	(51)
RE	DJ3071	(37)
RE	DK3130	(12)
RE	DL3142	(35)
RE	DM3177	(13)
RE	EM3212	(23)
RE	EN3235	(23)
RE	VS3260	(24)
RE	BP3304	(27)
RE	CV3333	(20)
RE	RS3353	(3)
RE	ST3356	(3)
RE	SA3361	(4)
RE	CN3365	(73)
RE	RC3460	(13)
RE	CT3473	(6)
RE	GL3501	(1)
RE	TE3502	(5)
RE	PF3507	(5)

RE	CW1211	Obtain CW for constant
RE	LS1465	Insert Library CW in List I
RE	OP1047	Write Generated routine and Op File I on tape
RE	BR537	Compiler of computer error routine
RE	KI1336	Illegal jump check routine
RE	LW1250	Sentence CW locating routine
RE	UP421	On line print routine
RE	WA653	Print heading routine
RE	BQ632	To rewind tapes
RE	BK2242	Sentence string-out
RE	PL5360	Op File I
RE	GC5366	Generated coding
RE	VF47101	Vary File

	IA	VY		
0	MJ	0	30000	Exit line
	CA	VY1		
	IA	IN		
0	RP	10006	IN2	Zeroize Prelude area
1	TP	CN	PL	
2	RP	10005	IN4	Zeroize Op File I area
3	TP	CN	PF	
4	MJ	0	VA	{VA0 = IN4}
	CA	IN5		

①

VA 0	IA	VA	
	TP	CN34	GL
1	RJ	RS2	RS
2	TP	CN34	GL
3	RJ	RS2	RS
4	TP	CN34	GL
5	SP	BK7	0
6	ZJ	VE	VA7
7	TU	VF	VA11
10	SP	BK6	0
11	RP	30000	BR3
12	EJ	[VF1]	VA13
13	SP	VA11	0
14	LQ	Q	17
15	TU	Q	VA11
16	SS	Q	0
17	SA	VA12	0
20	TU	A	VA12
21	TU	A	VA25
22	TU	A	VA30
23	TU	A	VA31
24	TP	CN15	Q
25	QT	30000	A
26	EJ	BK3	VA30
27	MJ	0	VA10
30	TU	30000	CT
31	SP	30000	45
32	SJ	VA33	VA34
33	TV	CN31	CE1
34	LT	10024	Q
35	TV	Q	CT
36	TP	CN55	Q
37	SP	BK3	0
40	TJ	CN54	VA43
41	QT	CT	A
42	MJ	0	VA45
43	QT	CT	A
44	TP	CN34	Q
45	RJ	KI	KI1
46	TV	CT	GL
47	MJ	0	CE
	CA	VA50	

②

④

MJ 0 - → GL
Store
MJ 0 1003 → GL
Store
MJ 0 - → GL

Is exit sentence number stated?

No, so search Vary file for sentence
number of last sentence in range

Add r to VF1 to find Exit call word

Set to continue search

Set up

Vary call word for this Vary File
item → A

Has the cor. Vary File item been found ?
No, back to continue search
Save Exit call word
Is there an implied resume?

Yes, so set ① to ②

Save Exit call word in CT_v
0 → Q₃₅
Sentence CW → Ace
Is 26000 > Sentence CW? If yes, then 22—
Exit CW → A_u

Exit CW → A_u
1 → Q₃₅ indicates within Pseudo-Op.
Illegal sentence number check
Fill v of Exit line

③ VE 0 RJ LW LW1
 1 TU A CT
 2 TV Q CT
 3 TP BK10 Q
 4 QJ VE5 VA36
 5 TP CN15 Q
 6 QT CT CT2
 7 TJ CN54 VE12
 10 TJ CN60 VE32
 11 MJ 0 VE34
 12 SP VF 17
 13 SA CN24 0
 14 TU A VE26
 15 LT 6 A
 16 TP CN61 Q
 17 QT A CT3
 20 QT VF Q
 21 RS CT3 Q
 22 LT 43 CT3
 23 IJ CT3 VE25
 24 MJ 0 VE34
 25 TP CN15 Q
 26 QT 30000 A
 27 EJ CT2 VE32
 30 RA VE26 CN53
 31 MJ 0 VE23
 32 TV CN31 CE1
 33 MJ 0 VA36
 34 RJ WA WA2
 35 TP CN62 UP3
 36 RJ UP2 UP
 37 MJ 0 BQ6
 CA VE40

Obtain call word for Exit
 Save call word

Is stated Exit a Resume?

Yes, so Exit CW to CT2_v and Acc

26000 > CW?

No, so is 27000 > CW? If so OK

No, so to alarm

$m + VF2 \rightarrow A_u$
 Set beginning of search
 $j_n \rightarrow A_v$
 $n \rightarrow CT3$
 $m \rightarrow Q$
 $n - m \rightarrow CT3$
 $\div 2$
 Checked all items?
 Yes, and not found, so
 error
 Does exit CW agree
 with next Vary CW
 in Vary File?
 No, so advance to next item
 and back
 Set ③ to ② to avoid gen. 10--line

VF 0 jn m
 SN CW26
 SN CW26
 :
 SN CW26
 SN CW22
 SN CW22
 :
 SN CW22

Print alarm

Rewind tapes and stop

	CE	0	IA	CE	RS	Store Exit line
①		1	RJ	RS2	[CE2]	
②		2	MJ	0		10 0 1 → GL
③		3	TP	CN17	GL	Store
④		4	RJ	ST2	ST	Set index to # of WITH words
⑤		5	TP	BK4	CT	String-out address of 1st indicator → CT2
⑥		6	TP	CN23	CT2	Store next variable info in TE-TE4
⑦		7	RJ	SA3	SA	
		10	TP	TE2	Q	Is initial value absolute value?
		11	QJ	CE11	CE13	Yes, so TM - - → GL
		12	TP	CN27	GL	
		13	MJ	0	CE14	No, so TP - - → GL
⑧		14	TP	CN26	GL	TP CW _{init} - -
⑨		15	TU	TE2	GL	TP CW _{init} CW _{var}
		16	TV	TE1	GL	Store
		17	RJ	RS2	RS	Advance to next variable's indicator
		20	RA	CT2	CN5	Have we built all 'set initial value'
		21	IJ	CT	CE6	instructions?
		22	MJ	0	VB	Yes { VBO = CE21
			CA	CE22		
⑩	VB	0	IA	VB	CT3	Save address of JUMP TO RANGE
		1	TP	CT1	GL	MJ 0 - GL
		2	TP	CN34	GL	
		3	SP	BK5	0	To find CW of first sentence of range
		4	RJ	LW	LW1	Insert CW in MJ instruction
		5	TV	Q	GL	Store
		6	RJ	RS2	RS	10 0 1 → GL
		7	TP	CN17	GL	Store
		10	RJ	ST2	ST	Fill in RESUME entry line
⑪		11	TV	CT1	GC	Set index to # of WITH words
		12	TP	BK4	CT	Set address of first variable
		13	TP	CN23	CT2	Transfer variable to temporary area
⑫		14	RJ	SA3	SA	CW of variable → A.
		15	TP	CN15	Q	
		16	QT	TE1	A	
		17	TJ	CN12	VB21	64000 > CW?
		20	TJ	CN13	FX	No, 65000 > CW?
		21	MJ	0	FL	No, so floating variable
⑬		22	TP	TE1	Q	CW is 63..., so determine if fixed or
			QJ	FL	FX	floating
			CA	VB23		

		IA	FX	Fixed Variable (Build test for completion)
(12)	FX	0	TP CN27	GL TM — Q → GL
		1	TU TE3	GL TM CW _{inc} Q → GL
		2	RJ RS2	RS Store
		3	TP TE4	Q
		4	QJ FX22	FX5 } Is limit variable absolute value?
		5	TP VB2	GL No, so SP — 0 → GL
				SP 0
				TM CW _L A → GL
(14)		6	TU TE4	GL
		7	RJ RS2	RS Store
		10	TP BP4	GL SS — 0 → GL
		11	TU TE1	GL SS CW _{var} 0 → GL
		12	RJ RS2	RS Store
		13	TP CN30	GL TM A A → GL
		14	RJ RS2	RS Store
(15)		15	TP CN33	GL TJ Q 1002 → GL
(16)		16	RJ RS2	RS Store
		17	RA CT2	CN5 Advance to next indicator address
		20	IJ CT	VB13 Back if all tests for completion not generated
(13)		21	MJ 0	VC
		22	TP CN30	GL TM - A → GL
		23	MJ 0	FX6
			CA FX24	

		IA	FL	Floating Variable. (Build for completion)
(17)	FL	0	TP CN27	GL TM — — → GL
		1	TV RC1	GL TM — 70000
		2	TU TE3	GL TM CW 70000 (CW of increment.)
		3	RJ RS2	RS Store
		4	TP TE4	Q
		5	QJ FL6	FL11 } Is limit variable absolute value?
		6	TP CN30	GL Yes, so TM — A → GL
		7	TU TE4	GL TM CW A (CW of limit)
		10	RJ RS2	RS
(18)		11	TP CN32	GL FS A — → GL
		12	TV TE1	GL FS A CW (CW of variable)
		13	TP TE4	Q
		14	QJ FL16	FL15 } Is limit variable absolute value?
		15	TU TE4	GL No, so FS CW CW (CW _u of limit.)
(19)		16	RJ RS2	RS Store
		17	TP CN30	GL TM — A → GL
		20	TU CN27	GL TM Q A
		21	RJ RS2	RS Store
(20)		22	TP CN33	GL TJ — 01002 → GL
		23	TU RC1	GL TJ 70000 01002
		24	TU CN16	PL2 Set # working temps into 3rd word of Prelude
		25	TV CN1	PF1 Set word 2 of Op File 1 item = 1
		26	MJ 0	FX16 To store
			CA FL27	

		IA	VC	(Increment Variable Control)	
②1	VC	0	TP	BK4	CT Set index to # of WITH words
		1	TP	CN23	CT2 Set address of first variable indicator
②2		2	RJ	SA3	SA To store group of CW's in TE
		3	TP	CN15	Q } Send CW of variable to A
		4	QT	TE1	A }
		5	TJ	CN12	VC10 64000 > CW?
		6	TJ	CN13	FD 65000 > CW?
②3		7	MJ	O	FG No, so floating variable
		10	TP	TE1	Q } CW is 63---; determine if fixed or
		11	QJ	FG	FD } floating
			CA	VC12	

		IA	FG	(Build Floating increment variable instructions)	
②4	FG	0	TP	TE3	Q Increment CW → Q
		1	QJ	FG2	FG5 Absolute value?
		2	TP	CN27	GL Yes TM ——— Q → GL
		3	TU	TE3	GL TM CW _{inc} Q
		4	RJ	RS2	RS Store
②5		5	TP	CN31	GL FA Q ——— → GL
		6	TV	TE1	GL FA Q CW _{var.}
		7	TP	TE3	Q } Is increment absolute value?
		10	QJ	FG12	FG11 } No; so FA CW _{inc} CW _{var.} → GL
		11	TU	TE3	GL Store
②6		12	RJ	RS2	RS To store
		13	TP	CN26	GL TP Q ——— → GL
		14	TV	TE1	GL TP Q CW _{var.}
		15	MJ	O	FD14
			CA	FG16	

		IA	FD	(Build fixed increment variable instrs. and Jump Back)
②⑦	FD 0	TP	TE3	Q Increment to Q
	1	QJ	FD2	FD5 Absolute value?
	2	TP	CN27	GL Yes, so TM — Q → GL
	3	TU	TE3	GL TM CW _{inc} Q → GL
	4	RJ	RS2	RS Store
②⑧	5	TP	FD15	GL RA — — → GL
	6	TU	TE1	GL RA CW _{var} — → GL
	7	TP	TE3	Q } Is increment abs. value?
	10	QJ	FD11	FD13 } Yes, RA CW _{var} Q → GL
	11	TV	CN27	GL
	12	MJ	0	FD14
②⑨	13	TV	TE3	GL No, RA CW _{var} CW _{inc.} → GL
③⑦	14	RJ	RS2	RS Store
	15	RA	CT2	CN5 Advance to next indicator
	16	IJ	CT	VC2 Back if all increment instructions not generated
	17	TP	CN34	GL MJ 0 — → GL
	20	TV	CT3	GL MJ 0 01--- (Back to Jump to Range)
	21	RJ	RS2	RS Store
	22	MJ	0	VT { VTO = FD22 }
		CA	FD23	

	IA	VT		
(31)	VT 0	TP	BK4	CT
	1	TP	CN23	CT2
(32)	2	RJ	SA3	SA
	3	SP	TE	0
	4	ZJ	VT5	VS1
(B)	5	MJ	0	[VT6]
(B1)	6	TV	CN36	VT5
(33)	7	TP	CT1	CT3
	10	TP	CN26	GL
	11	TV	RC1	GL
	12	SP	CN3	0
	13	RJ	CW	CW1
	14	TU	A	GL
	15	RJ	RS2	RS
	16	TP	CN26	GL
(34)	17	TU	CN35	GL
	20	TV	CN25	GL
	21	RJ	RS2	RS
	22	TP	CN20	GL
	23	RJ	ST2	ST
	24	TP	CN25	GL
	25	RJ	RS2	RS
(35)	26	TP	CN21	GL
	27	RJ	ST2	ST
	30	TP	CN36	GL
	31	TV	CN25	GL
	32	RJ	RS2	RS
	33	TP	CN22	GL
	34	RJ	ST2	ST
	35	TP	CN7	GL
	36	TV	CT1	GL
(36)	37	RJ	RS2	RS
	40	TV	CT1	GC1
(B2)	41	LQ	TE	Q20
	42	QJ	VT45	VT43
	43	LQ	Q	1
	44	QJ	DJ	DJ2
	45	QJ	DL	VT46
(37)	46	LQ	Q	1
	47	QJ	DJ4	VT50
(43)	50	QJ	DK	DJ16
		CA	VT51	

Set index to # of WITH words
Set address of 1st indicator
Transfer 5 words to temporary

Is indicator zero?

Set (B) to (B2)
(CT1) → CT3. Save alarm print address
TP — — — → GL
TP — 70000

Find CW for 79₁₀
TP 67... 70000 → GL
Store
TP — — — → GL
TP 10000 —
TP 10000 50002
Store
10 0 3 — → GL
Store
TP 10001 50002 → GL
Store
10 0 4 — → GL
Store
RJ 50002 — — → GL
RJ 50002 50002 → GL
Store
10 2 0 — → GL
Store
MS 0 — — → GL
MS 0 01000+...

Store
Change entry to jump into first test

DJ, indicator = 14 DJ2, indicator = 10
DL, indicator = 30

DJ4, indicator = 22
DK12, indicator = 21. DJ16, indicator = 20

		IA	DJ	Indicators: 10, 14, 20(part 1)
③⑧	Indic.= DJ 0	RJ	EM22	EM Build and store test instr. for case 2
	14 1	MJ	0	VS
③⑨	Indic.= 10 2	RJ	EN22	EN Build and store test instr. for case 1
	3	MJ	0	VS
④⑩	Indic.= 22 4	TP	CN26	GL TP — A → GL
	5	TU	TE3	GL TP CW _{inc} A → GL
	6	RJ	RS2	RS Store
	7	TP	VA6	GL ZJ — — → GL
	10	SP	CT1	0
	11	SA	CN1	17
	12	TU	A	GL ZJ NI — → GL
	13	TV	CT3	GL ZJ NI alarm entry → GL
	14	RJ	RS2	RS Store
	15	MJ	0	VS
④⑪	Indic.= 20 16	TP	TE3	Q
	17	QJ	DJ4	DJ20 } Is increment absolute value?
	20	TP	VA5	GL SP — 0 → GL
	21	SP	CN	0
	22	RJ	CW	CW1 } Find CW for zero
	23	TU	A	GL SP CW _{zero} 0 → GL
	24	RJ	RS2	RS Store
	25	TP	CN33	GL TJ — — → GL
	26	TU	TE3	GL TJ CW _{inc} — → GL
④⑫	27	SP	CN2	0
	30	SA	CT1	0
	31	TV	A	GL TJ CW _{inc} NI + 1 → GL
	32	RJ	RS2	RS Store
	33	TP	CN34	GL MJ 0 — → GL
	34	TV	CT3	GL MJ 0 alarm entry → GL
	35	RJ	RS2	RS Store
	36	MJ	0	VS
		CA	DJ37	
		IA	DK	
④⑬	Indic.= DK 0	TP	CN26	GL TP — A → GL
	21 1	TU	TE3	GL TP CW _{inc} A → GL
	2	RJ	RS2	RS Store
	3	TP	CN35	GL SJ — — → GL
	4	SP	CT1	0
	5	SA	CN1	17
	6	TU	A	GL SJ NI — → GL
	7	TV	CT3	GL SJ NI alarm entry
	10	RJ	RS2	RS Store
	11	MJ	0	VS
		CA	DK12	

(44) Indic.= 30DL 0 IA DL (b variable, a and/or c variable Indicator = 30)

1	QJ	DL4	DL2	} Is increment absolute value?	
2	TP	CN26	GL		
3	MJ	0	DL5	No, TP — A —> GL	
(45)	4	TP	CN30	GL	Yes, TM — A —> GL
(46)	5	TU	TE3	GL	T(P) _M CW _{inc} A
	6	RJ	RS2	RS	Store
	7	TP	TE3	Q	} Is increment absolute value?
	10	QJ	DL24	DL11	
(47)	11	TP	CN35	GL	SJ — — —> GL
	12	SP	CT1	0	
	13	SA	CN1	0	
	14	TV	A	GL	SJ — NI —> GL
	15	TP	TE2	Q	} Is initial value absolute value?
	16	QJ	DL21	DL17	
	17	SA	CN4	17	No, so add 4 to (A)
	20	MJ	0	DL22	
(48)	21	SA	CN5	17	Yes, so add 5 to (A)
(49)	22	TU	A	GL	SJ b < 0 NI —> GL
	23	RJ	RS2	RS	Store
(50)	24	TP	VA6	GL	ZJ — — —> GL
	25	SP	CT1	0	
	26	SA	CN1	17	
	27	TU	A	GL	ZJ NI —> GL
	30	TV	CT3	GL	ZJ NI alarm GL
	31	RJ	RS2	RS	Store entry
	32	RJ	EN22	EN	
(51)	33	TP	TE3	Q	} Is increment absolute value?
	34	QJ	VS	DM	
		CA	DL35		

	IA	DM		
DM 0	TP	CN34	GL	MJ 0 —> GL
1	SP	CT1	0	
2	TP	TE4	Q	} Is limit absolute value?
3	QJ	DM6	DM4	
4	SA	CN52	0	No, so add 3 to current address in GC region
5	MJ	0	DM7	
6	SA	CN4	0	Yes, so add 4 to current address in GC region
7	TV	A	GL	MJ 0 NI + {2}{3}
10	RJ	RS2	RS	Store
11	RJ	EM22	EM	Now build and store tests for b < 0 case
12	MJ	0	VS	
	CA	DM13		

		IA	EM		
	0	TP	TE2	Q	} Is initial value absolute value?
	1	QJ	EM4	EM2	
	2	TP	CN26	GL	TP — A —> GL
	3	MJ	0	EM5	
(59)	4	TP	CN30	GL	TM — A —> GL
(60)	5	TU	TE2	GL	TM CW _{init} A —> GL
	6	RJ	RS2	RS	Store
(61)	7	TP	TE4	Q	} Is limit absolute value?
	10	QJ	EM11	EM14	
	11	TP	CN27	GL	Yes, TM — Q —> GL
	12	TU	TE4	GL	TM CW ₁ Q —> GL
	13	RJ	RS2	RS	Store
(62)	14	TP	CN33	GL	TJ Q —> GL
	15	TV	CT3	GL	TJ Q alarm —> GL entry
	16	TP	TE4	Q	} Is limit value absolute value?
	17	QJ	EM21	EM20	
(63)	20	TU	TE4	GL	No, so change u of TJ to CW of limit
(64)	21	RJ	RS2	RS	Store
	22	MJ	0	30000	Out
		CA	EM23		
		IA	EN		
EN	0	TP	TE4	Q	} Is limit absolute value?
	1	QJ	EN4	EN2	
	2	TP	CN26	GL	TP — A —> GL
	3	MJ	0	EN5	
(65)	4	TP	CN30	GL	TM — A —> GL
(66)	5	TU	TE4	GL	TM CW ₁ A —> GL
	6	RJ	RS2	RS	
	7	TP	TE2	Q	} Is initial value absolute value?
	10	QJ	EN11	EN14	
(67)	11	TP	CN27	GL	TM — Q —> GL
	12	TU	TE2	GL	TM CW _{init} Q —> GL
	13	RJ	RS2	RS	Store
(68)	14	TP	CN33	GL	TJ Q —> GL
	15	TV	CT3	GL	TJ Q alarm —> GL entry
	16	TP	TE2	Q	} Is initial value absolute value?
	17	QJ	EN21	EN20	
	20	TU	TE2	GL	No, so change u of TJ to initial value CW
(69)	21	RJ	RS2	RS	
	22	MJ	0	30000	
		CA	EN23		

		IA	VS		
(54)	VS 0	TP	CN34	CT4	Set indicator when tests are generated
(55)	1	RA	CT2	CN5	Advance to next indicator's address
	2	IJ	CT	VT2	Have all variables been checked?
	3	TP	CT4	Q	} Yes, so have we put in tests?
	4	QJ	VS5	BP	
	5	TP	CN34	GL	Yes, so MJ 0 1003 → GL
	6	RJ	RS2	RS	Store
(56)	7	RJ	CV3	CV	Convert sentence # to Flex code
	10	TV	ST	VS12	
	11	RP	30013	VS13	} Put in relative constants and Flex codes
	12	TP	RC	30000	
	13	TP	CN1	PF	Set word 1 of Op File I = 1
	14	TV	CN7	PF1	Set word 2 of Op File I = 14 ₈
	15	TU	CN36	PF4	Insert 50002 CW into 5th word of Op. File I
	16	TV	RC	PL	Set V of 1st word of Prelude = 11 ₈
	17	TP	CN2	PL1	Set 2nd word of Prelude = 2
	20	TP	CN16	PL2	Build 3rd word of Prelude
	21	TP	CN55	Q	} 50002 call word → A _u
	22	QT	CN36	A	
	23	RJ	LS	LS1	Insert call word in List I
	24	MJ	0	BP	To build Prelude and Op File I {BPO = VS24}
		CA	VS25		

		IA	BP		
(57)	BP 0	LQ	BK3	Q17	
	1	TU	Q	PL	Sentence CW to Prelude to Op File I
	2	TU	Q	PF	
	3	SP	ST	0	} # lines subject to address modification 1 to word 2 of Prelude
	4	SS	CN24	0	
	5	SA	PL1	0	
	6	TV	A	PL1	} # lines in Prelude + Rtne to 1st word of Prelude
	7	SA	PL	0	
	10	SA	CN6	0	
	11	TV	A	PL	
	12	TP	BK1	PL5	Store sentence number in word 6 of Prelude
	13	SP	CT1	0	} # words in running rtne. to 2nd word of Op File I item
	14	SS	CN10	0	
	15	AT	PF1	PF1	
(58)	16	SP	BK6	0	} CW of last sentence to word 3 of Op File I
	17	RJ	LW	LW1	
	20	TU	A	PF2	} CW of Exit sentence to word 4 of Op File I
	21	SP	GC2	17	
	22	TU	A	PF3	
	23	RA	PF	CN4	Add 4 to # lines in Op File I item
	24	TP	CN57	OP1	} To write Op File I and generated routines on tape
	25	RJ	OP	OP2	
Exit	26	MJ	0	VY	To Exit
		CA	BP27		

(70)

		IA	CV		
CV	0	TP	CN6	CT5	Set index
	1	TP	BK1	Q	Sentence # → Q
	2	IJ	CT5	CV4	Finished?
	3	MJ	0	30000	Yes, so out
	4	LA	RC5	6	
	5	LQ	Q	6	
	6	QT	CN51	A	Next digit → A
	7	TJ	CN52	CV15	3 > digit?
	10	SS	CN52	0	No, so subtract 3
	11	SA	CN32	0	Add Base Flex code address
	12	TV	A	CV13	Set address of Flex code
	13	RA	RC5	30000	Add into Flex word
	14	MJ	0	CV2	
	15	SA	CN52	0	Add 3
	16	AT	RC5	RC5	Add into Flex word
	17	MJ	0	CV2	
		CA	CV20		
		IA	RS		
RS	0	RJ	ST2	ST	
	1	RA	CT1	CN1	
	2	MJ	0	[30000]	
		CA	RS3		
		IA	ST		
ST	0	TP	GL	[GC]	Store generated coding
	1	RA	ST	CN1	
	2	MJ	0	[30000]	
		CA	ST3		
		IA	SA		
SA	0	TU	CT2	SA2	
	1	RP	30005	SA3	
	2	TP	[30000]	TE	
	3	MJ	0	[30000]	
		CA	SA4		

	IA	CN	
0	0	0	0
1	0	0	1
2	0	0	2
3	0	0	117
4	0	0	4
5	0	5	5
6	0	0	6
7	MS	0	14
10	0	0	1000
11	0	0	63000
12	0	0	64000
13	0	0	65000
14	0	0	67000
15	0	0	77777
16	0	10	2011
17	10	0	1
20	10	0	3
21	10	0	4
22	10	2	0
23	0	BK11	BK11
24	0	VF2	GC
25	TP	10001	50002
26	TP	Q	A
27	TM	Q	Q
30	TM	A	A
31	FA	Q	CE4
32	FS	A	CN37
33	TJ	Q	1002
34	MJ	0	1003
35	SJ	10000	0
36	RJ	50002	VT41
37	0	0	37
40	0	0	52
41	0	0	74
42	0	0	70
43	0	0	64
44	0	0	62
45	0	0	66
46	0	0	72
47	0	0	60
50	0	0	33
51	0	0	77
52	0	0	3
53	0	2	0
54	0	0	26000
55	0	77777	0
56	0	0	42
57	0	PL	PF
60	0	0	27000
61	0	0	07777
62	40	CN63	10

63	54	30656	74730
64	01	30723	46601
65	47	67656	60154
66	30	31305	40166
67	51	01240	17024
70	54	73016	53050
71	66	30502	63001
72	22	77777	77777
	CA	CN73	

R	E	S	U	M	E
△	E	X	I	T	△
M	U	S	T	△	R
E	F	E	R	△	T
O	△	A	△	V	A
R	Y	△	S	E	N
T	E	N	C	E	△

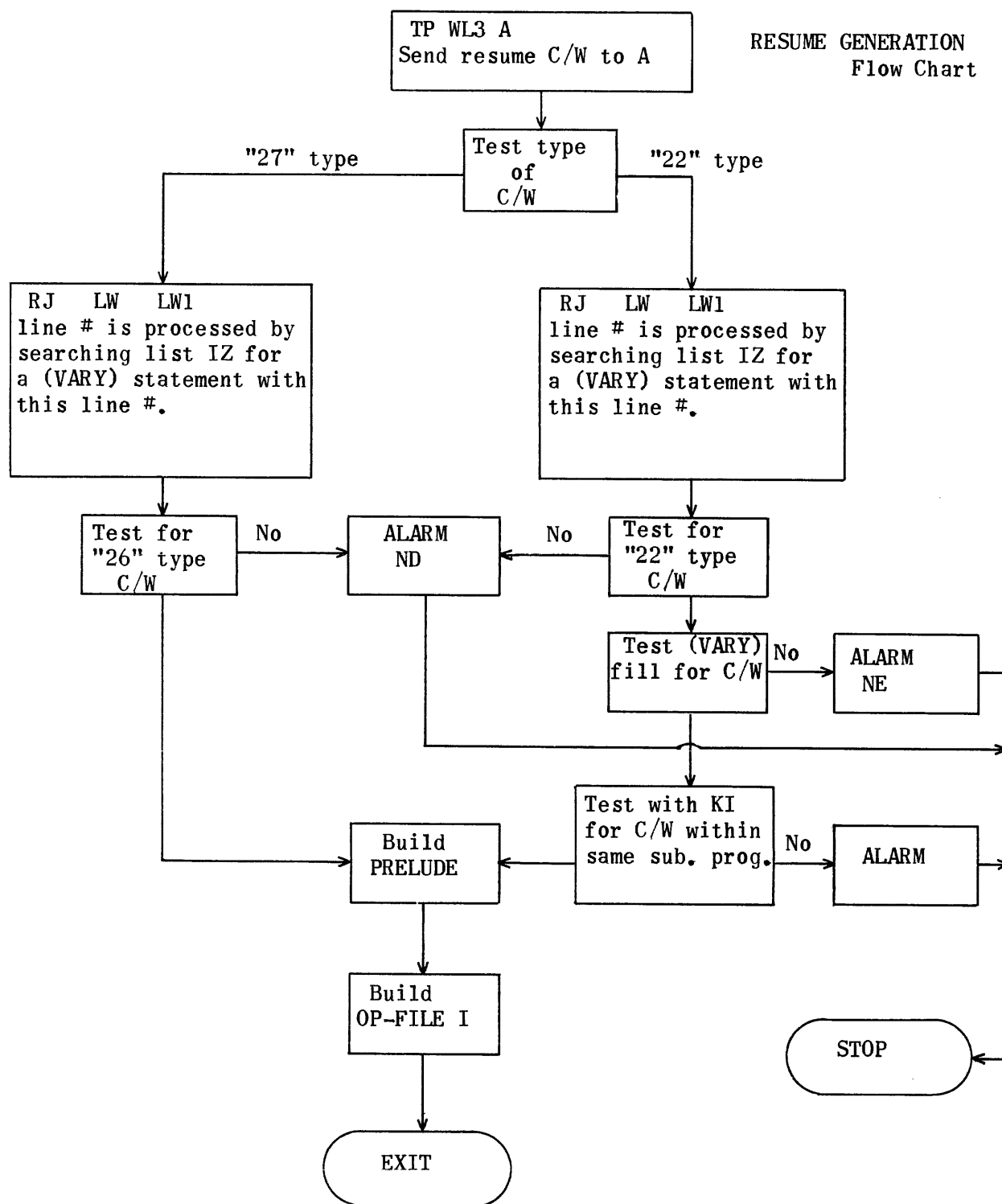
	IA	RC	
RC 0	0	20000	11
1	0	70000	70000
2	45	47173	01225
3	04	24200	60120
4	06	16200	45700
5	0	0	0
6	47	04053	02404
7	30	06041	40622
10	20	26140	61401
11	20	04110	30315
12	57	42454	50000
	CA	RC13	

CR	↑	V	A	R	Y
△	S	E	N	T	E
N	C	E	△	↓	
↑	△	H	A	S	△
A	N	△	I	N	D
E	F	I	N	I	T
E	△	L	O	O	P
↓	.	CR	CR		

	IA	CT	
CT 0	0	0	0
1	0	0	01000
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
	CA	CT6	

Holds CW for Exit line; then Index for
WITH words
lines running routine + 1000 (exclude
10 lines)
Holds string-out address of next test
indicator
Address of instruction to jump to Range;
then, address of Alarm Print instruction
Indicator that test coding already
appears
Index for Sentence number conversion (not
stored on MD)

RESUME GENERATION
Flow Chart



RESUME GENERATION

REGIONS

RE NC2512
RE ND2651
RE NE2667
RE WL2242
RE LW1250
RE VF47101
RE GL5360
RE TF2706
RE OP1047
RE KI1336
RE UP421
RE BQ632
RE WA653

RESUME GENERATOR

		IA	NC		
	0	MJ	0	30000	
	1	TP	WL3	Q	Send CW to Q from Resume String-Out
	2	QT	NC61	A	Mask i_{14-9}
	3	SS	NC62	0	Test for 27____ (subtract 27000 _v)
	4	ZJ	NC13	NC5	
27____ Branch	5	TP	WL4	A	Send line no. k (referring to Vary sent.)
	6	RJ	LW	LW1	Routine D finds CW for vary sent, in IZ
	7	TP	Q	NC66	Save CW in NC66
	10	QT	NC61	A	Mask i_{14-9} (xx000) _v
	11	SS	NC63	0	Test for 26____ (subtract 26000 _v)
	12	ZJ	ND1	NC41	To prelude generation or error
Test for 22____	13	TP	WL4	A	Send line no. k (referring to Vary sent.)
	14	RJ	LW	LW1	Routine D finds CW for Vary sent, in IZ
	15	TP	Q	NC66	Save CW in NC66
	16	QT	NC61	A	Mask i_{14-9} (xx000) _v
	17	SS	NC64	0	Test for 22____ (subtract 22000) _v
	20	ZJ	ND1	NC21	To test Vary file or error
22____ Branch	21	TP	VF	Q	Code word of Vary file (no. of words)
	22	QT	NC57	NC56	Mask "n" & store in Index = NC56
	23	QT	NC60	NC107	Mask "m" & save in NC107
	24	MJ	0	NC120	Jump to correction (1)
	25	0	0	0	
Test Vary File for CW	26	RS	NC56	NC110	Subtract 1 from Index
	27	RA	NC107	NC111	Add 2 to m
	30	MJ	0	NC130	Jump to correction (2)
	31	0	0	0	
	32	TP	VF	Q	Changed to VF _m + 2
	33	QT	NC60	A	Mask CW

34	EJ	NC66	NC37	Test with CW in NC66 for equality
35	RA	NC32	NC106	Increment $VF_m + 2$ by 2
36	MJ	0	NC112	Jump to correction (3)
37	TP	NC66	A	Vary CW in A
40	MJ	0	NC103	Jump to correction (4)
41	RP	30010	NC43	} Generate Prelude Format
42	TP	NC70	GL	
43	LQ	WL3	17	Move CW of Resume to "u" portion
44	TU	Q	GL	$(0, CW, 0)_Q \rightarrow (0, 0, 0)_{gl}$
45	TP	WL1	GL5	Line no.
46	TV	NC66	GL7	CW
47	RP	30003	NC51	} Generate Op File I format
50	TP	NC100	TF	
51	TU	GL	TF	CW Resume
52	MJ	0	NC125	Jump to correction (5)
53	TP	NC114	OP1	(0 GL TF)
54	RJ	OP	OP2	
55	MJ	0	NC	
56	0	0	0	Index
57	0	7777	0	
60	0	0	77777	
61	0	0	77000	
62	0	0	27000	
63	0	0	26000	
64	0	0	22000	
65	0	1	0	
66	0	0	0	L(Vary CW)
67	40	0	0	
70	0	0	10	} CW Resume in u
71	0	0	2	
72	0	0	0	
73	0	0	0	} Prelude format
74	0	0	0	
75	0	0	0	Line number
76	MJ	0	30000	
77	MJ	0	0	CW Vary in v
100	0	0	3	CW Resume in u
101	0	0	2	Op File I format
102	0	0	0	CW Vary in u
103	TP	NC67	Q	} Send parameter to Q
104	RJ	KI	KI1	
105	MJ	0	NC41	(4)
106	0	2	0	
107	0	0	0	
110	0	0	1	
111	0	0	2	
112	IJ	NC56	NC32	} (3)
113	MJ	0	NE1	
114	0	GL	TF	Parameter for Op routine
115	0	0	0	
116	0	0	0	
117	0	0	0	

120	TP	NC56	A	}	$n \rightarrow A$
121	LT	25	NC56		Shift n to A_0
122	RS	NC56	NC107		(1) $n-m \rightarrow NC56 = \text{Index}$
123	LT	43	NC56		Index shifted right 1 or $NC56/2$
124	MJ	0	NC26	}	
125	TP	NC66	A		Move CW of Vary sent. to u address,
126	LT	10017	TF2	}	jump to correction (6)
127	MJ	0	NC134		(5)
130	LT	63	NC133	}	Shift to u position
131	RA	NC32	NC133		(2) Add to (VF), i.e., $VF_m + 2$
132	MJ	0	NC32	}	
133	O	0	0		
134	TP	NC136	NC32	}	Preset NC32
135	MJ	0	NC53		(6)
136	TP	VF	Q	}	Presetter for NC32
	CA	NC137			

ERROR ROUTINE

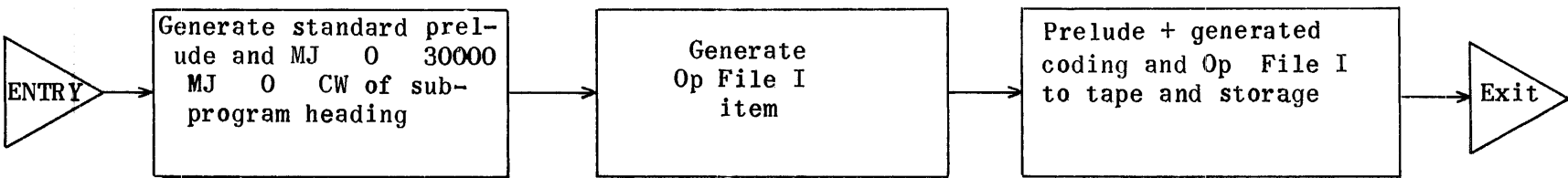
	IA	ND		
0	MJ	0	BQ6	
1	RJ	WA	WA1	
2	TP	ND15	UP3	
3	RJ	UP2	UP	
4	MJ	0	ND	
5	66	33305	43001	THERE Δ
6	34	65015	05101	IS Δ NO Δ
7	54	30313	05430	REFERE
10	50	26300	16651	NCE Δ TO
11	01	24011	77024	Δ A Δ (VA
12	54	73430	16566	RY) Δ ST
13	24	66304	73050	ATEMEN
14	66	22777	77777	T.
15	40	ND5	10	
	CA	ND16		

ERROR ROUTINE

	IA	NE		
0	MJ	0	BQ6	
1	RJ	WA	WA1	
2	TP	NE16	UP3	
3	RJ	UP2	UP	
4	MJ	0	NE	
5	65	66246	63047	STATEM
6	30	50660	15430	ENT Δ RE
7	31	30543	05026	FERENC
10	30	65012	40150	ES Δ A Δ N
11	51	50023	07234	ON-EXI
12	65	66305	06601	STENT Δ

13	17	70245	47343	(VARY)
14	01	65305	06630	ΔSENTE
15	50	26302	27777	NCE.
16	40	NE5	11	
	CA	NE17		

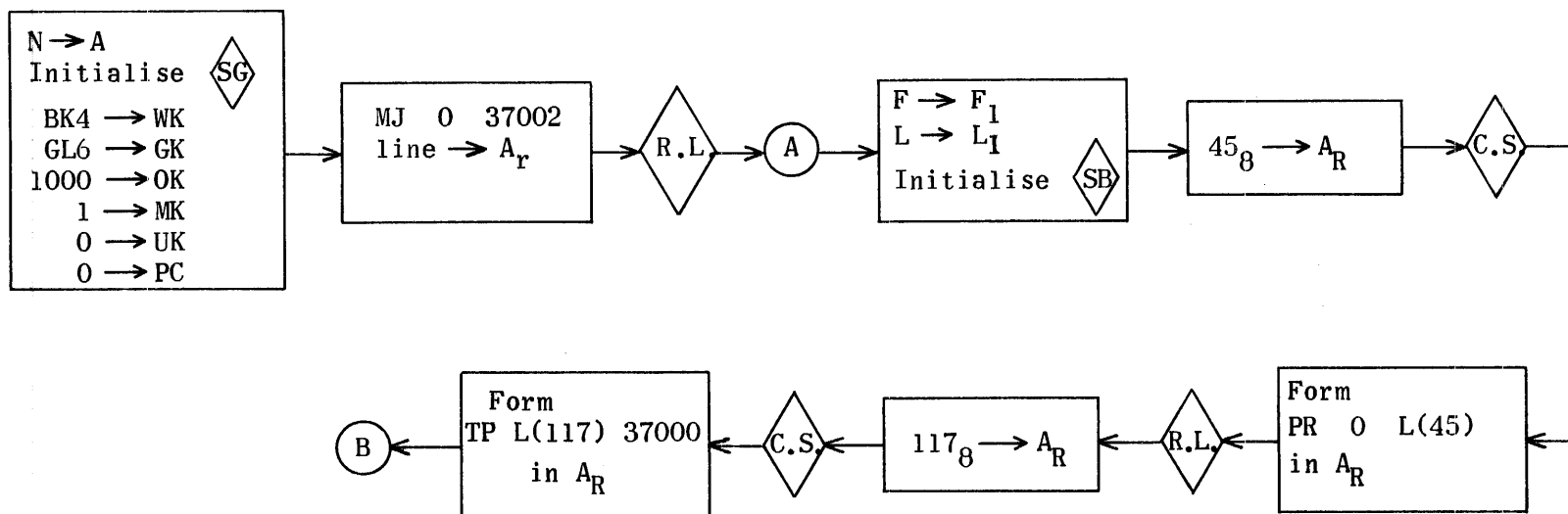
Flow Chart for Exit Generator

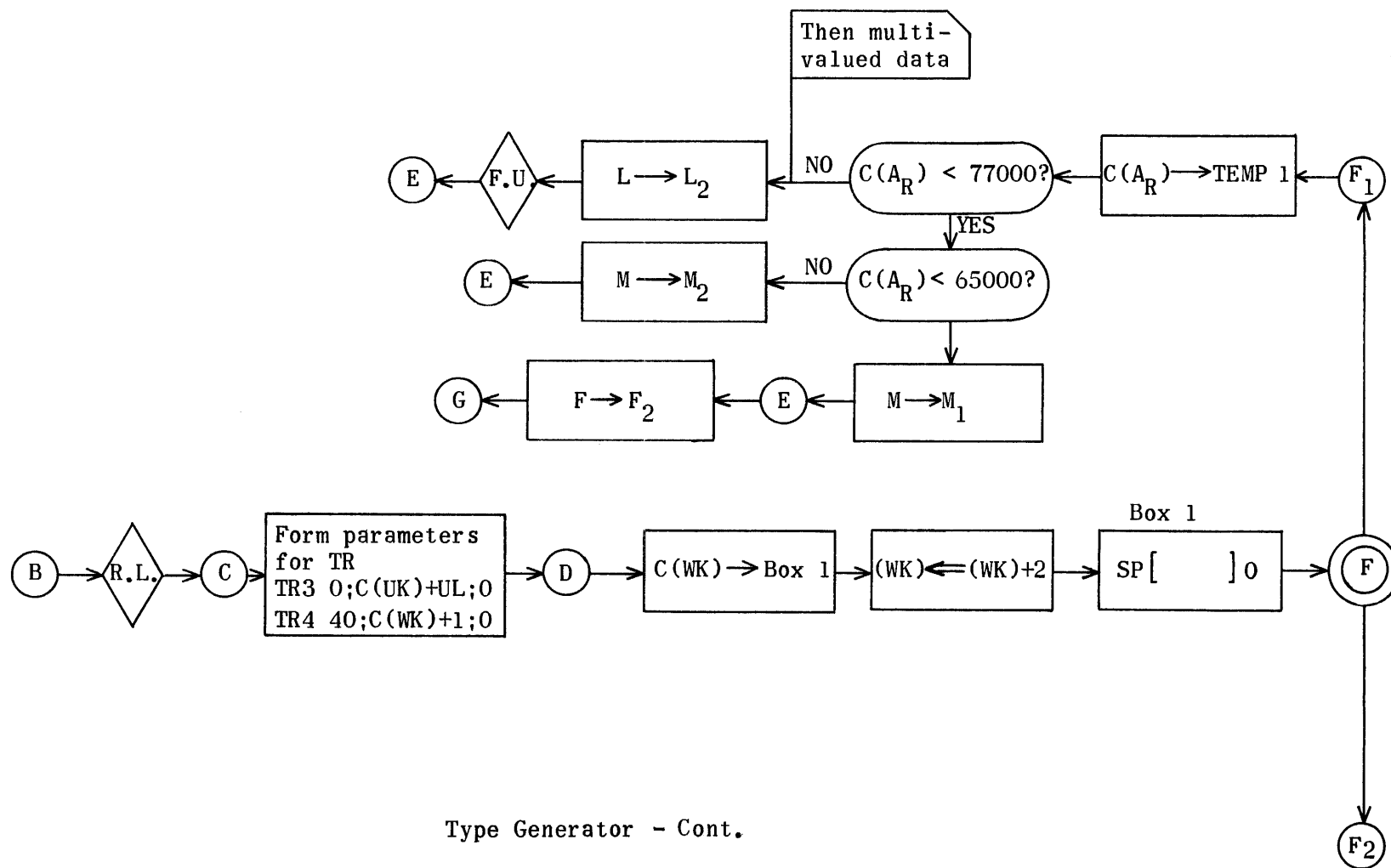


EXIT GENERATOR

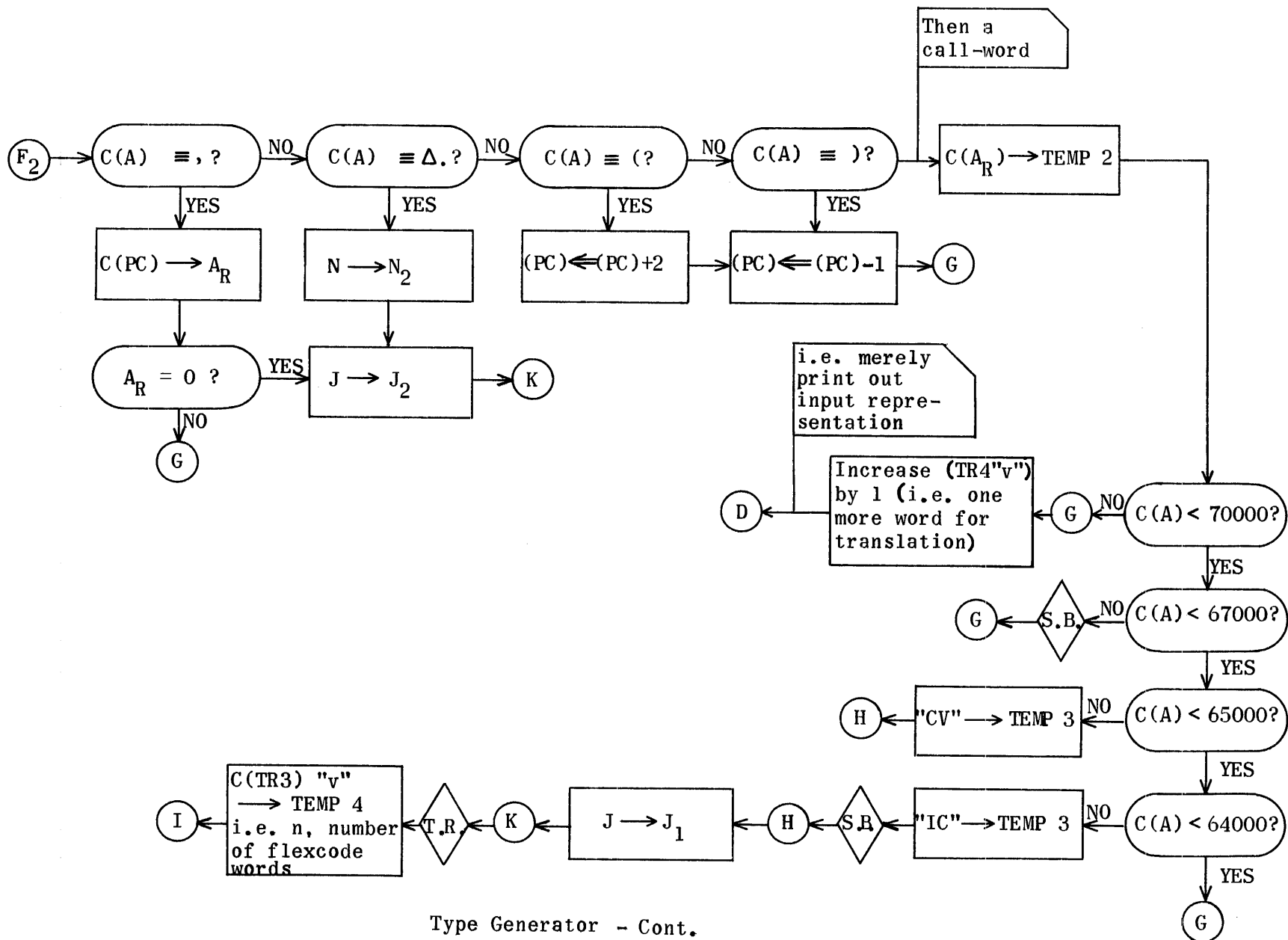
RE	PX2512			
RE	PL5360			
RE	FL2546			
RE	OP1047			
RE	WL2242			
IA	PX			
0 MJ	0	30000		
1 RP	30010	PX3	}	Generate Prelude Format
2 TP	PX20	PL		
3 SP	WL3	17	}	Sentence call word to prelude
4 TU	A	PL		
5 TP	WL1	PL5		Standard line no.
6 TV	WL4	PL7		CW Sub-routine heading → 2nd line of coding for sentence
7 RP	30003	PX11	}	Generate Op File I Format
10 TP	PX30	FL		
11 SP	WL3	17	}	Sentence call word to Op. File I
12 TU	A	FL		
13 SF	WL4	17	}	CW subroutine heading to Op File I
14 TU	A	FL2		
15 TP	PX33	OP1	}	Prelude + Generated Coding and Op File I to tape and storage
16 RJ	OP	OP2		
17 MJ	0	PX		Exit
20 0	0	10	}	Prelude Format
21 0	0	2		
22 0	0	0		
23 0	0	0		
24 0	0	0		
25 0	0	0	}	Object Program Coding
26 MJ	0	30000		
27 MJ	0	30000		
30 0	0	3	}	Op File I Format
31 0	0	2		
32 0	[0]	0		CW of Subroutine heading
33 0	PL	FL		Parameters
CA	PX34			

Type Generator

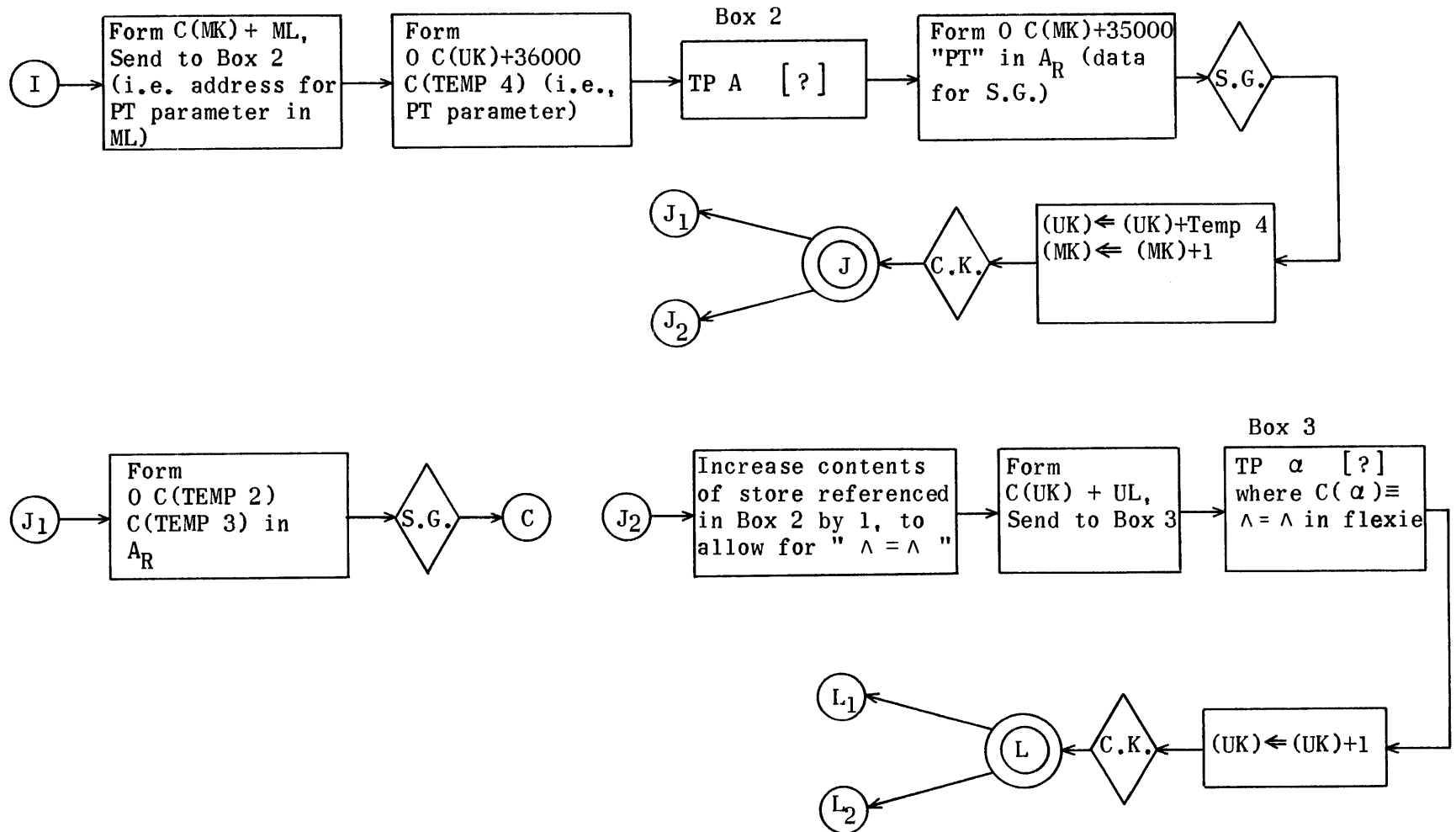




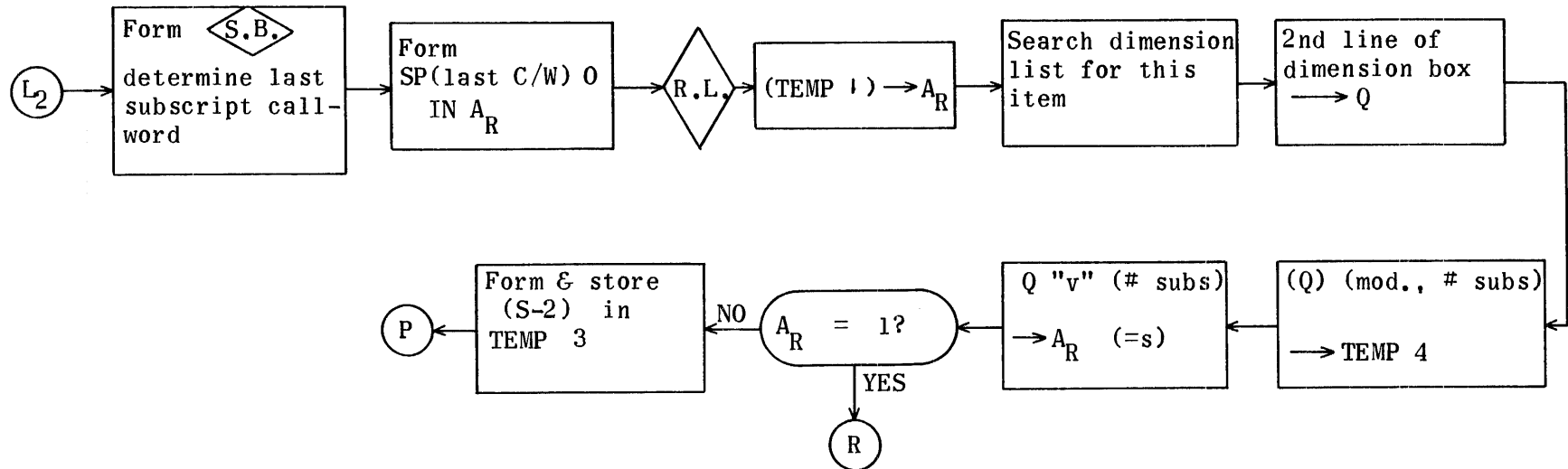
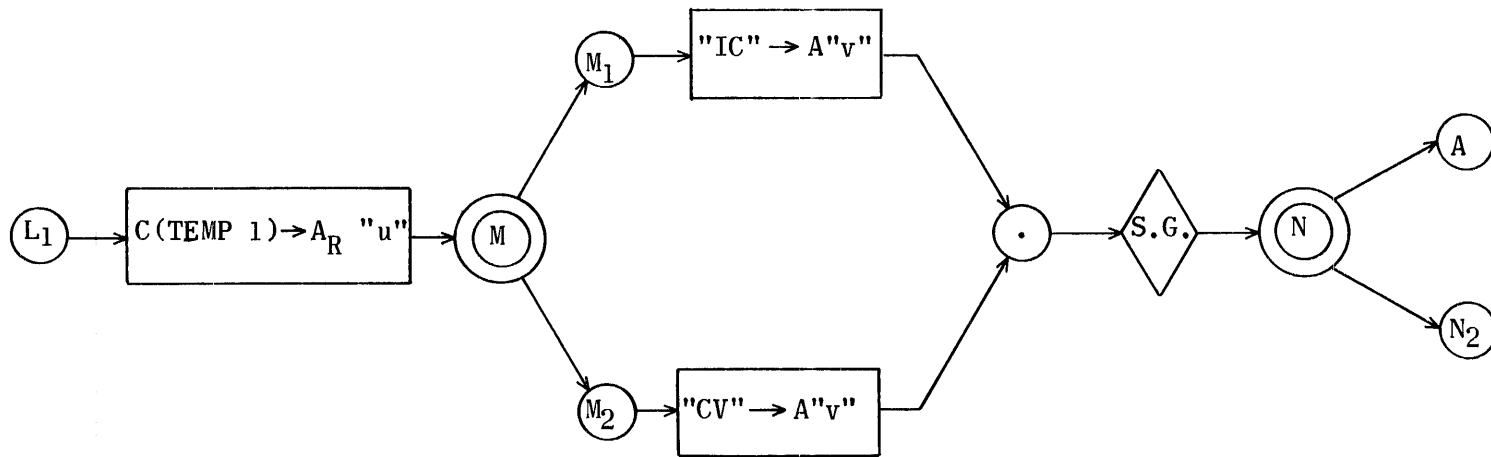
Type Generator - Cont.



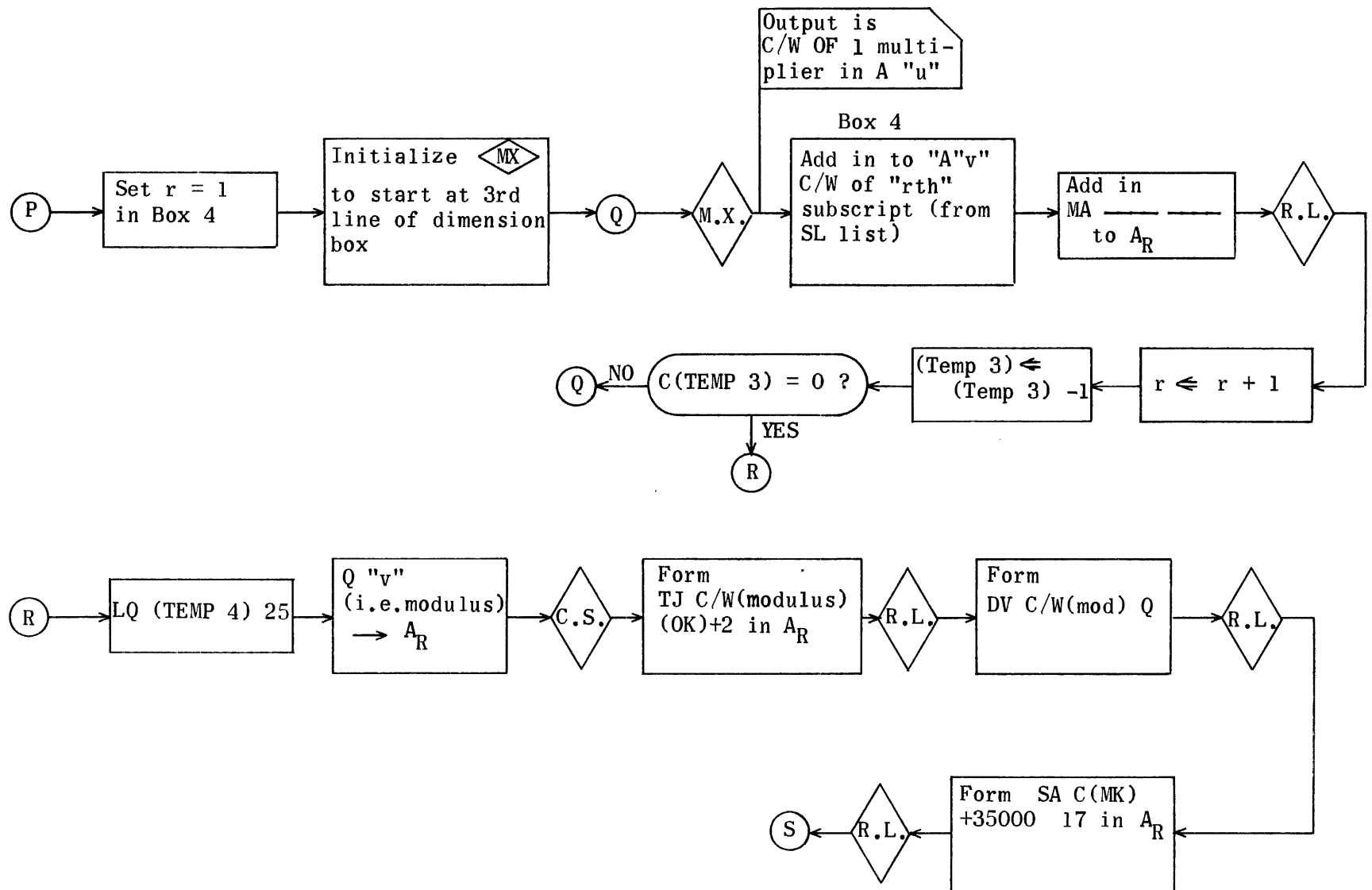
Type Generator - Cont.



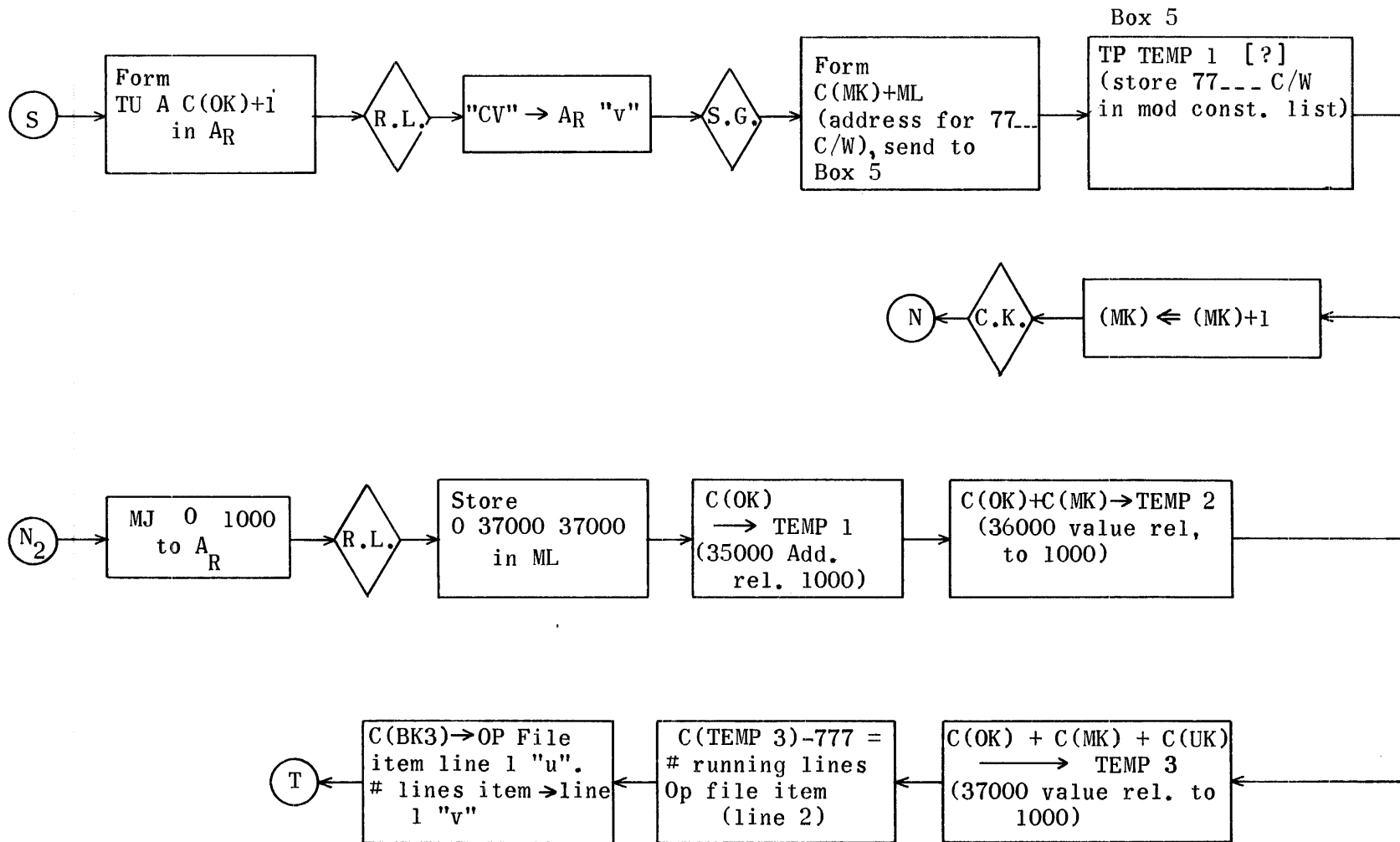
Type Generator - Cont.



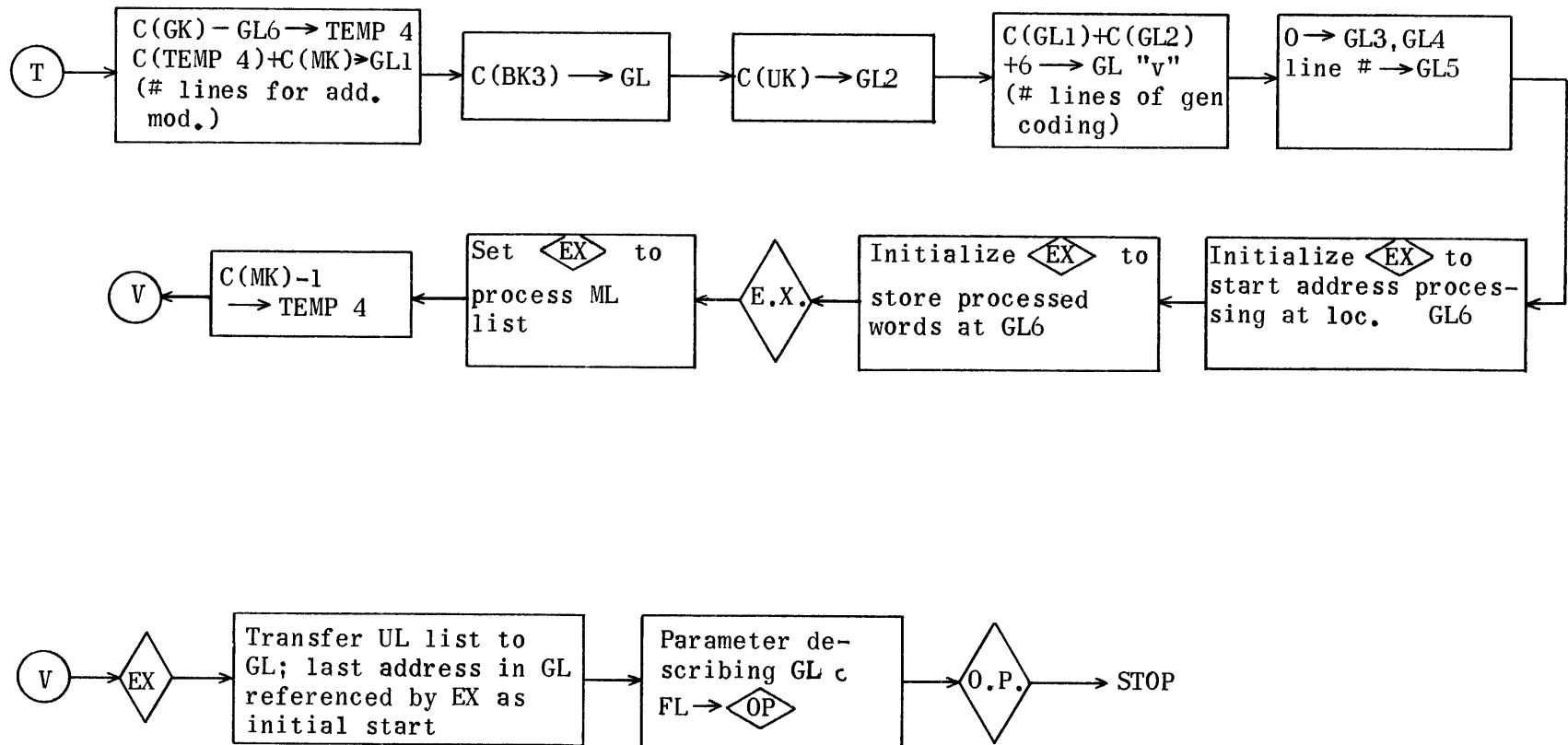
Type Generator - Cont.



Type Generator - Cont.



Type Generator - Cont.



Type Generator - Cont.

TYPE GENERATOR

REGIONS

Generation Subroutines	RE	LS1465	Library List Routine
	RE	CS1211	CS = CW Constant Call Word Routine
	RE	BR537	Machine Error Routine
	RE	UP421	Excess Three Print-Out Routine
	RE	OP1047	Loads Generators and Op Files on Tape
	RE	WA653	Routine to Print: Sentence — (Type)
	RE	BQ632	Rtne.to rewind all tapes & stop computer
	RE	TR1670	TR = VX Excess Three to Flex Code Routine
	RE	DL40102	Dimension List
	RE	BK2242	String-Out Buffer Input
	RE	GL5360	Buffer Output Region for Generated Rtne.
	RE	TY2512	Generator proper
	RE	IN2513	
	RE	TG2530	
	RE	MV2672	
	RE	PR2760	
	RE	MX3030	
	RE	EX3044	
	RE	SG3076	
	RE	RL3137	
	RE	ST3143	
	RE	CK3147	Working Space
	RE	FU3165	
	RE	SB3173	
	RE	CN3176	
	RE	PC3272	
	RE	OK3273	
	RE	GK3274	
	RE	UK3275	
	RE	MK3276	
	RE	WK3277	
	RE	TM3300	← Region used to build up Op File
	RE	SL3306	
	RE	VE3312	
	RE	FL3320	
	RE	ML3730	
	RE	UL4340	

Type Generator

IA	TY		
MJ	0	30000	Exit
CA	TY1		

Initialization

	IA	IN		
0	TV	TG141	TG131	(N) → (A)
1	RP	10003	IN3	} Initialize sub-generator
2	TV	SG36	SG10	
3	TV	CN73	FU5	Initialize Op File routine to FL2
4	TP	CN	PC	Clear PC
5	TP	CN	UK	UK → 0
6	TP	CN1	MK	MK → 1
7	TP	CN31	OK	OK → 1000
10	TP	CN70	GK	GK → GL6
11	TP	CN	WK	} WK → BK4
12	TU	CN73	WK	
13	SP	CN56	0	} MJ 0 37002 Generated
14	RJ	RL3	RL	
	CA	IN15		

Main Routine

	IA	TG		
0	TP	TG132	TG23	(F) → (F ₁)
1	TP	TG133	TG126	(L) → (L ₁)
2	TV	CN67	SB	Initialize subscript storing routine
3	SP	CN4	0	458 → A _R
4	RJ	CS	CS1	Place in constant pool
5	SP	CN57	0	} PR 0 L (carriage return flex) formed and put in output
6	SA	Q	0	
7	RJ	RL3	RL	
10	SP	CN5	0	79 → A
11	RJ	CS	CS1	Find CW
12	SA	CN47	0	} TP L(79) 37000 formed and put in output
13	RJ	RL3	RL	
14	SP	CN41	0	40 1 0 → A
15	AT	WK	TR4	Start to form entry line for TR
16	SP	UK	17	} and inform TR of location for output
17	AT	CN63	TR3	
20	TU	WK	TG22	r → BOX 1
21	RA	WK	CN45	r + 2 → r
22	SP	[30000]	0	Examine rth word
23	[0	30000	30000]	F ₁ ≡ TP A TM; F ₂ ≡ MJ 0 TG36

24	TJ	CN23	TG30
25	TP	TG134	TG126
26	RJ	FU4	FU
27	MJ	0	TG34

< 77000?
 No - i.e., 77--- type variable. (L) → (L₂)
 Insert in Op File item
 Jump to (E)

30	TJ	CN17	TG33
31	TU	CN71	TG127
32	MJ	0	TG34
33	TU	CN72	TG127
34	TP	TG135	TG23
35	MJ	0	TG55
36	RP	30004	TG53
37	EJ	CN25	TG40
	CA	TG40	

< 65000?
 No - i.e., 65,66 -- so "CV" wanted (M) → (M₂)
 Jump to (E)
 Must be 64---i.e., "IC" wanted (M) → (M₁)
 (F) → (F₂)
 Jump to (G)
 Test for, Δ () or none

	IA	TG40	
40	MJ	0	TG47
41	MJ	0	TG45
42	RA	PC	CN2
43	RS	PC	CN1
44	MJ	0	TG55
45	TV	TG136	TG131
46	MJ	0	TG51
47	SP	PC	0

Δ Jump to test PC
 Δ () Jump to set (N)
 (PC + 2 → PC
) PC - 1 → PC
 Jump to (G)
 (N) → (N₂)
 Jump to set (J)

50	ZJ	TG55	TG51
51	TP	TG137	TG111
52	MJ	0	TG71
53	TP	A	TM1
54	TJ	CN21	TG57
55	RA	TR4	CN1
56	MJ	0	TG20
57	TJ	CN20	TG62

If zero, continue; non-zero, jump to (G)
 (J) → (J₂)
 Jump to (K)
 Not any of above, CW. Save in TEMP 2
 C(A) < 70000?
 No, a dummy variable-count 1 more word
 for TR and jump to (C)
 C(A) < 67000?

60	RJ	SB2	SB
61	MJ	0	TG55
62	TJ	CN17	TG65
63	TP	CN14	TM2
64	MJ	0	TG70
65	TJ	CN16	TG55
66	TP	CN15	TM2
67	RJ	SB2	SB

No - a numerical subscript - so note
 and jump to (G)
 C(A) < 65000?
 No; a floating variable, so "CV" → TEMP 3
 and jump to (H)
 C(A) < 64000? If so DUMMY
 No - so subscript, so "IC" → TEMP 3
 Store in subscript list

70	TP	TG140	TG111
71	RJ	TR2	TR
72	TP	CN24	Q
73	QT	TR3	TM3
74	SP	MK	0
75	SA	CN65	0
76	TV	A	TG101
77	SP	UK	0
	CA	TG100	

(J) → (J₁)
 Go translate
 V mask → Q
 Save n (number flex words) in TEMP 4
 Current number modifiable constants
 Plus ML, is address at which to store
 Parameter about to be formed
 C(UK)

100	IA	TG100			
100	SA	CN11	17		+ 36000 = "u" of parameter
101	AT	TM3	[30000]		
102	SP	MK	0		C(MK)
103	SA	CN10	17		+ 35000 = CW for parameter location
104	SA	CN13	0		And PT is routine used
105	RJ	SG34	SG		Subgen to produce coding
106	RA	UK	TM3		UK + C (TEMP 4) → UK
107	RA	MK	CN1		MK + 1 → MK
110	RJ	CK3	CK		
111	[0	30000	30000]		$J_1 \equiv \text{SP TM1 17}; J_2 \equiv \text{MJ 0 TG115}$
112	SA	TM2	0		Form "CW required" "IC or CV"
113	RJ	SG34	SG		Subgenerator
114	MJ	0	TG14		Jump to (B)
115	SP	TG101	17		
116	TU	A	TG117		
117	RA	[30000]	CN1		Increase last parameter by 1 to allow for "↓ Λ = Λ"
120	SP	UK	0		
121	SA	CN64	0		
122	TV	A	TG123		Store "↓ Λ = Λ" in UL
123	TP	CN42	[30000]		
124	RA	UK	CN1		UK + 1 → UK
125	RJ	CK3	CK		
126	[0	30000	30000]		$L_1 \equiv \text{SP TM 17}; L_2 \equiv \text{MJ 0 MV}$
127	SA	[30000]	0		$M_1 \Rightarrow \text{add in "IC"}; M_2 \Rightarrow \text{add in "CV"}$
130	RJ	SG34	SG		Subgenerator
131	MJ	0	[30000]		Either back, or out to processor
132	TP	A	TM		F_1
133	SP	TM	17		L_1^1
134	MJ	0	MV		L_1^1
135	MJ	0	TG36		F_2^2
136	0	0	PR		N_2^2
137	MJ	0	TG115		J_2^2
140	SP	TM1	17		J_1
141	0	0	TG		N_1^1 (i.e., A)
	CA	TG142			
	IA	MV			
0	RS	SB	CN1		
1	LA	A	17		Obtain last stored subscript CW
2	TU	A	MV3		
3	SP	[30000]	17		
4	SA	CN52	0		Add in SP — —
5	RJ	RL3	RL		
6	SP	TM	0		Find appropriate dimension box
7	TU	6	MV10		
10	RP	20000	BR1		
11	EJ	DL	MV12		
12	SN	Q	17		- jn + r

13	SA	MV10	0	
14	SA	MV11	0	→ Address of 2nd line of box
15	TU	A	MV16	
16	TP	[30000]	Q	2nd line → Q
17	TP	Q	TM3	# subs, modulus → TEMP 4
20	QT	CN24	A	# subs → A
21	EJ	CN1	MV35	= 1?
22	ST	CN2	TM2	No, form & store S-2
23	TU	CN67	MV30	Prepare to extract CW in order
24	RA	MV16	CN33	Form address of 3rd line of dim. box
25	TU	A	MX	and initialize m-extractor
26	TV	MX12	MX1	
27	RJ	MX11	MX	Upon return, CW for 1 mult in A "u"
30	SA	[30000]	0	Add in CW of 1 subscript
31	SA	CN60	0	and MA ____
32	RJ	RL3	RL	Store
33	RA	MV30	CN33	
34	IJ	TM2	MV27	Prepare to pick up next subscript, then
35	LQ	TM3	25	back
36	QT	CN24	A	Modulus → Q "v"
37	RJ	CS	CS1	→ A
	CA	MV40		Place in constant pool
	IA	MV40		
40	SA	CN55	0	TJ L(Mod) 2
41	SA	OK	0	TJ L(Mod) C (OK) + 2
42	RJ	RL3	RL	Store
43	SP	Q	17	CW (Mod) → A "u"
44	SA	CN61	0	DV Mod Q
45	RJ	RL3	RL	Store
46	SP	MK	0	Form 35---type pseudo CW of base 77---
47	SA	CN10	17	in A "u"
50	SA	CN53	0	→ SA L(CW) 17
51	RJ	RL3	RL	Store
52	SP	CN51	0	TU A 1
53	SA	OK	0	TU A "N.I."
54	RJ	RL3	RL	Store
55	SP	CN14	0	And "CV" is routine used
56	RJ	SG34	SG	Subgenerator
57	SP	MK	0	C(MK) + ML is address in mod. list for
60	SA	CN65	0	77---CW
61	TV	A	MV62	
62	TP	TM	[30000]	Store base CW
63	RA	MK	CN1	MK + 1 → MK
64	RJ	CK3	CK	Check length
65	MJ	0	TG131	Jump back to (N) (Main Routine)
	CA	MV66		

Sub-Processor

0	IA	PR		
0	SP	CN6	0	MJ 0 1000
1	RJ	RL3	RL	Stored
2	TP	CN34	ML	Set up index address line
3	SP	OK	0	
4	TP	A	TM	35000 (i.e., ML) Val. rel. 1000
5	AT	MK	TM1	36000 (i.e., UL) val. rel. 1000
6	AT	UK	TM2	37000 (i.e., temp) val. rel. 1000
7	ST	CN7	FL1	Total # lines in running prog.(incl.temps)
10	SP	BK3	17	CW this routine
11	TP	A	GL	CW this routine to u of output line
12	SA	FU5	0	CW and number of lines complete Op File I
13	ST	CN66	FL	item
14	SP	GK	0	C(GK) - GL6 = # lines for modification by
15	ST	CN70	TM3	this routine
16	AT	MK	GL1	
17	TP	UK	GL2	
20	TP	CN	GL3	Complete prelude
21	TP	CN	GL4	
22	TP	BK1	GL5	
23	RS	TM3	CN1	Set up index for GL mod.
24	SP	CN70	17	
25	TU	A	EX	Extract words from GL6
26	TV	CN70	EX7	Store words at GL6
27	RJ	EX13	EX	Exchange 35, 36, 37's in GL
30	TU	CN65	EX	
31	SP	MK	0	
32	ST	CN1	TM3	Now process mod. consts.
33	RJ	EX13	EX	
34	TV	EX7	PR40	Set in address in GL for UL consts.
35	SP	UK	17	
36	AT	CN62	PR37	Form RP order
37	[0	30000	30000]	
40	TP	UL	[30000]	Repeated transfer from UL → GL
41	SP	GL1	0	
42	SA	GL2	0	Total # output lines
43	SA	CN3	0	
44	TV	A	GL	
45	TP	CN43	OP1	Parameter → OP
46	RJ	OP	OP2	Output
47	MJ	0	TY	EXIT
	CA	PR50		

Subroutine to Extract Multipliers and Obtain Appropriate CW's

	IA	MX		
0	TP	[30000]	Q	1 line → Q
1	MJ	0	[30000]	Switch (initially MX2)
2	TV	MX13	MX1	FLIP
3	MJ	0	MX7	
4	LQ	Q	25	Reverse "u" and "v"
5	RA	MX	CN33	Modify entry for next application
6	TV	MX12	MX1	FLOP
7	QT	CN24	A	Extract n
10	RJ	CS	CS1	Place in constant pool
11	MJ	0	[30000]	Exit
12	0	0	MX2	
13	0	0	MX4	
	CA	MX14		

Replacing 35---, 36---, 37--- Addresses with Values Relative 1000

	IA	EX		
0	TP	[30000]	Q	1 line to Q
1	QT	CN44	A	Inspect "Op. code"
2	EJ	CN35	EX7	If a 10 line, O.K.
3	RJ	EX31	EX14	Process "v"
4	LQ	Q	25	Shift "u" → "v"
5	RJ	EX31	EX14	Process "u"
6	LQ	Q	17	Restore Q
7	TP	Q	[30000]	And output it in GL
10	RA	EX	CN33	
11	RA	EX7	CN1	
12	IJ	TM3	EX	Back for next line
13	MJ	0	[30000]	EXIT
14	TP	Q	TM4	Save Q
15	QT	CN23	TM5	Mask off 1st two digits
16	RP	30003	EX31	
17	EJ	CN10	EX20	
20	MJ	0	EX26	35
21	MJ	0	EX24	36
22	RA	TM4	TM2	37 Add in val (37)
23	MJ	0	EX27	
24	RA	TM4	TM1	36 Add in val (36)
25	MJ	0	EX27	
26	RA	TM4	TM	35 Add in val. (35)
27	SS	TM5	0	Subtract original 1st two digits
30	TP	A	Q	Set result back in Q
31	MJ	0	[30000]	Exit back to main level
	CA	EX32		

Subgenerator, to Produce Coding Handling Parameters

	IA	SG		
0	TP	A	Q	"u" & "v" → Q for PT, IC, CV
1	SA	CN46	0	Add in TP ____
2	RJ	RL3	RL	Store
3	SP	CN36	0	10 0 3
4	RJ	ST3	ST	Store
5	QT	CN24	SG37	Name of routine ("V") to temp.
6	EJ	CN14	SG11	Is it CV?
7	EJ	CN15	SG12	Is it IC?
10	RJ	SG40	[30000]	PT
11	RJ	SG40	[30000]	CV
12	RJ	SG40	[30000]	IC
13	RJ	FU4	FU	} Used before? { Initialized to SG13
14	SP	SG37	17	
15	RJ	LS	LS1	No, insert in Op File 1, List 1, then
16	RS	SG40	CN1	ensure this path not taken again
17	TV	A	SG20	Find address to be changed
20	TV	SG35	[30000]	
21	SP	SG37	0	Name to A "v"
22	SA	CN50	0	Add in TP 35000 ____
23	RJ	RL3	RL	Store
24	SP	CN37	0	10 0 4
25	RJ	ST3	ST	Store
26	SP	SG37	17	
27	SA	SG37	0	
30	SA	CN54	0	Add in RJ ____
31	RJ	RL3	RL	Store
32	SP	CN40	0	10 2 0
33	RJ	ST3	ST	
34	MJ	0	[30000]	EXIT
35	0	0	SG26	
36	0	0	SG13	
37	0	30000	30000	} Erasable
40	0	30000	30000	
	CA	SG41		
	IA	RL		
0	RJ	ST3	ST	Store
1	RA	OK	CN1	Increment OK
2	RJ	CK3	CK1	Check size of routine
3	MJ	0	[30000]	Exit
	CA	RL4		

	IA	ST		
0	TV	GK	ST1	
1	TP	A	[30000]	Store
2	RA	GK	CN1	Increment GK
3	MJ	0	[30000]	Exit
	CA	ST4		

Error Print-Out Routine

	IA	CK		
0	SP	OK	0	} Sum 3 counters
1	SA	MK	0	
2	SA	UK	0	
3	TJ	CN32	[30000]	
4	RJ	WA	WA1	Print-Out: Sentence ____ (Type)
5	TP	CK10	UP3	} Print-Out: Generated Sentence Too Long
6	RJ	UP2	UP	
7	MJ	0	BQ6	Jump to rewind tapes and computer stop
10	40	CK11	5	Parameter for print-out routine
11	32	30503	05424	GENERA
12	66	30270	16530	TEDASE
13	50	66305	02630	NTENCE
14	01	66515	10146	ΔT00ΔL
15	51	50322	27777	ONG.
	CA	CK16		

Updating Op File 1 Item

	IA	FU		
0	LA	A	17	Shift CW to "u" field
1	TV	FU5	FU2	Where?
2	TP	A	[30000]	Store in item
3	RA	FU5	CN1	Increment counter
4	MJ	0	[30000]	Exit
5	0	0	[30000]	Counter
	CA	FU6		

Storing SS in Subscript List (SL)

	IA	SB		
0	TP	A	30000	Store subscript CW
1	RA	SB	CN1	Increment "v" address
2	MJ	0	30000	and exit
	CA	SB3		

Constants

	IA	CN	
0	0	0	0
1	0	0	1
2	0	0	2
3	0	0	6
4	0	0	45

5	0	0	117	
6	MJ	0	1000	
7	0	0	777	
10	0	0	35000	} Pseudo-address codes
11	0	0	36000	
12	0	0	37000	
13	0	0	50002	} Library call words
14	0	0	50062	
15	0	0	50112	
16	0	0	64000	
17	0	0	65000	
20	0	0	67000	
21	0	0	70000	
22	0	0	76000	
23	0	0	77000	
24	0	0	77777	v-mask
25	0	0	40	} Various string-out symbols
26	0	0	120	
27	0	1	0	
30	0	2	0	} Various string-out symbols
31	0	0	1000	
32	0	0	1776	
33	0	1	0	
34	0	37000	37000	
35	10	0	0	
36	10	0	3	
37	10	0	4	
	CA	CN40		
	IA	CN40		
40	10	2	0	
41	40	1	0	
42	57	04440	40000	
43	0	GL	FL	
44	77	0	0	
45	0	2	0	
46	TP	0	0	
47	TP	0	37000	
50	TP	35000	0	
51	TU	32000	1	
52	SP	0	0	
53	SA	0	17	
54	RJ	0	0	
55	TJ	0	2	
56	MJ	0	37002	
57	PR	0	0	

Check on length of gen. coding

60	MA	0	0
61	DV	0	31000
62	RP	30000	PR41
63	0	UL	0
64	0	0	UL
65	0	ML	ML
66	0	0	FL
67	0	SL	SL
70	0	0	GL6
71	0	CN14	0
72	0	CN15	0
73	0	BK4	FL2
	CA	CN74	

START

OK $\rightarrow$ 14₁₀
 TK $\rightarrow$ 3
 MK $\rightarrow$ 0
 Set up VK
 Start to scan at BK7
 Start to store subs at SL
 Start to store M's at MM

Prepare Op
 File 1 item
 and prelude
 so far as
 possible

A

Inspect rth
 word of
 stringout

$\geq 77000?$

Yes

No

D

Scan dimension
 list for this entry

Store this C/W
 in Op File I item

Extract 2nd line
 of dim. box,
 note # sub-
 scripts in ER

Build SL list
 (# subscripts of
 vbl's in
 stringout)

S = 1?

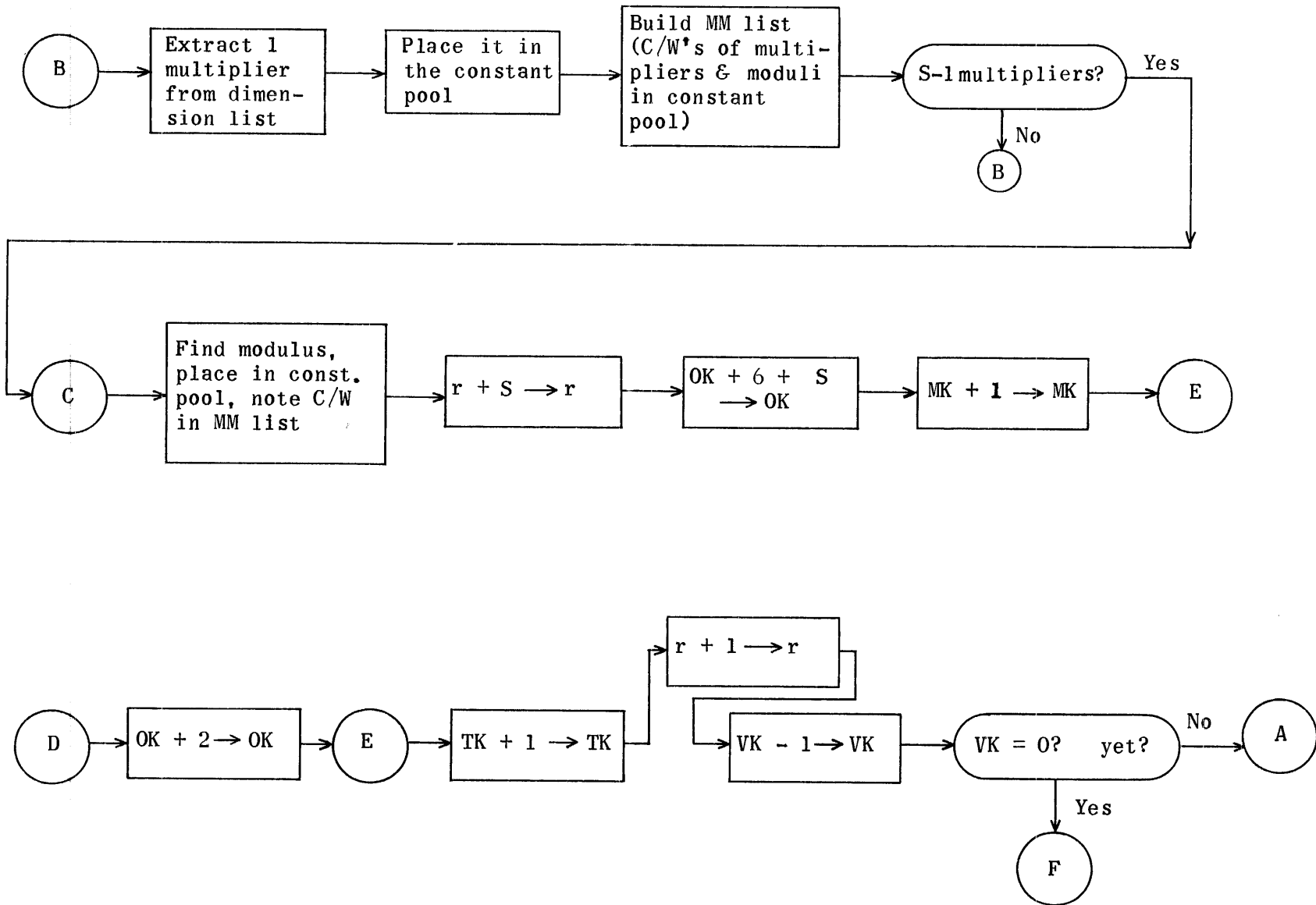
No

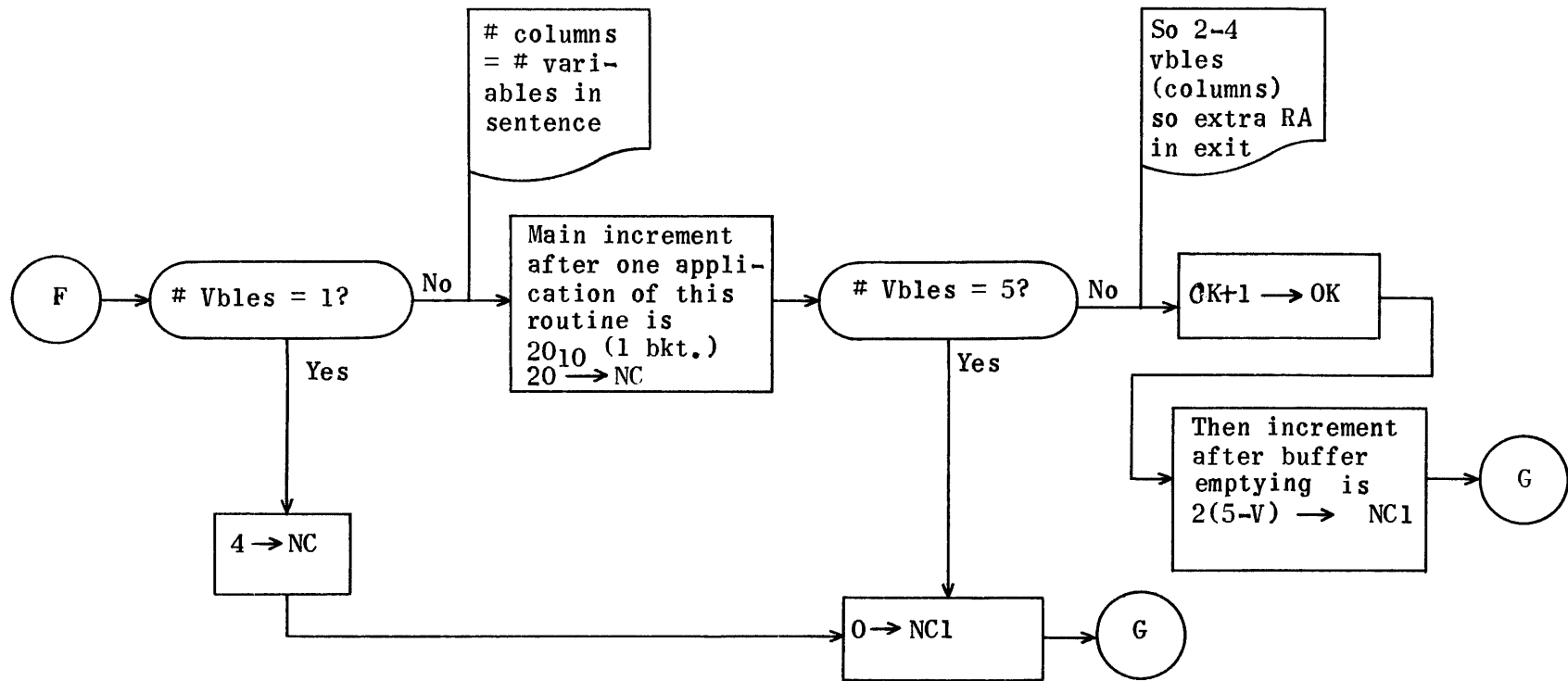
B

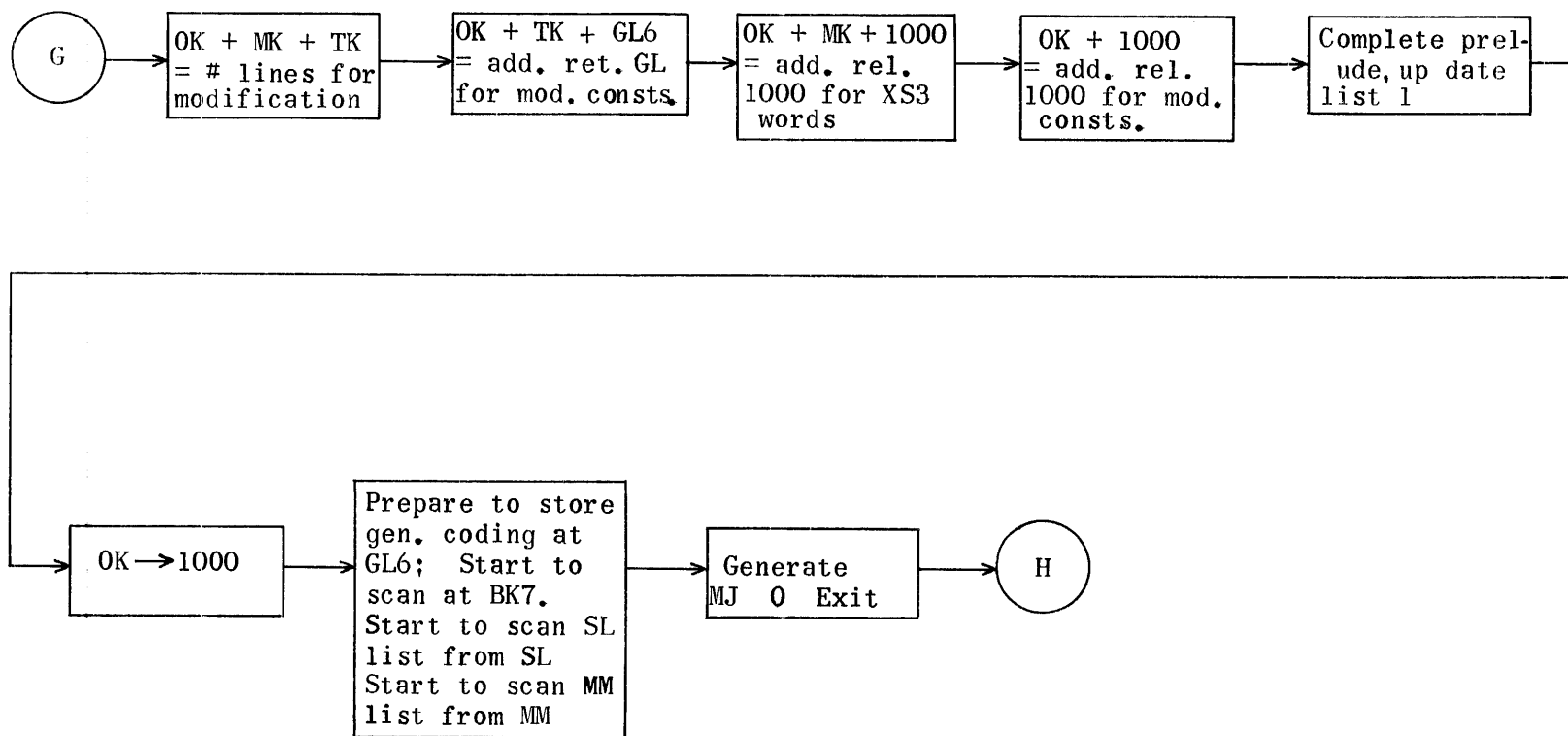
Yes

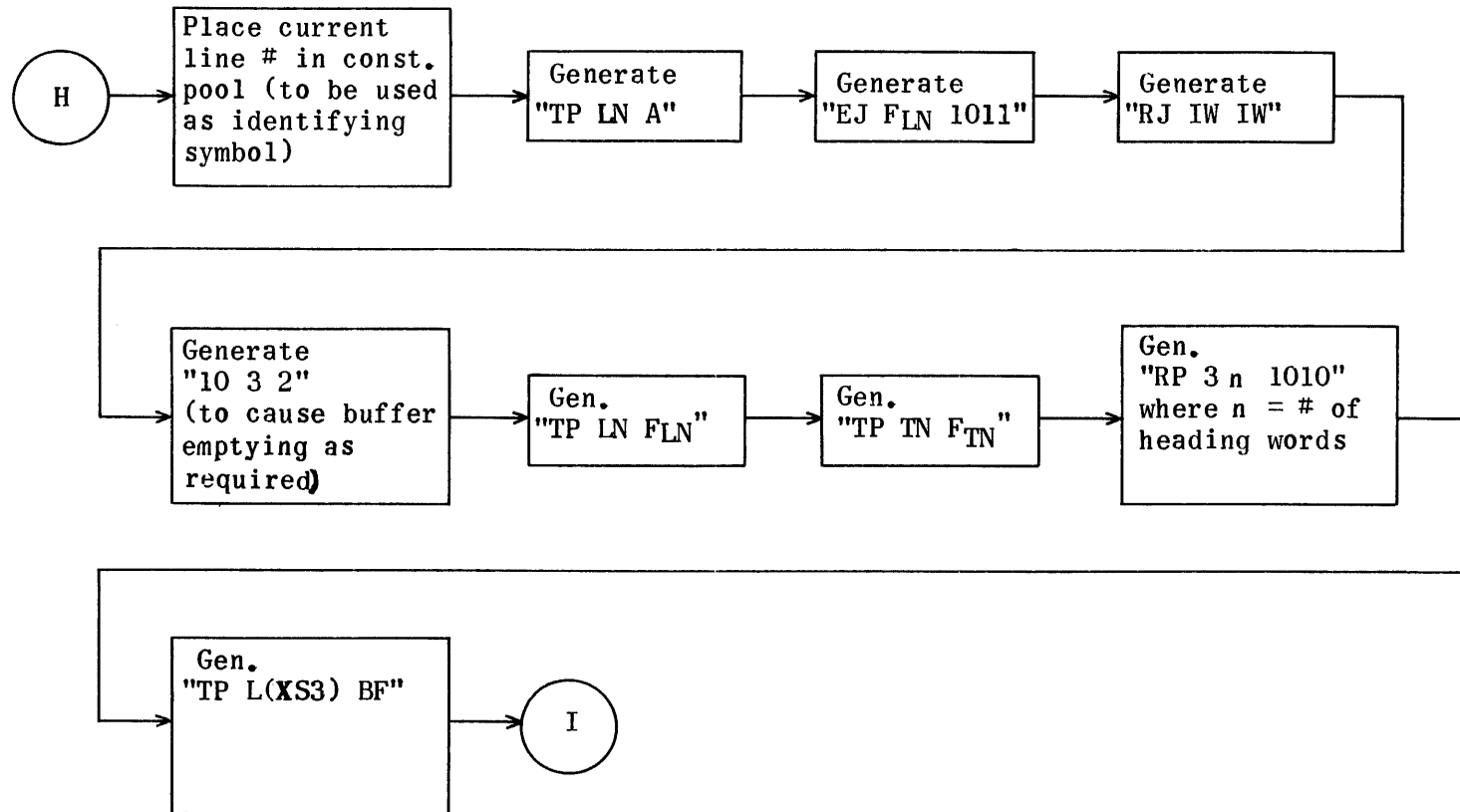
C

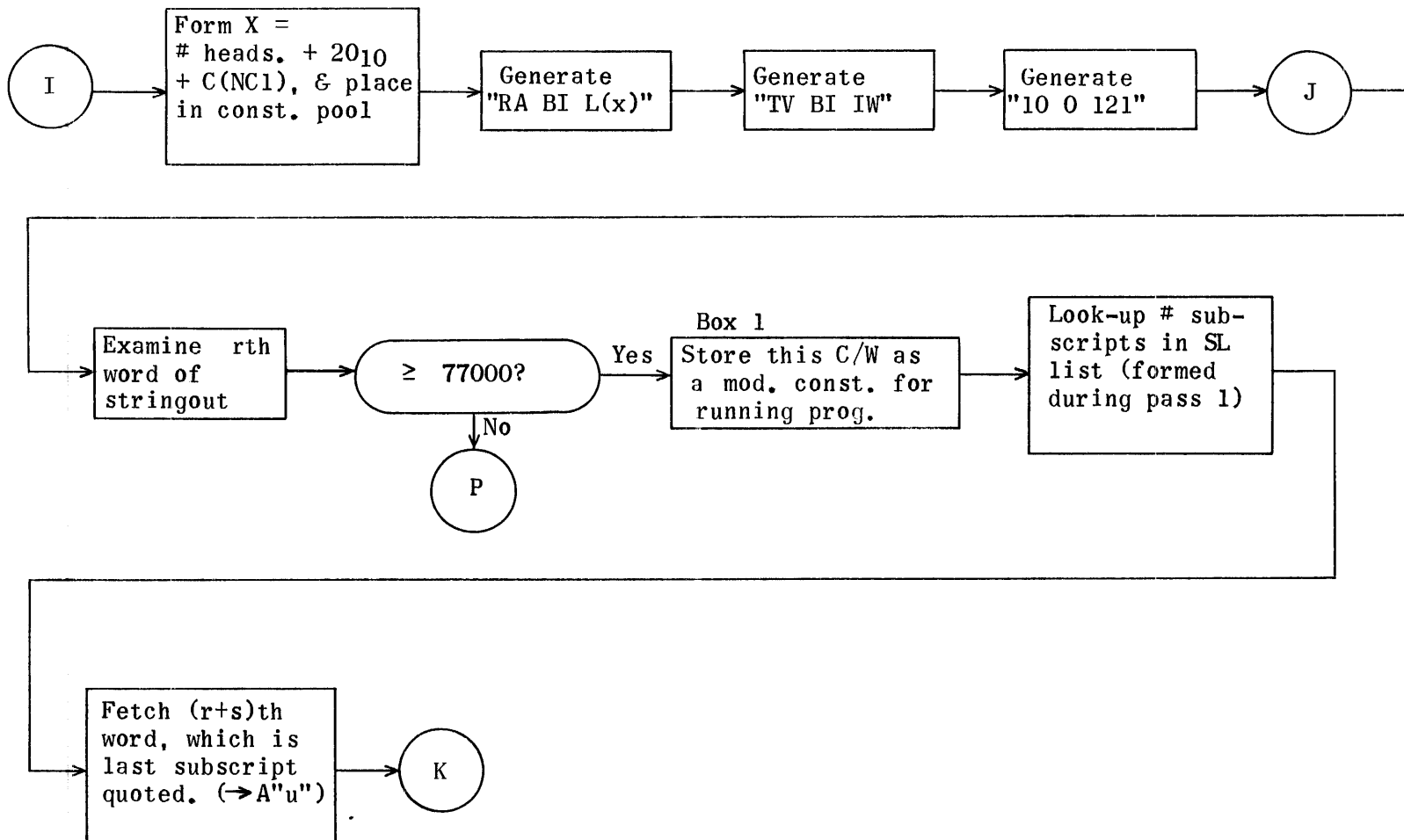
List Generator Flow Charts

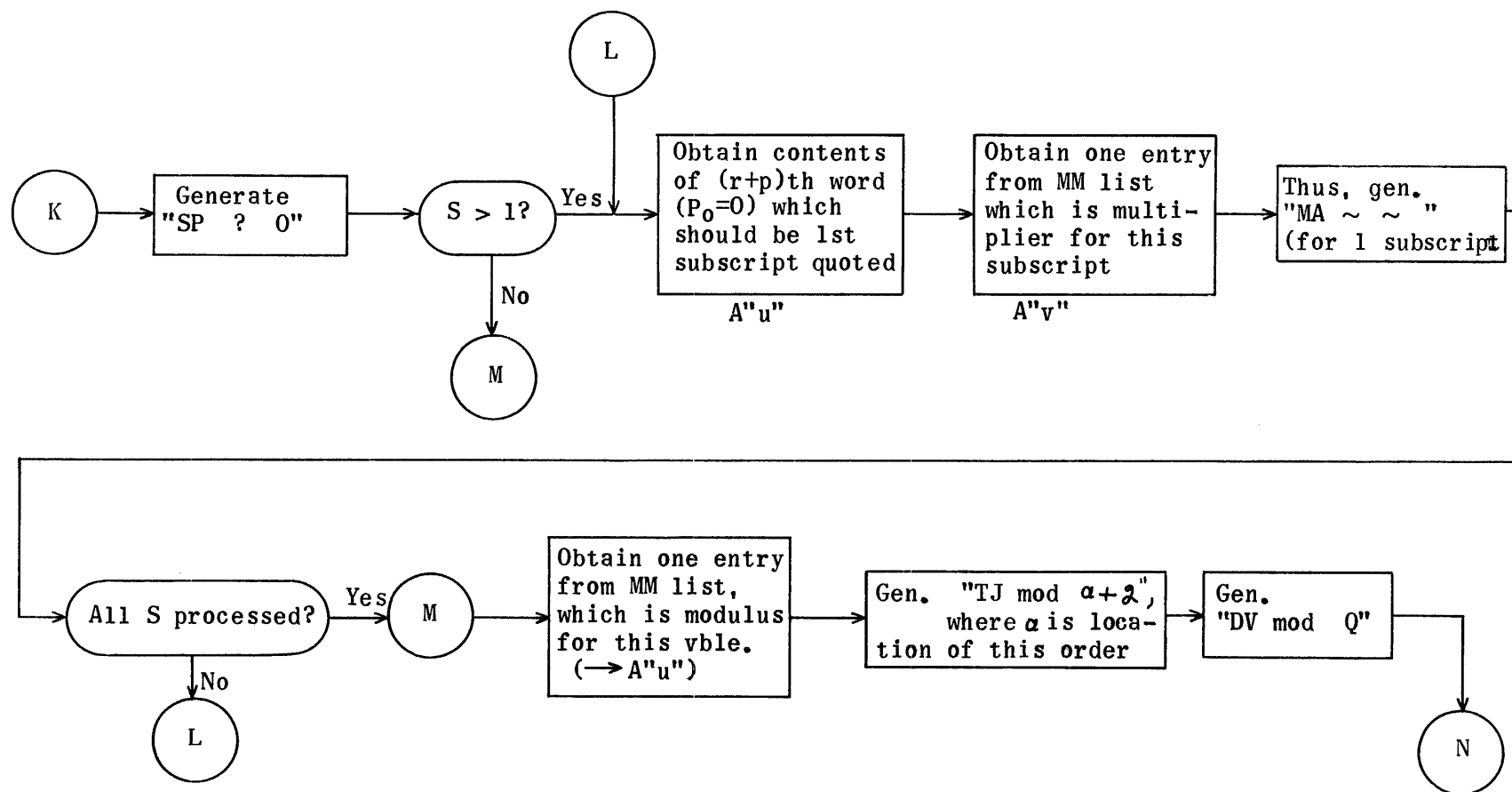


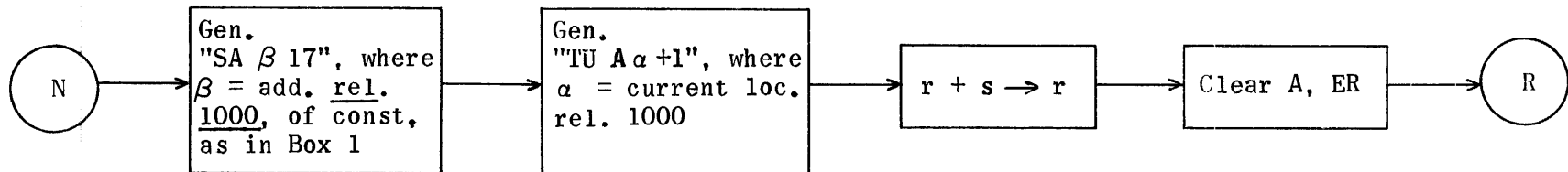


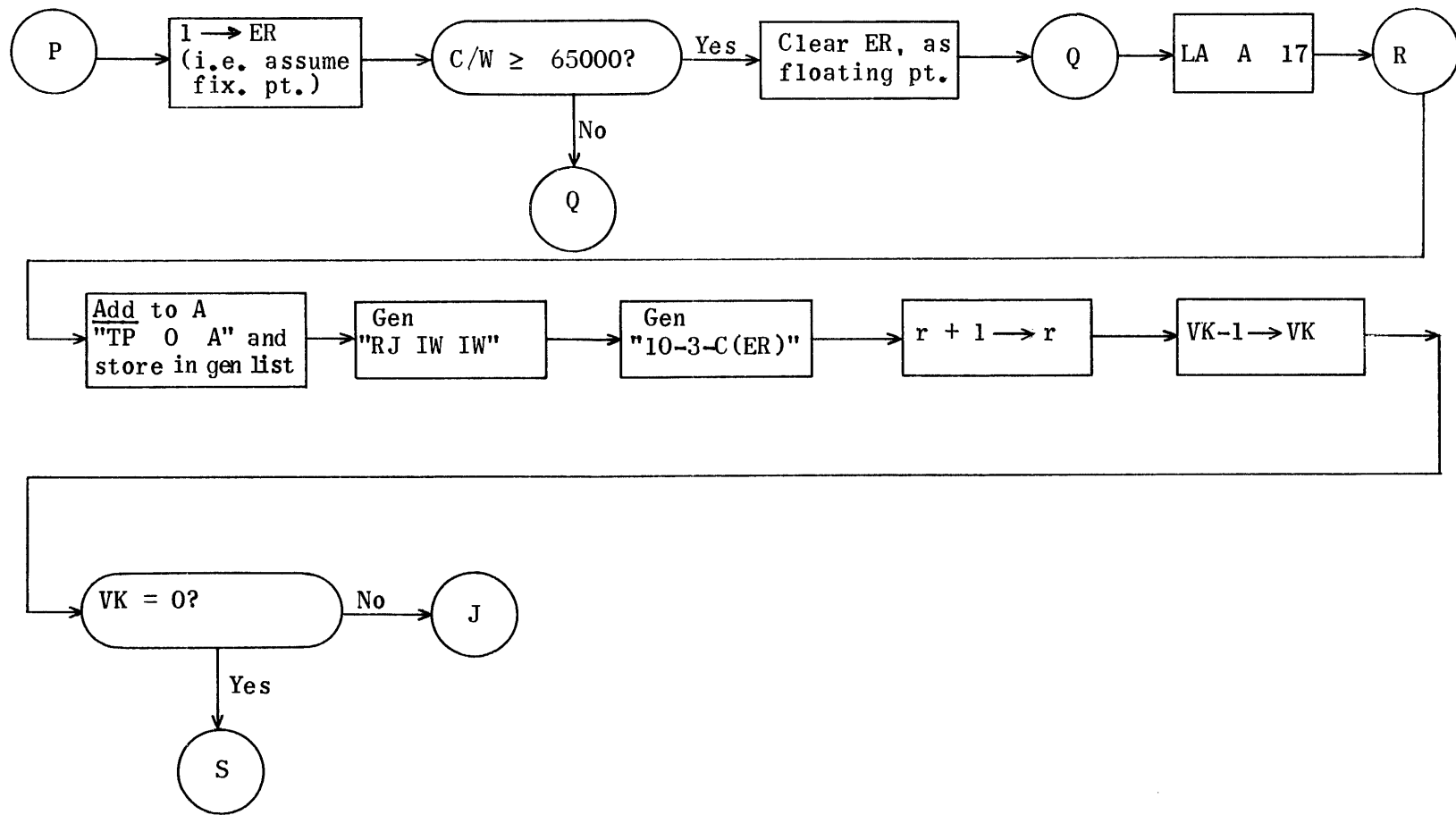


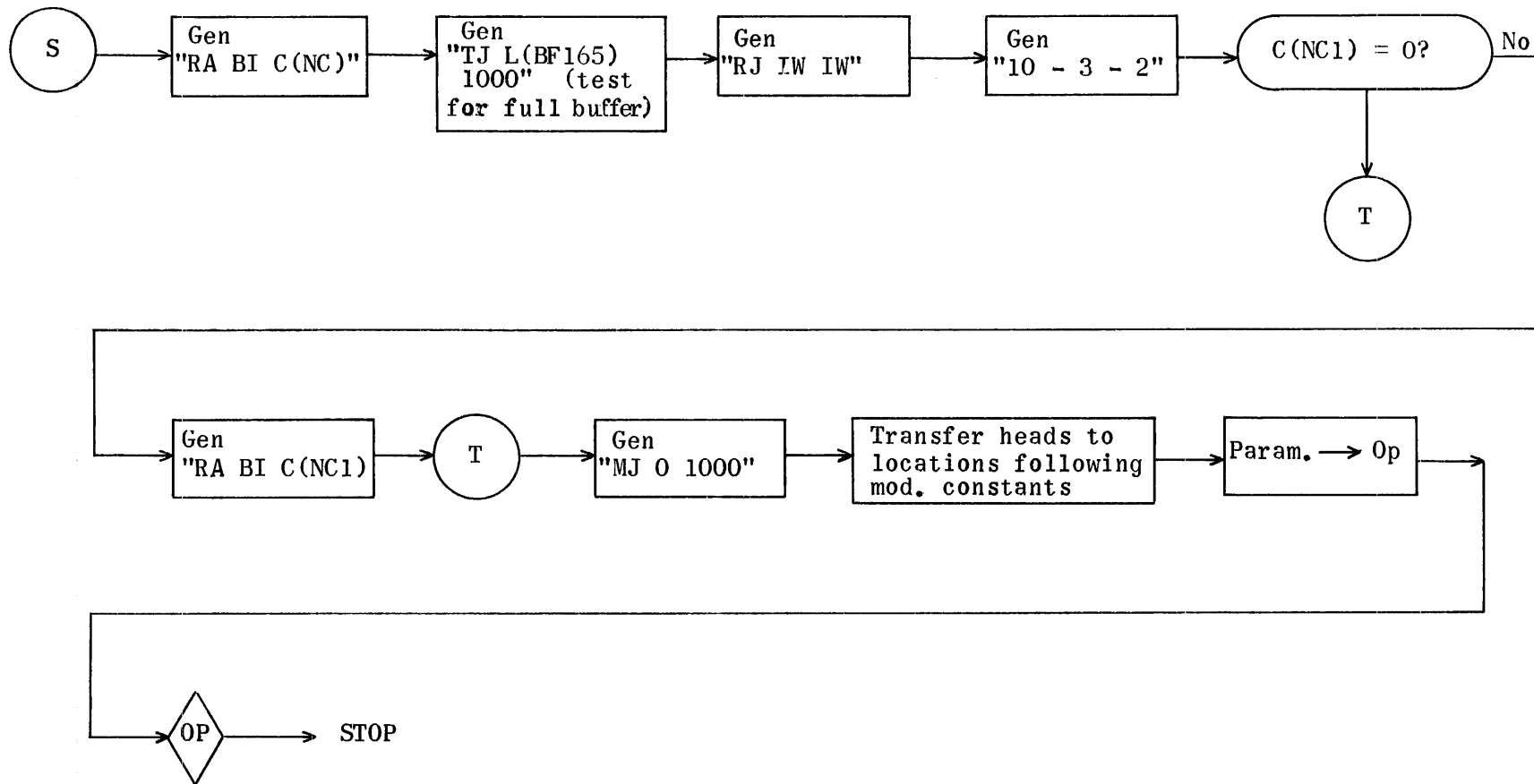












LIST GENERATOR

REGIONS

Generation Subroutines	RE	LS1465	Library List Routine	
	RE	CS1211	CS = CW Constant Call Word Routine	
	RE	BR537	Machine Error Routine	
	RE	OP1047	Routine to Transfer Generated Routine and Op File to Tape	
	RE	RW50023	List of call words of tape numbers referenced for use of STOP instruction	
	RE	DL40102	Dimension List	
	RE	BK2242	Buffer input region for string-out	
	RE	GL5360	Buffer output region for generated routine	
	RE	WW12	Fixed location where READ and LIST indicators are stored	
	RE	LI2512	Generator proper	
	RE	IN2513		
	RE	PA2532		
	RE	PB2632		
	RE	PC2666		
	RE	RL3113		
	RE	ST3116		
	RE	OR3121		
	RE	CN3136		
	RE	OK3173		
	RE	TK3174		
	RE	MK3175		
	RE	VK3176		
	RE	NC3177		
	RE	ER3201		
	RE	TC3204		
	RE	SL3205		
	RE	MM3212		
	RE	FL3236	Used to store Op File for LIST	
	RE	NL71002	Fixed output locations	7 in these addresses is a code figure to keep processor from modifying addresses
	RE	TN71003		
	RE	BI71004		
	RE	BF71005	LIST Buffer location	

IA	LI		
MJ	0	30000	Exit
CA	LI1		

Preliminary Initialization

	IA	IN		
0	TP	CN7	OK	1410 running lines at least
1	TP	CN3	TK	3 "10 lines"
2	TP	CN	MK	0 modifiable constants
3	TP	BK5	VK	Set up variable counter
4	TU	CN24	PA	Start scanning string-out at BK7
5	TV	CN25	PA15	Build subscript List from SL
6	TV	CN26	PA34	Build mult. & mod. List from MM
7	SP	BK3	17	} Routine CW to Prelude
10	TP	A	GL	
11	TP	CN27	FL2	
12	TV	CN24	PA5	Note CW of IW routine
13	AT	CN3	FL	Future cross-refs. to be stored at FL3
				Call word and at least no. 3 of lines to
				1st line of Op File
14	TP	CN	GL3	0 inputs
15	TP	CN	GL4	0 outputs
16	TP	BK1	GL5	Line number to Prelude
	CA	IN17		

"LIST" Generator - First Pass

	IA	PA		
0	SP	[30000]	0	Inspect rth word of string-out
1	TJ	CN14	PA55	Multivalued?
2	TU	6	PA3	Yes, scan Dimension List
3	RP	[30000]	BR1	
4	EJ	DL	PA5	
5	LT	10017	[30000]	} Build Op File 1 item
6	RA	FL	CN1	
7	SN	Q	17	
				-jn + r
10	SA	PA3	0	+r
11	SA	PA4	0	+DL => add. of 2nd line (modulus; S)
12	TU	A	PA13	
13	TP	[30000]	Q	
14	QT	CN20	ER	Using v-mask, note # subscripts (s)
15	TP	A	[30000]	Build subscript list
16	EJ	CN1	PA40	Is s = 1?
17	ST	CN2	ER1	No, form & store (S-2)
20	TU	PA13	PA37	Note address of 2nd line
21	RA	PA13	CN15	Form address of 3rd line
22	TU	A	PA24	
23	TP	PA76	PA25	Set flip-flop of m-extractor
24	TP	[30000]	Q	1 line of dim. box -> Q
25	[0	30000	30000]	① TP ② PA25; ② MJ 0 PA27

26	MJ	0	PA32	Jump to extract v-field
27	LQ	Q	25	Q "u" $\rightarrow$ Q "v"
30	RA	PA24	CN15	Modify for next line
31	TP	PA76	PA25	Reset flip-flop to ①
32	QT	CN20	A	Extract v field
33	RJ	CS	CS1	Insert in constant pool, and find CW
34	TP	A	[30000]	Build MM list (A "u")
35	RA	PA34	CN1	
36	IJ	ER1	PA24	Back for more multipliers
37	TP	[30000]	Q	2nd line again to Q
	CA	PA40		
	IA	PA40		
40	LQ	Q	25	Shift modulus to "v" and extract it
41	QT	CN20	A	
42	RJ	CS	CS1	Place in pool
43	TV	PA34	PA44	} and place in MM list
44	TP	A	[30000]	
45	RA	PA34	CN1	
46	SP	ER	17	S $\rightarrow$ A "u"
47	AT	PA	PA	Bring r count up to date
50	RA	PA5	CN1	Op File Item building op. updated
51	RA	PA15	CN1	SL list building op. updated
52	RA	OK	CN4	OK + 4 $\rightarrow$ OK
53	AT	ER	OK	OK + 5 $\rightarrow$ OK
54	RA	MK	CN1	1 mod constant
55	RA	OK	CN2	OK + 2 $\rightarrow$ OK (S.V. variable entry)
56	RA	TK	CN1	One "10 line"
57	RA	PA	CN15	r + 1 $\rightarrow$ r
60	RS	VK	CN1	vk - 1 $\rightarrow$ vk
61	ZJ	PA	PA62	If zero, wrap up this pass.
62	SP	BK5	0	Number variables $\rightarrow$ A
63	EJ	CN1	PA73	= 1?
64	TP	CN10	NC	No, > 1. 20 ₁₀ $\rightarrow$ INC
65	EJ	CN5	PA74	= 5?
66	RA	OK	CN1	No, OK + 1 $\rightarrow$ OK
67	SP	CN5	0	5 $\rightarrow$ A
70	SS	BK5	1	} 2(5 - v) = SUBINC
71	TP	A	NC1	
72	MJ	0	PB	
73	TP	CN4	NC	Jump to initialization, Pass 2
74	TP	CN	NC1	1 vble, INC = 4
75	MJ	0	PB	1 or 5 vbles., SUBINC = 0
76	TP	PA77	PA25	Jump to init., Pass 2
77	MJ	0	PA27	} Constants for m flip-flop
	CA	PA100		

Initialization, Pass 2

	IA	PB		
0	SP	OK	0	
1	AT	MK	ER	
2	AT	TK	GL1	$C(OK)+C(MK)+C(TK) = \# \text{ lines for add. mod.}$
3	SS	MK	0	
4	SA	CN23	0	$C(OK)+C(TK)+GL6 = \text{add. rel. GL for mods.}$
5	TV	A	PC53	
6	SP	ER	0	$C(OK)+C(MK)+1000$
7	AT	CN12	TC	$= XS3 \text{ add. rel. } 1000$
10	ST	MK	MK	$C(OK)+1000 = \text{mod. const. add. rel. } 1000$
11	TP	BK6	GL2	$\# \text{ unmods.}$
12	SP	BK6	0	
13	AT	ER	FL1	Total number of running lines
14	TP	BK5	VK	Set variable counter
15	TU	CN24	PC51	Set to scan at BK7
16	TU	CN25	PC55	Start obtaining $\#$ subs from SL list
17	TU	CN26	PC74	Start obtaining mults. and mods. from MM list
20	TP	CN12	OK	Set OK to 1000
21	TV	CN23	ST	Start storing output at GL6
22	SP	CN27	0	"50077" to A
23	RJ	LS	LS1	Update list 1
24	SP	GL1	0	
25	SA	GL2	0	
26	SA	CN6	0	$+ 6 = \# \text{ lines for prelude}$
27	TV	A	GL	
30	SP	TC	0	$XS3 \text{ add. rel. } 1000$
31	SA	BK6	0	$+ \# \text{ heads}$
32	SA	OR14	0	
33	RJ	RL2	RL	Store MJ 0 Exit
	CA	PB34		

Second Pass

	IA	PC		
0	SP	BK1	0	This line number
1	RJ	CS	CS1	Insert in constant pool (CW $\rightarrow$ A "u")
2	SA	OR	0	Add in TP —A
3	RJ	RL2	RL	
4	SP	OR1	0	EJ NL 1011
5	RJ	RL2	RL	
6	SP	OR2	0	RJ IW IW
7	RJ	RL2	RL	
10	SP	CN16	0	Basic "ten line"
11	SA	CN2	0	$+ 2 \text{ in "v"}$
12	RJ	ST2	ST	Store it
13	TP	GL7	ER	"TP CW (LN) A

14	TV	CN21	ER	→"TP CW(LN) F _{LN}
15	SP	ER	0	
16	RJ	RL2	RL	
17	TP	OR	ER	
				TP CW(TN) F _{TN}
20	TU	BK4	ER	
21	TV	CN22	ER	
22	SP	ER	0	
23	RJ	RL2	RL	Store this line
24	SP	GL2	17	# heads → A "u"
25	SA	OR3	0	→ RP 3n 1010
26	RJ	RL2	RL	Store
27	SP	TC	17	XS3 add. rel. 1000
30	TP	OR	ER	
31	TU	A	ER	TP L(XS3) BF
32	TV	CN17	ER	
33	SP	ER	0	
34	RJ	RL2	RL	Store this line
35	SP	GL2	0	# heads
36	SA	CN10	0	+ 20 ₁₀
37	SA	NC1	0	+ SUBINC = total RA const.
	CA	PC40		
	IA	PC40		
40	RJ	CS	CS1	Insert in constant pool
41	SP	OR4	0	
42	SA	Q	0	RA BI L (INC 1) to output
43	RJ	RL2	RL	
44	SP	OR5	0	TV BI IW to output
45	RJ	RL2	RL	
46	SP	CN32	0	Basic 10 line
47	SA	CN11	0	+ appropriate value to reach IW "store" line
50	RJ	ST2	ST	
51	SP	[30000]	0	Examine rth word
52	TJ	CN14	PC130	Multivalued
53	TP	A	[30000]	Yes, store this CW as a running mod.const.
54	RA	PC53	CN1	Set for next time
55	TP	[30000]	ER	# SS → ER
56	RA	PC55	CN15	Modify
57	SP	ER	17	# SS → A "u"
60	SA	PC51	0	Find location of
61	TU	A	PC62	Last subscript for "SP ____ 0" order
62	SP	[30000]	17	Last CW → A "u"
63	SA	OR6	0	Add in SP ____ 0
64	RJ	RL2	RL	Store this line
65	SP	ER	0	More than 1 subscript?
66	EJ	CN1	PC102	
67	ST	CN2	ER1	Yes, form and store (S-2)
70	SP	PC51	0	
71	SA	CN15	0	Add 1 u to find first subscript

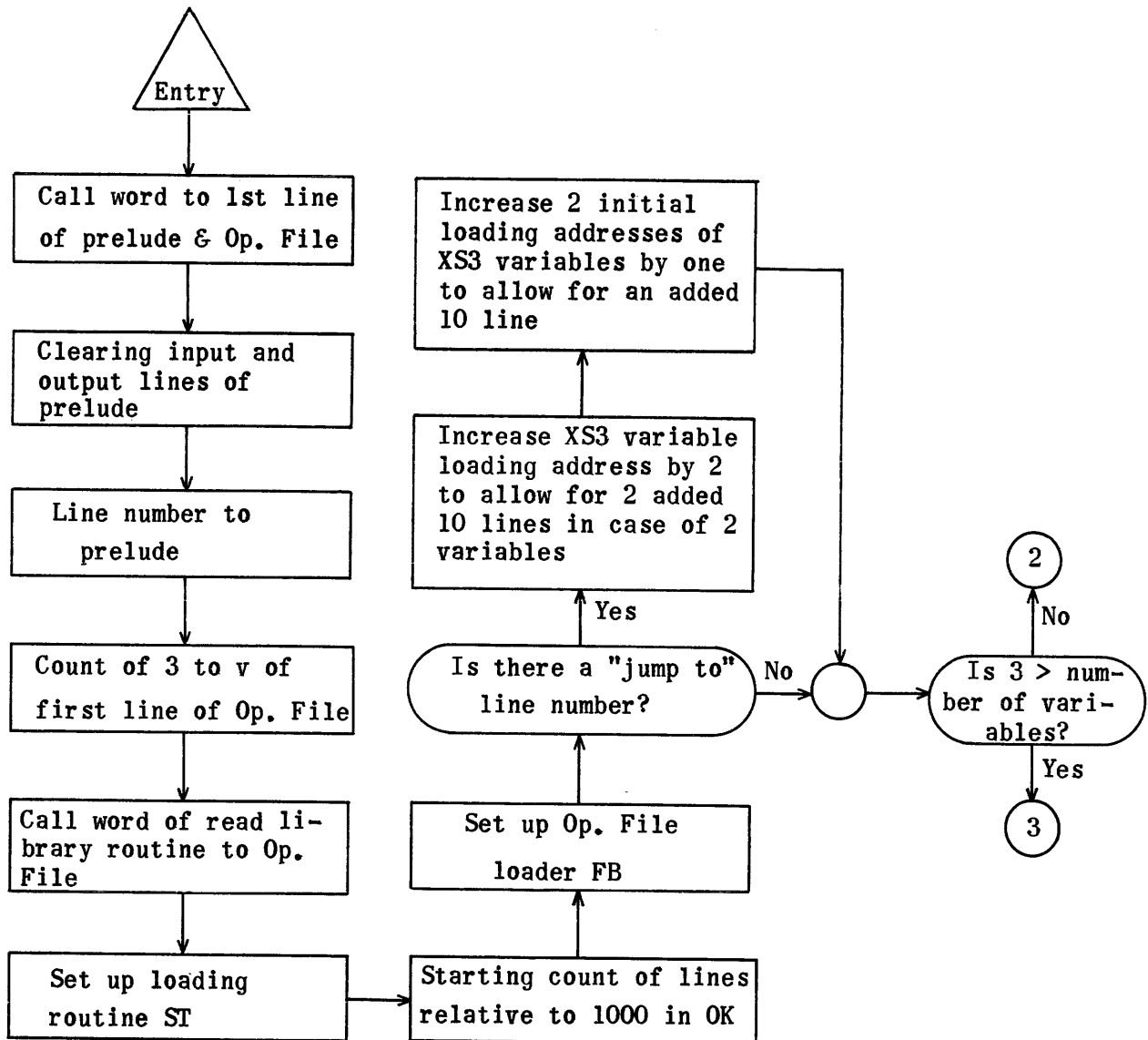
72	TU	A	PC73	
73	SP	[30000]	0	Subscript → v field
74	SA	[30000]	0	Multiplier → u field
75	SA	OR7	0	MA → Op field
76	RJ	RL2	RL	Store
77	RA	PC73	CN15	Modify to pick up next subscript
	CA	PC100		
	IA	PC100		
100	RA	PC74	CN15	To pick up next multiplier
101	IJ	ER1	PC73	
102	TU	PC74	PC105	To pick up modulus
103	RA	PC74	CN15	Set for next time
104	SP	OR10	0	TJ — 2
105	TP	[30000]	ER1	Save CW (Mod)
106	SA	ER1	0	TJ CW (Mod) 2
107	SA	OK	0	Complete TJ line
110	RJ	RL2	RL	Store
111	SP	ER1	0	— CW (Mod) —
112	SA	OR11	0	DV — Q
113	RJ	RL2	RL	Store
114	SP	MK	17	Add. rel. 1000 of mod. consts.
115	SA	OR12	0	"SA CW (Base) 17"
116	RJ	RL2	RL	
117	RA	MK	CN1	Modify MK for next time
120	SP	OR13	0	TU A 1
121	SA	OK	0	Complete line
122	RJ	RL2	RL	
123	SP	ER	17	S → A "u"
124	AT	PC51	PC51	r + s → r
125	TP	CN	A	Clear A and ER
126	TP	A	ER	then jump to form
127	MJ	0	PC134	TP [] A
130	TP	CN1	ER	Single valued. 1 → ER "v" (i.e., assume fixed)
131	TJ	CN13	PC133	Text v, 65000
132	TP	CN	ER	Floating, clear ER "v"
133	LA	A	17	CW → A "u"
134	SA	OR	0	TP — A
135	RJ	RL2	RL	Store
136	SP	OR2	0	RJ IW IW
137	RJ	RL2	RL	Store this line
	CA	PC140		

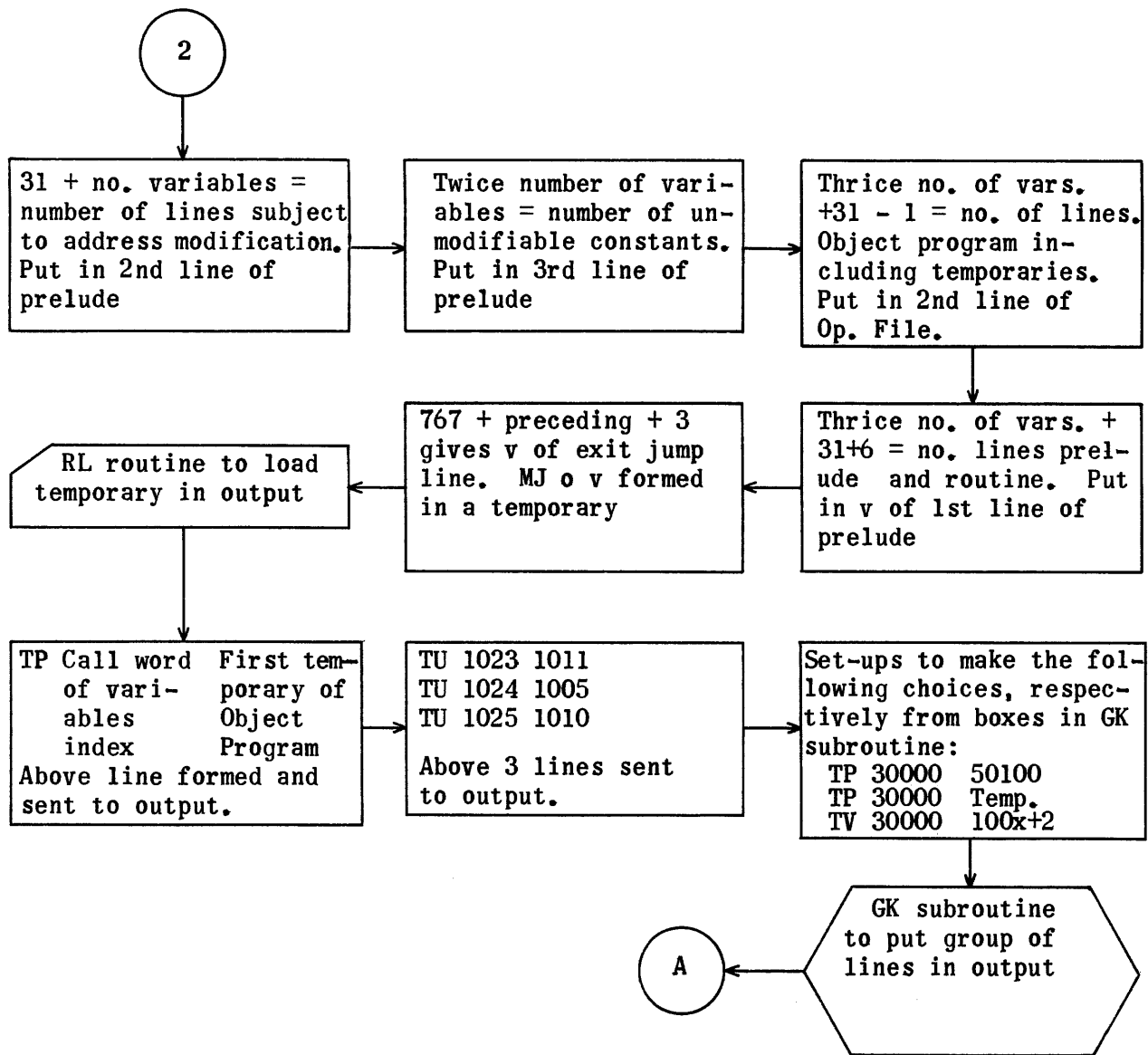
	IA	PC140		
140	SP	CN16	0	Basic 10-line (10 3 0)
141	SA	ER	0	Plus 0 or 1
142	RJ	ST2	ST	
143	RA	PC51	CN15	$r + 1 \rightarrow r$
144	RS	VK	CN1	Vble count decreased by 1
145	ZJ	PC51	PC146	
146	SP	NC	0	INC $\rightarrow$ A
147	RJ	CS	CS1	Place in const. pool
150	SP	OR4	0	RA BI —
151	SA	Q	0	Complete line
152	RJ	RL2	RL	Store
153	SP	CN30	0	BF165 $\rightarrow$ A
154	RJ	CS	CS1	Insert in pool
155	TP	OR10	ER	} TJ L(BF165) 1000 formed and put in output
156	TU	A	ER	
157	TV	CN12	ER	
160	SP	ER	0	} RJ IW IW to output
161	RJ	RL2	RL	
162	SP	OR2	0	
163	RJ	RL2	RL	
164	SP	CN16	0	Basic 10-line
165	SA	CN2	0	Add 2 to complete it
166	RJ	ST2	ST	
167	SP	NC1	0	Subinc $\rightarrow$ A
170	ZJ	PC171	PC175	Zero?
171	RJ	CS	CS1	No, insert in pool
172	SP	OR4	0	} RA BI Subinc formed & put in output
173	SA	Q	0	
174	RJ	RL2	RL	
175	TP	OR14	ER	
176	TV	CN12	ER	} MJ 0 1000 formed in A
177	SP	ER	0	
	CA	PC200		
	IA	PC200		
200	RJ	RL2	RL	Store
201	SP	GL2	17	# heads $\rightarrow$ A "u"
202	SA	CN17	0	+ 30000
203	TU	A	PC206	} Transfer heading codes
204	TU	PC51	PC207	
205	TV	PC53	PC207	
206	RP	[0]	PC210	
207	TP	[30000]	[30000]	
210	TP	CN31	OP1	Parameter to output handler
211	RJ	OP	OP2	Go output
212	TP	CN15	Q	$1 \rightarrow Q$
213	QS	CN15	WW	Inform processor of "LIST"

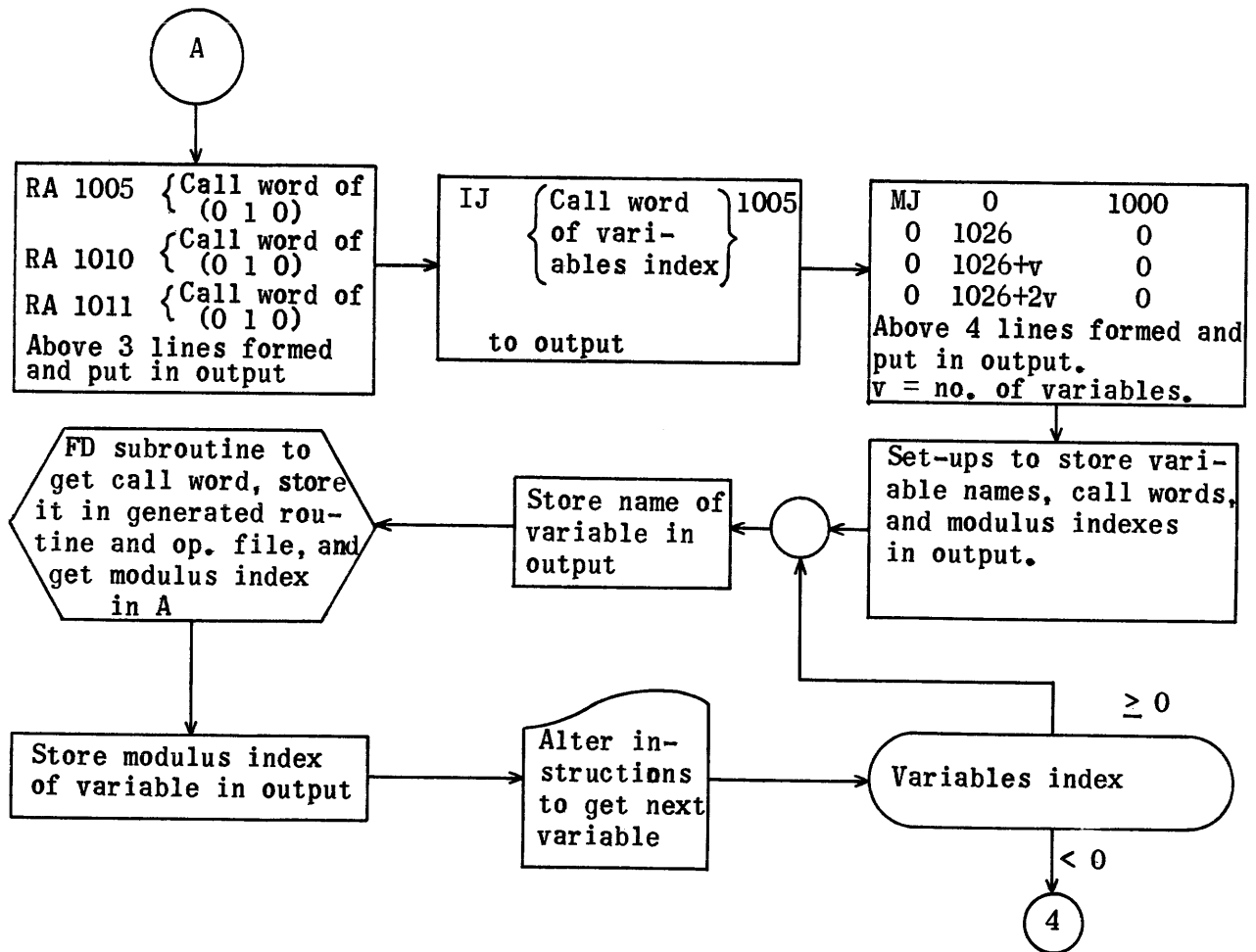
214	SP	BK4	0	Tape # CW → A
215	TU	RW	PC216	
216	RP	[0]	PC220	
217	EJ	RW1	PC224	} Already in list of CW's?
220	RA	RW	CN33	No, increment index by 1
221	SA	CN34	0	Add [RW]
222	TV	A	PC223	
223	TP	BK4	[30000]	And insert in list
224	MJ	0	LI	Exit
	CA	PC225		
	IA	RL		
0	RJ	ST2	ST	
1	RA	OK	CN1	
2	MJ	0	[30000]	
	CA	RL3		
	IA	ST		
0	TP	A	[30000]	
1	RA	ST	CN1	
2	MJ	0	[30000]	
	CA	ST3		
	IA	OR		
0	TP	0	32000	
1	EJ	NL	1011	
2	RJ	50077	50077	
3	RP	[30000]	1010	
4	RA	BI	0	
5	TV	BI	50077	
6	SP	0	0	
7	MA	0	0	
10	TJ	0	2	
11	DV	0	31000	
12	SA	0	17	
13	TU	32000	1	
14	MJ	0	1	
	CA	OR15		
	IA	CN		
0	0	0	0	
1	0	0	1	
2	0	0	2	
3	0	0	3	
4	0	0	4	
5	0	0	5	
6	0	0	6	
7	0	0	16	
10	0	0	24	
11	0	0	121	

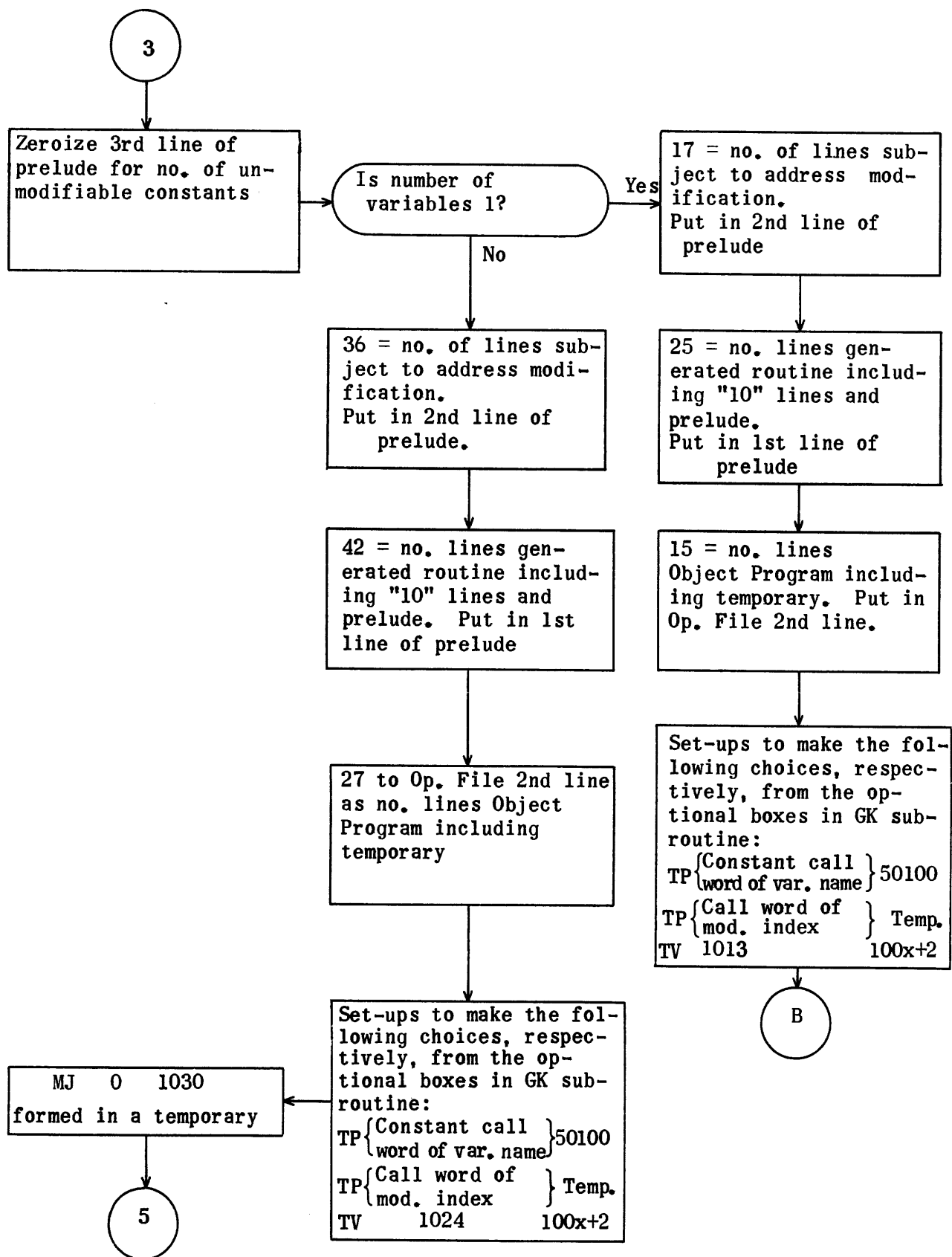
12	0	0	1000
13	0	0	65000
14	0	0	77000
15	0	1	0
16	10	3	0
17	0	30000	BF
20	0	0	77777
21	0	0	NL
22	0	0	TN
23	0	0	GL6
24	0	BK7	FL3
25	0	SL	SL
26	0	MM	MM
27	0	50077	0
30	0	0	1172
31	0	GL	FL
32	10	0	0
33	0	1	1
34	0	0	RW
	CA	CN35	

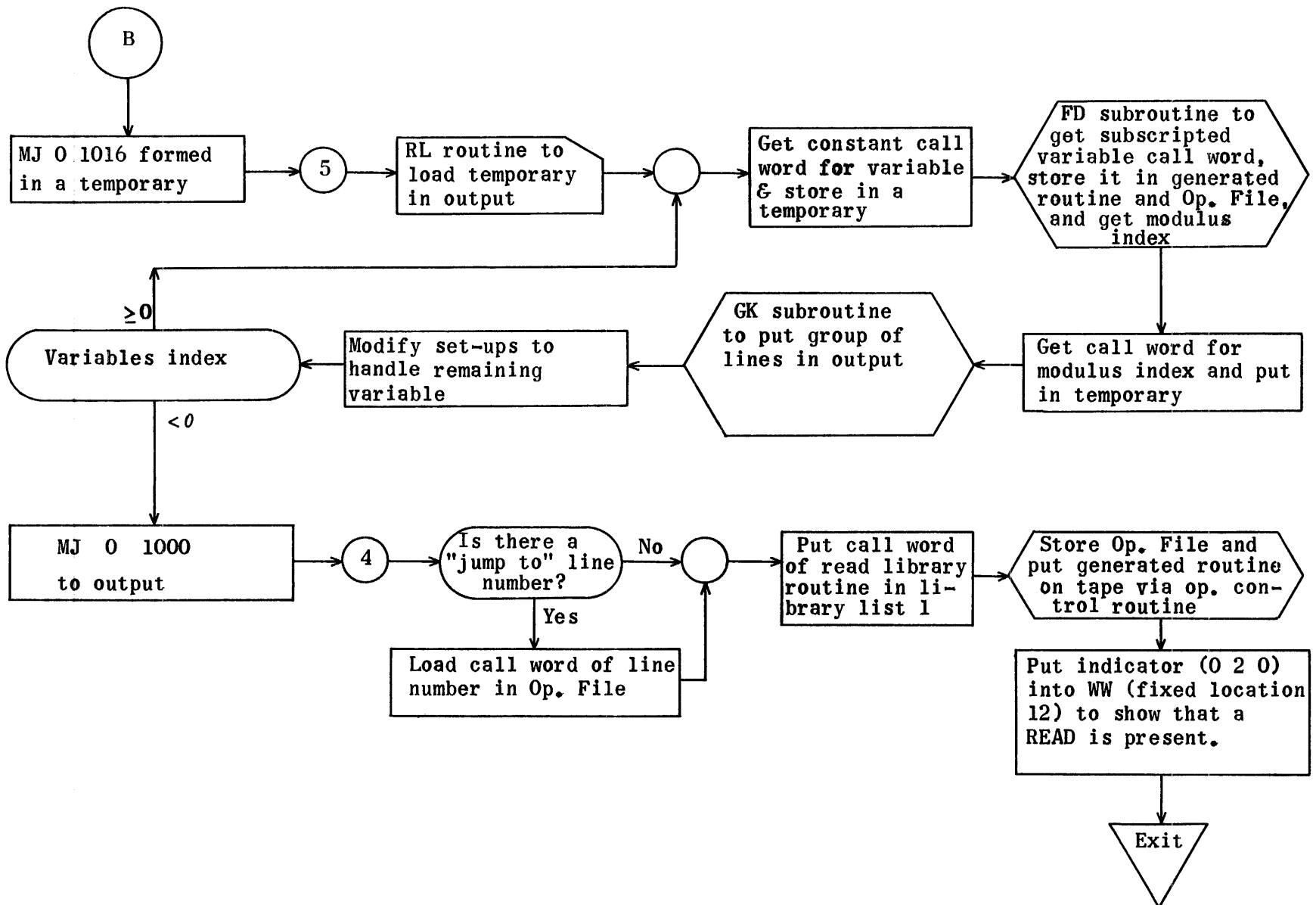
Flow Charts of Read Generator



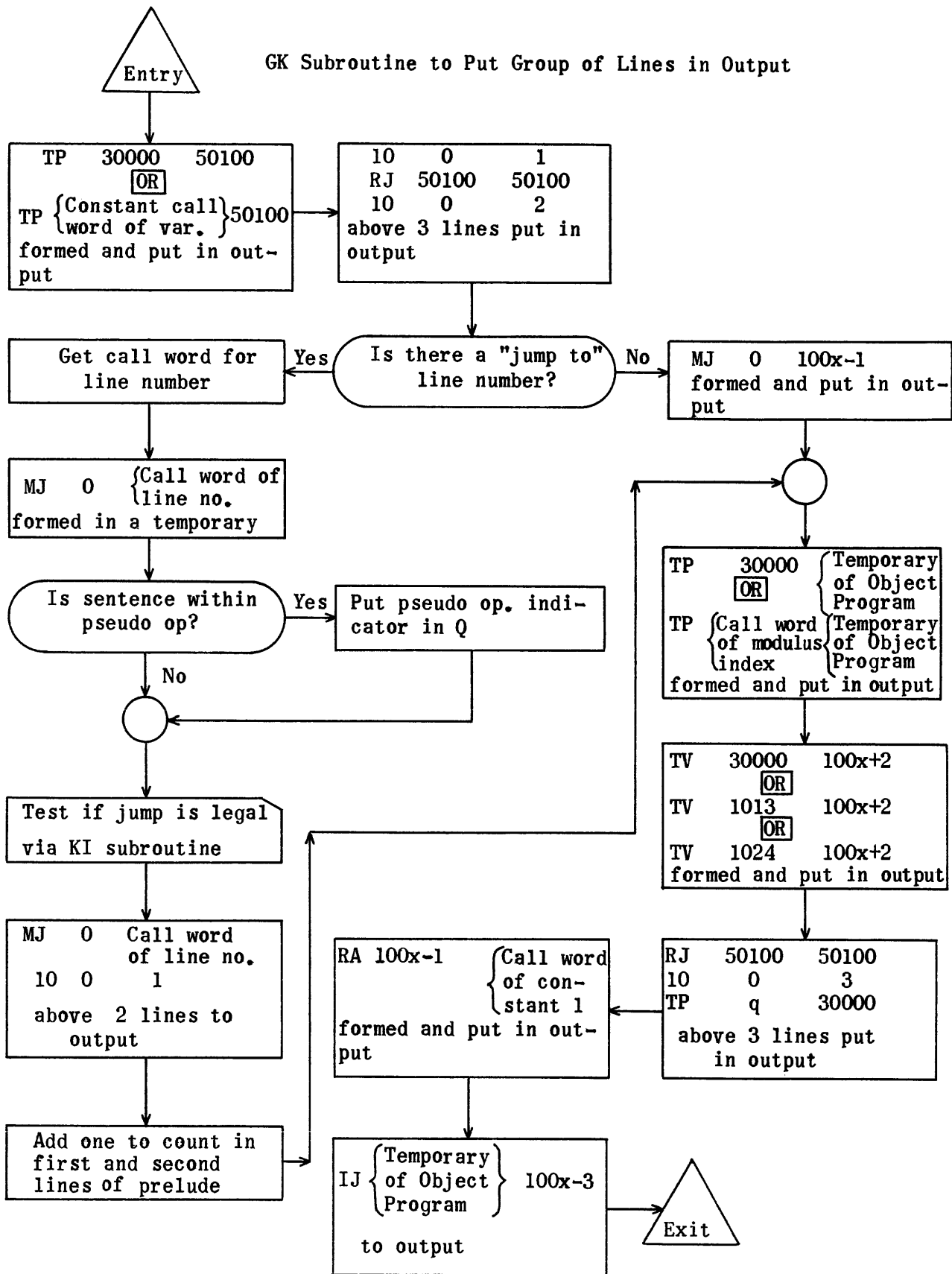




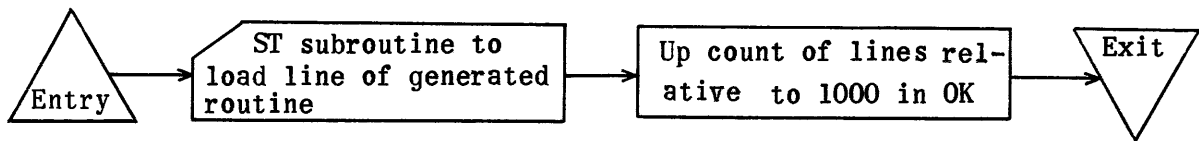




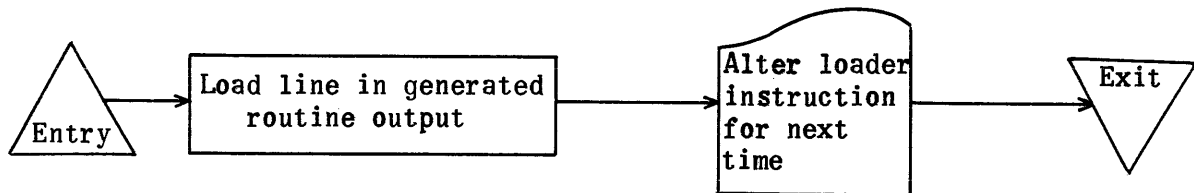
GK Subroutine to Put Group of Lines in Output



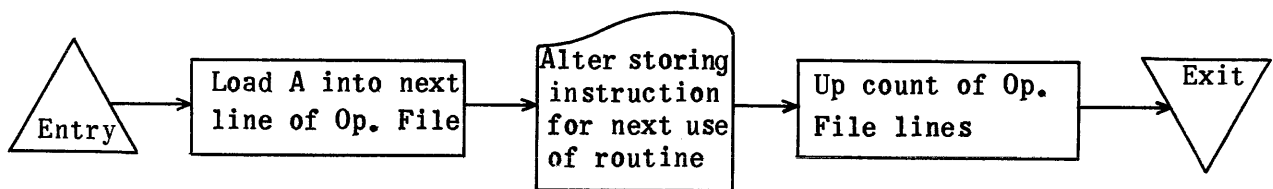
RL Routine to Load Temporary in Output



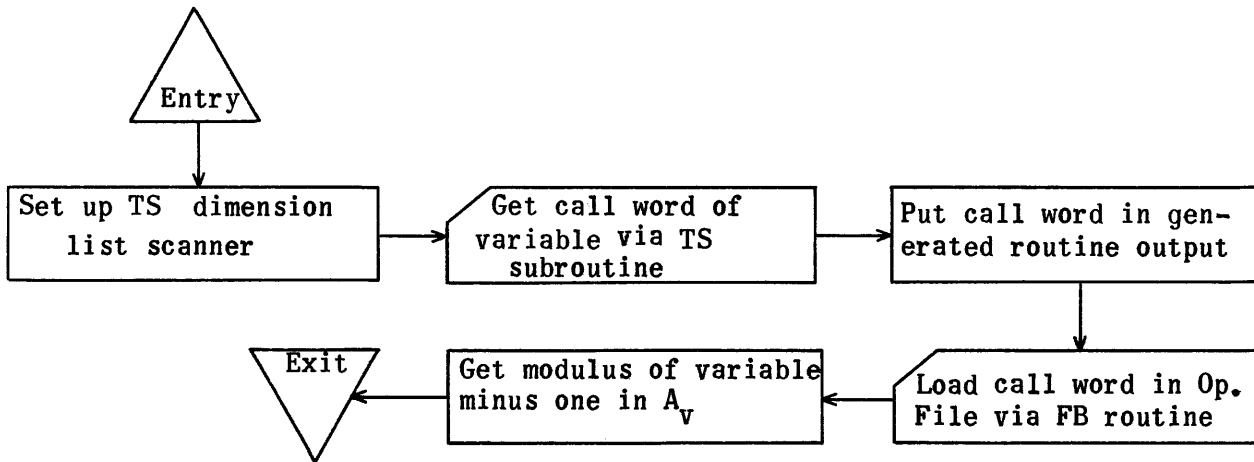
ST Subroutine to Load Lines of Generated Routine



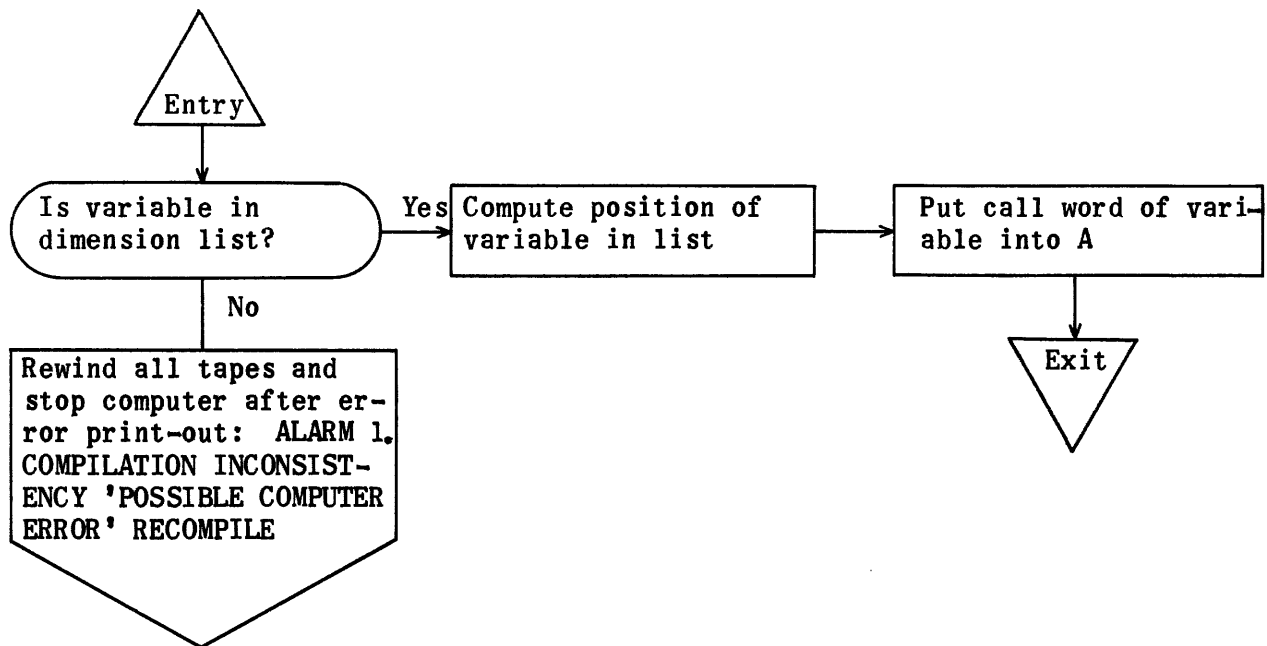
FB Subroutine to Load Op. File



FD Subroutine to Get Call Word of Variable, Store It in Output, Put it in Op. File and Get Modulus Index in A



TS Subroutine to Get Call Word of Subscripted Variable From Dimension List



READ GENERATOR REGIONS

	RE	CS1211	}	= CW, Call-word subroutine
	RE	LS1465		
	RE	OP1047	}	Generation subroutines
	RE	BR537		
	RE	LW1250		
	RE	KI1336		
	RE	VX1670		
	RE	BP564		
	RE	BQ632		
	RE	DL40102		Dimension List
	RE	LL47246		= IZ, Referenced-line-number List
	RE	BK2242		Input buffer
	RE	GL5360		Generated routines output buffer
	RE	WW12		Fixed location holding read indicator
Generator + tempo- raries	RE	RE2512	}	
	RE	IN2513		
	RE	FW2647		
	RE	GK2726		
	RE	RL3041		
	RE	ST3044		
	RE	FD3047		
	RE	TS3063		
	RE	FB3073		
	RE	CN3077		
	RE	OK3142		
	RE	ER3143		
	RE	FL3150		
				Where Op File I is built up

READ GENERATOR

Entry	IA	RE	MJ	0	30000	Exit
	CA	RE1				
	IA	IN				
0	SP	BK3	17			
1	TP	A	GL			Call word to 1st line of prelude, GL _u
2	TP	A	FL			Call word to 1st line of Op File 1 item
3	TP	CN	GL3			
4	TP	CN	GL4			Clearing input and output lines of prelude
5	TP	BK1	GL5			Line number to prelude
6	TV	CN3	FL			Number (3) to v of 1st line of Op File
7	TP	CN40	FL2			Call word (50100) to 3rd line of Op File.
10	TV	CN31	ST			Library call word reference
11	TP	CN13	OK			Initializing ST (load routine) to GL6
12	TV	CN32	FB			Initializing OK to read 0 0 1000
13	SP	BK4	0			Sets up v of FB to FL3
14	ZJ	GK110	IN15			Is there a "jump to" line number in the sentence?
15	SP	BK5	0			
16	TJ	CN3	FW			Is 3 > number of variables?
17	AT	CN10	GL1			31 + number variables = no. lines subject to address modification
20	SP	BK5	1			Number of unmodifiable constants = twice number of variables
21	TP	A	GL2			
22	SA	GL1	0			Thrice no. of variables + 31 - 1 = no. of lines of running program including temporaries
23	ST	CN1	FL1			
24	SA	CN4	0			Thrice no. of variables + 31 + 6 = no. of lines prelude and routine
25	TV	A	GL			767 + above enumeration gives address of 1st temporary of running program
26	AT	CN12	ER1			
27	SA	CN3	0			MJ 0 above +3 formed to make 1st line a jump to 2nd line of next sentence
30	TP	RE	ER			
31	TV	A	ER			Line to generated routine output
32	RJ	RL2	RL			
33	SP	BK5	0			Getting constant call word for no. variables minus one
34	SS	CN1	0			
35	RJ	CS	CS1			TP Call Word of First temp of Variables Index Object Program formed and sent to output
36	TP	CN33	ER			
37	TU	A	ER			
40	TV	ER1	ER			
41	RJ	RL2	RL			
42	TP	CN41	ER			
43	RJ	RL2	RL			TU 1023 1011 to output
44	RA	ER	BP45			
45	ST	BQ16	ER			TU 1024 1005 to output
46	RJ	RL2	RL			
47	RA	ER	CN15			
50	RJ	RL2	RL			TU 1025 1010 to output

Setups for a sub- routine	51	RP	10003	IN53	} 00 30000 30000 to 3 temporaries
	52	TP	CN26	GK73	
	53	RA	ER1	CN1	
	54	SP	ER1	17	
	55	AT	ER1	ER1	
	56	RJ	GK72	GK	
	57	SP	BP45	0	
	60	RJ	CS	CS1	
	61	TP	CN32	ER	
	62	TU	CN27	ER	
	63	TV	Q	ER	
	64	RJ	RL2	RL	} Puts a succession of lines in output Gets constant call word for 0 1 0
	65	RA	ER	CN16	
	66	RJ	RL2	RL	
	67	RA	ER	BP45	
	70	RJ	RL2	RL	
	71	TP	CN34	ER	
	72	SP	GL7	17	
	73	TU	A	ER	
	74	TV	GL11	ER	
	75	RJ	RL2	RL	
	76	TP	RE	ER	
	77	TV	CN13	ER	} RA 1005 Call word of (0 1 0) to output RA 1010 Call word of (0 1 0) to output RA 1011 Call Word of (0 1 0) to output IJ Call word of Variables 1005 to output Index MJ 0 1000 to output 1 → ER1 0 1026 0 to output 0 (1026 + v) 0 to output 0 (1026 + 2 v) 0 to output Setup to get variables from string-out at BK6 Setup to store call words in output at certain location Setup to store call words at above locat. + v. v = no. variables Setup to store modulus indexes at 1st location + 2 v Variables index (v - 1) formed in ER1 Get variable from string-out input. Store variable as fixed const. in output. Get Dimension List call word of subscripted variable, store in output and Op File, & get modulus - 1 and store in output Modifying instr. to get next variable Index jump to get remaining variables
	100	RJ	RL2	RL	
	101	TP	CN1	ER1	
	102	SP	CN14	17	
	103	TP	A	ER	
	104	RJ	RL2	RL	
	105	SP	BK5	17	
	106	AT	ER	ER	
	107	RJ	RL2	RL	
	110	IJ	ER1	IN105	
	111	TU	CN31	IN122	
	112	TV	CN33	FD3	} 113 SP CN33 0 114 SA BK5 0 115 TV A IN123 116 SA BK5 0 117 TV A IN125 120 SP BK5 0 121 ST CN1 ER1 122 TP 30000 ER 123 TP ER 30000 124 RJ FD13 FD 125 TP A 30000 126 RA IN122 BP45 127 RA IN123 CN1 130 RA IN125 CN1 131 RA FD3 CN1 132 IJ ER1 IN122
	113	SP	CN33	0	
	114	SA	BK5	0	
	115	TV	A	IN123	
	116	SA	BK5	0	
	117	TV	A	IN125	
	120	SP	BK5	0	
	121	ST	CN1	ER1	
	122	TP	30000	ER	
	123	TP	ER	30000	
	124	RJ	FD13	FD	
	125	TP	A	30000	
	126	RA	IN122	BP45	
	127	RA	IN123	CN1	
	130	RA	IN125	CN1	
	131	RA	FD3	CN1	
	132	IJ	ER1	IN122	

133	MJ	0	FW44	Jump to termination of generation	
	CA	IN134			
	IA	FW			
0	TP	CN	GL2	Zeroize number of unmodifiable constants	
1	TU	CN31	FW25	Setup to scan string-out at BK6, beginning of X3 variable list	
2	TP	RE	ER	MJ 0 ____ to temporary ER	
3	ST	CN1	ER2	Variables index set up in ER2	
1 vari- able	4	ZJ	FW15	FW5	Is number of variables one?
	5	TP	CN6	GL1	17 = number of lines subject to address modification
6	TV	CN27	GL	25 = number of lines generated routine, including "10" lines & prelude	
7	TP	CN5	FL1	15 = Op File count of running program including temporaries	
10	TP	CN17	GK75	0 1013 0 to temporary as address holding call word of variable	
11	TP	CN23	ER1	0 1014 1014 to temporary as address of temporary in Object Program	
12	TV	CN34	FD3	Sets up FD routine to store call word in output	
	13	TV	CN30	ER	MJ 0 1016 formed in ER
2 vari- ables	14	MJ	0	FW24	
	15	TP	CN11	GL1	36 = no. of lines subject to address modification
16	TV	CN36	GL	42 = no. of lines gen. routine incl. 10 lines and prelude	
17	TP	CN7	FL1	27 = Op File count of Object Program lines, incl. temps	
20	TP	CN20	GK75	0 1024 0 to temp. as address holding call word of variable	
21	TP	CN24	ER1	0 1026 1026 to temp. as address of temporary of Object Program	
22	TV	CN35	FD3	Setup FD rtne. to store call word in output	
23	TV	CN37	ER	MJ 0 1030 formed in ER	
24	RJ	RL2	RL	Line to output	
25	TP	30000	ER	} Get constant call word for name of subscripted variable and store name in constant pool	
26	TP	ER	A		
27	RJ	CS	CS1		
30	TP	A	GK73	Store constant call word in temporary	
31	RJ	FD13	FD	} Get Dimension List call word of subscripted variable, store in output & Op. File, and get modulus minus one in A	
32	RJ	CS	CS1		
33	TP	A	GK74	Get call word of modulus - 1 and store in temporary	
34	RJ	GK72	GK	Put a succession of lines into output	
35	RA	GK75	BP45	} Altering instructions to get a remaining variable	
36	RA	FW25	BP45		
37	RA	FD3	CN1		
40	IJ	ER2	FW25	Jump back to get a remaining variable	
41	TP	RE	ER	} MJ 0 1000 to output	
42	TV	CN13	ER		
43	RJ	RL2	RL		

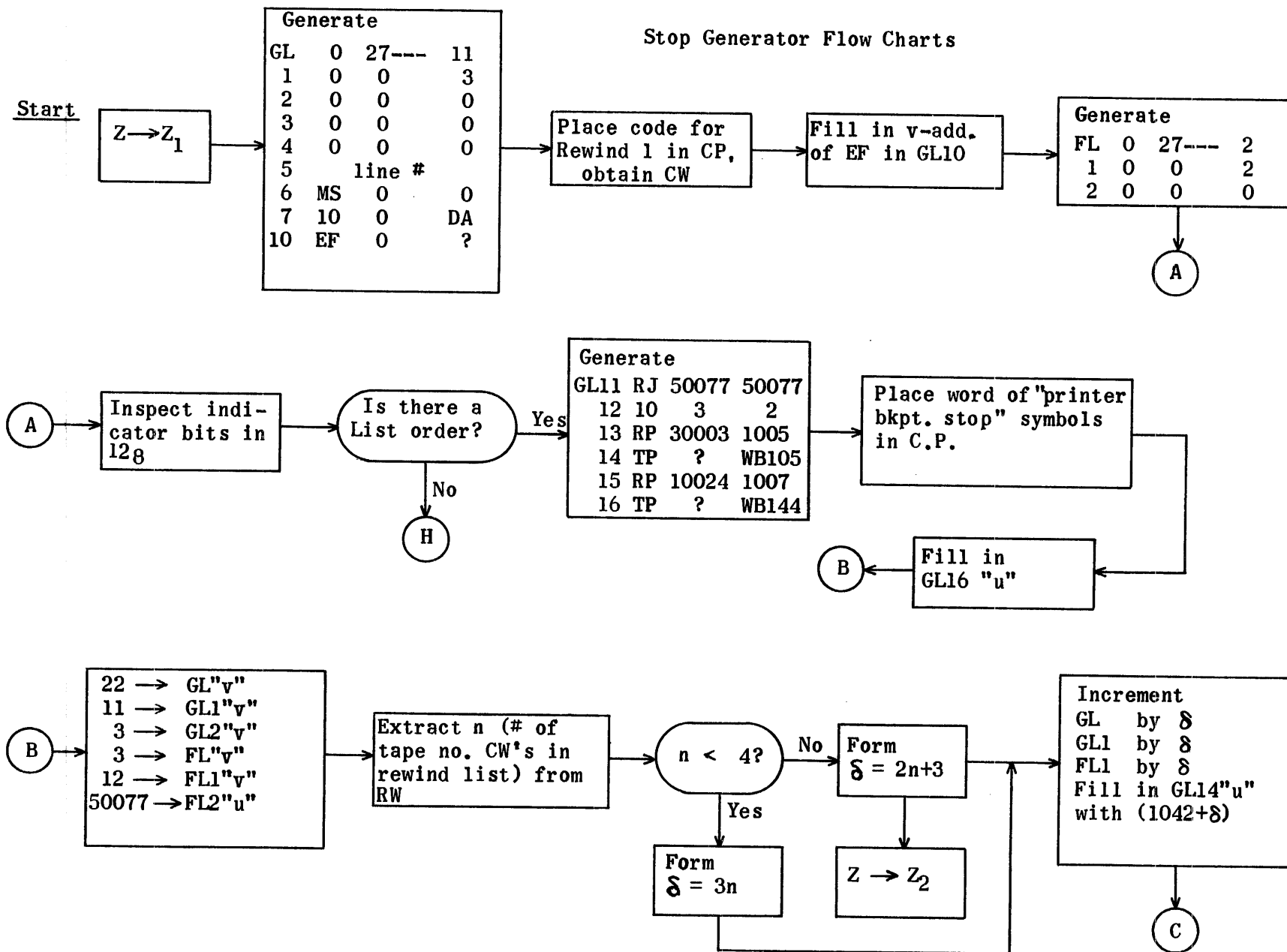
Termination	44	SP	BK4	0	}	Is there a "jump to" line number?
	45	ZJ	FW46	FW50		
	46	SP	ER3	0		Store call word of "jump to" line no. as
	47	RJ	FB3	FB		cross reference in Op. File
	50	SP	CN40	0		Call word referencing read routine in
	51	RJ	LS	LS1		Library put into Library List
	52	TP	CN42	OP1		Op File to storage and generated routine
	53	RJ	OP	OP2		to tape
	54	TP	VX153	Q		Indicator bit (0 2 0) put in WW (fixed
	55	QS	VX153	WW		location 12) to show a "READ" is present
	56	MJ	0	RE		
		CA	FW57			
		IA	GK			
	0	TP	CN33	ER	}	TP 30000 50100
	1	TV	CN21	ER		or
	2	TU	GK73	ER		TP Constant 50100
	3	RJ	RL2	RL		Call Word of Variable
	4	TP	CN25	ER	}	10 0 1 formed in ER
	5	TV	CN1	ER		Line to output without upping count in OK
	6	RJ	ST2	ST		of lines relative to 1000
	7	TP	CN21	ER	}	RJ 50100 50100 to output
	10	RJ	RL2	RL		
	11	TP	CN25	ER		10 0 2 to output
	12	TV	CN2	ER		
	13	RJ	ST2	ST	}	MJ 0 0 to ER
	14	TP	RE	ER		Is there a "jump to" line no. in sentence?
	15	TP	BK4	A		
	16	ZJ	GK76	GK17		
	17	SP	OK	0	}	MJ 0 100X - 1 to output
	20	SS	CN1	0		
	21	TV	A	ER		
	22	RJ	RL2	RL		
	23	MJ	0	GK32	}	MJ 0 Call Word of Line No. to output
	24	RJ	RL2	RL		
	25	TP	CN25	ER		10 0 1 to output
	26	TV	CN1	ER		
	27	RJ	ST2	ST	}	Adding one to prelude counts
	30	RA	GL	CN1		
	31	RA	GL1	CN1		
	32	TP	CN33	ER		
	33	TU	GK74	ER	}	TP 30000 or Temporary of Object to output
	34	TV	ER1	ER		Call Word of Modulus Index Program
	35	RJ	RL2	RL		
	36	TP	CN35	ER		
	37	TU	GK75	ER	}	TV { 30000 or 1013 or 1024 } 100X + 2 to output
	40	TV	OK	ER		More than 2 var.. 1 var.. or 2
	41	RA	ER	CN2		var. setups determine u of
	42	RJ	RL2	RL		generated coding

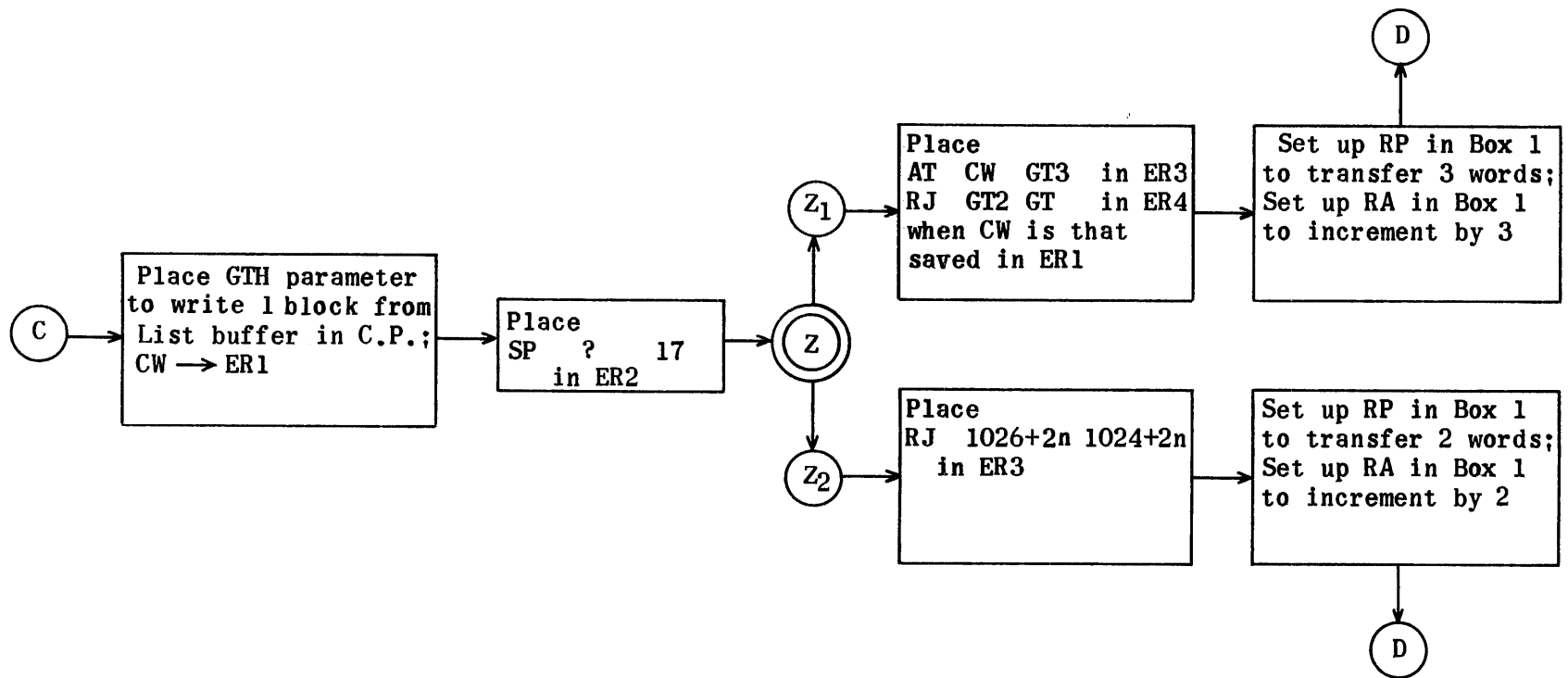
43	TP	CN21	ER	}	RJ 50100 50100 to output
44	RJ	RL2	RL		
45	TP	CN25	ER	}	10 0 3 to output
46	TV	CN3	ER		
47	RJ	ST2	ST	}	TP q 30000 to output
50	TP	CN33	ER		
51	TV	CN26	ER	}	Putting 1 into constant pool and getting its call word into Q _v
52	TU	CN30	ER		
53	RJ	RL2	RL	}	RA 100X-1 Call Word of 1 to output
54	SP	CN1	0		
55	RJ	CS	CS1	}	IJ Temporary of Object Program 100X-3 to output
56	TP	CN32	ER		
57	SP	OK	0	}	Subroutine exit
60	SS	CN1	17		
61	TU	A	ER	}	Temporaries used in generation
62	TV	Q	ER		
63	RJ	RL2	RL	}	Getting call word for line number MJ 0 Call word of to ER Line No.
64	TP	CN34	ER		
65	TU	ER1	ER	}	Storing call word of line no. in 2 temp.s.
66	SP	OK	0		
67	SS	CN3	0	}	Is call word of sentence > 22777? Put a word with U ₃₅ filled into Q to show Pseudo-Op. condition Call word of line no. to A _u Test if jump is legal
70	TV	A	ER		
71	RJ	RL2	RL	}	Increase excess-3 variable loading address by 2 to allow for 2 added "ten" lines Increase initial loading address of excess- three variables to allow for added "10" line following jump to line number
72	MJ	0	30000		
73	0	30000	30000	}	Does not include "10" lines.
74	0	30000	30000		
75	0	30000	30000	}	Loads line into output of generated routine, Ups count of 100X. Ordinal count of number of generated routine line relative to 1000.
76	RJ	LW	LW1		
77	TV	Q	ER	}	Generated 0 Routine 1 Loader Plus 2 100X Counter
100	TP	A	ER3		
101	TP	A	ER4	}	CA GK113
102	TP	CN22	A		
103	TJ	BK3	GK105	}	IA RL
104	TP	GK107	Q		
105	TP	ER4	A	}	RJ ST2 ST
106	RJ	KI	KI1		
107	MJ	0	GK24	}	RA OK CN1
110	RA	CN35	CN2		
111	RP	20002	IN15	}	MJ 0 30000
112	RA	CN33	CN1		
	CA	GK113		}	CA RL3
	IA	RL			

Generated	0	IA	ST		
Routine	1	TP	ER	30000	Loads line into output
Line		RA	ST	CN1	Alters loading line for next line of output
Loader	2	MJ	0	30000	
		CA	ST3		
Control Subroutine to get call word of a variable from Dimension List, store it in output and Op File, and get modulus - 1 in A _v					
		IA	FD		
	0	TP	ER	A	} Get call word of variable from Dimension List
	1	TU	6	TS	
	2	RJ	TS7	TS	
	3	TP	A	30000	} Put call word in output
	4	LA	A	17	
	5	RJ	FB3	FB	} Put call word in Op File 1
	6	RA	TS6	BP45	
	7	TU	A	FD10	} Extract modulus from Dimension List, subtract one from it, and put into A _v
	10	TP	30000	Q	
	11	QT	CS35	A	
	12	SS	BP45	71	
	13	MJ	0	30000	
		CA	FD14		
Get Call		IA	TS		
Word of	0	RP	0	BR1	} If variable is not in list, computer stops after a rewind of tapes and print-out: ALARM 1 COMILATION INCONSISTENCY 'POSSIBLE COMPUTER ERROR' RECOMPILE
a vari-	1	EJ	DL	TS2	
able in					
Dimen-					
sion List	2	SN	Q	17	} Computing position of variable in list and getting call word of it into A
	3	SA	TS	0	
	4	SA	TS1	0	
	5	TU	A	TS6	
	6	TP	30000	A	
	7	MJ	0	30000	
		CA	TS10		
Load		IA	FB		
Op	0	TP	A	30000	Storing line in Op File
File 1	1	RA	FB	CN1	Increase storing order
	2	RA	FL	CN1	Up count of number of lines
	3	MJ	0	30000	
		CA	FB4		
Constants		IA	CN		
and	0	0	0	0	
Dummy	1	0	0	1	
Commands	2	0	0	2	
Used to	3	0	0	3	
Make up	4	0	0	7	
Generated	5	0	0	15	
Routine	6	0	0	17	
	7	0	0	27	

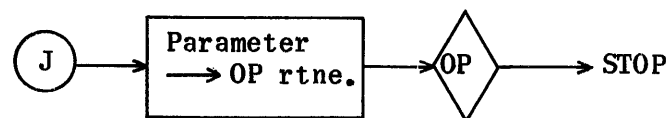
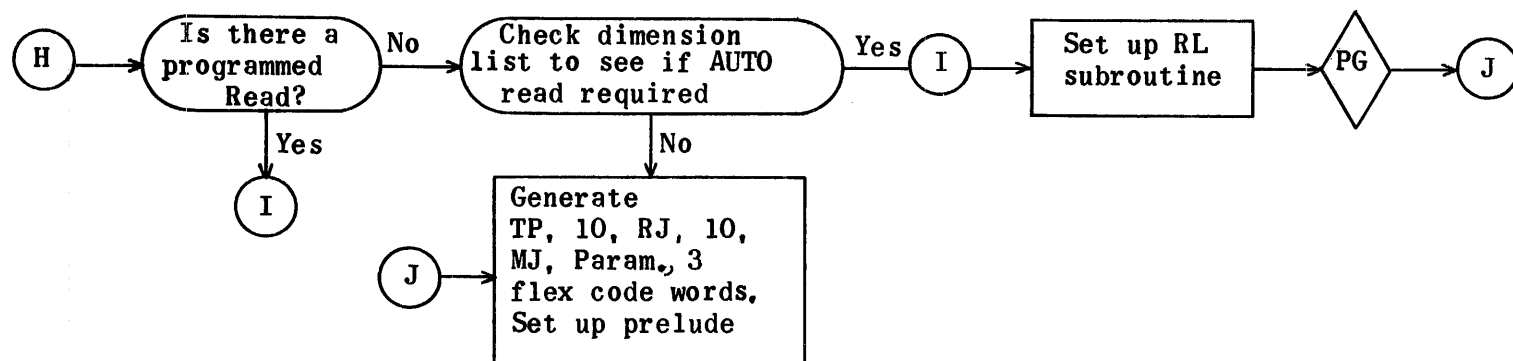
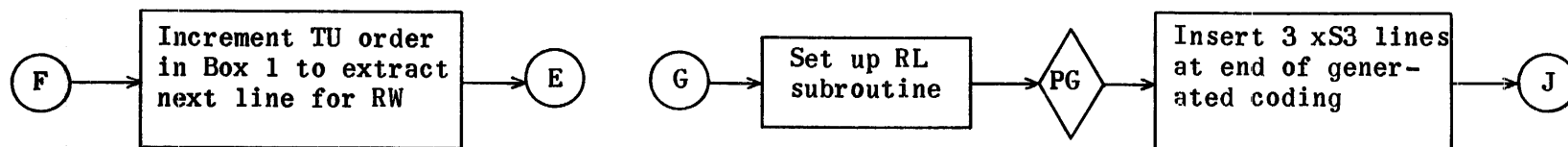
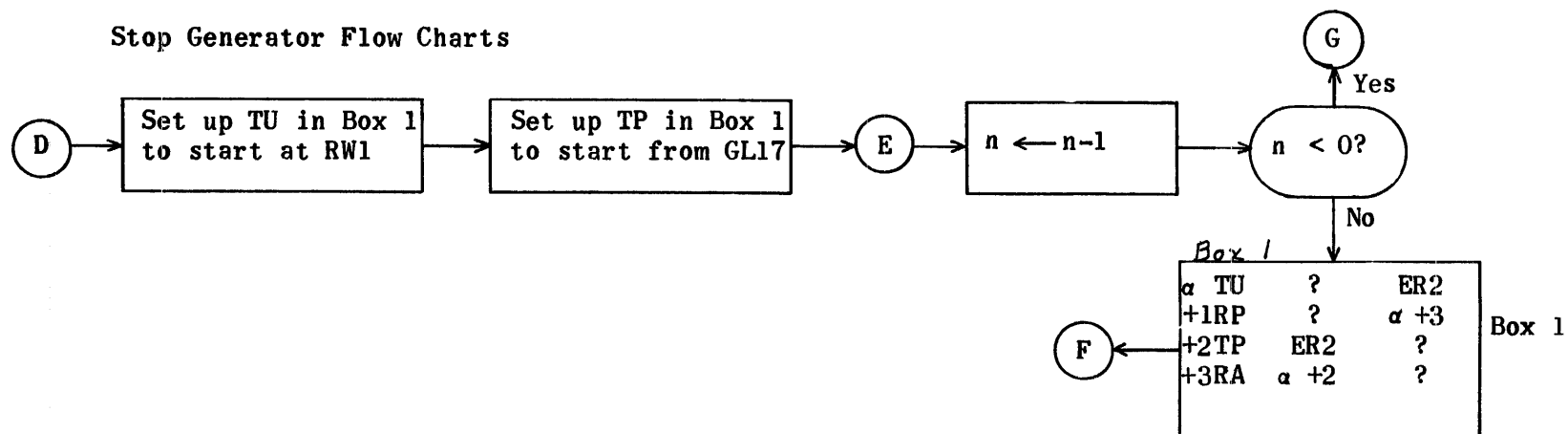
	10	0	0	31	
	11	0	0	34	
	12	0	0	767	
	13	0	0	1000	
	14	0	0	1026	Address holding call word of 1st variable for more-than-2-variable case
	15	0	1	3	
	16	0	3	0	
	17	0	1013	0	Address holding call word of variable for 1-variable case
	20	0	1024	0	Address holding call word of 1st variable for 2-variable case
	21	RJ	50100	50100	
	22	0	0	22777	
	23	0	1014	1014	Address of temporary of Object Program for 1-variable case
	24	0	1026	1026	Address of temporary of Object Program for 2-variable case
	25	10	0	0	
	26	0	30000	30000	
	27	0	1005	25	
	30	0	Q	1016	
	31	0	BK6	GL6	
	32	RA	DL	FL3	DL is initial location of Dimension List
When more	33	TP	LL	GL37	
than 2	34	IJ	0	GL24	GL24 is loading address of variable when only one
vars.,	35	TV	0	GL40	GL40 is initial loading addr. of variables when there are two
GL37 is	36	0	0	42	
initial	37	0	0	1030	
loading	40	0	50100	0	Call word of Read permanent library routine
address	41	TU	1023	1011	
of X3	42	0	GL	FL	
names		CA	CN43		

Stop Generator Flow Charts

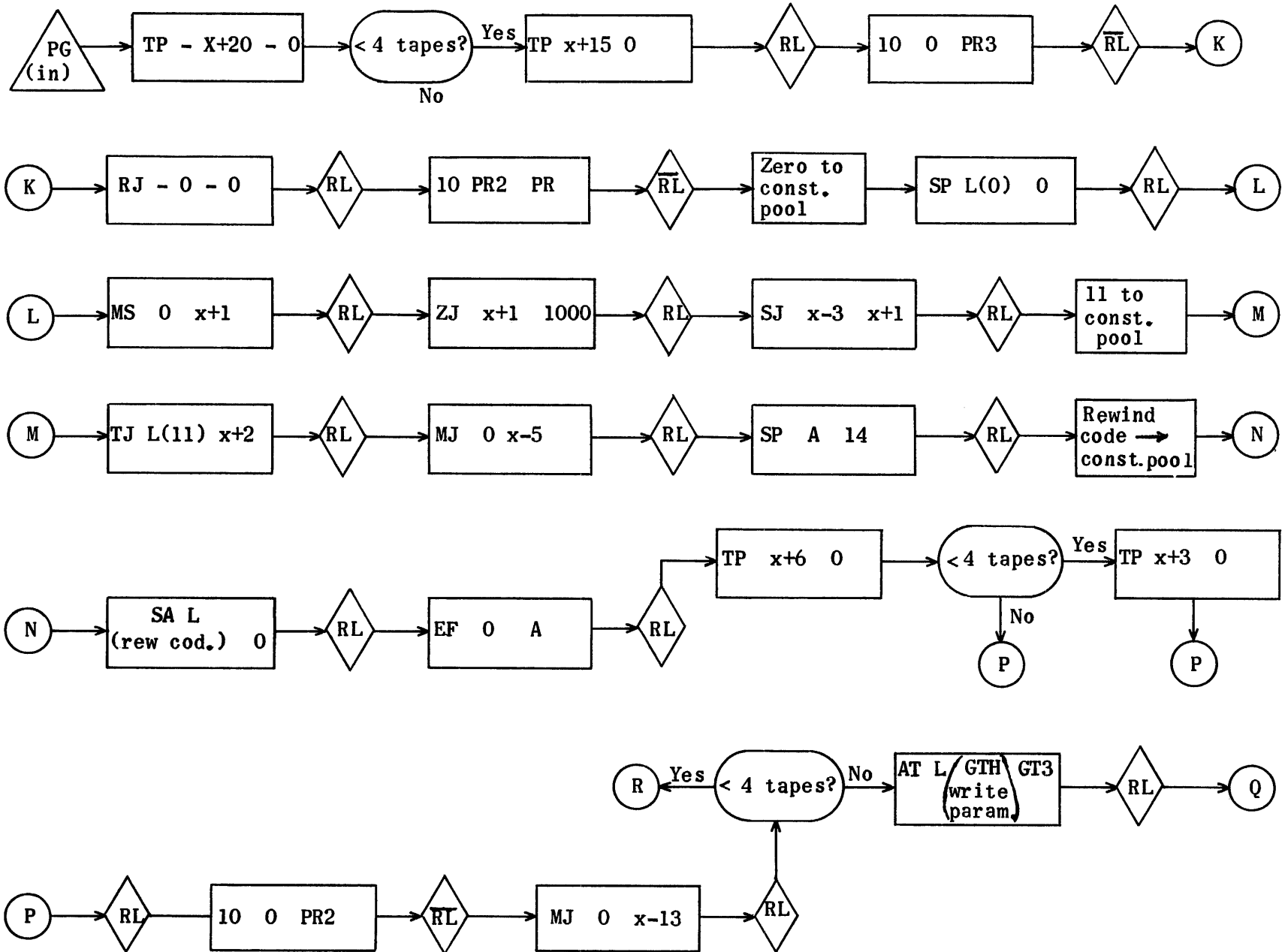




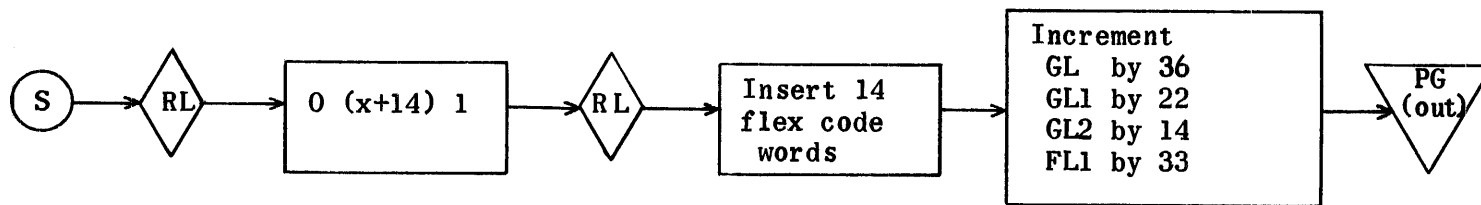
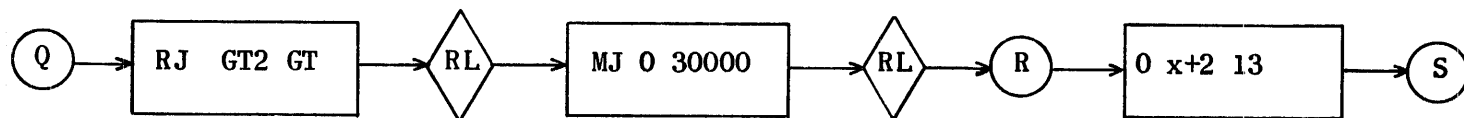
Stop Generator Flow Charts



Stop Generator Flow Charts



Stop Generator Flow Charts



In above routine X = address rel. 1000 of order currently being generated

STOP GENERATOR REGIONS

Generation Subroutines	{	RE	CS1211	Call word routine (= CW)
		RE	BR537	Machine error print-out routine
		RE	OP1047	Op control routine
		RE	DL40102	Dimension List
		RE	BK2242	Buffer input region
		RE	GL5360	Buffer output region for generated routine
		RE	RW50023	Rewind list of call words of tape no.'s referenced by read and list instructions
		RE	WW12	Fixed location of read, list indicator
		RE	ST2512	} Stop Generator
		RE	SG2513	
		RE	HT2513	
		RE	PG2702	
		RE	RL3017	
		RE	CN3024	} Temporaries
		RE	ER3137	
		RE	FL3144	Where Op File I is built
		RE	DA77300	Object Program service routine
		RE	PR77250	UNICODE print-out routine
		RE	GT210	Object program tape handler
		RE	WB71005	List buffer (170)
		RE	IW50077	Call word for LIST permanent library routine

STOP GENERATOR

	IA	ST		
	MJ	0	30000	Exit
	CA	ST1		
	IA	SG		
HT 0	TP	CN	HT166	Set indicator to zero
1	RP	10005	HT3	} Zeroize 5 lines of GL
2	TP	CN	GL	
3	SP	BK3	0	} Call word
4	LA	A	17	
5	TU	A	GL	
6	TV	CN4	GL	11 → GL "v"
7	TV	CN2	GL1	3 → GL1 "v"
10	TP	BK1	GL5	Line number
11	RP	30003	HT13	} MS, "10" line, EF
12	TP	CN32	GL6	
13	SP	CN27	0	Rew. 1 code
14	RJ	CS	CS1	
15	TV	Q	GL10	Fill in EF "v"
16	RP	10003	HT20	} Zeroize 3 lines for Op File item
17	TP	CN	FL	
20	TV	CN24	FL	2 → FL "v"
21	TV	CN24	FL1	2 → FL1 "v"
22	SP	BK3	0	} Call word
23	LA	A	17	
24	TU	A	FL	
25	TP	WW	Q	I/O indicator word
26	QT	CN17	A	} Is there a list?
27	ZJ	HT30	HT125	
30	RP	30006	HT32	} RJ, 10, RP, TP, RP, TP
31	TP	CN35	GL11	
32	SP	CN30	0	} Stop word to constant pool
33	RJ	CS	CS1	
34	TU	A	GL16	Fill in TP ₂ "u"
35	TV	CN7	GL	22 → GL "v"
36	TV	CN4	GL1	11 → GL1 "v"
37	TV	CN2	GL2	3 → GL2 "v"
	CA	SG40		
	IA	SG40		
40	TV	CN2	FL	3 → FL "v"
41	TV	CN5	FL1	12 → FL1 "v"
42	TU	CN35	FL2	IW CW → FL2 "u"
43	TP	CN12	Q	Mask for n from RW list
44	QT	RW	ER	Extract # tape call words (n)
45	TP	A	ER1	n → ER1
46	TJ	CN3	HT51	Test v < 4
47	TP	CN25	HT166	4 or more, set indicator to large no.
50	TP	CN2	ER1	3 → ER1

51	SP	A	1	2n
52	AT	ER1	ER1	Form inc. for prelude (3n or 2n + 3)
53	SA	CN15	17	Add 1042, form (1042 + 3n, 1045 + 2n)
54	TU	A	GL14	Fill in TP ₂ "u"
55	RA	GL	ER1	} Increment prelude counters
56	RA	GL1	ER1	
57	RA	FL1	ER1	
60	SP	CN31	0	
61	RJ	CS	CS1	General GTH parameter to write 1 block to C.P.
62	TP	A	ER1	Save it
63	TP	PG	ER2	SP—17→ER2
64	IJ	HT166	HT73	How many tapes?
65	TP	CN100	ER3	< 4; AT — GT3 → ER3
66	TU	ER1	ER3	AT L(GTHp) GT3 — ER3
67	TP	CN101	ER4	RJ GT2 GT → ER4
70	TU	CN37	HT106	Set up RP to write 3 words
71	TV	CN112	HT110	Set up RA to inc. by 3
72	MJ	0	HT101	Jump
73	SP	ER	1	4 or more tapes; 2n → A
74	AT	CN14	ER3	Add 1024 → 1024 + 2n in ER3 "v"
75	SA	CN24	17	Add 0 37 2 → RJ 1026 + 2n ? in A
76	AT	ER3	ER3	Complete order
77	TU	CN40	HT106	Set up RP to write 2 words
	CA	SG100		
	IA	SG100		
100	TV	CN105	HT110	Set up RA to inc. by 2
101	TU	CN42	HT105	Set up TU order to start at RW1
102	TV	CN103	HT107	Start at GL17
103	IJ	ER	HT105	Count down on n index
104	MJ	0	HT113	Exit
105	TU	[30000]	ER2	Complete SP order
106	RP	[0]	HT110	
107	TP	ER2	[30000]	
110	RA	HT107	[30000]	
111	RA	HT105	CN17	Increase by 1 "u"
112	MJ	0	HT103	
113	TV	HT107	RL1	Now HT 103 "v" holds next address in GL
114	RS	HT107	CN103	Subtract GL17
115	SA	CN41	0	Add 1007
116	TP	CN	RL4	Zeroize RL4
117	TV	A	RL4	And set it up (add. rel. 1000)
120	RJ	PG114	PG	Execute PG coding
121	TV	RL1	HT124	
122	RA	HT124	CN6	Increment by 14
123	RP	30003	HT163	} Insert XS3 - "END OF OUTPUT" then go to conclusion
124	TP	CN106	30000	
125	QT	CN20	A	
126	ZJ	HT151	HT127	
127	TP	CN	ER	No. prog. Read - investigate auto-read
130	TV	6	ER	Set up ER with #77 CW's
131	TP	CN16	ER1	77000 is 1st CW

132	TU	6	HT136	Set up RP
133	IJ	ER	HT135	Count down on index
134	MJ	0	HT155	Exit - no auto-read
135	SP	ER1	0	
136	RP	[0]	BR1	
137	EJ	DL	HT140	
	CA	SG140		
	IA	SG140		
140	SN	Q	17	
141	SA	HT136	0	
142	SA	HT137	0	
143	TU	A	HT144	
144	TP	[30000]	Q	
145	QT	CN25	A	Inspect X
146	ZJ	HT147	HT151	If zero, auto-read
147	RA	ER1	CN1	No - prepare to search for next CW
150	MJ	0	HT133	
151	TV	CN102	RL1	Read coding - prepare to gen. coding
152	TP	CN13	RL4	Gen from GL11, add entr → 1002
153	RJ	PG114	PG	Execute PG coding
154	MJ	0	HT163	Go to conclusion
155	RP	30011	HT157	NO READ, NO LIST
156	TP	CN55	GL11	
157	TV	CN7	GL	22 → GL "v"
160	TV	CN4	GL1	11 → GL1 "v"
161	TV	CN2	GL2	3 → GL2 "v"
162	TV	CN4	FL1	11 → FL1 "v"
163	TP	CN77	OP1	Conclusion - parameter to Op
164	RJ	OP	OP2	
165	MJ	0	ST	Exit
166	0	0	0	Indicator word
	CA	SG167		

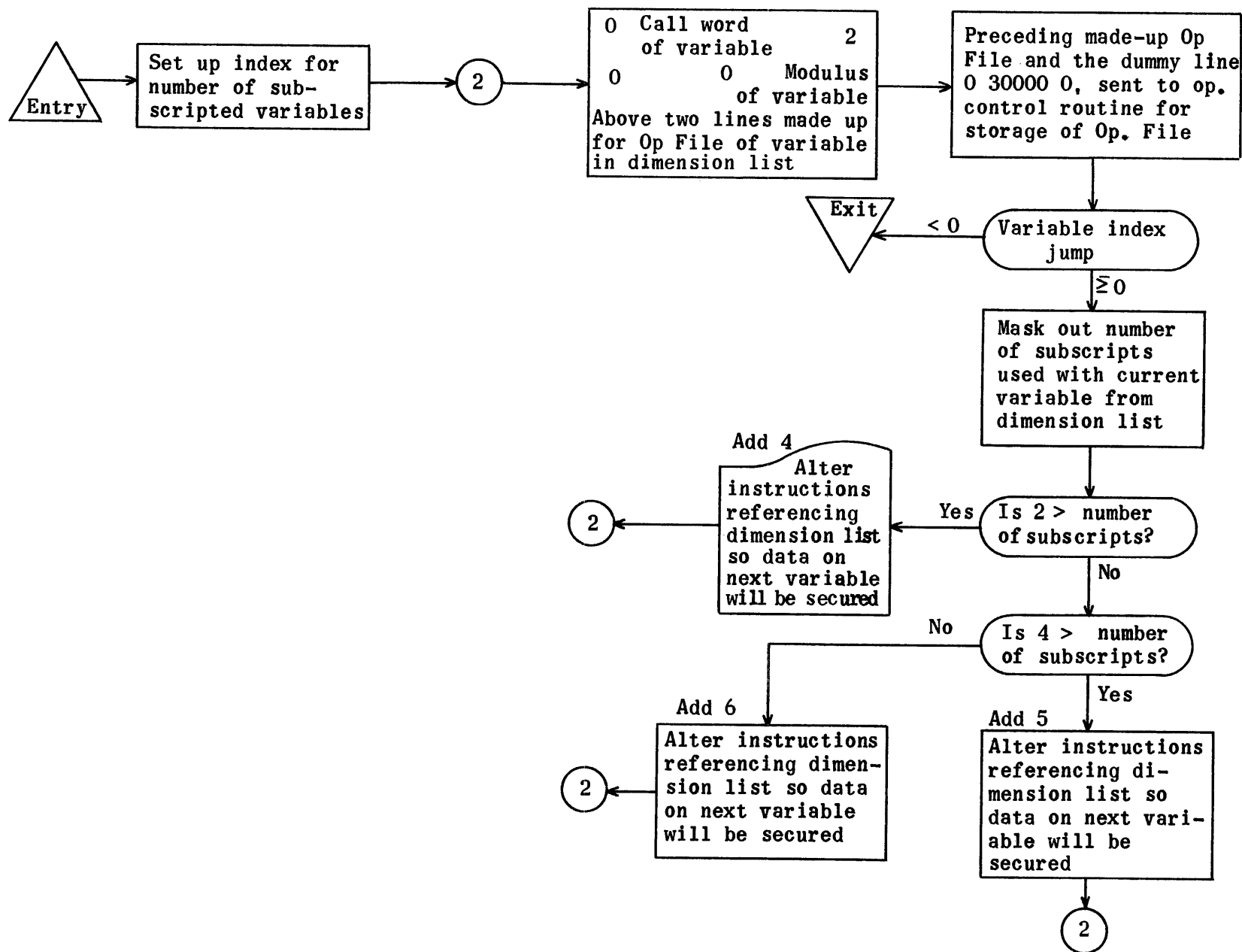
PG	0	IA	SG167		
	1	SP	RL4	17	Add. entr. → A "u"
	2	AT	CN43	ER	Add TP 20 0
	3	IJ	HT166	PG4	Or < 4 tapes?
	4	RS	ER	CN22	Yes, subtract 3 "u"
	5	RJ	RL3	RL	Store this running line
	6	TP	CN56	ER	10 0 PR3
	7	RJ	RL3	RL1	Store this 10 line
		TP	CN57	ER	RJ 0 0
	10	RJ	RL3	RL	Store this running line
	11	TP	CN60	ER	10 PR2 PR
	12	RJ	RL3	RL1	to output
	13	SP	CN	0	Clear A
	14	RJ	CS	CS1	→ Constant Pool
	15	TP	HT13	ER	
	16	TU	A	ER	SP 1(0) 0 to output
	17	RJ	RL3	RL	
	20	SP	RL4	0	Add entr. → A "v"
	21	AT	CN44	ER	Add MS 0 1
	22	RJ	RL3	RL	Store this running line
	23	SP	RL4	17	Add entr. → A "u"
	24	AT	CN45	ER	Add ZJ 1 1000
	25	RJ	RL3	RL	Store it
	26	SP	RL4	17	Entr. → A "u"
	27	SA	RL4	0	and A "v"
	30	AT	CN46	ER	Add 45 77775 00001
	31	RJ	RL3	RL	Store this (SJ line)
	32	SP	RL4	0	Entr. → A "v"
	33	AT	CN47	ER	Add TJ — 2
	34	SP	CN111	0	1110 → A
	35	RJ	CS	CS1	→ Constant Pool
	36	TU	A	ER	TJ L(11) X + 2
	37	RJ	RL3	RL	Store this
		CA	SG227		
	40	IA	SG227		
	41	SP	RL4	0	Entr. → A "v"
	42	AT	CN50	ER	Add 44 77777 77773 → MJ line
	43	RJ	RL3	RL	Store it
	44	TP	CN51	ER	SP A 14
	45	RJ	RL3	RL	Store it
	46	SP	CN26	0	Rewind code → A
	47	RJ	CS	CS1	→ C.P. (Constant Pool)
		TP	HT115	ER	SA — 0
	50	TU	A	ER	
	51	RJ	RL3	RL	Store it
	52	TP	CN34	ER	EF 0 A
	53	RJ	RL3	RL	Store it
	54	SP	RL4	17	Add entr. → A "u"

55	AT	CN52	ER	Add TP 6 0
56	IJ	HT166	PG60	or less than 4?
57	RS	ER	CN22	Yes - subtract 3 "u"
60	RJ	RL3	RL	Store it
61	TP	CN56	ER	10 0 PR3
62	RJ	RL3	RL1	Store this 10 line
63	SP	RL4	0	Add entr. → A "v"
64	AT	CN53	ER	Add 44 77777 77765 (⇒ MJ)
65	RJ	RL3	RL	Store it
66	IJ	HT166	PG70	How many tapes?
67	MJ	0	PG77	< 4 - jump
70	TP	CN100	ER	AT — GT3
71	TU	ER1	ER	AT CW(Write param.) GT3
72	RJ	RL3	RL	Store it
73	TP	CN101	ER	RJ GT2 GT
74	RJ	RL3	RL	Store it
75	TP	CN54	ER	MJ 0 30000
76	RJ	RL3	RL	Store it
77	SP	RL4	17	Add entr. → A "u"
	CA	SG267		
	IA	SG267		
100	AT	CN21	ER	Add 0 2 13
101	RJ	RL3	RL	Store it
102	SP	RL4	17	Entr. → A "u"
103	AT	CN23	ER	Add 0 14 1
104	RJ	RL3	RL	Store it
105	TV	RL1	PG107	Extract address in GL list
106	RP	30014	PG110	} Insert Flex codes
107	TP	CN63	[30000]	
110	RA	GL	CN11	Inc. GL "v" by 368
111	RA	GL1	CN7	Inc. GL1 "v" by 228
112	RA	GL2	CN6	Inc. GL2 "v" by 148
113	RA	FL1	CN10	Inc. FL1 "v" by 338
114	MJ	0	[30000]	Exit
	CA	SG304		
	IA	SG304		
RL 0	RA	RL4	CN1	
1	TP	ER	[30000]	
2	RA	RL1	CN1	
3	MJ	0	[30000]	
4	[0	0	0]	
	CA	SG311		

	IA	SG311		
CN 0	0	0	0	
1	0	0	1	
2	0	0	3	
3	0	0	4	
4	0	0	11	
5	0	0	12	
6	0	0	14	
7	0	0	22	
10	0	0	33	
11	0	0	36	
12	0	0	77	
13	0	0	1002	
14	0	0	1024	
15	0	0	1042	
16	0	0	77000	
17	0	1	0	
20	0	2	0	
21	0	2	13	
22	0	3	0	
23	0	14	1	
24	0	37	2	RJ in "u"
25	7	0	0	
26	2	200	0	General rewind
27	2	200	10000	Rewind #1
30	0	61616	16161	Breakpoint stop word
31	74	20100	1005	Parameter to write 1 block from list buffer
32	MS	0	0	
33	10	0	DA	
34	EF	0	A	
35	RJ	IW	IW	
36	10	3	2	
37	RP	30003	1005	
	CA	SG351		
	IA	SG351		
40	TP	30002	WB105	
41	RP	10024	1007	
42	TP	RW1	WB144	
43	TP	20	0	
44	MS	0	1	
45	ZJ	1	1000	
46	45	77775	00001	
47	TJ	0	2	
50	44	77777	77773	
51	SP	A	14	
52	TP	6	0	
53	44	77777	77765	
54	MJ	0	30000	
55	TP	1005	0	
56	10	0	PR3	
57	RJ	0	0	

60	10	PR2	PR	
61	MJ	0	1000	
62	0	1006	3	
63	45	47200	62204	cr ↑ ENDΔ
64	03	26041	23406	OFΔ RUN
65	45	45000	00000	cr cr _____
66	01	03041	22031	TO Δ REW
67	14	06220	41454	IND Δ I/
70	03	04013	01520	0 Δ TAPE
71	24	04242	00104	SΔ SETΔ
72	24	20121	70304	SERVOΔ
73	06	03041	40604	NO Δ INΔ
74	30	04051	40104	A Δ HITΔ
75	24	01301	20157	START ↓
76	45	22140	10103	cr ditto
77	0	GL	FL	
	CA	SG411		
	IA	SG411		
100	AT	0	GT3	
101	RJ	GT2	GT	
102	0	0	GL11	
103	0	0	GL17	
104	0	0	2	
105	0	0	CN104	L(2)
106	30	50270	15131	ENDΔOF
107	01	51676	65267	ΔOUTPU
110	66	22010	10101	T. ⌘⌘
111	0	0	13	
112	0	0	CN2	L(3)
	CA	SG424		

Flow Chart for Dimension Generator



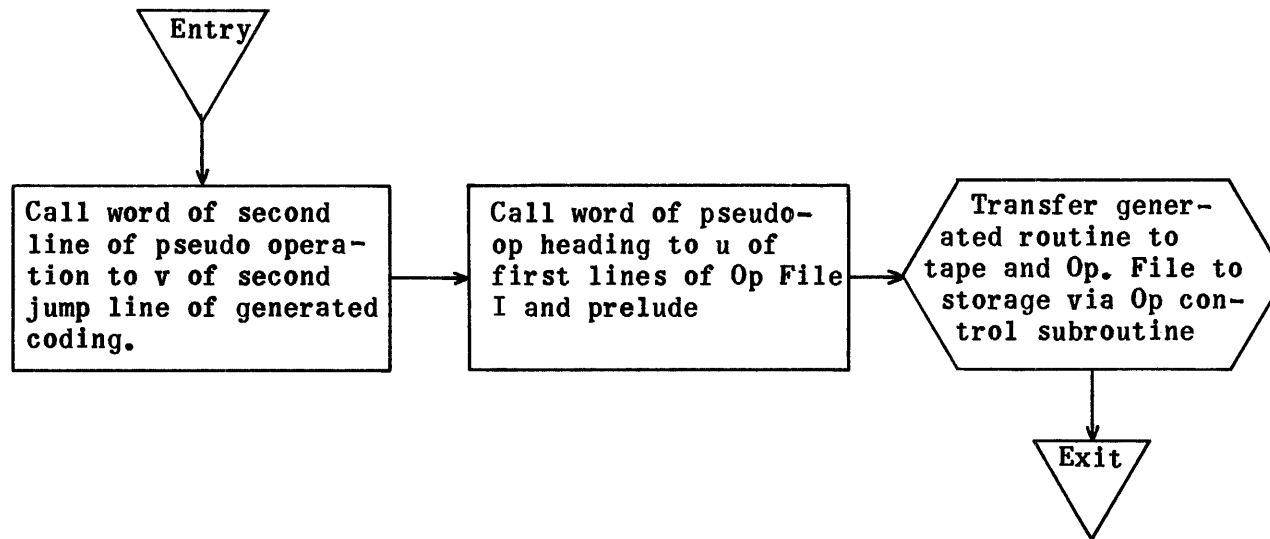
DIMENSION GENERATOR

RE CQ2512
RE C02547
RE OP1047
RE DL40102

	IA	CQ		
0	MJ	0	30000	Exit
1	TP	6	Q	} Mask # of arrays to establish index
2	QT	C011	A	
3	TP	A	C010	
4	IJ	C010	CQ5	} Set index
5	SP	DL2	17	
6	TU	A	C013	
7	SP	DL3	71	} Call word → Op File 1
10	TV	A	C014	
11	TP	C012	OP1	
12	RJ	OP	OP2	} Modulus → Op File 1
13	IJ	C010	CQ15	
14	MJ	0	CQ	
15	TP	DL3	Q	} Exit
16	QT	C011	A	
17	TJ	C02	CQ31	
20	TJ	C04	CQ25	} Mask # of subscripts
21	RA	CQ5	C07	
22	RA	CQ7	C07	
23	RA	CQ15	C07	} 2 > (A)? No-add 4 (it must be 0,1) 4 > (A)?
24	MJ	0	CQ5	
25	RA	CQ5	C06	
26	RA	CQ7	C06	} Add 6
27	RA	CQ15	C06	
30	MJ	0	CQ5	
31	RA	CQ5	C05	} Add 5
32	RA	CQ7	C05	
33	RA	CQ15	C05	
34	MJ	0	CQ5	} Add 4
	CA	CQ35		
	IA	CO		

0	0	0	0	} No. of subscripts
1	0	0	1	
2	0	0	2	
3	0	0	3	
4	0	0	4	} Modifiers
5	0	4	0	
6	0	5	0	
7	0	6	0	
10	0	0	[0]	} Index
11	0	0	77777	
12	0	C015	C013	
13	0	0	2	} Mask
14	0	0	0	
15	0	30000	0	
	CA	C016		} Parameters

Pseudo-Operation Heading (Generation) - Flow Chart



PSEUDO-OP HEADING-GENERATION

	RE	BK2242		
	RE	OP1047		
	RE	PS2512		
	IA	PS		
0	MJ	0	30000	Exit
1	TV	BK4	PS21	Call word of second line of subprogram to v of output line
2	LQ	BK3	Q17	} Call word of Pseudo-Op heading to u of 1st line of Op File 1.
3	TU	Q	PS10	
4	TU	Q	PS12	Same to u of 1st line of prelude
5	TP	PS23	OP1	} Transfer generated routine to tape and Op File to storage
6	RJ	OP	OP2	
7	MJ	0	PS	Exit
10	0	[0]	2	} Op File 1
11	0	0	2	
12	0	[0]	11	} Prelude of generated routine
13	0	0	3	
14	0	0	0	
15	0	0	0	
16	0	0	0	
17	0	0	0	
20	MJ	0	30000	} Generated routine plus "10" line
21	MJ	0	[0]	
22	10	0	1	
23	0	PS12	PS10	Parameter word for transfer of generator to tape and Op File to storage
	CA	PS24		

The line number in input buffer line bkl is inserted in PS17 by generation control prior to operation of this routine in the core.

End of Tape--Generation

Control of the End of Tape Generator is in the last part of the Control Generation subroutine. First the Op File is transferred to the proper storage area. Then RJ KB KB1 does the following things:

It closes out the Op File block, sending it to tape, and adds on an "End of Entry" block. In the "End of Entry" block following the first two lines, the contents of locations 5-17 are reproduced. Prior to this, the value of 14 is computed and a warning print-out given if the number of blocks scheduled to be put on tape 5 will exceed 2500.

Two blocks of Z's are added to the tape holding the generation subroutines following an "End of Entry" block.

List I of the library routines referenced is sent to tape 5 with an initial block having in its second line the contents of counter 5. An "End of Entry" block is added at the end.

Next, RJ UG UG1 puts the Dimension List on tape 5 with opening and closing blocks. RJ IG IG1 puts the Constant List on tape 5. RJ BU BU1 references a BX subroutine to build an excess-three symbol list of single-valued variables. Then the BU subroutine sends this list to tape 5 with the appropriate beginning and ending block.

RJ EG EG1 rewinds the string-out input tape and the generation-routine tapes and moves backward on tape 5 until the reader head is positioned just before the entry blocks of the Op File I data.

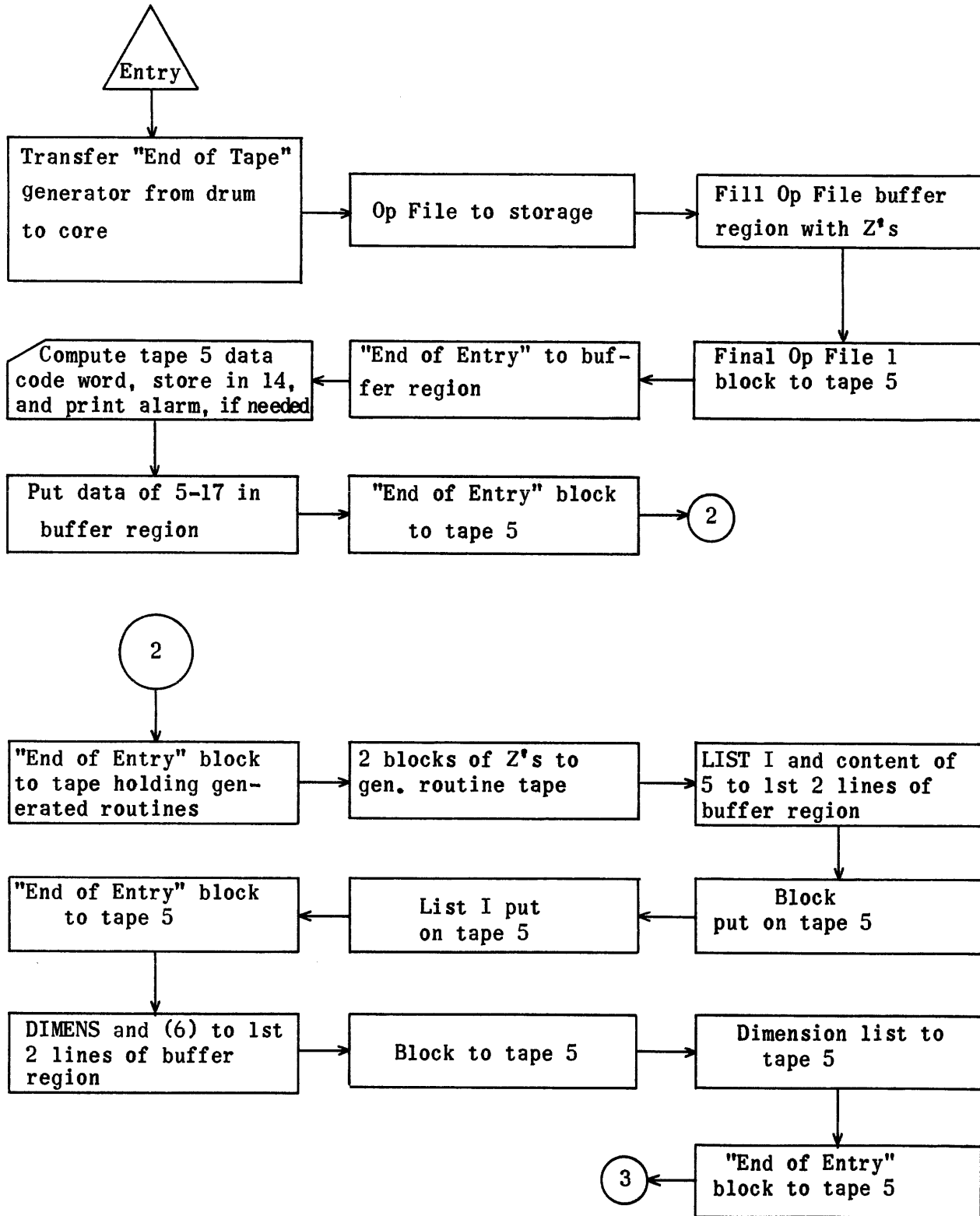
RJ BE BE1 gives the termination print-out for generation and provides the option of interruption or continuation of the UNICODE program.

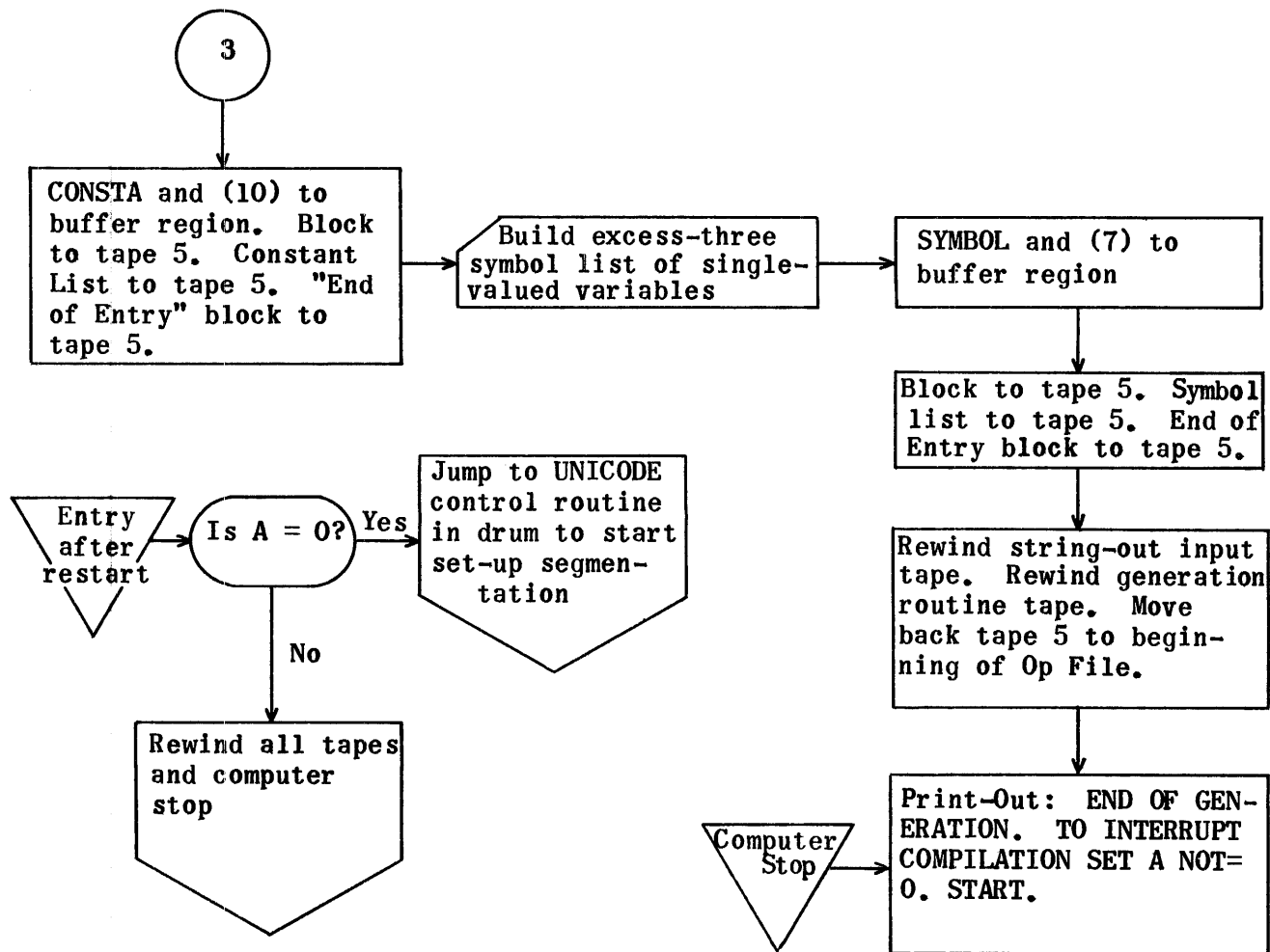
Subsidiary subroutine TE is another name for KB during the "End of Tape" operation. TF is a list of labels and a mask. EW is a subroutine which adds Z's to a title block and transfers it to tape and contains a portion which puts List I on tape. IW is a subroutine which takes any length list and adds it to tape, computing the number of blocks and adding at the end an "End of Entry" block.

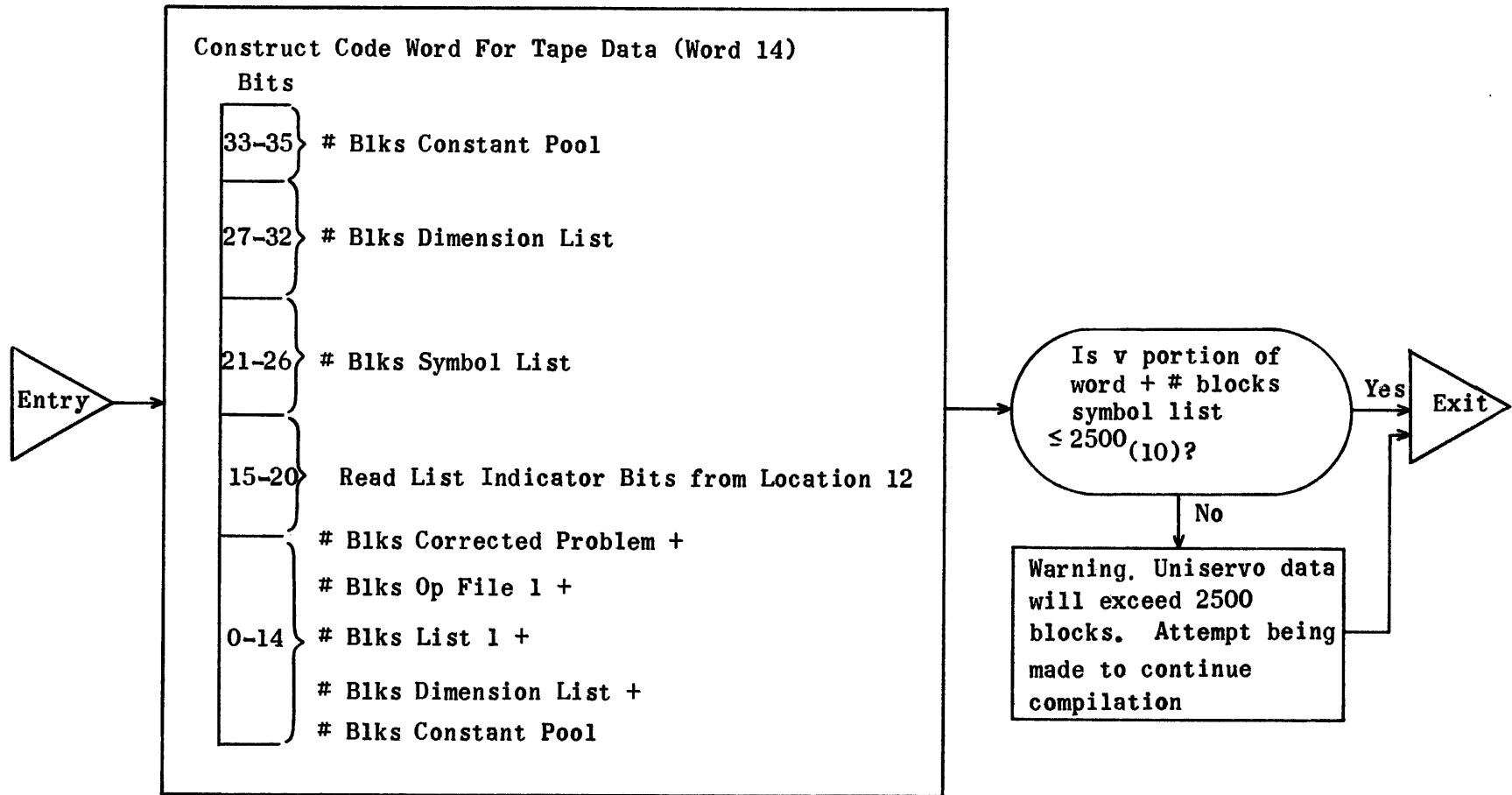
RJ BV BV1 is an instruction within TE which in turn references regions BM, BN, and BO and computes the value of 14.

Explanation of FS, a region of 7 temporaries used during operation of IW, is given in annotated form following the print-out of the Reco coding.

End of Tape - Generation - Flow Chart







End of Tape--Generation
Regions

Generation Subroutine regions are also needed for assembly of this tape.

RE	TE2512
RE	TF2565
RE	EW2573
RE	IW2610
RE	IG2675
RE	UG2713
RE	EG2732
RE	BE2750
RE	BU2774
RE	BX3012
RE	BV3052
RE	BM3104
RE	BN3123
RE	BO3154
RE	FS3214
RE	BZ3223
RE	BW4223
RE	BY2777

End of Tape — Generation

(See Control Generation Routine (CG77-111) for control portion of these End-of-Tape Subroutines.)

0	IA	TE		
0	MJ	0	30000	Exit
1	TP	ES	A	Entry
2	ZJ	TE3	TE16	Last block of Op File 1 filled with Z's and transferred to tape 5
3	AT	RC1	A	
4	TV	A	TE12	
5	TP	GP5	A	
6	SS	ES	17	
7	AT	GP4	A	
10	TU	A	TE11	
11	RP	10000	TE13	
12	TP	GP2	30000	
13	TP	RC	GT3	
14	RJ	GT2	GT	To set up code tape data word 14 and test length of data to go on Uniservo 5
15	RA	ES5	GP10	
16	RJ	BV	BV1	End of entry block of Op File 1 with data on 5-17 goes to Uniservo 5
17	TP	TF	NP	
20	TP	TF1	NP1	
21	RP	30013	TE23	
22	TP	5	NP2	
23	RP	10153	TE25	
24	TP	GP2	NP15	
25	RJ	GT2	GT	
26	RA	ES5	GP10	
27	TP	TF	GN	End of entry block of subroutines for Uniservo 4 or 7
30	TP	TF1	GN1	
31	RP	10166	TE33	
32	TP	GP2	GN2	
33	TP	RC4	GT3	
34	RJ	GT2	GT	
35	RA	ES6	GP10	
36	TP	GP2	GN	
37	TP	GP2	GN1	
40	RJ	GT2	GT	2 blocks of Z's following subroutines on tape 4 or 7
41	RA	ES6	GP10	
42	RJ	GT2	GT	
43	RA	ES6	GP10	
44	TP	ES5	FS5	Setting up counter of blocks from beginning of Op File on in tape 5
45	TP	TF2	NP	Start of transfer of List I to tape 5
46	TP	TF5	Q	
47	QT	5	NP1	
50	ZJ	TE51	TE	
51	RJ	EW4	EW	
52	MJ	0	EW5	
	CA	TE53		

	IA	TF		
0	30	50270	15131	ENDΔOF
1	01	30506	65473	ΔENTRY
2	46	34656	60104	LISTΔI
3	26	51506	56624	CONSTA
4	27	34473	05065	DIMENS
5	0	7777	0	Mask
	CA	TF6		

	IA	EW		
0	RP	10166	EW2	} Subroutine to fill a block containing 2 title lines with lines of Z's and transfer to tape 5
1	TP	GP2	NP2	
2	TP	RC	GT3	
3	RJ	GT2	GT	
4	MJ	0	30000	
5	TP	EW14	IW64	} To finish transferring List I of library routine call words to tape 5
6	TP	GP3	Q	
7	QT	5	FS2	
10	RJ	IW	IW1	} No. of blocks of library list goes to FS4
11	TP	FS	FS4	
12	RA	FS5	FS4	} Adding in no. blocks of Library List I to accumulative block counter
13	MJ	0	TE	
14	0	0	LN	
	CA	EW15		

Subroutine Used to Transfer a List to Tape

	IA	IW		
0	MJ	0	30000	Exit
1	TP	GP10	FS	Entry. Starting count of number of blocks with 1
2	TP	IW64	A	} Is list located in drum or core?
3	TJ	GP6	IW20	
4	TP	FS2	A	} Is 170>no. lines left in list? If so, go to partial-block portion
5	TJ	GP5	IW35	
6	SP	IW64	17	} Setting up transfer line for block transfer
7	TU	A	IW11	
10	RP	30170	IW12	} Transferring block lines to buffer in core
11	TP	30000	NP	
12	TP	RC	GT3	} Writing 1 block on tape
13	RJ	GT2	GT	
14	RA	FS	GP10	Counting number of blocks
15	RS	FS2	GP5	Subtracting no. lines transferred from FS2
16	RA	IW64	GP5	Increasing address to next line to be transferred
17	MJ	0	IW4	Jump back to continue transferring data, block by block, to tape
20	TP	FS2	A	} Is 170>no. lines left in list? If so, go to partial block portion
21	TJ	GP5	IW35	

List in Drum {

List in Core	22	DV	GP5	FS1	No. of whole blocks to be transferred → FS1 Remainder → FS2
	23	TP	A	FS2	
	24	TP	FS1	A	Setting up parameter and transferring no. whole blocks of list to tape in one reference to tape handler
	25	SS	GP10	25	
	26	AT	RC	GT3	
	27	TV	IW64	GT3	
	30	RJ	GT2	GT	
	31	RA	FS	FS1	Upping block count
	32	MP	FS1	GP5	Updating address for next line to be transferred
	33	AT	IW64	IW64	
Partial Block to Tape 5 after Filling With Zs	34	TP	FS2	A	No. lines left in list → A
	35	ZJ	IW36	IW57	
	36	SA	GP6	17	Setting up repeat lines to transfer re- mainder of data to NP
	37	TU	A	IW42	
	40	SP	IW64	17	
	41	TU	A	IW43	Transferring data to NP
	42	RP	30000	IW44	
	43	TP	30000	NP	
	44	TP	GP5	A	Calculating number of Z lines needed and setting up repeat lines accordingly
	45	SS	FS2	17	
	46	TU	A	IW52	
	47	RA	IW52	GP4	
	50	TV	FS2	IW53	
	51	RA	IW53	RC1	Transferring Z lines to buffer region to fill up block
	52	RP	10000	IW54	
	53	TP	GP2	30000	
	54	TP	RC	GT3	Transferring final block to tape
	55	RJ	GT2	GT	
End of Entry Block	56	RA	FS	GP10	Upping block count
	57	TP	TF	NP	END Δ OF } Δ ENTRY } to terminating block
	60	TP	TF1	NP1	
	61	RJ	EW4	EW	Block to tape
	62	RA	FS	GP10	Upping block count of list
	63	MJ	0	IW	
	64	0	0	0	Input line holds address of 1st line of list in v
	CA	IW65			

Routine to Put List of Constants on Tape 5

	IA	IG		
0	MJ	0	30000	Exit
1	TP	B03	A	Entry If B03 is zero, the list of constants is nonexistent
2	ZJ	IG3	IG	
3	TP	TF3	NP	CONSTA to title line of block
4	TP	10	NP1	Counter to 2nd line of 1st blockette of block
5	RJ	EW4	EW	Sends 1st block to tape (title block)
6	TP	IG15	IW64	Sends input line to subroutine
7	TP	GP3	Q	Masks out no. of words of constants to FS2
10	QT	10	FS2	

11	RJ	IW	IW1	Sends list to tape via subroutine
12	TP	FS	FS3	Count of blocks to FS3
13	RA	FS5	FS3	Accumulative block counter updated
14	MJ	0	IG	
15	0	0	CL	
	CA	IG16		

Routine to Put Dimension List on Tape 5

IA	UG			
MJ	0	30000		Exit
TP	BO4	A	}	If BO4 is zero, the dimension list is nonexistent
ZJ	UG3	UG		
TP	TF4	NP	}	DIMENS } to title block
tp	6	NP1		
RJ	EW4	EW		Title block to tape
TP	UG16	IW64		Input line to subroutine IW
TP	TF5	Q	}	Length of List masked out to FS2
QT	6	FS2		
LQ	FS2	25		
RJ	IW	IW1		Send list to tape
TP	FS	FS6		Count of blocks to FS6
RA	FS5	FS6		Updating accumulative block counter
MJ	0	UG		
0	0	DL		
CA	UG17			

Routine to Rewind Tapes 3 and 4 (or Tapes 6 and 7) and Move To Front of Op File 1 on Tape 5

	IA	EG		
	0	MJ	0	30000
	1	SP	EG13	0
	2	AT	TN	GT3
	3	RJ	GT2	GT
	4	SP	EG14	0
	5	AT	TN	GT3
	6	RJ	GT2	GT
Moves to	7	SP	FS5	25
Front of	10	AT	EG15	GT3
Op File 1	11	RJ	GT2	GT
on Tape	12	MJ	0	EG
on Servo	13	10	3	0
5	14	10	4	0
	15	40	5	0
	CA	EG16		

Exit

} Rewind Uniservo 3 or 6

} Rewind Uniservo 4 or 7

Accumulative block counter (FS5) to right position to add to parameter for generalized tape handler

Parameter to rewind 3

Parameter to rewind 4

Parameter to move backward 5

End of Generation Print-Out Routine

	IA	BE		
0	MJ	0	30000	Exit
1	TP	BE10	UP3	} Print-Out: END OF GENERATION. TO INTERRUPT COMPILATION SET A NOT = 0. START.
2	RJ	UP2	UP	
3	SP	BE7	0	
4	MS	0	BE5	Clears A
5	ZJ	BE6	BE	Stop with Next Instruction in PAK
6	MJ	0	BQ6	} If A ≠ 0, jump to rewind tapes and stop
7	0	0	0	
10	0	BE11	13	
11	30	50270	15131	Parameter for Print-Out
12	01	32305	03054	ENDΔOF
13	24	66345	15022	ΔGENER
14	01	01665	10134	ATION.
15	50	66305	45467	ΔΔTO Δ I
16	52	66012	65147	NTERRU
17	52	34462	46634	PTΔCOM
20	51	50016	53066	PILATI
21	01	24015	05166	ONΔSET
22	01	76032	20101	ΔAΔNOT
23	65	66245	46622	Δ = 0.ΔΔ
	CA	BE24		START.

Routine to Write Excess-Three Symbol List on Tape (Uniservo 5)

	IA	BU		
0	MJ	0	30000	Exit
1	TP	7	A	} Is 7 is zero, there are no single-valued variables
2	ZJ	BU3	BU	
3	TP	A	FS2	
4	TP	BU14	NP	Length of List to FS2
5	TP	7	NP1	SYMBOL } To title block
6	RJ	EW4	EW	Counter 7 }
7	RJ	BX	BX1	Title block to tape
10	TP	BU15	IW64	Builds excess-three symbol list
11	RJ	IW	IW1	Input line to subroutine IW
12	RA	FS5	FS	List to tape with end of entry block
13	MJ	0	BU	Cumulative count of blocks updated
14	65	73472	55146	SYMBOL
15	0	0	BZ	Initial address of excess-three symbol list
	CA	BU16		

Routine to Build Excess-Three Symbol List

	IA	BX		
0	MJ	0	30000	Exit
1	TP	7	BX1	} One less than number of words in final list → working storage BX1. If number of words is zero, bypass the routine
2	IJ	BX1	BX4	
3	MJ	0	BX	
4	TP	BX30	Q	} Adjust to start first tnf after the Dimension List
5	QT	6	A	
6	AT	BX11	BX11	
7	TN	BX31	Q	Mask → Q for search
10	RP	BY30001	BX12	} One segment → core
11	TP	CB1	BW	
12	RA	BX11	BX36	
13	TU	BX37	BX15	} Set up to get next block and start this one
14	TP	BX33	BX2	
15	QT	30000	A	
16	EJ	BX34	BX22	Mask all but last 4 digits → A
17	RA	BX15	BX32	If = 0 0 60000 → BX22
20	IJ	BX2	BX15	Advance address
21	MJ	0	BX10	Recycle to continue search
22	TU	BX15	BX24	Recycle to load a second segment
23	RS	BX24	BX32	} Assemble the final list
24	TP	30000	BZ	
25	RA	BX24	BX35	
26	IJ	BX1	BX17	} Check for completion
27	MJ	0	BX	
30	0	7777	0	
31	0	0	7777	Exit
32	0	1	0	Four-digit extractor
33	0	0	BY	Negative of mask used for search
34	0	0	60000	u advance
35	0	0	1	Length of segment - 1
36	0	BY1	0	Used for EJ test
37	0	BW	0	v advance
	CA	BX40		Length of segment
				Segment location

Code Word for Tape Data Subroutines

	IA	BV		
0	MJ	0	30000	Exit
1	RP	10005	BV3	} Clear temporaries
2	TP	B0	B03	
3	TP	B0	14	
4	TP	10	Q	} Constant pool
5	QT	B01	A	
6	ZJ	BV7	BV16	
7	DV	B02	B03	(A) = 0 = No constant pool
10	ZJ	BV13	BV11	(A) ≠ 0 - Divide and store q
11	RA	B03	B011	(A) = 0 = No remainder
12	MJ	0	BV14	Quotient + label + end

	13	RA	B03	B014	(A) $\neq$ 0 = remainder - add 1 + label + end
	14	SP	B03	41	
Constant	15	TP	A	14	$\rightarrow$ 1436,35,34
Pool	16	TP	6	Q	
	17	QT	B012	A	} Dimension List
	20	ZJ	BV21	BM	(A) = 0 = No Dimension List
	21	LQ	A	25	(A) $\neq$ 0 shift to v position for divide
	22	DV	B02	[B04]	Divide and store Q
	23	ZJ	BV26	BV24	(A) = 0 = No remainder
	24	RA	B04	B011	Add quotient + label + end
	25	MJ	0	BV27	
	26	RA	B04	B014	(A) $\neq$ 0 = remainder add quotient + 1 + label + end
	27	SP	B04	33	
	30	AT	14	14	
	31	MJ	0	BM	
		CA	BV32		
		IA	BM		+ Symbol List + Read List indicators
	0	TP	7	Q	
	1	QT	B01	A	} Symbol List
	2	ZJ	BM3	BM12	(A) = 0 = No symbol List. Go to next step
	3	DV	B02	B05	(A) $\neq$ 0 divide by 170
	4	ZJ	BM7	BM5	(A) = 0 = no remainder
	5	RA	B05	B011	Add quotient + label + end
	6	MJ	0	BM10	add quotient + 1
	7	RA	B05	B014	(A) $\neq$ 0 = remainder + label + end
	10	SP	B05	25	Shift
	11	AT	14	14	Symbol List blks $\rightarrow$ 14
	12	TP	12	Q	
	13	QT	B013	A	} Read List indicator
	14	ZJ	BM15	BN	(A) = 0 = No Read List indicators
	15	AT	14	14	(A) $\neq$ 0 add $\rightarrow$ 14
	16	MJ	0	BN	
		CA	BM17		
		IA	BN	14 _v	
	0	TP	13	Q	
	1	QT	B01	A	} Number blocks corrected problem
	2	AT	ES5	[B06]	Corr. prob. + (ES5) $\rightarrow$ temp.
	3	RA	B06	B010	" " " + 1 $\rightarrow$ temp. } Op File 1
	4	TP	5	Q	
	5	QT	B01	A	} List 1
	6	ZJ	BN7	BN15	(A) = 0 = No List 1
	7	DV	B02	[B07]	No. blks List 1 $\rightarrow$ temp.
	10	ZJ	BN14	BN11	
	11	RA	B07	B011	(A) = 0 = No remainder Add quotient + label + end
	12	RA	B07	B06	Subtotal
	13	MJ	0	BN16	
	14	RA	B07	B014	(A) $\neq$ 0 = remainder add quotient + 1 + label + end
	15	RA	B07	B06	Subtotal
	16	RA	B07	B03	Add constant pool

17	RA	B07	B04	Add Dimension List
20	SP	B07	0	
21	AT	14	14	Totals thru constant pool → 14 _v
22	TP	14	Q	
23	QT	B01	A	
24	AT	B05	A	(A) = 14 _v + Symbol List
25	TJ	B015	BV	(u) > (A) its o.k. exit
26	TP	B016	UP3	(u) not > (A) print warning: WARNING.
27	RJ	UP2	UP	UNISERVO 5 DATA WILL EXCEED 2500 BLOCKS.
				ATTEMPT BEING MADE TO CONTINUE COMPILATION.
30	MJ	0	BV	Exit
	CA	BN31		
	IA	BO		
0	0	0	0	
1	0	0	77777	Mask
2	0	0	170	Words ₈ 1 blk
3	0	0	0	Constant pool
4	0	0	0	Dimension List
5	0	0	0	Symbol List
6	0	0	0	Corr. prob. + (ES5) + 1
7	0	0	0	Temp.
10	0	0	1	
11	0	0	2	Label + end
12	0	7777	0	Mask
13	0	3	0	Mask
14	0	0	3	Remainder + label + end
15	0	0	4705	2500 ₁₀ + 1
16	00	B017	21	Parameter for print
17	71	24545	03450	WARNIN
20	32	22010	16750	G,ΔΔUN
21	34	65305	47051	ISERVO
22	01	10012	72466	Δ5ΔDAT
23	24	01713	44646	AΔWILL
24	01	30722	63030	ΔEXCEE
25	27	01051	00303	DΔ2500
26	01	25465	12645	ΔBLOCK
27	65	22010	12466	S,ΔΔAT
30	66	30475	26601	TEMPTΔ
31	25	30345	03201	BEINGΔ
32	47	24273	00166	MADEΔT
33	51	01010	10101	0ΔΔΔΔΔ
34	01	01265	15066	ΔΔCONT
35	34	50673	00126	INUEΔC
36	51	47523	44624	OMPILA
37	66	34515	02277	TION.
	CA	B040		

Temporary Region FS

- 0 Number of blocks of any List transferred to tape
- 1 Number of full blocks of a List transferred
- 2 Number of lines in partial block or first-number of lines in List
- 3 Number of blocks of constants on tape
- 4 Number of blocks of Library List on tape
- 5 Number of blocks accumulative back to beginning of Op File 1
- 6 Number of blocks of Dimension List