DOCUMENT RESUME

ED 054 705	48	FL 002 630

AUTHOR Price, James D.
TITLE A Computerized Phrase-Structure Grammar (Modern Hebrew): Part IV, An Algorithm for Analyzing Hebrew Sentences. Final Report.
INSTITUTION Franklin Inst. Research Labs., Philadelphia, Pa.
SPONS AGENCY Office of Education (DHEW), Washington, D.C. Bureau of Research.
BUREAU NO BR-9-7722
PUB DATE Jun 71
CONTRACT OEC-0-9-097722-4411
NOTE 232p.

EDRS PRICE MF-$0.65 HC-$9.87
DESCRIPTORS *Algorithms; Computational Linguistics; *Computer Programs; Deep Structure; Flow Charts; Grammar; *Hebrew; Language Patterns; Language Research; Language Universals; Logic; Phrase Structure; Semitic Languages; Sentence Structure; *Structural Analysis; Syntax; *Transformation Generative Grammar; Transformation Theory (Language)

ABSTRACT

The final part of a four-part report of research on the development of a computerized, phrase-structure grammar of modern Hebrew describes the computerized algorithm for analyzing the sentences generated based on a complex-constituent-phrase structure grammar. The first section here discusses a structural model for modern Hebrew; the second provides a detailed description of the procedure for analyzing sentences; the third gives a detailed description of the computer program for the procedure; the last section describes the tests and the verifications of the algorithm. Appendix A lists the grammar rules of Hebrew syntax for the analysis of sentences. Appendix B gives the source-language listing of the computer program and subprograms used. Appendix C provides a sample output. Appendix D gives examples of the exhaustive snytactic analysis of the computer. For related reports see FL 002 627, FL 002 628, and FL 002 629. (VM)

04/115

Title VI, NDEA, Section 602 PA-48 Bureau#9-7722

FL

ED054705

FINAL REPORT

Project No. 097722

Contract No. OEC-0-9-097722-4411

Franklin Institute Report No. F-C2585-4

A COMPUTERIZED PHRASE-STRUCTURE GRAMMAR (MODERN HEBREW)

PART IV

James D. Price

Franklin Institute Research Laboratories

20th and Race Street

Philadelphia, Penn. 19103

June 1971

U.S. DEPARTMENT OF

HEALTH, EDUCATION, AND WELFARE

Office of Education

Institute of International Studies

FL002630

FINAL REPORT

Project No. 097722
Contract No. OEC-0-9-097722-4411
Franklin Institute Report No. F-C2585-4

A COMPUTERIZED PHRASE-STRUCTURE GRAMMAR (MODERN HEBREW) PART IV

AN ALGORITHM FOR ANALYZING HEBREW SENTENCES

James D. Price
Franklin Institute Research Laboratories
Philadelphia, Pennsylvania 19103

June 1971

The research reported herein was performed pursuant to a contract with the Office of Education, U. S. Department of Health, Education, Welfare.

U. S. DEPARTMENT OF
HEALTH, EDUCATION, AND WELFARE
Office of Education
Institute of International Studies

ABSTRACT

This is the fourth part of a four-part report of research for the development of a Computerized Phrase-Structure Grammar of Modern Hebrew. This part describes a computerized algorithm for analyzing sentences in modern Hebrew which is based on a generalized complex-constituent phrase-structure grammar (defined in Part I) as it was applied to the syntax of modern Hebrew (described in Part II).

A computer program of the algorithm is described which includes, for the main program and each subprogram, (1) a flow chart, (2) a written description of its operation, and (3) a source language listing in FORTRAN IV.

The algorithm was made operational in a UNIVAC 1108 computer and used to systematically test the grammar of modern Hebrew syntax by analyzing sentences in the language. A total of 26 sentences were analyzed in the process of which 57 of the 17 grammar rules were tested. The tests demonstrate the capability of the algorithm for analyzing sentences. Due to limitations on the predictive logic of the algorithm, some difficulties were experienced in the analysis of complex sentences. However, the algorithm proved to be a valuable tool for testing, validating, and, in numerous cases, correcting the grammar rules.

Three aspects of the algorithm need further attention: (1) several computations must be freed from dependence on the "content" of the source grammar, (2) the input data preparation should be simplified, and (3) the predictive logic should be extended to a greater depth. With these modifications the algorithm can be used to generate sentences in other Semitic languages whenever grammars become available, and to aid in the development of such grammars.

TABLE OF CONTENTS

TABLE OF CONTENTS (CONT.)

LIST OF FIGURES

LIST OF TABLES

7

PART IV

AN ALGORITHM FOR ANALYZING HEBREW SENTENCES

This part of the report describes a compterized algorithm for analyzing Hebrew sentences, and algorithm essentially complete in that it defines all the procedures and associated computer programs for analyzing sentences in modern Hebrew, but incomplete in that improvements can be made in the grammar upon which it operates. Because the algorithm is dependent only on the "form" of the grammar and not on its "content"[1], it may be used for training research workers in the field of computational linguistics without being limited to the Hebrew language.

The algorithm consists of: (1) a structural model of the language, (2) a procedure for using the model to analyza sentences in the language, and (3) a program for performing the procedure by means of a computer.

The first section discusses the structural model of modern Hebrew. The second section provides a detailed description of the procedure for analyzing sentences. The third section provides a detailed description of the computer program of the procedure. The last section describes the tests and verifications of the algorithm.

4.1 A Structural Model of Hebrew Sentences

The structural model of modern Hebrew sentences consists of a complex-constituent phrase-structure grammar of modern Hebrew the generalized characteristics of which are defined in Part I of this report and the specific details for which are defined in Part II. Those details of the structural model required to explain the algorithm are presented in the appropriate places throughout the text. Reference should be made to Parts I and II for further information.

For the purposes of the algorithm, the structural model is viewed as the following:

(1) a set of mapping functions

(2) a set of variables

---

[1]A few exceptions are noted throughout the text; these will be corrected in a subsequent revision.

4.1.1 The Mapping Functions

The mapping functions enable the user to define the specific details of the structural model that are peculiar to the given language (in this case Hebrew). The use of mapping functions makes the algorithm independent of the "content" of grammar it processes by enabling the user to define the "content" for a given language.

The mapping functions consist of the following:

(1) a table of the transliteration

(2) a table of symbols

(3) a table of grammar rules

(4) a table of restraints

(5) a table of symbol names

(6) a table of analysis predicates

(7) an index of (6)

(8) a table of feature values

4.1.1.1 The Transliteration

The table of transliteration provides a list of alphabetic and numeric characters of the given language. These data are used by the algorithm to transform the input characters to numbers for use in computation and to transform to resultant output numbers to their corresponding characters. Table 4-1 is a listing of the transliteration used for Hebrew (see Table 2-1, Part II, for Hebrew equivalents).

4.1.1.2 The Symbols

The table of symbols provides a list of the symbols used in the grammar of the given language. The list is used by the algorithm to transform the input symbol names of the grammar rules to numbers for use in computation and to transform the resultant output numbers to their corresponding symbol names for use in tree diagrams and output listings. Table 4-2 is a listing the symbols used for Hebrew.[2]

---

[2]See Table 2-2 and 2-3 of Part II for further description of the symbols.

Table 4-1

ALPHA NUMERIC TRANSLITERATION

L	TRANSL(L)	IA	L	TRANSL(L)	IA
1	0	0	26	O	25
2	1	1	27	P	26
3	2	2	28	&	27
4	3	3	29	Q	28
5	4	4	30	R	29
6	5	5	31	S	30
7	6	6	32	T	31
8	7	7	33	(	32
9	8	8	34	)	33
10	9	9	35	,	34
11	A	10	36	:	35
12	B	11	37	?	36
13	D	12	38	.	37
14	G	13	39	!	38
15	H	14	40	–	39
16	W	15	41	(space)	0
17	Z	16	42	F	41
18	X	17	43	I	42
19	@	18	44	V	43
20	Y	19	45	*	44
21	K	20	46	#	45
22	L	21	47	Δ	46
23	M	22	48	'	47
24	N	23	49	?	48
25	C	24	50	?	49

Table 4-2

LIST OF SYMBOLS

SYMBOL No.	SYMBOL Name	SYMBOL No.	SYMBOL Name	SYMBOL No.	SYMBOL Name
1	Z	21	B_{p}	41	R_{sp}
2	V_{qo}	22	N_{pb}	42	N_{sp}
3	A_{pa}	23	N_{pa}	43	N_{op}
4	A_{p}	24	N_{pc}	44	N_{ip}
5	A_{s}	25	N_{ap}	45	N_{px}
6	N_{a}	26	N_{p}	46	V_{ma}
7	S_{qo}	27	D_{pa}	47	V_{mb}
8	N_{s}	28	D_{pb}	48	V_{mc}
9	R_{d}	29	D_{pc}	49	V_{md}
10	R_{o}	30	D_{p}	50	V_{m}
11	B_{aa}	31	E_{a}	51	V_{mr}
12	B_{ab}	32	V_{b}	52	V_{mi}
13	B_{ac}	33	V_{bb}	53	V_{p}
14	B_{ad}	34	V_{c}	54	V_{rb}
15	B_{ae}	35	V_{aa}	55	V_{ri}
16	B_{af}	36	V_{a}	56	N_{v}
17	B_{ba}	37	B_{c}	57	N_{w}
18	B_{bb}	38	X_{p}	58	E_{pb}
19	B_{a}	39	N_{o}	59	E_{pa}
20	B_{bc}	40	D_{pd}	60	E_{p}

Table 4-2 (Continued)

LIST OF SYMBOLS

SYMBOL No.	SYMBOL Name	SYMBOL No.	SYMBOL Name
61	S_{aa}	81	E
62	S_{ab}	82	G
63	S_{ac}	83	H
64	S_a	84	I
65	S_{ro}	85	J
66	S_{ri}	86	L
67	R_g	87	N
68	K_n	88	O
69	K_c	89	P
70	K_k	90	Q
71	K_i	91	R
72	K_d	92	T
73	S_d	93	U
74	S	94	V
75	S_i	95	W
76	[illegible]$_c$	96	Y
77	A	97	F
78	B		
79	C		
80	D		

4.1.1.3 The Rules

The table of grammar rules provides a list of the rules of the complex-constituent phrase-structure grammar of the given language. Appendix A of this part of the report contains a listing of the grammar rules used for Hebrew.[3] The rules of the analysis grammar differ from those of the synthesis grammar in two respects:

(1) the left side of the analysis rules is the right side of the synthesis rules and *visa versa.*

(2) synthesis rules that have an optimal symbol as the first element of the right side must have two corresponding analysis rules, one with the first symbol as mandatory, and one without the first symbol.

4.1.1.4 The Restraints

The table of restraints provides a list of limitations to be used by the rules of the grammar. The grammar specifies a certain horizontal row of the table to limit the value of a given symbol subscript. The restraints are interpreted as follows:

(1) The first number in the row specifies the number of restraint values in the row.

(2) If the numbers in the specified row are positive, the value of the given subscript must be one of the numbers in the row.

(3) If the numbers in the specified row are negative, the value of the given subscript must not be one of the numbers in the row.

Table 4-3 is a list of the restraints used in the grammar of Hebrew.

---

[3]Reference should be made to Section 2.2.3 of Part II of this report for a detailed description of the rules.

Table 4-3

LIST OF RESTRAINTS

Row No.	No. of Items	Item			
		1	2	3	4
1	1	-1			
2	1	-2			
3	1	-3			
4	2	0	1		
5	2	1	2		
6	2	-1	-2		
7	2	1	3		
8	2	2	5		
9	2	4	5		
10	2	6	7		
11	3	1	2	3	
12	3	4	5	6	
13	4	1	2	3	4
14	4	-1	-2	-3	-4

4.1.1.5 The Symbol Names

The table of symbol names provides a list of the names assigned to the symbols of the grammar. The list is used by the algorithm for constructing analysis statements about the sentences being analyzed. The analysis statements are of the general form

$$N_s + N_a$$

where N_s stands for a symbol name and its associated derivational history and N_a stands for an analysis predicate (see next section). For example, the algorithm may construct the statement "<u>The basic past-nom. adjective phrase (APA)</u> expresses the superlative degree," the underlined part of which is the symbol name (N_s) the remainder is the analysis predicate. Table 4-4 is a listing of the symbol names used for Hebrew.

4.1.1.6 The Analysis Predicates and Index

The table of analysis predicates provides a list of the syntactic functions of all the catagories of the symbols used by the grammar. The list is used by the algorithm to serve as the predicate of analysis statements (see example in the previous section). Table 4-5 is a listing of the analysis predicates for Hebrew. Table 4-6 is an index of the analysis predicates that maps the correspondence between a given class of a symbol and the associated predicate. For example, Symbol 3 class 3 is associated with predicate 5.

4.1.1.7 The Feature Values

The table of feature values provides a list of the various semantic values that a given linguistic feature of the symbols may assume. The table maps the correspondence of the numerical values of the symbol subscripts with the semantic value of the associated linguistic feature. Table 4-7 is the table of feature values for Hebrew. For example, the table specifies for subscript n=2 that this corresponds to the semantic value *dual* for the linguistic feature *number*. The algorithm uses these data to exhaustively define the surface structure elements in analysis statements.

4.1.2 The Variables

The set of variables consist of 29 subscripts on the set of symbols of the grammar that enable the user to define the sentence to be analyzed by the algorithm. The algorithm uses three types of variables:

(a) Fixed variables -- those with values fixed by the rules of the grammar.

(b) Dependent variables -- those the value of which are computed by the grammar.

(c) Independent variables -- those the value of which the user defines in the process of describing the sentence being analyzed.

Table 4-4

LIST OF SYMBOL NAMES

Symbol No.	Symbol Name
1	PREPOSITIONAL PRON. PHRASE(Z)
2	OBJECTIVE-INTEROG. VERB PHRASE(VQO)
3	BASIC POST-NOM.ADJECTIVE PHRASE(APA)
4	POST-NOMINAL ADJECTIVE PHRASE(AP)
5	ADJECTIVAL POSSESS. PHRASE(AS)
6	BASIC NOUN PHRASE(NA)
7	OBJECTIVE INTEROGATIVE PHRASE(SQO)
8	POSSESS.-PRON. NOUN PHRASE(NS)
9	BASIC DEMONSTRATIVE PRON. PHRASE(RD)
10	DIRECT-OBJECT PRON.PHRASE(RO)
11	UNITS NUMBER PHRASE(BAA)
12	TENS NUMBER PHRASE(BAB)
13	TEENS NUMBER PHRASE(BAC)
14	MULTI-TENS NUMBER PHRASE(BAD)
15	HUNDREDS NUMBER PHRASE(BAE)
16	THOUSANDS NUMBER PHRASE(BAF)
17	1-TO-99 NUMBER PHRASE(BBA)
18	100-TO-999 NUMBER PHRASE(BBB)
19	1-TO-999 NUMBER PHRASE(BA)
20	1000-TO-9999 NUMBER PHRASE(BBC)
21	1-TO-9999 NUMBER PHRASE(BP)
22	SIMPLE NOUN PHRASE(NPB)
23	REGULAR NOUN PHRASE(NPA)
24	APPOSITIONAL NOUN PHRASE(NPC)
25	APPOSITIONAL PHRASE(NAP)
26	GENERAL NOUN PHRASE(NP)
27	SPECIFIC QUANTITY ADVERB PHRASE(DPA)
28	GENERAL QUANTITY ADVERB PHRASE(DPB)
29	QUALITATIVE ADVERB PHRASE(APC)
30	ADVERB PHRASE(DP)
31	BASIC PARTICIPLE PHRASE(EA).
32	VERB MOOD PHRASE(VB)
33	THREE-TENSE VERB PHRASE(VBB)
34	EMPHATIC VERB PHRASE(VC)
35	SEVEN-TENSE VERB PHRASE(VAA)
36	VERB PHRASE(VA)
37	DEFINITE NUMBER PHRASE(BC)
38	PREPOSITION PHRASE(XP)
39	DIRECT OBJECT PHRASE(NO)
40	COPULATIVE ADVERB PHRASE(DPD)
41	SUBJECT PRONOUN PHRASE(RSP)
42	SUBJECT PHRASE(NSP)
43	OBJECT PHRASE(NOP)
44	INDIRECT PHRASE(NIP)
45	COPULATIVE PHRASE(NPX)
46	DIRECT-OBJ. VERB MODIF. PHRASE(VMA)
47	INDIRECT-OBJ.VERB MODIF.PHRASE(VMB)
48	DISCOURSE VERB MODIF. PHRASE(VMC)

Table 4-4 (Continued)

Symbol No.	Symbol Name
49	DOUBLE-ACCUS.VERB MODIF.PHRASE(VMD)
50	VERB MODIFYING PHRSE(VM)
51	OBJECT REL.CLAUSE VRB.MOD.PHRSE(VMR)
52	INDIR.REL.CLAUSE VRB.MOD.PHRASE(VMI)
53	PREDICATE PHRASE(VP)
54	OBJ.REL.CLAUSE VERB PHRSE(VRB)
55	INDIRECT REL.CLAUSE VERB PHRASE(VRI)
56	INFINITIVE CONSTRUCT PHRSE(NV)
57	INFINITIVE ABSOLUTE PHRASE(NW)
58	CONSTRUCT PARTICIPLE PHRS(EPB)
59	ABSOLUTE/CONSTRUCT PART.PHRASE(EPA)
60	PARTICIPLE PHRASE(EP)
61	POSSESSIVE INDEPENDENT CLAUSE(SAA)
62	DEFINITE INDEPENDENT CLSE(SAB)
63	INDEFINITE INDEPENDENT CLAUSE(SAC)
64	INDEPENDENT CLAUSE(SA)
65	OBJECTIVE RELATIVE PHRASE(SRO)
66	INDIRECT RELATIVE PHRASE(SRI)
67	RELATIVE PRONOUN CLAUSE(RG)
68	SUBJECT-OBJ.DEPENDENT CLAUSE(KN)
69	CIRCUMSTANTIAL DEPENDENT CLAUSE(KC)
70	CONDITIONAL CLAUSE(KK)
71	INTEROGATIVE CLAUSE(KI)
72	DISCOURSE CLAUSE(KD)
73	DEPENDENT CLAUSE SENTENCE(SD)
74	BASIC SENTENCE(S)
75	INTEROGATIVE SENTENCE(SI)
76	COMPLETED SENTENCE(SC)
77	ADJECTIVE(A)
78	NUMBER ABSOLUTE(B)
79	CONJUNCTION(C)
80	ADVERB(D)
81	PARTICIPLE ABSOLUTE(E)
82	PARTICIPLE CONSTRUCT(G)
83	DEFINITE ARTICLE(H)
84	NUMBER CONSTRUCT(I)
85	NOUN CONSTRUCT(J)
86	NEGATIVE(L)
87	NOUN ABSOLUTE(N)
88	SIGN OF DIRECT OBJECT(O)
89	PREPOSITION(P)
90	INTEROGATIVE(Q)
91	PRONOUN(R)
92	PUNCTUATION MRK(T)
93	PARTICLE(U)
94	VERB(V)
95	INFINITIVE ABSOLUTE(W)
96	INFINITIVE CONSTRUCT(Y)

Table 4-5

LIST OF ANALYSIS PREDICATES

Statement No.	Analysis Predicate
1	CONTAINS A PREPOSITION AND A PRON.
2	IS THE PREDICATE OF OBJ.INTEROG.PHS.
3	EXPRESSES THE NONCOMPARATIVE DEGREE.
4	EXPRESSES THE COMPARATIVE DEGREE.
5	EXPRESSES THE SUPERLATIVE DEGREE.
6	IS A BASIC POST-NOMINAL ADJ.PHRASE.
7	EXPRESSES POSSESSION BY A PRONOUN.
8	EXPRESSES POSSESSION BY A NOUN PHRS.
9	HAS A NONDETERMINATE NOUN.
10	HAS A DETERMINATE NOUN.
11	HAS A (DETERMINATE) PROPER NOUN.
12	NAMES THE SUBJECT.
13	DOES NOT NAME THE SUBJECT.
14	EXPRESSES POSSESSION BY A CONSTRUCT.
15	STANDS IN PLACE OF A NOUN.
16	MODIFIES A DETERMINATE NOUN.
17	CONSISTS OF A PERSONAL PRONOUN(DET).
18	CONSISTS OF AN OBJECT PRONOUN.
19	IS THE NUMBER ONE.
20	IS THE NUMBER TWO.
21	IS A NUMBER BETWEEN 3 AND 9.
22	IS THE NUMBER 10.
23	IS THE NUMBER 11.
24	IS THE NUMBER 12.
25	IS A NUMBER BETWEEN 13 AND 19.
26	IS THE NUMBER 20.
27	IS EITHER 30,40,50,60,70,80, OR 90.
28	IS THE NUMBER 100.
29	IS THE NUMBER 200.
30	IS EITHER 300, 400,....,800 OR 900.
31	IS THE NUMBER 1000.
32	IS THE NUMBER 2000.
33	IS EITHER 3000,4000,...., OR 9000.
34	IS A UNITS NUMBER PHRASE.
35	IS A TENS NUMBER PHRASE.
36	IS A TEENS NUMBER PHRASE.
37	IS A MULTI-TENS NUMBER PHRASE.
38	CONSISTS OF A MULTI-TEN AND UNITS.
39	IS THE NUMBER 100 OR MULTIPLE OF IT.
40	IS A NUMBER BETWEEN 100 AND 999.
41	IS A NUMBER BETWEEN 1 AND 99.
42	IS A NUMBER BETWEEN 100 AND 999.
43	IS THE NUMBER 1000 OR ITS MULTIPLE.

Table 4-5 (Continued)

Statement No.	Analysis Predicate
44	IS A NUMBER BETWEEN 1000 AND 9999.
45	IS A NUMBER BETWEEN 1 AND 999.
46	IS A NUMBER BETWEEN 1000 AND 9999.
47	HAS A BASIC NOUN PHRASE AS NUCLEUS.
48	HAS A POSSESS.NOUN PHRS.AS NUCLEUS.
49	HAS A PARTICIPLE PHRASE AS NUCLEUS.
50	CONTAINS NO CONSTRUCT NOUNS.
51	CONTAINS CONSTRUCT NOUN(S).
52	IS A DEFINITE NUMBER PHRASE.
53	IS A REGULAR NOUN PHRASE.
54	IS AN APPOSITIONAL NOUN PHRASE.
55	IS A PREPOSITIONAL PHRASE.
56	IS A RELATIVE CLAUSE.
57	IS A REG.NOUN PHRS.+(APP.NOUN.PHRS).
58	IS A SPEC. QUANT. ADV.+(NUMBER).
59	IS A NUMBER + A SPEC. QUANT.ADVERB.
60	IS A CLASS 4 ADVERB +(MODIFIER).
61	IS A CLASS 5 ADVERB +(MODIFIER).
62	IS A TEMPORAL ADVERB PHRASE.
63	IS A LOCATIVE ADVERB PHRASE.
64	IS A SPECIFIC QUANTITY ADV. PHRASE.
65	IS A GENERAL QUANTITY ADVERB PHRASE.
66	IS A QUALITATIVE ADVERB PHRASE.
67	IS AN INTENSITY ADVERB PHRASE.
68	IS A PREPOSTIONAL PHRASE.
69	IS AN UNDERTERMINATED PARTICIPLE.
70	IS A DETERMINATED PARTICIPLE.
71	IS A VERB OR INFINITIVE ABSOLUTE.
72	IS A VERB OR PARTICIPLE.
73	OMITS VERB FOR PRES.ACT.INDIC.COPUL.
74	EXPRESSES EMPHASIS OF CERTAINTY(BH).
75	EXPRESSES EMPHASIS OF DURATION(BH).
76	EXPRESSES NO SPECIAL EMPHASIS.
77	IS AN EMPHATIC VERB PHRASE.
78	IS A SEVEN-TENSE VERB PHRASE.
79	IS A NONDETERMINATE NUMBER PHRASE.
80	IS A DETERMINATE NUMBER PHRASE.
81	GOVERNS A NOUN PHRASE.
82	IS A PREPOSITIONAL PRONOUN PHRASE.
83	GOVERNS A RELATIVE PRONOUN CLAUSE.
84	IS A NOUN PHRASE.
85	IS A RELATIVE PRONOUN CLAUSE.

Table 4-5 (Continued)

Statement No.	Analysis Predicate
86	IS A TEMPORAL ADVERB.
87	IS A LOCATIVE ADVERB.
88	IS A SUBJECT PRONOUN +(APPOS.N.PH.).
89	IS A DEMONSTRATIVE PRONOUN.
90	IS A NOUN PHRASE.
91	IS A SUBJECT PRONOUN PHRASE.
92	IS A SUBJECT-OBJECT DEPENDENT CLAUSE
93	IS AN INFINITIVE CONSTRUCT PHRASE.
94	IS A NONDETERMINATE NOUN PHRASE.
95	IS A DETERMINATE DIRECT OBJECT PHRS.
96	IS A PREPOSITIONAL PHRASE.
97	IS AN INFINITIVE PHRASE.
98	IS AN ADJECTIVE PHRASE.
99	IS A COPULATIVE ADVERB PHRASE.
100	IS A NOUN PHRASE.
101	IS A SUBJECT PRONOUN PHRASE.
102	IS A PREPOSITIONAL PHRASE.
103	IS A CIRCUMSTANTIAL DEPENDENT CLSE.
104	IS A PRONOUN +(ADVERB PHRASE).
105	IS A DIRECT OBJ. PRON.+(ADV.PHSE).
106	IS A DIRECT OBJECT PHRASE.
107	HAS AN OBJ. PRON.AND AN IND.PHRSE.
108	HAS A DIR.OBJ.PRON.AND.AN IND.PHRSE.
109	HAS AN OBJ.PHRSE. AND AN IND.PHRASE.
110	HAS AN OBJ.PRON. AND A DISC. CLAUSE.
111	HAS A DIR.OBJ.PRON.AND A DISC.CLSE.
112	HAS A DISC.CLAUSE +(OTHER MODIFS).
113	HAS AN OBJ.PRON.AND A NOUN PHRASE.
114	HAS A DIR.OBJ.PRON.AND A NOUN PHRSE.
115	HAS AN OBJ.PHRASE AND A NOUN PHRASE.
116	IS A COPULATIVE PHRASE.
117	IS AN ADVERB PHRASE.
118	IS A DIRECT OBJECT VERB MOD.PHRASE.
119	IS AN INDIR.OBJECT VERB MOD.PHRASE.
120	IS A PREPOSITIONAL PHRASE.
121	IS AN INFINITIVE PHRASE.
122	IS A DISCOURSE VERB MODIF.PHRASE.
123	IS A DOUBLE ACCUS.VERB MOD.PHRASE.
124	IS AN ADVERB PHRASE.
125	IS A DIR.OBJ.VERB MODIF.PHRASE.
126	IS AN INDIR.OBJ. VERB MODIF.PHRASE.
127	IS A PREPOSITION-PRON.+(ADV.MODIF.).

Table 4-5 (Continued)

Statement No.	Analysis Predicate
128	IS A DOUBLE-ACC.VERB MODIF. PHRASE.
129	IS AN OBJ.PRN.+AND A PREP.PRON.PHRSE.
130	IS A DIR-OBJ.PRON.+ PREP.PRON.PHRSE.
131	IS A PREP.PRON.PHRSE + DIR.OBJ.PHSE.
132	IS A VERB PHRASE + VRB.MODIF.PHRASE.
133	IS A VRB.PHRSE.+ IND.REL.CLS.VB.M.P.
134	HAS PREVIOUSLY NAMED SUBJECT.
135	REFERS TO SUBJECT BY MEANS OF PRON.
136	NAMES SUBJECT BY MEANS OF NOUN PHSE.
137	IS AN INFIN.ABS.AND VRB.MOD.PHRASE.
138	HAS A NOUN PHRASE AS OBJECT.
139	HAS A PRONOUN AS OBJECT.
140	IS A PART.PHRS.+ VRB.MODIF.PHRASE.
141	IS A CONSTRUCT PARTICIPLE PHRASE.
142	IS A ABS/CONST. PARTICIPLE PHRASE.
143	EMPHASIZES THE POSESSOR.
144	EMPHASIZES THE THING POSSESSED.
145	HAS NO SPECIAL EMPHASIS.
146	HAS SPECIAL SYNTAX FOR PRES.ACT.IND.
147	HAS A NAMED SUBJECT.
148	NAMES SUBJECT. SPECL.PRES.ACT.IND.
149	HAS PRON.SUB.. SPECL.PRES.ACT.IND.
150	HAS PRON.SUB.. SPECL.NEG.PR.ACT.IND.
151	NAMES SUBJ.. SPECL.NEG.PR.ACT.IND.
152	EMPHASIZES VERBAL IDEA, SPEC.PR.A.I.
153	REFERS TO SUBJECT PREVIOUSLY NAMED.
154	IS A POSSESSIVE INDEPENDENT CLAUSE.
155	IS AN INDEFINITE INDEPENDENT CLAUSE.
156	IS A DEFINITE INDEPENDENT CLAUSE.
157	REFERS TO PREVIOUSLY NAMED SUBJECT.
158	NAMES THE SUBJECT OF THE VERB.
159	HAS REL.PRON.AS SUBJECT OF VERB.
160	HAS REL.PRON.AS OBJECT OF VERB.
161	HAS REL.PRON.RELATED TO VERB BY PRP.
162	USES BIBLICAL HEBREW CONJUNCT.(KY).
163	USES MODERN HEBREW CONJUNCTION(S).
164	IS A TIME DEPENDENT CLAUSE.
165	IS A PURPOSE-RESULT DEPENDENT CLSE.
166	IS A CAUSE-REASON DEPENDENT CLAUSE.
167	IS A CIRCUMSTANTIAL PREP.PHRASE.
168	IS THE PROTASIS OF THE COND.SENTNCE.
169	QUESTIONS TRUTH/CIRCUMSTANCES OF SN.
170	ASKS WHO/WHAT IS SUBJECT OF VERB.

Table 4-5 (Continued)

Statement No.	Analysis Predicate
171	ASKS WHO/WHAT IS OBJECT OF VERB.
172	ASKS WHO/WHAT IS IND.OBJ.OF VERB.
173	QUOTES THE DISCOURSE INDIRECTLY.
174	QUOTES THE DISCOURSE DIRECTLY.
175	HAS NO EMPHASIS ON DEPENDENT CLAUSE.
176	HAS SOME EMPHASIS ON DEPENDENT CLSE.
177	IS A SIMPLE SENTENCE.
178	IS A DEPENDENT CLAUSE SENTENCE.
179	IS A CONDITIONAL SENTENCE.
180	HAS NO DEPENDENT CLAUSE.
181	HAS A DEPENDENT CLAUSE (NO EMPHAS).
182	HAS A DEPENDENT CLAUSE (EMPHASIS).
183	IS A DECLARATIVE SENTENCE.
184	IS AN INTEROGATIVE SENTENCE.
185	IS AN IMPERATIVE SENTENCE.

Table 4-6

INDEX OF ANALYSIS STATEMENTS

Symbol No.	Symbol Class 1	2	3	4	5	6	7	8
1	1	0	0	0	0	0	0	0
2	2	0	0	0	0	0	0	0
3	3	4	5	0	0	0	0	0
4	6	0	0	0	0	0	0	0
5	7	8	0	0	0	0	0	0
6	9	10	11	0	0	0	0	0
7	12	13	0	0	0	0	0	0
8	14	0	0	0	0	0	0	0
9	15	16	17	0	0	0	0	0
10	18	0	0	0	0	0	0	0
11	19	20	21	0	0	0	0	0
12	22	0	0	0	0	0	0	0
13	23	24	25	0	0	0	0	0
14	26	27	0	0	0	0	0	0
15	28	29	30	0	0	0	0	0
16	31	32	33	0	0	0	0	0
17	34	35	36	37	38	0	0	0
18	39	40	0	0	0	0	0	0
19	41	42	0	0	0	0	0	0
20	43	44	0	0	0	0	0	0
21	45	46	0	0	0	0	0	0
22	47	48	49	0	0	0	0	0
23	50	51	52	0	0	0	0	0
24	53	0	0	0	0	0	0	0
25	54	55	56	0	0	0	0	0
26	57	0	0	0	0	0	0	0
27	58	59	0	0	0	0	0	0
28	60	0	0	0	0	0	0	0
29	61	0	0	0	0	0	0	0
30	62	63	64	65	66	67	68	0
31	69	70	0	0	0	0	0	0
32	71	0	0	0	0	0	0	0
33	72	73	0	0	0	0	0	0
34	74	75	76	0	0	0	0	0
35	77	0	0	0	0	0	0	0
36	78	0	0	0	0	0	0	0
37	79	80	0	0	0	0	0	0

Table 4-6 (Continued)

Symbol No.	Symbol Class 1	2	3	4	5	6	7	8
1	81	82	83	0	0	0	0	0
2	84	85	0	0	0	0	0	0
3	86	87	0	0	0	0	0	0
4	88	89	0	0	0	0	0	0
5	90	91	92	93	0	0	0	0
6	94	95	0	0	0	0	0	0
7	96	97	0	0	0	0	0	0
8	98	99	100	101	102	103	0	0
9	104	105	106	0	0	0	0	0
10	107	108	109	0	0	0	0	0
11	110	111	112	0	0	0	0	0
12	113	114	115	0	0	0	0	0
13	116	117	118	119	120	121	122	123
14	124	125	126	127	128	0	0	0
15	129	130	131	0	0	0	0	0
16	132	0	0	0	0	0	0	0
17	132	0	0	0	0	0	0	0
18	133	0	0	0	0	0	0	0
19	134	135	136	0	0	0	0	0
20	137	0	0	0	0	0	0	0
21	138	139	0	0	0	0	0	0
22	140	141	0	0	0	0	0	0
23	142	0	0	0	0	0	0	0
24	143	144	145	146	0	0	0	0
25	147	148	149	150	151	152	0	0
26	153	0	0	0	0	0	0	0
27	154	155	155	0	0	0	0	0
28	157	158	0	0	0	0	0	0
29	157	158	0	0	0	0	0	0
30	159	160	161	0	0	0	0	0
31	162	163	0	0	0	0	0	0
32	164	165	166	167	0	0	0	0
33	168	0	0	0	0	0	0	0
34	169	170	171	172	0	0	0	0
35	173	174	0	0	0	0	0	0
36	175	176	0	0	0	0	0	0
37	177	178	179	0	0	0	0	0
38	180	181	182	0	0	0	0	0
39	183	184	185	0	0	0	0	0

Table 4-7

LIST OF FEATURE VALUES

Feature Subscript	Feature Name	Subscript valve							
		1	2	3	4	5	6	7	8
f	compounding	f=1	f=2	f=3	f=4	-	-	-	-
k	com.	once	twice	three	four	five	six	seven	eight
b	compounding class	conj.	disj.	-	-	-	-	-	-
c	Symbol class	one	two	three	four	five	six	seven	eight
ℓ	negation type	one	two	three	four	five	six	seven	eight
y	negation	neg.	neg.	neg.	neg.	neg.	neg.	neg.	neg.
d	determination	indef.	def.	-	-	-	-	-	-
n	number	sing.	dual.	pl.	-	-	-	-	-
g	gender	masc.	fem.	-	-	-	-	-	-
p	pers.	first.	sec.	third	-	-	-	-	-
r	preposition class	r=1	r=2	r=3	r=4	r=5	r=6	r=7	r=8
a	verb class	a=1	a=2	a=3	a=4	a=5	a=6	a=7	a=8
v	voice	act.	pass.	refl.	-	-	-	-	-
i	mood	ind.	impv.	subj.	-	-	-	-	-
t	tense	past	futr.	pres.	pst. C.	fut. C.	plpf.	f. prf.	-

Table 4-8 is a list of the variables used by the algorithm together with a description of each[4].

4.2 The Procedure

This section describes the procedure for using the structural model to analyze Hebrew sentences. It consists of the logical interaction of a set of operational functions that manipulate the data of the mapping functions in accordance with the specified values of the independent variables supplied by the user. Basically the procedure begins with a string of initial symbols of the Grammar of Hebrew Syntax[5] and applies the replacement rules of the grammar until only one terminal symbol remain. The major operational functions required to perform this task are listed below:

(1) Read in the initial symbols

(2) Write the initial symbols on the "A" list of symbols

(3) Initialize symbol counting indices

(4) Compare computed maximum value with 1

(5) Increment symbol counting index

(6) Compare symbol counting index with the computed maximum value

(7) Read a symbol from the "A" list of symbols

(8) Write a symbol on the "B" list from "A" list

(9) Locate a grammar rule that applies to a given symbol

(10) Determine that a rule exists

(11) Determine that the rule matches the given symbol with respect to all subscripts

(12) Initialize rule element counting indices

(13) Increment rule element counting indices

(14) Compare rule element counting index with the computed maximum value

(15) Determine that an element of the rule matches an element on the "A" list with respect to all subscripts

(16) Compute values for dependent-variable subscripts

---

[4]See Section 2.2.1 of Part II for a detailed description of the variables as they apply to Hebrew.

[5]Defined in Part II of this report.

Table 4-4

LIST OF VARIABLES

Var. No.	Var. Name	Description	Var. No.	Var. Name	Description
1	SN	Symbol No.	16	i	mood
2	m	opt./mand.	17	t	tense
3	f	comp. class	18	s	stem
4	k	comp. no.	19	w_1	Root 1
5	b	comp. type	20	w_2	Root 2
6	c	class	21	w_3	Root 3
7	ℓ	neg. class	22	w_4	Root 4
8	y	neg./pos	23	ST	Symbol Type
9	d	definiteness	24	RN	Rule No.
10	n	number	25	EN	Element No.
11	g	gender	26	NRH	No. Rt. Elements
12	p	person	27	RT	Restraint type
13	r	prep. class	28	RS	Restraint Subs.
14	a	verb. class	29	x	No./gend. Trans.
15	v	voice			

(17) Predict success of a rule after one or two passes.[7]

(18) Write a rule element on the "B" list

(19) Reset value of symbol counting index

(20) Erase the "A" list of symbols and transfer the symbols on the "B" list to the "A" list

(21) Write sequence of analysis statements

(22) Construct a tree diagram of the generated sentence.

Many minor operational functions that are associated with these are not listed. They are defined in Section 4.3 that describes the computer program of the algorithm. The above are sufficient for explaining the procedure of the algorithm.

The procedure consists of a logical manipulation of the operational functions so as to analyze a sentence. The logical interrelationship of the functions is defined in flow chart form in Figures 4.1a and 4.1b. The actual program of the procedure is more complex than this flow chart, but this is sufficient for explaining the procedure.

The initial symbol is read in (Block 1)[6] together with the value of all independent variable subscripts, and it is written on the "A" list of symbols (LIST1) as the first symbol (Block 2). The number of symbols on LIST1 (J_{max}) is computed and counting register (J) is initialized (Block 3). The value of J_{max} is compared with 1 (Block 4), if $J_{max} \leq 1$ computation procedes to Block 21, otherwise to Block 5. In Block 5 the counting resister (J) is indexed by one and the value of J is compared with J_{max} to determine whether or not all the symobls on list LIST1 have been processed (Block 6); if not, computation procedes to Block 7, otherwise to Block 20.

In Block 7, the J-th symbol of LIST1 is read, and an applicable grammar rule is located (Block 9). If there is no applicable rule (Block 10) computation goes to Block 8, otherwise to Block 11. In Block 8 the J-th symbol of LIST1 is written on LIST2 and computation returns to Block 5.

In Block 11, a test is made to determine that the first left hand element of the rule matches the J-th symbol of LIST1 in accordance with the following criteria:

(1) The symbol number must be the same for both symbols, or if not, the grammar rule symbol must be a variable symbol; and

---

[6]The "Block" number corresponds to operation function number previously listed.

[7]Predictive logic is required to synchronize the production of related constituents because the rules of the grammar are unordered (by original definition). Prediction to a depth of two passes enables the algorithm to analyze most simple Hebrew sentences. Greater predictive power is required for more complex sentences.

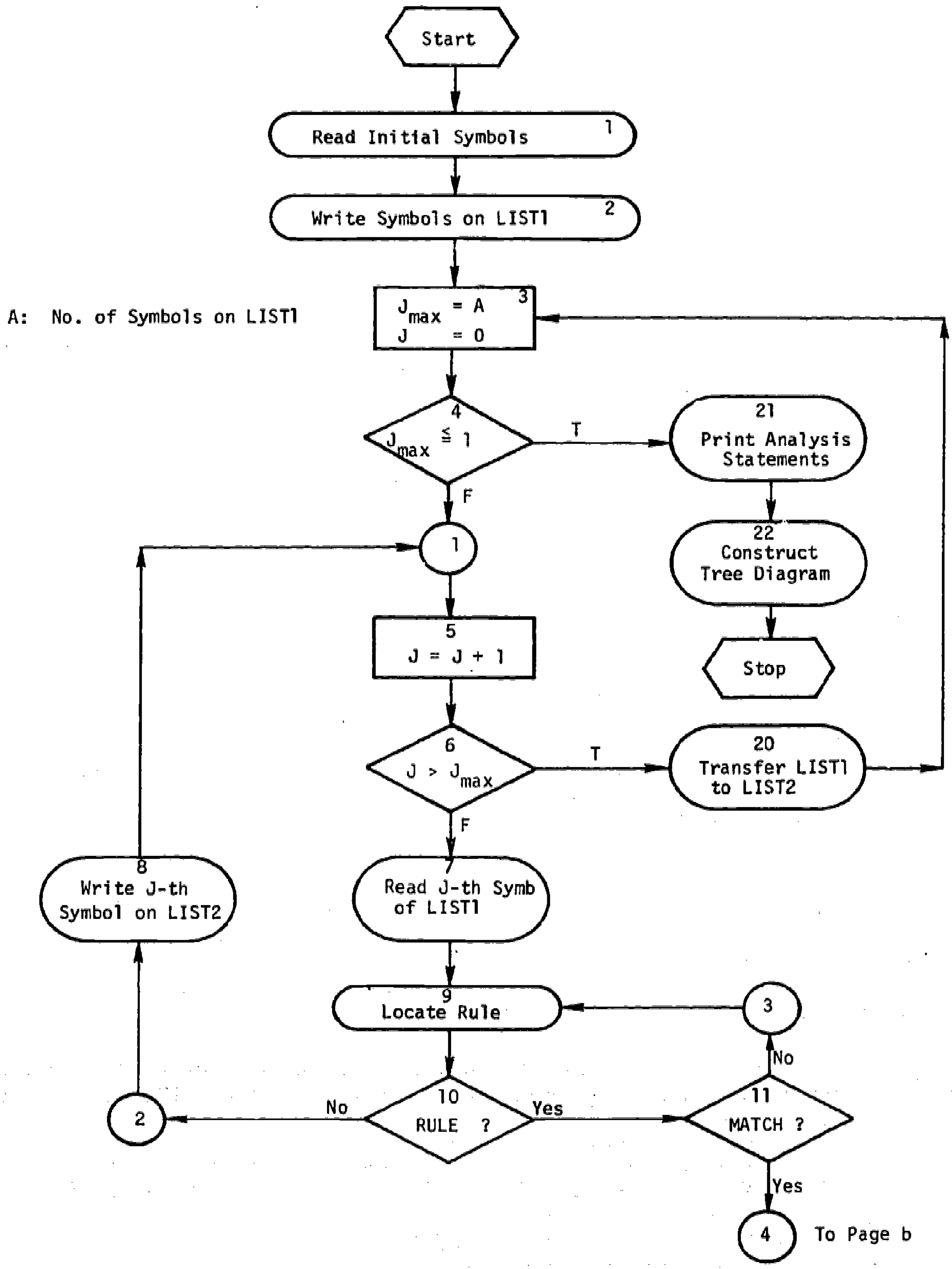
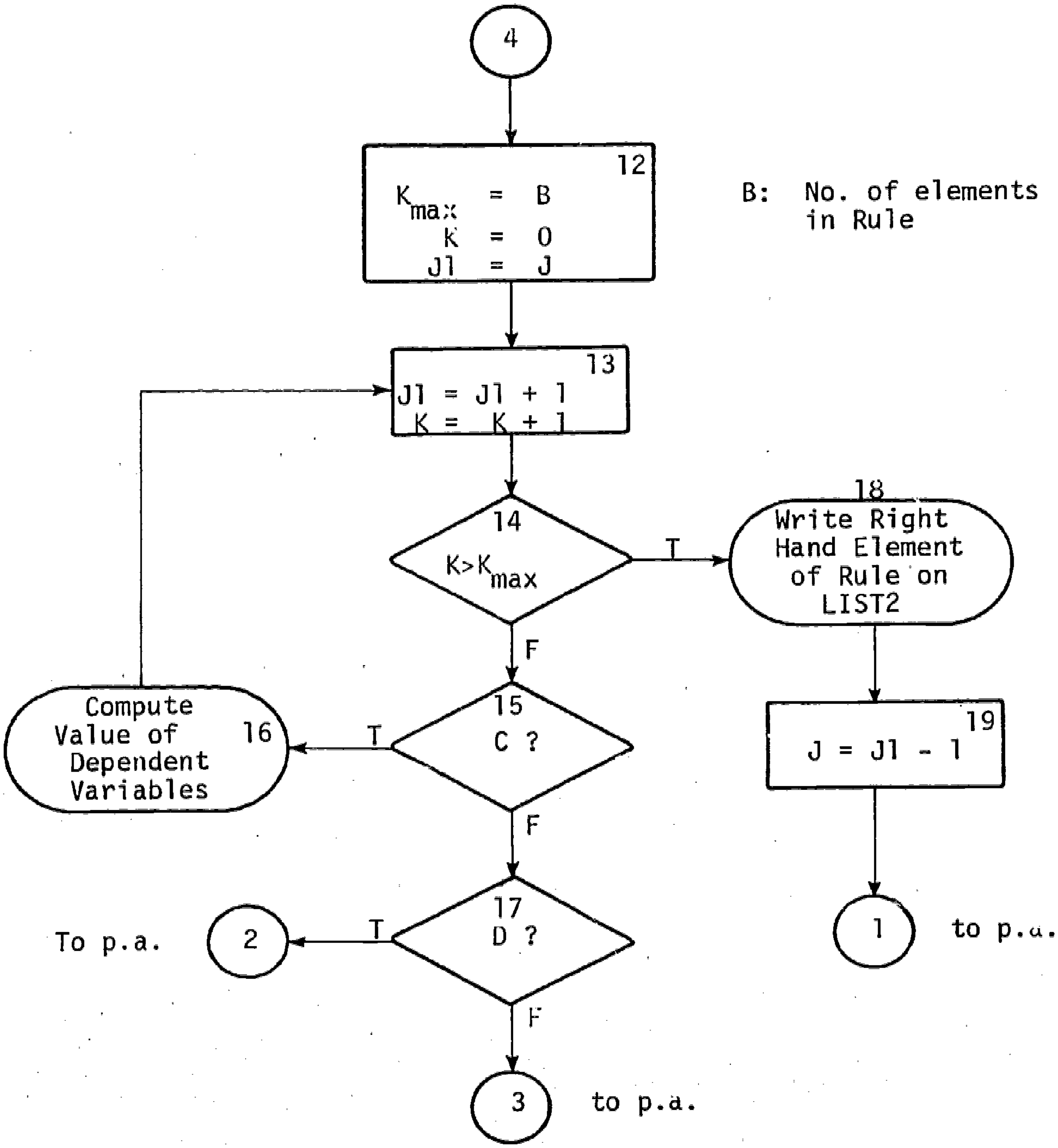


Figure 4.1a: Flow Chart of Procedure



C: Does J1-th Symbol of LIST1 Match K-th element of Rule?

D: Predict Success in 1 or 2 Future Passes?

Figure 4-1b: Flow Chart of Procedure

(2) the symbol class must be the same for both symbols; and

(3) every fixed-valued subscript of the rule symbol must be the same as the corresponding subscript of the J-th symbol; and

(4) the restraints on the rule symbol must be met.

If the criteria are not met (Block 11) computation returns to Block 9 for location of the next applicable grammar rule. If the criteria are met, computation procedes to Block 12.

In Block 12, the number of symbols on the right side of the rule (K_{max}) is computed and counting register (K) is initialized to zero and register J1 is set equal to J. Then registers K and J1 are indexed by one (Block 13) and the value of K is compared with K_{max} (Block 14) to determine whether or not all the right left elements of the rule been processed; if not, computation procedes to Block 15, otherwise to Block 18. In Block 15, a test is made to determine that the J1-th symbol of LIST1 matches the K-th element of the given rule in accordance with the criteria listed for Block 11. If the symbols match computation procedes to Block 16, otherwise to Block 17. In Block 16, the values of the variable subscripts of the K-th element of the rule are computed and computation returns to Block 13.

In Block 17, a test is made to predict whether the present rule would be satisified after one or two more passes, that is, whether the J1-th symbol of LIST1 will develop in one or two passes into the symbol required by the present rule. If success is predicted computation returns to Block 8 and the J-th symbol is held over so that the predicted development can take place; if success is not predicted, computation returns to Block 9 for a new rule. In Block 18, the right hand element of the rule is written on LIST2 with the computed values assigned to its subscripts, and computation procedes to Block 19.

In Block 19, the value of the symbol counting index is changed to the value J1-1, and computation returns to Block 5 to begin with the next symbol on LIST1.

Referring back to Block 6, if all symbols on LIST1 have been processed, computation procedes to Block 20 where the symbols on LIST1 are erased, those on LIST2 are transferred to LIST1, LIST2 is erased, and computation returns to Block 3 for processing the new symbols on LIST1.

Referring back to Block 4, if only one symbol remains on LIST1, the analysis is complete, so a list of analysis statements is assembled (Block 21), a tree deagram of the analyzed sentence is constructed (Block 22), and computation stops.

4.3 Computer Program of the Algorithm

This section describes a computer program of the Algorithm for Analyzing Hebrew Sentences. First a flow-chart description of the program is given, then an input map is provided with instructions on how to use the program. Appendix B of this part of the report contains a source language listing of the program in FORTRAN IV.

4.3.1 Flow-Chart Description of Computer Program

This section describes the computer program of the Algorithm for Analyzing Hebrew Sentences in terms of flow diagrams. The program consists of a main program ANALYZ and the following subprograms:

(1) ALPHA
(2) DIAGRM
(3) LIMIT
(4) MACHER
(5) OUTPUT
(6) PARSE
(7) PROPH1
(8) PROPH2
(9) RERITE
(10) RULENO
(11) SYMACH
(12) VARATT

Figure 4-2 is a map of the program showing the hierarchy of the calling sequences. The following sections contain descriptions of each portion of the program.

4.3.1.1 Main Program ANALYZ

This section describes the operation of the main program of the Algorithm for Analyzing Hebrew Sentences. This program manages the overall operation of the algorithm and calls the various subprograms at the appropriate times.

Figures 4.3 is a flow diagram of main program ANALYZ. The main program performs the following operations:

(1) Reads the program options (Fig. 4.3a)

(2) Reads the Transliteration Table (Fig. 4.3a)

(3) Reads the Matrix of Grammar Rules, transforms the alpha-numeric data to integers using Subprogram ALPHA, and stores the transformed rules in Matrix RULE (Fig. 4.3b).

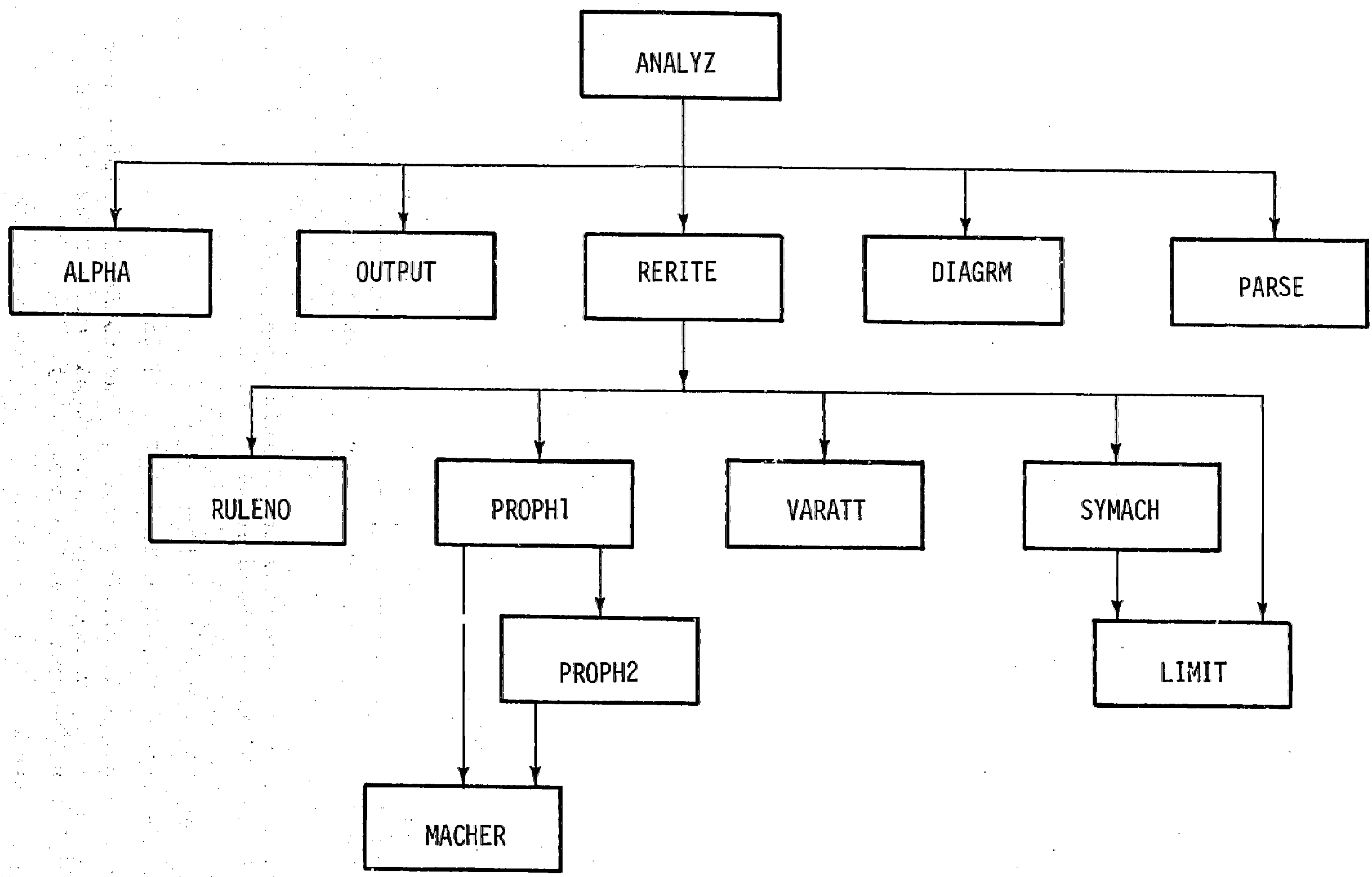


Figure 4.2: Map of Procedure

(1) Read Program Options

Read rules etc. from Mag. Tape if not MODE 0

Read rules etc. from Cards

(2) Read in transliteration

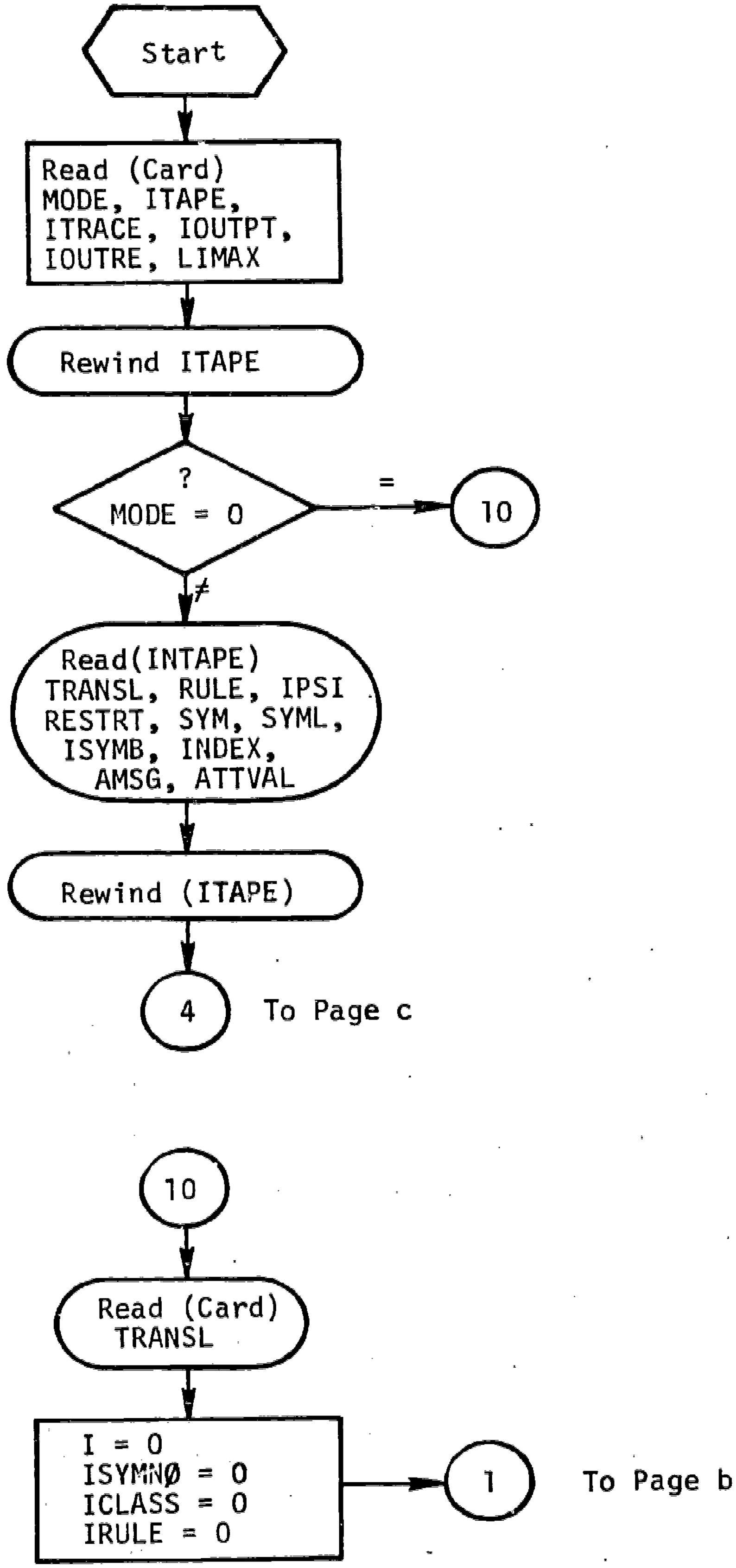


Figure 4.3a: Program ANALYZ

(3)
Read in Grammar Rules

(4)
Compute Storage address of Rules

(5)
Read Restraint Matrix

(6)
Read Symbol Names

(7)
Write on Mag. Tape

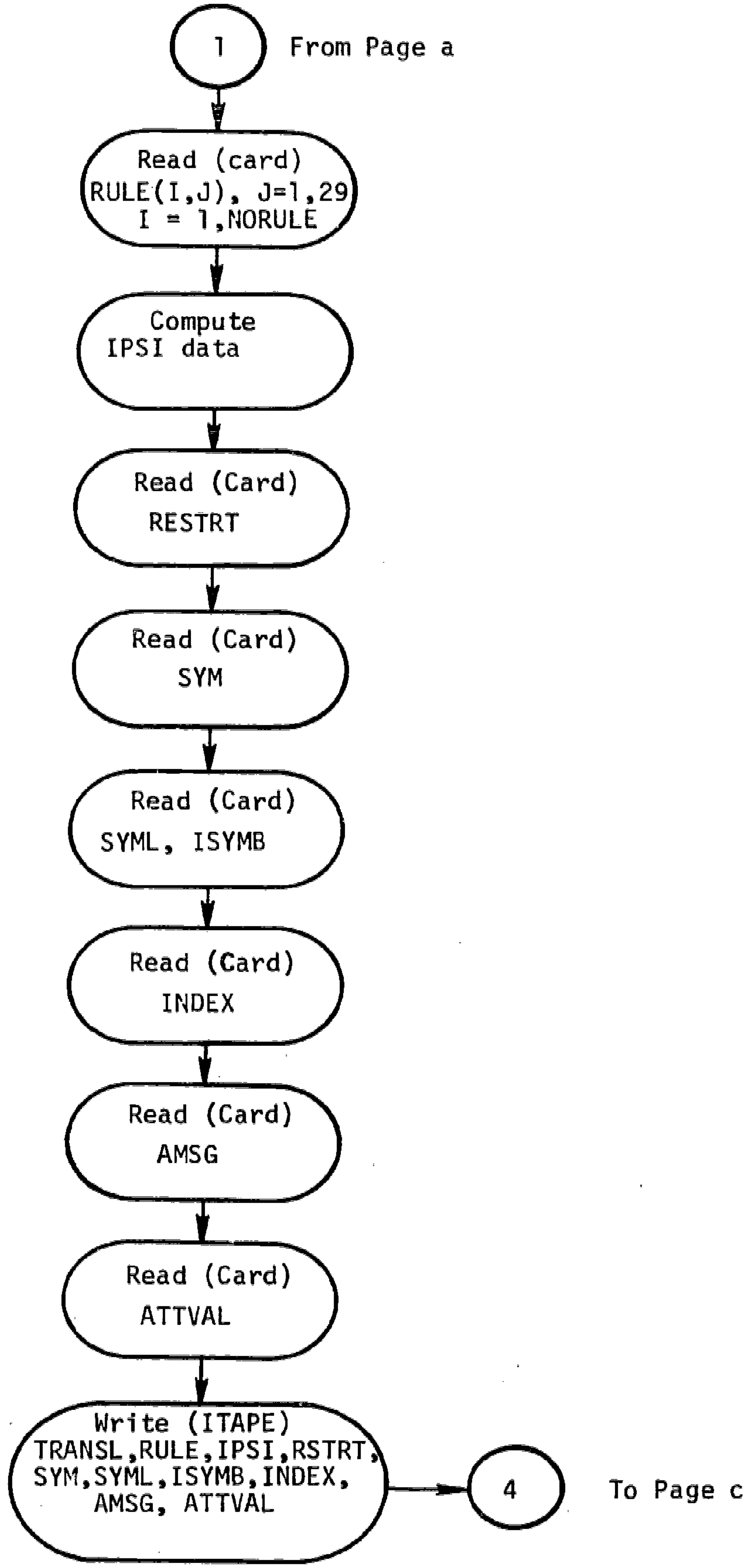


Figure 4.3b: Program ANALYZ

(8)
Read in Equivalent Hebrew Sentence

(9)

(10)
Read in Initial Symbols

(11)
Initialize Tree Matrix

(12)
Initialize Indices

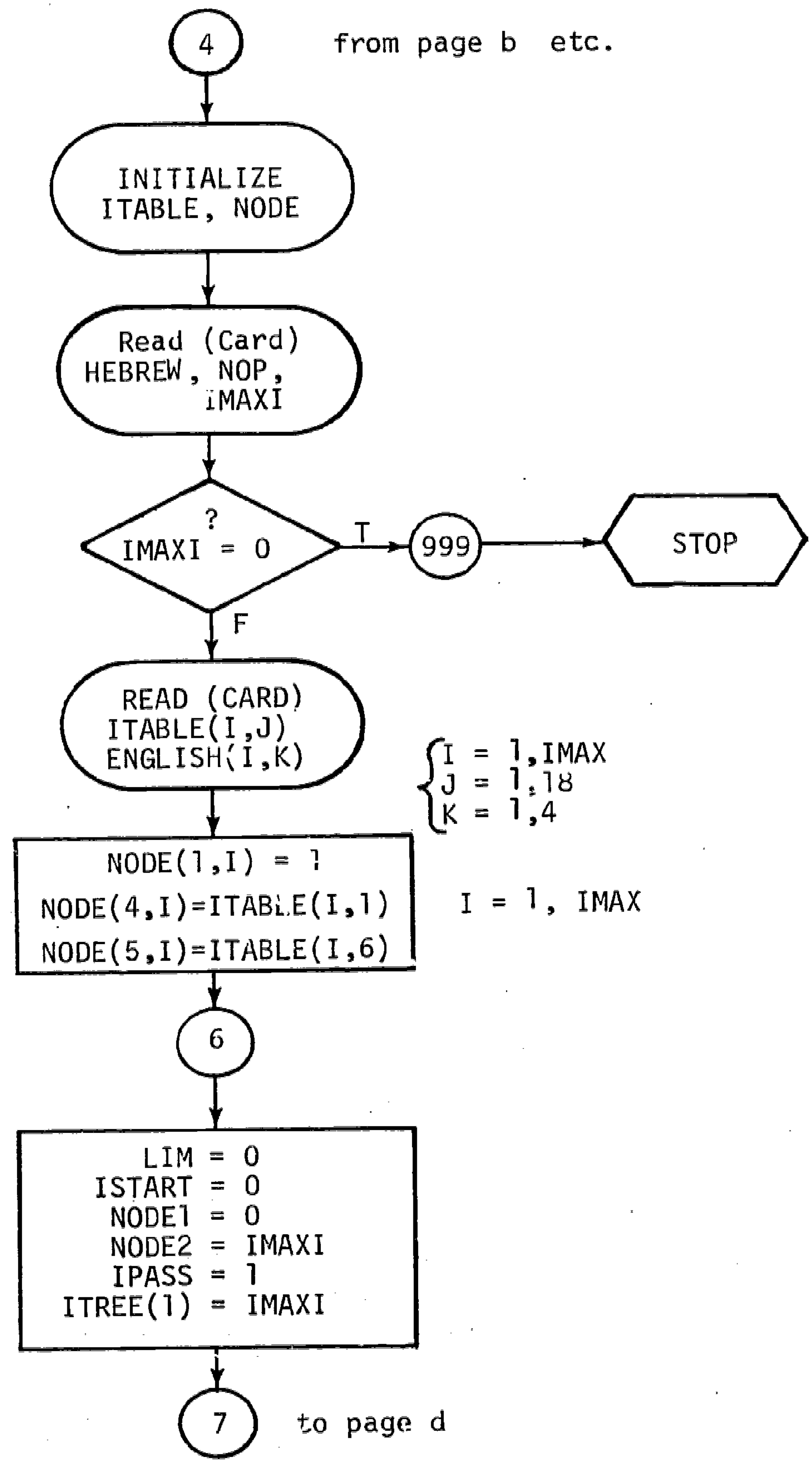


Figure 4.3c: Program ANALYZ

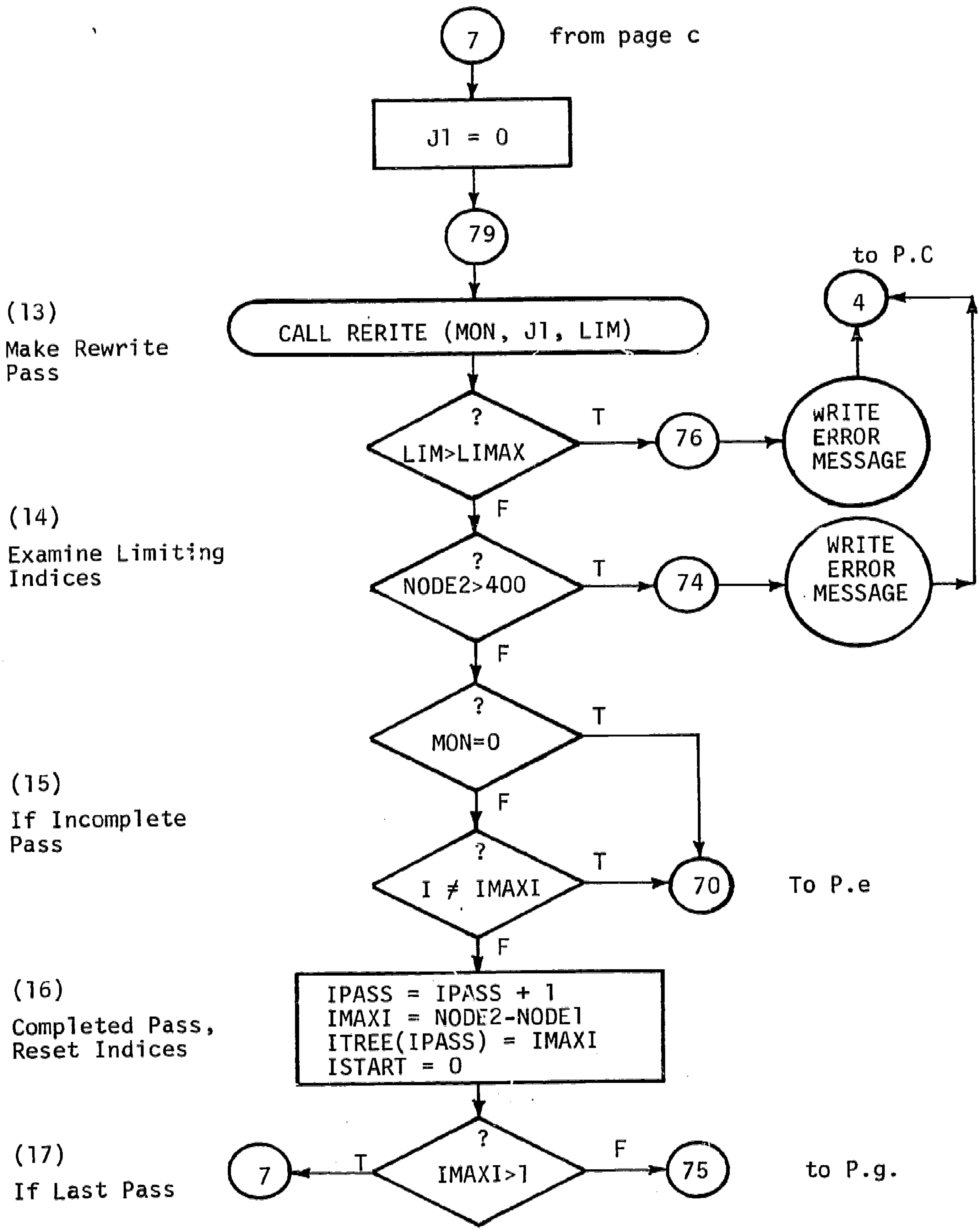


Figure 4.3.d: Program ANALYZ

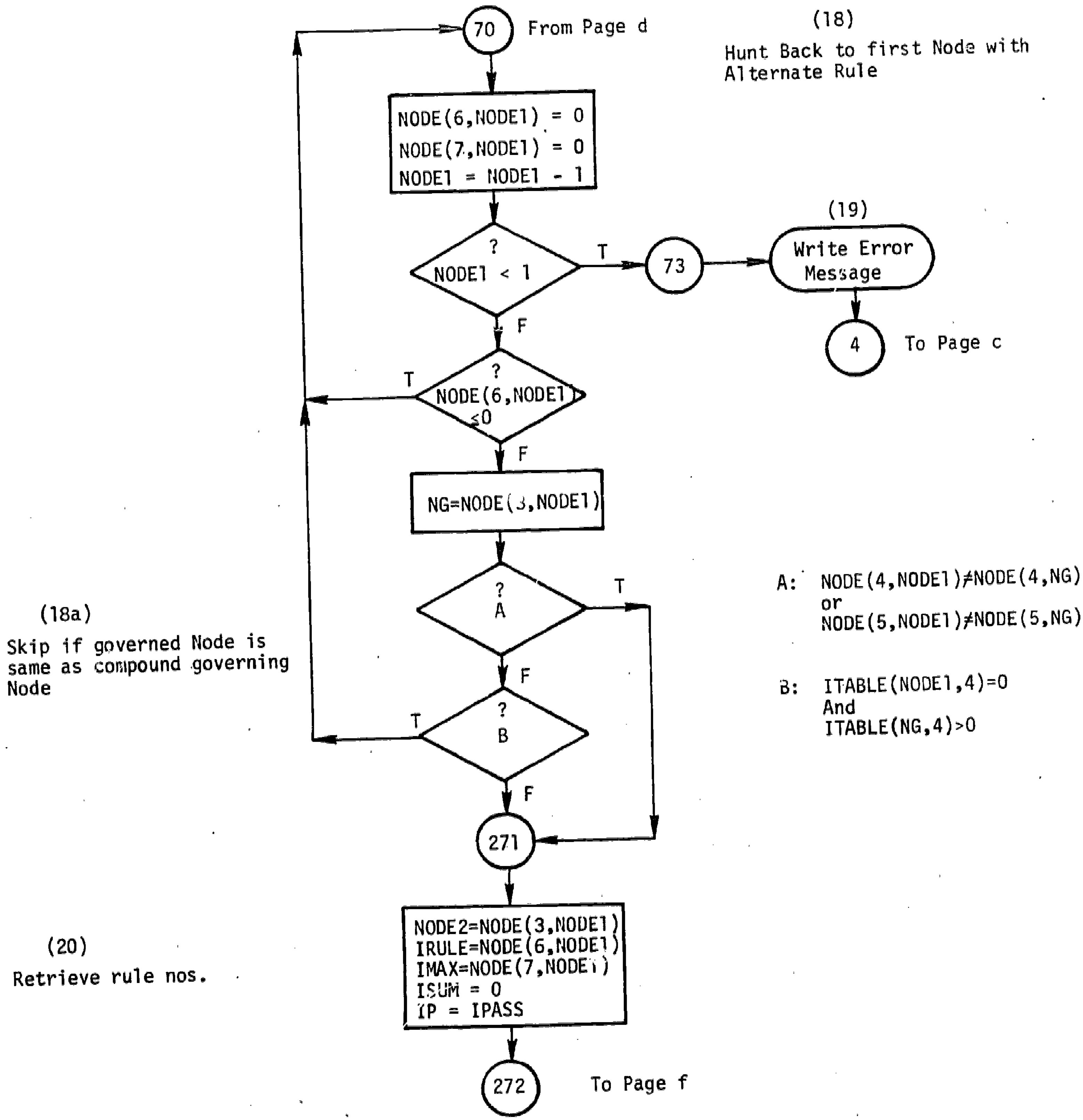


Figure 4.3e: Program ANALYZ

(21)

Cancel any
Previous
Prediction

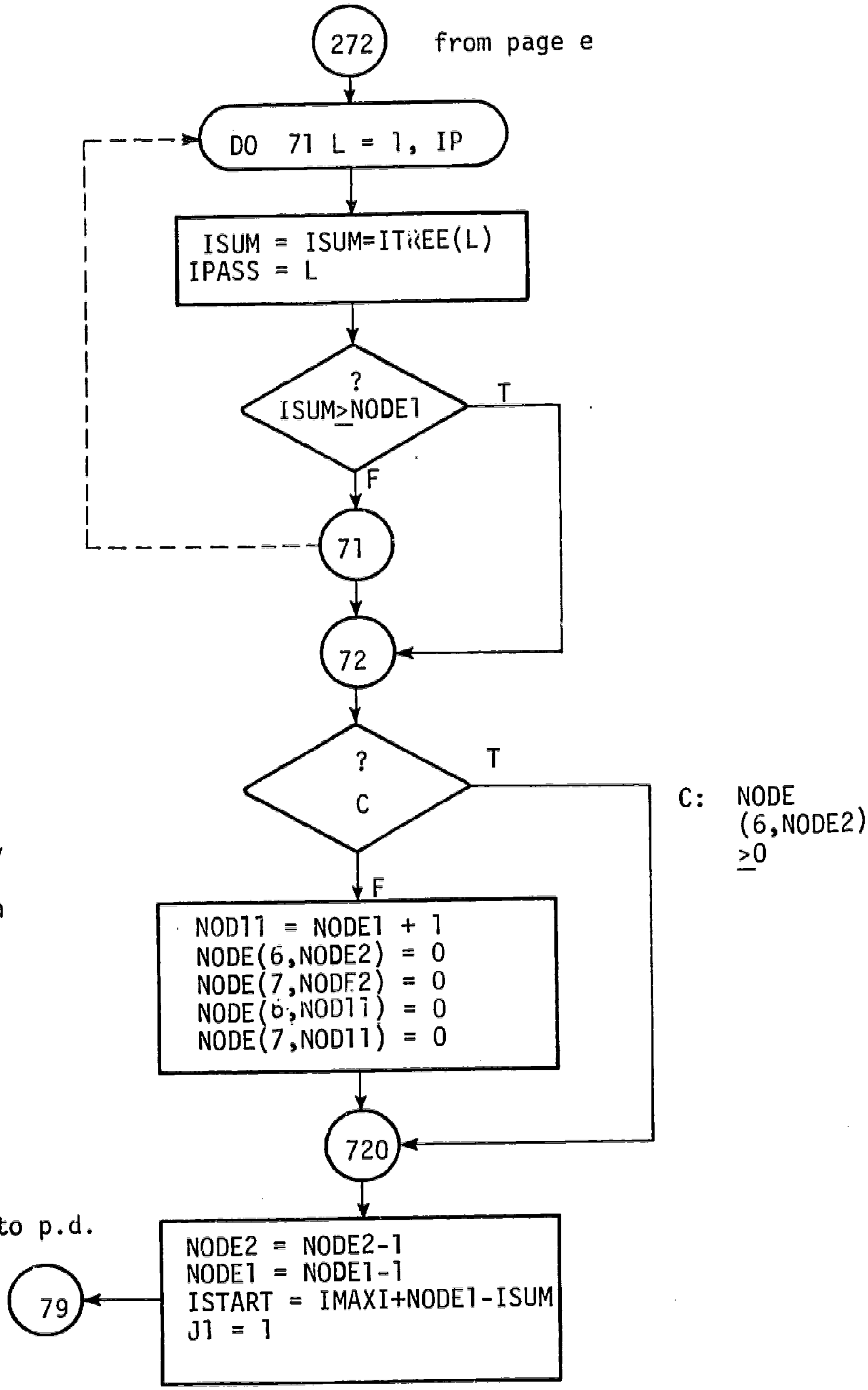


Figure 4.3f: Program ANALYZ

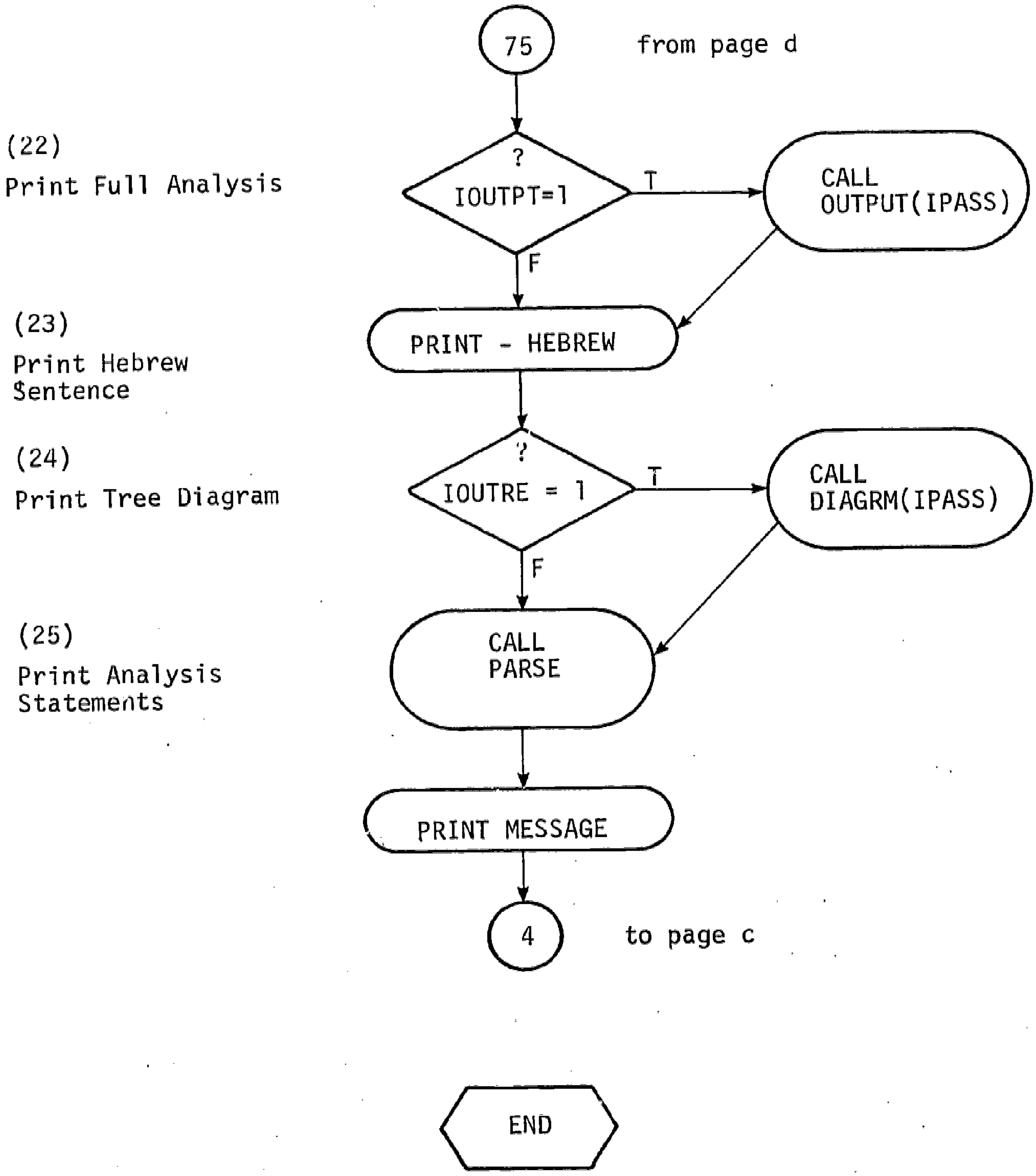


Figure 4.3g: Program ANALYZ

(4) Computes a catalog of the Rules (Fig. 4.3b)

(5) Reads the Restraint Table (Fig. 4.3c)

(6) Reads the table of Symbol Names that are associated with a symbol number (Fig. 4.3c), and other mapping functions

(7) Writes these data on a magnetic tape for permanent storage (Fig. 4.3c)

(8) Reads the equivalent Hebrew sentence and certain indices (Fig. 4.3c)

(9) Stops the program after the analysis of the last sentence (Fig. 4.3c)

(10) Reads in initial symbols and their English equivalents (Fig. 4.3c)

(11) Initializes tree diagram matrix data

(12) Initializes certain bookkeeping indices

(13) Makes a rewrite pass on the symbols in the string using Subroutine RERITE

(14) Examines the value of monitoring indices which limit the maximum range of computation; if the range is exceeded, error messages are written and computation ceases.

(15) Examines the value of indices MON and I, if no rules were applied on the last pass (MON=0) or if the pass did not process all symbols in the string (I≠IMAXI), computation skips to step (18), otherwise to (16).

(16) If the pass was complete, reset certain indices

(17) If not last pass, that is, there is still more than 1 symbol in the string, repeat from step (13), otherwise skip to step (22).

(18) Hunts back in derivation to first symbol with an alternate rule yet available, but (18a) skips the symbol if it is identical with its governing symbol.

(19) If no symbols with an alternate rule are found, writes a diagnostic message and terminates computation.

(20) When the first symbol with an alternate rule is found, the rule numbers are retrieved and certain indices are reset.

(21) Previous predictions about the symbol are cancelled, other indices are reset and computation returns to step (13).

(22) When analysis is completed, print full analysis (upon request) consisting of complete detailed listing of every symbol in every rewrite pass, using subprogram OUTPUT.

(23) Prints the Hebrew sentence being analyzed.

(24) Prints a tree diagram using subprogram DIAGRAM (upon request).

(25) Prints a complete set of analysis statements using subprogram PARSE, prints a completion message and returns to step (8) for another sentence.

4.3.1.2 Subprogram ALPHA

Subprogram ALPHA (Fig. 4.4) is used to transform alpha-numeric symbols into integers for use in the computations of the program. This subprogram permits input data to be submitted in a form more meaningful to the user. The subprogram functions in the following manner:

(1) A given alpha-numeric symbol (input argument A) is compared with each element in the table of Transliteration (array TRANSL).

(2) When a match is found, an integer value for output argument IA is computed such that IA is one less than the index L.

(3) However, if this value is 40, the value of IA is made zero. This converts all spaces to zero.[8]

The tranformation of the data is shown in Table 4-1. Other transformations can be obtained by changing the data in the Table of Transliteration. Subprogram ALPHA is called by the Main Program ANALYZ.

4.3.1.3 Subprogram DIAGRM

This subprogram (Figure 4.5) is called by the Main Program ANALYZ (step 24) to construct a tree diagram of an analyzed Hebrew sentence. At this stage the analysis of the sentence is complete and data defining the nodal structure of the tree diagram has been computed and stored in Matrix NODE, and Array ITREE. Matrix NODE contains a row corresponding to each node in the tree diagram. The data in Matric NODE is as follows:

NODE(I,J):

J = Node number

I = 1, Number of nodes the J-th node governs

I = 2, Line position of J-th node

I = 3, node number of node governing J-th node

---

[8]This computation is presently dependent on the "content" of the grammar.

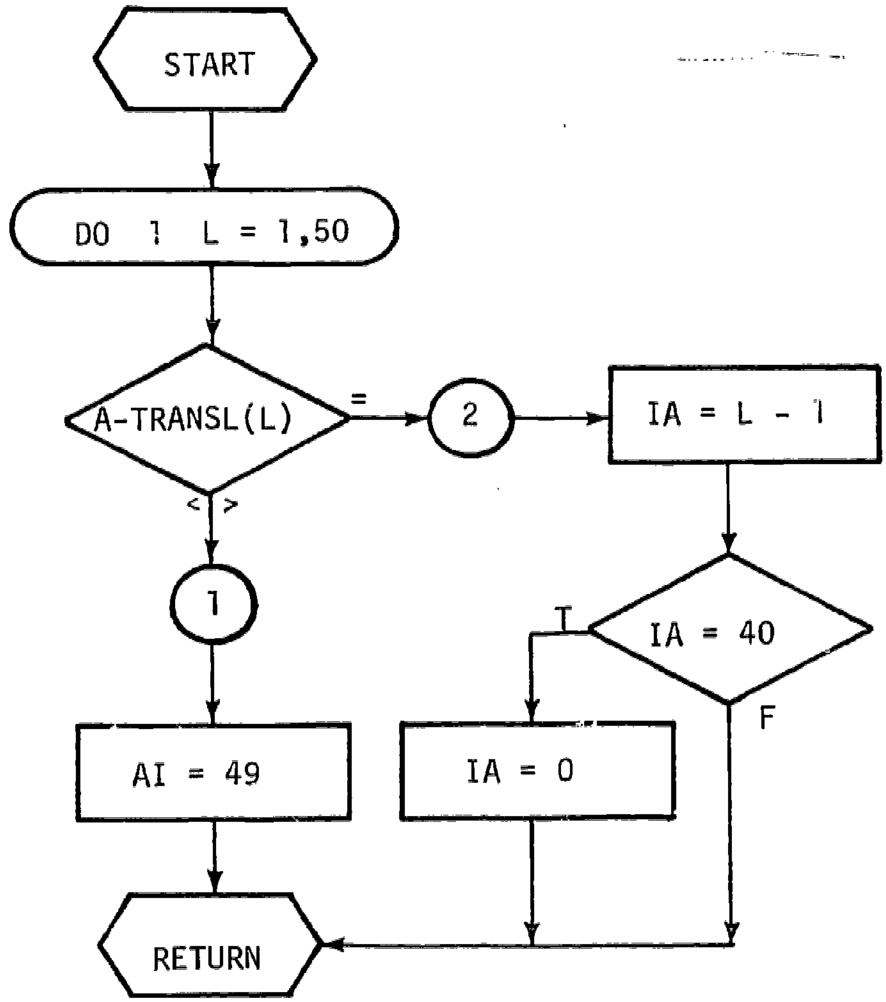


Figure 4.4: Subprogram ALPHA

I = 4, Symbol type at J-th node

I = 5, Symbol class at J-th node

Array ITREE defines the number of nodes at each level of the tree as follows:

ITREE (I):

I = level number

ITREE(I) = number of nodes in I-th level.

Figure 4.5a outlines the four main operations of this subprogram and refers to the corresponding figure for the detailed flow diagram of that operation.

These operations are:

(1) Compute the line position of each node. (Fig. 4.5b&c). This operation consists of filling in the data for Matrix NODE(2,J). Computation starts with the first level of nodes each of which is assigned a line position beginning with position 1 for the first, position 2 for the second, and so forth up to a maximum of 20. Computation then goes to the next level of nodes and the position of each node is computed to be midway between the position of the nodes it governs. Successively lower levels are computed until the last node is reached.

Steps 2 through 4 (following) are repeated in sequence for each level of nodes in the tree diagram. These steps cause the computer to print out the tree diagram of the analyzed sentence a level at a time beginning with the first level. Three parts are required for each level: (a) the upper connectors that show the relationship of the nodes in a given level to the governing nodes at the next highest level; (b) the symbol name and class of each node; and (c) the lower connectors that show the governing relationship of a given node to the nodes at the next lowest level.

(2) Write upper connectors (Figure 4.5d). The upper connectors occupy two lines on the printed output. The first one connects together (with a horizontal line) nodes governed by a common higher level node. The second line provides a vertical bar above each nodes position. This is accomplished by using various combinations of the following 3-character words:

BLANK = ΔΔΔ

DASHES = ---

BAR = IΔΔ

Where Δ represents a space. The logic for the first line is as follows:

(a) If the governing node governs only one node, BLANK + BAR is written above the node position, that is ΔΔΔIΔΔ.

(b) If the governing node governs more then one node, BLANK + DASHES (ΔΔΔ---) is written above the first of the governed nodes, DASHES + DASHES (------) is written above intermediate governed nodes, and DASHES + BLANK (---ΔΔΔ) is written above the last of the governed nodes. This provides a continuous line of dashes from the center of the position of the first node to the center of the position of the last node that are governed by one common higher level node. For the second line of the upper connectors, BLANK + BAR (ΔΔΔIΔΔ) is written above each node that governs at least 1 node and BLANK + BLANK (ΔΔΔΔΔΔ) is written above each node that governs zero nodes.

Upper connectors are omitted for the first level of nodes.

(3) Write line of nodes (Fig. 4.5e). This operation consists of printing the symbol name and class at each nodal position in the line. The symbol type is specified by NS=NODE (4,J), and the corresponding symbol name is obtained from SYM(NS). The symbol class is specified by NT=NODE(5,J) which is transformed to alpha-numeric characters by CLS(NT). However, if the symbol and class of a given node are the same as the symbol and class of the governing node, BLANK + BAR (ΔΔΔIΔΔ) is written in the position of the node; this eliminates redundant data from the tree.

(4) Write lower connectors (Figure 4.5f). This operation consists of writing BLANK + BAR (ΔΔΔIΔΔ) under the position of a given node if it governs one or more nodes and BLANK + BLANK (ΔΔΔΔΔΔ) if it governs zero nodes. This is the same as line 2 of the upper connectors, so the same coding is used for both under control of an index (IUPLOW). The lower connectors are omitted for the last level of nodes.

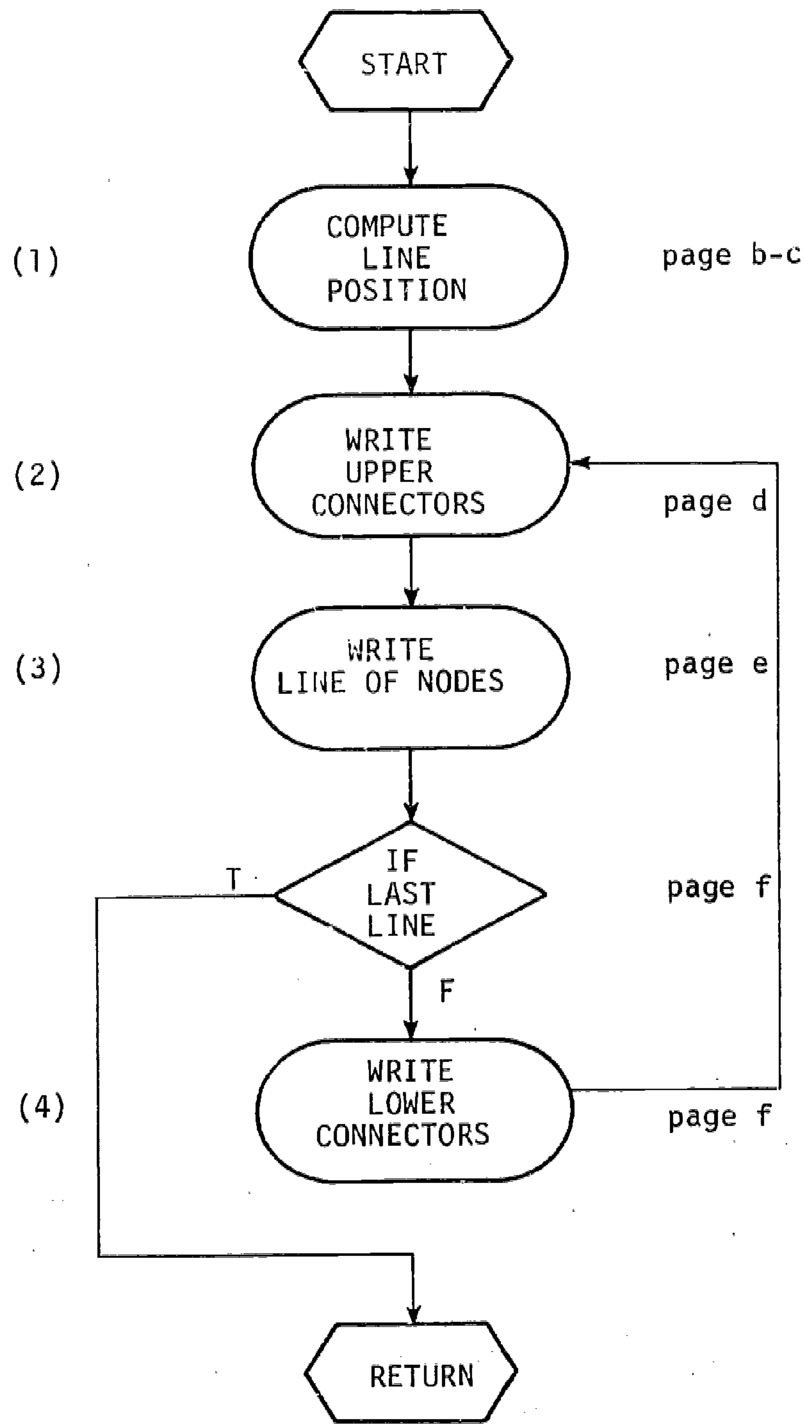


Figure 4.5a: Subprogram DIAGRM

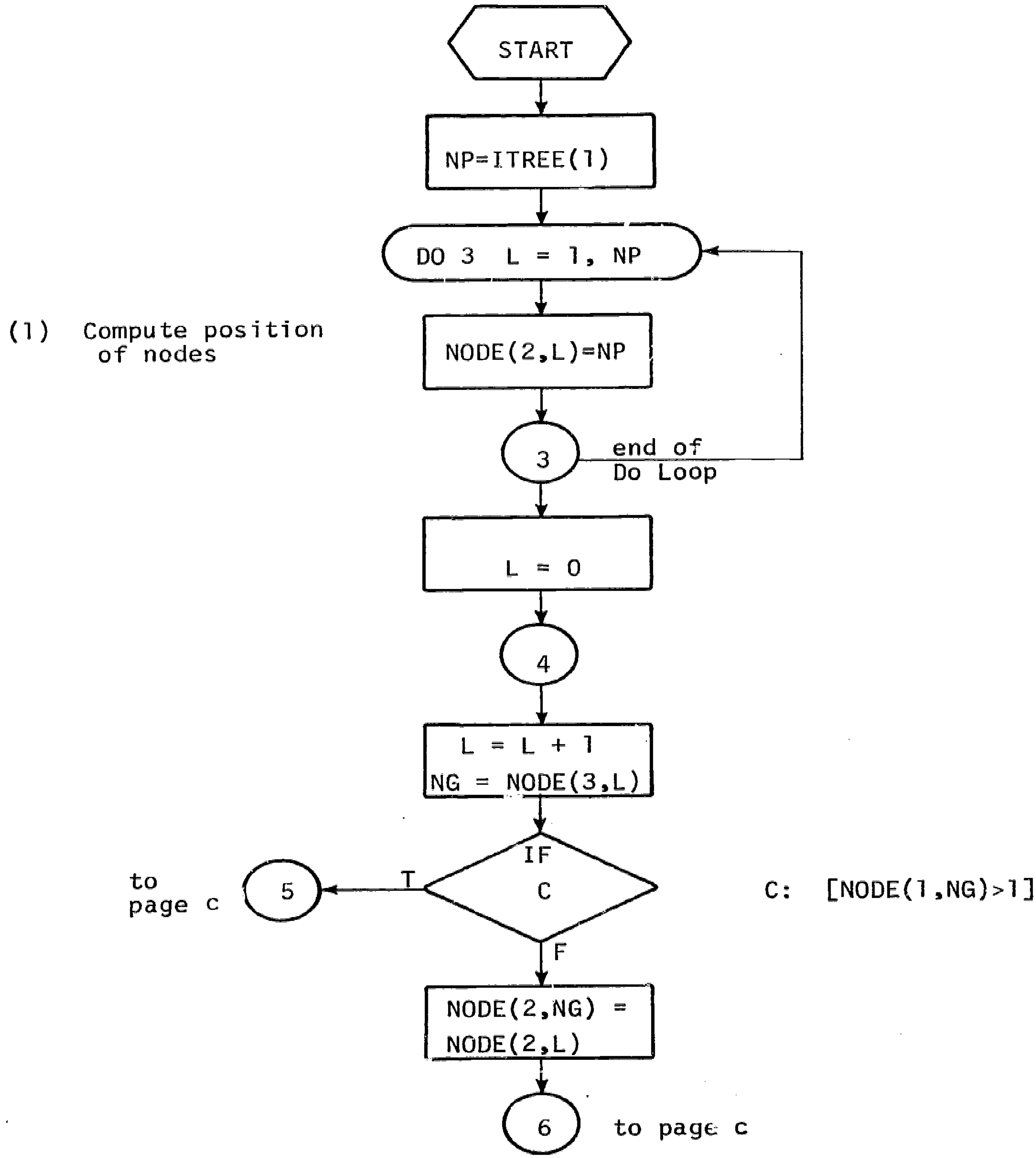


Figure 4.5b: Subprogram DIAGRM

Governing node put mid-way between first and last governed node.

Continue If Last Node, Otherwise Repeat For Next.

Produce IPASS Lines of Nodes

Omit Upper Connectors For First Line

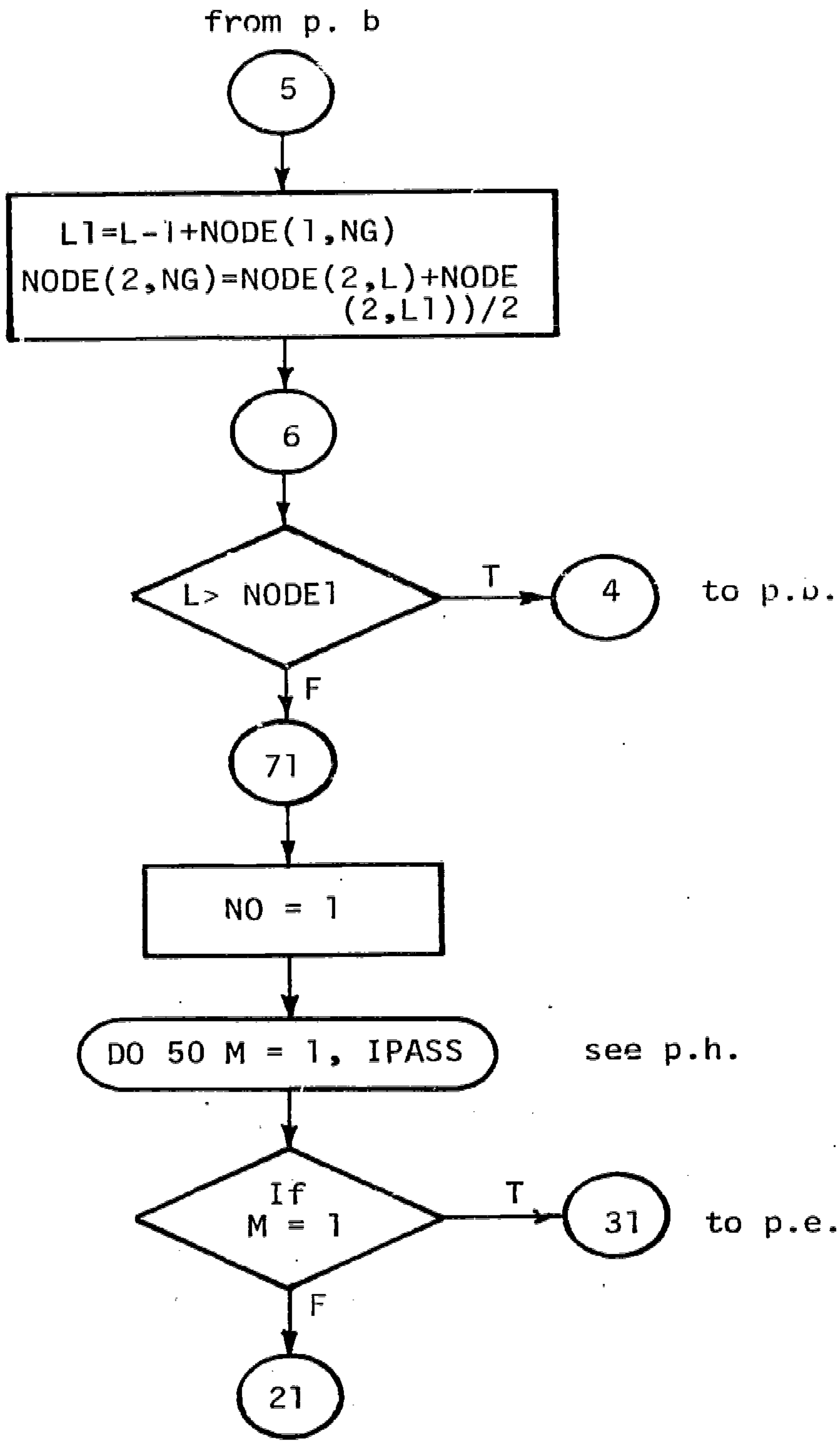


Figure 4.5c: Subprogram DIAGRM

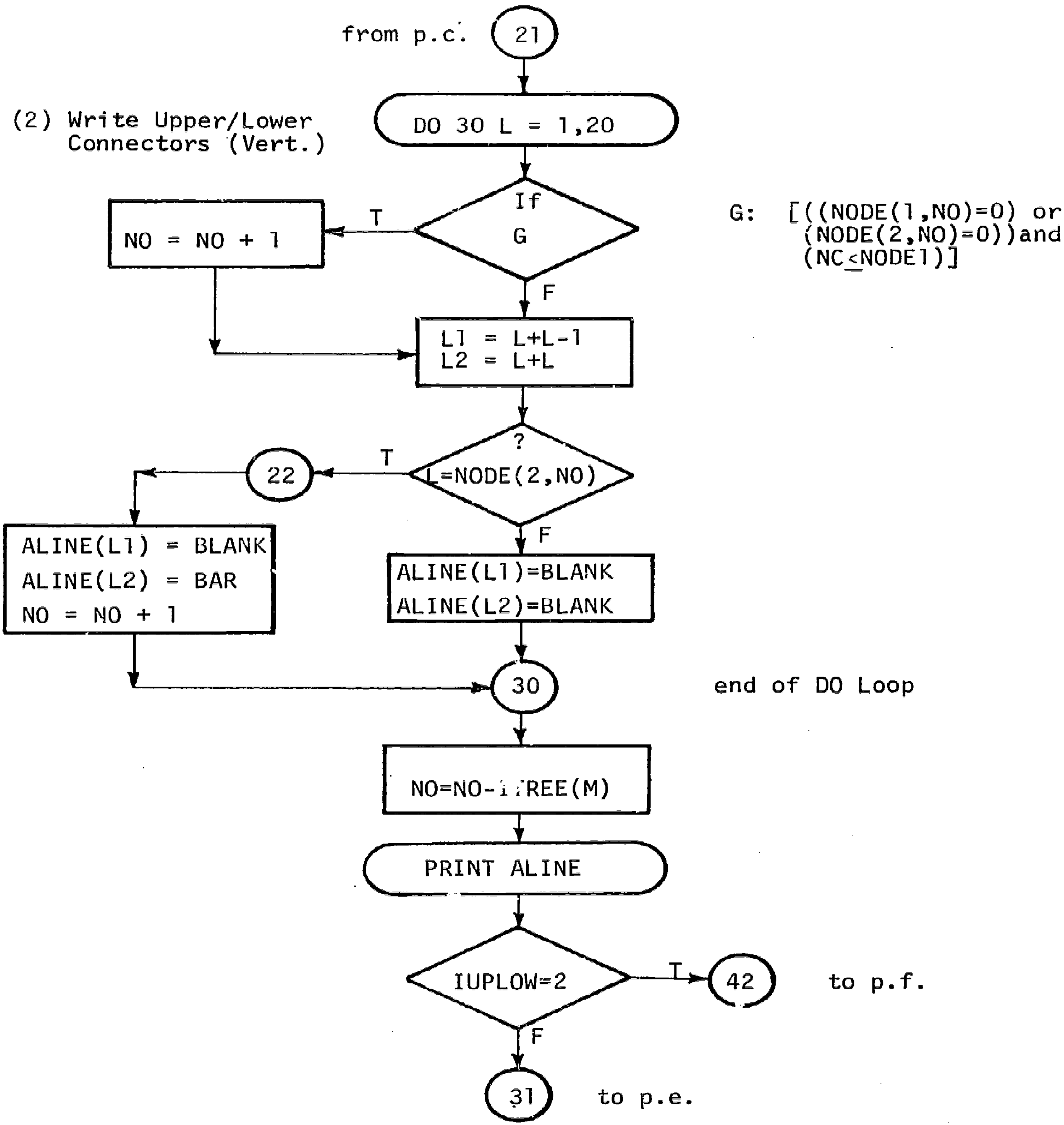


Figure 4.5d: Subprogram DIAGRM

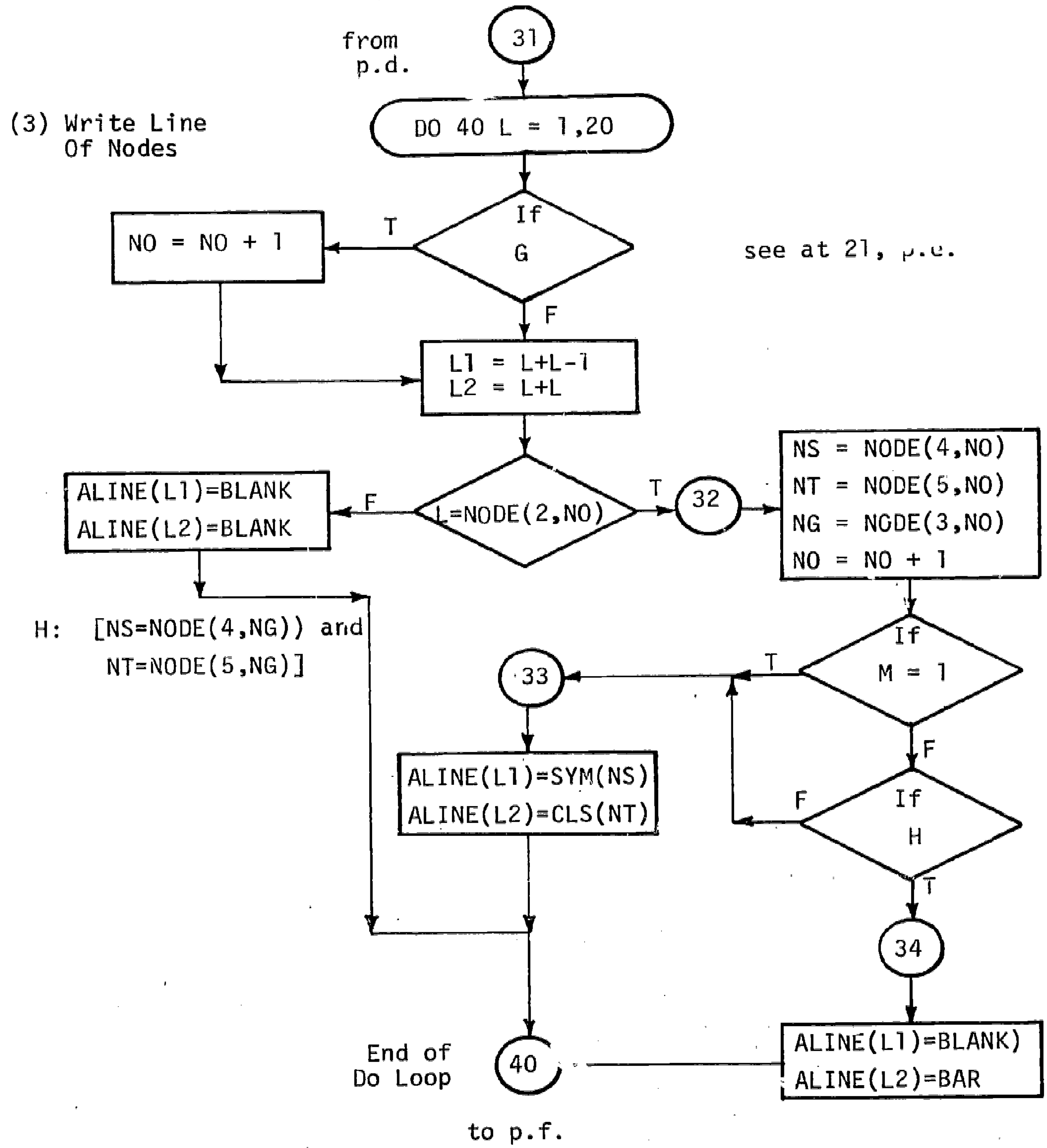


Figure 4.5e: Subprogram DIAGRM

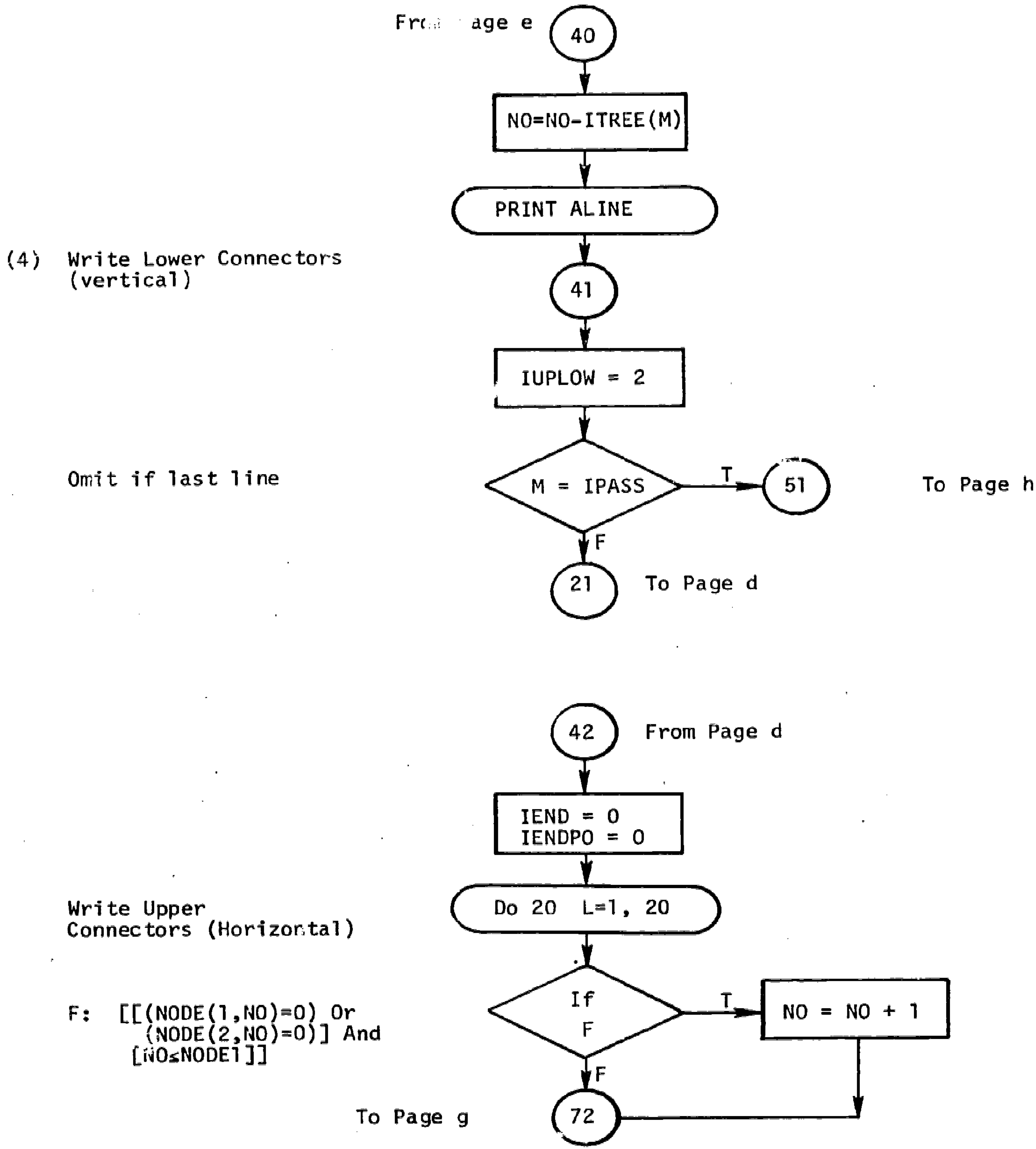


Figure 4.5f: Subprogram DIAGRM

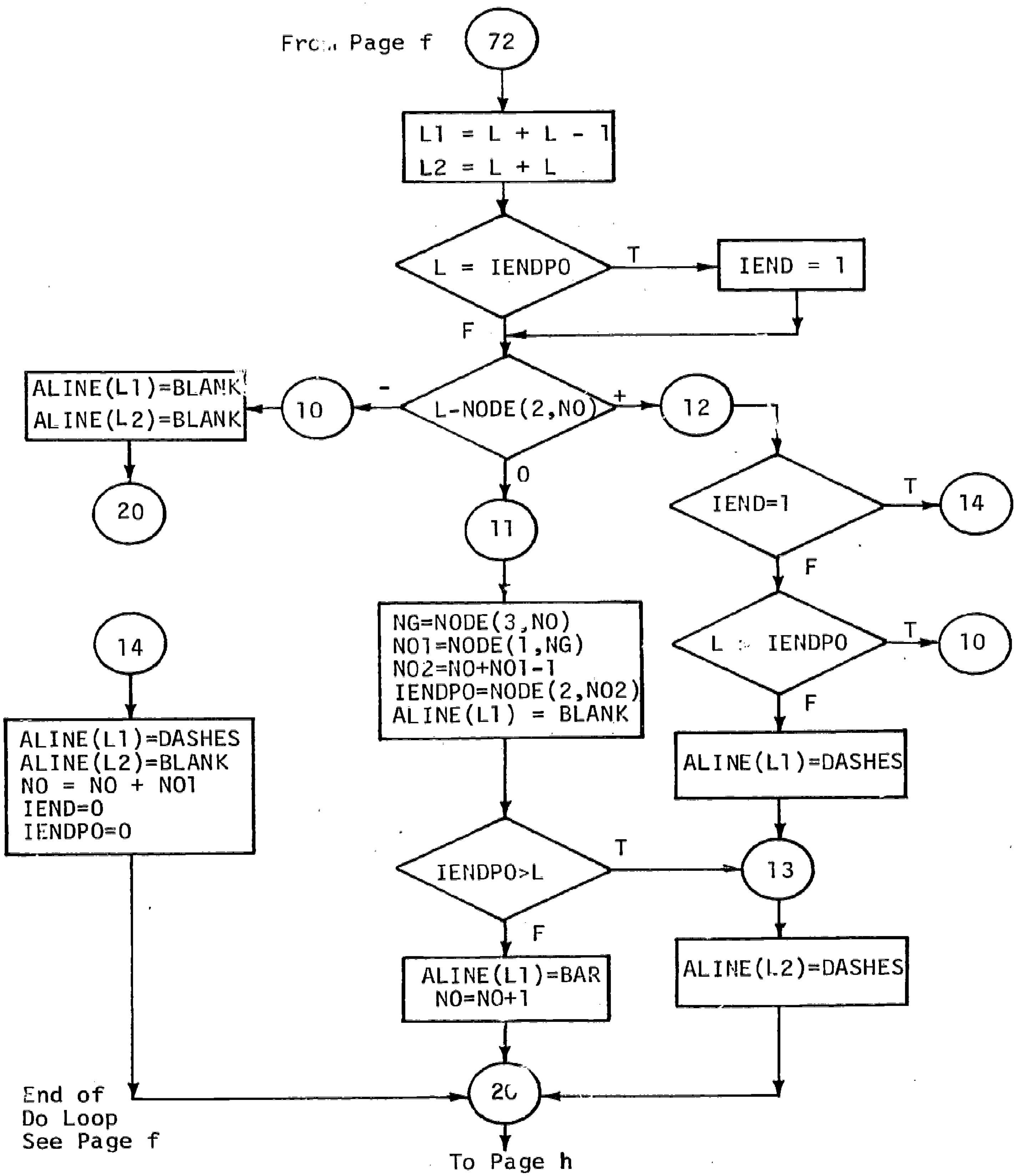


Figure 4.5g: Subprogram DIAGRM

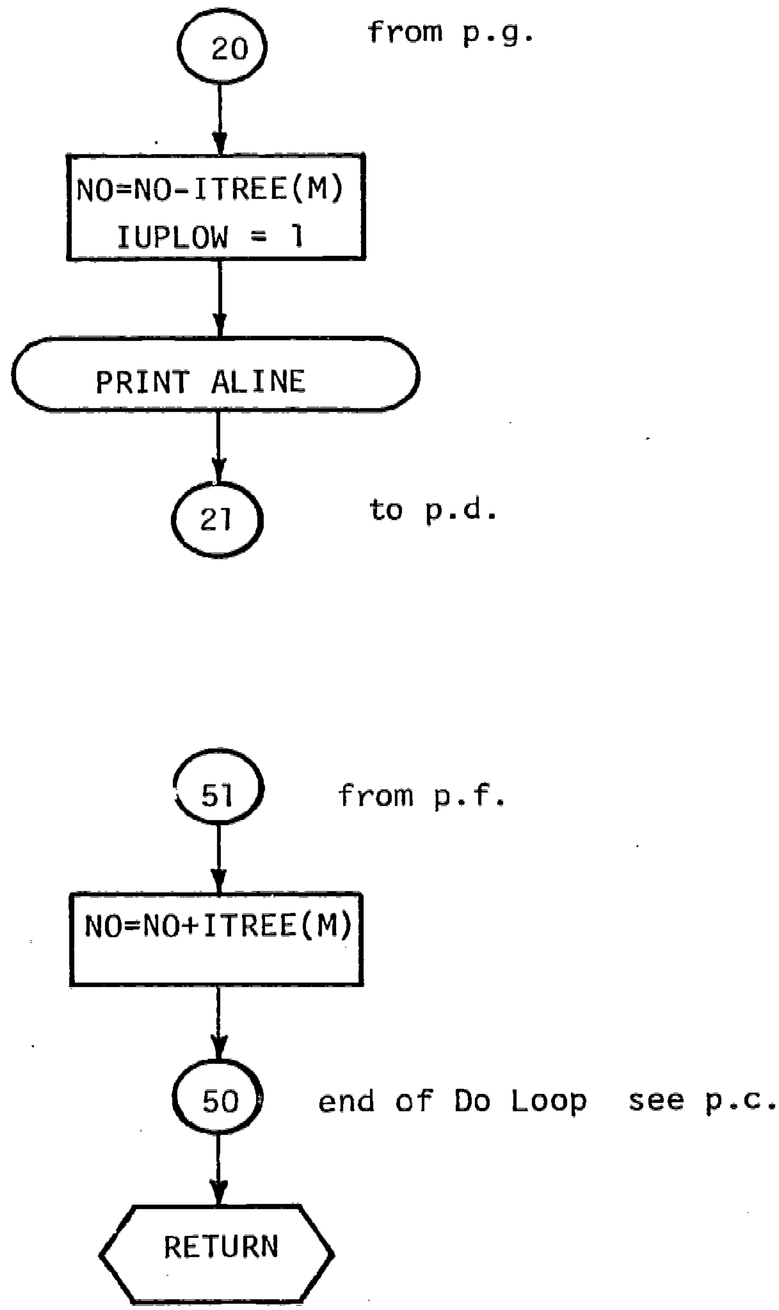


Figure 4.5h: Subprogram DIAGRM

4.3.1.4 Subprogram LIMIT

This subprogram (Figure 4.6) is called by Subprogram RERITE and by Subprogram SYMACH to test symbols for certain specified limitations on the values of their subscripts. The limitations are specified by Subscripts 27 and 28. Subscript 27 specifies the restraint type and Subscript 28 specifies the subscript number to which the restraint applies.

The restraint type refers to a row number IR in Restraint Matrix RESTRT(L,IR). The first number in this row, RESTRT(1, IR), tells how many restraint numbers are on the list; the remaining numbers in this row are limitations on the specified subscript. The logic of the limitations is as follows: the value of the specified subscript must be equal to one of the positive numbers on the list, or it must not be equal to the absolute value of the negative numbers on the list. If these restraints are not met the subprogram sets the logic monitor MATCH to "false."

4.3.1.5 Subprogram MACHER

This subprogram (Figure 4.7) is called by Subprogram PROPH1 and PROPH2 to compare two symbols in accordance with the criteria listed below. Subprograms PROPH1 and PROPH2 make predictions of the possible successful satisfaction of a given rule after one or two more passes on the present string of symbols. In order to do this, the subprograms select likely rules and put their appropriate symbols in registers (IS1 and IS2) which are used by MACHER to determine whether a symbol of a given rule (IS2) could be applied to a given symbol under consideration (IS1) in the process of predictions.

The criteria, which apply to the first 17 subscripts (excluding 3) are as follows:

(1) The symbol types must be the same and

(2) one of the following must be true for each subscript

   (a) the subscripts are equal, or

   (b) the subscript ℓ and the symbol in IS1 is not negated (test A), or

   (c) the subscript of one or the other of the symbols is an independent variable (test B), or

   (d) the subscript of the symbol in IS1 is a dependent variable and the subscript of the symbol in IS2 is a negative number (test C), or

   (e) the subscript of the symbol in IS1 is a fixed variable and the subscript of the symbol in IS2 is a dependent variable.

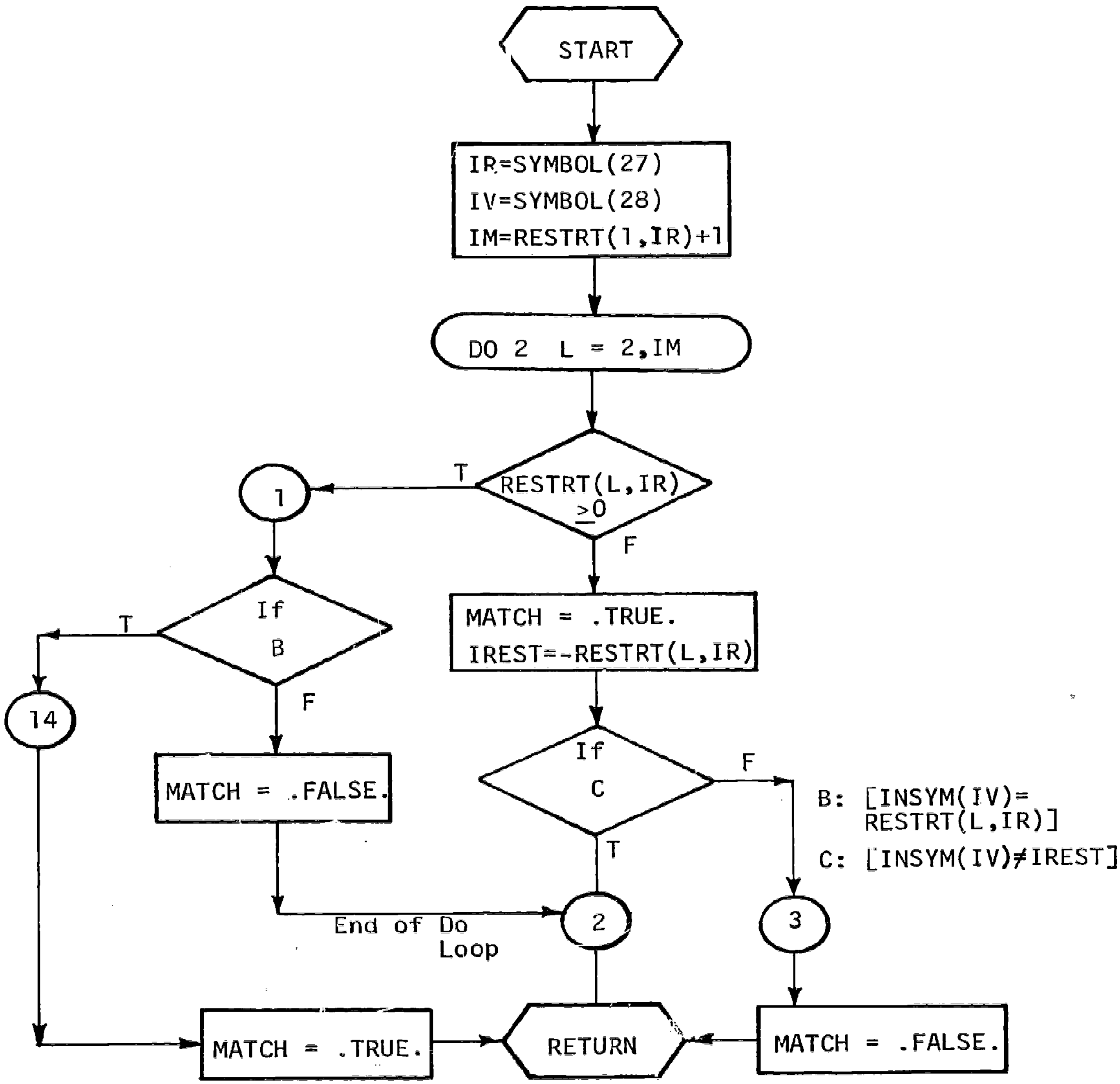


Figure 4.6: Subprogram LIMIT

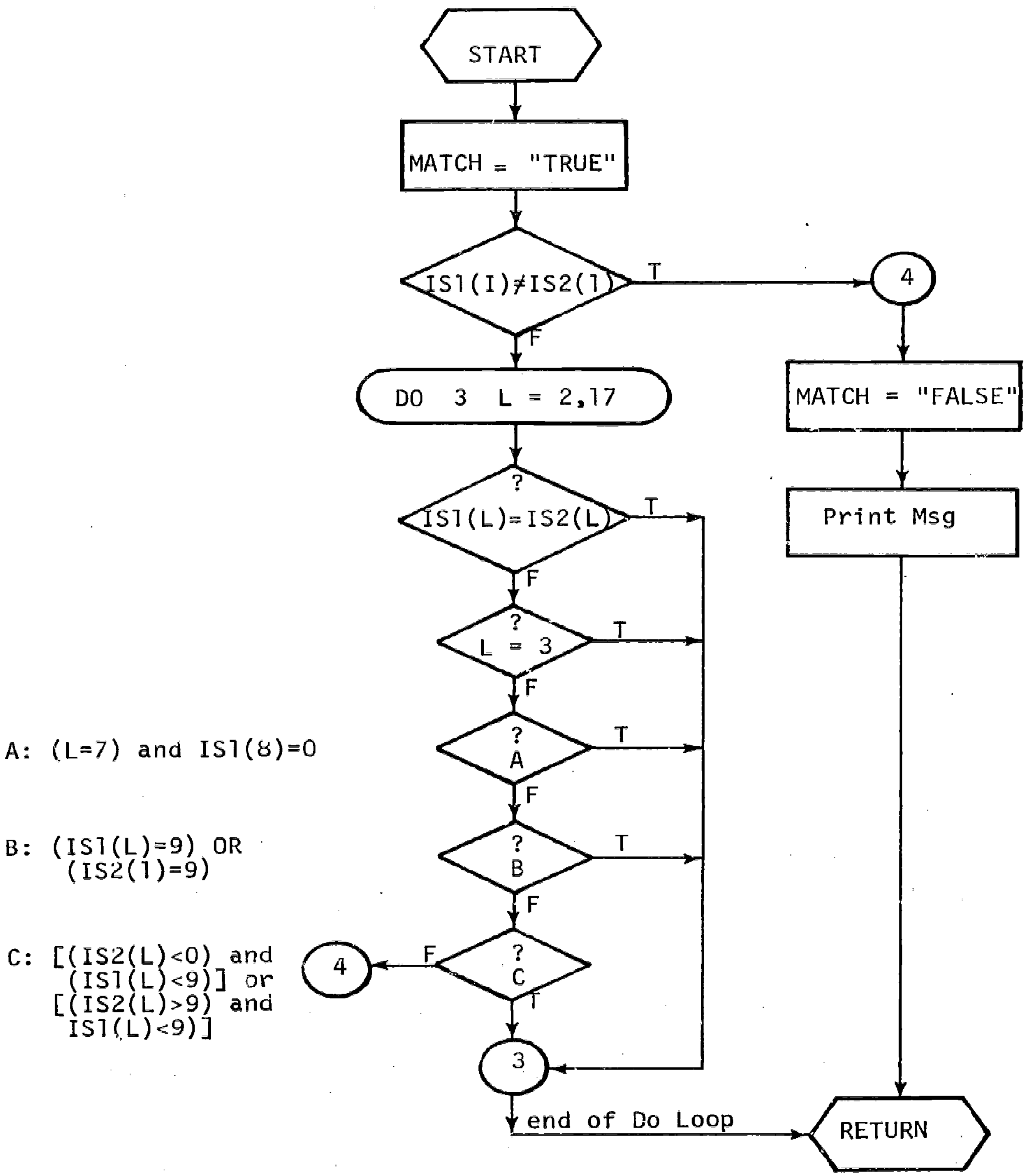


Figure 4.7: Subprogram MACHER

If the above criteria are met, the logical variable MATCH is set to "true", otherwise ͻ "false".

4.3.1.6 Subprogram OUTPUT

This subprogram (Figure 4.8) is called by the Main Program ANALYZ (step 22) to print out the data associated with each symbol in the string after each rerite pass of the grammar on a string of symbols. See Appendix C for an example of the resultant output from this subprogram.

4.3.1.7 Subprogram PARSE

This subprogram (Figure 4.9) is called by the main program ANALYZ (step 25) in order to print out a list of analysis statements about the sentence being analyzed. At this stage of the computation the analysis is complete and data defining the deep structure of the analysis is stored in matrix NODE, and array ITREE (see Section 4.3.1.3 for more detail). These data include the symbol number, symbol type and the index number of the governing symbol for each constituent of the analysis. For each unique constituent of the analysis, this subprogram assembles an analysis statement of the form

$$N_s + N_a$$

where N_s stands for a sequence of one or more symbol names in the form

$$\text{The } N_1 \text{ of the } N_2 \ldots \text{ of the } N_j$$

where N_1 is the name of the symbol about which the statement is made, N_2 is the name the symbol immediately governing N_1, and N_j is the name of the symbol immediately governing N_{j-1}. N_a is the analysis predicate that applies to N_1. Examples of the output of this subprogram are contained in Appendix D. The data for a given analysis statement is accumulated in array ACCUM. When the statement is complete, it is printed out and the array is cleared for the next statement.

The operations of the subprogram are as follows:

(1) For each pass produced in the analysis the following steps are performed:

(2) For each symbol recorded in the given pass, the following steps are performed:

(3) Retrieve the symbol number of the J-th symbol

(4) Accumulate the name of the J-th symbol from the table of symbol names SYML.

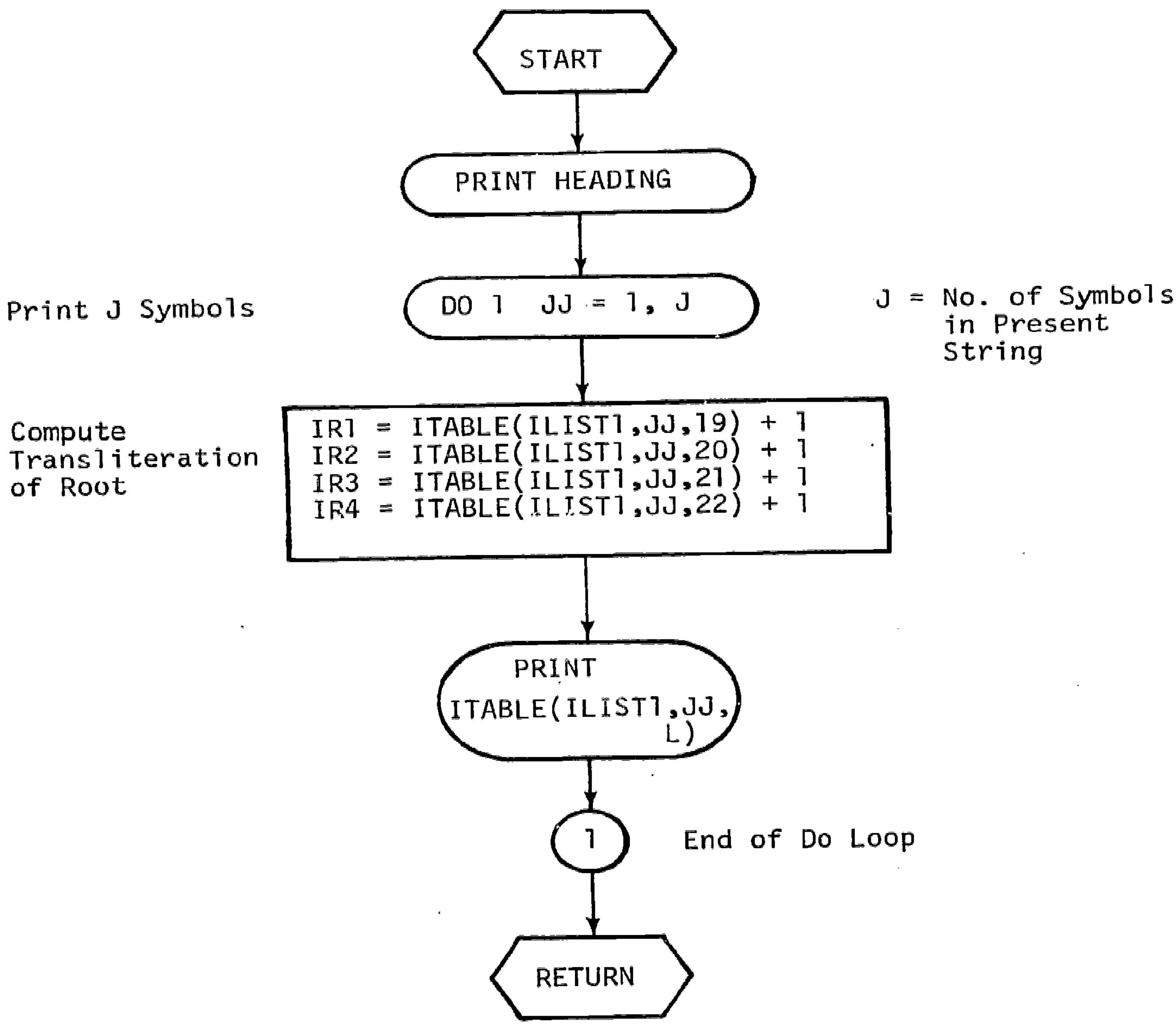


Figure 4.8: Subprogram OUTPUT

(5) Compute the index number of the governing node.

(6) If the symbol is not governed, skip to step (13).

(7) Compute the symbol number of the governing symbol.

(8) If this symbol has previously been treated omit this analysis statement and return back to step (3) for the next symbol on the list.

(9) If the governing symbol is the same as the governed symbol, the information is redundant, skip to step (5).

(10) Accumulate the words "of the" in ACCUM followed by the name of the governing symbol.

(11) Compute the index number of the symbol that governs the present governing symbol.

(12) If the symbol is not governed skip to step (13), otherwise return to step (7).

(13) If the first symbol was a terminal symbol, skip to step (16).

(14) Compute predicate index number.

(15) Print out the content of ACCUM followed by the given predicate from AMSG, and return to step (3) for the next symbol.

(16) Print out the content of ACCUM followed by the word "IS" followed by the English equivalent of the first symbol.

(17) For the subscripts $\underline{f}$, $\underline{k}$, $\underline{b}$, $\underline{c}$, $\underline{\ell}$, $\underline{y}$, $\underline{d}$, $\underline{n}$, $\underline{g}$, $\underline{p}$, $\underline{r}$, $\underline{a}$, $\underline{v}$, $\underline{i}$, and $\underline{t}$, accumulate (in ACCUM) the semantic values of those subscripts that pertain to the first symbol.

(18) Print out the semantic values accumulated in ACCUM and return to step (3) for the next symbol.

4.3.1.8 Subprogram PROPH1

This subprogram (Figure 4.10) is called by Subprogram RERITE (step 12) in order to predict the possibility of satisfying a given grammar rule if its application were delayed one or two passes. This procedure is used in order to avoid the premature abandonment of a rule which is not being satisfied. Thus in the process of analysis, when a given symbol does not satisfy the requirements of the rule under consideration, before a new rule is requested, this subprogram is called to answer the question "will the requirements of this rule be met if one (or two) other rules are applied first, and if so what is the first rule that should be applied?" If a favorable prediction is made, the analysis is required to take the predicted route before abandoning the rule under consideration. The following operations are performed by this subprogram:

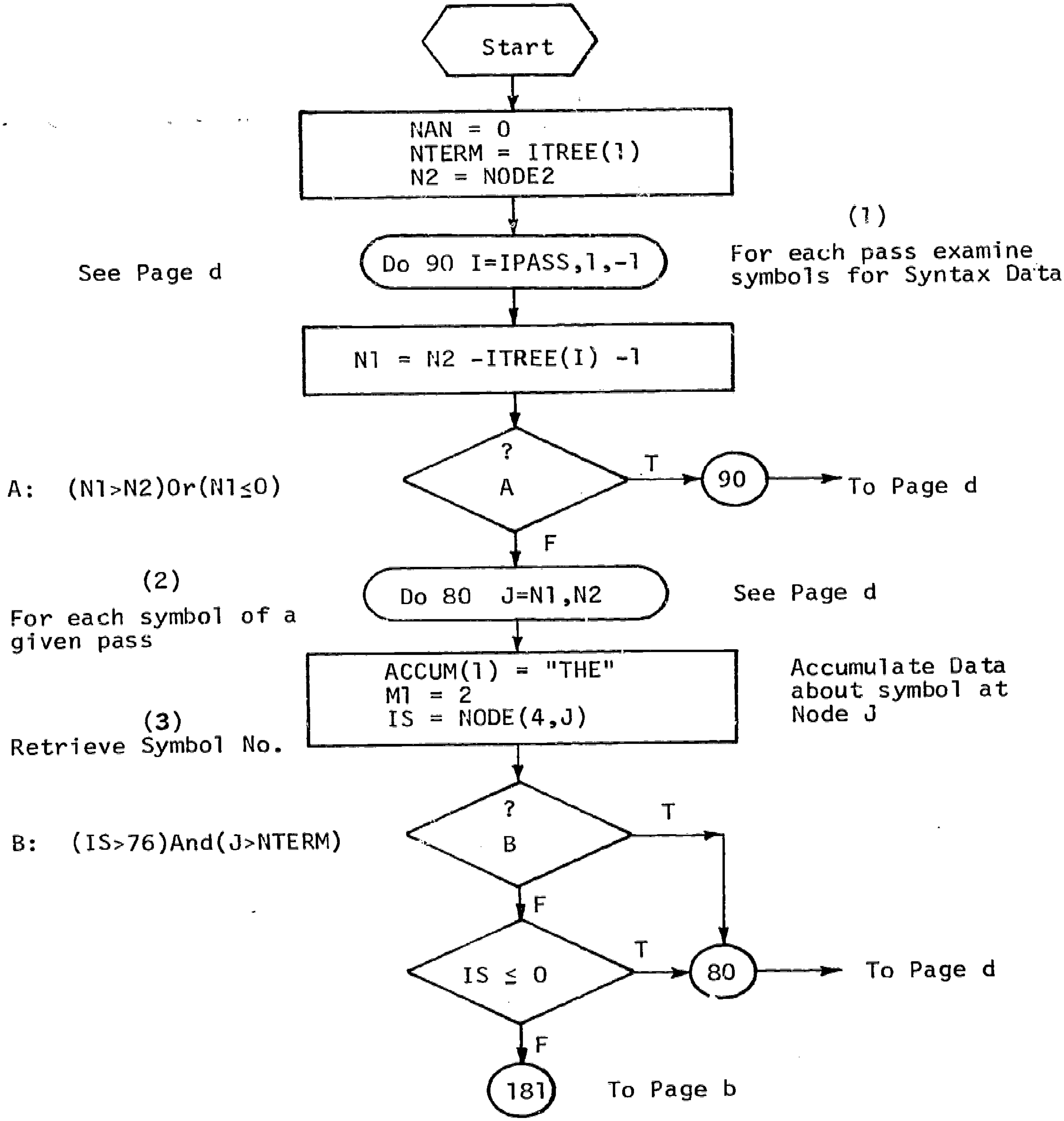


Figure 4.9a: Subprogram PARSE

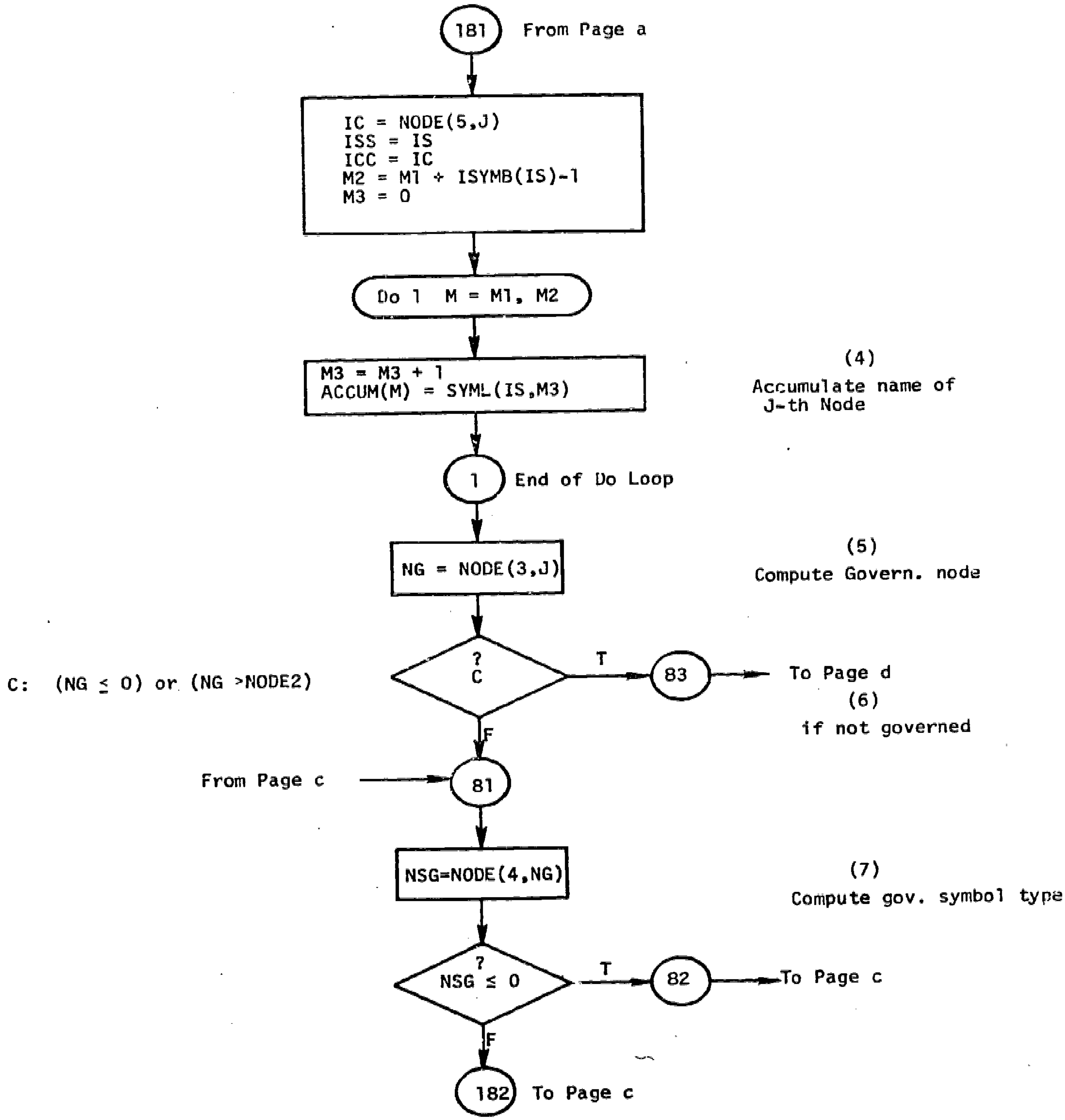


Figure 4.9b: Subprogram PARSE

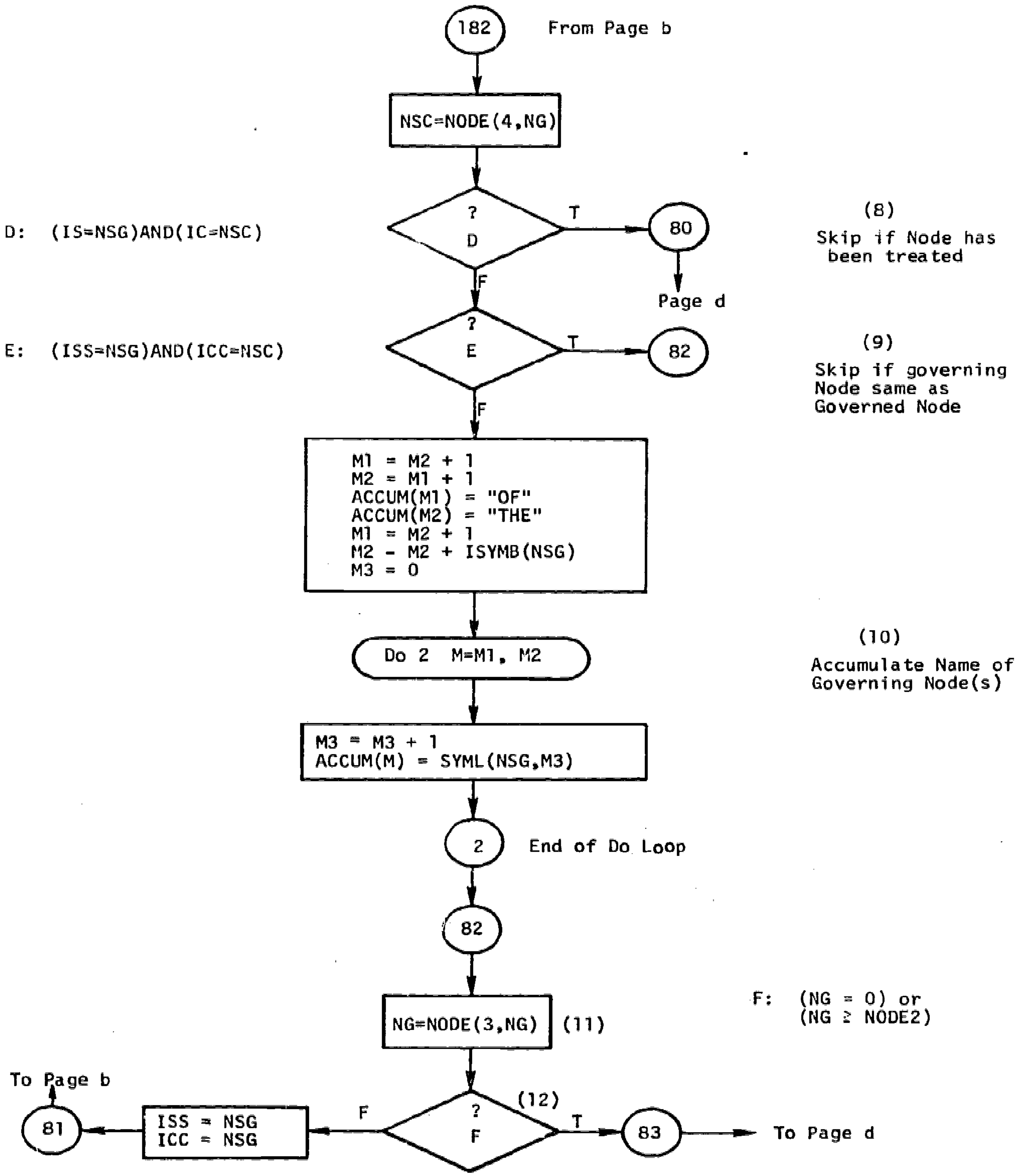


Figure 4.9c: Subprogram PARSE

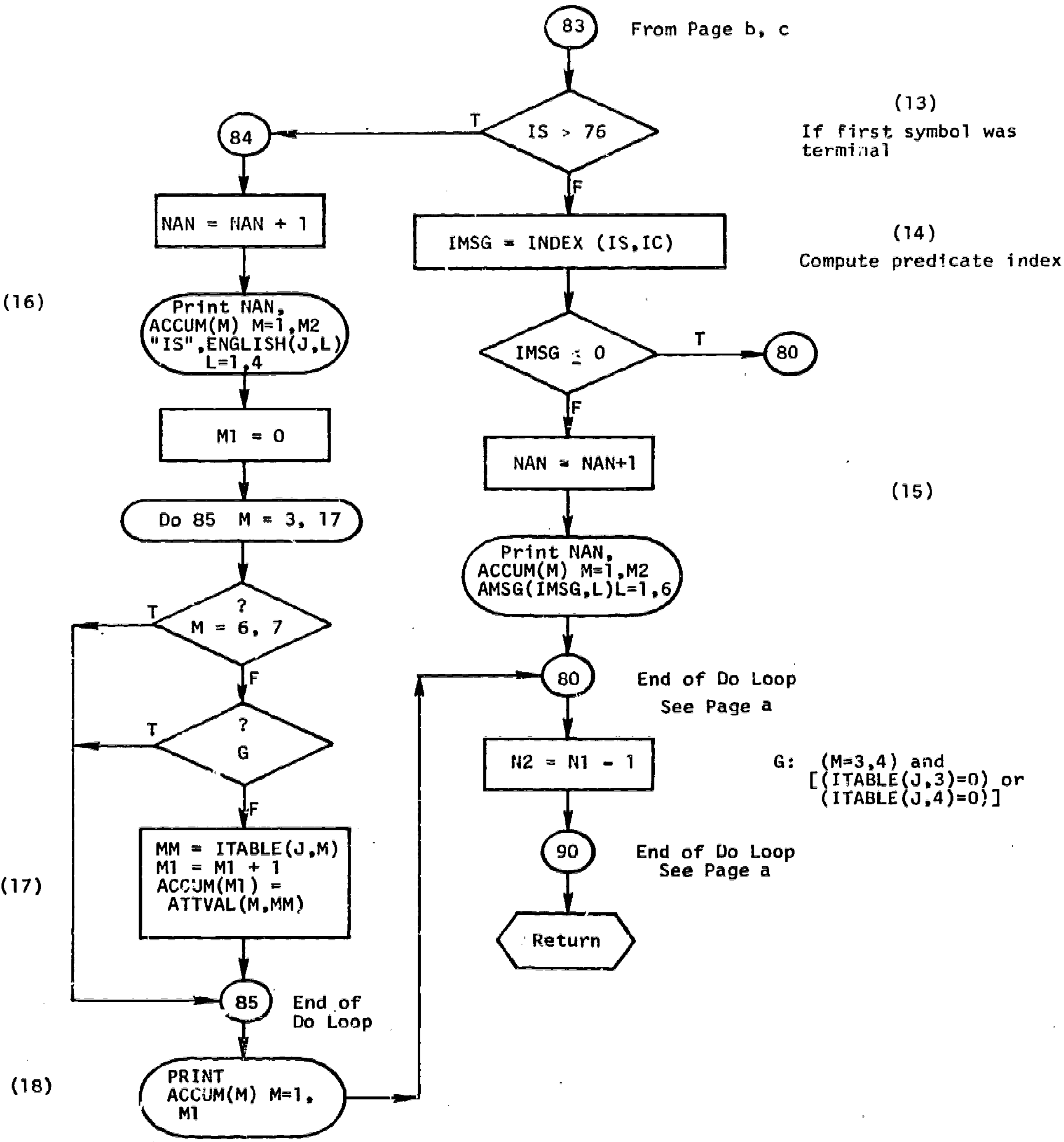


Figure 4.9d: Subprogram PARSE

(1) A test is made to determine whether the present symbol under consideration (stored in register SYMIN) has failed to meet the requirements of the given rule for other reasons. This is true if present symbol matches the rule element as to symbol type and symbol class. If this is the case, skip to step (11).

(2) Compute the address of those prediction rules that apply to the symbol in SYMIN, by computing

(a) the starting rule number (IR1) and

(b) the ending rule number (IR3)

If there are no prediction rules, skip to step (12).

(3) Compute the address (IR2) of the right hand member of the present prediction rule on the list.

(4) Put the current symbol under consideration in register IS1, the IR1-th rule element in register IS2, the present rule element (defined by IRULE1) in register IS3, and the right hand element of the prediction rule in IS4.

(5) Using Subprogram MACHER, determine that the symbols in IS1 and IS2 match, if not skip to step (7).

(6) Using subprogram MACHER, determine that the symbols in IS3 and IS4 match, if not skip to step (9).

(7) Advance the index (IR1) of the prediction rule element and of the present symbol (IN)

(8) If IR1 $\geq$ IR2, the symbols in the string will satisfy the prediction rule, skip to step (14), otherwise return to step (4) and repeat from there for the new set of indices.

(9) Using subprogram PROPH2, predict satisfaction of the present "prediction" rule at the next pass, if prediction is favorable (ID=4) skip to step (14).

(10) Reset ID to zero and compute the address (IR1) of the next available prediction rule; if there are no more (IR1>IR3) skip to step (13), otherwise return to step (3) for processing the next prediction rule.

(11) No future satisfaction is predicted because the present rule failed to meet the requirements of subprogram SYMACH or LIMIT (previously applied); set ID=1 and return.

(12) There are no rules that apply to the present symbol under consideration, therefore no future satisfaction is predicted; set ID=2 and return.

(13) The prediction rules were exhausted and none predicted future satisfaction, set ID=3 and return.

(14) Satisfaction is precicted if the rule at address IR4 is applied first; set ID=4, IR=IR4, and IM=IR3, and return.

4.3.1.9 Subprogram PROPH2

This subprogram (Figure 4.11) is called by subprogram PROPH1 (step 9) in order to predict the possibility of satisfying a given grammar rule if its application were delayed two passes (see Section 4.3.1.8 for further discussion). The following operations are performed by this subprogram:

(1) Compute the address of those prediction rules that apply to rule element defined by input argument IR, by computing

  (a) the starting rule number (IR1) and

  (b) the ending rule number (IR3)

  If there are no rules, skip to step (7).

(2) Compute the address (IR2) of the right hand member of the present prediction rule on the list.

(3) Put the IR-th rule element in register IS1, the IR1-th element in register IR2, the present rule element (defined by IRULE1) in register IS3, and the IR2-th rule element in register IS4.

(4) Using subprogram MACHER determine that the symbols in IS1 and IS2 match, if not skip to step (6).

(5) Using subprogram MACHER determine that the symbols in IS3 and IS4 match, if not skip to step (6), otherwise to step (9).

(6) Compute the address (IR1) of the next available prediction rule; if there are no more (IR1>IR3), skip to step (8) otherwise return to step (2) for processing the next prediction rule.

(7) There are no rules that apply to the symbol under consideration, therefore, no future satisfaction is predicted; set ID=2 and return.

(8) The prediction rules were exhausted and none predicted future satisfaction; set ID=3 and return.

(9) Satisfaction is predicted by the rule just tested; set ID=4 and return.

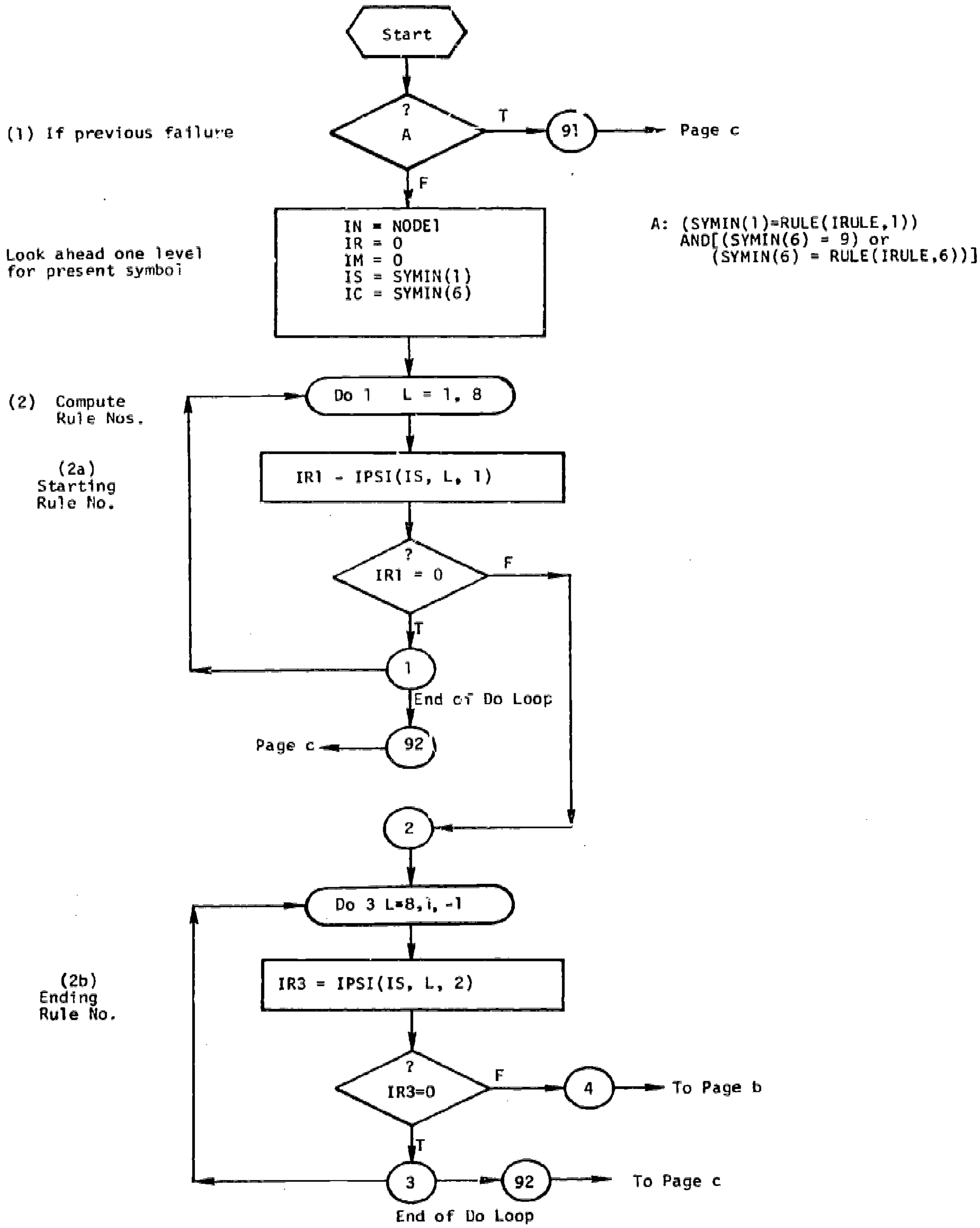


Figure 4.10a: Subprogram PROPH1

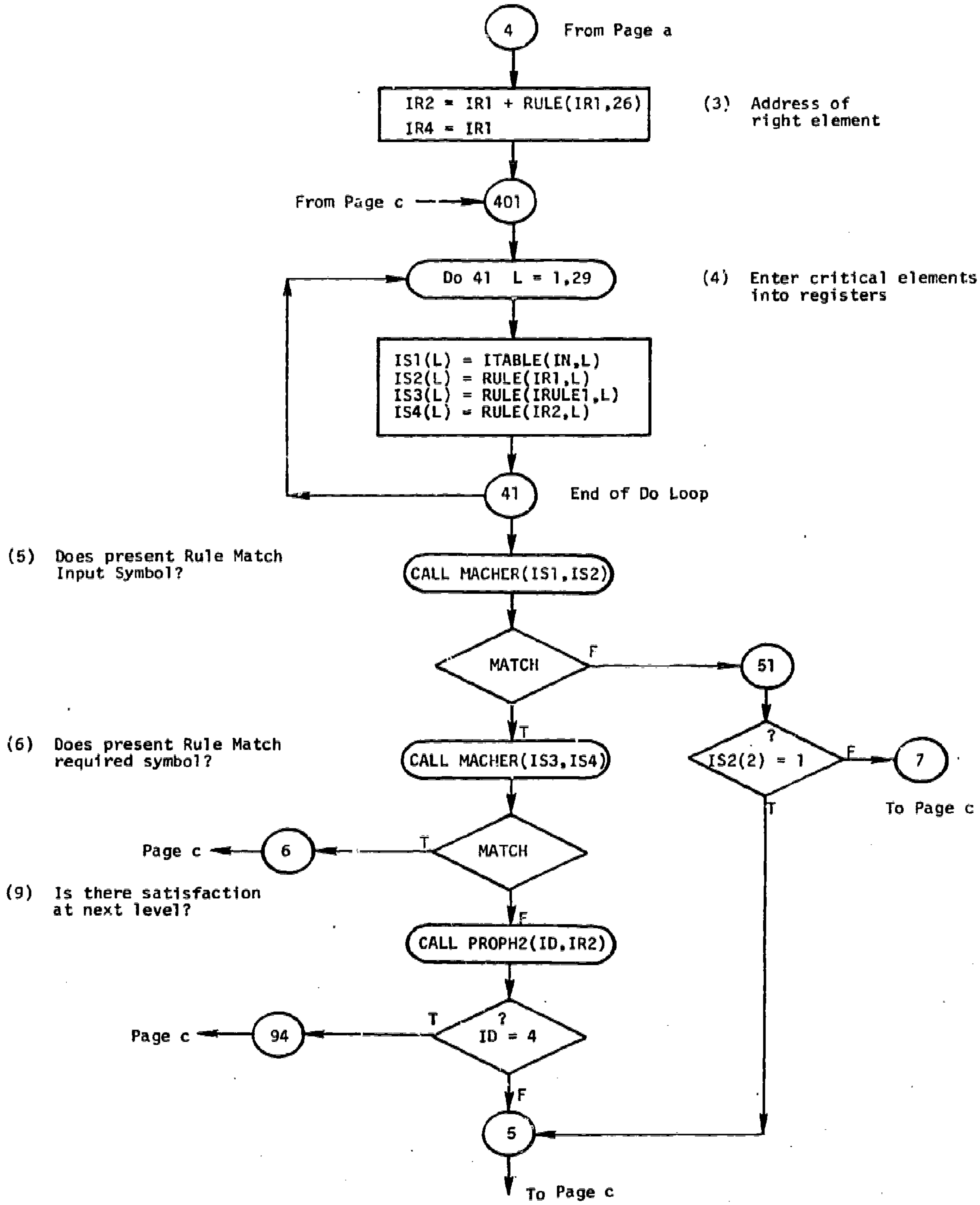


Figure 4.10b: Subprogram PROPH1

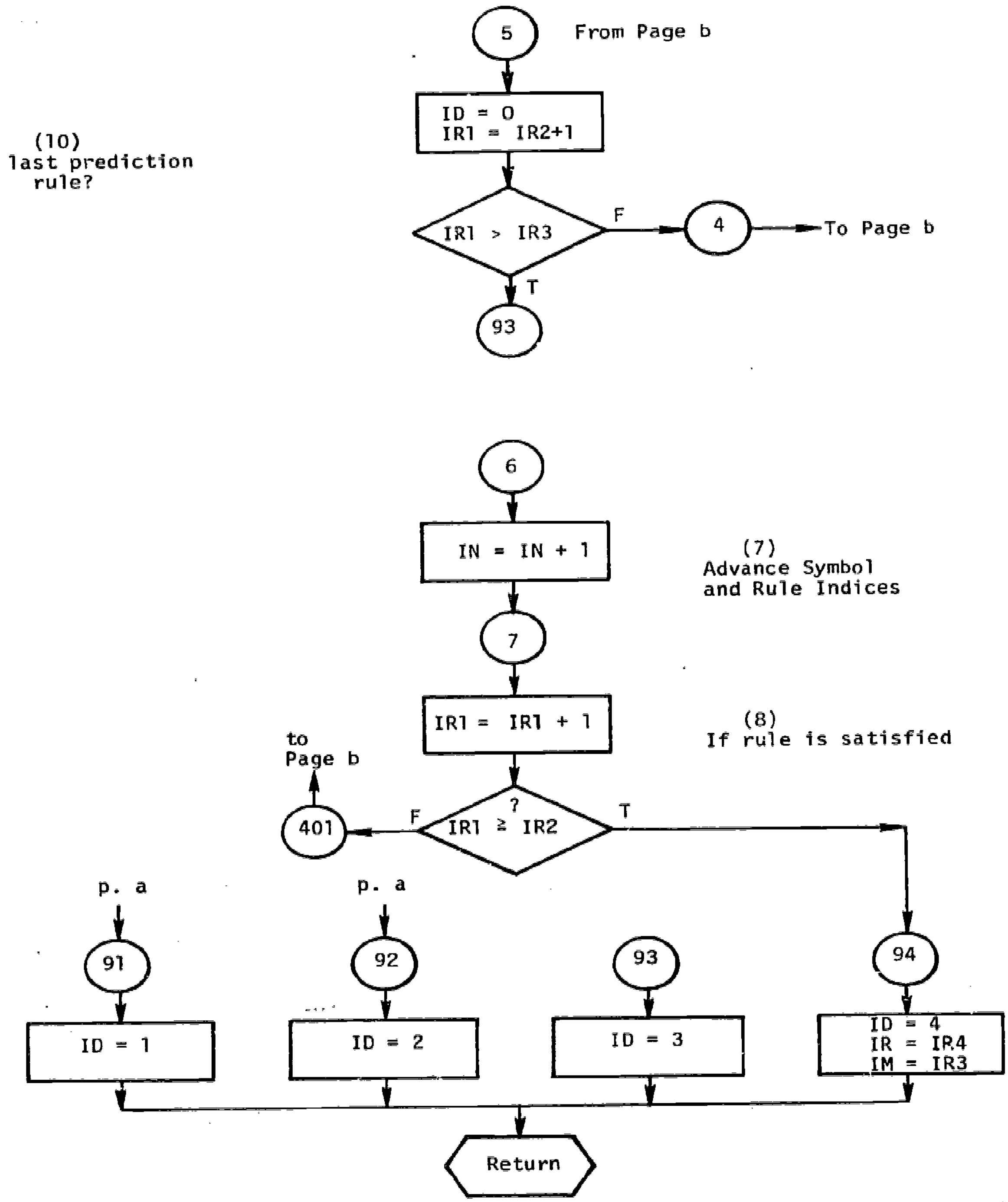


Figure 4.10c: Subprogram PROPH1

(1) Compute Rule No.

(1a)
Starting No.

(1b)
Ending No.

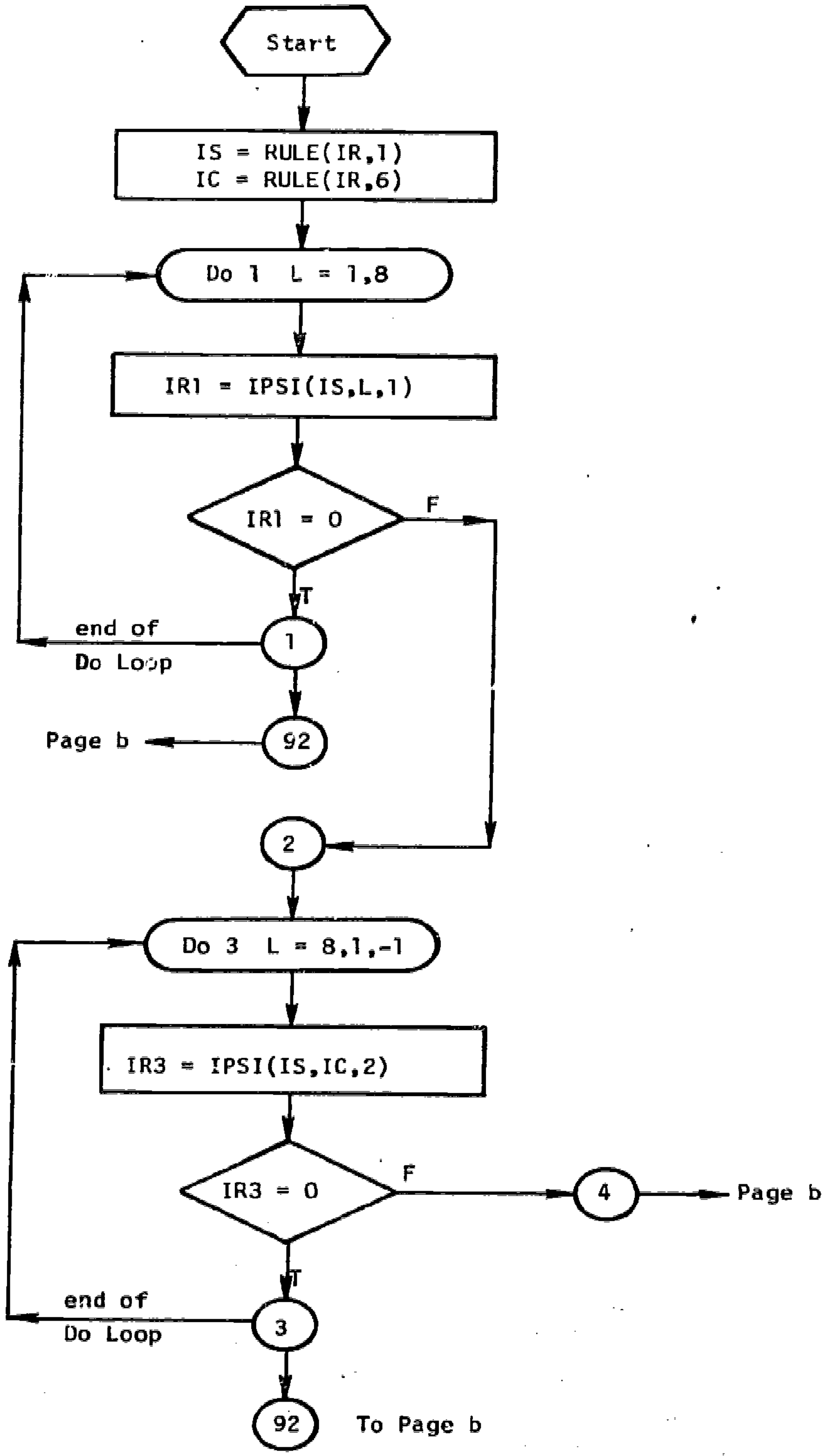


Figure 4.11a: Subprogram PROPH2

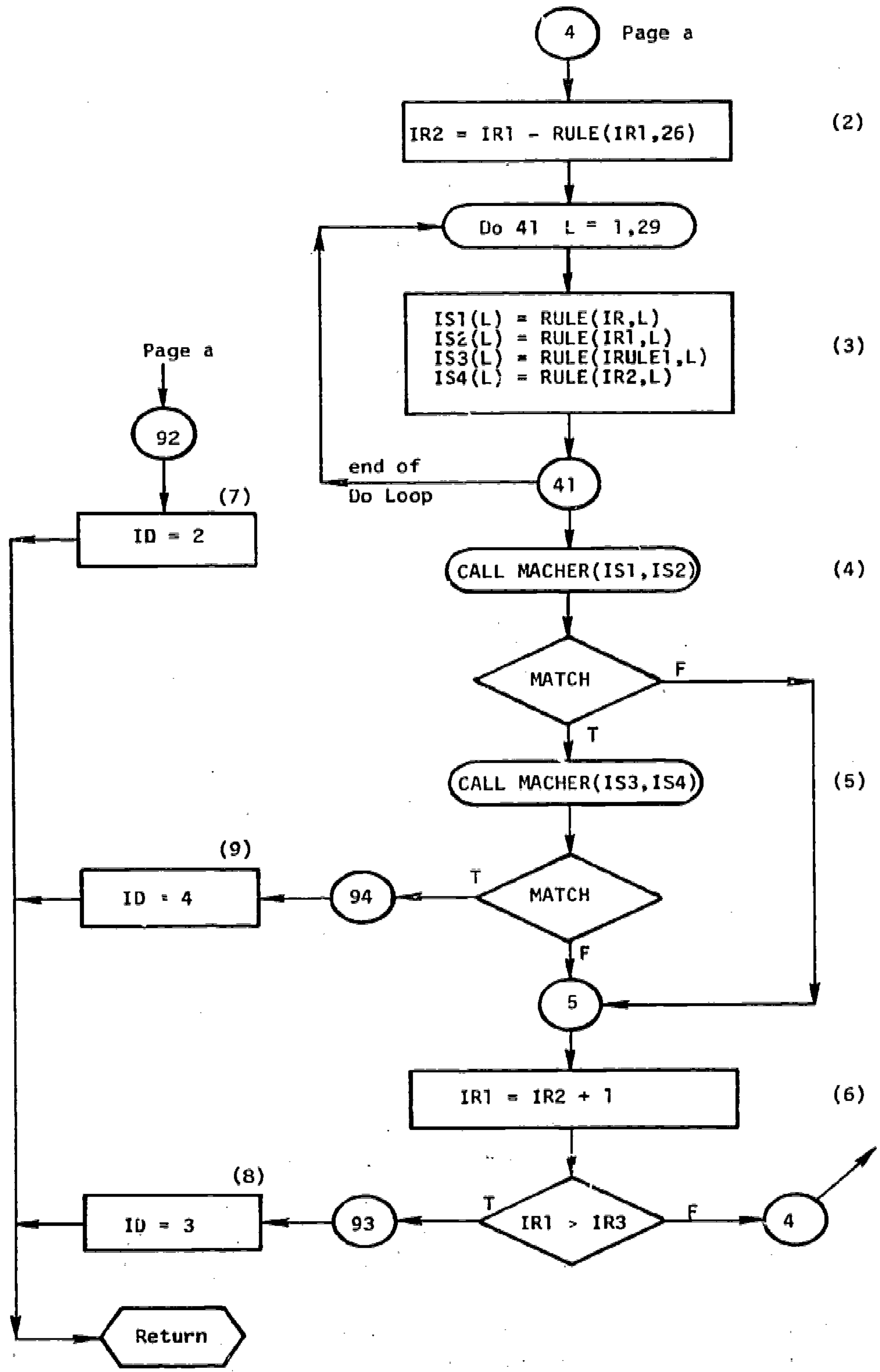


Figure 4.11b: Subprogram PROPH2

.3.1.10 Subprogram RERITE

This subprogram (Fig. 4.12) is called by the Main Program ANALYZ (step 13) to apply the appropriate replacement rule to given symbols and write the replacement symbol on the rerite symbol list.

The subprogram performs the following operations:

(1) Initialize certain indices

(2) Examine the present value of symbol counting indix I, if it is greater than the number of symbols in the present pass (IMAX) return, otherwise continue.

(3) Increment indices I, LIM, and NODE1, and place the NODE1-th symbol in register SYMIN.

(4) Examine the value of index J, if J≠0 skip to step (9), otherwise continue.

(5) Using subprogram RULENO, compute the indices of the rules that apply to the present symbol under consideration; store these values in matrix NODE for the symbol under consideration; and compute the index of the right hand element of the first rule.

(6) If there is only one rule, clear any rule numbers stored in matrix NODE for the present symbol.

(7) If there is no applicable rule skip to step (24), otherwise continue.

(8) Initialize certain indices in preparation for checking symbols in the string against the left hand elements of the present rule, place the right hand element of the rule in register SYMBOL, and skip to step (10).

(9) Clear any rule numbers stored in matrix NODE for the present symbol.

(10) Increment index J and IRULE1, and compare J with JMAX; if J>JMAX skip to step (18), otherwise continue.

(11) Using subprogram SYMACH, test to see if the symbol in register SYMIN matches the J-th symbol of the given rule; if they match skip to step (16), otherwise continue.

(12) At this point, the symbol under consideration has been found not to match the present rule, however before abandoning the rule, subprogram PROPH1 is used to predict possible future satisfaction of this rule if its application is delayed one or two passes; if the prediction is favorable (ID=4) skip to step (27), otherwise continue.

(13) If the present element of the rule is optional (test A), make certain bookkeeping adjustments and skip back to step (10) to compare the present symbol with the next element of the rule, otherwise continue.

(14) If the present rule involves compounding (J3≠0) skip to step (15), otherwise make certain bookkeeping adjustments associated with abandoning the present rule and return to step (2) for consideration of any other applicable rules.

(15) Make adjustments to certain indices associated with compounding and return to step (4).

(16) At this point, the symbol under consideration has been found to match the J-th element of the given rule, so using subprogram VARATT, the values of the dependent variables associated with the symbol are computed, and the governing mode for the symbol is recorded.

(17) If all the elements of the rule have been satisfied (J=JMAX) skip to step (19), otherwise return to step (2) for further consideration of the remaining elements of the rule.

(18) Reduce the value of indices NODE1 and I by one and continue.

(19) Using subprogram LIMIT check the right hand element of the present rule for any imposed restraints. If the restraints are not met return to step (14) to select a new rule, otherwise continue.

(20) If the present rule involves compounding (J2=1), Increment J3 by 1, transfer the symbol in register SYMBOL to register SYMIN, make bookkeeping adjustments and return to step (4), otherwise continue.

(21) Make adjustments to certain bookkeeping indices, and increment index NODE2 by one.

(22) If the rule involves compounding, make further bookkeeping adjustments.

(23) Write the right hand element of the rule with the computed values of its subscripts (as contained in register SYMBOL) as the NODE2-th symbol of the analysis, and skip to step (30).

(24) If the present rule involves compounding that is not yet complete (test C) return to step (1a) otherwise continue.

(25) If the present rule involves compounding which is now complete (J3≠0) return to step (21), otherwise continue.

(26) If the present rule does not involve compounding (J2=2) skip to step (27), otherwise set J1=0 and J2=2 and return to step (14a).

(27) Make bookkeeping adjustments to certain indices and con- inue.

(28) If a prediction of future satisfaction of the present rule has been made (ID=4), store the rule number (ne- gated) as a prediction for the next symbol of the derivation and store the number of the prediction rule as a prediction for the present symbol, and in either case continue.

(29) Make bookkeeping adjustments and copy the present symbol (index NODE1) on the next list (index NODE2) and continue.

(30) Initialize certain indices in preparation for the next rule and return to step (1a).

4.3.1.11 Subprogram RULENO

This subprogram (Figure 4.13) is called by subprogram RERITE (step 5) to compute the grammar rule number that applies to a given sym- bol. The grammar rules are stored in Matrix RULE, and the computed rule numbers are as follows:

IRULE = the number of the first element of the first rule that applies to the given symbol

IMAX = the number of the last element of the last rule that applies to a given symbol.

For simple (non-compounded) symbols the applicable rule numbers initially were computed in Main Program ANALYZ (step 4) and stored in rule catalogue IPSI. The data is stored so that for Symbol IS, Class IC, IRULE = IPSI(IS,IC,1) and IMAX = IPSI(IS,IC,2).

The subprogram computes the rule numbers according to the follow- ing hierarchy:

(a) Compound symbols

(b) Simple, non-negated symbols.

The following operations are performed:

(1) If the symbol under consideration is being presented for the first time (J1=0) continue, but if at least one rule has been previously tried (J1≠0) skip to step (18).

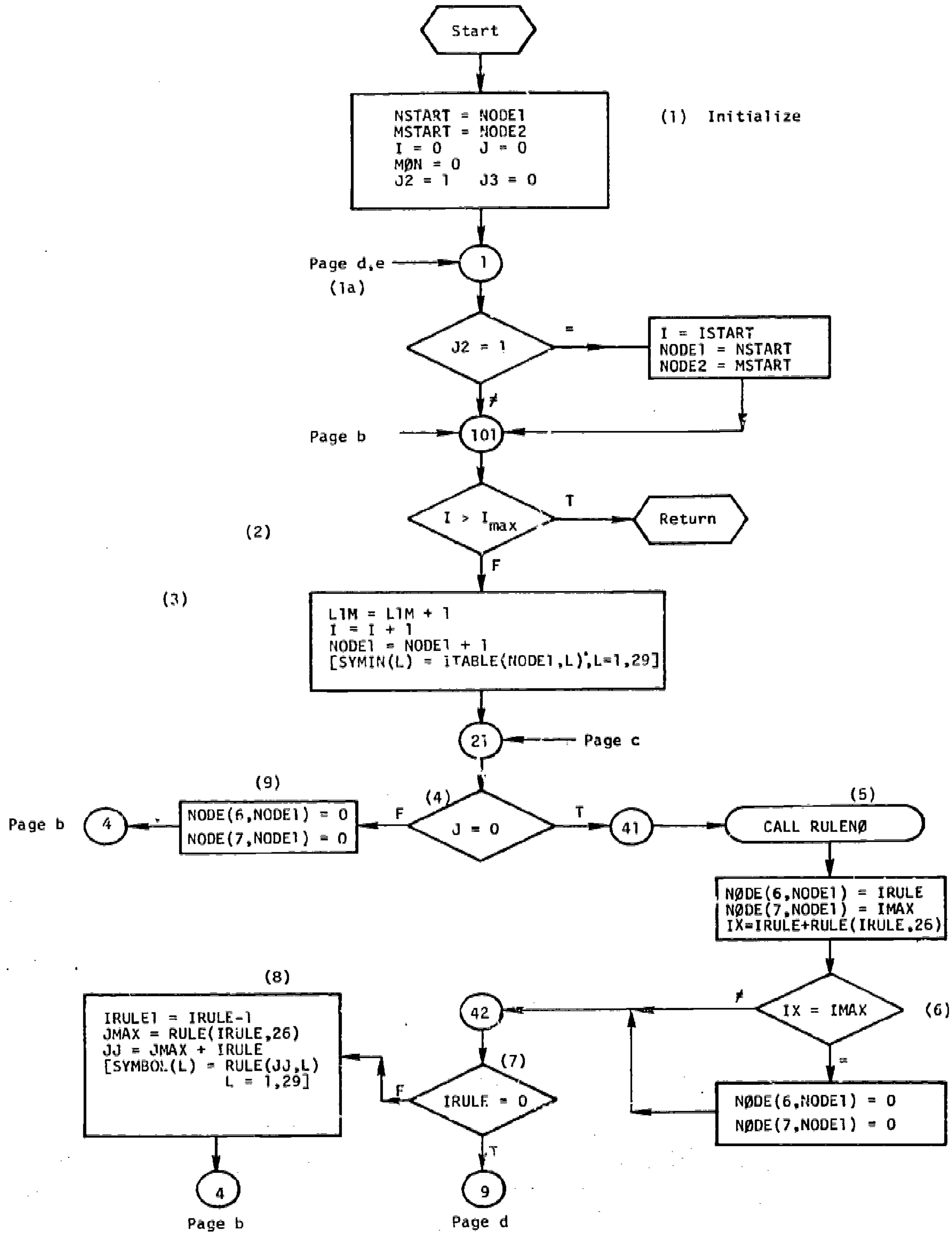


Figure 4.12a: Subprogram RERITE

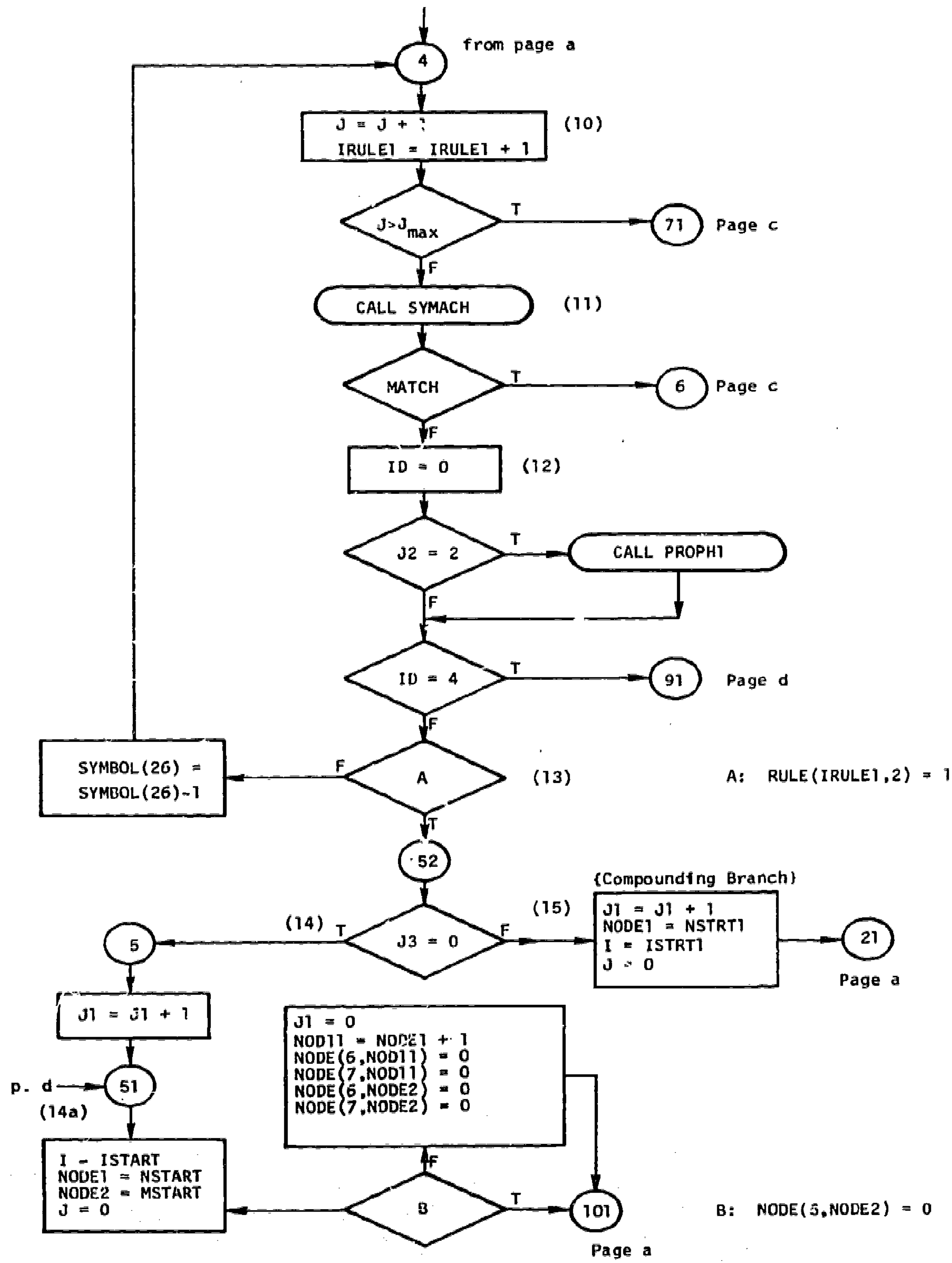


Figure 4.12b: Subprogram RERITE

75

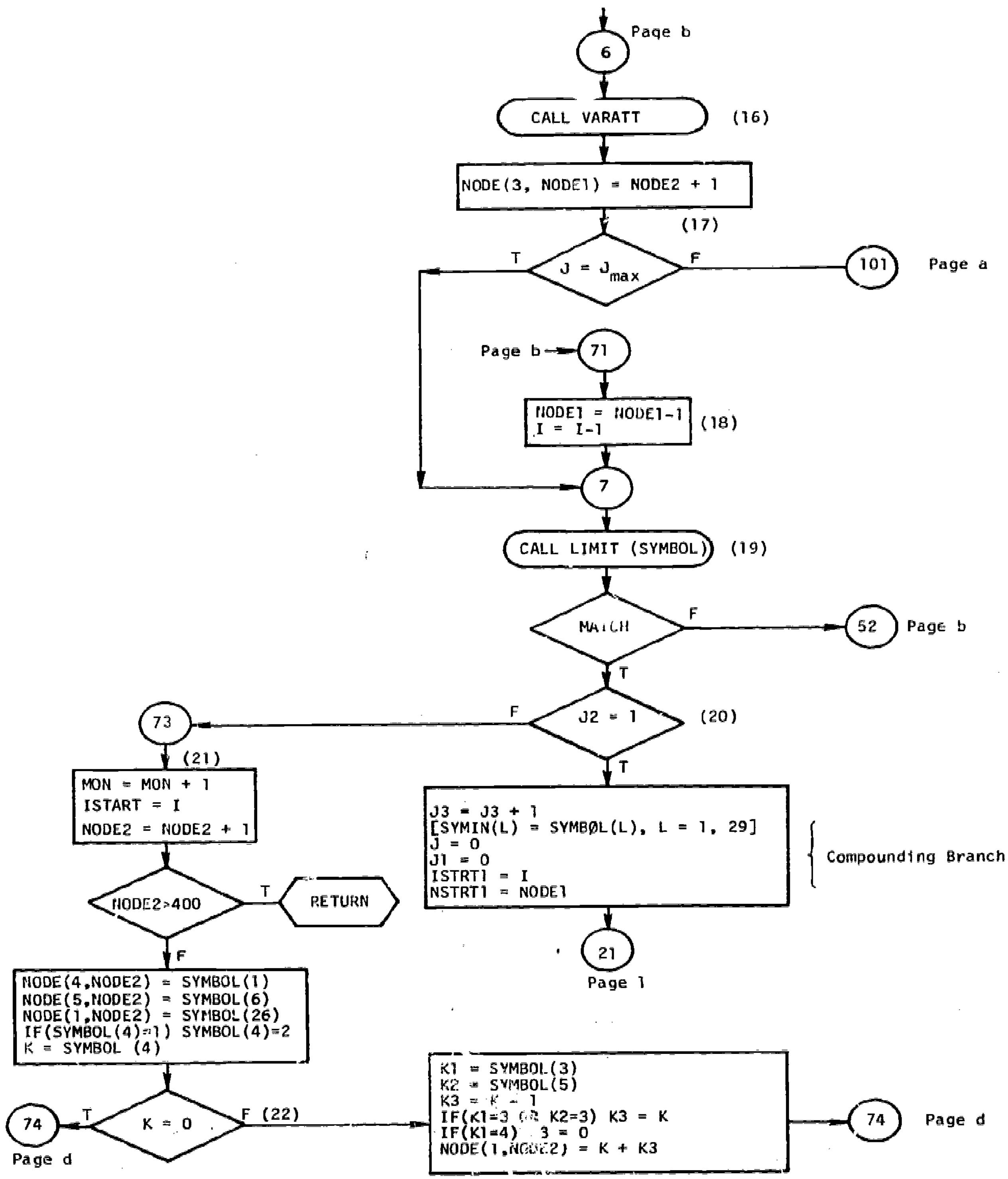


Figure 4.12c: Subprogram RERITE

76

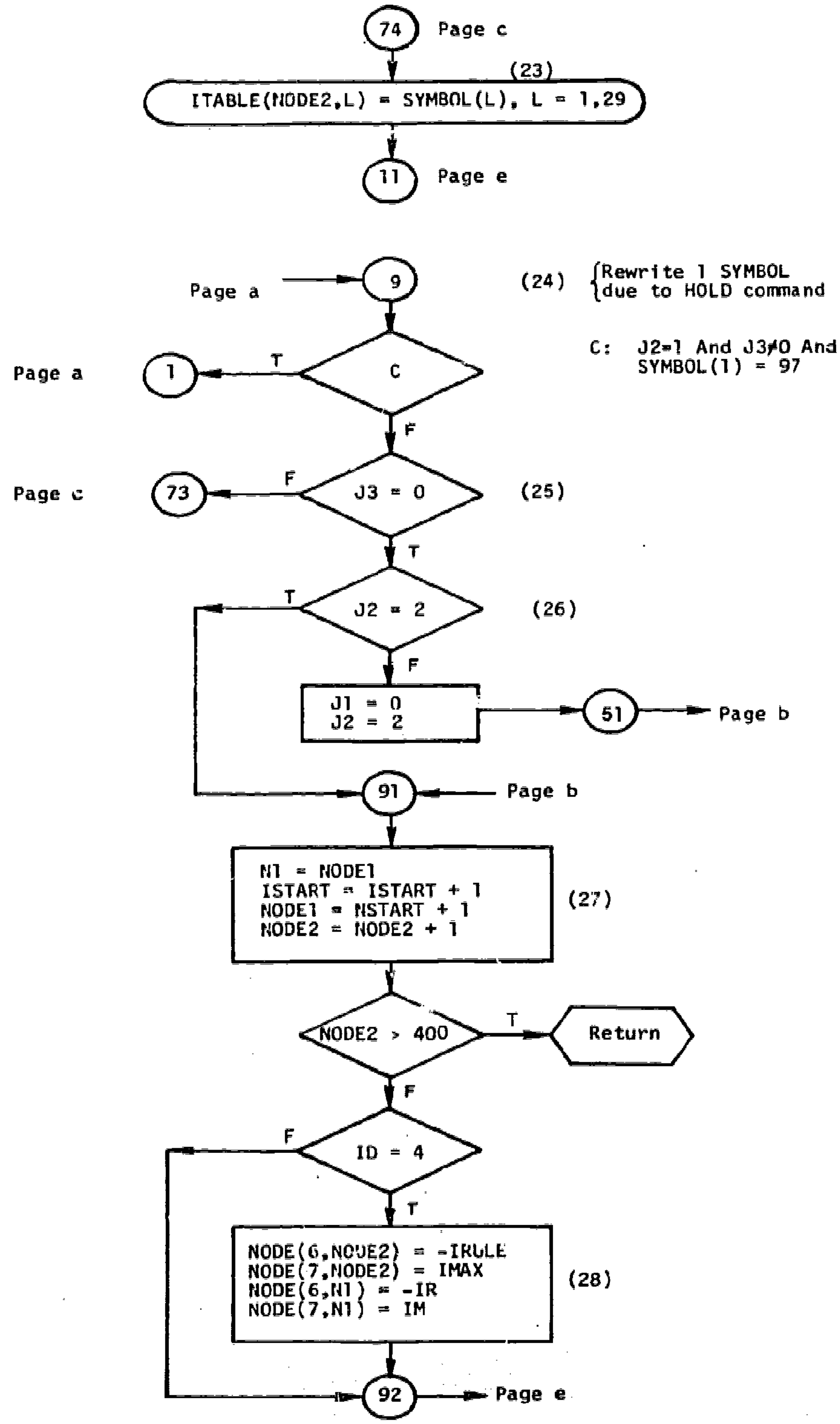


Figure 4.12d: Subprogram RERITE

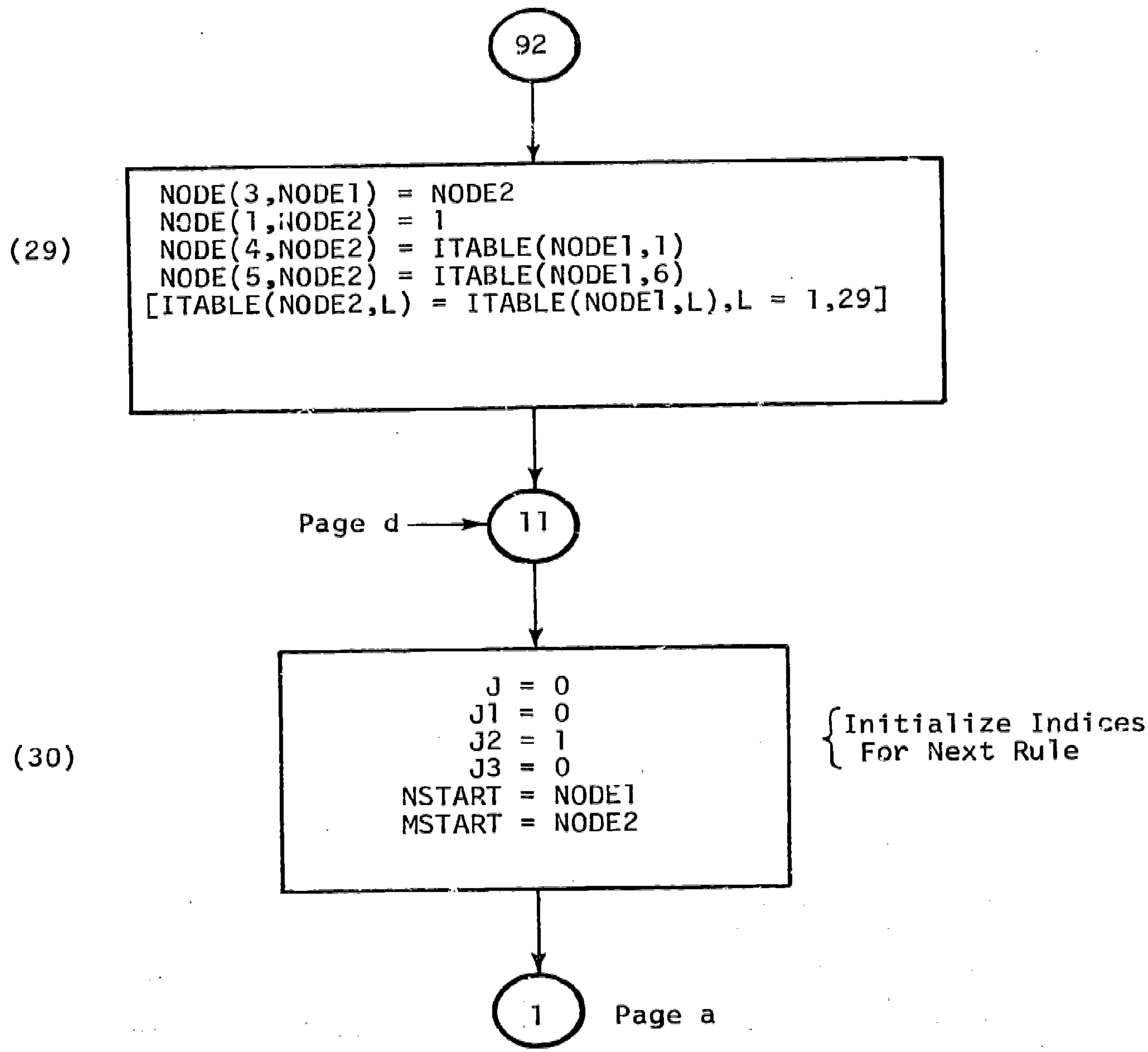


Figure 4.12e: Subprogram RERITE

(2) If universal rules should be considered (J2=1) continue, but if phrase rules should be considered (J2=2) skip to step (10).

(3) At this phase the symbol under consideration is being considered for possible application of universal rules, so certain indices are initialized.

(4) If this symbol and the next one are both of the same type (i.e., possible compounding pattern FF), set index IC=4 and skip to step (14), otherwise continue.

(5) If this symbol is followed by a conjunction and another symbol of the same type (i.e., possible compounding pattern F+C+F), skip to step (7) otherwise continue.

(6) If this symbol is followed by a comma, another symbol of the same type, and either a comma or a conjunction (i.e., possible compounding pattern F +, + F +,/C), continue, otherwise skip to step (8).

(7) At this point, it has been determined that the rules governing compounding apply; set IRULE=IPSI (97,1,1) the index of the first element of the first compounding rule for class 1; set IMAX=IPSI(97,3,2) the index of the last element of the last compounding rule for class 3; set Ic=1 and return.[9]

(8) At this point it has been determined that no compounding rules apply to this symbol; if, however, previous compounding rules have been successfully executed (J3≠0) on this symbol, set IRULE=0, IMAX=0 and return, otherwise continue.

(9) At this point it has been determined that compounding rules are not applicable to this symbol either now or in the past, set J2=2 and continue.

(10) At this point consideration is given to phrase rules (non-universal) that may apply to the symbol; set the index IS=SYMIN(1) the symbol type, and index IC=SYMIN(6) the symbol class and continue.

(11) If the present symbol has a prediction rule recorded for it [NODE (6,NODE1) ≠0], assign the indices of that rule to IRULE and IMAX, and return, otherwise continue.

(12) If the present symbol has an undefined class (IC=9) skip to step (17), otherwise continue.

(13) At this point, a test is made for the presence of rules that apply to all classes of the symbol, these rules

[9]The computation of this step is dependent on the "content" of the grammar.

are filed under class 1, so here the rules in class 1 are examined; IRULE is set to the index of the first class [IPSI (IS,1,1)], if no rule exists (IRULE=0) skip to step (14); if a rule exists and its class is undefined [RULE (IRULE,6)=9] skip to step (15), otherwise continue.

(14) At this point it has been determined that the grammar has no rules on the given symbol that apply to all classes; set IRULE to the index of the rule for the given class IC[IRULE=IPSI (IS, IC,1)] and continue.

(15) Set IMAX to the index of the last element of the last rule for the given class IC [IMAX=IPSI(IS,IC,2)], and continue.

(16) If no rule exists (IRULE=0) skip to step (17); however, if a rule exists, and if IMAX=0, it means that rules exist that govern all classes but none exists for the given class only, set IMAX=IPSI(IS,1,2), and return.

(17) At this point it has been determined that the given symbol has an undefined class, so all available rules on the symbol must be supplied; set IRULE=0 and (17a) find the index of the first element of the first rule that applies to lowest numbered class of the given symbol and if one exists, set IRULE equal to that index; and (17b) find the index of the last element of the last rule that applies to the highest numbered class of the given symbol, and if one exists, set IMAX equal to that index, and return.

(18) At this point it has been determined that a set of rules has already been located for the given symbol and that the most recent one examined was not satified; the following sequence of operations computes the index of the next available rule in the set; if the rules of the set are "universal" rules [IRULE > IPSI (97,1,1)], set index J2=2, and in either case continue.[10]

(19) Set the index IS (symbol type) equal to the type for the most recent rule (IS=RULE(IRULE,1)]; and set the index IC (symbol class) equal to the class of the most recent rule [IC=RULE(IRULE,6)], with the exception that if the rules are universal (IS=97), IC is set equal to the value of subscript f [IC=RULE(IRULE,3)], then continue.[10]

[10]The computation of this step is dependent on the "content" of the grammar.

(20) Set the value of IRULE to its present value plus the number of elements in the specified rule, plus 1 (the index of the next rule in the set), with the exception that if the most recent rule was the last rule (IRULE > IMAX), set both IRULE and IMAX to zero, and return.

4.3.1.12 Subprogram SYMACH

This subprogram (Figure 4.14) is called by Subprogram RERITE (step 11) to check that the "present rule symbol" matches the "present constituent" with respect to the criteria of the grammar. The subprogram performs the following functions:

(1) Initializes the logic monitor MATCH to "true".

(2) If the symbol types do not match or if the "present rule symbol" is a variable symbol, skip to step 14.

(3) For each subscript up to 17 perform steps 4 through 15.

(4) If subscript 3, and the "present rule symbol" is a variable symbol, skip to step 15.

(5) If subscript 7, and the "present rule symbol" is not a negative, skip to step 15.

(6) If the value of the given subscript of the "present rule symbol" is 9, that is, it is a independent variable, skip to step 15.

(7) If the value of the given subscript of the "present rule symbol" equals the value of the subscript of the "present constituent," skip to step 15.

(8) If the given subscript of the "present constituent" is an independent variable skip to step (15).

(9) If the given subscript is a negative number for the "present rule symbol" and it is a fixed value (<9) for the "present constituent", skip to step (15).

(10) If the given subscript is not a dependent variable for the "present rule symbol", skip to step (14).

(11) If the given subscript of register SYMBOL is equal to the value specified by the rule, skip to step (15).

(12) If the given subscript of register SYMBOL is less than 9 and equal to the value of the corresponding subscript of the "present constituent", skip to step (15).

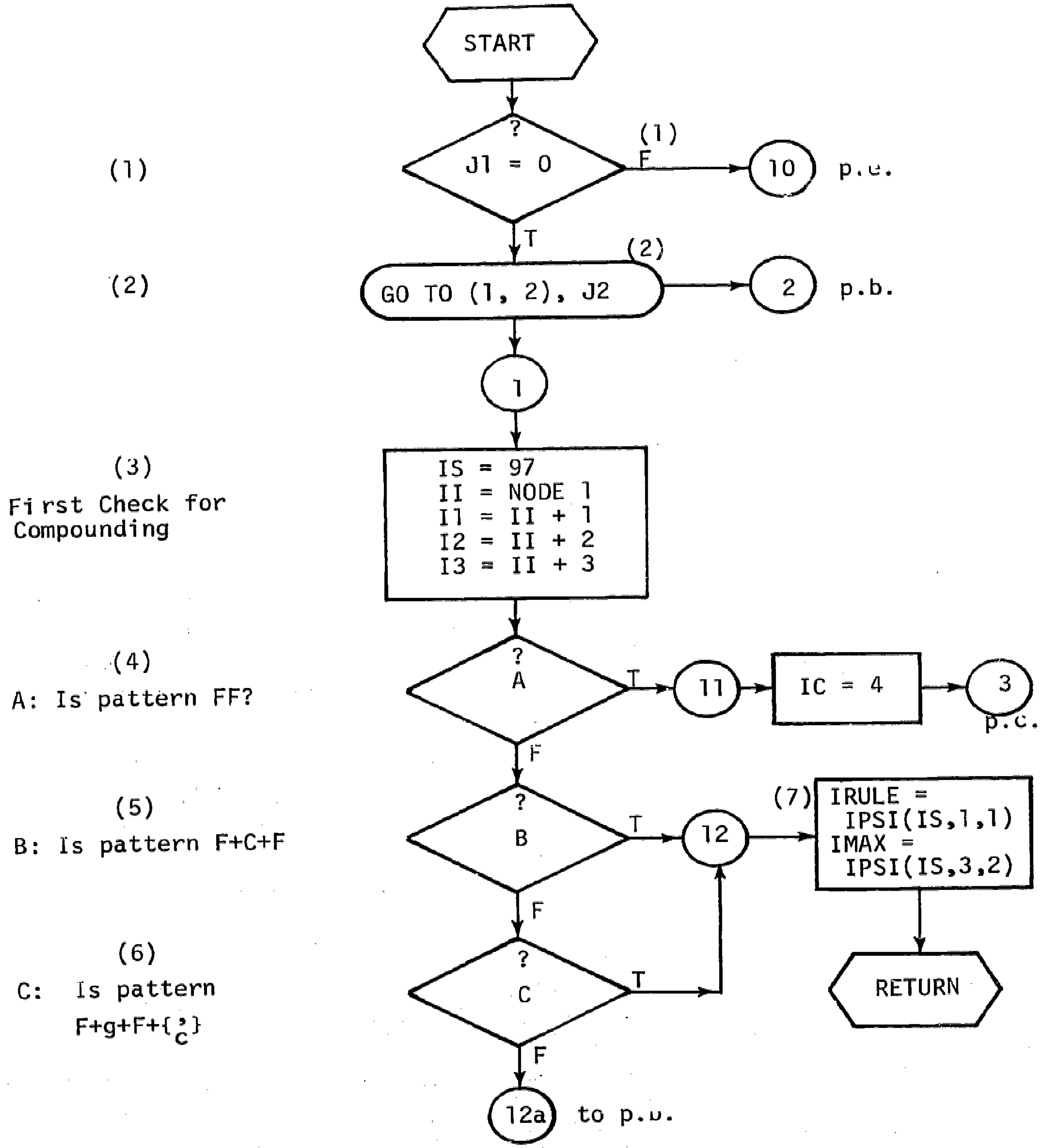


Figure 4.13a: Subprogram RULENO

(8) No Compounding Rules apply; were Compounding Rules in Operation?

(9) Go to "Phrase" rules

(10) Assign "Phrase" Rules

(11) Is There a prev. prediction?

D: NODE(6,NODE1)=0

(12) Is Class undefined?

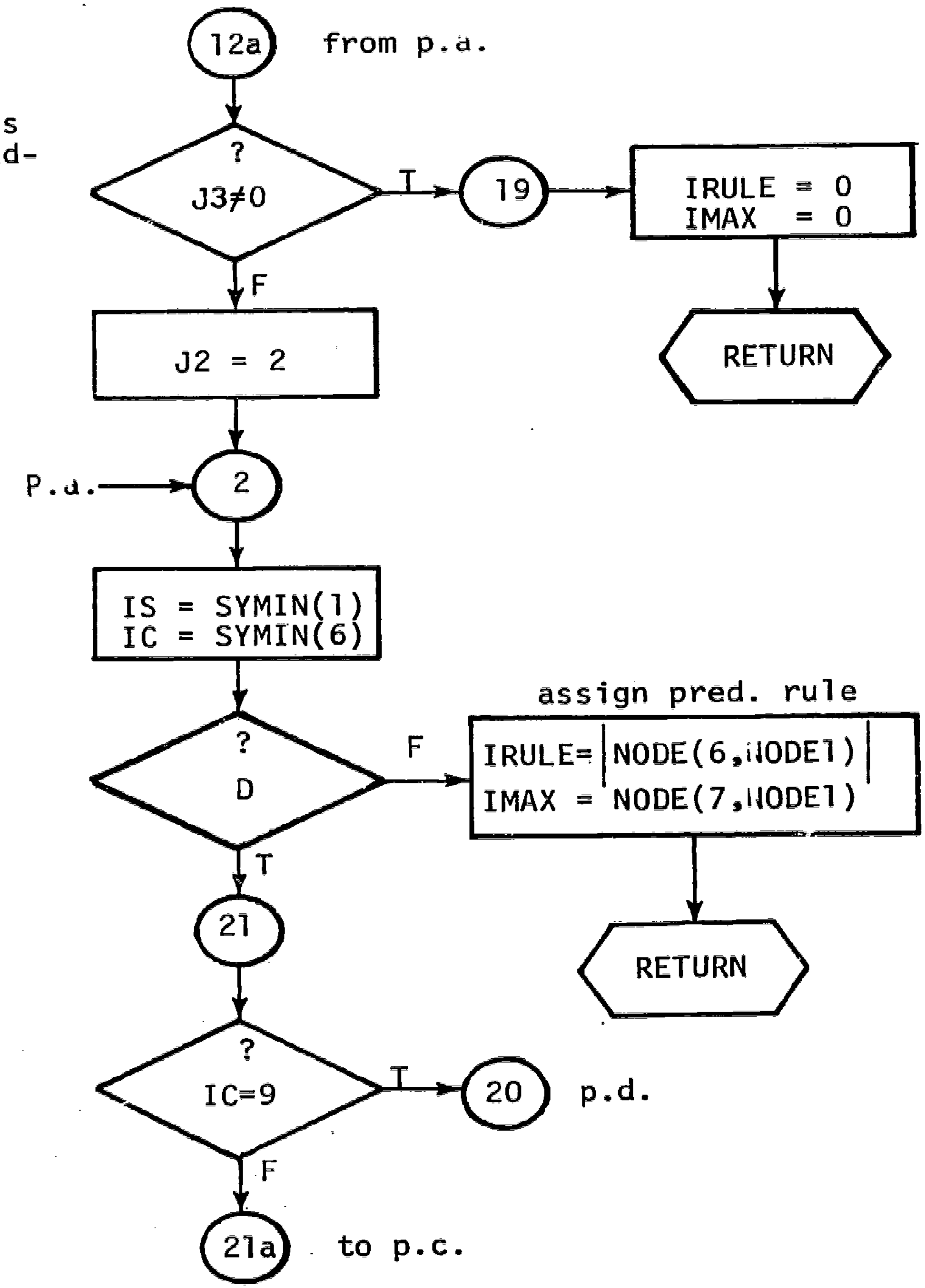


Figure 4.13b: Subprogram RULENO

(13) Try for rule on undefined class

If none, use rule for class IC

E: RULE(IRULE,6) = 9

(14)

(15)

(16) If no rule try for any rule on IS.

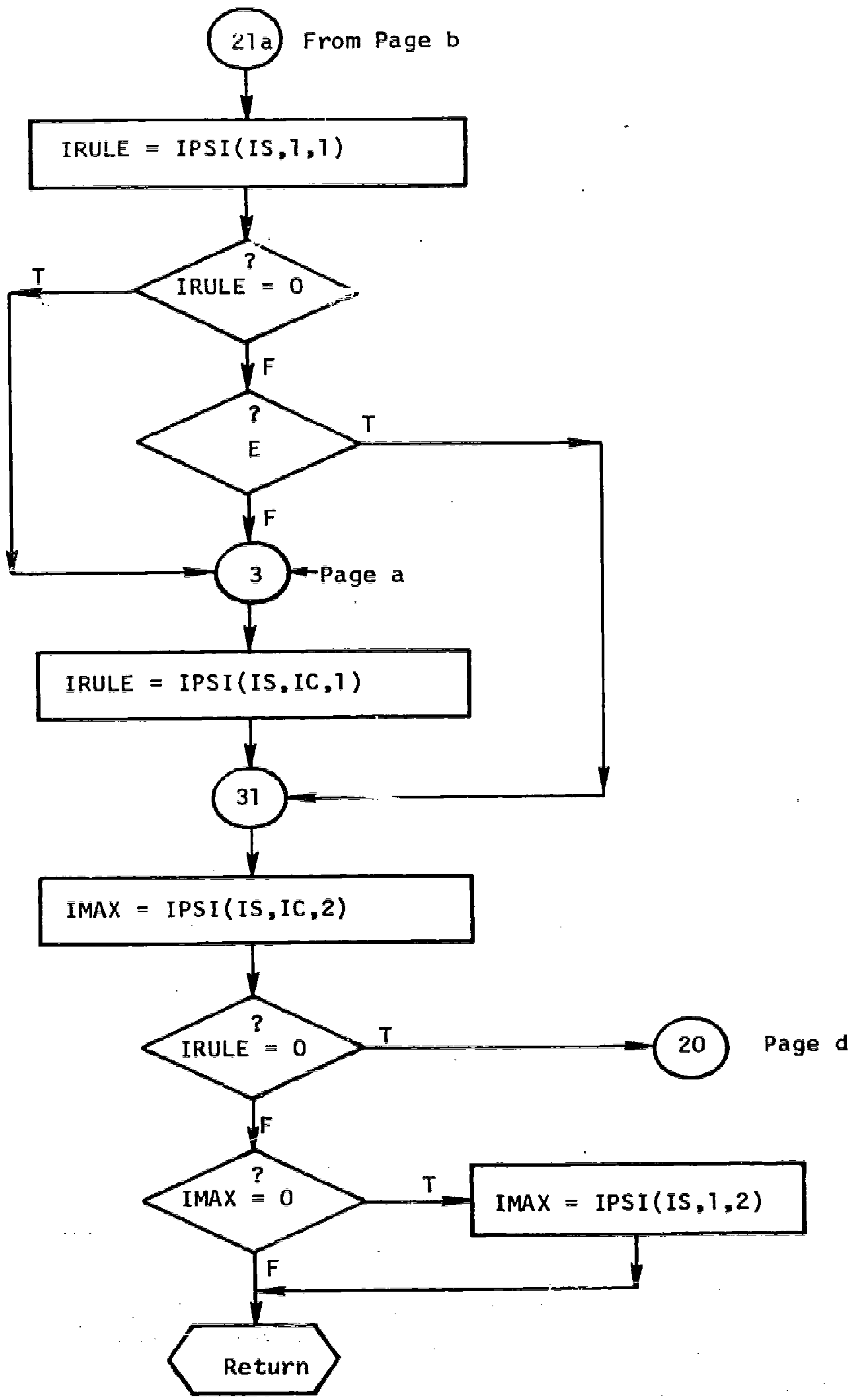


Figure 4.13c: Subprogram RULENO

(17) Find rules for all classes

(17a)

G: IPSI(IS,L,1)=0

(17b)

H: IPSI(IS,L,2)=0

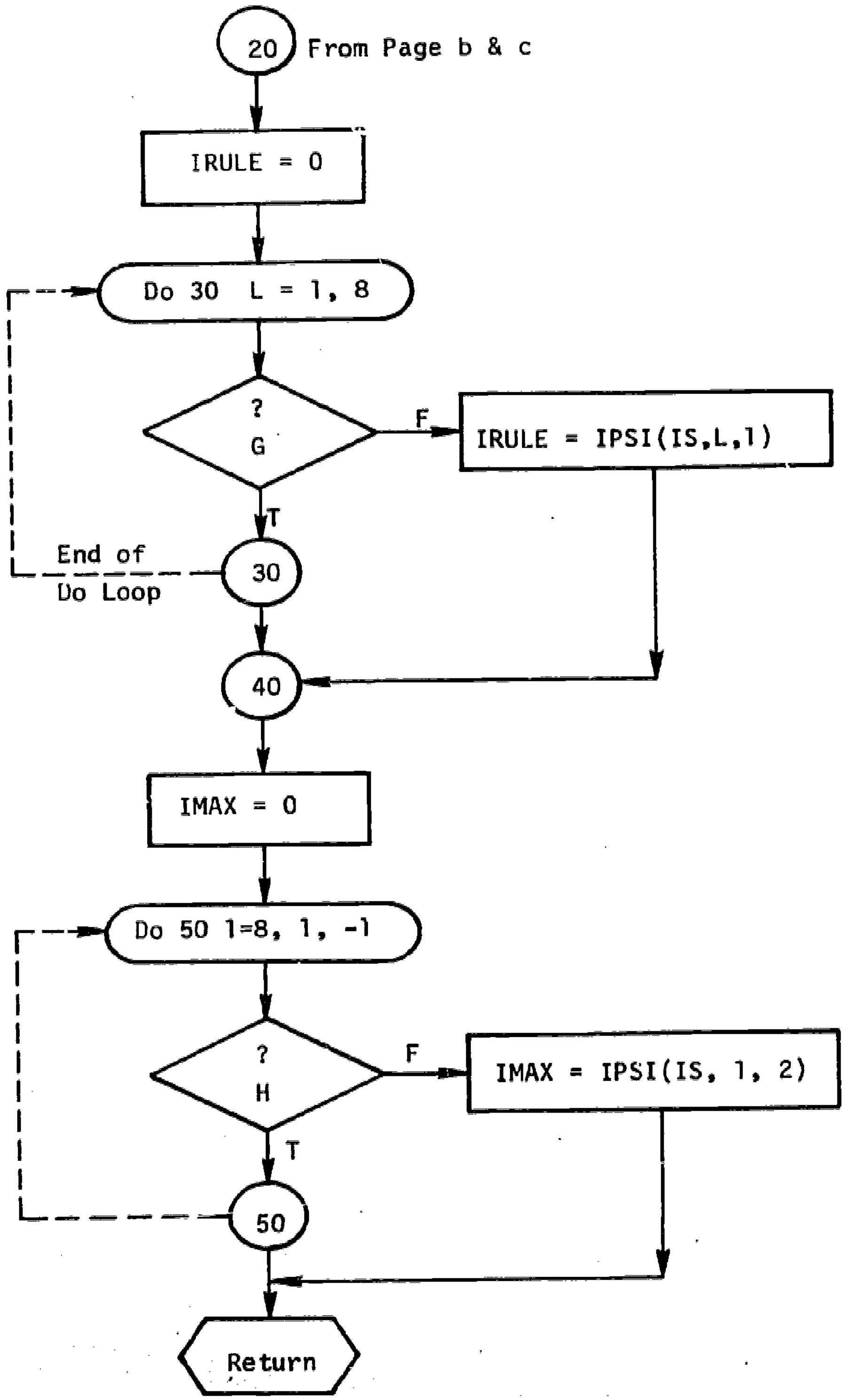


Figure 4.13d: Subprogram RULENO

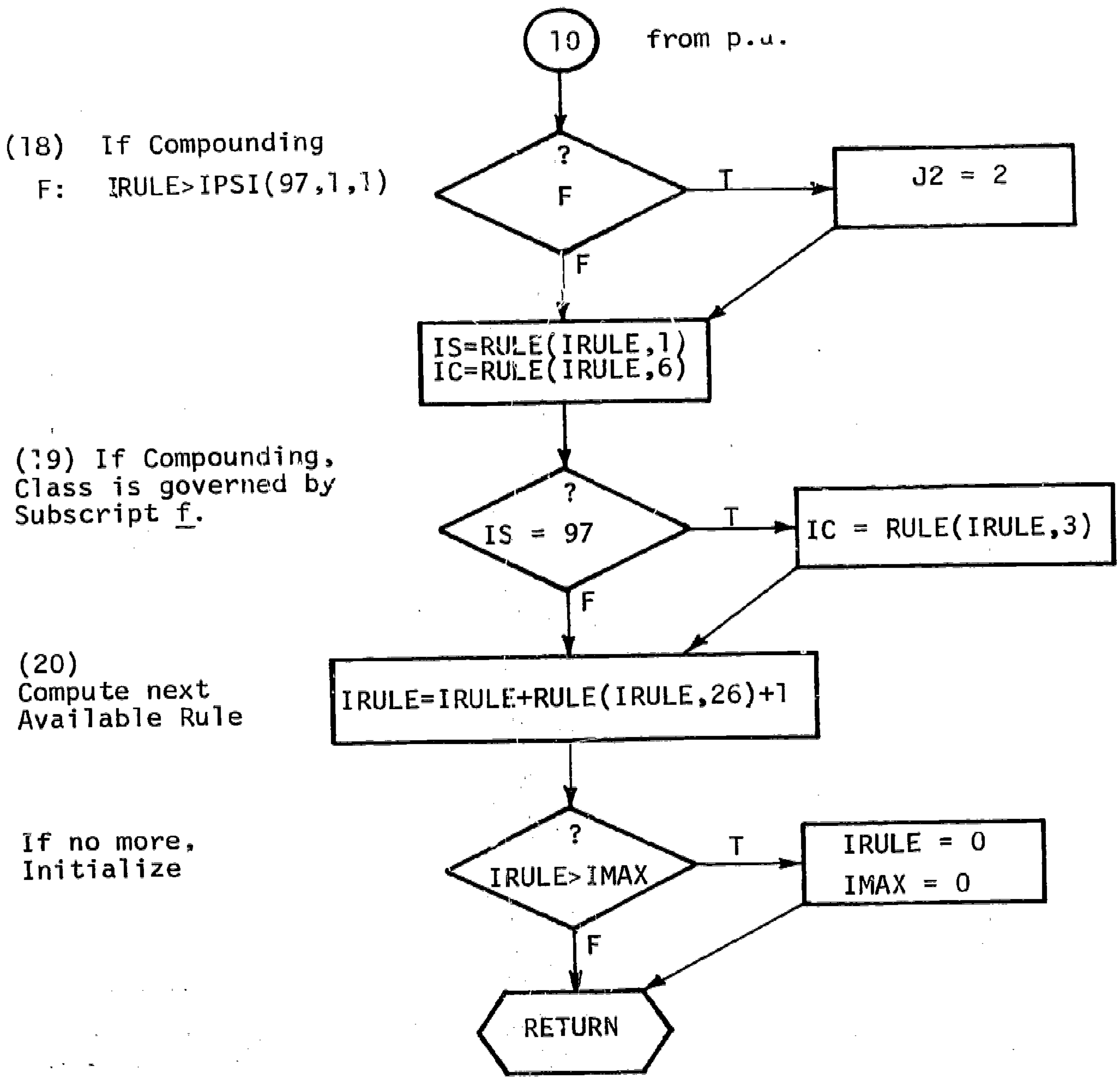
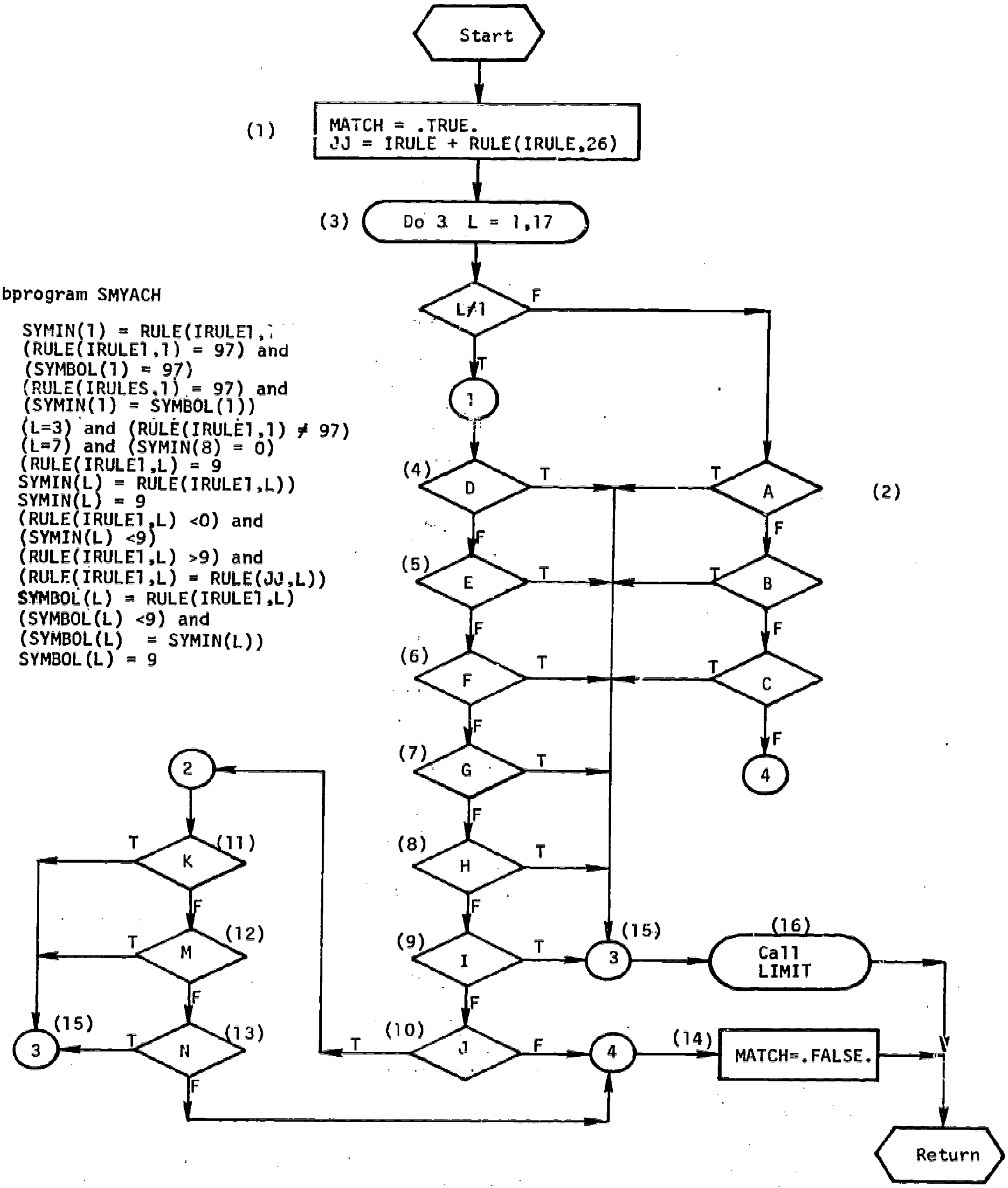


Figure 4.13.e: Subprogram RULENO



Subprogram SMYACH

A: SYMIN(1) = RULE(IRULE1,

B: (RULE(IRULE1,1) = 97) and (SYMBOL(1) = 97)

C: (RULE(IRULES,1) = 97) and (SYMIN(1) = SYMBOL(1))

D: (L=3) and (RULE(IRULE1,1) ≠ 97)

E: (L=7) and (SYMIN(8) = 0)

F: (RULE(IRULE1,L) = 9

G: SYMIN(L) = RULE(IRULE1,L))

H: SYMIN(L) = 9

I: (RULE(IRULE1,L) <0) and (SYMIN(L) <9)

J: (RULE(IRULE1,L) >9) and (RULE(IRULE1,L) = RULE(JJ,L))

K: SYMBOL(L) = RULE(IRULE1,L)

M: (SYMBOL(L) <9) and (SYMBOL(L) = SYMIN(L))

N: SYMBOL(L) = 9

Figure 4.14: Subprogram SYMACH

(13) If the given subscript of register SYMBOL is 9, skip to step (15).

(14) Set the logic monitor MATCH to "false" and return.

(15) If the last subscript (L=17), skip to step 16, otherwise increment L and return to step (3).

(16) Using subprogram LIMIT, test the symbol for any specified restraints, and return.

4.4.1.13 Subprogram VARATT

This subprogram (Fig. 4.15) is called by subprogram RERITE (step 16) to compute the value of the various dependent variable subscripts for the "present rule symbol." For a given rule, certain of the subscripts may be desiginated as dependent variables--that is, the values of these subscripts depend on the values of the corresponding subscripts in the "present constituent." These dependent variable subscripts are written as lower case alphabetic characters in the rules; these were transformed to integer numbers when the rules were read in by the main program (step 3). Reference to Table 4-1 indicates that all alphabetic characters have been transformed to an integer value greater than 9, so that this test is used to identify dependent variable subscripts.

This subprogram performs the following operations:

(1) If the "present rule symbol" is a variable symbol (used in rules for negating or compounding), set the symbol type subscript of the "present rule symbol" to the same value as that of the "present constituent" and store this value in register SYMBOL.

(2) For subscripts 2 through 22, perform steps 3 through 5.

(3) If the given subscript of the "present rule symbol" is -1, set the corresponding subscript of the register SYMBOL to 1 greater than the value of the subscript of the "present constituent," and in either case continue.

(4) If the given subscript of the "present rule symbol" is a dependent variable (>9), set the corresponding subscript of register SYMBOL equal to the value of the subscript of the "present constituent," and in either case continue.

(5) If not the 22-nd subscript, advance index L by one and return to step (3), otherwise continue.

(6) If the symbol type of the "present constituent" is not the same as the symbol stored in register SYMBOL, return, otherwise continue.

(7) If the symbol stored in register SYMBOL indicates that compounding after pattern 2 is being performed, continue, otherwise return.

(8) For symbols being compounded after pattern 2, the following operations are performed:[11]

(9) For phrases with conjunctive compounding (b=1), the *number* feature of the phrase is set to *plural* (n=3).

(10) For phrases with undefined *number* (n=0), the *number* of the phrase is set to that of the "present constituent."

(11) For phrases with undefined *gender* (g=0), the *gender* of the phrase is set to that of the "present constituent."

(12) For phrases with undefined *person* (p=0), the person of the phrase is set to that of the "present constituent."

(13) For the feature *number*, the phrase is assigned the largest value found in itself or any of its like constituent elements.

(14) For the feature *gender*, the phrase is assigned the smallest value found in any of its like constituent elements.

(15) For the feature *person*, the phrase is assigned the smallest value found in any of its like constituent elements.

(16) Return to calling program.

[11]These computation may be found to be dependent on the "content" of the grammar in that they may not be universally true for all Semitic Languages.

4.3.2 Input Map for Computer Program

This section describes the input data required to use the computer program of the Algorithm for Analyzing Hebrew Sentences. Three types of data cards are required:

(1) Program option card,

(2) Data cards describing the grammar and its rules.

(3) Data cards describing the sentences to be analyzed.

The following sections describe each type of card.

4.3.2.1 Program Option Card

This card is used to control the various options available in the program. The data and the associated options are as follows:

Card Col.	Format	Variable	Description
1-5	I5	MODE	= 0, grammar rule are to be read in from cards = 1, grammar rules are stored on tape ITAPE.
6-10	I5	ITAPE	logical unit number of magnetic tape for storage of grammar rules.
11-15	I5	ITRACE	= 0, no diagnostic messages to be printed out. = 1, print out diagnostic messages of program.
16-20	I5	IOUTPT	= 0, do not print out full description of nodes at every level of synthesis. = 1, print out full description of nodes at every level of synthesis.
21-25	I5	IOUTRE	= 0, do not print out tree diagram of synthesis, = 1, print out tree diagram.
26-30	I5	LIMAX	Maximum number of symbol tests.

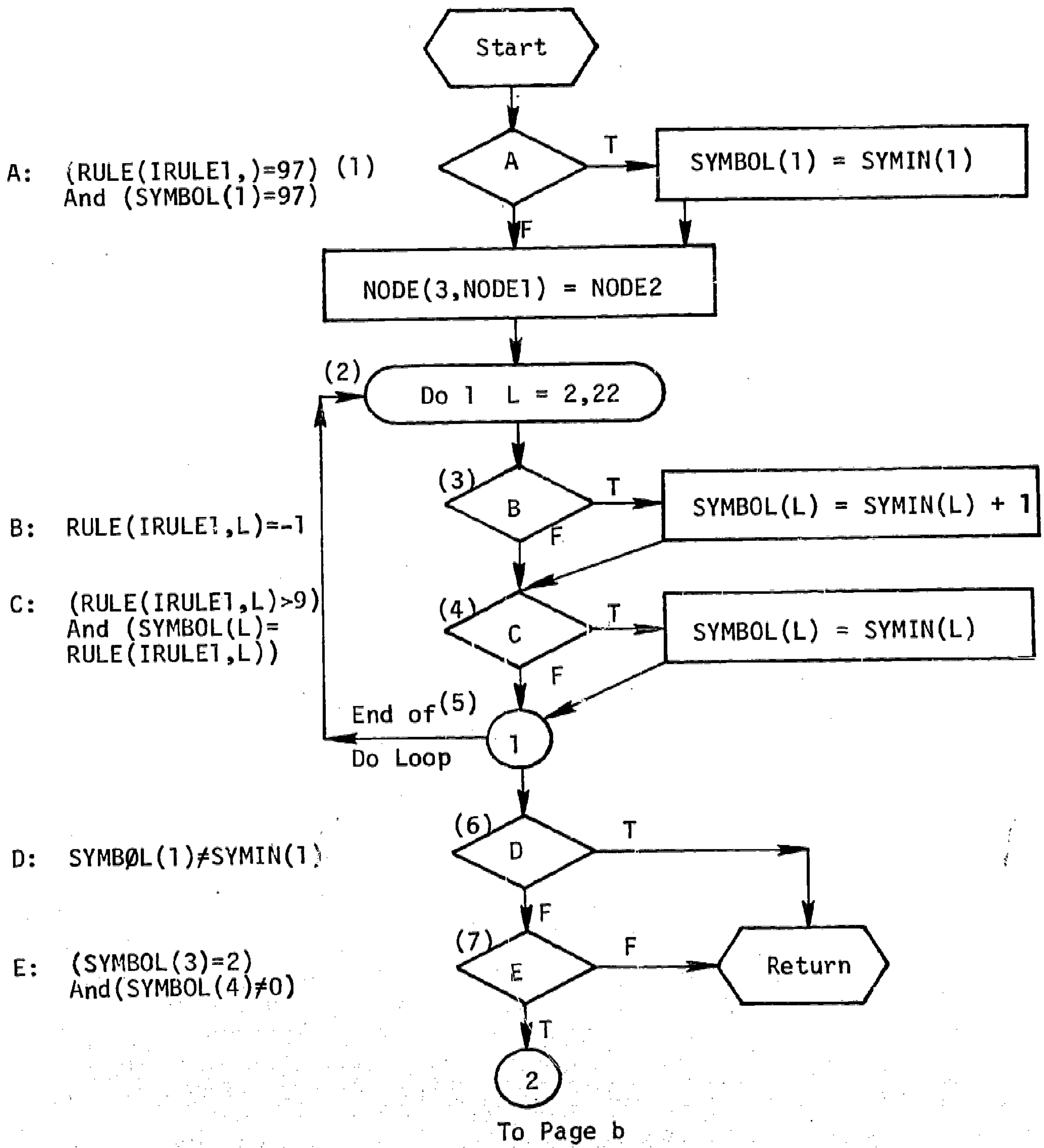


Figure 4.15a: Subprogram VARATT

F: SYMBOL(5) = 1

G: SYMBOL(10) = 0

H: SYMBOL(11) = 0

I: SYMBOL(12) = 0

J: SYMBOL(10) < SYMIN(10)

K: SYMBOL(11) > SYMIN(11)

L: SYMBOL(12) > SYMIN(12)

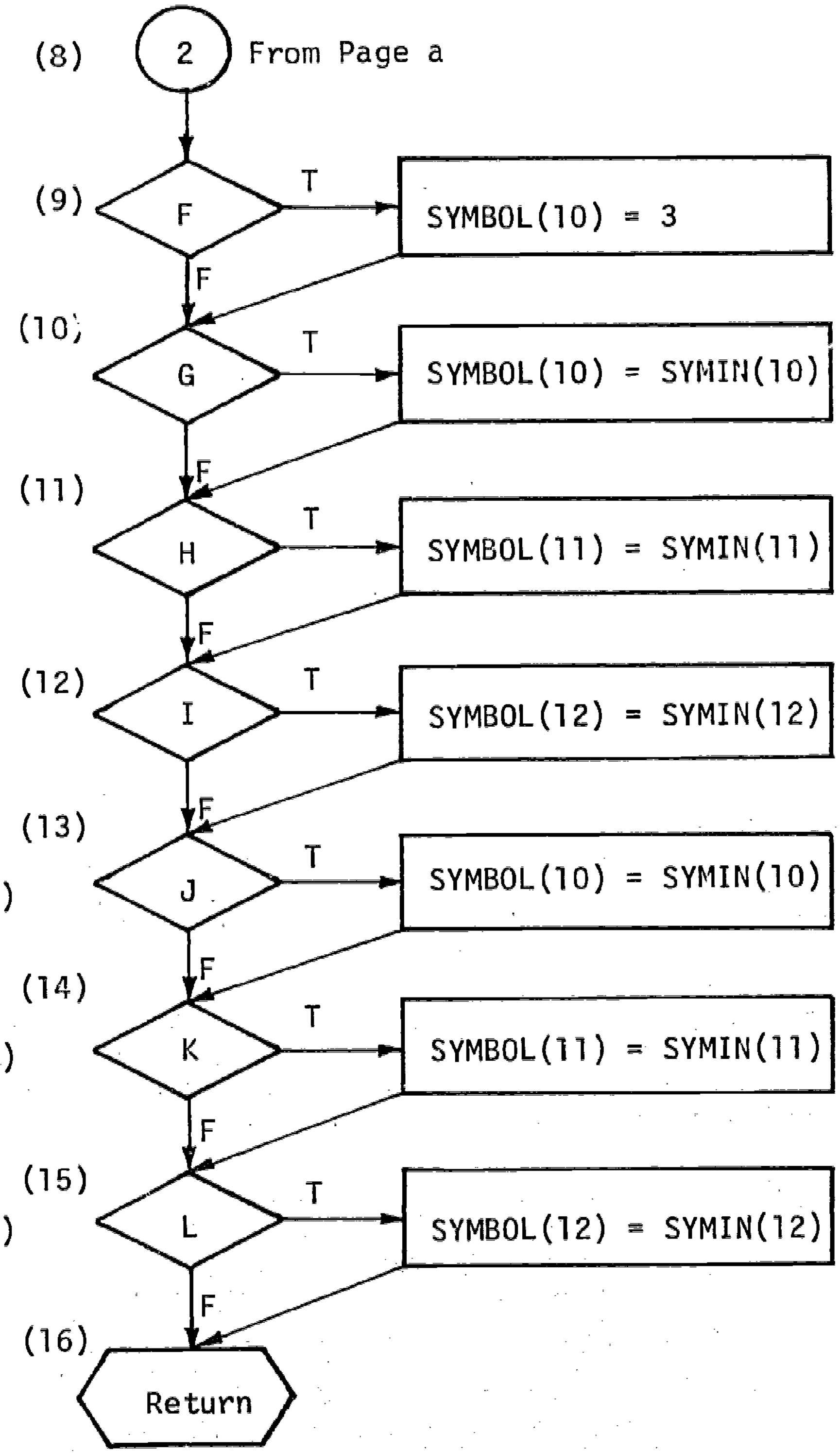


Figure 4.15b: Subprogram VARATT

The variable MODE is used to control the source of the grammar rules used by the program. Initially the rules are read in from cards (MODE=0) and stored on the magnetic tape mounted on logical unit ITAPE. On subsequent runs the rules are read from the tape (MODE=1) unless changes in the rules are to be made.

Variable ITAPE defines the logical unit number containing the magnetic tape on which the grammar rules are stored (MODE=1) or are to be stored (MODE=0).

Variable ITRACE is used for tracing the progress of the program throughout its operation. If ITRACE=1, the program prints out numerous messages together with the values of certain critical variables at key steps of the program. The use of this variable is for program debugging only, it should normally be set to zero.

Variable IOUTPT is used to control the output of the computer. The computer will list a complete description of each nodal point at every level of the synthesis if IOUTPT = 1 (see Appendix C for a sample of this output). This output is useful when the user wants to examine the output results in fine detail, otherwise IOUTPT should be zero.

Variable IOUTRE is used to control the output of the computer. The computer will construct and print out a tree diagram of the analyzed sentence. Each nodal point is identified as to symbol type and class only (see Section 2.3.2 of Part II of this report for a sample of this output). If more detailed data are required the IOUPUT option should be used also. Note: either IOUTPT or IOUTRE (or both) must be 1.

Variable LIMAX is used to control the maximum number of symbol test permitted in a given analysis. If an analysis is not found in less than LIMAX symbol tests, the program terminates the analysis and goes to the next sentence. This prevents the program from continuing to search for analyses beyond a reasonable limit.

4.3.2.2 Program Grammar Cards

The program grammar cards consists of four types:

Type 2-1: Transliteration Table

Type 2-2: Grammar Rules

Type 2-3: Restraint Matrix

Type 2-4: The Symbols

Type 2-5: The Symbol Names

Type 2-6: The Predicate Index

Type 2-7: The Analysis Predcates

Type 2-8: The Feature Values

The program grammar cards are not required if MODE=1; in this case these data are read from the magnetic tape (ITAPE). The following sections define the program grammar cards.

4.3.2.2.1 Transliteration Table: Card 2-1

This card defines the transliteration Table (see Table 4-1). The data are as follows:

Card Col.	Format	Variable	Description
1-6	I6	NL	= number of elements in Table (50 max.).
7-56	50A1	TRANSL(L) L=1,NL	Transliteration Table

4.3.2.2.2 Grammar Rules: Cards 2-2

These cards define the rules of the grammar. Each card defines one constituent of a rule. Thus the rule

B + C = A

requires three cards: the first defines B, the second C, and the third A. A maximum of 800 cards is permitted. A card with 999 in columns 1-3 should follow the last grammar rule card. The rules should be placed in numeric order by rule number. See Appendix A for a complete listing of the rules used for the present research. The following is the data format for each card:

Card Col.	Format	Variable	Description
1-3	A3	SN	Symbol number (See Table 4-2)
6	A1	M	Subscript m - optional/mandatory
9	A1	F	Subscript f - compound class
11-12	A2	K	Subscript k - compound number
15	A1	B	Subscript b - compound type
18	A1	C	Subscript c - symbol class
21	A1	ℓ	Subscript ℓ - negative class
24	A1	Y	Subscript y - negative/positive
27	A1	D	Subscript d - definiteness
30	A1	N	Subscript n - number
33	A1	G	Subscript g - gender
36	A1	P	Subscript p - person
39	A1	R	Subscript r - prepostion class
42	A1	A	Subscript a - verb class
45	A1	V	Subscript v - voice
48	A1	I	Subscript i - mood
51	A1	T	Subscript t - tense
53-54	A2	S	Subscript s - stem
57	A1	W1	Subscript w_1 - root letter 1
60	A1	W2	Subscript w_2 - root letter 2
63	A1	W3	Subscript w_3 - root letter 3
66	A1	W4	Subscript w_4 - root letter 4
69	A1	ST	Symbol Type
70-72	I3	RN	Rule Number
73	I1	EN	Element Number
74	I1	NRH	No. of Rt. Hand Elements
75-76	I2	RT	Restraint Type
77-78	I2	RS	Restraint Subscript

Variable SN is the symbol number of the given rule element as specified on Table 4-2. A card with SN=999 should follow the last grammar rule.

Variables m, f, k, b, c, ℓ, y, d, n, g, p, r, a, v, i, t, s, w_1, w_2, w_3, and w_4 are the subscripts of the symbols are defined in Part II, Section 2.2.1.

Variable ST identifies the symbol as follows:

ST = 0 for non-terminal symbols

ST = 1 for terminal symbols

Variable RN is the rule number associated with the given card. For example: in the rule

$$B + C = A \tag{52.1}$$

Variable RN = 521 for all cards of the rule. Variable EN is the element number of the symbol defined by the card. In the above example:

on the card defining B, EN=1

on the card defining C, EN=2

on the card defining A, EN=0

Variable NRH defines the number of left hand elements in the rule. In the above example, NRH=2 for all cards of the rule.

Variable RT defines the restraint type that applies to the symbol. The value of RT refers to a row of restraints in the Restraint Matrix which is defined by Cards 2-3. Variable RS defines the subscript of the symbol to which the restraint applies. For example,

RT = 3, RS = 10, means that restraint type 3 applies to the 10-th subscript n.

4.3.2.2.3 Restraint Matrix: Cards 2-3

These cards define the restraints that are placed on a given symbol in a grammar rule. See Section 4.3.1.4 for a discussion and explanation of the data in Matrix RESTRT. The following is format of the data:

Card Col.	Format	Variable	Description
1-5	I5	IM	Number of restraints on I-th list (Max.=4)
6-25	4I5	RESTRT(J,I) J=2, IM + 1	I-th list of restraint codes

Number of cards 15.

The data on these cards are listed in Table 4-9.

Table 4-9

RESTRAINT MATRIX DATA

Card Col.					Card No.
1-5	6-10	11-15	16-20	21-25	
1	-1	0	0	0	2-3.1
1	-2	0	0	0	2-3.2
1	-3	0	0	0	2-3.3
2	0	1	0	0	2-3.4
2	1	2	0	0	2-3.5
2	-1	-2	0	0	2-3.6
2	1	3	0	0	2-3.7
2	2	5	0	0	2-3.8
2	4	5	0	0	2-3.9
2	6	7	0	0	2-3.10
3	1	2	3	0	2-3.11
3	4	5	6	0	2-3.12
4	1	2	3	4	2-3.13
4	-1	-2	-3	-4	2-3.14
0	0	0	0	0	2-3.15

4.3.2.2.4 The Table of Symbols: Cards 2-4

These cards define the symbols used in the grammar. See Section 4.1.1.2 for further discussion. Ten cards are required for these data which are in 10(2X,A3) format. The data for the Hebrew grammar are given in Table 4-10.

4.3.2.2.5 The Table of Symbol Names: Cards 2-5

These cards define the English name of each symbol used in the grammar. See Section 4.1.1.5 for further discussion. Ninety-six cards are required for these data (one for each symbol) which are in the following format:

Card Col.	Format	Variable	Description
1-36	6A6	SYML(I)	Name of I-th symbol
67-72	I6	ISYMB(I)	Number of 6 character words used by the Ith name.

Table 4-4 contains the symbol names for the Hebrew grammar.

Table 4-10

TABLE OF SYMBOLS

Card Column										Card
0-5	6-10	11-15	16-20	21-25	26-30	31-35	36-40	41-45	46-50	No.
Z-	VQO	APA	AP	AS	NA	SQO	NS	RU	RO	2-4-1
BAA	BAB	BAC	BAD	BAE	BAF	BBA	BBB	BA	BBC	2-4.2
BP	NPB	NPA	NPC	NAP	NP	DPA	DPB	DPC	DP	2-4.3
EA	VB	VBB	VC	VAA	VA	BC	XP	NO	DPD	2-4.4
RSP	NSP	NOP	NIP	NPX	VMA	VMB	VMC	VMD	VM	2-4.5
VMR	VMI	VP	VRB	VRI	NV	NW	EPB	EPA	EP	2-4.6
SAA	SAB	SAC	SA	SRO	SRI	RG	KN	KC	KK	2-4.7
KI	KD	SD	S-	SI	SC	A-	B-	C-	D-	2-4.8
E-	G-	H-	I-	J-	L-	N-	O-	P-	Q-	2-4.9
R-	T-	U-	V-	W-	Y-	F-	98	99	100	2-4.10

4.3.2.2.6 The Predicate Index: Card 2-6

These cards provide an index for cataloging the analysis predicates. See Section 4.1.1.6 for further discussion. One-hundred cards are required for these data which are in the following format:

Card Col.	Format	Variable	Description
1-40	8I5	INDEX(I,J)	Index number of the analysis predicate for the I-th symbol, J-th class.

Table 4-6 contains the data for the Hebrew grammar.

4.3.2.2.7 The Analysis Predicates: Card 2-7

These cards provide the list of analysis predicates used in the algorithm. See Section 4.1.1.6 for further discussion. One-hundred eighty-five cards are required for these data (one for each predicate) which are in the following format:

Card Col.	Format	Variable	Description
1-36	6A6	AMSG(I)	the I-th analysis predicate

Table 4-5 contains the list of analysis predicates for the Hebrew grammar.

4.3.2.2.8 The Feature Values: Cards 2-8

These cards provide the table of feature values used in the grammar. See Section 4.1.1.7 for further discussion. Seventeen (17) cards are required for these data (one for each subscript involved) which are in the following format:

Card Col.	Format	Variable	Description
1-48	8A6	ATTVAL(I,J)	Feature value of the I-th subscript, J-th catagory.

Table 4-7 contains a list of the Feature Values for the Hebrew grammar.

4.3.3 The Sentence Description Cards

The sentence analysis algorithm uses a set of data cards to describe the Hebrew sentence being analyzed. These cards consist of two types:

Type 3-1: Hebrew Sentence Card
Type 3-2: Word Description Cards

A set of sentence description cards is required for each sentence to be analyzed. A blank card should follow the last set of sentence description cards.

4.3.3.1 Hebrew Sentence: Card 3-1

This card provides a listing (in transliterated characters) of the Hebrew sentence to be analyzed, a sentence number assigned by the user, and the number of words in the sentence. The transliterated Hebrew sentence is printed on the tree diagram analysis computed by the algorithm. The sentence number is used to identify the sentence in the analysis statements assembled by the algorithm. One card type 3-1 is required for each sentence to be analyzed. The following is the data format:

Card Col.	Format	Variable	Description
1-72	12A6	HEBREW	Transliterated Hebrew sentence to be analyzed.
73-75	I3	NOP	Sentence number assigned by user.
76-78	I3	IMAXI	Number of words in sentence. Count punctuation, prefixes and suffixes as separate words.

4.3.3.2 Word Description: Cards 3-2

These cards provide a complete grammatical description of each word in the sentence being analyzed, one card for each word, prefix (including definite article), pronominal suffix, and punctuation mark. These data provide the equivalent of the output of an automatic word analysis algorithm which presently is not incorporated in the program. Whenever the word analysis algorithm is included, these no longer will be required. The following is the format of the data:

Card Col.	Format	Variable	Description
1-6	I6	SN(I)	Symbol Number of the I-th word (see Table 4-2). Only values between 77 and 96 are used.
11-12	I2	M(I)	Subscript m of the I-th word. Always = 1.
19-20	I2	C(I)	Symbol Class of the I-th word
21-22	I2	L(I)	Negative class of the I-th word
27-28	I2	N(I)	Number attribute of I-th word = 0 - does not apply = 1 - singular = 2 - dual = 3 - plural
29-30	I2	G(I)	Gender attribute of I-th word. = 0 - does not apply = 1 - masculine = 2 - feminine
31-32	I2	P(I)	Personal attribute of I-th word = 0 - does not apply = 1 - first person = 2 - second person = 3 - third person Most nouns are third person.

Card Col.	Format	Variable	Description
33-34	I2	R(I)	Subscript r for I-th word. = 0 for all words except prepositions and verbals
35-36	I2	A(I)	Subscript a for I-th word. = 0 for all words except verbals.
37-38	I2	V(I)	Voice attribute of I-th word. = 0 for all words except verbals = 1 - active voice = 2 - passive voice = 3 - reflexive voice
39-40	I2	IM(I)	Mood attribute of I-th word. = 0 for all words except verbs = 1 - indicative mood = 2 - imperative mood = 3 - subjunctive mood
41-42	I2	T(I)	Tense attribute of I-th word = 0 for all words except verbs = 1 - past tense = 2 - future tense This attribute applies to the tense inflection of the specific word itself, not to modifications of tense due to auxiliaries or adverbs. Present tense verb is classified as a participle.
43-44	I2	S(I)	Suffix s (stem) for the I-th word.
45-48	A4	W(I)	The four-letter root of the I-th word. Roots are in English transliteration. For non-Hebrew words, omit data.
49-72	4A6	ENGLISH (I)	The English equivalent of the Hebrew word.

Variables SN, C, L, R, A, V, S, W, and ENGLISH are obtained from the dictionary (see Part II of this report, Appendix A). Variables N, G, P, IM, and T, are defined by the inflection of the given word as found in the sentence.

4.4 Test and Demonstration of Algorithm

In order to test the computerized algorithm and to verify the grammar of modern Hebrew syntax, a total of 26 sentences were analyzed on the computer.

The sentences correspond to those generated by the synthesis algorithm (see Part III), except that not all are included and some are in simplified form.

Computations were performed on a UNIVAC 1108. No record was kept of the computing time required to generate a sentence, however, a conservative estimate would be about 2.0 seconds of computer time per sentence. Computing time is roughly proportional to the length and complexity of the sentence.

4.4.1 Sentences Analyzed by Algorithm

This section contains a list of the Hebrew sentences analyzed by the computerized algorithm preceded by their English equivalent. The number accompanying each pair of sentences refers to the corresponding tree diagrams generated by the algorithm which are contained in Section 2.3.2 of Part II of this report.

(1) Chaim Nuchman Biyalik was a great poet in the land of Israel. XYYM NXMN BYALYQ HYH MSWRR GDWL BAR& YSRAL.

(101) Was Chaim Nuchman Biyalik a great poet in the land of Israel? HAM XYYM NXMN BYALYQ HYH MSWRR GDWL BAR& YSRAL?

(102) Chaim Nuchman Biyalik be a great poet in the land of Israel! XYYM NXMN BYALYQ HYH[12] MSWRR GDWL BAR& YSRAL!

(103) Chaim Nuchman Biyalik will be a great poet in the land of Israel! XYYM NXMN BYALYQ YHYH MSWRR GDWL BAR& YSRAL.

(2) He was greater than all the poets. HWA HYH GDWL MKL HMSWRRYM.

(201) Who was greater than all the poets? MY HYH GDWL MKL HMSWRRYM?

(4) He lived in a small village. HWA YSB BKPR Q@N.

(401) He will live in a small village. HWA YYSB BKPR Q@N.

---

[12] This illustrates the imperative sentence. The HYH is the imperative הֱיֵה!.

(402) He is living in a small village.
HWA YWSB BKPR Q@N.

(403) He used to sit on a small chair.
HWA HYH YWSB OL KCA Q@N.

(404) He will continually live in a small village.
HWA YHYH YWSB BKPR Q@N.

(5) He loved to study law.
HWA AHB LLMWD TWRH

(6) But he also loved the fields and the forests.
ABL HWA AHB GM AT HSDWT WAT HYORYM

(7) He went to study in a large academy.
HLK LLMWD BYSYBH GDWLH.

(701) Did he go to study in a large academy?
HHLK[13] LLMWD BYSYBH GDWLH?

(702) Did he go to study in a large academy?
HAM[14] HLK LLMWD BYSYBH GDWLH?

(703) Did Biyalik go to study in a large academy?
HAM BYALYQ HLK LLMWD BYSYBH GDWLH?

(8) He wrote the first poems.
HWA KTB AT HSYRYM HRASWNYM

(9) He wrote about the sun.
HWA KTB OL HSMS.

(12) The students studies these poems.
HTLMYDYM LMDW AT HSYRYM HALH.

(121) They knew them by heart.
YDOW AWTM OL PH.

(13) Biyalik traveled to the land of Israel.
BYALYQ NCO LAR& YSRAL.

(23) All the Jews wept over his death.
KL HYHWDYM BKW OL MWTW.

(26) Do you want to see the synagogue?
HAM ATH RW&H LRAWT AT BYT HKNCT?

(11) Joseph does not have a room.
AYN LYWCP XDR.

(112) Joseph has a room
YS LYWCP XDR.

---

[13]This illustrates the classical Hebrew option.
[14]This illustrates the modern Hebrew option.

4.4.2 The Algorithm Tested by Computer

The 26 sentences analyzed by the computer were selected to test the algorithm. In analyzing these sentences, 57 of the 179 rules on the intermediate symbols were used. Of the 73 intermediate symbols, 40 were tested for a least one class, and 15 were tested for all class variants. Of the 20 terminal symbols, 16 were tested for at least one class, and 10 were tested for all class variants. The purpose of this test was to verify the analytic capability of the algorithm, not to further verify the grammar rules. Time limitation prevented more extensive tests. However, sufficient tests were performed to demonstrate the analytic capability of the algorithm and to reveal the areas where improvement is required.

Table 4-11 is a listing of the sentence numbers of the test sentences that demonstrate the symbols of the grammar and thus, the corresponding rules. For example, for symbol No. 6, N_a, test Sentences 1 and 4 (plus others) demonstrate the use of class 1 (Rule 9.1), test Sentences 2 and 6 (plus others) demonstrate class 2 (Rule 9.3), test sentences 1 and 701 (plus others) demonstrate class 3 (Rule 9.4); the asterisk (*) indicates that the symbol does not have classes 4 through 8. Tree diagrams of the test sentences are contained in Section 2.3.2 of Part II of this report which show the specified symbol as it relates to the structure of the sentence.

Because the rules of the grammar are unordered (by original definition), the analysis algorithm requires predictive logic to synchronize the production of related constituents. This logic omputes the possibility of satisfying a grammar rule if its application is postponed one or more passes on the string of symbols. This feature causes the algorithm to delay execution of high level rules until applicable lower level rules have been satisfied.

The analysis algorithm is presently equipped with prediction logic to a depth of two passes by the use of Subprograms PROPH1 and PROPH2. With this depth of predictive logic, the algorithm is capable of analyzing most simple sentences. Difficulties are experienced with complex sentences that contain compound verb modifying phrases, compound predicates, predicates with verbs of class 4, 7, and 8, and with sentences containing subordinate clauses or relative clauses. The analysis of these sentences requires predictive logic that is more powerful. This logic must delay the execution of a rule until a deep structure relationship with the preceding symbol is predicted for the next pass on the string. Time has not permitted the incorporation of this feature into the algorithm, but there is nothing to prevent this and other refinements from being added in the future.

Table 4-11

SENTENCES ILLUSTRATING GRAMMAR SYMBOLS

Symbol No.	Symbol Name	Symbol Class							
		1	2	3	4	5	6	7	8
1	Z		*	*	*	*	*	*	*
2	Vgϕ		*	*	*	*	*	*	*
3	Apa	4,7,8	1,2	*	*	*	*	*	*
4	Ap	1,2,4,7	*	*	*	*	*	*	*
5	As				*	*	*	*	*
6	Na	1,4+	2,6+	1,701+	*	*	*	*	*
7	Sqϕ			*	*	*	*	*	*
8	Ns	23	*	*	*	*	*	*	*
9	Rd		12		*	*	*	*	*
10	Ro	121	*	*	*	*	*	*	*
11	Baa				*	*	*	*	*
12	Bab		*	*	*	*	*	*	*
13	Bac				*	*	*	*	*
14	Bad			*	*	*	*	*	*
15	Bae				*	*	*	*	*
16	Baf				*	*	*	*	*
17	Bba						*	*	*
18	Bbb			*	*	*	*	*	*
19	Ba			*	*	*	*	*	*
20	Bbc			*	*	*	*	*	*
21	Bp			*	*	*	*	*	*
22	Npb	1,2,5+	23	403	*	*	*	*	*
23	Npa	1,4+	1,2,13+		*	*	*	*	*
24	Npc		*	*	*	*	*	*	*

Table 4-11 (continued)

SENTENCES ILLUSTRATING GRAMMAR SYMBOLS

Symbol No.	Symbol Name	Symbol Class							
		1	2	3	4	5	6	7	8
25	Nap		1,101+		*	*	*	*	*
26	Np	1,2,5+	*	*	*	*	*	*	*
27	(Dpa)[15]	*	*	*	*	*	*	*	*
28	(Dpb)[15]	*	*	*	*	*	*	*	*
29	(Dpc)[15]	*	*	*	*	*	*	*	*
30	Dp	6					*	*	*
31	Ea	402,26+	*	*	*	*	*	*	*
32	Vb	1,4,5+	*	*	*	*	*	*	*
33	Vbb	1,2,4+	*	*	*	*	*	*	*
34	Vc		1,2,5+	*	*	*	*	*	*
35	Vaa	1,2,+	*	*	*	*	*	*	*
36	Va	1,2,7,+	*	*	*	*	*	*	*
37	Bc			*	*	*	*	*	*
38	Xp	1,2,4+			*	*	*	*	*
39	No	6,8+		*	*	*	*	*	*
40	Dpd			*	*	*	*	*	*
41	Rsp	2,4,5+		*	*	*	*	*	*
42	Nsp	A,1,23+	2,4,5+			*	*	*	*
43	Nop	5	6,8+	*		*	*	*	*
44	Nip	121		*	*	*	*	*	*
45	Npx	201		403,1		23	*	*	*
46	Vma			5,6+	*	*	*	*	*
47	Vmb		121		*	*	*	*	*
48	Vmc					*	*	*	*

[15] Symbols D_{pa}, D_{pb}, and D_{pc} have been removed from the grammar as a result of modifications made on this project.

Table 4-11 (continued)

SENTENCES ILLUSTRATING GRAMMAR SYMBOLS

Symbol No.	Symbol Name	Symbol Class 1	2	3	4	5	6	7	8
49	Vmd				*	*	*	*	*
50	Vm	1,2,		5,6+	121	4,7+	5.7+		
51	Vmr						*	*	*
52	Vmi				*	*	*	*	*
53	Vp	1,2,5+	*	*	*	*	*	*	*
54	Vrb		*	*	*	*	*	*	*
55	Vri		*	*	*	*	*	*	*
56	Nv	5,7+			*	*	*	*	*
57	Nw		*	*	*	*	*	*	*
58	Epb			*	*	*	*	*	*
59	Epa	403		*	*	*	*	*	*
60	Ep	403	*	*	*	*	*	*	*
61	Saa				111,112	*	*	*	*
62	Sab	1,2+						*	*
63	Sac	7,701	*	*	*	*	*	*	*
64	Sa	111	7	1,2,4+	*	*	*	*	*
65	Sro			*	*	*	*	*	*
66	Sri			*	*	*	*	*	*
67	Rg				*	*	*	*	*
68	Km			*	*	*	*	*	*
69	Kc					*	*	*	*
70	Kk		*	*	*	*	*	*	*
71	Ki	701,26+	201			*	*	*	*
72	Kd			*	*	*	*	*	*

Table 4-11 (continued)

SENTENCES ILLUSTRATING GRAMMAR SYMBOLS

Symbol No.	Symbol Name	Symbol Class 1	2	3	4	5	6	7	8
73	Sd			*	*	*	*	*	*
74	S	1,2,7+			*	*	*	*	*
75	S_i	701,201			*	*	*	*	*
76	S_c	1,2,4+	201,701+	101	*	*	*	*	*
77	A	1,2,4,+	*	*	*	*	*	*	*
78	B							*	*
79	C								6
80	D	6						*	*
81	E	26,402+	*	*	*	*	*	*	*
82	G			*	*	*	*	*	*
83	H	2,6+	*	*	*	*	*	*	*
84	I							*	*
85	J	1,2,13+	*	*	*	*	*	*	*
86	L	111				*	*	*	*
87	N	1,2,4+	111,112	1,701+	*	*	*	*	*
88	O	6,8,12	*	*	*	*	*	*	*
89	P	1,2,5,8,13+	*	*	*	*	*	*	*
90	Q	701+	26,702	*	*	*	*	*	*
91	R	12	2,4,5+	121,23+		201	*	*	*
92	T					201,701	1,2,4+	102	
93	U			112					*
94	V	1,2,+	*	*	*	*	*	*	*
95	W		*	*	*	*	*	*	*
96	Y	5,7+	*	*	*	*	*	*	*

4.5 Conclusion

Tests of the algorithm have demonstrated its capability for analyzing most simple Hebrew sentences. Present limitations on the depth of its predictive logic have prevented the successful analysis of more complex sentences. However, the general requirements for extending its capability are known, and there is nothing to prevent the incorporation of these refinements in future versions.

Two additional aspects of the algorithm need further attention. First, there are several computations of the algorithm that are dependent upon the "content" of the grammar upon which it operates. If the algorithm were modified to free these computations from such dependence, a modification which is feasible, the algorithm could be used to generate sentences in other Semitic languages, such as Arabic, whenever grammars of these languages become available.

The second aspect of the algorithm needing further attention is the format of the input data prepared by the user. The sentence description input to this algorithm is much less complex than that of the synthesis algorithm. However, it still requires a complete grammatical description of each word in the sentence. This requirement could be eliminated by the incorporation of an automatic word analysis algorithm. Such an algorithm exists for Hebrew but it has not been made a part of this present program.

Appendix

PART IV

APPENDIX A

LISTING OF THE

GRAMMAR OF

HEBREW SYNTAX FOR

ANALYSIS OF SENTENCES

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-  /X/Z)              RULE NO.

   Z-(10 001000000R0000 00000 0 0) +              ( 55. 1)
 NOP(11 99909900000000 00000 0 0) +
  DP(94 99909000000000 00000 0 0) =
 VMI(10 003000000R4000 00000 0 0)

   Z-(11 99909000090000 00000 0 0) +              ( 52. 1)
 NOP(11 99909900000000 00000 0 0) +
  NP(11 99109999900000 00000 0 0) +
  DP(94 99909000000000 00000 0 0) =
 VMD(10 00300000008000 00000 0 0)

   Z-(11 99909000090000 00000 0 0) +              ( 51. 1)
 NOP(91 99909900000000 00000 0 0) +
  DP(94 99909000000000 00000 0 0) +
  KD(11 99909000000000 00000 0 0) =
 VMC(10 00300000007000 00000 0 0)

   Z-(11 99909000090000 00000 0 0) +              ( 49. 1)
 NOP(11 99909900000000 00000 0 0) +
  DP(94 99909000000000 00000 0 0) =
 VMA(10 00300000003000 00000 0 0)

   Z-(11 991090000R0000 00000 0 0) +              ( 54. 1)
  DP(94 99909000000000 00000 0 0) =
 VMR(10 004000000R5000 00000 0 0)

   Z-(10 0010Y0000R0000 00000 0 0) =              ( 41. 1)
  XP(10 0020Y0000R0000 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)            RULE NO.


   VQO(11 9910YONGPR9999 00000 0 0) =              ( 10. 1)
   SQO(10 0010YONGPR0000 00000 0 0)


   APA(10 0090YDNG000000 00000 210) =              (  7. 1)
    AP(10 0010YDNG000000 00000 210)


   APA(10 0090YD3G000000 00000 0 0) =              (  7. 2)
    AP(10 0010YD2G000000 00000 0 0)


    AP(11 9910YONG000000 00000 0 0) =              ( 48. 1)
   NPX(10 0010Y9NG900000 00000 0 0)


    NA(10 0091Y999900000 00000 0 0) +              ( 41. 1)
    U-(10 0070000000000 00000 0 0) =
    XP(10 0010Y000010000 00000 0 0)


    NA(12 9990YDNGP00000 00000 0 0) +              ( 25. 1)
    AP(91 99909DNG000000 00000 0 0) +
    RD(90 00900DNGP00000 00000 0 0) +
    AS(91 9990000000000 00000 0 0) =
   NPB(10 0010YDNGP00000 00000 0 0)


    NS(10 0010Y2NGP00000 00000 0 0) +              ( 25. 2)
    AP(91 999092NG000000 00000 0 0) =
   NPB(10 0020Y2NGP00000 00000 0 0)


    RO(11 99109299900000 00000 0 0) +              ( 52. 1)
    Z-(91 99909000090000 00000 0 0) +
    NP(11 99109999900000 00000 0 0) +
    DP(94 99909000000000 00000 0 0) =
   VMD(10 00200000008000 00000 0 0)
```

112

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)          RULE NO.


  RO(19 99109299900000 00090 0 0) +          ( 51. 1)
  Z-(91 99909000090000 00000 0 0) +
  DP(94 99909000000000 00000 0 0) +
  KD(11 99909000000000 00000 0 0) =
 VMC(10 00200000000700 00000 0 0)


  RO(11 99109299900000 00000 0 0) +          ( 55. 1)
  Z-(10 00100000R00000 00000 0 0) +
  DP(94 99909000000000 00000 0 0) =
 VMI(10 00200000R4000 00000 0 0)


  RO(11 99109299900000 00000 0 0) +          ( 50. 1)
 NIP(11 99909000R0000 00000 0 0) +
  DP(94 99909000000000 00000 0 0) =
 VMB(10 00200000R4000 00000 0 0)


  RO(10 00109299900000 00000 0 0) +          ( 49. 1)
  Z-(91 99909000090000 00000 0 0) +
  DP(94 99909000000000 00000 0 0) =
 VMA(10 00200000003000 00000 0 0)


 BAA(10 009000NG000000 00000 0 0) =          ( 20. 1)
 BBA(10 001000NG000000 00000 0 0)


 BAB(10 0010003G000000 00000 0 0) =          ( 20. 2)
 BBA(10 0020003G000000 00000 0 0)


 BAC(10 0090003G000000 00000 0 0) =          ( 20. 3)
 BBA(10 0030003G000000 00000 0 0)
```

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                    RULE NO.

  BAD(10 0090003G000000 00000 0 0) =                    ( 20. 4)
  BBA(10 0040003G000000 00000 0 0)

  BAD(10 0090003G000000 00000 0 0) +                    ( 20. 5)
   C-(10 0010000000000 00000 0 0) +
  BAA(10 0090009G000000 99999 0 0) =
  BBA(10 0050003G000000 00000 0 0)

  BAE(10 0090003G000000 00000 0 0) =                    ( 21. 1)
  BBB(10 0010003G000000 00000 0 0)

  BAE(10 0090003G000000 00000 0 0) +                    ( 21. 2)
   C-(10 0010000000000 00000 0 0) +
  BBA(10 0090009G000000 00000 0 0) =
  BBB(10 0020003G000000 00000 0 0)

  BAF(10 0090003G000000 00000 0 0) =                    ( 23. 1)
  BBC(10 0010003G000000 00000 0 0)

  BAF(10 0090003G000000 00000 0 0) +                    ( 23. 2)
   C-(10 0010000000000 00000 0 0) +
   BA(10 0090003G000000 00000 0 0) =
  BBC(10 0020003G000000 00000 0 0)

  BBA(10 009000NG000000 00000 0 0) =                    ( 22. 1)
   BA(10 001000NG000000 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)             RULE NO.

  BBB(10 009000300000000 00000 0 0) =              ( 22. 2)
   BA(10 002000300000000 00000 0 0)

   BA(10 009000NG000000 00000 0 0) =               ( 24. 1)
   BP(10 001000NG000000 00000 0 0)

  BBC(10 009000300000000 00000 0 0) =              ( 24. 2)
   BP(10 002000300000000 00000 0 0)

   BP(10 009000NG000000 00000 0 0) +               ( 26. 1)
   J-(10 001000NGP00000 99999 0 0) +
   J-(94 99100099900000 99999 0 0) +
  NPB(10 00900D99900000 00000 0 0) +
   D-(94 99200000000000 99999 0 0) =
  NPA(10 00200DNGP00000 00000 0 0)

   BP(10 009000NG000000 00000 0 0) +               ( 26. 2)
  NPB(10 00900DNGP00000 00000 0 0) +
   D-(94 99200000000000 99999 0 0) =
  NPA(10 00100DNGP00000 00000 0 0)

   BP(10 00900099000000 00000 0 0) +               ( 30. 1)
   D-(10 00319000000000 99999 0 0) =
  DPA(10 00200000000000 00000 0 0)

   BP(10 009000NG000000 00000 0 0) =               ( 40. 1)
   BC(10 00100DNG000000 00000 4 9)
```

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                    RULE NO.

   NPB(10 00900DNGP00000 00000 0 0) +                  ( 26. 1)
    D-(94 99200000000000 99999 0 0) =
   NPA(10 00100DNGP00000 00000 0 0)

   NPA(10 00900DNGP00000 00000 0 0) +                  ( 29. 1)
   NAP(90 009002NGP00000 00000 0 0) =
    NP(10 00100DNGP00000 00000 0 0)

   NPA(12 99919DNGP00000 00000 0 0) =                  ( 27. 1)
   NPC(10 00100DNGP00000 00000 0 0)

    NP(10 001009NGP00000 00000 0 0) +                  ( 65. 1)
    L-(10 00120000000000 00000 0 0) +
    R-(10 003000NGP00000 00000 0 0) +
    VP(11 999000NGP99113 00000 0 0) =
   SAB(10 006010NGP00113 00000 0 0)

    NP(10 001009NGP00000 00000 0 0) +                  ( 65. 2)
    U-(10 00300000000000 00000 0 0) +
    R-(10 003000NGP00000 00000 0 0) +
    VP(11 999000NGP99113 00000 0 0) =
   SAB(10 006000NGP00113 00000 0 0)

    NP(12 99100DNGP00000 00000 0 0) =                  ( 45. 1)
   NSP(10 00100DNGP00000 00000 0 0)

    NP(11 9910YD99900000 00000 2 9) =                  ( 46. 1)
   NOP(10 0010YD00000000 00000 2 9)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                    RULE NO.

     NP(12 9910YDNGP00000 00000 0 0) =                ( 48. 1)
    NPX(10 0030YDNGP00000 00000 0 0)

    DPA(11 99119000000000 00000 0 0) =                ( 33. 1)
     DP(10 00300000000000 00000 0 0)

    DPB(11 99119000000000 00000 0 0) =                ( 33. 2)
     DP(10 00400000000000 00000 0 0)

    DPB(14 9910Y000000000 00000 0 0) =                ( 48. 1)
    NPX(10 0020Y999900000 00000 0 0)

    DPC(11 99119000000000 00000 0 0) =                ( 33. 1)
     DP(10 00500000000000 00000 0 0)

     DP(14 99909000000000 00000 0 0) +                ( 51. 1)
     RO(19 99109299900000 00000 0 0) +
     Z-(91 99909000090000 00000 0 0) +
     DP(94 99909000000000 00000 0 0) +
     KD(11 99909000000000 00000 0 0) =
    VMC(10 00200000000700 00000 0 0)

     DP(14 99909000000000 00000 0 0) +                ( 51. 2)
     Z-(91 99909000090000 00000 0 0) +
    NOP(91 99909900000000 00000 0 0) +
     DP(94 99909000000000 00000 0 0) +
     KD(11 99909000000000 00000 0 0) =
    VMC(10 00300000007000 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
YMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)          RULE NO.

  DP(14 99909000000000 00000 0 0) +          ( 52. 1)
  RO(11 99109299900000 00000 0 0) +
  Z-(91 99909000090000 00000 0 0) +
  NP(11 99109999900000 00000 0 0) +
  DP(94 99909000000000 00000 0 0) =
 VMD(10 00200000008000 00000 0 0)

  DP(14 99909000000000 00000 0 0) +          ( 52. 2)
  Z-(91 99909000090000 00000 0 0) +
 NOP(11 99909900000000 00000 0 0) +
  NP(11 99109999900000 00000 0 0) +
  DP(94 99909000000000 00000 0 0) =
 VMD(10 00300000008000 00000 0 0)

  DP(14 99909000000000 00000 0 0) +          ( 49. 1)
  RO(10 00109299900000 00000 0 0) +
  Z-(91 99909000090000 00000 0 0) +
  DP(94 99909000000000 00000 0 0) =
 VMA(10 00200000003000 00000 0 0)

  DP(14 99909000000000 00000 0 0) +          ( 49. 2)
  Z-(91 99909000090000 00000 0 0) +
 NOP(11 99909900000000 00000 0 0) +
  DP(94 99909000000000 00000 0 0) =
 VMA(10 00300000003000 00000 0 0)

  DP(14 99909000000000 00000 0 0) +          ( 50. 1)
  RO(11 99109299900000 00000 0 0) +
 NIP(11 99909000R00000 00000 0 0) +
  DP(94 99909000000000 00000 0 0) =
 VMB(10 00200000R4000 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)            RULE NO.


   DP(14 9990900000000 00000 0 0) +             ( 50. 2)
  NOP(11 9990990000000 00000 0 0) +
  NIP(11 999090000R0000 00000 0 0) +
   DP(94 9990900000000 00000 0 0) =
  VMB(10 00300000R4000 00000 0 0)


   DP(14 9990900000000 00000 0 0) +             ( 53. 1)
   P-(10 0010000010000 00000 0 0) +
   NV(10 0090000099000 00000 0 0) +
   DP(94 9990900000000 00000 0 0) =
   VM(10 0060000006000 00000 0 0)


   DP(14 9990900000000 00000 0 0) +             ( 55. 1)
   RO(11 9910929990000 00000 0 0) +
   Z-(10 00100000R0000 00000 0 0) +
   DP(94 9990900000000 00000 0 0) =
  VMI(10 00200000R4000 00000 0 0)


   DP(14 9990900000000 00000 0 0) +             ( 55. 2)
   Z-(10 00100000R0000 00000 0 0) +
  NOP(11 9990990000000 00000 0 0) +
   DP(94 9990900000000 00000 0 0) =
  VMI(10 00300000R4000 00000 0 0)


   DP(14 9990900000000 00000 0 0) +             ( 56. 1)
   VA(10 0010YONGP01VIT 99999 0 0) +
   VM(12 999000NGP01000 00000 0 0) =
   VP(10 0010YONGP01VIT 00000 0 0)


   DP(14 9990900000000 00000 0 0) +             ( 56. 2)
   VA(10 0010YONGPRAVIT 99999 114) +
   VM(11 999000999RA000 00000 114) =
   VP(10 0010YONGPRAVIT 00000 114)
```

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                    RULE NO.

   DP(14 99909000000000 00000 0 0) +                    ( 54. 1)
   Z-(11 99109000R0000 00000 0 0) +
   DP(94 99909000000000 00000 0 0) =
  VMR(10 004000000R5000 00000 0 0)

   DP(14 99909000000000 00000 0 0) +                    ( 53. 1)
   XP(11 99909000R0000 00000 0 0) +
   DP(94 99909000000000 00000 0 0) =
   VM(10 00500000R5000 00000 0 0)

   DP(14 99909000000000 00000 0 0) +                    ( 53. 2)
  NPX(11 999099NGP00000 00000 0 0) +
   DP(94 99909000000000 00000 0 0) =
   VM(10 001000NGP01000 00000 0 0)

   DP(14 99909000000000 00000 0 0) +                    ( 51. 1)
   KD(11 99909000000000 00000 0 0) =
  VMC(10 00300000007000 00000 0 0)

   DP(14 99909000000000 00000 0 0) =                    ( 54. 1)
  VMR(10 00100000001000 00000 0 0)

   DP(14 99909000000000 00000 0 0) =                    ( 53. 1)
   VM(10 00200000002000 00000 0 0)

   EA(10 0012YDNGPRAV00 99999 0 0) +                    ( 62. 1)
   VM(11 999090999RA000 00000 0 0) =
  EPA(10 0010YDNGPRAV00 00000 0 0)
```

120

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                RULE NO.

   EA(10 001000NGPR1100 SWWWW 0 0) =                  ( 36. 1)
  VBB(10 002000NGPR1113 SWWWW 0 0)

   EA(10 0010Y1NGPRAV00 SWWWW 0 0) =                  ( 36. 2)
  VBB(10 0010YONGPRAVI3 SWWWW 0 0)

   VB(10 0010YONGPRAVIT SWWWW 517) =                  ( 36. 3)
  VBB(10 0010YONGPRAVIT SWWWW 517)

  VBB(10 0090YONGPRAVIT SWWWW1117) +                  ( 37. 1)
   W-(10 0010000000RAV00 SWWWW 0 0) =
   VC(10 0020YONGPRAVIT SWWWW1117)

  VBB(10 0090YONGPRAVIT SWWWW1117) =                  ( 37. 2)
   VC(10 0030YONGPRAVIT SWWWW1117)

   VC(10 0090YONGPRAVIT SWWWW1117) =                  ( 38. 1)
  VAA(10 0010YONGPRAVIT SWWWW1117)

   VC(10 0030YONGP01VI212HYH* 0 0) +                  ( 38. 2)
   VC(10 009000NGPRAVI3 SWWWW 0 0) =
  VAA(10 0010YONGPRAVI5 SWWWW 0 0)

   VC(10 0030YONGP01VI112HYH* 0 0) +                  ( 38. 3)
   VC(10 009000NGPRAVI3 SWWWW 0 0) =
  VAA(10 0010YONGPRAVI4 SWWWW 0 0)
```

```
SYMBOL(MF/KBCLYONGPRAVIT/S/-W-/X/Z)          RULE NO.


   VAA(10 0010YONGPRAVIT SWWWW 210) =         ( 39. 1)
    VA(10 0010YONGPRAVIT SWWWW 210)


   VAA(10 0010Y03GPRAVIT SWWWW 0 0) =         ( 39. 2)
    VA(10 0010Y02GPRAVIT SWWWW 0 0)


    VA(10 0010YONGP01VIT 99999 0 0) +         ( 64. 1)
    XP(11 999000001000 00000 0 0) +
   NSP(12 999099NGP00000 00000 0 0) +
    DP(94 999090000000 00000 0 0) =
   SAA(10 0030YONGP00VIT 00000 317)


    VA(10 0010YONGPRAVIT 99999 0 0) +         ( 57. 1)
   VMR(11 999090999RA000 00000 0 0) =
   VRB(10 0010YONGPRAVIT 00000 0 0)


    VA(10 0010YONGPR4VIT 99999 0 0) +         ( 58. 1)
   VMI(11 999090000R4000 00000 0 0) =
   VRI(10 0010YONGPR4VIT 00000 0 0)


    VA(10 0010YONGPRAVIT 99999 114) +         ( 56. 1)
    VM(11 999000999RA000 00000 114) =
    VP(10 0010YONGPRAVIT 00000 114)


    VA(10 0010YONGP01VIT 99999 0 0) +         ( 56. 2)
    VM(12 999000NGP01000 00000 0 0) =
    VP(10 0010YONGP01VIT 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                     RULE NO.


    BC(10 00900DNG000000 00000 0 0) =                  ( 26. 1)
   NPA(10 00300DNG900000 00000 0 0)


    XP(11 99900000010000 00000 0 0) +                  ( 64. 1)
    VA(10 0010YONGP01VIT 99999 0 0) +
   NSP(12 999099NGP00000 00000 0 0) +
    DP(94 99909000000000 00000 0 0) =
   SAA(10 0010YONGP00VIT 00000 317)


    XP(11 99900000010000 00000 0 0) +                  ( 64. 2)
   NSP(12 999099NGP00000 00000 0 0) +
    DP(94 99909000000000 00000 0 0) =
   SAA(10 004000NGP00113 00000 0 0)


    XP(11 999090000R0000 00000 0 0) +                  ( 53. 1)
    DP(94 99909000000000 00000 0 0) =
    VM(10 005000000R5000 00000 0 0)


    XP(11 9990Y000090000 00000 0 0) =                  ( 72. 1)
    KC(10 0040Y000000000 00000 0 0)


    XP(11 9990Y000090000 00000 0 0) =                  ( 48. 1)
   NPX(10 0050Y999900000 00000 0 0)


    XP(11 9990Y0000R0000 00000 0 0) =                  ( 47. 1)
   NIP(10 0010Y0000R0000 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                   RULE NO.

  XP(14 9990900090000 00000 0 0) =                   ( 28. 1)
 NAP(10 0020099900000 00000 0 0)

  XP(11 9990Y00090000 00000 0 0) =                   ( 33. 1)
  DP(10 0070000000000 00000 0 0)

  NO(11 9990Y20000000 00000 0 0) =                   ( 46. 1)
 NOP(10 0020Y20000000 00000 0 0)

 RSP(12 999002NGP00000 00000 0 0) =                  ( 45. 1)
 NSP(10 00200DNGF00000 00000 0 0)

 RSP(12 9990Y2NGP00000 00000 0 0) =                  ( 48. 1)
 NPX(10 0040YDNGP00000 00000 0 0)

 NSP(12 999099NGP00000 00000 0 0) +                  ( 64. 1)
  VA(10 0010Y0NGP01VIT 99999 0 0) +
  XP(11 9990000010000 00000 0 0) +
  DP(94 9990900000000 00000 0 0) =
 SAA(10 0020Y0NGP00VIT 00000 317)

 NSP(12 999099NGP00000 00000 0 0) +                  ( 65. 1)
  VP(11 9990Y0NGP99VIT 00000 0 0) =
 SAB(10 0010Y0NGP00VIT 00000 0 0)

 NSP(12 999099NGP00000 00000 0 0) +                  ( 68. 1)
 VRB(11 9990Y0NGP99999 00000 0 0) =
 SRO(10 0020Y0NGP00000 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                    RULE NO.


    NSP(12 999099NGP00000 00000 0 0) +                  ( 69. 1)
    VRI(11 9990YONGPR9999 00000 0 0) =
    SRI(10 0020YONGPR0000 00000 0 0)


    NSP(12 999099NGP00000 00000 0 0) +                  ( 10. 1)
    VQO(11 9910YONGPR9999 00000 0 0) =
    SQO(10 0020YONGPR0000 00000 0 0)


    NSP(10 009009NGP00000 00000 0 0) +                  ( 65. 1)
     VP(11 999000NGP01113 00000 0 0) =
    SAB(10 002000NGP00113 00000 0 0)


    NOP(11 99909900000000 00000 0 0) +                  ( 52. 1)
     NP(11 99109999900000 00000 0 0) +
     DP(94 99909000000000 00000 0 0) =
    VMD(10 00300000008000 00000 0 0)


    NOP(11 99909900000000 00000 0 0) +                  ( 51. 1)
     DP(94 99909000000000 00000 0 0) +
     KD(11 99909000000000 00000 0 0) =
    VMC(10 00300000007000 00000 0 0)


    NOP(11 99909900000000 00000 0 0) +                  ( 50. 1)
    NIP(11 99909000R0000 00000 0 0) +
     DP(94 99909000000000 00000 0 0) =
    VMB(10 003000000R4000 00000 0 0)


    NOP(11 99909900000000 00000 0 0) +                  ( 49. 1)
     DP(94 99909000000000 00000 0 0) =
    VMA(10 00300000003000 00000 0 0)
```

SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z) RULE NO.

```
NPX(11 999099NGP00000 00000 0 0) +          ( 53. 1)
 DP(94 999090000000000 00000 0 0) =
 VM(10 001000NGP01000 00000 0 0)

VMA(10 00900000003000 00000 0 0) =          ( 53. 2)
 VM(10 00300000003000 00000 0 0)

VMA(10 00900000003000 00000 5 6) =          ( 54. 1)
VMR(10 00200000003000 00000 0 0)

VMB(10 0090000000R4000 00000 0 0) =         ( 53. 1)
 VM(10 0040000000R4000 00000 0 0)

VMB(10 00900000004000 00000 5 6) =          ( 54. 1)
VMR(10 00300000004000 00000 0 0)

VMC(10 00900000007000 00000 0 0) =          ( 53. 1)
 VM(10 00700000007000 00000 0 0)

VMD(10 00900000008000 00000 0 0) =          ( 53. 2)
 VM(10 00800000008000 00000 0 0)

VMD(10 00900000008000 00000 5 6) =          ( 54. 1)
VMR(10 00500000008000 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                RULE NO.


  VP(10 0010YONGP99VIT 00000 317) =               ( 66. 1)
 SAC(10 0010YONGP00VIT 00000 317)


  VP(10 0010YONGPR3VIT 00000 0 0) =               (  5. 1)
 VQO(10 0010YONGPR8VIT 00000 0 0)


  VP(10 0010YONGPR2VIT 00000 0 0) =               (  5. 2)
 VQO(10 0010YONGPR3VIT 00000 0 0)


  VP(10 0010YONGPR5VIT 00000 0 0) =               (  5. 3)
 VQO(10 0010YONGPR4VIT 00000 0 0)


 VRB(11 9910YONGP99999 00000 0 0) =               ( 68. 1)
 SRO(10 0010YONGP00000 00000 0 0)


 VRI(11 9910YONGPR9999 00000 0 0) =               ( 69. 1)
 SRI(10 0010YONGPR0000 00000 0 0)


  NV(11 9990Y000099000 00000 0 0) =               ( 47. 1)
 NIP(10 0020Y0000R0000 00000 0 0)


  NV(10 00900000099000 00000 0 0) =               ( 45. 1)
 NSP(10 00400D11300000 00000 0 0)


 EPB(10 0090YDNGPR3100 00000 0 0) =               ( 62. 1)
 EPA(10 0020YDNGPR3100 00000 0 0)
```

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                 RULE NO.

 EPA(10 0010YD3GP99900 00000 0 0) =                 ( 63. 1)
  EP(10 0010YD2GP00000 00000 0 0)

 EPA(10 0090YDNGP99900 00000 210) =                 ( 63. 2)
  EP(10 0010YDNGP00000 00000 210)

  EP(10 0010YDNGP00000 00000 0 0) =                 ( 25. 1)
 NPB(10 0030YDNGP00000 00000 0 0)

 SAA(10 0090YONGP00VIT 00000 0 0) =                 ( 67. 1)
  SA(10 0010YONGP00VIT 00000 0 0)

 SAB(10 0090YONGP00VIT 00000 0 0) =                 ( 67. 2)
  SA(10 0030YONGP00VIT 00000 0 0)

 SAC(10 0010YONGP00VIT 00000 0 0) =                 ( 67. 3)
  SA(10 0020YONGP00VIT 00000 0 0)

  SA(11 9990Y0999009IT 00000 0 0) +                 ( 76. 1)
  T-(90 0030000000000 00000 0 0) +
  KC(11 9990900000000 00000 0 0) =
  SD(10 0020Y0000001T 00000 0 0)

  SA(11 9990Y0999009IT 00000 0 0) =                 ( 77. 1)
  S-(10 0010Y0000000IT 00000 0 0)
```

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                    RULE NO.


    RG(11 99909DNGP00000 00000 0 0) =                   ( 28. 1)
   NAP(10 00300DNGP00000 00000 0 0)


    KN(10 0090Y000000000 00000 0 0) =                   ( 75. 1)
    KD(10 0010Y000000000 00000 0 0)


    KN(10 00900000000000 00000 0 0) =                   ( 45. 1)
   NSP(10 00300211300000 00000 0 0)


    KC(11 99909000000000 00000 0 0) +                   ( 76. 1)
    T-(90 00300000000000 00000 0 0) +
    SA(11 9990Y0999009IT 00000 0 0) =
    SD(10 0010Y00000001T 00000 0 0)


    KC(11 99909000000000 00000 0 0) +                   ( 78. 1)
    T-(10 00300000000000 00000 0 0) +
    KI(10 0090Y099990000 00000 0 0) =
    SI(10 0020Y000000000 00000 0 0)


    KC(11 9990Y000000000 00000 0 0) =                   ( 48. 1)
   NPX(10 0060YDNGP00000 00000 0 0)


    KK(11 99109000000001 00000 0 0) +                   ( 77. 1)
    T-(90 00300000000000 00000 0 0) +
    SA(11 9990Y0999009I1 00000 0 0) =
    S-(10 0030Y0000000I1 00000 0 0)
```

```
(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)              RULE NO.

(11 99109000000002 00000 0 0) +             ( 77. 2)
(90 00300000000000 00000 0 0) +
(11 9990Y099900912 00000 0 0) =
(10 0030Y000000012 00000 0 0)

(10 0090Y099990000 00000 0 0) +             ( 78. 1)
(10 00300000000000 00000 0 0) +
(11 99909000000000 00000 0 0) =
(10 0030Y000000000 00000 0 0)

(10 0090Y099990000 00000 0 0) =             ( 78. 2)
(10 0010Y000000000 00000 0 0)

(10 0090Y0000000IT 00000 0 0) =             ( 77. 1)
(10 0020Y0000000IT 00000 0 0)

(10 0090Y000000022 00000 0 0) +             ( 79. 1)
(10 00700000000000 00000 0 0) =
(10 0030Y000000000 00000 0 0)

(10 0090Y000000099 00000 0 0) +             ( 79. 2)
(10 00600000000000 00000 0 0) =
(10 0010Y000000000 00000 0 0)

(11 9990Y000000000 00000 0 0) +             ( 79. 3)
(10 00500000000000 00000 0 0) =
(10 0020Y000000000 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                    RULE NO.


   A-(10 0011YDNG000000 99999 0 0) +                   (  6. 1)
   XP(11 99900000090000 00000 913) =
  APA(10 0020Y9NG000000 00000 0 0)


   A-(10 0011Y2NG000000 99999 0 0) +                   (  6. 2)
   XP(11 99900000020000 00000 0 0) =
  APA(10 0030Y2NG000000 00000 0 0)


   A-(10 0011YDNG000000 99999 0 0) +                   (  6. 3)
   D-(94 90600000000000 99999 0 0) =
  APA(10 0010YDNG000000 00000 0 0)


   B-(10 0010001G000000 00000 0 0) +                   ( 16. 1)
   B-(10 0040001G000000 00000 0 0) =
  BAC(10 0010003G000000 00000 0 0)


   B-(10 0010001G000000 00000 0 0) =                   ( 14. 1)
  BAA(10 0010001G000000 00000 0 0)


   B-(10 0020003G000000 00000 0 0) +                   ( 16. 1)
   B-(10 0040001G000000 00000 0 0) =
  BAC(10 0020003G000000 00000 0 0)


   B-(10 0020002G000000 00000 0 0) =                   ( 14. 1)
  BAA(10 0020002G000000 00000 0 0)


   B-(10 00300011000000 99999 0 0) +                   ( 16. 1)
   B-(10 00400012000000 00000 0 0) =
  BAC(10 00300032000000 00000 0 0)
```

```
YMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                    RULE NO.


 B-(10 0030001200000O 99999 0 0) +                  ( 16. 2)
 B-(10 00400011000000 00000 0 0) =
BAC(10 00300031000000 00000 0 0)


 B-(10 00300012000000 99999 0 0) =                  ( 14. 1)
BAA(10 00300031000000 00000 0 0)


 B-(10 00300011000000 99999 0 0) =                  ( 14. 2)
BAA(10 00300032000000 00000 0 0)


 B-(10 00300031000000 99999 0 0) =                  ( 17. 1)
BAD(10 00200039000000 00000 0 0)


 B-(10 00400031000000 00000 0 0) =                  ( 17. 2)
BAD(10 00100039000000 00000 0 0)


 B-(10 00400012000000 00000 0 0) =                  ( 15. 1)
BAB(10 00100031000000 00000 0 0)


 B-(10 00400011000000 00000 0 0) =                  ( 15. 2)
BAB(10 00100032000000 00000 0 0)


 B-(10 00500012000000 00000 0 0) =                  ( 18. 1)
BAE(10 00100039000000 00000 0 0)
```

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                        RULE NO.


   B-(10 0050002200000 00000 0 0) =                       ( 18. 2)
  BAE(10 0020003900000 00000 0 0)


   B-(10 0060001100000 00000 0 0) =                       ( 19. 1)
  BAF(10 0010003900000 00000 0 0)


   B-(10 0060002100000 00000 0 0) =                       ( 19. 2)
  BAF(10 0020003900000 00000 0 0)


   C-(10 0030000000000 00000 0 0) +                       ( 71. 1)
   SA(11 9990Y099900999 00000 0 0) =
   KN(10 0010Y000000000 00000 0 0)


   C-(10 0040000000000 99999 0 0) +                       ( 72. 1)
   SA(11 9990Y099900999 00000 0 0) =
   KC(10 0010Y000000000 00000 0 0)


   C-(10 0050000000000 99999 0 0) +                       ( 72. 2)
   SA(11 9990Y099900999 00000 817) =
   KC(10 0020Y000000000 00000 0 0)


   C-(10 0060000000000 99999 0 0) +                       ( 72. 3)
   SA(11 9990Y099900999 00000 0 0) =
   KC(10 0030Y000000000 00000 0 0)


   C-(10 0080000000000 99999 0 0) +                       ( 79. 1)
   S-(10 0090Y000000099 00000 0 0) +
   T-(10 0060000000000 00000 0 0) =
   SC(10 0010Y000000000 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                    RULE NO.


   C-(10 0060000000000 99999 0 0) +                   ( 79. 2)
  SI(11 9990Y000000000 00000 0 0) +
   T-(10 0050000000000 00000 0 0) =
  SC(10 0020Y000000000 00000 0 0)


   C-(10 0060000000000 99999 0 0) +                   ( 79. 3)
   S-(10 0090Y000000022 00000 0 0) +
   T-(10 0070000000000 00000 0 0) =
  SC(10 0030Y000000000 00000 0 0)


   D-(11 9911900000000 99999 0 0) =                   ( 33. 1)
  DP(10 0010000000000 00000 0 0)


   D-(11 9911Y000000000 99999 0 0) =                  ( 43. 1)
 DPD(10 0010Y000000000 00000 0 0)


   D-(11 9921900000000 99999 0 0) =                   ( 33. 1)
  DP(10 0020000000000 00000 0 0)


   D-(11 9921Y000000000 99999 0 0) =                  ( 43. 1)
 DPD(10 0020Y000000000 00000 0 0)


   D-(10 0031900000000 99999 0 0) +                   ( 30. 1)
  BP(90 0090009900000 00000 0 0) =
 DPA(10 0010000000000 00000 0 0)


   D-(10 0041900000000 99999 0 0) +                   ( 31. 1)
   D-(90 0060000000000 99999 0 0) =
 DPB(10 0010000000000 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                    RULE NO.

  D-(10 00519000000000 99999 0 0) +                    ( 32. 1)
  D-(90 00600000000000 99999 0 0) =
DPC(10 00100000000000 00000 0 0)

  D-(10 00619000000000 99999 0 0) +                    ( 33. 1)
  D-(90 00600000000000 99999 0 0) =
 DP(10 00600000000000 00000 0 0)

  D-(10 0071Y000000000 00000 0 0) +                    ( 38. 1)
 VC(10 009000NGPRAVI1 SWWWW 0 0) =
VAA(10 0010YONGPRAVI6 SWWWW 0 0)

  D-(10 0081Y000000000 00000 0 0) +                    ( 38. 2)
 VC(10 009000NGPRAVI2 SWWWW 0 0) =
VAA(10 0010YONGPRAVI7 SWWWW 0 0)

  E-(10 0011YONGPRAV00 SWWWW 2 9) =                    ( 34. 1)
 EA(10 0010Y9NGPRAV00 SWWWW 2 9)

  G-(11 9912YONGP03100 99999 0 0) +                    ( 61. 1)
NPA(11 99909D99900000 00000 0 0) =
EPB(10 0010YDNGPR3100 00000 0 0)

  G-(10 0012YONGP03100 99999 0 0) +                    ( 61. 2)
  R-(10 00300099900000 00000 0 0) =
EPB(10 0020Y2NGPR3100 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                RULE NO.


H-(10 0011Y000000000 00000 0 0) +                  (  9. 1)
N-(10 001000NGP00000 99999 0 0) =
NA(10 0020Y2NGP00000 00000 0 0)


H-(10 0011Y000000000 00000 0 0) +                  ( 12. 1)
R-(10 001000NG300000 00000 0 0) =
RD(10 0020Y2NG300000 00000 0 0)


H-(10 0011Y000000000 00000 0 0) +                  ( 12. 2)
R-(10 002000NG300000 00000 0 0) =
RD(10 0030Y2NG300000 00000 0 0)


H-(10 0011Y000000000 00000 0 0) +                  ( 34. 1)
E-(10 001000NGPRAV00 SWWWW 0 0) =
EA(10 0010Y2NGPRAV00 SWWWW 0 0)


H-(10 0010000000000 00000 0 0) +                   ( 40. 1)
BP(10 009000NG000000 00000 0 0) =
BC(10 002002NG000000 00000 0 0)


H-(10 0010000000000 00000 0 0) +                   ( 80. 1)
A-(10 001000NG000000 SWWWW 0 0) =
A-(10 001002NG000000 SWWWW 0 0)


I-(10 0020003G000000 00000 0 0) +                  ( 16. 1)
B-(10 0040001G000000 00000 0 0) =
BAC(10 0020003G000000 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)          RULE NO.

   I-(10 0020003200000 00000 0 0) +          ( 18. 1)
   B-(10 0050003200000 00000 0 0) =
  BAE(10 0020003900000 00000 0 0)

   I-(10 0020003100000 00000 0 0) +          ( 19. 1)
   B-(10 0060003100000 00000 0 0) =
  BAF(10 0020003900000 00000 0 0)

   I-(10 0030001100000 99999 0 0) +          ( 18. 1)
   B-(10 0050003200000 00000 0 0) =
  BAE(10 0030003900000 00000 0 0)

   I-(10 0030003100000 99999 0 0) +          ( 19. 1)
   B-(10 0060003100000 00000 0 0) =
  BAF(10 0030003900000 00000 0 0)

   J-(10 001000NGP00000 99999 0 0) +         ( 26. 1)
   J-(94 9910009990000 99999 0 0) +
  NPB(10 00900D99900000 00000 0 0) +
   D-(94 9920000000000 99999 0 0) =
  NPA(10 00200DNGP00000 00000 0 0)

   J-(10 0011Y0NGP00000 99999 0 0) +         ( 11. 1)
   R-(10 0030009990000 00000 0 0) =
   NS(10 0010Y2NGP00000 00000 0 0)

   L-(10 0012000000000 00000 0 0) +          ( 64. 1)
   XP(11 9990000001000 00000 0 0) +
  NSP(12 999099NGP00000 00000 0 0) +
   DP(94 9990900000000 00000 0 0) =
  SAA(10 004010NGP00113 00000 0 0)
```

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                RULE NO.


  L-(10 0012000000000 00000 0 0) +                  ( 65. 1)
  R-(10 003000NGP00000 00000 0 0) +
  VP(11 999000NGP99113 00000 0 0) =
 SAB(10 004010NGP00113 00000 0 0)


  L-(10 0012000000000 00000 0 0) +                  ( 65. 2)
  NP(10 001009NGP00000 00000 0 0) +
  VP(11 999000NGP99113 00000 0 0) =
 SAB(10 005010NGP00113 00000 0 0)


  L-(10 001L0000000000 00000 0 0) +                 (  2. 1)
  F-(1F 00CL0DNGPRAVIT SWWWW 0 0) =
  F-(1F 00CL1DNGPRAVIT SWWWW 0 0)


  N-(10 0011Y0NGP00000 99999 0 0) +                 (  9. 1)
  B-(10 001000NG000000 00000 0 0) =
  NA(10 0010Y1NGP00000 00000 0 0)


  N-(10 0011Y0NGP00000 99999 2 9) =                 (  9. 2)
  NA(10 0010Y9NGP00000 00000 2 9)


  N-(14 9991Y0NGP00000 99999 1 6) =                 (  9. 3)
  NA(10 0030Y2NGP00000 00000 0 0)


  O-(10 0011Y000000000 00000 0 0) +                 ( 13. 1)
  R-(10 003000NGP00000 00000 0 0) =
  RO(10 0010Y2NGP00000 00000 0 0)
```

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                    RULE NO.

   O-(10 0011Y000000000 00000 0 0) +                   ( 42. 1)
   NP(10 0010029990000 00000 0 0) =
   NO(10 0010Y200000000 00000 0 0)

   O-(10 0011Y000000000 00000 0 0) +                   ( 42. 2)
   RG(10 0090029990000 00000 0 0) =
   NO(10 0020Y200000000 00000 0 0)

   P-(10 00100000R0000 00000 0 0) +                    ( 74. 1)
   R-(10 00500000000000 99999 0 0) +
  SRI(10 0090YONGPR0000 00000 0 0) =
   KI(10 0040YONGPR0000 00000 0 0)

   P-(10 00100000010000 00000 0 0) +                   ( 53. 1)
   NV(10 00900000099000 00000 0 0) +
   DP(94 99909000000000 00000 0 0) =
   VM(10 00600000006000 00000 0 0)

   P-(10 0011Y0000R0000 00000 0 0) +                   (  4. 1)
   R-(10 00300099900000 00000 0 0) =
   Z-(10 0010Y0000R0000 00000 0 0)

   P-(10 0011Y0000R0000 00000 0 0) +                   ( 41. 1)
   NP(11 99100999900000 00000 0 0) =
   XP(10 0010Y0000R0000 00000 0 0)

   P-(10 0011Y0000R0000 00000 0 0) +                   ( 41. 2)
   RG(11 99900999900000 00000 0 0) =
   XP(10 0030Y0000R0000 00000 0 0)
```

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)          RULE NO.

   W-(10 00900000000000 00000 0 0) +           ( 74. 1)
   SA(11 9990YONGP00999 00000 0 0) =
   KI(10 0010YONGP00000 00000 0 0)

   R-(12 991000NGP00000 00000 0 0) =           ( 44. 1)
  RSP(10 002002NGP00000 00000 0 0)

   R-(10 0011YONG300000 00000 0 0) =           ( 12. 1)
   RD(10 0010YDNG900000 00000 0 0)

   R-(10 002000NGP00000 00000 0 0) +           ( 44. 1)
  NPC(93 991002NGP00000 00000 0 0) =
  RSP(10 001002NGP00000 00000 0 0)

   R-(10 002000NG300000 00000 0 0) =           ( 36. 1)
  VBB(10 003000NG3R1113 SWWWW 0 0)

   R-(10 00300099900000 00000 0 0) +           ( 55. 1)
   DP(94 99909000000000 00000 0 0) +
   Z-(10 001000000R0000 00000 0 0) +
   DP(94 99909000000000 00000 0 0) =
  VMI(10 001000000R4000 00000 0 0)

   R-(10 00300099900000 00000 0 0) +           ( 52. 1)
   DP(94 99909000000000 00000 0 0) +
   Z-(91 99909000090000 00000 0 0) +
   NP(11 99109999900000 00000 0 0) +
   DP(94 99909000000000 00000 0 0) =
  VMD(10 00100000008000 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z) RULE NO.

```
 R-(10 0030009990000 00000 0 0) +            ( 51. 1)
 Z-(91 9990900009U000 00000 0 0) +
 DP(94 9990900000U000 00000 0 0) +
 KD(11 99909000000000 00000 0 0) =
VMC(10 00100000007000 00000 0 0)

 R-(10 00300099900000 U0000 0 0) +           ( 50. 1)
 DP(94 99909000000000 00000 0 0) +
NIP(11 999090000R0000 00000 0 0) +
 DP(94 99909000000000 00000 0 0) =
VMB(10 0010000000R4000 00000 0 0)

 R-(10 00300099900000 00000 0 0) +           ( 49. 1)
 DP(94 99909000000000 00000 0 0) +
 Z-(91 99909000090000 00000 0 0) +
 DP(94 99909000000000 00000 0 0) =
VMA(10 00100000003000 00000 0 0)

 R-(10 0041Y000000000 00000 0 0) +           ( 70. 1)
 VP(11 999090NGP00999 00000 0 0) =
 RG(10 0010YDNGP00000 00000 0 0)

 R-(10 0041Y000000000 00000 0 0) +           ( 70. 2)
SRO(10 00909099900000 00000 0 0) =
 RG(10 0020Y999900000 00000 0 0)

 R-(10 0041Y000000000 00000 0 0) +           ( 70. 3)
SRI(10 00909099900000 00000 0 0) =
 RG(10 0030Y999900000 00000 0 0)
```

SM1

4-A-31

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)   RULE NO.

```
  R-(10 0040000000000 00000 0 0) +          ( 71. 1)
  SA(11 9990Y099900999 00000 0 0) =
  KN(10 0020Y000000000 00000 0 0)

  R-(10 0050000000000 99999 0 0) +          ( 74. 1)
  VP(11 9990YONGP09999 00000 0 0) =
  KI(10 0020YONGP00000 00000 0 0)

  R-(10 0050000000000 99999 0 0) +          ( 74. 2)
 SQO(10 0090YONGPR0000 00000 0 0) =
  KI(10 0030YONGPR0000 00000 0 0)

  T-(10 0030000000000 00000 0 0) +          ( 28. 1)
 NPC(13 99119DNGP00000 00000 0 0) +
  T-(10 0030000000000 00000 0 0) =
 NAP(10 00100DNGP00000 00000 0 0)

  T-(10 0040000000000 00000 0 0) +          ( 75. 1)
  T-(10 0010000000000 00000 0 0) +
  SC(11 9990Y000000000 00000 0 0) +
  T-(10 0020000000000 00000 0 0) =
  KD(10 0020Y000000000 00000 0 0)

  U-(10 0010000000000 00000 0 0) +          (  8. 1)
  R-(10 0030009990000 00000 0 0) =
  AS(10 0010000000000 00000 0 0)

  U-(10 0010000000000 00000 0 0) +          (  8. 2)
  NP(10 0010099900000 00000 0 0) =
  AS(10 0020000000000 00000 0 0)
```

LISTING OF MODERN HEBREW SY,,TAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                    RULE NO.


   U-(10 0030000000000O 00000 0 0) +                    ( 64. 1)
   XP(11 99900000010000 00000 0 0) +
  NSP(12 999099NGP00000 00000 0 0) +
   DP(94 99909000000000 00000 0 0) =
  SAA(10 004000NGP00113 00000 0 0)


   U-(10 00300000000000 00000 0 0) +                    ( 65. 1)
  NSP(10 009009NGP00000 00000 0 0) +
   VP(11 999000NGP01113 00000 0 0) =
  SAB(10 002000NGP00113 00000 0 0)


   U-(10 00300000000000 00000 0 0) +                    ( 65. 2)
   R-(10 003000NGP00000 00000 0 0) +
   VP(11 999000NGP99113 00000 0 0) =
  SAB(10 003000NGP00113 00000 0 0)


   U-(10 00400000000000 00000 0 0) +                    ( 73. 1)
   SA(11 99900099900911 00000 0 0) =
   KK(10 00100000000001 00000 0 0)


   U-(10 00500000000000 00000 0 0) +                    ( 73. 2)
   SA(11 99900099900911 00000 0 0) =
   KK(10 00101000000001 00000 0 0)


   U-(10 00600000000000 00000 0 0) +                    ( 73. 3)
   SA(11 9990Y099900912 00000 0 0) =
   KK(10 0010Y000000002 00000 0 0)


   V-(10 0014YONGPRAV12 SWWWW 0 0) +                    ( 35. 1)
   U-(10 00200000000000 00000 0 0) =
   VB(10 0010YONGPRAV32 SWWWW 0 0)
```

4-A-33

SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z) RULE NO.

```
V-(10 001000NGPRAV22 SWWWW 0 0) +          ( 35. 2)
U-(10 00200000000000 00000 0 0) =
VB(10 001000NGPRAV32 SWWWW 0 0)

V-(10 001410NGPRAV22 SWWWW 0 0) =          ( 35. 3)
VB(10 001010NGPRAV32 SWWWW 0 0)

V-(10 0011YONG2RAV22 SWWWW 0 0) =          ( 35. 4)
VB(10 0010YONG2RAV22 SWWWW 0 0)

V-(10 0011YONGPRAV1T SWWWW 0 0) =          ( 35. 5)
VB(10 0010YONGPRAV1T SWWWW 0 0)

W-(10 0010000000RAV00 SWWWW 0 0) +         ( 37. 1)
VBB(10 0090YONGPRAVIT SWWWW1117) =
VC(10 0010YONGPRAVIT SWWWW1117)

W-(10 0011Y0000RAV00 SWWWW 0 0) =          ( 35. 1)
VB(10 0010Y0992RAV22 SWWWW 0 0)

W-(11 9913Y0000RA000 99999 0 0) +          ( 60. 1)
VM(10 009000000RA000 00000 0 0) =
NW(10 0010Y0000RA000 00000 0 0)

Y-(10 0013Y0000RA000 99999 0 0) +          ( 59. 1)
VM(11 999090999RA000 00000 0 0) =
NV(10 0010Y0000RA000 00000 0 0)
```

LISTING OF MODERN HEBREW SYNTAX ANALYSIS RULES 7/21/70--JAMES D. PRICE

```
SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                     RULE NO.


   Y-(10 0013Y0000RA000 99999 0 0) +                    ( 59. 2)
   R-(10 00300099900000 00000 0 0) +
   VM(11 999090999RA000 00000 0 0) =
   NV(10 0020Y0000RA000 00000 0 0)


   Y-(10 0013Y0000RA000 99999 0 0) +                    ( 59. 3)
  NSP(11 99909999900000 00000 0 0) +
   VM(11 999090999RA000 00000 0 0) =
   NV(10 0030Y0000RA000 00000 0 0)


   F-(11-1BCLYDNGPRAVIT SWWWW 0 0) +                    (  3. 1)
   T-(10 00300000000000 00000 0 0) +
   F-(11 00CLYDNGPRAVIT SWWWW 0 0) =
   F-(11 KBCLYDNGPRAVIT SWWWW 0 0)


   F-(11-10CLYDNGPRAVIT SWWWW 0 0) +                    (  3. 2)
   C-(10 00100000000000 00000 0 0) +
   F-(11 00CLYDNGPRAVIT SWWWW 0 0) =
   F-(11 K1CLYDNGPRAVIT SWWWW 0 0)


   F-(11-10CLYDNGPRAVIT SWWWW 0 0) +                    (  3. 3)
   C-(10 00200000000000 00000 0 0) +
   F-(11 00CLYDNGPRAVIT SWWWW 0 0) =
   F-(11 K2CLYDNGPRAVIT SWWWW 0 0)


   F-(12-10CLYD999RAVIT SWWWW 0 0) +                    (  3. 4)
   T-(10 00300000000000 00000 0 0) +
   F-(12 0BCLYD999RAVIT SWWWW 0 0) =
   F-(12 KBCLYDNGPRAVIT SWWWW 0 0)
```

SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z) RULE NO.

```
F-(12-10CLYD999RAVIT SWWWW 0 0) +          ( 3. 5)
C-(10 0010000000000 00000 0 0) +
F-(12 00CLYD999RAVIT SWWWW 0 0) =
F-(12 K1CLYDNGPRAVIT SWWWW 0 0)


F-(12-10CLYD999RAVIT SWWWW 0 0) +          ( 3. 6)
C-(10 0020000000000 00000 0 0) +
F-(12 00CLYD999RAVIT SWWWW 0 0) =
F-(12 K2CLYDNGPRAVIT SWWWW 0 0)


F-(13 00CLYDMGPRAVIT SWWWW 0 0) +          ( 3. 7)
T-(10 0030000000000 00000 0 0) +
F-(13 00CLYDMGPRAVIT SWWWW 0 0) =
F-(13 20CLYDNGPRAVIT SWWWW 0 0)


F-(13-10CLYDNGPRAVIT SWWWW 0 0) +          ( 3. 8)
T-(10 0030000000000 00000 0 0) +
F-(13 00CLYDNGPRAVIT SWWWW 0 0) =
F-(13 K0CLYDNGPRAVIT SWWWW 6 4)


F-(14 00CLYDNGPRAVIT SWWWW 0 0) +          ( 3. 9)
F-(14 00CLYDNGPRAVIT SWWWW 0 0) =
F-(14 20CLYDNGPRAVIT SWWWW 0 0)


F-(14-10CLYDNGPRAVIT SWWWW 0 0) +          ( 3.10)
F-(14 00CLYDNGPRAVIT SWWWW 0 0) =
F-(14 K0CLYDNGPRAVIT SWWWW 6 4)
```

Appendix

PART IV

APPENDIX B

SOURCE LANGUAGE LISTING

OF

COMPUTER PROGRAM ANALYZ

AND

ASSOCIATED SUBPROGRAMS

PART IV

APPENDIX B

This appendix contains the source language listing of the following computer programs and associated subprograms:

(1)	Main Program	ANALYZ
(2)	Subprogram	ALPHA
(3)	Subprogram	DIAGRM
(4)	Subprogram	LIMIT
(5)	Subprogram	MACHER
(6)	Subprogram	OUTPUT
(7)	Subprogram	PARSE
(8)	Subprogram	PROPH1
(9)	Subprogram	PROPH2
(10)	Subprogram	RERITE
(11)	Subprogram	RULENO
(12)	Subprogram	SYMACH
(13)	Subprogram	VARATT
(14)	Main Program	RULIST

MAIN PROGRAM ANALYZ

```
C
C    ****************************************************************************
C    *                          PROGRAM ANALYZ                                  *
C    *           THIS PROGRAM PERFORMS SYNTACTIC ANALYSIS                       *
C    *                  OF HEBREW SENTENCES                                     *
C    *                          WRITTEN BY JAMES D. PRICE   JULY,1970          *
C    ****************************************************************************
C
      COMMON RULE(900,28), IPSI(100,8,2), RESTRT(5,15), ITABLE(400,29),
     1 SYMIN(29), SYMBOL(29), I, IMAXI,J, JMAX, IRULE, IMAX, IRULE1,
     2 ISTART, ITRACE, MATCH
      COMMON/TREE/ ITREE(50), NODE(7,400), NODE1, NODE2
      COMMON/PARS/ SYML(100,6), INDEX(100,8),AMSG(200,6),ENGLSH(20,4),
     1 ATTVAL(17,8),ISYMB(100)
      COMMON/ABC/ TRANSL(50)
      COMMON/SYMB/ SYM(100)
      INTEGER RULE, RESTRT,SYMIN,SYMBOL
      LOGICAL MATCH
      DIMENSION REEDER(36),ROOT(4), HEBREW(12)
      READ(5,107) MODE,ITAPE, ITRACE, IOUTPT,IOUTRE,LIMAX
  107 FORMAT(12I5)
      REWIND ITAPE
      IF(MODE.EQ.0) GO TO 10
      READ(ITAPE) TRANSL,RULE,IPSI,RESTRT ,SYM,SYML,ISYMB,INDEX,
     1 AMSG,ATTVAL
      REWIND ITAPE
      GO TO 4
C
C ***                    READ IN TRANSLITERATION
C
   10 READ(5,100) NL,(TRANSL(L),L=1,NL)
  100 FORMAT(I6,50A1)
C
C ***                    READ IN GRAMMAR RULES
C
      I=0
      ISYMNO=0
      ICLASS=0
      IRULE=0
    1 I=I+1
      READ(5,101) (REEDER(L),L=1,36)
  101 FORMAT(3A1,2(2X,A1),1X,2A1,13(2X,A1),1X,2A1,5(2X,A1),9A1)
      RULE(I,1)=0
      DO 11 L=1,3
      CALL ALPHA(REEDER(L),N)
   11 RULE(I,1)=RULE(I,1)+ N*10**(3-L)
      IF(RULE(I,1).EQ.999) GO TO 3
      DO 12 L=4,5
      L1=L-2
      CALL ALPHA(REEDER(L), N)
   12 RULE(I,L1)=N
```

```
      CALL ALPHA(REEDER(6),II)
      CALL ALPHA(REEDER(7),N)
      RULE(I,4)=N
      IF(II.EQ.39) RULE(I,4)=-RULE(I,4)
      DO 121 L=8,20
      L1=L-3
      CALL ALPHA(REEDER(L),N)
  121 RULE(I,L1)=N
      CALL ALPHA(REEDER(21),N)
      CALL ALPHA(REEDER(22),N1)
      RULE(I,18)=N1+N*10
      IF(N.GT.0)  RULE(I,18)=-RULE(I,18)
      DO 122 L=23,27
      L1=L-4
      CALL ALPHA(REEDER(L),N)
  122 RULE(I,L1)=N
      RULE(I,24)=0
      DO 13 L=28,30
      CALL ALPHA(REEDER(L),N)
   13 RULE(I,24)=RULE(I,24)+ N*10**(30-L)
      DO 14 L=31,32
      L1=L-6
      CALL ALPHA(REEDER(L),N)
   14 RULE(I,L1)=N
      RULE(I,27)=0
      DO 15 L=33,34
      CALL ALPHA(REEDER(L),N)
   15 RULE(I,27)=RULE(I,27)+ N*10**(34-L)
      RULE(I,28)=0
      DO 16 L=35,36
      CALL ALPHA(REEDER(L),N)
   16 RULE(I,28)=RULE(I,28)+N*10**(36-L)
      IF(RULE(I,25).NE.1) GO TO 2
      IC=RULE(I,6)
      IF(IC.EQ.9) IC=1
      IF(IC.EQ.ICLASS.AND.RULE(I,1).EQ.ISYMNO) GO TO 2
      IF(RULE(I,1).EQ.97.AND.ISYMNO.EQ.97.AND.RULE(I,3).EQ.ICLASS)GOTO2
      ISYMNO=RULE(I,1)
      ICLASS=RULE(I,6)
      IF(ISYMNO.EQ.97) ICLASS = RULE(I,3)
      IF(ICLASS.EQ.9) ICLASS=1
      IPSI(ISYMNO,ICLASS,1)=I
      GO TO 1
    2 IPSI(ISYMNO,ICLASS,2)=I
```

```
      GO TO 1
C
C ***                   READ IN RESTRICTION MATRIX
C
    3 IRULE=I
      READ(5,102)((RESTRT(I,J),I=1,5),J=1,15)
  102 FORMAT(5I5)
C
C ***                   READ IN SYMBOL ARRAY
C
      READ(5,104) SYM
  104 FORMAT(10(2X,A3))
      DO 31 I=1,96
   31 READ(5,400) (SYML(I,J),J=1,6),ISYMB(I)
  400 FORMAT(6A6,30X,I6)
      DO 32 I=1,100
   32 READ(5,401) (INDEX(I,J),J=1,8)
  401 FORMAT(8I5)
      DO 33 I=1,185
   33 READ(5,400) (AMSG(I,J),J=1,6)
      DO 34 I=1,17
   34 READ(5,402) (ATTVAL(I,J),J=1,8)
  402 FORMAT(8A6)
      WRITE(ITAPE) TRANSL,RULE,IPSI,RESTRT, SYM,SYML,ISYMB,INDEX,
     1 AMSG,ATTVAL
      REWIND ITAPE
      WRITE(6,200) TRANSL
  200 FORMAT(5X,50A1)
      WRITE(6,201)(NN,(RULE(NN,L),L=1,28),NN=1,IRULE)
  201 FORMAT(29I4)
      WRITE(6,202)(MM,((IPSI(MM,NN,L),L=1,2),NN=1,8),MM=1,100)
  202 FORMAT(17I5)
      WRITE(6,203)((RESTRT(NN,L),NN=1,5),L=1,15)
  203 FORMAT(5I5)
      WRITE(6,104) SYM
      WRITE(6,403) ((SYML(I,J),J=1,6),I=1,100)
  403 FORMAT(1H1/(10X,6A6))
      WRITE(6,404) ISYMB
  404 FORMAT(///5X,100I1)
      WRITE(6,405) ((INDEX(I,J),J=1,8),I=1,100)
  405 FORMAT(1H1/(10X,8I5))
      WRITE(6,403) ((AMSG(I,J),J=1,6),I=1,200)
      WRITE(6,406) ((ATTVAL(I,J),J=1,8),I=1,17)
  406 FORMAT(1H1/(10X,8A6))
```

```
C
C ***                    CLEAR REWRITE LIST MATRIX
C
    4 CONTINUE
      DO 5 I=1,400
      DO 51 J=1,29
   51 ITABLE(I,J) =0
      DO 52 J =1,7
   52 NODE(J,I) =0
    5 CONTINUE
C
C ***              READ EQUIVALENT HEBREW SENTENCE AND INITIAL SYMBOLS
C
      READ(5,105) HEBREW, NOP, IMAXI
  105 FORMAT(12A6,2I3)
      IF(IMAXI.EQ.0) GO TO 999
      DO 6 I=1,IMAXI
      READ(5,103)(ITABLE(  I,K),K=1,18),(ROOT(L),L=1,4),
     1 (ENGLSH(I,J),J=1,4)
  103 FORMAT(I6,4X,17I2,4A1,4A6)
      DO 61  J=1,4
      J1= J+18
      CALL ALPHA(ROOT(J),NN)
   61 ITABLE(  I,J1)=NN
      NODE(1,I)=1
      NODE(4,I)=ITABLE(I,1)
      NODE(5,I)=ITABLE(I,6)
    6 CONTINUE
      LIM=0
      ISTART=0
      NODE1=0
      NODE2=IMAXI
      IPASS = 1
      ITREE(1)= IMAXI
    7 CONTINUE
      J1=0
   79 CONTINUE
      IF(ITRACE.NE.0) WRITE(6,301) IPASS,NODE1,NODE2
  301 FORMAT(10X,18HNEW PASS---IPASS= ,I3,9H, NODE1= ,I3,
     1 9H, NODE2= ,I3,1H.)
      CALL RERITE(MON,J1,LIM)
      IF(LIM.GT.LIMAX) GO TO 76
      IF(NODE2.GT.400) GO TO 74
      IF(MON.EQ.0) GO TO 70
      IF(I.NE.IMAXI) GO TO 70
      IF(ITRACE.NE.0) WRITE(6,302) IPASS,IMAXI,NODE1,NODE2
  302 FORMAT(10X,18HCOMPLETED PASS NO.,I3,10H.  IMAXI= ,I3,
     1 9H, NODE1= ,I3,9H, NODE2= ,I3,1H.)
```

```
      IPASS = IPASS+1
      IMAXI = NODE2-NODE1
      ITREE(IPASS) = IMAXI
      ISTART =0
      IF(IMAXI.GT.1) GO TO 7
      GO TO 75
C
C ***     HUNT BACKWARDS TO FIND FIRST NODE WITH ALTERNATE RULE.
C
   70 CONTINUE
      NODE(6,NODE1) =0
      NODE(7,NODE1) =0
      NODE1 = NODE1-1
      IF(NODE1.LT.1) GO TO 73
      IF(NODE(6,NODE1).LE.0) GO TO 70
C
C ***     SKIP IF GOVERNED NODE IS SAME AS COMPOUND GOVERNING NODE.
C
      NG=NODE(3,NODE1)
      IF(NODE(4,NODE1).NE.NODE(4,NG)) GO TO 271
      IF(NODE(5,NODE1).NE.NODE(5,NG)) GO TO 271
      IF(ITABLE(NODE1,4).EQ.0.AND.ITABLE(NG,4).GT.0) GO TO 70
  271 CONTINUE
      NODE2= NODE(3, NODE1)
      IRULE=NODE(6,NODE1)
      IMAX =NODE(7,NODE1)
      ISUM=0
      IP=IPASS
      DO 71 L=1,IP
      ISUM=  ISUM + ITREE(L)
      IPASS = L
      IF(ISUM.GE.NODE1) GO TO 72
   71 CONTINUE
   72 IMAXI = ITREE(IPASS)
C
C ***          CANCEL ANY PREVIOUS PREDICTION
C
      IF(NODE(6,NODE2).GE.0) GO TO 720
      NOD11=NODE1+1
      NODE(6,NODE2)=0
      NODE(7,NODE2)=0
      NODE(6,NOD11)=0
      NODE(7,NOD11)=0
  720 CONTINUE
      NODE2=NODE2-1
      NODE1=NODE1-1
      ISTART= IMAXI + NODE1 - ISUM
      IF(ITRACE.NE.0) WRITE(6,303) IPASS,ISTART,IMAXI,NODE1,NODE2
  303 FORMAT(10X,45HHIT ALTERNATE RULE BRANCH.  WENT BACK TO PASS,I3,
     1 6H, NODE,I3,10H.  IMAXI= ,I3,9H, NODE1= ,I3,9H, NODE2= I3,1H.)
```

```
      J1=1
      GO TO 79
   73 WRITE(6,173) NOP
  173 FORMAT(5X,54HERROR DETECTED IN ALTERNATE RULE BRANCH FOR PROB. NO.
     1 ,I3,23H.  ANALYSIS TERMINATED.//5X,22HSEE PROGRAM ANALYZ 70.)
      GO TO 4
   74 WRITE(6,174) NOP
  174 FORMAT(5X, 32HNODES EXCEED 400 FOR PROBLEM NO.,I3,
     1 23H.  ANALYSIS TERMINATED.)
      GO TO 4
   75 CONTINUE
      IF(IOUTPT.EQ.1) CALL OUTPUT(IPASS)
      WRITE(6,106) HEBREW
  106 FORMAT(1H1//10X,26HHEBREW SENTENCE ANALYZED--  /10X,12A6)
      IF(IOUTRE.EQ.1) CALL DIAGRM(IPASS)
      CALL PARSE(IPASS,NOP)
      WRITE(6,304) NOP,LIM
  304 FORMAT(///5X,36HANALYSIS COMPLETED FOR SENTENCE NO.   ,I3,
     1 23H, NO. OF SYMBOL TESTS=    ,I4, 1H.)
      GO TO 4
   76 WRITE(6,176) LIMAX,NOP
  176 FORMAT(5X,21HNO ANALYSIS FOUND BY ,I6,52H SYMBOL TESTS.  ANALYSIS
     1TERMINATED FOR PROBLEM NO.    ,I3,1H.)
      GO TO 4
  999 WRITE(6,199)
  199 FORMAT(1H1,10(/),1X,17(7H--END--))
      STOP
      END
```

SUBPROGRAM ALPHA

```
      SUBROUTINE ALPHA(A,IA)
C
C     ****************************************************************************
C     * THIS SUBROUTINE CONVERTS THE ALPHA-NUMERIC SYMBOL(A)  TO AN             *
C     * INTEGER EQUIVANENT BASED ON DATA IN ARRAY TRANSL.                        *
C     *                              FOR USE WITH PROGRAM ANALYZ                 *
C     *                              WRITTEN BY JAMES D. PRICE   JULY,1970       *
C     ****************************************************************************
C
      COMMON/ABC/ TRANSL(50)
C
      DO 1 L=1,50
      IF(A-TRANSL(L)) 1,2,1
    1 CONTINUE
      IA=49
      RETURN
    2 IA=L-1
      IF(IA.EQ.40) IA=0
      RETURN
      END
```

157

SUBPROGRAM DIAGRM

```
      SUBROUTINE DIAGRM(IPASS)
C
C    ****************************************************************
C    *     THIS SUBROUTINE CONSTRUCTS A TREE DIAGRAM                *
C    *     OF THE SENTENCE ANALYSIS.                                *
C    *                             FOR USE WITH PROGRAM ANALYZ      *
C    *                             WRITTEN BY JAMES D. PRICE   JULY,1970   *
C    ****************************************************************
C
      COMMON RULE(900,28), IPSI(100,8,2), RESTRT(5,15), ITABLE(400,29),
     1 SYMIN(29), SYMBOL(29), I, IMAXI,J, JMAX, IRULE, IMAX, IRULE1,
     2 ISTART, ITRACE, MATCH
      COMMON/TREE/ ITREE(50), NODE(7,400), NODE1, NODE2
      INTEGER RULE, RESTRT,SYMIN,SYMBOL
      COMMON/SYMB/ SYM(100)
      DIMENSION ALINE(40), CLS(9)
      DATA BLANK/3H   /
      DATA DASHES/3H---/
      DATA BAR/3H  I/
      DATA CLS/3H1  ,3H2  ,3H3  ,3H4  ,3H5  ,3H6  ,3H7  ,3H8  ,3H9  /
      IF(ITRACE.NE.0) WRITE(6,99)
   99 FORMAT(30H      SUBROUTINE DIAGRAM CALLED  )
C
C ***                 COMPUTE LINE POSITION FOR EACH NODE
C
      NP=ITREE(1)
      DO 3 L=1,NP
    3 NODE(2,L)=L
      L=0
    4 L=L+1
      NG=NODE(3,L)
      IF(NODE(1,NG).GT.1) GO TO 5
      NODE(2,NG)=NODE(2,L)
      GO TO 6
    5 L1=L+NODE(1,NG)-1
      NODE(2,NG)= (NODE(2,L)+NODE(2,L1))/2
      L=L1
    6 IF(L.LT.NODE1) GO TO 4
      IF(ITRACE.EQ.0) GO TO 71
      WRITE(6,103)
  103 FORMAT(1H1//10X,11HNODE MATRIX//)
      DO 7 L=1,NODE1
    7 WRITE(6,102)(NODE(I,L),I=1,5)
  102 FORMAT(10X,5I5)
   71 CONTINUE
```

```
C
C ***                    WRITE OUT TREE DIAGRAM
C
      WRITE(6,100)
  100 FORMAT(//10X,35H TREE DIAGRAM OF HEBREW SENTENCE   //)
      NO=1
      DO 50 M=1,IPASS
      IF(M.EQ.1) GO TO 31
C
C ***                    WRITE UPPER/LOWER  CONNECTORS(VERTICAL)
C
   21 DO 30 L=1,20
      IF((NODE(1,NO).EQ.0.OR.NODE(2,NO).EQ.0).AND.NO.LE.NODE1) NO=NO+1
      L1=L+L-1
      L2=L+L
      IF(L.EQ.NODE(2,NO)) GO TO 22
      ALINE(L1)=BLANK
      ALINE(L2)=BLANK
      GO TO 30
   22 ALINE(L2)=BLANK
      ALINE(L1)=BAR
      NO=NO+1
   30 CONTINUE
      NO=NO-ITREE(M)
      WRITE(6,101) ALINE
      IF(IUPLOW.EQ.2) GO TO 42
C
C ***                    WRITE LINE OF NODES
C
   31 DO 40 L=1,20
      IF((NODE(1,NO).EQ.0.OR.NODE(2,NO).EQ.0).AND.NO.LE.NODE1) NO=NO+1
      L1=L+L-1
      L2=L+L
      IF(L.EQ.NODE(2,NO)) GO TO 32
      ALINE(L1)=BLANK
      ALINE(L2)=BLANK
      GO TO 40
   32 NS=NODE(4,NO)
      NT=NODE(5,NO)
      NG=NODE(3,NO)
      NO=NO+1
      IF(M.EQ.1) GO TO 33
      IF(NS.EQ.NODE(4,NG).AND.NT.EQ.NODE(5,NG)) GO TO 34
   33 ALINE(L1)=SYM(NS)
      ALINE(L2)=CLS(NT)
      GO TO 40
   34 IF(M.EQ.IPASS) GO TO 33
      ALINE(L2)=BLANK
      ALINE(L1)=BAR
   40 CONTINUE
```

```
      NO=NO-ITREE(M)
      WRITE(6,101) ALINE
C ***                  WRITE LOWER CONNECTORS (VERTICAL)
C
   41 IUPLOW=2
      IF(M.EQ.IPASS) GO TO 50
      GO TO 21
C
C ***                  WRITE LOWER CONNECTORS (HORIZONTAL)
C
   42 IEND=0
      IENDPO=0
      DO 20 L=1,20
      IF((NODE(1,NO).EQ.0.OR.NODE(2,NO).EQ.0).AND.NO.LE.NODE1) NO=NO+1
      L1=L+L-1
      L2=L+L
      IF(L.EQ.IENDPO)  IEND=1
      IF(L-NODE(2,NO))  10,11,12
   10 ALINE(L1)=BLANK
      ALINE(L2)=BLANK
      GO TO 20
   11 NG=NODE(3,NO)
      NO1=NODE(1,NG)
      NO2=NO+NO1-1
      IENDPO=NODE(2,NO2)
      ALINE(L1)=BLANK
      IF(IENDPO.GT.L) GO TO 13
      ALINE(L1)=BAR
      NO=NO+1
      GO TO 20
   12 IF(IEND.EQ.1) GO TO 14
      IF(L.GT.IENDPO) GO TO 10
      ALINE(L1)=DASHES
   13 ALINE(L2)=DASHES
      GO TO 20
   14 ALINE(L1)=DASHES
      ALINE(L2)=BLANK
      NO=NO+NO1
      IEND=0
      IENDPO=0
   20 CONTINUE
      NO=NO-ITREE(M)
      IUPLOW=1
      WRITE(6,101) ALINE
  101 FORMAT(1X,40A3)
   50 NO=NO+ ITREE(M)
      RETURN
      END
```

SUBPROGRAM LIMIT

```
      SUBROUTINE LIMIT(INSYM)
C
C     ************************************************************************
C     * THIS SUBROUTINE CHECKS A GIVEN SYMBOL AGAINST SPECIFIED              *
C     * LIMITATIONS.    IF NOT SATISFIED,VARIABLE MATCH IS SET TO FALSE.  *
C     *                          FOR USE WITH PROGRAM ANALYZ                 *
C     *                          WRITTEN BY JAMES D. PRICE   JULY,1970       *
C     ************************************************************************
C
      COMMON RULE(900,28), IPSI(100,8,2), RESTRT(5,15), ITABLE(400,29),
     1 SYMIN(29), SYMBOL(29), I, IMAXI,J, JMAX, IRULE, IMAX, IRULE1,
     2 ISTART, ITRACE, MATCH
      INTEGER RULE, RESTRT,SYMIN,SYMBOL
      DIMENSION  INSYM(29)
      LOGICAL MATCH
      IF(ITRACE.NE.0) WRITE(6,99)
   99 FORMAT(    30H     SUBROUTINE LIMIT  CALLED )
      IR=INSYM(27)
      IV=INSYM(28)
      IM=RESTRT(1,IR)+1
      DO 2 L=2,IM
      IF(RESTRT(L,IR).GE.0) GO TO 1
      MATCH=.TRUE.
      IREST=-RESTRT(L,IR)
      IF( INSYM(IV).NE.IREST) GO TO 2
      GO TO 3
    1 IF( INSYM(IV).EQ.RESTRT(L,IR)) GO TO 14
      MATCH=.FALSE.
    2 CONTINUE
      RETURN
    3 MATCH=.FALSE.
      RETURN
   14 MATCH = .TRUE.
      RETURN
      END
```

SUBPROGRAM MACHER

```
      SUBROUTINE MACHER(IS1,IS2,MATCH)
C
C    ****************************************************************
C    *           THIS SUBROUTINE TESTS FOR A MATCH BETWEEN          *
C    *           SUBSCRIPTS OF GIVEN SYMBOLS IS1 AND IS2.
C    *           CALLED BY PROPH1 AND PROPH2.                       *
C    *                         WRITTEN BY JAMES D. PRICE OCTOBER,1970   *
C    ****************************************************************
C
      DIMENSION IS1(29),IS2(29)
      LOGICAL MATCH
C
      MATCH= .TRUE.
      IF(IS1(1).NE.IS2(1)) GO TO 4
      DO 3 L=2,17
      IF(IS1(L).EQ.IS2(L)) GO TO 3
      IF(L.EQ.3) GO TO 3
      IF(L.EQ.7.AND.IS1(8).EQ.0) GO TO 3
      IF(IS1(L).EQ.9) GO TO 3
      IF(IS2(L).EQ.9) GO TO 3
      IF(IS2(L).LT.0.AND.IS1(L).LT.9) GO TO 3
      IF(IS2(L).GT.9.AND.IS1(L).LT.9) GO TO 3
      GO TO 4
    3 CONTINUE
      RETURN
    4 MATCH= .FALSE.
      RETURN
      END
```

SUBPROGRAM OUTPUT

```
      SUBROUTINE OUTPUT(IPASS)
C
C   *************************************************************************
C   * THIS SUBROUTINE WRITES THE RESULTS OF THE ITH REWRITE PASS            *
C   * OF THE GRAMMAR.                                                       *
C   *                           FOR USE WITH PROGRAM ANALYZ                 *
C   *                           WRITTEN BY JAMES D. PRICE    JULY,1970      *
C   *************************************************************************
C
      COMMON RULE(900,28), IPSI(100,8,2), RESTRT(5,15), ITABLE(400,29),
     1 SYMIN(29), SYMBOL(29), I, IMAXI,J, JMAX, IRULE, IMAX, IRULE1,
     2 ISTART, ITRACE, MATCH
      COMMON/TREE/ ITREE(50), NODE(7,400), NODE1, NODE2
      COMMON/ABC/ TRANSL(50)
      INTEGER RULE, RESTRT,SYMIN,SYMBOL
      LOGICAL  MATCH
      IF(ITRACE.NE.0) WRITE(6,99)
   99 FORMAT(    30H     SUBROUTINE OUTPUT CALLED )
      J1=1
      DO 2 ITH =1,IPASS
      WRITE(6,100) ITH
  100 FORMAT(1H1///9X,28HRESULTS OF REWRITE PASS NO.-,I3//
     1105H     SYMBOL     M  F  K  B  C  L  Y  D  N  G  P  R  A  V  I  T
     2 S      W       TY  RULE EL NE  REST SUB    X//)
      J2=J1+ITREE(ITH)-1
      DO 1 JJ= J1,J2
      IR1=ITABLE(JJ,19)+1
      IR2=ITABLE(JJ,20)+1
      IR3=ITABLE(JJ,21)+1
      IR4=ITABLE(JJ,22)+1
    1 WRITE(6,101) (ITABLE(         JJ,L),L=1,18),TRANSL(IR1),
     1 TRANSL(IR2), TRANSL(IR3), TRANSL(IR4), (ITABLE(JJ,L),L=23,29)
  101 FORMAT(6X,I3,4X,17I3,4X,4A1,3X,I3,3X,3I3,3(2X,I3))
      J1=J2+1
    2 CONTINUE
      RETURN
      END
```

SUBPROGRAM PARSE

```
      SUBROUTINE PARSE(IPASS,NOP)
C
C   ****************************************************************
C   *          THIS SUBROUTINE ASSEMBLES STATEMENTS ABOUT THE       *
C   *          SYNTACTIC ANALYSIS OF A SENTENCE. CALLED FROM        *
C   *          PROGRAM ANALYZ.                                      *
C   *                         WRITTEN BY JAMES D.PRICE              *
C   *                                   FIRL AUG,1970               *
C   ****************************************************************
C
      COMMON RULE(?90,28), IPSI(100,8,2), RESTRT(5,15), ITABLE(400,29),
     1 SYMIN(29), SYMBOL(29), I, IMAXI,J, JMAX, IRULE, IMAX, IRULE1,
     2 ISTART, ITRACE, MATCH
      COMMON/TREE/ ITREE(50), NODE(7,400), NODE1, NODE2
      COMMON/PARS/ SYML(100,6), INDEX(100,8),AMSG(200,6),ENGLSH(20,4),
     1 ATTVAL(17,8),ISYMB(100)
      INTEGER RULE, RESTRT,SYMIN,SYMBOL
      DIMENSION  ACCUM(200)
      DATA  THE/6H THE /,OF/6H  OF  /,AS/6H  IS  /,COMMA/6H,     /
     1 ,DOT/6H.     /
C
      WRITE(6,200) NOP
  200 FORMAT(1H1//10X,31HSYNTAX ANALYSIS IF SENTENCE NO.,I4,1H.//)
C
C ***          FOR EACH PASS, EXAMINE SYMBOLS FOR SYNTAX DATA.
C
      NAN=0
      NTERM=ITREE(1)
      N2= NODE2
      DO 90 I=IPASS,1,-1
      N1= N2 -ITREE(I)+1
      IF(N1.GT.N2.OR.N1.LE.0) GO TO 90
      DO 80  J=N1,N2
C
C ***          ACCUMULATE DATA ABOUT SYMBOL AT NODE J
C
      ACCUM(1)=THE
      M1=2
      IS= NODE(4,J)
      IF(IS.GT.76.AND.J.GT.NTERM) GO TO 80
      IF(IS.LE.0) GO TO 80
      IC= NODE(5,J)
      ISS = IS
      ICC = IC
      M2=M1+ISYMB(IS)-1
      M3=0
      DO 1 M=M1,M2
      M3=M3+1
    1 ACCUM(M)=SYML(IS,M3)
```

```
      NG= NODE(3,J)
      IF(NG.LE.0.OR.NG.GT.NODE2) GO TO 83
   81 NSG=NODE(4,NG)
      IF(NSG.LE.0) GO TO 82
      NSC = NODE(5,NG)
C
C ***      SKIP IF NODE HAS ALREADY BEEN TREATED.
C ***     SKIP IF GOVERNING NODE IS SAME AS GOVERNED NODE.
C
      IF(IS.EQ.NSG.AND.IC.EQ.NSC) GO TO 80
      IF(ISS.EQ.NSG.AND.ICC.EQ.NSC) GO TO 82
      M1= M2+1
      M2=M1+1
      ACCUM(M1)=OF
      ACCUM(M2)=THE
      M1=M2+1
      M2=M2+ISYMB(NSG)
      M3=0
      DO 2 M=M1,M2
      M3=M3+1
    2 ACCUM(M)=SYML(NSG,M3)
   82 NG =NODE(3,NG)
      IF(NG.EQ.0.OR.NG.GE.NODE2) GO TO 83
      ISS = NSG
      ICC = NSC
      GO TO 81
   83 CONTINUE
      IF(IS.GT.76) GO TO 84
      IMSG = INDEX(IS,IC)
      IF(IMSG.EQ.0) GO TO 80
      NAN=NAN+1
      WRITE(6,100) NAN, (ACCUM(M),M=1,M2),(AMSG(IMSG,L),L=1,6)
  100 FORMAT(/10X,1H(,I3,2H) ,10A6/20(10X,11A6/))
      GO TO 80
   84 CONTINUE
      NAN=NAN+1
      WRITE(6,100) NAN,  (ACCUM(M),M=1,M2),AS,(ENGLSH(J,L),L=1,4)
      M1=0
```

```
      DO 85 M=3,17
      IF(M.EQ.6.OR.M.EQ.7) GO TO 85
      IF((M.EQ.3.OR.M.EQ.4).AND.(ITABLE(J,3).EQ.0.OR.ITABLE(J,4).EQ.0))
     1 GO TO 85
      MM= ITABLE(J,M)
      IF(MM.EQ.0) GO TO 85
      M1=M1+1
      ACCUM(M1)= ATTVAL(M,MM)
   85 CONTINUE
      IF(M1.EQ.0) GO TO 80
      M11=M1-1
      WRITE(6,101) (ACCUM(M),COMMA,M=1,M11), ACCUM(M1)   ,DOT
  101 FORMAT(10X, 9(A6,A2))
   80 CONTINUE
      N2 = N1-1
   90 CONTINUE
      RETURN
      END
```

171

SUBPROGRAM PROPH1

```
      SUBROUTINE PROPH1(ID,IR,IM)
C
C    ***************************************************************************
C    *           THIS SUBROUTINE EXAMINES THE PRESENT SYMBOL                  *
C    *           FOR POSSIBLE FUTURE SATISFACTION OF GRAMMAR                  *
C    *           RULE AFTER ONE MORE PASS.                                    *
C    *           CALLED FROM SUBROUTINE RERITE.                               *
C    *                          WRITTEN BY JAMES D. PRICE AUG,1970            *
C    ***************************************************************************
C
      COMMON RULE(900,28), IPSI(100,8,2), RESTRT(5,15), ITABLE(400,29),
     1 SYMIN(29), SYMBOL(29), I, IMAXI,J, JMAX, IRULE, IMAX, IRULE1,
     2 ISTART, ITRACE, MATCH
      COMMON/TREE/ ITREE(50), NODE(7,400), NODE1, NODE2
      DIMENSION  IS1(29),IS2(29),IS3(29),IS4(29)
      INTEGER RULE, RESTRT,SYMIN,SYMBOL
      LOGICAL  MATCH
      IF(ITRACE.NE.0) WRITE(6,199)
  199 FORMAT(10X,25HSUBROUTINE PROPH1 CALLED.)
      IF(SYMIN(1).EQ.RULE(IRULE1,1).AND.(SYMIN(6).EQ.9.OR.
     1  SYMIN(6).EQ.RULE(IRULE1,6))) GO TO 91
C
C ***      LOOK AHEAD ONE  LEVEL
C
      IN=NODE1
      IR=0
      IM=0
      IS= SYMIN(1)
      IC=SYMIN(6)
      DO 1  L=1,8
      IR1 = IPSI(IS,L,1)
      IF(IR1.NE.0) GO TO 2
    1 CONTINUE
      GO TO 92
    2 DO 3  L=8,1,-1
      IR3  = IPSI(IS,L,2)
      IF(IR3.NE.0) GO TO 4
    3 CONTINUE
      GO TO 92
    4 IR2 = IR1 + RULE(IR1,26)
      IR4=IR1
  401 CONTINUE
      DO 41 L=1,29
      IS1(L)= ITABLE(IN,L)
      IS2(L)= RULE(IR1,L)
      IS3(L)= RULE(IRULE1,L)
      IS4(L)= RULE(IR2,L)
   41 CONTINUE
```

173

```
      CALL MACHER(IS1,IS2,MATCH)
      IF(.NOT.MATCH) GO TO 51
      CALL MACHER(IS3,IS4,MATCH)
      IF(MATCH) GO TO 6
      CALL PROPH2(ID,IR2)
      IF(ID.EQ.4) GO TO 94
    5 CONTINUE
      ID=0
      IR1= IR2+1
      IF(IR1.GT.IR3) GO TO 93
      GO TO 4
   51 IF(IS2(2).EQ.1) GO TO 5
      GO TO 7
    6 CONTINUE
      IN=IN+1
    7 CONTINUE
      IR1=IR1+1
      IF(IR1.GE.IR2) GO TO 94
      GO TO 401
C
C ***     NO FUTURE SATISFACTION, PRESENT RULE FAILED IN SYMACH OR LIMIT
C
   91 ID=1
      GO TO 99
C
C ***     NO FUTURE SATISFACTION, NO RULES TO PREDICT CONDITION.
C
   92 ID=2
      GO TO 99
C
C ***     NO FUTURE SATISFACTION, PREDICTION RULES EXHAUSTED.
C
   93 ID=3
      GO TO 99
C
C ***     FUTURE SATISFACTION PREDICTED.
C
   94 ID=4
      IR=IR4
      IM=IR3
   99 IF(ITRACE.NE.0) WRITE(6,299) ID
  299 FORMAT(10X, 3HID=,I2,1H.)
      RETURN
      END
```

SUBPROGRAM PROPH2

```
      SUBROUTINE PROPH2(ID,IR)
C
C     ******************************************************************
C     *          THIS SUBROUTINE EXAMINES THE SYMBOL DEFINED BY        *
C     *          IR  IN THE RULE MATRIX FOR POSSIBLE FUTURE            *
C     *          SATISFACTION OF THE GIVEN GRAMMAR RULE AFTER          *
C     *          TWO PASSES.  CALLED FROM SUBROUTINE PROPH1.           *
C     *                        WRITTEN BY JAMES D. PRICE  OCTOBER,1970 *
C     ******************************************************************
C
      COMMON RULE(900,28), IPSI(100,8,2), RESTRT(5,15), ITABLE(400,29),
     1 SYMIN(29), SYMBOL(29), I, IMAXI,J, JMAX, IRULE, IMAX, IRULE1,
     2 ISTART, ITRACE, MATCH
      DIMENSION  IS1(29),IS2(29),IS3(29),IS4(29)
      INTEGER RULE, RESTRT,SYMIN,SYMBOL
      LOGICAL  MATCH
      IF(ITRACE.NE.0) WRITE(6,199)
  199 FORMAT(10X,25HSUBROUTINE PROPH2 CALLED.)
C
C ***     LOOK AHEAD ONE LEVEL
C
      IS= RULE(IR,1)
      IC= RULE(IR,6)
      DO 1 L=1,8
      IR1 =IPSI(IS,L,1)
      IF(IR1.NE.0) GO TO 2
    1 CONTINUE
      GO TO 92
    2 DO 3 L=8,1,-1
      IR3 =IPSI(IS,L,2)
      IF(IR3.NE.0) GO TO 4
    3 CONTINUE
      GO TO 92
    4 IR2 = IR1 +RULE(IR1,26)
      DO 41 L=1,29
      IS1(L)= RULE(IR,L)
      IS2(L)= RULE(IR1,L)
      IS3(L)= RULE(IRULE1,L)
      IS4(L)= RULE(IR2,L)
   41 CONTINUE
      CALL MACHER(IS1,IS2,MATCH)
      IF(.NOT.MATCH) GO TO 5
      CALL MACHER(IS3,IS4,MATCH)
      IF(MATCH) GO TO 94
    5 CONTINUE
      IR1=IR2+1
      IF(IR1.GT.IR3) GO TO 93
      GO TO 4
```

```
C
C ***     NO FUTURE SATISFACTION, NO RULES TO PREDICT CONDITION
C
   92 ID= 2
      GO TO 99
C
C ***     NO FUTURE SATISFACTION, PREDICTION RULES EXHAUSTED.
C
   93 ID= 3
      GO TO 99
C
C ***     FUTURE SATISFACTION PREDICTED.
C
   94 ID=4
   99 IF(ITRACE.NE.0) WRITE(6,299) ID
  299 FORMAT(10X,10HPROPH2 ID=,I2,1H.)
      RETURN
      END
```

177

SUBPROGRAM RERITE

```
      SUBROUTINE RERITE(MON,J1,LIM)
C
C    ****************************************************************
C    *     THIS SUBROUTINE REWRITES A STRING OF SYMBOLS ACCORDING TO   *
C    *     APPLICABLE GRAMMAR RULES.  CALLED FROM ANALYZ              *
C    *                         FOR USE WITH PROGRAM ANALYZ            *
C    *                         WRITTEN BY JAMES D. PRICE   JULY,1970  *
C    ****************************************************************
C
      COMMON RULE(900,28), IPSI(100,8,2), RESTRT(5,15), ITABLE(400,29),
     1 SYMIN(29), SYMBOL(29), I, IMAXI,J, JMAX, IRULE, IMAX, IRULE1,
     2 ISTART, ITRACE, MATCH
      COMMON/TREE/ ITREE(50), NODE(7,400), NODE1, NODE2
      INTEGER RULE, RESTRT,SYMIN,SYMBOL
      LOGICAL  MATCH
      IF(ITRACE.NE.0) WRITE(6,99)
   99 FORMAT(    30H     SUBROUTINE RERITE CALLED )
      NSTART = NODE1
      MSTART = NODE2
      I=0
      J=0
      MON=0
      J2=1
      J3=0
    1 IF(J2.NE.1) GO TO 101
      I = ISTART
      NODE1 = NSTART
      NODE2 = MSTART
  101 IF(I.GE.IMAXI) RETURN
      LIM=LIM+1
      I = I +1
      NODE1 = NODE1 +1
      DO 2 L=1,29
    2 SYMIN(L)= ITABLE( NODE1, L)
   21 CONTINUE
      IF(J.EQ.0) GO TO 41
      NODE(6,NODE1)=0
      NODE(7,NODE1)=0
      GO TO 4
   41 CALL RULENO(J1,J2,J3)
      NODE(6,NODE1) = IRULE
      NODE(7,NODE1) = IMAX
      IX=IRULE+RULE(IRULE,26)
      IF(IX.NE.IMAX)  GO TO 42
      NODE(6,NODE1)=0
      NODE(7,NODE1)=0
   42 CONTINUE
```

179

```
      IF(IRULE.EQ.0) GO TO 9
      IRULE1= IRULE-1
      JMAX =RULE(IRULE,26)
      JJ = JMAX +IRULE
      DO 3 L=1,28
    3 SYMBOL(L) = RULE(JJ,L)
    4 J= J+1
      IRULE1 = IRULE1+1
      IF(J.GT.JMAX) GO TO 71
      CALL SYMACH
      IF(MATCH) GO TO 6
      ID=0
      IF(J2.EQ.2) CALL PROPH1(ID,IR,IM)
      IF(ID.EQ.4) GO TO 91
      IF(RULE(IRULE1,2).EQ.1) GO TO 52
      SYMBOL(26) = SYMBOL(26)-1
      GO TO 4
    5 J1= J1+1
   51 I = ISTART
      NODE1 = NSTART
      NODE2 = MSTART
      J = 0
      IF(NODE(6,NODE2).EQ.0) GO TO 101
      J1=0
      NOD11=NODE1+1
      NODE(6,NOD11)=0
      NODE(7,NOD11)=0
      NODE(6,NODE2)=0
      NODE(7,NODE2)=0
      GO TO 101
   52 CONTINUE
      IF(J3.EQ.0) GO TO 5
      J1=J1+1
      NODE1=NSTRT1
      I=ISTRT1
      J=0
      GO TO 21
    6 CALL VARATT
      NODE(3,NODE1) = NODE2 +1
      IF(J.EQ.JMAX) GO TO 7
      GO TO 101
   71 NODE1=NODE1-1
      I=I-1
    7 CALL LIMIT(SYMBOL)
      IF(.NOT.MATCH) GO TO 52
      IF(J2.NE.1) GO TO 73
      J3=J3+1
      DO 72 L=1,29
   72 SYMIN(L)=SYMBOL(L)
      J=0
      J1=0
      ISTRT1=I
      NSTRT1=NODE1
      GO TO 21
```

181

```
 73 CONTINUE
    MON=MON+1
    ISTART = I
    NODE2 = NODE2 +1
    IF(NODE2.GT.400) RETURN
    NODE(4,NODE2)=SYMBOL(1)
    NODE(5,NODE2)=SYMBOL(6)
    NODE(1,NODE2) = SYMBOL(26)
    IF(SYMBOL(4).EQ.1) SYMBOL(4)=2
    K=SYMBOL(4)
    IF(K.EQ.0) GO TO 74
    K1=SYMBOL(3)
    K2=SYMBOL(5)
    K3=K-1
    IF(K1.EQ.3.OR.K2.EQ.3) K3=K
    IF(K1.EQ.4) K3=0
    NODE(1,NODE2)=K+K3
 74 CONTINUE
    DO 8  L=1,29
  8 ITABLE( NODE2,  L)= SYMBOL(L)
    GO TO 11
  9 CONTINUE
    IF(J2.EQ.1.AND.J3.NE.0.AND.SYMBOL(1).EQ.97) GO TO 1
    IF(J3.NE.0) GO TO 73
    IF(J2.EQ.2) GO TO 91
    J1=0
    J2=2
    GO TO 51
 91 CONTINUE
    N1=NODE1
    ISTART=ISTART+1
    NODE1=NSTART+1
    NODE2 = NODE2 +1
    IF(NODE2.GT.400) RETURN
    IF(ID.NE.4) GO TO 92
    NODE(6,NODE2)=-IRULE
    NODE(7,NODE2)=IMAX
    NODE(6,N1) = -IR
    NODE(7,N1) =  IM
 92 CONTINUE
    NODE(3,NODE1) = NODE2
    NODE(1,NODE2) = 1
    NODE(4,NODE2)=ITABLE(NODE1,1)
    NODE(5,NODE2)=ITABLE(NODE1,6)
    DO 10 L=1,29
 10 ITABLE( NODE2,  L)= ITABLE(NODE1,L)
 11 J=0
    J1=0
    J2=1
    J3=0
    NSTART = NODE1
    MSTART = NODE2
    IF(ITRACE.NE.0) WRITE(6,300) NODE2,(ITABLE(NODE2,L),L=1,29)
300 FORMAT(10X,8HNODE NO.,I3,4H IS ,29I3)
    GO TO 1
    END
```

181

SUBPROGRAM RULENO

```
      SUBROUTINE RULENO(J1,J2,J3)
C
C     ***************************************************************************
C     *      THIS SUBROUTINE COMPUTES THE ROW NUMBER  OF THE FIRST AND LAST
C     *      SYMBOL OF THE SET OF RULES GOVERNING A GIVEN INPUT SYMBOL.       *
C     *      CALLED FROM SUBROUTINE RERITE.                                   *
C     *                            FOR USE WITH PROGRAM ANALYZ                *
C     *                            WRITTEN BY JAMES D. PRICE   JULY,1970      *
C     ***************************************************************************
C
      COMMON RULE(900,28), IPSI(100,8,2), RESTRT(5,15), ITABLE(400,29),
     1 SYMIN(29), SYMBOL(29), I, IMAXI,J, JMAX, IRULE, IMAX, IRULE1,
     2 ISTART, ITRACE, MATCH
      COMMON/TREE/ ITREE(50), NODE(7,400), NODE1, NODE2
      INTEGER RULE, RESTRT,SYMIN,SYMBOL
      LOGICAL MATCH
      IF(ITRACE.NE.0) WRITE(6,99)
   99 FORMAT(    30H      SUBROUTINE RULENO CALLED )
      IF(J1.NE.0) GO TO 10
      GO TO(1,2),J2
    1 IS=97
      II=NODE1
      I1=II+1
      I2=II+2
      I3=II+3
      IF(ITABLE(I1,1).EQ.ITABLE(II,1)) GO TO 11
      IF(ITABLE(I2,1).EQ.ITABLE(II,1).AND.ITABLE(I1,1).EQ.79.AND.
     1 ITABLE(I1,6).LT.3) GO TO 12
      IF(ITABLE(I2,1).EQ.ITABLE(II,1).AND.ITABLE(I1,1).EQ.92.AND.
     1 ITABLE(I1,6).EQ.3.AND.((ITABLE(I3,1).EQ.79.AND.ITABLE(I3,6).LT.
     2 3).OR.(ITABLE(I3,1).EQ.92.AND.ITABLE(I3,6).EQ.3))) GO TO 12
      IF(J3.NE.0) GO TO 19
      J2=2
      GO TO 2
   11 IC=4
      GO TO 3
   12 IRULE = IPSI(IS,1,1)
      IMAX  = IPSI(IS,3,2)
      IC=1
      GO TO 6
    2 CONTINUE
      IS=SYMIN(1)
      IC=SYMIN(6)
      IF(NODE(6,NODE1).EQ.0) GO TO 21
      IRULE=IABS(NODE(6,NODE1))
      IMAX=NODE(7,NODE1)
      GO TO 6
```

```
21 CONTINUE
   IF(IC.EQ.9) GO TO 20
   IRULE=IPSI(IS,1,1)
   IF(IRULE.EQ.0) GO TO 3
   IF(RULE(IRULE,6).EQ.9) GO TO 31
 3 IRULE=IPSI(IS,IC,1)
31 IMAX =IPSI(IS,IC,2)
   IF(IRULE.EQ.0) GO TO 20
   IF(IMAX.EQ.0) IMAX=IPSI(IS,1,2)
   GO TO 6
10 CONTINUE
   IF(IRULE.LT.IPSI(97,1,1)) J2=2
   IS=RULE(IRULE,1)
   IC=RULE(IRULE,6)
   IF(IS.EQ.97) IC=RULE(IRULE,3)
   IRULE=IRULE +RULE(IRULE,26)+1
   IF(IRULE.GT.IMAX) IRULE=0
   IF(IRULE.EQ.0) IMAX=0
   GO TO 6
19 IRULE=0
   IMAX =0
   GO TO 6
C
C        FOR CLASS=9, INCLUDE ALL RULES ON THIS SYMBOL
C
20 IRULE=0
   DO 30 L=1,8
   IF(IPSI(IS,L,1).EQ.0) GO TO 30
   IRULE = IPSI(IS,L,1)
   GO TO 40
30 CONTINUE
40 IMAX = 0
   DO 50 L=8,1,-1
   IF(IPSI(IS,L,2).EQ.0) GO TO 50
   IMAX = IPSI(IS,L,2)
   GO TO 6
50 CONTINUE
 6 IF(ITRACE.NE.0) WRITE(6,300) IS,IC,IRULE,IMAX
300 FORMAT(10X,14HFOR SYMBOL NO.,I3,8H, CLASS ,I3,9H, IRULE= ,I3,
   1 8H, IMAX= ,I3,1H.)
   RETURN
   END
```

182

SUBPROGRAM SYMACH

185

```
      SUBROUTINE SYMACH
C     *****************************************************************
C     *                  THIS SUBROUTINE TESTS FOR A MATCH            *
C     *                  BETWEEN SUBSCRIPTS OF A GIVEN SYMBOL         *
C     *                  AND A CORRESPONDING  GRAM. RULE SYMBOL.      *
C     *                  CALLED BY RERITE FROM ANALYZ.                *
C     *                          WRITTEN BY JAMES D. PRICE  JULY,1970*
C     *****************************************************************
C
      COMMON RULE(900,28), IPSI(100,8,2), RESTRT(5,15), ITABLE(400,29),
     1 SYMIN(29), SYMBOL(29), I, IMAXI,J, JMAX, IRULE, IMAX, IRULE1,
     2 ISTART, ITRACE, MATCH
      COMMON/TREE/ ITREE(50), NODE(7,400), NODE1, NODE2
      INTEGER RULE, RESTRT,SYMIN,SYMBOL
      LOGICAL MATCH
      IF(ITRACE.NE.0) WRITE(6,99) NODE1, SYMIN(1)
   99 FORMAT(39H      SURBOUTINE SYMACH CALLED FOR NODE  ,I3,
     1  7H, SYM.  ,I3,1H.)
      MATCH= .TRUE.
      JJ=IRULE+RULE(IRULE,26)
      DO 3 L=1,17
      IF(L.NE.1) GO TO 1
      IF(SYMIN(1).EQ.RULE(IRULE1,1))GO TO 3
      IF(RULE(IRULE1,1).EQ.97.AND.SYMBOL(1).EQ.97) GO TO 3
      IF(RULE(IRULE1,1).EQ.97.AND.SYMIN(1).EQ.SYMBOL(1)) GO TO 3
      GO TO 4
    1 CONTINUE
      IF(L.EQ.7.AND.SYMIN(8).EQ.0) GO TO 3
      IF(L.EQ.3.AND.RULE(IRULE1,1).NE.97)  GO TO 3
      IF(RULE(IRULE1,L).EQ.9) GO TO 3
      IF(SYMIN(L).EQ.RULE(IRULE1,L)) GO TO 3
      IF(SYMIN(L).EQ.9) GO TO 3
      IF(RULE(IRULE1,L).LT.0.AND.SYMIN(L).LT.9) GO TO 3
      IF(RULE(IRULE1,L).GT.9.AND.RULE(IRULE1,L).EQ.RULE(JJ,L))GO TO 2
      GO TO 4
    2 IF(SYMBOL(L).EQ.RULE(IRULE1,L)) GO TO 3
      IF(SYMBOL(L).LT.9.AND.SYMBOL(L).EQ.SYMIN(L)) GO TO 3
      IF(SYMBOL(L).EQ.9) GO TO 3
      GO TO 4
    3 CONTINUE
      CALL LIMIT(SYMIN)
      RETURN
    4 MATCH=.FALSE.
      IF(ITRACE.NE.0) WRITE(6,300) L
  300 FORMAT(10X,21HMATCH IS FALSE FOR L=,I3,1H.)
      RETURN
      END
```

SUBPROGRAM VARATT

```
      SUBROUTINE VARATT
C
C ***********************************************************************
C *                  THIS SUBROUTINE COMPUTES THE VALUE OF              *
C *                  THE DEPENDENT SUBSCRIPTS OF A GIVEN                *
C *                  GRAMMAR RULE.  CALLED BY RERITE                    *
C *                           FOR USE WITH PROGRAM ANALYZ               *
C *                                WRITTEN BY JAMES D. PRICE JULY,1970 *
C ***********************************************************************
C
      COMMON RULE(900,28), IPSI(100,8,2), RESTRT(5,15), ITABLE(400,29),
     1 SYMIN(29), SYMBOL(29), I, IMAXI,J, JMAX, IRULE, IMAX, IRULE1,
     2 ISTART, ITRACE, MATCH
      COMMON/TREE/ ITREE(50), NODE(7,400), NODE1, NODE2
      INTEGER RULE, RESTRT,SYMIN,SYMBOL
      LOGICAL MATCH
      IF(ITRACE.NE.0) WRITE(6,99)
   99 FORMAT(    30H     SUBROUTINE VARATT CALLED )
      IF(RULE(IRULE1,1).EQ.97.AND.SYMBOL(1).EQ.97) SYMBOL(1)=SYMIN(1)
      NODE(3,NODE1)=NODE2
      DO 1 L=2,22
      IF(RULE(IRULE1,L).EQ.-1) SYMBOL(L)=SYMIN(L)+1
      IF(RULE(IRULE1,L).GT.9.AND.SYMBOL(L).EQ.RULE(IRULE1,L))
     1 SYMBOL(L)= SYMIN(L)
    1 CONTINUE
      IF(SYMBOL(1).NE.SYMIN(1)) RETURN
      IF(SYMBOL(3).EQ.2.AND.SYMBOL(4).NE.0) GO TO 2
      RETURN
    2 IF(SYMBOL(5).EQ.1) SYMBOL(10)=3
      IF(SYMBOL(10).EQ.0)  SYMBOL(10)=SYMIN(10)
      IF(SYMBOL(11).EQ.0)  SYMBOL(11)=SYMIN(11)
      IF(SYMBOL(12).EQ.0)  SYMBOL(12)=SYMIN(12)
      IF(SYMBOL(10).LT.SYMIN(10)) SYMBOL(10)=SYMIN(10)
      IF(SYMBOL(11).GT.SYMIN(11)) SYMBOL(11)=SYMIN(11)
      IF(SYMBOL(12).GT.SYMIN(12)) SYMBOL(12)=SYMIN(12)
      RETURN
      END
```

PROGRAM RULIST

189

```
C
C   ***************************************************************************
C   *   THIS PROGRAM LISTS THE SYMTAX GRAMMAR RULES USED WITH                 *
C   *     PROGRAM ANALYX.                                                     *
C   *                                 WRITTEN BY JAMES D. PRICE JULY, 1970$
C   ***************************************************************************
C
      DIMENSION A(28),B(6),TITLE(12),SYM(100)
      DATA B/6H999   ,6H0     ,6H=     ,6H+     ,6H      ,6H1     /
      RULE=0.
      IPAGE=0
      LINE=0
      READ(5,100) TITLE
  100 FORMAT(12A6)
      READ(5,105) SYM
  105 FORMAT(10(2X,A3))
    1 IF(LINE.GT.0) GO TO 2
      IPAGE=IPAGE+1
      WRITE(6,101) TITLE,IPAGE
  101 FORMAT(1H1///10X,12A6,10X,5HPAGE-,I3///
     1 58H     SYMBOL(MF/KBCLYDNGPRAVIT/S/-W-/X/Z)                RULE NO.)
    2 READ(5,102) I,(A(L),L=2,28)
  102 FORMAT(I3,2(2X,A1),1X,A2,13(2X,A1),1X,A2,4(2X,A1),2A3,2A1,2A2)
      IF(I.EQ.999) GO TO 9
    3 IF(A(25)-B(6)) 5,4,5
    4 WRITE(6,103)
  103 FORMAT(//)
      LINE=LINE+3
      K=4
      IF(A(25)-A(26)) 43,42,43
   42 K=3
   43 CONTINUE
      ISUB=ISUB+1
      IF(RULE-A(24)) 41,10,41
   41 ISUB=1
      RULE=A(24)
      GO TO 10
    5 IF(A(25)-A(26)) 6,7,6
    6 K=4
      IF(A(25)-B(2)) 8,61,8
   61 K=5
      GO TO 8
```

191

```
  7 K=3
  8 WRITE(6,104) SYM(I),(A(L),L=2,22),A(27),A(28),B(K)
104 FORMAT(8X,A3,1H(,2A1,A2,13A1,A2,4A1,2A2,1H),1X,A1)
    GO TO 11
 10 WRITE(6,106) SYM(I),(A(L),L=2,22),A(27),A(28),B(K),A(24),ISUB
106 FORMAT(8X,A3,1H(,2A1,A2,13A1,A2,4A1,2A2,1H),1X,A1,11X,1H(,A3,1H.,
   1 I2,1H))
 11 CONTINUE
    LINE=LINE+1
    IF(K.EQ.5.AND.LINE.GT.40) LINE=0
    GO TO 1
  9 WRITE(6,109)
109 FORMAT(1H1,10(/),1X,17(7H--END--))
    STOP
    END
```

191

Appendix

PART IV

APPENDIX C

SAMPLE OUTPUT

FROM SUBPROGRAM OUTPUT

RESULTS OF REWRITE PASS NO.- 1

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
87	1	0	0	0	3	1	0	0	1	1	3	0	0	0	0	0	0	XYYM	0	0	0	0	0	0	0
87	1	0	0	0	3	1	0	0	1	1	3	0	0	0	0	0	0	NXMN	0	0	0	0	0	0	0
87	1	0	0	0	3	1	0	0	1	1	3	0	0	0	0	0	0	BALQ	0	0	0	0	0	0	0
94	1	0	0	0	1	0	0	0	1	1	3	0	1	1	1	1	12	HYH*	0	0	0	0	0	0	0
87	1	0	0	0	1	1	0	0	1	1	3	0	0	0	0	0	12	SRR*	0	0	0	0	0	0	0
77	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	3	GDL*	0	0	0	0	0	0	0
89	1	0	0	0	1	0	0	0	0	0	0	2	0	0	0	0	1	B***	0	0	0	0	0	0	0
85	1	0	0	0	1	0	0	0	1	1	3	0	0	0	0	0	1	AR&*	0	0	0	0	0	0	0
87	1	0	0	0	3	1	0	0	1	1	3	0	0	0	0	0	0	YSRL	0	0	0	0	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 2

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
87	1	4	3	0	3	1	0	0	1	1	3	0	0	0	0	0	0	XYYM	0	3	0	2	6	4	0
32	1	0	0	0	1	0	0	0	1	1	3	0	1	1	1	1	12	HYH*	0	35	0	1	0	0	0
6	1	0	0	0	1	0	0	9	1	1	3	0	0	0	0	0	0	0000	0	9	0	1	2	9	0
77	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	3	GDL*	0	0	0	0	0	0	0
89	1	0	0	0	1	0	0	0	0	0	0	2	0	0	0	0	1	B***	0	0	0	0	0	0	0
85	1	0	0	0	1	0	0	0	1	1	3	0	0	0	0	0	1	AR&*	0	0	0	0	0	0	0
6	1	0	0	0	3	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	9	0	1	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 3

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
6	1	0	0	0	3	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	9	0	1	0	0	0
33	1	0	0	0	1	0	0	0	1	1	3	0	1	1	1	1	12	HYH*	0	36	0	1	5	17	0
6	1	0	0	0	1	0	0	9	1	1	3	0	0	0	0	0	0	0000	0	9	0	1	2	9	0
77	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	3	GDL*	0	0	0	0	0	0	0
89	1	0	0	0	1	0	0	0	0	0	0	2	0	0	0	0	1	B***	0	0	0	0	0	0	0
85	1	0	0	0	1	0	0	0	1	1	3	0	0	0	0	0	1	AR&*	0	0	0	0	0	0	0
22	1	0	0	0	1	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	25	0	1	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 4

SYMBOL	M	F	E	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
22	1	0	0	0	1	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	25	0	1	0	0	0
34	1	0	0	0	3	0	0	0	1	1	3	0	1	1	1	1	12	HYH*	0	37	0	1	11	17	0
6	1	0	0	0	1	0	0	9	1	1	3	0	0	0	0	0	0	0000	0	9	0	1	2	9	0
77	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	3	GDL*	0	0	0	0	0	0	0
89	1	0	0	0	1	0	0	0	0	0	0	2	0	0	0	0	1	B***	0	0	0	0	0	0	0
23	1	0	0	0	2	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	26	0	2	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 5

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
23	1	0	0	0	1	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	26	0	1	0	0	0
35	1	0	0	0	1	0	0	0	1	1	3	0	1	1	1	1	12	HYH*	0	38	0	1	11	17	0
6	1	0	0	0	1	0	0	9	1	1	3	0	0	0	0	0	0	0000	0	9	0	1	2	9	0
77	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	3	GDL*	0	0	0	0	0	0	0
89	1	0	0	0	1	0	0	0	0	0	0	2	0	0	0	0	1	B***	0	0	0	0	0	0	0
26	1	0	0	0	1	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	29	0	1	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 6

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
26	1	0	0	0	1	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	29	0	1	0	0	0
36	1	0	0	0	1	0	0	0	1	1	3	0	1	1	1	1	12	HYH*	0	39	0	1	2	10	0
6	1	0	0	0	1	0	0	9	1	1	3	0	0	0	0	0	0	0000	0	9	0	1	2	9	0
77	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	3	GDL*	0	0	0	0	0	0	0
38	1	0	0	0	1	0	0	0	0	0	0	2	0	0	0	0	0	0000	0	41	0	2	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 7

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
42	1	0	0	0	1	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	45	0	1	0	0	0
36	1	0	0	0	1	0	0	0	1	1	3	0	1	1	1	1	12	HYH*	0	39	0	1	2	10	0
6	1	0	0	0	1	0	0	9	1	1	3	0	0	0	0	0	0	0000	0	9	0	1	2	9	0
3	1	0	0	0	2	0	0	9	1	1	0	0	0	0	0	0	0	0000	0	6	0	2	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 8

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
42	1	0	0	0	1	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	45	0	1	0	0	0
36	1	0	0	0	1	0	0	0	1	1	3	0	1	1	1	1	12	HYH*	0	39	0	1	2	10	0
6	1	0	0	0	1	0	0	9	1	1	3	0	0	0	0	0	0	0000	0	9	0	1	2	9	0
4	1	0	0	0	1	0	0	9	1	1	0	0	0	0	0	0	0	0000	0	7	0	1	2	10	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 9

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
42	1	0	0	0	1	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	45	0	1	0	0	0
36	1	0	0	0	1	0	0	0	1	1	3	0	1	1	1	1	12	HYH*	0	39	0	1	2	10	0
22	1	0	0	0	1	0	0	9	1	1	3	0	0	0	0	0	0	0000	0	25	0	2	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 10

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
42	1	0	0	0	1	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	45	0	1	0	0	0
36	1	0	0	0	1	0	0	0	1	1	3	0	1	1	1	1	12	HYH*	0	30	0	1	2	10	0
23	1	0	0	0	1	0	0	9	1	1	3	0	0	0	0	0	0	0000	0	26	0	1	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 11

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
42	1	0	0	0	1	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	45	0	1	0	0	0
36	1	0	0	0	1	0	0	0	1	1	3	0	1	1	1	1	12	HYH*	0	39	0	1	2	10	0
26	1	0	0	0	1	0	0	9	1	1	3	0	0	0	0	0	0	0000	0	29	0	1	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 12

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	C	0	0
42	1	0	0	0	1	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	45	0	1	0	0	0
36	1	0	0	0	1	0	0	0	1	1	3	0	1	1	1	1	12	HYH*	0	39	0	1	2	10	0
45	1	0	0	0	3	0	0	9	1	1	3	0	0	0	0	0	0	0000	0	48	0	1	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 13

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
42	1	0	0	0	1	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	45	0	1	0	0	0
36	1	0	0	0	1	0	0	0	1	1	3	0	1	1	1	1	12	HYH*	0	39	0	1	2	10	0
50	1	0	0	0	1	0	0	0	1	1	3	0	1	0	0	0	0	0000	0	53	0	1	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

4-C-13

RESULTS OF REWRITE PASS NO.- 14

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
42	1	0	0	0	1	0	0	2	1	1	3	0	0	0	0	0	0	0000	0	45	0	1	0	0	0
53	1	0	0	0	1	0	0	0	1	1	3	0	1	1	1	1	0	0000	0	56	0	2	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 15

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
62	1	0	0	0	1	0	0	0	1	1	3	0	0	1	1	1	0	0000	0	65	0	2	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 16

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
90	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	HAM*	0	0	0	0	0	0	0
64	1	0	0	0	3	0	0	0	1	1	3	0	0	1	1	1	0	0000	0	67	0	1	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 17

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
71	1	0	0	0	1	0	0	0	1	1	3	0	0	0	0	0	0	0000	0	74	0	2	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

STI

RESULTS OF REWRITE PASS NO.- 18

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULF	EL	NE	REST	SUB	X
75	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0000	0	78	0	1	0	0	0
92	1	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	?***	0	0	0	0	0	0	0

RESULTS OF REWRITE PASS NO.- 19

SYMBOL	M	F	K	B	C	L	Y	D	N	G	P	R	A	V	I	T	S	W	TY	RULE	EL	NE	REST	SUB	X
76	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0000	0	79	0	2	0	0	0

Appendix

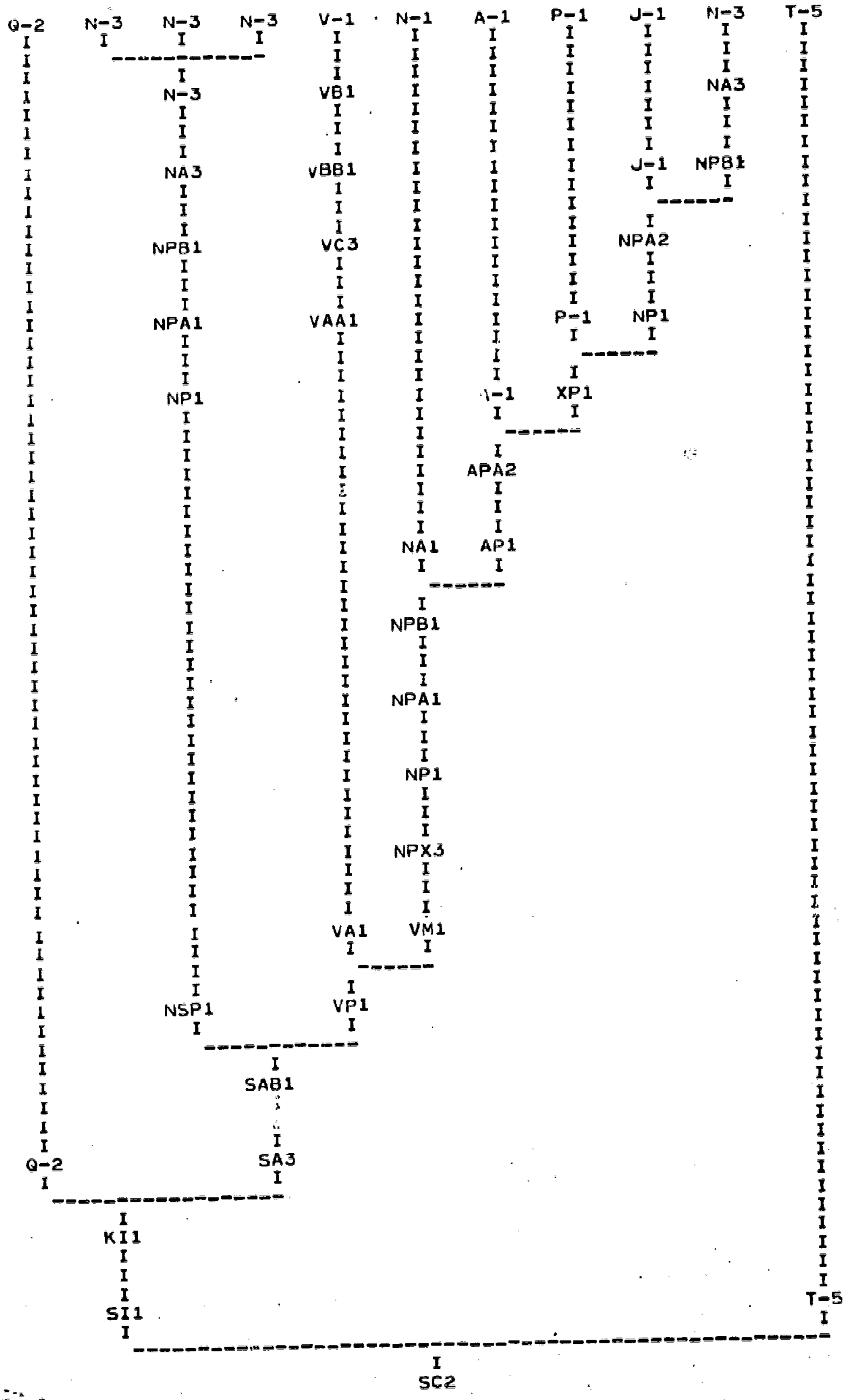
PART IV

APPENDIX D

Examples of Exhaustive Syntactic Analysis by Computer (With Associated Tree Diagrams). Tree diagrams are produced by subprogram DIAGRM; syntax analysis statements are produced by subprogram PARSE.

HEBREW SENTENCE ANALYZED--
HAM XYYM NXMN BYALYQ HYH MSWRR GDWL BAR& YSRAL?

TREE DIAGRAM OF HEBREW SENTENCE 101.



SYNTAX ANALYSIS IF SENTENCE NO. 101.

(1) THE COMPLETED SENTENCE(SC) IS AN INTEROGATIVE SENTENCE.

(2) THE INTEROGATIVE SENTENCE(SI) OF THE COMPLETED SENTENCE(SC) HAS NO DEPENDENT CLAUSE.

(3) THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) QUESTIONS TRUTH/CIRCUMSTANCES OF SN.

(4) THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) IS A DEFINITE INDEPENDENT CLAUSE.

(5) THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) HAS A NAMED SUBJECT.

(6) THE SUBJECT PHRASE(NSP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) IS A NOUN PHRASE.

(7) THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) IS A VERB PHRASE + VRB.MODIF.PHRASE.

(8) THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) IS A SEVEN-TENSE VERB PHRASE.

(9) THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) IS A COPULATIVE PHRASE.

(10) THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) IS A NOUN PHRASE.

(11) THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) IS A REG.NOUN.PHRS.+(APP.NOUN.PHRS).

(12) THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRAS E(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYI NG PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINI TE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) CONTAINS NO CONSTRUCT NOUNS.

(13) THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRAS E(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE P HRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDIC ATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) HAS A BASIC NOUN PHRASE AS NUCLEUS.

(14) THE BASIC NOUN PHRASE(NA) OF THE SIMPLE NOUN PHRASE (NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB M ODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTE NCE(SI) HAS A NONDETERMINATE NOUN.

(15) THE POST-NOMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTERO GATIVE SENTENCE(SI) IS A BASIC POST-NOMINAL ADJ.PHRASE.

(16) THE BASIC POST-NOM.ADJECTIVE PHRASE(APA) OF THE POST-N OMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRAS E(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYI NG PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINI TE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) EXPRESSES THE COMPARATIVE DEGREE.

(17) THE GENERAL NOUN PHRASE(NP) OF THE SUBJECT PHRASE(NSP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPE NDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) IS A REG.NOUN PHRS.+(APP.NOUN.PHRS).

(18) THE PREPOSITION PHRASE(XP) OF THE BASIC POST-NOM.ADJ ECTIVE PHRASE(APA) OF THE POST-NOMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRAS E(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE P HRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDIC ATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) GOVERNS A NOUN PHRASE.

(19) THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRAS

E(NP) OF THE SUBJECT PHRASE(NSP) OF THE DEFINITE IND EPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) CONTAINS NO CONSTRUCT NOUNS.

(20) THE SEVEN-TENSE VERB PHRASE(VAA) OF THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE IND EPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) IS AN EMPHATIC VERB PHRASE.

(21) THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRAS E(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE SUBJECT PHRA SE(NSP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) HAS A BASIC NOUN PHRASE AS NUC LEUS.

(22) THE EMPHATIC VERB PHRASE(VC) OF THE SEVEN-TENSE VERB P HRASE(VAA) OF THE VERB PHRASE(VA) OF THE PREDICATE PH RASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) EXPRESSES NO SPECIAL EMPHASIS.

(23) THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRAS E(NP) OF THE PREPOSITION PHRASE(XP) OF THE BASIC POST-N OM.ADJECTIVE PHRASE(APA) OF THE POST-NOMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULA TIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUS E(KI) OF THE INTEROGATIVE SENTENCE(SI) CONTAINS CONSTRUCT NOUN(S).

(24) THE BASIC NOUN PHRASE(NA) OF THE SIMPLE NOUN PHRASE (NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE SUBJECT PHRASE(NSP) OF THE DEFINI TE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) HAS A (DETERMINATE) PROPER NOUN.

(25) THE THREE-TENSE VERB PHRASE(VBB) OF THE EMPHATIC VER B PHRASE(VC) OF THE SEVEN-TENSE VERB PHRASE(VAA) OF THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTE NCE(SI) IS A VERB OR PARTICIPLE.

(26) THE VERB MOOD PHRASE(VB) OF THE THREE-TENSE VERB P HRASE(VBB) OF THE EMPHATIC VERB PHRASE(VC) OF THE SEVEN- TENSE VERB PHRASE(VAA) OF THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUS E(KI) OF THE INTEROGATIVE SENTENCE(SI) IS A VERB OR INFIN ITIVE ABSOLUTE.

(27) THE BASIC NOUN PHRASE(NA) OF THE SIMPLE NOUN PHRASE (NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE PREPOSITION PHRASE(XP) OF THE BASIC POST-NOM.ADJECTIVE PHRASE(APA) OF THE POST-NOMINAL ADJECTIVE P HRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULA R NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLS E(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) HAS A (DETER MINATE) PROPER NOUN.

(28) THE VERB(V) OF THE VERB MOOD PHRASE(VB) OF THE THREE-TENSE VERB PHRASE(VBB) OF THE EMPHATIC VERB PHRA SE(VC) OF THE SEVEN-TENSE VERB PHRASE(VAA) OF THE VERB P HRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINI TE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) IS IS.

SING., MASC., THIRD, A=1, ACT., IND., PAST.

(29) THE NOUN ABSOLUTE(N) OF THE BASIC NOUN PHRASE(NA) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRAS E(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE P HRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDIC ATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) IS POET.

SING., MASC., THIRD.

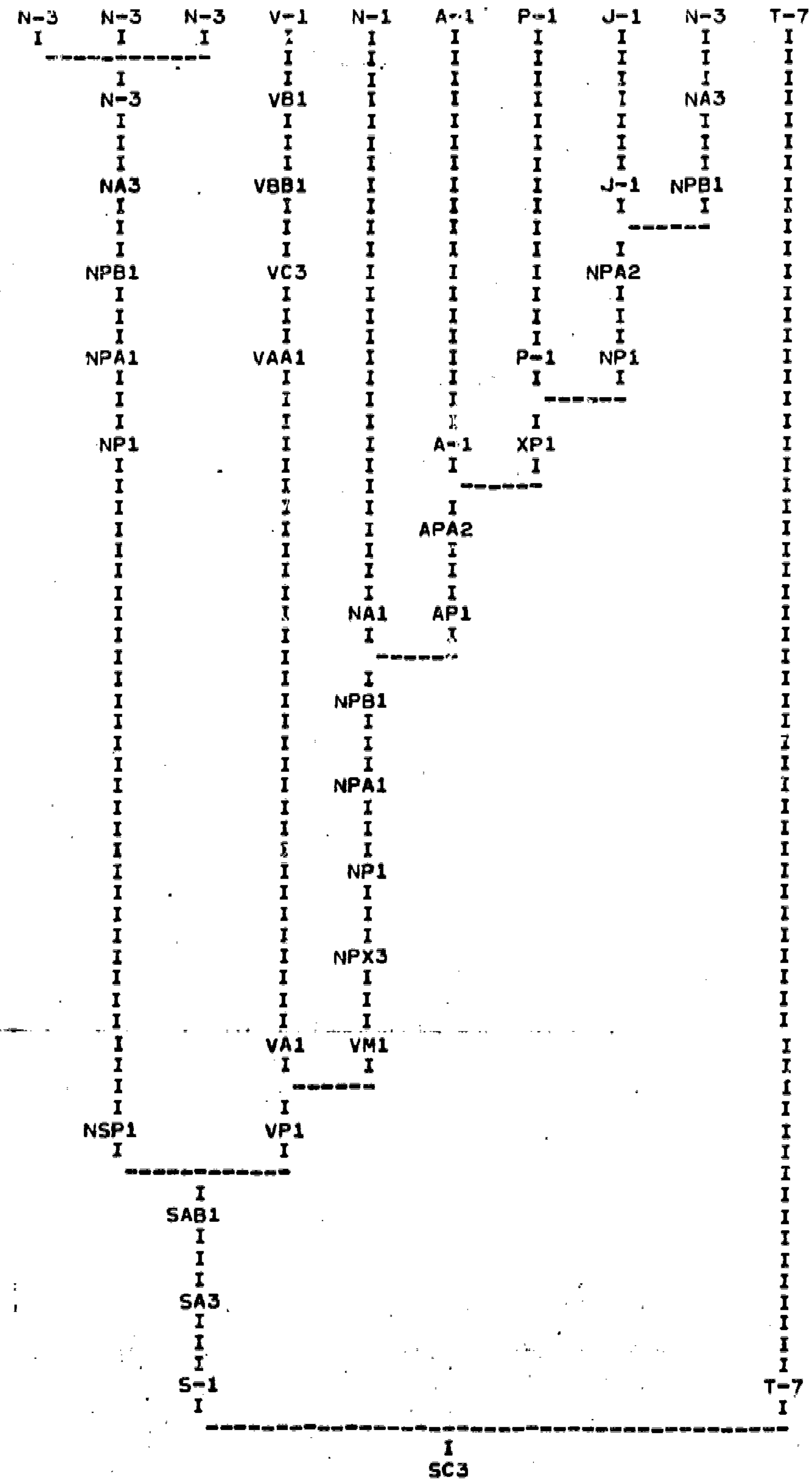
(30) THE NOUN ABSOLUTE(N) OF THE BASIC NOUN PHRASE(NA) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRAS E(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE PREPOSITION PHRASE(XP) OF THE BASIC POST-NOM.ADJECTIVE PHRASE(APA) OF THE POST-NOMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERA L NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE (SA) OF THE INTEROGATIVE CLAUSE(KI) OF THE INTEROGATIVE SENTENCE(SI) IS ISRAEL.

SING., MASC., THIRD.

ANALYSIS COMPLETED FOR SENTENCE NO. 101, NO. OF SYMBOL TESTS= 265.

HEBREW SENTENCE ANALYZED--
XYYM NXMN BYALYQ HYH MSWRR GDWL BAR& YSRAL!

TREE DIAGRAM OF HEBREW SENTENCE 102



SYNTAX ANALYSIS IF SENTENCE NO. 102.

(1) THE COMPLETED SENTENCE(SC) IS AN IMPERATIVE SENTENCE.

(2) THE BASIC SENTENCE(S) OF THE COMPLETED SENTENCE(SC) IS A SIMPLE SENTENCE.

(3) THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A DEFINITE INDEPENDENT CLAUSE.

(4) THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) HAS A NAMED SUBJECT.

(5) THE SUBJECT PHRASE(NSP) OF THE DEFINITE INDEPENDE NT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A NOUN PHRASE.

(6) THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDE NT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A VERB PHRASE + VRB.MODIF.PHRASE.

(7) THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A SEVEN-TENSE VERB PH RASE.

(8) THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(V P) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPE NDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A COPULATIVE PH RASE.

(9) THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHR SE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE IND EPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A NOUN PHRASE.

(10) THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PH RASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A REG.NOU N PHRS.+(APP.NOUN.PHRS).

(11) THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRAS E(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYI NG PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINI TE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) CONTAINS NO CONSTRUCT NOUNS.

(12) THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRAS E(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE P HRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDIC

ATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) HAS A BASIC NOUN PHRASE AS NUCLEUS.

(13) THE BASIC NOUN PHRASE(NA) OF THE SIMPLE NOUN PHRASE (NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB M ODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) HAS A NONDETERMINATE NOUN.

(14) THE POST-NOMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A BASIC POST-NOMINAL ADJ.PHRASE.

(15) THE BASIC POST-NOM.ADJECTIVE PHRASE(APA) OF THE POST-N OMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRAS E(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYI NG PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINI TE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) EXPRESSES THE COMPARATIVE DEGREE.

(16) THE GENERAL NOUN PHRASE(NP) OF THE SUBJECT PHRASE(NSP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPE NDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A REG.NOUN PHRS .+(APP.NOUN.PHRS).

(17) THE PREPOSITION PHRASE(XP) OF THE BASIC POST-NOM.ADJ ECTIVE PHRASE(APA) OF THE POST-NOMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRAS E(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE P HRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDIC ATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) GOVERN S A NOUN PHRASE.

(18) THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRAS E(NP) OF THE SUBJECT PHRASE(NSP) OF THE DEFINITE IND EPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) CONTAINS NO CONSTRUCT NOUNS.

(19) THE SEVEN-TENSE VERB PHRASE(VAA) OF THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE IND EPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS AN EMPHATIC VERB PHRASE.

(20) THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRAS E(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE SUBJECT PHRA SE(NSP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) HAS A BASIC NOUN PHRASE AS NUCLEUS.

(21)   THE EMPHATIC VERB PHRASE(VC)   OF   THE SEVEN-TENSE VERB P
HRASE(VAA)   OF   THE VERB PHRASE(VA)   OF   THE PREDICATE PH
RASE(VP)   OF   THE DEFINITE INDEPENDENT CLSE(SAB)   OF   THE
INDEPENDENT CLAUSE(SA)   OF   THE BASIC SENTENCE(S) EXPRESSES NO
SPECIAL EMPHASIS.

(22)   THE REGULAR NOUN PHRASE(NPA)   OF   THE GENERAL NOUN PHRAS
E(NP)   OF   THE PREPOSITION PHRASE(XP)   OF   THE BASIC POST-N
OM.ADJECTIVE PHRASE(APA)   OF   THE POST-NOMINAL ADJECTIVE PHRASE(
AP)   OF   THE SIMPLE NOUN PHRASE(NPB)   OF   THE REGULAR NOUN
PHRASE(NPA)   OF   THE GENERAL NOUN PHRASE(NP)   OF   THE COPULA
TIVE PHRASE(NPX)   OF   THE VERB MODIFYING PHRSE(VM)   OF   THE
PREDICATE PHRASE(VP)   OF   THE DEFINITE INDEPENDENT CLSE(SAB)
OF   THE INDEPENDENT CLAUSE(SA)   OF   THE BASIC SENTENCE(S)
CONTAINS CONSTRUCT NOUN(S).

(23)   THE BASIC NOUN PHRASE(NA)   OF   THE SIMPLE NOUN PHRASE
(NPB)   OF   THE REGULAR NOUN PHRASE(NPA)   OF   THE GENERAL NOUN
PHRASE(NP)   OF   THE SUBJECT PHRASE(NSP)   OF   THE DEFINI
TE INDEPENDENT CLSE(SAB)   OF   THE INDEPENDENT CLAUSE(SA)   OF
THE BASIC SENTENCE(S) HAS A (DETERMINATE) PROPER NOUN.

(24)   THE THREE-TENSE VERB PHRASE(VBB)   OF   THE EMPHATIC VER
B PHRASE(VC)   OF   THE SEVEN-TENSE VERB PHRASE(VAA)   OF   THE
VERB PHRASE(VA)   OF   THE PREDICATE PHRASE(VP)   OF   THE
DEFINITE INDEPENDENT CLSE(SAB)   OF   THE INDEPENDENT CLAUSE(SA)
OF   THE BASIC SENTENCE(S) IS A VERB OR PARTICIPLE.

(25)   THE VERB MOOD PHRASE(VB)   OF   THE THREE-TENSE VERB P
HRASE(VBB)   OF   THE EMPHATIC VERB PHRASE(VC)   OF   THE SEVEN-
TENSE VERB PHRASE(VAA)   OF   THE VERB PHRASE(VA)   OF   THE
PREDICATE PHRASE(VP)   OF   THE DEFINITE INDEPENDENT CLSE(SAB)
OF   THE INDEPENDENT CLAUSE(SA)   OF   THE BASIC SENTENCE(S)
IS A VERB OR INFINITIVE ABSOLUTE.

(26)   THE BASIC NOUN PHRASE(NA)   OF   THE SIMPLE NOUN PHRASE
(NPB)   OF   THE REGULAR NOUN PHRASE(NPA)   OF   THE GENERAL NOUN
PHRASE(NP)   OF   THE PREPOSITION PHRASE(XP)   OF   THE BASIC
POST-NOM.ADJECTIVE PHRASE(APA)   OF   THE POST-NOMINAL ADJECTIVE P
HRASE(AP)   OF   THE SIMPLE NOUN PHRASE(NPB)   OF   THE REGULA
R NOUN PHRASE(NPA)   OF   THE GENERAL NOUN PHRASE(NP)   OF   THE
COPULATIVE PHRASE(NPX)   OF   THE VERB MODIFYING PHRSE(VM)   OF
THE PREDICATE PHRASE(VP)   OF   THE DEFINITE INDEPENDENT CLS
E(SAB)   OF   THE INDEPENDENT CLAUSE(SA)   OF   THE BASIC SENTEN
CE(S) HAS A (DETERMINATE) PROPER NOUN.

(27)   THE VERB(V)   OF   THE VERB MOOD PHRASE(VB)   OF
THE THREE-TENSE VERB PHRASE(VBB)   OF   THE EMPHATIC VERB PHRA
SE(VC)   OF   THE SEVEN-TENSE VERB PHRASE(VAA)   OF   THE VERB P
HRASE(VA)   OF   THE PREDICATE PHRASE(VP)   OF   THE DEFINI
TE INDEPENDENT CLSE(SAB)   OF   THE INDEPENDENT CLAUSE(SA)   OF
THE BASIC SENTENCE(S)   IS   IS.
SING.,   MASC.,   SECD.,   A=1,   ACT.,   IMPV.,   FUTR..

(28)   THE NOUN ABSOLUTE(N)   OF   THE BASIC NOUN PHRASE(NA)

THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS
POET.
SING., MASC., SECD..

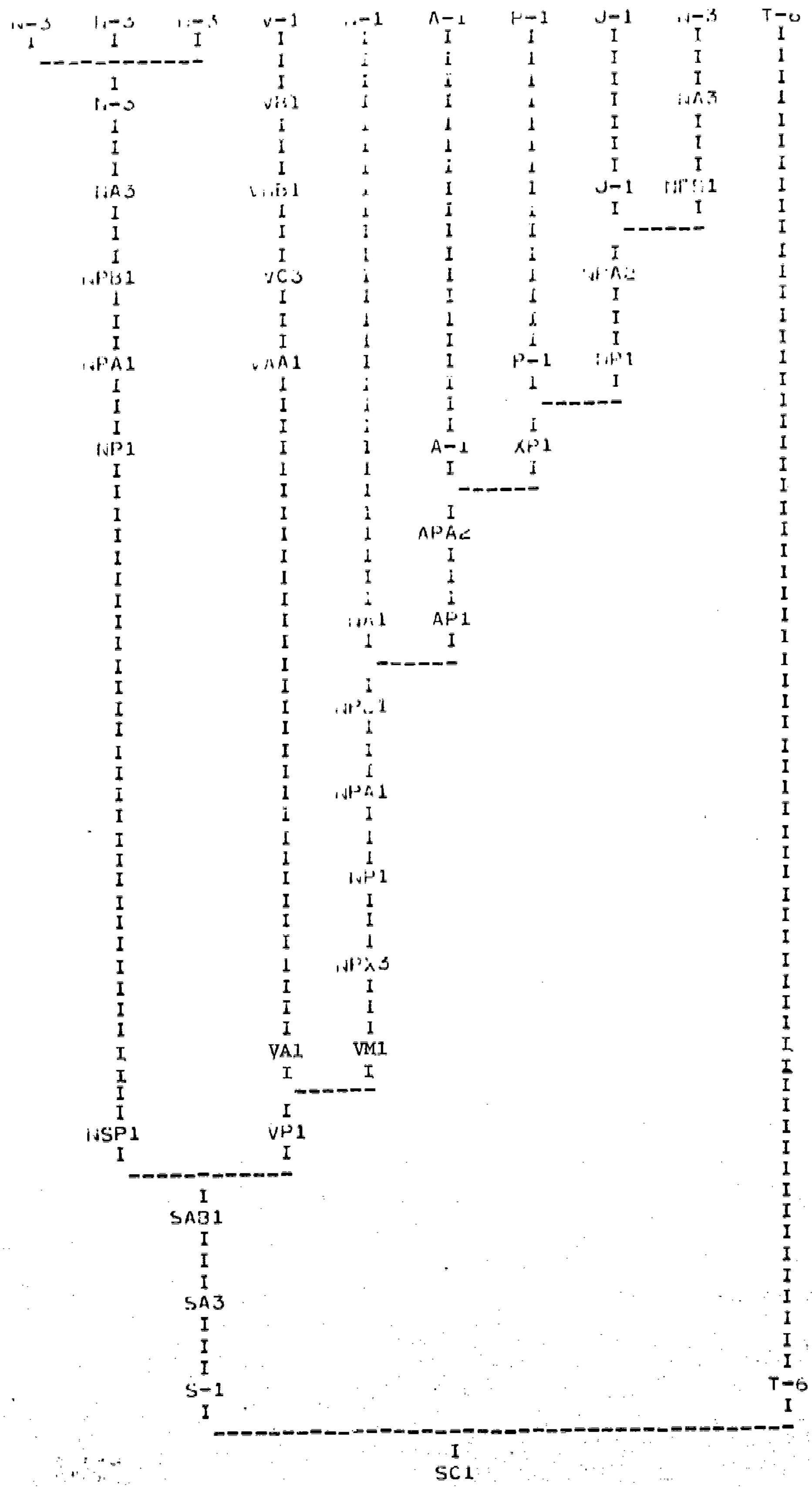
(29) THE NOUN ABSOLUTE(N) OF THE BASIC NOUN PHRASE(NA)
OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRAS
E(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE PREPOSITION
PHRASE(XP) OF THE BASIC POST-NOM.ADJECTIVE PHRASE(APA) OF
THE POST-NOMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN
PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERA
L NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE
VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF
THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE
(SA) OF THE BASIC SENTENCE(S) IS ISRAEL.

SING., MASC., THIRD.

YSIS COMPLETED FOR SENTENCE NO. 102, NO. OF SYMBOL TESTS= 261.

HEBREW SENTENCE ANALYZED--
AYYM NXMN BYALYG YHYH MSWRR GDWL BARS YSRAL.

TREE DIAGRAM OF HEBREW SENTENCE 103.



SYNTAX ANALYSIS IF SENTENCE NO. 103.

(1) THE COMPLETED SENTENCE(SC) IS A DECLARATIVE SENTENCE.

(2) THE BASIC SENTENCE(S) OF THE COMPLETED SENTENCE(SC) IS A SIMPLE SENTENCE.

(3) THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A DEFINITE INDEPENDENT CLAUSE.

(4) THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) HAS A NAMED SUBJECT.

(5) THE SUBJECT PHRASE(NSP) OF THE DEFINITE INDEPENDE NT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A NOUN PHRASE.

(6) THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDE NT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A VERB PHRASE + VRB.MODIF.PHRASE.

(7) THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A SEVEN-TENSE VERB PH RASE.

(8) THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(V P) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPE NDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A COPULATIVE PH RASE.

(9) THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHR SE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE IND EPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A NOUN PHRASE.

(10) THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PH RASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A REG.NOU N PHRS.+(APP.NOUN.PHRS).

(11) THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRAS E(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYI NG PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINI TE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) CONTAINS NO CONSTRUCT NOUNS.

(12) THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRAS E(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE P HRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDIC

ATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) HAS A BASIC NOUN PHRASE AS NUCLEUS.

(13) THE BASIC NOUN PHRASE(NA) OF THE SIMPLE NOUN PHRASE (NPB) OF THE REGULAR NOUN .RASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB M ODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) HAS A NONDETERMINATE NOUN.

(14) THE POST-NOMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A BASIC POST-NOMINAL ADJ.PHRASE.

(15) THE BASIC POST-NOM.ADJECTIVE PHRASE(APA) OF THE POST-N OMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRAS E(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYI NG PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINI TE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) EXPRESSES THE COMPARATIVE DEGREE.

(16) THE GENERAL NOUN PHRASE(NP) OF THE SUBJECT PHRASE(NSP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPE NDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A REG.NOUN PHRS .+(APP.NOUN.PHRS).

(17) THE PREPOSITION PHRASE(XP) OF THE BASIC POST-NOM.ADJ ECTIVE PHRASE(APA) OF THE POST-NOMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRAS E(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE P HRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDIC ATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) GOVERN S A NOUN PHRASE.

(18) THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRAS E(NP) OF THE SUBJECT PHRASE(NSP) OF THE DEFINITE IND EPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) CONTAINS NO CONSTRUCT NOUNS.

(19) THE SEVEN-TENSE VERB PHRASE(VAA) OF THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE IND EPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS AN EMPHATIC VERB PHRASE.

(20) THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRAS E(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE SUBJECT PHRA SE(NSP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) HAS A BASIC NOUN PHRASE AS NUCLEUS.

(21) THE EMPHATIC VERB PHRASE(VC) OF THE SEVEN-TENSE VERB P
HRASE(VAA) OF THE VERB PHRASE(VA) OF THE PREDICATE PH
RASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE
INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) EXPRESSES NO
SPECIAL EMPHASIS.

(22) THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRAS
E(NP) OF THE PREPOSITION PHRASE(XP) OF THE BASIC POST-N
OM.ADJECTIVE PHRASE(APA) OF THE POST-NOMINAL ADJECTIVE PHRASE(
AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN
PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULA
TIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE
PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB)
OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S)
CONTAINS CONSTRUCT NOUN(S).

(23) THE BASIC NOUN PHRASE(NA) OF THE SIMPLE NOUN PHRASE
(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN
PHRASE(NP) OF THE SUBJECT PHRASE(NSP) OF THE DEFINI
TE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF
THE BASIC SENTENCE(S) HAS A (DETERMINATE) PROPER NOUN.

(24) THE THREE-TENSE VERB PHRASE(VBB) OF THE EMPHATIC VER
B PHRASE(VC) OF THE SEVEN-TENSE VERB PHRASE(VAA) OF THE
VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE
DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA)
OF THE BASIC SENTENCE(S) IS A VERB OR PARTICIPLE.

(25) THE VERB MOOD PHRASE(VB) OF THE THREE-TENSE VERB P
HRASE(VBB) OF THE EMPHATIC VERB PHRASE(VC) OF THE SEVEN-
TENSE VERB PHRASE(VAA) OF THE VERB PHRASE(VA) OF THE
PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB)
OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S)
IS A VERB OR INFINITIVE ABSOLUTE.

(26) THE BASIC NOUN PHRASE(NA) OF THE SIMPLE NOUN PHRASE
(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN
PHRASE(NP) OF THE PREPOSITION PHRASE(XP) OF THE BASIC
POST-NOM.ADJECTIVE PHRASE(APA) OF THE POST-NOMINAL ADJECTIVE P
HRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULA
R NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE
COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF
THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLS
E(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTEN
CE(S) HAS A (DETERMINATE) PROPER NOUN.

(27) THE VERB(V) OF THE VERB MOOD PHRASE(VB) OF
THE THREE-TENSE VERB PHRASE(VBB) OF THE EMPHATIC VERB PHRA
SE(VC) OF THE SEVEN-TENSE VERB PHRASE(VAA) OF THE VERB P
HRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINI
TE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF
THE BASIC SENTENCE(S) IS IS.
SING., MASC., THIRD, A=1, ACT., IND., FUTR..

(28) THE NOUN ABSOLUTE(N) OF THE BASIC NOUN PHRASE(NA)

OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS POET.

SING., MASC., THIRD.

(29) THE NOUN ABSOLUTE(N) OF THE BASIC NOUN PHRASE(NA) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE PREPOSITION PHRASE(XP) OF THE BASIC POST-NOM.ADJECTIVE PHRASE(APA) OF THE POST-NOMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE COPULATIVE PHRASE(NPX) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE (SA) OF THE BASIC SENTENCE(S) IS ISRAEL.

SING., MASC., THIRD.

ANALYSIS COMPLETED FOR SENTENCE NO. 103, NO. OF SYMBOL TESTS= 263.

HEBREW SENTENCE ANALYZED--
HWA YSB BKPR QTN.

TREE DIAGRAM OF HEBREW SENTENCE NO. 4

```
 R-2    V-1    P-1    N-1    A-1    T-6
  I      I      I      I      I      I
  I      I      I      I      I      I
  I      I      I      I      I      I
 RSP1   VB1     I      I    APA1     I
  I      I      I      I      I      I
  I      I      I      I      I      I
  I      I      I      I      I      I
  I     VBB1    I     NA1    AP1     I
  I      I      I      I      I      I
  I      I      I      +-------      I
  I      I      I      I             I
  I     VC3     I     NPB1           I
  I      I      I      I             I
  I      I      I      I             I
  I      I      I      I             I
  I     VAA1    I     NPA1           I
  I      I      I      I             I
  I      I      I      I             I
  I      I      I      I             I
  I      I     P-1    NP1            I
  I      I      I      I             I
  I      I      ----+---             I
  I      I      I                    I
  I      I     XP1                   I
  I      I      I                    I
  I      I      I                    I
  I      I      I                    I
  I     VA1    VM5                   I
  I      I      I                    I
  I      --+-----                    I
  I      I                           I
 NSP2   VP1                          I
  I      I                           I
  ---+----                           I
  I                                  I
 SAB1                                I
  I                                  I
  I                                  I
  I                                  I
 SA3                                 I
  I                                  I
  I                                  I
  I                                  I
 S-1                                T-6
  I                                  I
  -------+-------------+--------------
                I
               SC1
```

SYNTAX ANALYSIS IF SENTENCE NO. 4.

(1) THE COMPLETED SENTENCE(SC) IS A DECLARATIVE SENTENCE.

(2) THE BASIC SENTENCE(S) OF THE COMPLETED SENTENCE(SC) IS A SIMPLE SENTENCE.

(3) THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A DEFINITE INDEPENDENT CLAUSE.

(4) THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) HAS A NAMED SUBJECT.

(5) THE SUBJECT PHRASE(NSP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A SUBJECT PRONOUN PHRASE.

(6) THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A VERB PHRASE + VRB.MODIF.PHRASE.

(7) THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A SEVEN-TENSE VERB PHRASE.

(8) THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A PREPOSITIONAL PHRASE.

(9) THE PREPOSITION PHRASE(XP) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) GOVERNS A NOUN PHRASE.

(10) THE GENERAL NOUN PHRASE(NP) OF THE PREPOSITION PHRASE (XP) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A REG.NOUN PHRS.+(APP.NOUN.PHRS).

(11) THE SEVEN-TENSE VERB PHRASE(VAA) OF THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS AN EMPHATIC VERB PHRASE.

(12) THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE PREPOSITION PHRASE(XP) OF THE VERB MODIFYI PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINI INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF

THE BASIC SENTENCE(S) CONTAINS NO CONSTRUCT NOUNS.

(13) THE EMPHATIC VERB PHRASE(VC) OF THE SEVEN-TENSE VERB PHRASE(VAA) OF THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) EXPRESSES NO SPECIAL EMPHASIS.

(14) THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE PREPOSITION PHRASE(XP) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) HAS A BASIC NOUN PHRASE AS NUCLEUS.

(15) THE THREE-TENSE VERB PHRASE(VBB) OF THE EMPHATIC VERB PHRASE(VC) OF THE SEVEN-TENSE VERB PHRASE(VAA) OF THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A VERB OR PARTICIPLE.

(16) THE BASIC NOUN PHRASE(NA) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE PREPOSITION PHRASE(XP) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) HAS A NONDETERMINATE NOUN.

(17) THE POST-NOMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE PREPOSITION PHRASE(XP) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A BASIC POST-NOMINAL ADJ.PHRASE.

(18) THE SUBJECT PRONOUN PHRASE(RSP) OF THE SUBJECT PHRASE(NSP) , OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A SUBJECT PRONOUN +(APPOS.N.PH.).

(19) THE VERB MOOD PHRASE(VB) OF THE THREE-TENSE VERB PHRASE(VBB) OF THE EMPHATIC VERB PHRASE(VC) OF THE SEVEN-TENSE VERB PHRASE(VAA) OF THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS A VERB OR INFINITIVE ABSOLUTE.

(20) THE BASIC POST-NOM.ADJECTIVE PHRASE(APA) OF THE POST-NOMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE PREPOSITION PHRASE(XP) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) EXPRESSES THE NONCOMPARATIVE DEGREE.

(21) THE PRONOUN(R) OF THE SUBJECT PRONOUN PHRASE(RSP) OF THE SUBJECT PHRASE(NSP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS HE.
SING., MASC., THIRD.

(22) THE VERB(V) OF THE VERB MOOD PHRASE(VB) OF THE THREE-TENSE VERB PHRASE(VBB) OF THE EMPHATIC VERB PHRASE(VC) OF THE SEVEN-TENSE VERB PHRASE(VAA) OF THE VERB PHRASE(VA) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS DWELL.
SING., MASC., THIRD, R=2, A=5, ACT., IND., PAST.

(23) THE NOUN ABSOLUTE(N) OF THE BASIC NOUN PHRASE(NA) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE PREPOSITION PHRASE(XP) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) OF THE BASIC SENTENCE(S) IS VILLAGE.
SING., MASC., THIRD.

(24) THE ADJECTIVE(A) OF THE BASIC POST-NOM.ADJECTIVE PHRASE(APA) OF THE POST-NOMINAL ADJECTIVE PHRASE(AP) OF THE SIMPLE NOUN PHRASE(NPB) OF THE REGULAR NOUN PHRASE(NPA) OF THE GENERAL NOUN PHRASE(NP) OF THE PREPOSITION PHRASE(XP) OF THE VERB MODIFYING PHRSE(VM) OF THE PREDICATE PHRASE(VP) OF THE DEFINITE INDEPENDENT CLSE(SAB) OF THE INDEPENDENT CLAUSE(SA) F THE BASIC SENTENCE(S) IS SMALL.

SING., MASC..

ANALYSIS COMPLETED FOR SENTENCE NO. 4, NO. OF SYMBOL TESTS= 196.