

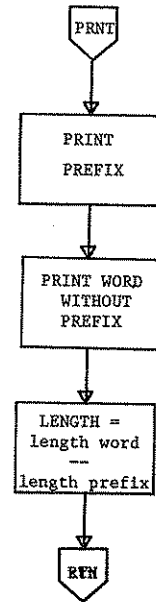
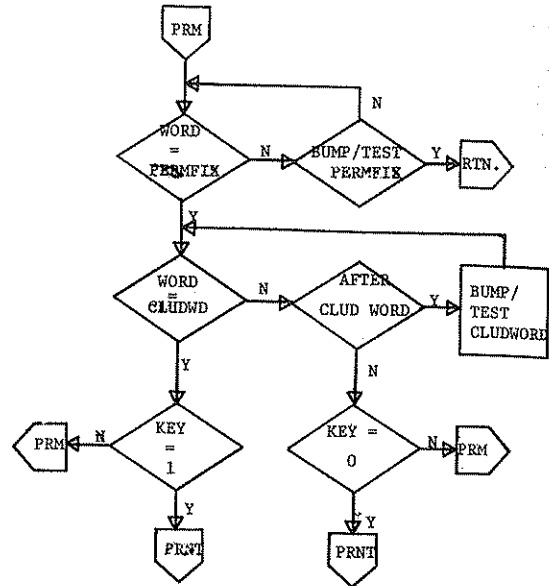
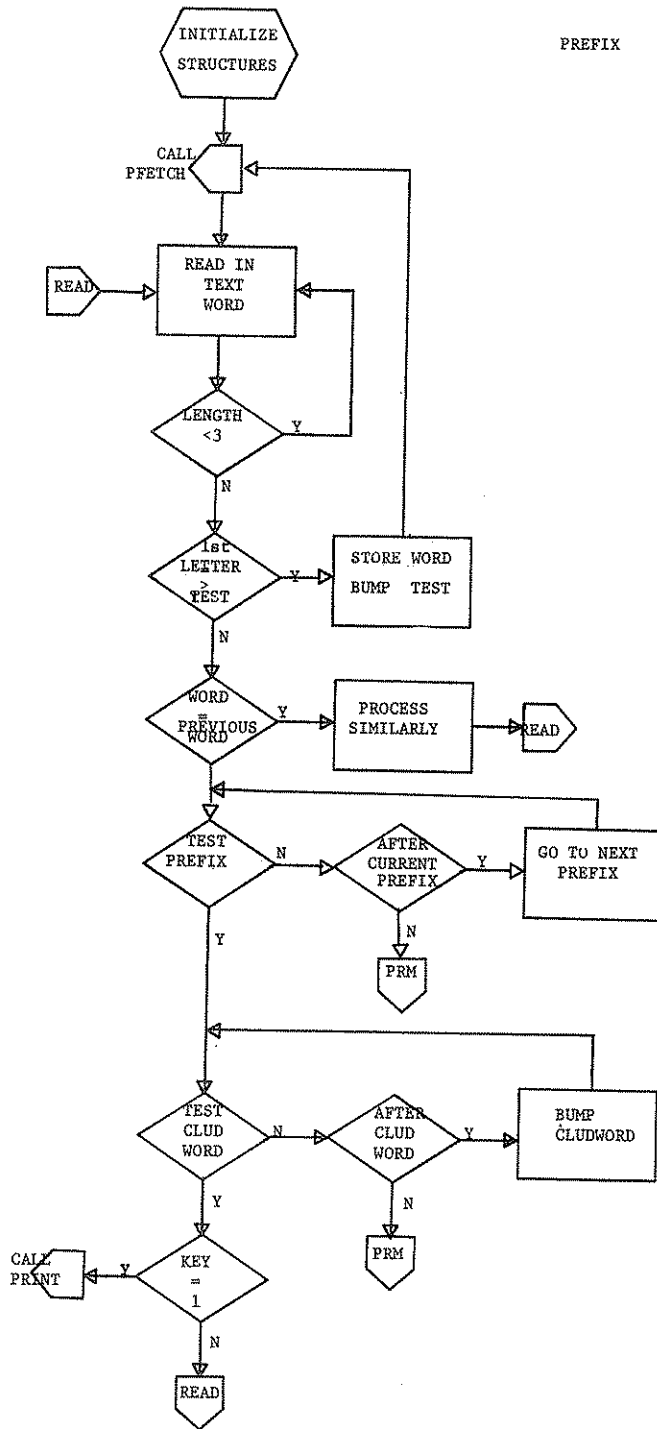
APPENDIX F

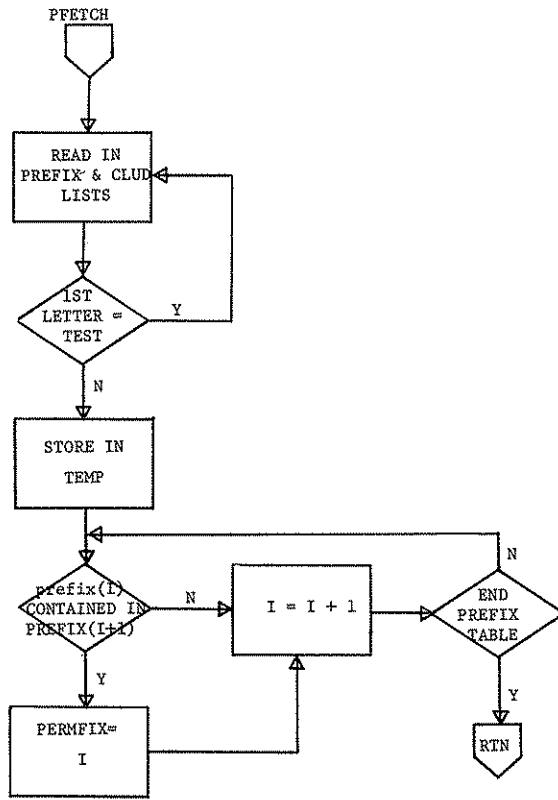
PREFIX Program and Table Listing

by

John B. Smith

PREFIX





RTN.

BUMP /
TEST
CLUDWORD

RM

PREFIX: PROCEDURE OPTIONS(MAIN):

STMT LEVEL NEST

1 PREFIX: PROCEDURE OPTIONS(MAIN):

```

/* ***** */
/* PREFIX IS A GENERAL PURPOSE PROGRAM USED */
/* TO ANALYZE USAGE OF ENGLISH PREFIXES. */
/* IF DOES THIS BY STRIPPING A WORD OF ITS */
/* PREFIX AND REPRODUCING THE ROOT FORM OF */
/* THE WORD WITH ITS DETACHED PREFIX. */
/* THE SPECIFIC FUNCTION OF PREFIX IN THE */
/* VIA PACKAGE IS TO CREATE DUPLICATES OF WORDS */
/* WITH PREFIX DETACHED AND TO INSERT THESE */
/* FORMS INTO THE DATA STREAM ALONG WITH THE */
/* ORIGINAL FORM OF THE WORD. THE RESULT IS */
/* THAT THE FREQUENCY COUNTS OF THE ROOT FORM */
/* OF THE WORD WILL BE MODIFIED, PERHAPS */
/* FORCING THE TOTAL OVER PARAMETERS KEYING */
/* OTHER ANALYTIC STEPS. */
/* ***** */
/* ENGLISH PREFIXES, ARRANGED IN ALPHABETIC */
/* ORDER ARE LOADED INTO A STRUCTURE ALONG WITH */
/* A LIST OF WORDS THAT ARE EITHER EXCLUSION */
/* LISTS FOR A PARTICULAR PREFIX OR INCLUSION */
/* FORMS, I.E. WORDS THAT DO HAVE */
/* LEGITIMATE PREFIXES ATTACHED. THE NATURE OF */
/* THE LIST IS DETERMINED BY A KEY ALSO LOADED */
/* INTO THE STRUCTURE. */
/* NEXT WORDS OR WORDS UNDER ANALYSIS ARE ALSO */
/* ARRANGED IN ALPHABETIC ORDER AND ARE EX- */
/* AMINED ONE AT A TIME. IF THE FIRST X CHARAC- */
/* TERS OF A WORD (CORRESPONDING TO THE LENGTH */
/* OF THE PREFIX) MATCH THE PREFIX, THEN A */
/* SEARCH IS MADE OF THE 'CLUD' LIST ASSO- */
/* CIATED WITH THE PREFIX. IF A MATCH IS FOUND */
/* THEN THE FIRST X CHARACTERS ARE STRIPPED OR */
/* NOT DEPENDING UPON WHETHER THE LIST IS AN */
/* INCLUSION OR AN EXCLUSION LIST. */
/* ***** */
/* MAIN STRUCTURE THAT HOLDS PREFIXES, CLUD LIST */
/* AND KEY. THE PROGRAM READS IN ALL PREFIXES */
/* FOR A PARTICULAR LETTER OF THE ALPHABET. */
/* ***** */
DCL 01 PTABLE (35),
    02 PREFIX CHAR(8) VARYING,
    02 KEY FIXED DEC(1),
    02 CLJDD(300) CHAR(18) VARYING:

/* TEMP STORES THE FIRST PREFIX OF THE NEXT */
/* LETTER OF THE ALPHABET. TEMP BECOMES PTABLE */
/* ( ) WHEN THE STRUCTURE IS NEXT LOADED. */
/* ***** */
DCL 01 TEMP,
    02 TEMPFX CHAR(3) VARYING INITIAL (' '),
    02 TEMPFX FIXED DEC(1),
    02 TEMPCLD(300) CHAR(18) VARYING:

```

```

PREFIX: PROCEDURE OPTIONS (MAIN);

STMT LEVEL NEST
4 1 DCL LETR (26) CHAR(1); /* LETR = ARRAY HOLDING ALPHABET FOR TESTS AND
/* CONTROL OF MAIN DO LOOP.
5 1 DCL TOP FIXED DEC(2) INITIAL(0); /*
6 1 DCL WKWORD CHAR(18); /* WORD BEING TESTED FOR PREFIX
7 1 DCL ALLWORD CHAR(13);
8 1 DCL LSPWORD CHAR(18);
9 1 DCL REJECT CHAR(18);
10 1 DCL 01 DUPREC,
02 DUPFIX CHAR(8) VARYING,
02 DUPWRD CHAR(13) VARYING;
11 1 DCL CH CHAR(4);
12 1 DCL N FIXED DEC(2) INITIAL(1);
13 1 DCL P FIXED DEC(3);
14 1 DCL X FIXED DEC(1);
15 1 DCL PNO FIXED DEC(3) INITIAL(1);
16 1 DCL Y FIXED DEC(2);
17 1 DCL PERFIX (10) FIXED DEC(2);
18 1 DCL PERYCLD (10) CHAR(18) VARYING;
19 1 DCL PRSTL FIXED DEC(3) INITIAL(0);
20 1 DCL TEMPWD CHAR(18) INITIAL(' ');
21 1 DCL COUNT FIXED DEC(5) INITIAL(0);
22 1 DCL COUNTS FIXED DEC(5) INITIAL(0);
23 1 DCL LAST CHAR(18) INITIAL(' ');
24 1 ON ENDFILE(LIND) GO TO JUT;

26 1 PUT PAGE;
27 1 PUT EDIT('WORD', 'PREFIX', 'STEM + ENDINGS', 'PREFIX OUTPUT: PAGE ',
PNO(COL(1), A, COL(20), A, COL(30)), A, COL(80), A, P(3));
28 1 PNO = PNO + 1;
29 1 PUT SKIP(2);
30 1 ON ENDPAGE BEGIN;
32 2 PUT PAGE;
33 2 PUT EDIT('WORD', 'PREFIX', 'STEM + ENDINGS', 'PREFIX OUTPUT: PAGE ',
PNO(COL(1), A, COL(23), A, COL(33), A, COL(80), A, P(3));
34 2 PNO = PNO + 1;
35 2 PUT SKIP(2);
36 2 END;

37 1 GET EDIT(LETR, CH(26 A(1), X(53), A(1)));

/* MAIN DO LOOP THAT CONTROLS TEST LETTER OF
/* ALPHABET.
38 1 DO I = 1 TO 26;
39 1 IF LETR(I) = 'J' THEN GO TO BUMP;
41 1 N = 1;
42 1 P = 1;

```

PREFIX: PROCEDURE OPTIONS (MAIN);

STMT LEVEL NEST

```

43 1 1 CALL PFETCH; /* CALLS SUBPROCEDURE THAT LOADS PREFIX TABLES. */
/* TESTS TO SEE IF CURRENT BATCH OF PREFIXES
46 1 1 /* CORRESPOND WITH CURRENT TEST LETTER. */
/* TEMPWORD IS LAST WORD READ BEFORE PREFIX
47 1 1 /* STRUCTURE LOADED. ID DID NOT MATCH LETTER(L). */
48 1 1 /* AT THAT TIME. */
/* THEN DO:
49 1 1
50 1 1 A: IF TEMPWD = ' ' THEN DO:
51 1 1   A: IF TEMPWD = TEMPWD;
52 1 1   TEMPWD = ' ';
53 1 1   END;
54 1 1 ELSE DO:
55 1 1   A: DCL JUNK1 FIXED DEC(2) INITIAL(0);
56 1 1   DCL JUNK2 CHAR(16);
57 1 1   GET FILE (LIND) EDIT (JUNK1, JUNK2, ALLWORD, (F(2), A(16), A(18)));
58 1 1   COUNT = COUNT + 1;
59 1 1   PUT FILE (ADD) EDIT (JUNK1, JUNK2, ALLWORD, (F(2), A(16), A(18)));
60 1 1   DO I3 = 18 TO 1 BY -1;
61 1 1   IF SUBSTR(ALLWORD,I3,1) = ' ' THEN GO TO OUTA;
62 1 1   END;
63 1 1 OUTA:
64 1 1 IF I3 <= 3 THEN GO TO A;
65 1 1 IF I3 <= 3 THEN GO TO A;
66 1 1 IF I3 <= 3 THEN GO TO A;
67 1 1 IF I3 <= 3 THEN GO TO A;
68 1 1 IF I3 <= 3 THEN GO TO A;
69 1 1 IF I3 <= 3 THEN GO TO A;
/* PROGRAM DISCARDS ALL WORDS WITH FEWER THAN 4
/* LETTERS IN THEM, ASSUMING SUCH WORDS DO NOT
/* HAVE LEGITIMATE PREFIXES.
/* CHECKS CURRENT WORD FOR LETTER MATCH PREFIXES
/* TESTS TO SEE IF CURRENT WORD IDENTICAL TO
/* LAST WORD. IF SO AND IF LAST WORD DID NOT
/* HAVE AN ALLOWABLE PREFIX, THEN THIS WORD IS
/* SKIPPED ALSO.
/* IF LAST WORD STRIPPED OF PREFIX, THEN
/* CURRENT WORD IS STRIPPED WITHOUT GOING
/* THROUGH ENTIRE PROCEDURE.

```

A(36),
F(2),
A(18)

JUNK2, JUNK1

| *clary - prefix length*

PREFIX: PROCEDURE OPTIONS (MAIN);

STMT LEVEL NEST

```

71 1 1 IF KWWORD = LSTWORD THEN DO;
73 1 1 CALL PRINT;
74 1 1 GO TO A;
75 1 1 END;

76 1 1 PRSTL = N;
/* SINCE WORDS AND PREFIXES IN ALPHABETICAL
/* ORDER, LAST PREFIX ADDRESS IS STORED SO
/* THAT SEARCH CONTINUES FROM HERE.
77 1 1 IF KWWORD = REJECT THEN GO TO SKIP1;
79 1 1 IF KWWORD = LSTWORD THEN GO TO SKIP1;
81 1 1 PUF EDIT(KWWORD) (SKIP, COL(1), A);
82 1 1 SKIP1:
X = 1;

83 1 1 DO L = N TO TOP;

84 1 2 IF SUBSTR(KWWORD,1,LENGTH(PREFIX(L))) > PREFIX(L) THEN GO TO BUMPL;
/* IF FIRST X LETTERS PAST CURRENT PREFIX,
/* SKIPS TO NEXT PREFIX.

86 1 2 IF SUBSTR(KWWORD,1,LENGTH(PREFIX(L))) = PREFIX(L)
87 1 2 THEN DO N = P TO 300 WHILE (PTABLE(L).CLUDWD(N) = ' ');
/* WHEN MATCH OF FIRST X LETTERS WITH PREFIX IS
/* FOUND SEARCH IS MADE OF CLUD LIST.
/* TESTS TO SEE THAT WORD IS NOT PAST CLUD WORD.
88 1 3 IF PTABLE(L).CLUDWD(N) > SUBSTR(KWWORD,1,LENGTH(PTABLE(L).CLUDWD(N)))
89 1 3 THEN DO:
90 1 3 P = M;
91 1 3 IF SUBSTR(KWWORD,1,LENGTH(PREFIX(L + 1))) >= PREFIX(L + 1) THEN GO TO
BUMPL;
93 1 3 ELSE GO TO PRMLOOP;
94 1 3 END;
/* LOCATION OF LAST CLUD MATCH. NEXT SEARCH
/* BEGINS HERE.
95 1 3 IF SUBSTR(KWWORD,1,LENGTH(PTABLE(L).CLUDWD(N))) =
PTABLE(L).CLUDWD(N)
96 1 3 THEN DO:
97 1 3 P = M;

98 1 3 IF KEY(L) = '0' THEN DO;
100 1 3 REJECT = KWWORD;
101 1 3 GO TO A;
102 1 3 END;
/* WHEN MATCH WITH CLUD WORD IS FOUND, PROGRAM
/* CHECKS KEY TO DETERMINE WHETHER THE LIST IS
/* INCLUSION OR EXCLUSION LIST.

```

PREFIX: PROCEDURE OPTIONS (MAIN):

```

STMT LEVEL NEST
103 1 3 DO:
104 1 3 Y = L;
/* PRINT IS SUBPROCEDURE THAT PRINTS WORD WITH */
/* PREFIX REMOVED. */

105 1 3 CALL PRINT;
106 1 3 N=L;
107 1 3 GO TO A;
108 1 3 END;

109 1 3 END;
110 1 3 END;

111 1 2 DO:
112 1 2 IF SUBSTR(WORD,1,LENGTH(PREFIX(L + 1))) >= PREFIX(L + 1) THEN GO TO
    BUNPL;
114 1 2 ELSE GO TO PRMLOOP;
115 1 2 END;
116 1 2 BUNPL: P = 1;
117 1 2 END;

/* ***** */
/* SINCE SOME PREFIXES OVERLAP WITH PREFIXES */
/* THAT FOLLOW, IT IS POSSIBLE FOR WORDS THAT */
/* APPEAR AFTER THE SECOND PREFIX MAY ACTUALLY */
/* HAVE THE FIRST PREFIX. HOPEFULLY AN EXAMPLE */
/* WILL HELP. */
/* ATYPICAL HAS A LEGITIMATE A-PREFIX; HOWEVER */
/* IT WOULD COME AFTER ALL WORDS WITH AD/PREFIX- */
/* S. IN ORDER NOT TO LOSE THESE WORDS, A LIST */
/* OF ALL SUCH PREFIXES THAT OVERLAP THE */
/* FOLLOWING PREFIX IS MADE. WHEN NO MATCH IS */
/* FOUND WITH THE CURRENT PREFIX IN THE NORMAL */
/* PROCEDURE, THE PROGRAM JUMPS DOWN TO THIS */
/* LOOP AND TESTS WORDS AGAINST THESE */
/* OVERLAPPING PREFIXES. */

118 1 1 PRMLOOP: DO:
/* CHECKS EACH PREFIX AGAINST WORD */
119 1 1 DO K = 1 TO 10 WHILE (PREFIX(X) = J);
120 1 2 IF SUBSTR(WORD,1,LENGTH(PREFIX(X))) = PREFIX(PREFIX(X)) = 1
121 1 2 SUBSTR(WORD,1,LENGTH(PREFIX(X))) > PREFIX(X) THEN GO TO ENDX;

122 1 2 /* IF MATCH IS FOUND, CHECKS CLUD LIST */
123 1 3 IF SUBSTR(WORD,1,LENGTH(PREFIX(X)).CLUDWD(J) <= PREFIX(X);
124 1 3 TABLE(PREFIX(X)).CLUDWD(J) THEN GO TO ENDX;
125 1 3 IF SUBSTR(WORD,1,LENGTH(PREFIX(X)).CLUDWD(J)) =
126 1 3 TABLE(PREFIX(X)).CLUDWD(J) THEN DO:

```


PREFIX: PROCEDURE OPTIONS (MAIN);

```

127 1 3 IF KEY(PERFIX(X)) = 'J' THEN DO: /* CHECKS IN/EX-CLUSION LIST */
129 1 3 REJECT = #WORD;
130 1 3 GO TO A;
131 1 3 END;
132 1 3 ELSE DO:
133 1 3 Y = PERFIX(X);
134 1 3 IF N < PERFIX(X) THEN N = PERFIX(X); /*START SUCCESSIVE SCANS HERE*/
136 1 3 CALL PRINT;
137 1 3 GO TO A;
138 1 3 END;
139 1 3 END;
140 1 3 ENDJ: IF PTABLE(PERFIX(X)).CLUDWD(J) = PERMCLD(X)
141 1 3 THEN DO:
142 1 3 GO TO ENDX;
143 1 3 END;
144 1 3 END;
145 1 2 ENDX: END;

146 1 1 IF L > LOP THEN L = TOP;
148 1 1 DO L2 = FIRST TO L;
149 1 2 B: IF KEY(L2) = 'J'
150 1 2 THEN IF SUBSTR(#WORD,1,LENGTH(PREFIX(L2))) = PREFIX(L2)
151 1 2 THEN DO:
152 1 2 Y = L2;
153 1 2 IF N < L2 THEN N = L2;
155 1 2 CALL PRINT;
156 1 2 GO TO A;
157 1 2 END;
158 1 2 END;

159 1 1 DO X = 1 TO 10 WHILE(PERFIX(X) ^= 0);
160 1 2 IF KEY(PERFIX(X)) = 'J'
161 1 2 THEN IF SUBSTR(#WORD,1,LENGTH(PREFIX(PERFIX(X)))) = PREFIX(PERFIX(X));
162 1 2 THEN DO:
163 1 2 Y = PERFIX(X);
164 1 2 IF N < PERFIX(X) THEN N = PERFIX(X);
165 1 2 CALL PRINT;
167 1 2 GO TO A;
168 1 2 END;
169 1 2 END;

```

PREFIX: PROCEDURE OPTIONS (MAIN):

STMT LEVEL NEST

170 1 1 REJECT = KWWORD;
 171 1 1 GO TO A;
 172 1 1 END;

173 1 1 BUMP: 3ND;

174 1 PRINT: PROCEDURE;
 175 2 IF SUBSTR(KWWORD,LENGTH(PREFIX(Y)) + 1,1) = 1 THEN DO;
 177 2 DUPFIX = SUBSTR(KWWORD,1,LENGTH(PREFIX(Y)) + 1);
 178 2 DUPWRD = SUBSTR(KWWORD,LENGTH(PREFIX(Y)) + 2);
 179 2 GO TO PRNT;
 180 2 END;

181 2 DUPFIX = SUBSTR(KWWORD,1,LENGTH(PREFIX(Y)));
 182 2 DUPWRD = SUBSTR(KWWORD,((LENGTH(PREFIX(Y)) + 1));
 183 2 PRNT;

184 2 JUNK1 = JUNK1 - LENGTH(PREFIX(Y));
 185 2 PUT FILE(ADD) EDIT(JUNK1, JUNK2, DUPWRD) (F(2), A(16), A(18));
 187 2 IF KWWORD = LSTWORD THEN GO TO SKIP2;
 188 2 PUT EDIT(DUPFIX, DUPWRD)(SKIP(0), COL(23), A, COL(30), A);
 189 2 LSTWORD = KWWORD; /* SUCCEEDING WORDS TESTED TO AVOID RECOMPUTING
 190 2 COUNTS = COUNTS + 1; /* COUNT OF WORDS WITH PREFIXES KEPT
 END PRINT;

191 1 PREFIX: PROCEDURE;
 192 2 ON ENDFILE(PFIX) GO TO PTFSTF;
 194 2 DCL PREFIX1 CHAR(31);
 195 2 DCL PCLJD CHAR(18) INITIAL(' ');

196 2 DO J = 1 TO 35;
 197 2 PREFIX(J) = ' ';
 198 2 CLUSDAD(J,*) = ' ';
 199 2 END;
 200 2 TOP = 0;

/* PRINT IS THE SUBPROCEDURE THAT CREATES A DUP-
 /* LOCATE RECORD AND PRINTS THE RECORD. */

/* LENGTH OF THE WORD IS RECOMPUTED FOR SUPFIX
 /*

/* PREFIX IS THE SUBPROCEDURE THAT BUILDS THE
 /* PREFIX TABLES. IT ALSO DETERMINES WHEN A
 /* PREFIX IS 'CONTAINED' IN THE SUCCEEDING PREFIX
 /* SO THAT THESE PREFIXES MAY BE USED IN PERFIX. */

/* CLEARS PREFIX AND CLUD LISTS. */

254

JUNK2: PREFIX(Y), JUNK2, PREFIX(Y)
 A(25), A(8), F(2), A(18)

PREFIX: PROCEDURE OPTIONS (MAIN):

```

STAT LEVEL NEST
201 2 PREFIX = 0;
202 2 PERMCLD = ' ';
203 2 DO J = 1 TO 35;

/* PROGRAM SAVES THE LAST PREFIX WITH CLUD LISTS */
/* THAT WAS READ IN BUT FOUND TO COME IN LATER */
/* ALPHABETICAL SEQUENCE THAN THE CURRENT */
/* PROCESSING LETTER. THIS BECOMES PREFIX (1) */
/* OF THE CURRENT PREFIXES. */
204 2 1 IF TEMP.PREMPFIX ^= ' ' THEN DO;
205 2 1 PTABLE(1) = TEMP;
206 2 1 TEMP.PREMPFIX = ' ';
207 2 1 TEMP.TEMPCLD = ' ';
208 2 1 GO TO TEST;
209 2 1 END;
210 2 1 END;

211 2 1 GET FILE(PFIX) EDIT(PREFIX1, KEY(J), CH) (A(3), X(3), P(1), X(67),
A(1));

/* PREFIX IS STORED IN VARYING CHAR. SLOT SO THAT */
/* THE LENGTH OF THE PREFIX WILL BE AVAILABLE FOR */
/* SUBSTRING PARAMETER WHEN CHECKING WORD FOR */
/* MATCH. */
212 2 1 DO II = 8 TO 1 BY -1;
213 2 2 IF SUBSTR(PREFIX1,II,1) ^= ' ' THEN GO TO OUTI;
214 2 2 END;
215 2 1 OUTI: PREFIX(J) = SUBSTR(PREFIX1,1,II);
216 2 1 END;

217 2 1 DO K = 1 TO 300;
218 2 2 GET FILE(PFIX) EDIT(WLUD, CH) (X(2), A(18), X(59), A(1));
219 2 2 DO III = 18 TO 1 BY -1;
220 2 3 IF SUBSTR(WLUD,III,1) ^= ' ' THEN GO TO OUTCLD;
221 2 3 END;
222 2 2 OUTCLD: PTABLE(J).CLUDND(K) = SUBSTR(WLUD,1,III);
223 2 2 END;

224 2 2 IF PTABLE(J).CLUDND(K) = 'END CLUD'
225 2 2 THEN DO;
226 2 2 PTABLE(J).CLUDND(K) = ' ';
227 2 2 GO TO TEST;
228 2 2 END;

229 2 2 IF SUBSTR(PTABLE(J).CLUDND(K),1,1) = '...'
230 2 2 THEN DO;
231 2 2 PTABLE(J).CLUDND(K) = SUBSTR(PTABLE(J).CLUDND(K),2,(III-2));
232 2 2 GO TO TEST;
233 2 2 END;

234 2 2 ENDS: END;

/* STORES PREFIX AND CLUD LIST FOR NEXT CONST. OF */
/* PTABLES. */

```

```

PREFIX: PROCEDURE OPTIONS(MAIN);
STMT LEVEL NEST
235 2 1 TEST: IF SUBSTR(PREFIX(J),1,1) ^= LEFT(L)
236 2 1 THEN DO;
237 2 1 TEMP = TABLE(J);
238 2 1 GO TO PTESTF;
239 2 1 END;
240 2 1 TOP = TOP + 1;
241 2 1 END;

242 2 PTESTF: X = 1;
/* TESTS TO SEE IF PREFIX IS 'CONTAINED IN' SJC-
/* CREATING PREFIX.
243 2 DO J = 1 TO TOP;
244 2 1 IF SUBSTR(PREFIX(J),1,LENGTH(PREFIX(J))) = SUBSTR(PREFIX(J + 1),1,
245 2 1 LENGTH(PREFIX(J))) THEN DO;
246 2 1 PREFIX(X) = J;
247 2 1 DO L1 = 1 TO 300;
248 2 1 IF TABLE(J).CLUDWD(L1) ^= ' ' THEN GO TO LIEND;
249 2 1 ELSE DO;
250 2 2 PERMCLD(X) = TABLE(J).CLUDWD(L1 - 1);
251 2 2 X = X + 1;
252 2 2 GO TO JEND;
253 2 2 END;
254 2 2 END;
255 2 2 LIEND: END;
256 2 1 END;
257 2 1 JEND: END;

258 2 ENDFICH: END PREFIX;

/* COMPUTES THE TOTAL NUMBER OF WORDS WITH PREFIXES*/
/* AND THEIR PROPORTION IN THE TEXT.
259 1 OUT: PUT EDIT('TOTAL WORDS',COUNT) (SKIP(2),A, COL(30),F(6),);
260 1 PUT EDIT('TOTAL WORDS WITH PREFIXES',COUNTS) (SKIP,A, COL(30),F(5));
261 1 PUT EDIT('PERCENT WITH PREFIXES',100*COUNTS/COUNT)
(SKIP,A, COL(32),F(5.2));

262 1 END PREFIX;

```


PREFIX	KEY	CLUDLIST	ALTITUDE	END CLUD	ANTHROPO	ANTHROPOGEN	ANTHROPOGENOGRAPH	END CLUD	ANTHROPOGEN	ANTHROPOGENOGRAPH	END CLUD
AMBI	1	ALTIMERRY	ALTITUDE	END CLUD	AMBI	AMBI	AMBI	AMBI	AMBI	AMBI	AMBI
AMPHI	1	AMBIEXT	AMBILATERAL	END CLUD	AMPHI	AMPHI	AMPHI	AMPHI	AMPHI	AMPHI	AMPHI
AN	1	AMPHITHEATER	TWO, BOTH, ON BOTH SIDES	END CLUD	AN	AN	AN	AN	AN	AN	AN
ANDRO	1	ANALPHABETIC	NCT, WITHOUT, LACKING, VAR. OF 'AD', VAR. OF 'ANA'_UP, --	END CLUD	ANDRO	ANDRO	ANDRO	ANDRO	ANDRO	ANDRO	ANDRO
ANEMO	1	ANDROCENTRIC	ANDROPHOBIA	END CLUD	ANEMO	ANEMO	ANEMO	ANEMO	ANEMO	ANEMO	ANEMO
ANGLO	0	ANEMOGRAM	WIND	END CLUD	ANGLO	ANGLO	ANGLO	ANGLO	ANGLO	ANGLO	ANGLO
ANTI	1	ANGLOPHIL	ENGLISH	END CLUD	ANTI	ANTI	ANTI	ANTI	ANTI	ANTI	ANTI
ANTE	1	ANTACID	VAR OF 'ANTI' AGAINST	END CLUD	ANTE	ANTE	ANTE	ANTE	ANTE	ANTE	ANTE
ANTHROPO	1	ANTE-CHRISTIAN	BEFORE	END CLUD	ANTHROPO	ANTHROPO	ANTHROPO	ANTHROPO	ANTHROPO	ANTHROPO	ANTHROPO
ANTI	0	ANTE-WAR	ANTE-DAWN	END CLUD	ANTI	ANTI	ANTI	ANTI	ANTI	ANTI	ANTI
AP	1	ANTHERIDIAN	ANTE-HAMBER	END CLUD	AP	AP	AP	AP	AP	AP	AP
AR	1	ANTHRACID	ANTE-HARRIAGE	END CLUD	AR	AR	AR	AR	AR	AR	AR
ARCH	1	ANTHRACENTRI	ANTE-SPRING	END CLUD	ARCH	ARCH	ARCH	ARCH	ARCH	ARCH	ARCH
ARCHE	1	ANTHRACENTRI	ANTE-SPRING	END CLUD	ARCHE	ARCHE	ARCHE	ARCHE	ARCHE	ARCHE	ARCHE
ARCHI	1	ANTHRACENTRI	ANTE-SPRING	END CLUD	ARCHI	ARCHI	ARCHI	ARCHI	ARCHI	ARCHI	ARCHI
AT	1	ANTHRACENTRI	ANTE-SPRING	END CLUD	AT	AT	AT	AT	AT	AT	AT

PREFIX	KEY	CLUDLIST	ATTRIBUT	ATTUN	END CLUD
ATMO	1	AIR END CLUD			
AUDIO	0	AUDITORY AUDIOGEN AUDIOLOG (Y) END CLUD		AUDION	AUDIOPHLE
AUTO	0	(SELF, SAME) AUTOWAL AUTOCLAVE AUTOCHECHON	AUTOMETRY AUTOCGRAPH	AUTONM AUTOPSY	AUTONYM END CLUD
BACK	0	'BACK ' BACKSTAB	BACKING BACKWARD	BACKLOG END CLUD	BACKSLIDE
BE	1	BE-MITHEWARD BEBOOTED BEDAZZL BEDAZZL BEDAZZL BEFLOWER BEGEM BEGEM BEHSUIT BEMEAN BENIGHT BERIBSONED BESLOBBER BESTRID BEWAIL BEWRITE	COVER, TO MAKE, TO DUB, PROVIDED WITH NO MEANING BECHARM BEDASH BEDREW BEDRIVEL BEFOOL BEGLAR BEKNAVE BEMIRE BEMEDALLED BEREM BESPAN3L BESTRID BEWEP END CLUD	BECLUD BEDAUB BEDIANJND3J BEFLA3 BEFCUL BEGRIN BELATE BEMOAN BERHYH BESCRIBBL BESPRAD BETHOU3IF BEWITTH	BECRAVL BEDAVID BEDIM BEFLA BEFRIEND BEHAD BELITTL BEMOCK BERIBANED BESEIG BESRADLL BETHOU3IF BEWITTH
BI	0	TWO, TWICE, VAR. OF 'BIO' BIAS BICIPITAL BIDDY BIFID BILE BILLION BINARY BIO BIRTH 'BIT ' BITTER	BIBLE BIDDABL BIENNIAL BIGAM BIK BIMESTER BIND BIRCH BISCUIT BITE BIZARRE	BIBLIO BIDDEN BIER BIGOT BILL 'BIN ' BINOCLE BIRD BISECT BITING END CLUD	BIBULOUS BIDDING BIFARIOUS BIJOU BILLET BINAL BINOCULAR BIRR BISHOP BITTEN
BIBLIO	1	BOOK, BIBLE BIBLIOPHY	BIBLIOMANI		
BIN	1	TWO, TWO AT A TIME BINAURAL	END CLUD		
BOOK	0	'BOOK ' BOOKIE, BOOKING, BOOKISH BOOKLET	'BOOKS '		END CLUD
BY	0	BY-BLJW BYWORD END CLUD	BYRNE		BYTE

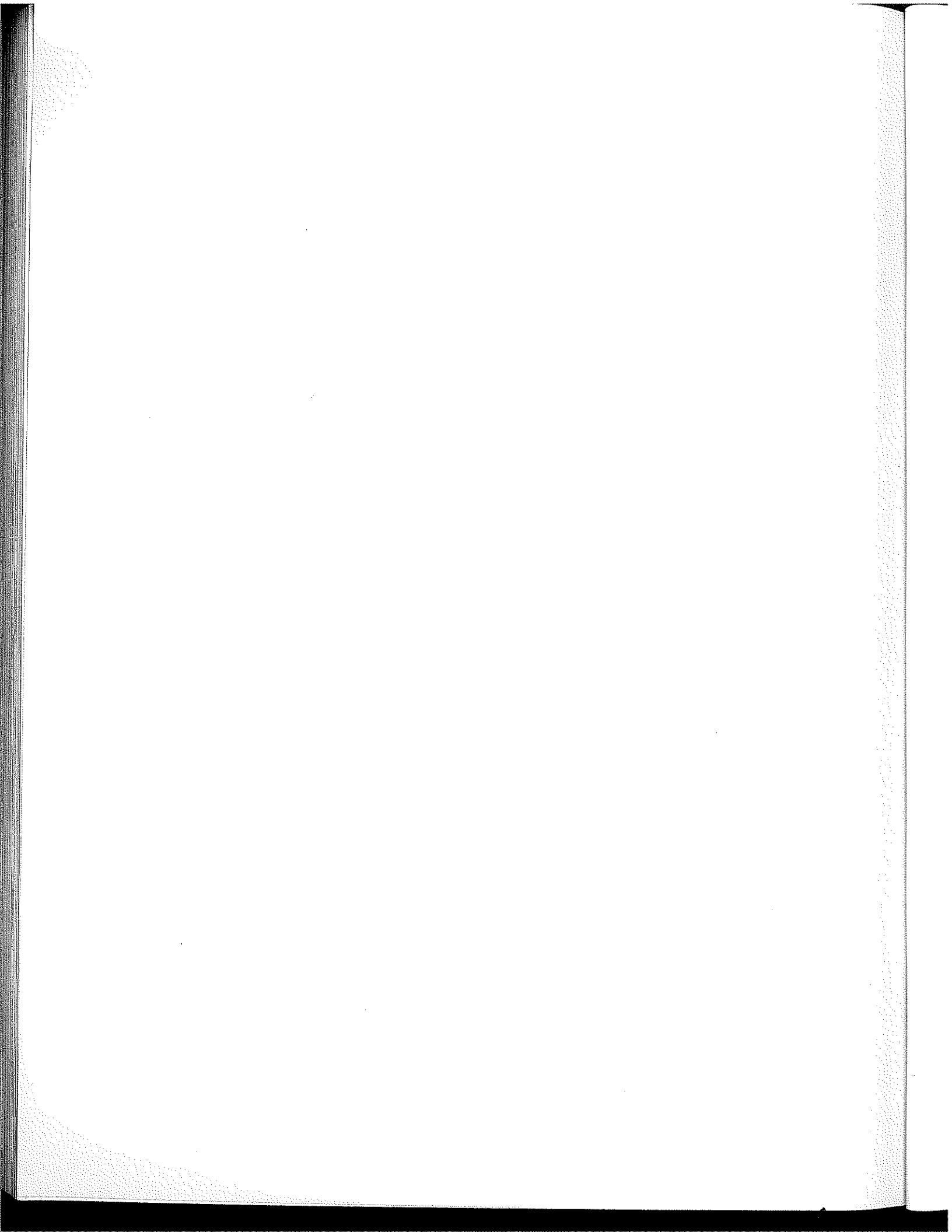
PREFIX	KEY	CLUDLIST					
BYE	1	BYELAW	VAR.OF 'BY'	END CLUD			
CENTRI	0	CENTRIC	(CENTER) CENTRIFUGAL	CENTRIOLE	END CLUD		
CHRONO	1	CHRONOGRAPH	TIME CHRONOMET	CHRONOSCOPE	END CLUD		
CIS	1	CISATLANTIC	(NEAR SIDE OF) CISLUNAR	END CLUD			
CO	1	COORDIN COORDJUT COORDUCAT COORDIN END CLUD	VAR OF COM, IN ASSOC WITH CO_STAR COAXIAL COEQUAL COINCIDEN COPARTNER END CLUD	CO WORKER CODEPEND COEY COMATE COTENAFY			
COL	1	COLLABORA END CLUD	VAR OF COM, WITH COLLAPS	COLLATERAL	COLLEASUE		COLLOCAT
COM	1	COMFASUR COMPACT COMPROMIS	WITH, TOGETHER, IN ASSOC COMMENDE COMPATERN END CLUD	COMMINGLS COMPATRIOT	COMMUTA COMPER		COMMUNUAL COMPOSSIBLE
CON	1	CONCAY CONDFSCOM CONSERVAL CONSEQUEN CONORTION	VAR OF COM CONCENTRIC CONFEDERA CONSERVAL CONSOLIDAT 'CONTRACT '	CONCORPORAT CONFIGUR CONJOIN CONSTRAIN END CLUD	CONCOURS CONFOR CONJUNCTUM CONSTRIC		CONDENS CONFRONT CONJUNCTJ CONTEMPOR
COUNTER	0	COUNTERCHANGE END CLUD	(OPPOSITE) COUNTERPEIT	CONJUTFRAN	COUNTERPART		COUNTERWORD
DE	1	DE_EMPHASI DE_CAMP DESERIF DECON DEFAC DEFOILLAT DESENTRA DEHORN DELEGALIZ DELOCAL DEMOYAN DEMORALI DENUMERA DEPEOPLE DEPPL	SEPARAT, PRIVATION, REMOV, DESCENT, REVERSAL DEBART 'DECANI ' DECLASS DEGRESC DEFANG DEFORM DEGLACIA DEHUM DELEGATION DELOCUS DEMERIF DEMOYNT DEODOR DEPERSON DEPOP	DEBRIEF DECENTER DECOLONI DECURY DEFEND DEPROSI DEGRAD DEICE DELINCAT DEMARK DEMORILL DENOMINAT 'DEPART ' DEPICTUR DEPORT	DEBRIEF DECENTER DECOLONI DECURY DEFEND DEPROSI DEGRAD DEICE DELINCAT DEMARK DEMORILL DENOMINAT 'DEPART ' DEPICTUR DEPORT		DEBAG DECENTR DECOM DEDUCT DEFLOWER DEJUM DELANINA DELIST DEMATERIALI DEMODULAT DENOT 'DEPARTING DEPLOY
DECA	1	TEN					

PREFIX	KEY	CLU DLIST	DECALITER	DECAMETER	END CLJD
DECI	1	DECAGRAM	TENTH DECILITER	DECIMETER	END CLUD
DEMI	1	DEMI BLOND	(HALP)	END CLUD	
DI	1	DIATOMIC	TWO, DOUBLE DICHROM	DITHEIS	END CLUD
DIS	0	DIABUS DISCER DISCREPAN DISDAIN 'DISHES' DISHWA DISPARA DISPIRIT DISPOSUR DISSECTED DISSIPAT DI STAFF DISFI DISTRICT END CLUD	APART, AWAY, UTTERLY, PNR DISAFFECT DISCI DISCREP DISERS 'DISHES' DISK DISPATCH DISPLAY DISREAL DISSEMINAT DISSOLUT DISTAIN DISTORT DISTURB END CLUD	DISBURS DISCORD 'DISCUS' 'DISH' DISHCOVER DISH TOWEL DISMISS DISPER DISPEN DISPOSE DISRUPT DISSERT DISSIDEN DISSONAN DISTAN DISTRAGHT DISURSIVE	'DISC' DISCREET DISCUS DISHCOVER DISH TOWEL DISMISS DISPER DISPEN DISPOSE 'DISSERT' DISSIDEN DISSONAN DISTAN DISTRAGHT DISURSIVE
DOWN	0	'DOWN' DOWNTIME	(DOWN) DOWN-TO-EARTH DOWNTOWN	DOWNPAYMENT DOWNWARD	DOWNSTAGE END CLUD
E	1	'EDUCE' ELOPE 'EMERGENT' ERUPT EVOK	VAR OP 'EX' UTTERLY, ETC. EDUCT ELUCIDAT 'EMERGING' EMERA EVAL END CLUD	ELAP 'EMERGE' ENUMERA EVISCEAT	ELOCUTION EMERGED ERADIAT EVOC
EM	1	EMBRIN EMBLAZ EMBOWEL EM PANEL END CLUD	ENCLCS, PUF INFO OR ON, GIVE THE QUALITY, AG EMANK EMBOJ 'EMBRACE' EMPLAC	EMBED EMBOJ EMBOLDER EMPOISON	EMBITTER EMBOW EMPOWER
EN	1	ENABL ENCAPSUL ENCOO ENDANGER ENFOLO ENGIRD ENJOIN ENLIGHT ENREGISTER ENSHROUD ENTHRON	IN, OR VB FORM. OR TRANSITIVE ENACT ENCAS ENCOMPASS ENDEAR ENFORC ENJORY ENJOY ENLIST ENRICH ENSLAV ENTAIL	ENAC ENCLASP ENCRUST ENFAC ENGAG ENGRAB ENLIAC ENMESH ENRAPT ENRAMP ENSUR ENTRAIN	ENCAMP ENCLOS ENCIPT ENFEREL ENGENDER ENHARTEN ENLARS ENRAPT ENSHRIN ENTRANL ENTRANC

PREFIX	KEY	CLUDLISE	ENTRENCH ENWRAP	ENTRUST ENWEATH	ENTWI END CLUD	ENVISAS
EPI	C	ENTRAP ENVISION 'EPIC ' EPILEP EPISTLE END CLUD	(AT,BEFORE,AFTER) EPICTURE EPILOS EPIPHAP	EPIDERM EPIPHENOMEN EPIPHELI	EPIGENE EPISCO EPIITHEI	EPIGRA EPIISO EPIIOME
ERE	1	ERELONG	(BEFORE,-ARCHAIC-) ERENJM	EREWILE	END CLUD	
EX	1	EX EXFORT	EX EXCENTRIC END CLUD	EXCHANG	EXCURRENT	EXCJRS
EXTRA	C	EXTRACT EXTRAVASAN	OUTSIDE, ADDITIONAL, MORE THAN USUAL, SUPERI EXTRAD EXTRAVER	EXTRAMURAL END CLUD	EXTRANEJUS	EXTRAPOLAT
PARM	0	PARME	(PARM)	END CLUD		
FAT	1	FAT-FACED	(FAT) FATFREE	FATHEAD	END CLUD	
FOR	1	FORBAD (?) 'FORBAR' (?) 'FORBED' (?) FORBOR (?),E,NE FORFEND (ARCHAIC) FORGIV (?) FORSAKE (?) FORSOO (?),K,IH FORSPENT (?),ARCHAIC) FORSHAR FORSWOR (E)	AWAY, OFF, EXTREMELY, WRONGLY, NEGATIV OR PRIVATIV FOR			
FORE	0	FORE AND FOREJIDING (?) FOREJID FOREJIN FORESTER	'FOREDO ' FORENSIC FORESTRY	FOREDOING 'FOREST ' FOREVER	FORMENT	END CLUD
GEO	1	GEOCENTRIC	THE EARTH GEOGRAPHIC	GEOPHYSIC		
GOAL	1	GOALK	(GOAL) GOALEND	END CLUD		
GUIDE	0	GUIDEJ	(GUIDE) END CLUD			
HAIR	0	'HAIR ' HAIRY	(HAIR) HAIRBRA END CLUD	HAIRDO	HAIRLES	HAIRSPILT
FORECAST				'FORECAST ' FOREDOCK 'FORESED ' END CLUD	'FORECASTS ' 'FOREGO ' FORESTATION	

HAIR * HAIRDO HAIRLESS HAIRSPILT
 HAIRERA HAIRY HAIRDO HAIRLESS HAIRSPILT
 END_CLOUD END_CLOUD

PRRFLX	KEY	CLUDDLST					
HALF	1	HALF AND HALF END CLOUD	HALF HALF_BLOOD	HALF_HEARTED	HALF_FRACK	HALFWAY	
HEMI	1	HEMISPHER	HALF END CLOUD				
HETERO	1	HETEROCHROM	DIFFERENT, OTHER HETEROSEX	END CLOUD			
HEXA	1	HEXAMETER	SIX HEXANGULAR	HEXASYLLABL	END CLJD		
HIND	1	HINDQUARTER	REAR, PART HINDSIGHT	END CLOUD			
HOMO	1	HOMOCENTRIC	SAME HOMOCERO	HOMOSEX	END CLJD		
HUMAN	1	HUMAN-INTER END CLOUD	(HUMAN) HUMANHEARI	HUMANHOOD	HUMANKIND	HUMANMIND	
HYDRO	0	HYDROCHLORIC HYDROPHOB	WATER, HYDROGEN HYDROGEN HYDROUS	HYDROGRAPHY END CLOUD	HYDROLOGY	HYDRJLY	
HYLO	1	HYLOTHEIS	WOOD, MATTER END CLOUD				
HYPER	0	HYPERBO	OVER, (SEM.DIF.EXCESS) HYPERURBAN	END CLOUD			
HYPNO	1	HYPOANALY (SIS) HYPNOS (EMESIS, RAPH) END CLOUD	SLEEP, HYPNOSIS				HYPNOTHERAP
HYPO	0	HYPOTHESIS	UNDER, LESS, LOW HYPOHETC	END CLOUD			
ICONO	1	ICONOGRAPH	IMAGE, LIKENESS END CLOUD				
IL	1	ILLAUDAB ILLEGAL ILLEGI (ELE, TIMEATE) ILLICIT ILLITERA (PA) ILLUMIN (A TEST)	VAR. OF IN, NOT, VB, FORM . ILLEGAL ILLIMIT				ILLIBERAL ILLOGIC
IM	1	IMBALANC IMBO (DY, SOM, #ER?,) IMBRES IMMOBIL IMMOD (GRATE, PST)	VAR. OF IN, NOT, ETC. IMBALM IMBRANSL IMMIGRA	IMBARK	IMBITTER	IMMERS IMMIX	



PREFIX	KEY	CLUDELIST	INFORMANT	INFORMANTS	INFORMANT	INFORMANTS	INFORMANT	INFORMANTS
		INDOCH						
		INDOL (E, RNT)						
		INDU (RRE, STRY)						
		INFEBRI (ELY, AFE)						
		INFIRMI (Y)						
		INFIRMI (TY)						
		INFLAP						
		'INFORMAT' (ION, ORY, LIVE)						
		INFRA						
		INHABIT						
		INHE (RIT, SION)						
		INI (NICAL, MITABLE, TIAL)						
		INK						
		INOC (ULAPE)						
		INQUIR						
		INSBRE						
		INSIST						
		INSOLA (TE,)						
		INSOLE (NI, E)						
		INSPECT (OR, ORATE)						
		INSPIR (IT, RE,)						
		INSTAL (L, LATION)						
		INSTAN (CE, Y, E)						
		INSTRU (CT, MENTAL)						
		INSURA (NCE)						
		INSURG (ENT)						
		INSURRECT						
		INTEG (ER, RAL, RATE)						
		INTELL (ECT, ISENCE)						
		INTEN (D, DANT, SE, T)						
		INTER (SHOULD WORK EXCEPT 'MINABLES')						
		INTERSTIN						
		INTIN (ATE)						
		'INTO'						
		INTOXICA (NT, ATE)						
		INTRA (INTRA SHOULD BE RUN-DEBUGGED)						
		INTRAM (VAR,)						
		INTRAS (PINAL, RATE)						
		INTRAV (VAR,)						
		INTREI (CATE, SANT, SUE, NSIC)						
		INTRO (VAR,)						
		INUNDA (TE)						
		INUR (E, N)						
		INVAS (ION)						
		INVEI (GH, GLE)						
		INVERSE						
		INWET (ERATE)						
		INVID (IOUS)						
		INVIS (ILATE, ORATE)						
		INUIT						
		END CLUF						
INFR	0	INFRACT						
		BELOW						
		INFRANJIB						
		END CLUD						
INTER	0	INTERCED						
		AMONG, BETWEEN, MUTUALLY, DURING, ETC.						
		INTERCEPT						

PREFIX	KEY	CLUDBLIST	INTERFER	INTERFERM	INTERFERM	INTERFERM
IR	0	INTERJESS (ION) INTERDICT INTERJACEN (CE,T) INTERMENT *INTERM INTERNIS INTERPEL (LANT) INTERPOLAT INTER (OGATE,UPT) INTERSPERS INTERSTIC (E) INTERSTIT (IAL) *INTERVAL END CLUD	INTERFER INTERJECT INTERMISS INTERNE	*INTERIM INTERLUSE INTERMIF INTERMRE	*INTERCOM INTERIOR INTERMEDAT INTERMURAL INTERMING *INTERPOL INTERSECT	INTERVOLV
IR	0	IN,NOF,VB,FORNATIVE & TRANSITIVE IRE IRRIGA IRRIGUOUS (ARCHAIC,WELL_WATRED)	*IRISH	IRO IRITA	IRREMEAB	
KEY	1	KEY (LOCK),CENTRAL IMPORTANCE KEYHO KEYSTONE KEYVAN KEYSTROK	KEYNOE KEYWORD	KEYPUNC END CLUD	KEYSHI	
LITHO	1	LITHOIRAPH LITHOPRINT STONE	LITHOSPHR	END CLUD		
MACRO	0	LARGE, LONG, EXCESSIVE, NO WORDS-SOME ARE PARE				
MAL	1	BAD, WRONG, ILL, FR. MALADAPT MALADJUST MALP (EASANCE, ORN, UNCTION)	MALADMINIS END CLUD	MALAPPORTION	MALCONTENT	
META	0	AFTER, AWAY, BEYOND, BEHIND METAL END CLUD	METAMR	METAMOR	METAPHOR	
MICRO	0	SMALL, ENLARGING SOMETHING SMALL END CLUD				
MID	1	MIDDLE, BETWEEN MIDA (IR,FTERNOON) MIDB (RAIN, AND) MIDC (CURSE) MIDL (AND, E3, I NE) MIDMO (SP, ON) MIDN (IGHT, JOY) MIDSEPIION MIDT (OWN, ERR) MIDW (ATCH, AY, ESK, EST, IFE, INTER)	*MIDSHIPS	MIDPOINT MIDSTREAM	MIDRASH MIDSUMMER END CLUD	
MIS	0	ILL, MISTAKEN, WRONG MISCE (SENATION, LLANY) MISCHA (NCE, NIER) MISCHIE (F) MISSIV MISO (=HATE, PREFIX)	MISCREEN MISHRASH	*MISE MISOMER	MISER	

PREFIX	KEY	CLUDELIST	MISLE	MISSEY	MISSTAK	MISTY
MON	1	MISPRLS (E, ION) MISSEY (LE, ON) MISTAK MISTY MISTRSS END CLUD	MISLE	MISSEY MISFOOK	'MISS ' 'MIST ' MISTRSS	'MISSED ' MISTAK MISTY
MONO	0	MONRURAL ALONE, SINGLE, ONE, VAR. OF 'MONO' END CLUD				END CLUD
MULTI	0	MONOLITH (IC) MONOPOL (Y) MANY MULTIFARIOUS MULTIPAR (A, OUS) MULTIPLI MULTIFID MULTIPLY MULTITUD				MULTIPLE
NEC	0	NEOLITH NEW, RECENT NEOLOG				END CLUD
NO	1	NO NOBODY (?) END CLUD				NOWAY
NON	0	'NONAGE ' NONCHALAN (F, CE) 'NONDESCRIPT ' NONSUCH NOT, 'LACKING_', NOT NECESSAIIY 'REVERSE ' 'NONCE ' 'NONE ' END CLUD				NONPARALL
OB	1	OBLIGAT TOWARD, ON, OVER, AGAINST OBLONG				OBVER
OFF	1	OFF OFFS (COURING, GREEN, SET) OFFBEAT				OFFHAND END CLUD
OUT	0	OUT OF OUTRAS (E, ROUS) END CLUD				OUTFIT OUTSIDE
OVER	0	'OVER ' OVER A LIMIT OVERLAP				'OVERER ' OVERSEER
PAN	1	PANSOPHISM ALL, GENERAL PANTHEISM				PANTROPIC END CLUD
PARA	1	PARABOMB PARAMILT PARATROOP PARACHUT, GUARD AGAINST, BESID, NEAR, AMISS, +IMP. ALTER PARAGLIDER PARAPHRAS END CLUD				'PARAMEDIC ' PARASCU PARAMAGNET PARASOL
PAY	1	PAYDAY TO PAY ETC. PAYLOAD				PAYMASTER END CLUD
PER	0	PER THROUGH, UTTERLY, VERY, THOROUGHLY				

PERFIX	KEY	CLUDLIST	PERMUTAT	'PERSON '	'PERFECT '	PERPERVID
		PERAMBULAT				
		PERDUR (E, ABLE)				
		'PERHAPS '				
		PERSUA (DE, SION)				
		END CLUD				
PERI	1	PERISCOPE	ABOUT, AROUND, BEYOND			
			END CLUD			
POLY	1	POLYANG (ULAR)	MULTIPLE, MUCH, MANY	POLYCHROM	POLYGENIC	POLYGENIC
		POLYGRAPH	POLYPHON	POLYSYLLAB	POLYTECHNIC	POLYTECHNIC
		POLYTON	POLYIYP	END CLUD		
POST	0	POSTAGE	BEHIND, AFTER, MAIL	POSTER	POSTIC	POSTING
		POSTPONE	POSTAL			
		POSTUM (LATE, NE)			END CLUD	
PRE	0	PRAECI (ORY)	BEFORE, PRIOR TO, EARLY, IN FRONT OF	PRECAPI		
		PRAECI (OR)	PRAEWEL			
		PRAECE (DE, PE, DUNT)				
		PRAECI (DICE, SE)				
		PRAECI (DDE)				
		PRAECOG (NITION?)	PREDECESS			PREDACIOUS
		PRAECI (T, AUNT, ABL)	'PREFAB '			PRESEMP
		'PREFEN '	PREFIX	PREFACE		
		PREFEN (CT, REFERENCE)				
		PREFMAN (T, TY)				PREHENS
		PRAEUDIC				
		PRAELA (TE, CY, TURF)				
		PRAELU (LE)				
		PRAELU (RE)				
		PRAEMON	PRAEMICE	'PREP '		PREMIUM
		PREPARED	PREPANS	PREPOSITION	PREPARA	'PREPARS '
		PREPUCE	PREPROG	PRESEBYTER	PREPOSSES	PREPOSITIONS
		PRESCRIPT	PRESSAN	PRESEANT	PRESCLIND	PRESCKIBE
		PRESS	'PRESTI '	PRESTI	PRESERV	PRESID
		PRESSUM (PTION)				
		PRESTEN (CE, D, DER, SE, SION, TIOUS)				
		PREVA (IL, RICATE)				
		PREVEN (T, IENT)				
		PREY	END CLUD			
PREPR	0	PREPRI	BEYOND, MORE THAN, BY, PAST	PRETER	PAPT	PREY
			PREPERMIT			
PRO	0	PRO SAT	FOR (A CAUSE, ETC.)			
		PROBAB (LE)				
		PROBAT (E)				
		PROBE	PROBITY			
		PROCEED (URE)	PROCESS			
		PROCEED	PROCLAIM			
		PROCLAIM (ACTION)	PROCLAIM			
		PROCLIVITY	PROCLAIM			

PREFIX	KEY	CLUMLIST		
PROCTO		PROCTO (PREFIX,ANUS,RECTUM) PROCDUR PRODIS (Y,AL,IOUS) PRODU (CE,CT) PROFF PROFICI (ENT,ENCY) PROFIL (E) PROFIT (E) PROGN (OSE,OSTIC) PROJEI PROLIF (IC,ERATE) PROLOG PROLONGAT (E,ION) PROMINEN (CE,T) PROMIS (E,SORY) PROMYX (? SP.DICT.) PROMOT PROMULG (ATE,E) PRONG PROPAG (ABLE,ANDR,AGARE) PROPH (ESI,ET,Y,ELACTIC) PROPI (NE,NQUITY,TIATORY) PROPON (E,ENT) PROPOS (E,ITION) PROPRIET PROPRIO (PREFIX) PROPULS (ION) PROSM (C,SM) PROSTRAT (?) PROSP PROSOD (EVISIC,Y,MST) PROSTHE (SIS,ICSI) PROTAG (ONIST,AGORAS) PROTEG (E) PROTEIN PROTIN PROTO (PREFIX) PROTRACT (? ,SUGGEST RUN DEBUG) PROTRU PROTU (BERANCE) PROVIC		PRODDE * PROF ' PROFOUND PROGRAM PROFAN PROFUM PROGRESS PROHIB PROLIX PROMENAD PRONUN PROPEN PRONTO PROPEL PROSCIB PROSPER PROSTITUT PROSTRAT PROJECT PROGRAM PROTEST PROTRACT (? ,SUGGEST RUN DEBUG) PROTRU END CLUD
PROCTO	1	FIRST, FORMOST, EARLIEST FORM OF MAYB SHOULD O IF SCI	PROTOPLASM	PROTOTYPE
PSEUDO		PSEUDOMORPH PSEUDONYM (ITY,OUS)	END CLUD	
QUASI		RESEMBLING, SEEMING		
RE		REACH READY * REAP ' REARWARD, AGAIN REACIA * REAL ' REALM REAPER * REAP ' * READ ' REALIS REALPOLIITIK REAPI READER REALIT REALTY * REAP ' REARMOST		

PREFIX	KEY	CLUDDLIS	REBATE	REBEL	REBUFF
		REARWARD	REBATE		
		REBOK	REBUTTAL		
		RECALCITRA (NT,TE)			
		RECANIT (?RUN DE_BUG)	RECEDE		
		RECEI (PT,VE)			
		RECESSION			
		RECEP (ACLE,IBLE,ION,IVE,OR)			
		RECEP (ACLE,IBLE,ION,IVE,OR)			
		RECIP (IENT,ROCAL)			
		RECK	RECLIN		
		RECOGNI (SE,TION,ZANCE,ZORI)			
		RECOIL (ONLY 'REHIND' MEANING SHOULD BE CHOPPED)			
		RECOLLECT (ONLY 'REGATHER' MEANING SHOULD BE CHOPPED)			
		RECOMPENSE	'RECON ' RECONCIL		'RECONDITE '
		RECONNAISSANCE	RECORD		RECOURS
		RECOVER	RECRIMINAT		
		RECT (VAR.)			
		RECUM (BENT)			
		RECUP (ERRATE)			
		'RECUR '			
		RECURS (ION)			
		RED (VAR.)			
		REDE (VAR.)			
		REDD (VAR.)			
		REDOLEN (P,CRI)			
		REDS (KIN)			
		REDUC (E)			
		REEK			
		'REEL ' (? ER,ED,)			
		REFECT			
		REFLEX			
		REFORMATORY (? REFORM?)			
		REFRAIN			
		REFRI (GERATOR,NGENT)			
		REFUT			
		REGICID			
		REGRET			
		REICH			
		'REINS ' (ARCHAIC - THE KIDNEYS)			
		REITERANT			
		REJUUVEN			
		KEYBO			
		RELAT (IVE)			
		RELAY			
		RELIAB			
		RELIE (P,VE)			
		RELIGIO (N)			
		RELUC			
		REMEDIAL			
		REMINISC			
		REMNAN (T)			
		REMONSTR (NT,CE,TE)			
		REMOVS (E IS INTENDED)			
		REMOVAL			
		RENUMERAT			
		RENOVAL			
		RENUMC			
		REPEA (L,T)			
		REPERO (IRE,RY)			
		REPETIT			
		REBEAT			
		REBEU ' REBEL			
		RECALCITRA (NT,TE)			
		RECANIT (?RUN DE_BUG)			
		RECEI (PT,VE)			
		RECESSION			
		RECEP (ACLE,IBLE,ION,IVE,OR)			
		RECEP (ACLE,IBLE,ION,IVE,OR)			
		RECIP (IENT,ROCAL)			
		RECK			
		RECOGNI (SE,TION,ZANCE,ZORI)			
		RECOIL (ONLY 'REHIND' MEANING SHOULD BE CHOPPED)			
		RECOLLECT (ONLY 'REGATHER' MEANING SHOULD BE CHOPPED)			
		RECOMPENSE			
		RECONNAISSANCE			
		RECOVER			
		RECT (VAR.)			
		RECUM (BENT)			
		RECUP (ERRATE)			
		'RECUR '			
		RECURS (ION)			
		RED (VAR.)			
		REDE (VAR.)			
		REDD (VAR.)			
		REDOLEN (P,CRI)			
		REDS (KIN)			
		REDUC (E)			
		REEK			
		'REEL ' (? ER,ED,)			
		REFECT			
		REFLEX			
		REFORMATORY (? REFORM?)			
		REFRAIN			
		REFRI (GERATOR,NGENT)			
		REFUT			
		REGICID			
		REGRET			
		REICH			
		'REINS ' (ARCHAIC - THE KIDNEYS)			
		REITERANT			
		REJUUVEN			
		KEYBO			
		RELAT (IVE)			
		RELAY			
		RELIAB			
		RELIE (P,VE)			
		RELIGIO (N)			
		RELUC			
		REMEDIAL			
		REMINISC			
		REMNAN (T)			
		REMONSTR (NT,CE,TE)			
		REMOVS (E IS INTENDED)			
		REMOVAL			
		RENUMERAT			
		RENOVAL			
		RENUMC			
		REPEA (L,T)			
		REPERO (IRE,RY)			
		REPETIT			

PREFIX	KEY	CLUDDLIST	
		SUBTRA (CT, HEND)	'SUBURB '
		'SUBURBIA '	END CLUD
		SUBVE (NE, NTION, RSION, RT)	
SUBTER	1	UNDER, BELOW	
		SUBTERNATURAL	END CLUD
SUPER	0	ABOVE, BEYOND, TO AN ESPEC. HIGH DEGREE	
		SUPER DUPER	SUPERANNUAT
		SUPERBIL	SUPERABLE
		SUPERBIO	SUPEREITE
		SUPERFLUITY	SUPERFUS
		SUPERFLUOUS	SUPERFUS
		SUPERLATIV	SUPERAL
		'SUPERPOSING '	SUPERORDINAT
		SUPERSESSION	SUPERSCRIBE
		SUPERSTIT	SUPERVENE
SUPRA	0	VAR. OF 'SUPER' EMPHASIZING POSITION	
		END CLUD	
SUR	1	VAR. OF 'SUPER', VAR. OF 'SUB'	
		SURCEAS (ARCHAIC DESIST, SUR (1)+CEASE)	SURCOAT
		SURFACE (?)	SURPASS
		SURPRINT	SURTAX
		SURREAL	SURPLUS
SYM	1	WITH, TOGETHER	END CLUD
		SYMMERIC (?)	
SYN	1	WITH, TOGETHER, IN ASSOC. (WITH)	
		SYNAESTHESIA	SYNEMESIS
		SYNECDOLOGY	
SYNCHRO	0	SYNCHRONOUS	
		END CLUD	SYNCHRONIZ
TAX	1	ORDERING, DIRECTION, TAX	
		TAXP (AID, AYER)	
		TAXP_ (AID, AYER)	
TAXI	1	TAXI (CAR), VAR. OF TAXO	
		TAXIPLANE	TAXIWAY
TETRA	0	FOUR	
		TETRAH	TETRAMER
THOROUGH	0	THOROUGH, THROUGH	
		THOROUGHFARE	END CLUD
THROUGH	1	THROUGH	
		THROUGHPUT	THROUGHWAY
THROUGH	0	ACROSS, BEYOND, THROUGH	
		THROUGHIVE	TRANSCLEN
		THROUUGH	TRANSFORMER
		THROUUGH	TRANSLITERAT
		THROUUGH	TRANSLIT
		THROUUGH	TRANSPARENT
		THROUUGH	TRANSPIR
		THROUUGH	TRANSVERS
THROUGH	1	THREE	
		THREE	TRICOLOR
		THREE (PUTPLACE)	
THROUGH	0	THROUGH, THROUGH	
		THROUGHFARE	END CLUD
THROUGH	1	THROUGH	
		THROUGHPUT	THROUGHWAY
THROUGH	0	ACROSS, BEYOND, THROUGH	
		THROUGHIVE	TRANSCLEN
		THROUUGH	TRANSFORMER
		THROUUGH	TRANSLITERAT
		THROUUGH	TRANSLIT
		THROUUGH	TRANSPARENT
		THROUUGH	TRANSPIR
		THROUUGH	TRANSVERS
THROUGH	1	THREE	
		THREE	TRICOLOR
		THREE (PUTPLACE)	

PREFIX	KEY	CLUDEST	TRIMORPH	TRILINGUAL	TRIMONTHLY
TROPO	1	TRICYCL TRIMORPH TRISYLLAB (LE) TRIBEK (LY) TURN, TURNING TROPOSPHER END CLUD	TRILINGUAL TRIPEDAL	TRIMONTHLY	
ULTRA	0	ULTRAISM BEYOND USUAL, EXCESSIVE END CLUD			
UN	0	UN UN, NOT, LACKING IN, ONE UNANIM UNCANNY ('CANNY' HAS ARCHAIC MEANING 'SUPERNATURAL') UNCHARY (?) 'UNCLE' UNDULA (NF, FE) UNGUL (AR) UNIC (ORN, CYCLE) UNID (IRECTIONAL) UNIL (INSJAL, OSED, ATRAL)	UNDER UNCI UNGUENT	UNDERS	UNIAXIAL UNIF
UNDER	0	UNDER UNDER, ONE? UNDER THE (PUTPLACE) UNDERSTAND UNDERTAK	UNDERLING UNDERFOOK	UNDERNEATH END CLUD	UNDERCATORY
UNI	1	UNI ONE UNIAXIAL UNIFORM (?WORD IN ITSELF?) UNILATERAL UNILING (UAL) UNILO (BED, CULAR) UNIP (ERSJNAL, LANAR, CLAR, ORENT) END CLUD	UNICYCL	UNIDIRECT	'UNIVERSE'
UP	C	UP UP AND UPHOLSTER UPWARD	UPBRAID UPP END CLUD	UPBRINGING UPSET	UPHEAVAL UPSHOT
VICE	0	VICE VICIES	DEPUTY VICEN END CLUD		
WELL	0	'WELL' WELL-CIL 'WELLS'	GOOD WELL-FAVOR WELL-SPRING END CLUD	WELL-HEEL WELL-FURN	WELL-OFF WELLARAY
WITH	1	WITH WITHDRAW WITHOUT (R) WITHSTAND (?)	COMBINING FORM OF WITH, SEPARATIVE OR OPPOSING FORC 'WITHDRAWING', WITHDREW	WITHOLD	WITHIN END CLUD
XYLO	1	XYLOGRAPH	WOOD XYLOPHON END CLUD		
YESTER	0	YESTER END CLUD			

PREFIX KEY CLUDLIST
ZY30 1 SCI. UNION, CONNECT
ZY3033NESIS ZYGOSPORE END CLUD

1 March 1969

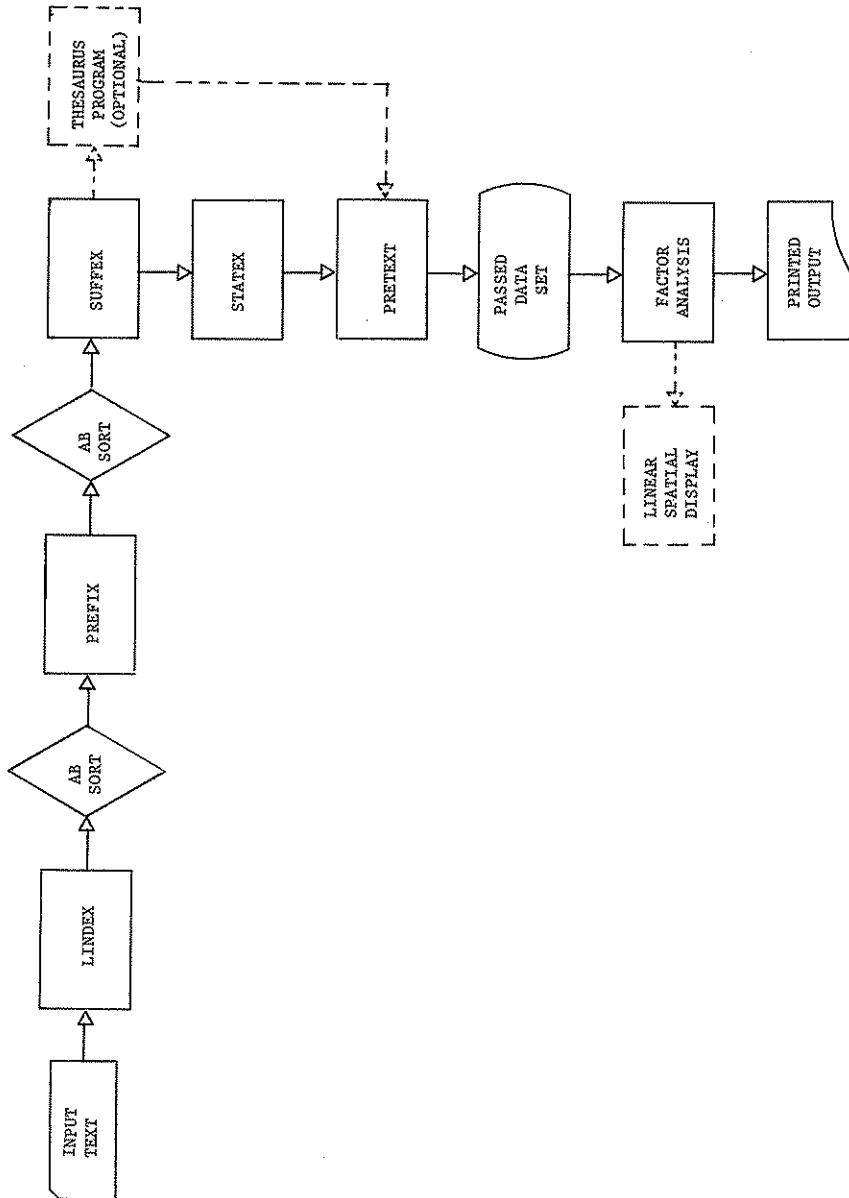
275

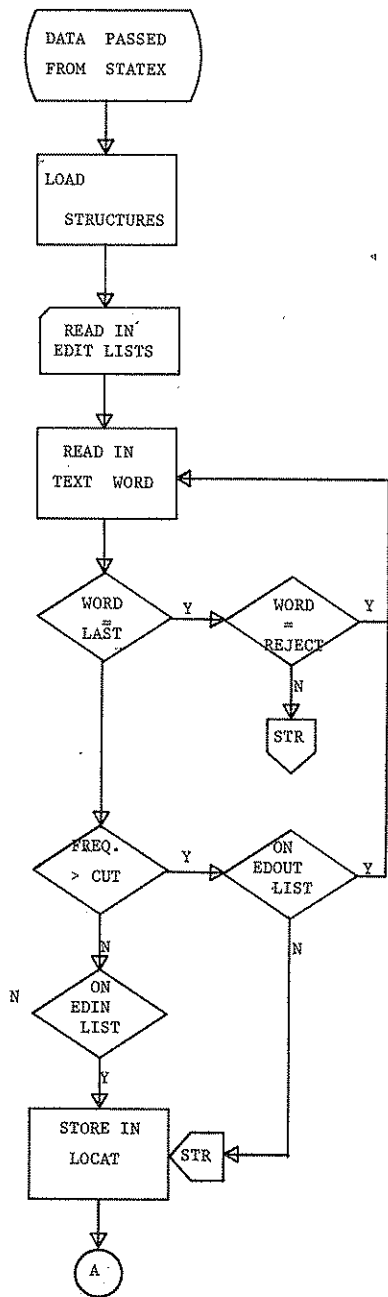
APPENDIX G

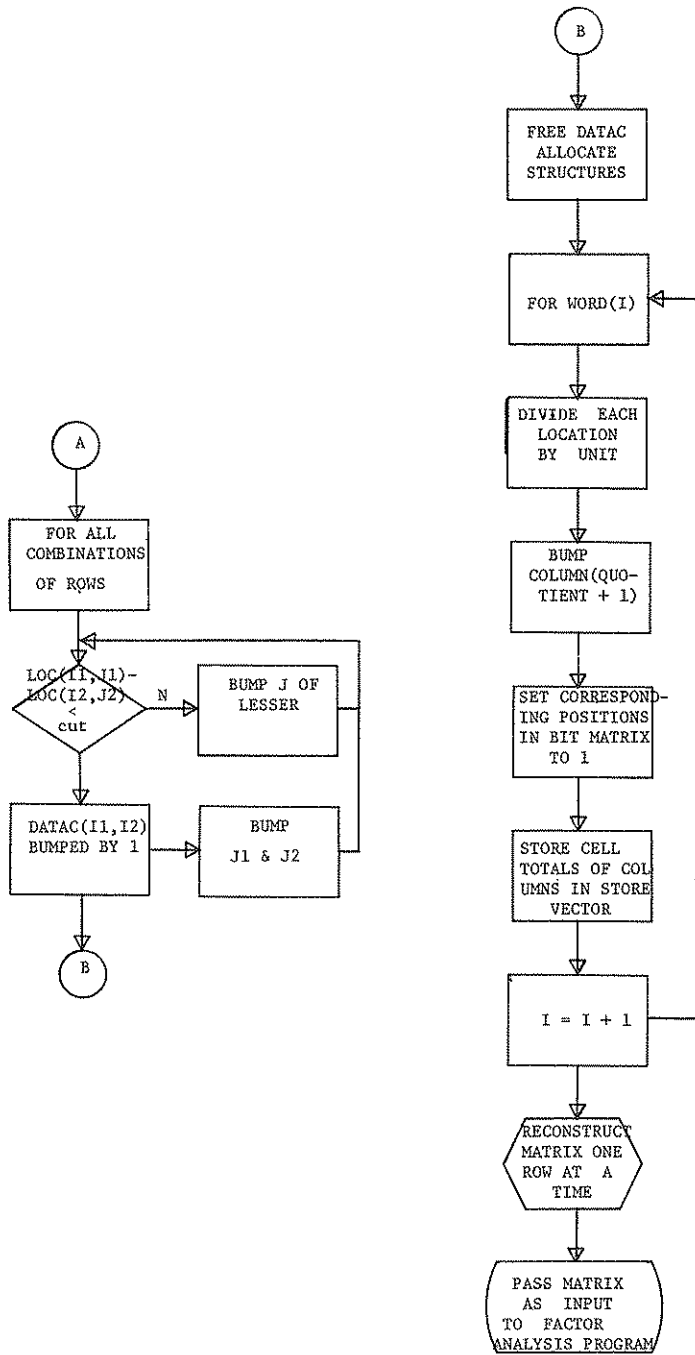
CONTEXT Programs

by

John B. Smith







PRETEXT: PROCEDURE OPTIONS(MAIN):

STMT LEVEL NEST

```

1
PRETEXT: PROCEDURE OPTIONS(MAIN):
/* PRETEXT IS THE PRINCIPLE PROGRAM IN THE CONTEXT SEQUENCE. */
/* IF RECEIVES AS INPUT THE TEXT DATA SET IN MATCHCOUNT. */
/* WORD, AND LINEAR ORDER. IT ALSO RECEIVES TWO EDIT LISTS: */
/* ONE TO EDIT IN WORDS REGARDLESS OF FREQUENCY, ONE FOR */
/* EDITING OUT REGARDLESS OF FREQUENCY. */
/* AN INPUT RECORD IS READ IN. IF THE FREQUENCY OF THE */
/* OF MEAN + N*S.D.) THE WORD IS SELECTED FOR FURTHER C.V.- */
/* SIDERATION, PROVIDING IT HAS NOT BEEN EDITED OUT. */
/* IF THE WORD IS SELECTED, IT IS LOADED INTO A STRUCTURE */
/* ALONG WITH THE LOCATIONS OF ALL TOKENS. FROM THIS */
/* STRUCTURE TWO THINGS ARE COMPUTED. A MATRIX OF */
/* CO-OCCURRENCES EXPRESSED IN ABSOLUTE TERMS IS COMPUTED. */
/* THIS PROCEDURE COMPUTES THE NUMBER OF TIMES EACH PAIR */
/* OF WORDS OCCURS WITHIN A GIVEN NUMBER OF WORDS OF EACH */
/* OTHER. AFTER THIS, THE PROGRAM COMPUTES A DATA MATRIX */
/* THAT IS FED TO THE 'CANNED' FACTOR ANALYSIS PROGRAM. */
/* THIS MATRIX CONSISTS OF THE NUMBER OF OCCURRENCES OF A */
/* GIVEN TERM PER GRID-JUNIT OF TEXT. FROM THIS THE FACTOR */
/* ANALYSIS PROGRAM DEVELOPS FACTORS OF WORD CLUSTERS. */

/* LOCAT IS THE MAIN STORAGE STRUCTURE INTO WHICH EACH WORD */
/* IS LOADED ALON WITH ITS LOCATIONS WITHIN THE TEXT. */
/* THE SIZE PARAMETER FOR THIS MATRIX IS PASSED FROM STMTX */
2 1 DCL 01 LOCAT(NMAP,NMAT) CONTROLLED,
      02 WORD CHAR(6),
      02 MATCH FIXED DEC(5),
      02 NLCC FIXED DEC(3),
      02 LOC (MAX) FIXED DEC(6);
3 1 DCL TEMP FIXED DEC(5) INITIAL(0);

/* DATAC HOLDS THE MATRIX OF ABSOLUTE CO-OCCURRENCE. */
4 1 DCL DATAC(NMAP,NMAT) FIXED DEC(3) CONTROLLED;

5 1 DCL LIN FIXED DEC(5) INITIAL(0),
      COJNT FIXED DEC(2) INITIAL(0),
      MAT FIXED DEC(5) INITIAL(0),
      FREQ FIXED DEC(4) INITIAL(0),
      LFMAT FIXED DEC(5) INITIAL(0),
      X FIXED DEC(3) INITIAL(1),
      CH CHAR(1);
6 1 DCL MAX FIXED DEC(3) INITIAL(0),
      CUT FIXED DEC(3) INITIAL(0),
      NMAP FIXED DEC(3) INITIAL(0);
7 1 DCL WRD CHAR(6) INITIAL(' ');
8 1 DCL ENVIR FIXED DEC(3) INITIAL(0);

/* ED1 AND ED2 HOLD THE TWO EDIT LISTS. */
9 1 DCL ED1 (100) FIXED DEC(6) INITIAL(0);
10 1 DCL ED2 (100) FIXED DEC(6) INITIAL(0);

11 1 DCL X1 FIXED DEC(3) INITIAL(1);
12 1 DCL X2 FIXED DEC(3) INITIAL(1);
13 1 DCL T1 FIXED DEC(1) INITIAL(0);
14 1 DCL T2 FIXED DEC(1) INITIAL(0);

```

PRETEXT: PROCEDURE OPTIONS(MAIN);

STMT LEVEL NEXT

```

15 1 DCL REJECT FIXED DEC(6) INITIAL(0);
16 1 DCL CLOSE1 FIXED DEC(1) INITIAL(0);
17 1 DCL CLOSE2 FIXED DEC(1) INITIAL(0);
18 1 DCL CT2 FIXED DEC(2) INITIAL(0);
19 1 DCL NAMRD12 FIXED DEC(2) INITIAL(0);
20 1 DCL NAMELEF1 FIXED DEC(3) INITIAL(0);
21 1 DCL BLANK CHAR(1) INITIAL(' ');
22 1 DCL NUMS FIXED DEC(2) INITIAL(0);
23 1 DCL CT1 FIXED DEC(3) INITIAL(0);
24 1 ON ENDPFILE(STAT) GO TO OUT1;

26 1 /* CUT, MAX, AND NMAT ARE RECEIVED FROM STATEX AS PARAMETERS.*/
27 1 GET FILE(SCRAT) EDIT(CUT,CH(F(3), X(76), A(1)));
28 1 GET FILE(SCRAT) EDIT(NMAT,CH(F(3), X(75), A(1)));
28 1 GET FILE(SCRAT) EDIT(MAX,CH(F(3), X(76), A(1)));

29 1 /* ENVIR IS THE USER-SPECIFIED PARAMETER FOR DETERMINING
/* CO-OCCURRENCE.
ENVIR = 40;

30 1 PUT EDIT(CUT) (SKIP, F(3));
31 1 PUT EDIT(NMAT) (SKIP, F(3));
32 1 PUT EDIT(MAX) (SKIP, F(3));

33 1 /* STRUCTURES AND VARIABLES ARE ALLOCATED AND INITIALIZED */
34 1 ALLOCATE DATAC(NMAT,NMAT);
35 1 ALLOCATE LOCAT;
36 1 WORD(*) = ' ';
37 1 MATCH(*) = 0;
38 1 NLOC(*) = 0;
39 1 LOC(*,*) = 0;
40 1 DATAC(*,*) = 0;
41 1 ED1(*) = 0;
41 1 ED2(*) = 0;

42 1 /* EDIN IS THE SUBPROCEDURE FOR EDITING IN WORDS REGARDLESS */
43 1 /* OF FREQUENCY */
44 1 EDIN: PROCEDURE;
46 2 T1 = 0;
46 2 IF CLOSE1 = 1 THEN GO TO ENDIN;
47 2 B: IF MAT = ED1(X1);
48 2 THEN DO;
49 2 T1 = 1;
50 2 MATCH(I + 1) = MAT;
51 2 WORD(I + 1) = WRD;
52 2 LSTMAT = MAT;
53 2 NLOC(I + 1) = NLOC(I + 1) + 1;
54 2 LOC(I + 1, 1) = LIN;
55 2 GO TO ENDIN;
56 2 END;
58 2 IF MAT < ED1(K1) THEN GO TO ENDIN;
59 2 ELSE X1 = X1 + 1;
59 2 IF ED1(K1) = 0 THEN DO;

```

PRETEXT: PROCEDURE OPTIONS(MAIN);

STMT LEVEL NEST

```

61 CLOSE1 = 1;
62 GO TO ENDIN;
63 END;
64 IF X1 > 100 THEN DO;
65 CLOSE1 = 1;
66 GO TO ENDIN;
67 END;
68 ELSE GO TO B;
69 ENDIN: END EDIN;
70

```

/* SIMILARLY, EDIUT REJECTS WORDS REGARDLESS OF FREQUENCY */

```

71 EDOUT: PROCEDURE;
72 T2 = 0;
73 IF CLOSE2 = 1 THEN GO TO ENDOUT;
74 C: IF MAT = ED2(X2)
75 THEN DO;
76 REJECT = MAT;
77 T2 = 1;
78 GO TO EDOUT;
79 END;
80 IF MAT < ED2(X2) THEN GO TO ENDOUT;
81 ELSE X2 = X2 + 1;
82 IF ED2(X2) = 0 THEN DO;
83 CLOSE2 = 1;
84 GO TO ENDOUT;
85 END;
86 IF X2 > 100 THEN DO;
87 CLOSE2 = 1;
88 GO TO ENDOUT;
89 END;
90 IF X2 > 100 THEN DO;
91 CLOSE2 = 1;
92 GO TO ENDOUT;
93 END;
94 ELSE GO TO C;
95 ENDOUT: END EDOUT;

```

/* THE EDIN AND EDOUT LISTS ARE READ IN */

```

96 DO I = 1 TO 100;
97 ON ENDFILE(SYSIN) GO TO OUT4;
98 GET FILE(SYSIN) EDIT(ED1(I), CH) (F(5), X(73), A(1));
99 IF ED1(I) = 999999
100 THEN DO;
101 ED1(I) = 0;
102 GO TO OUT3;
103 END;
104 OUT3: DO I = 1 TO 100;
105 GET FILE(SYSIN) EDIT(ED2(I), CH) (F(5), X(73), A(1));
106 END;
107 OUT4: DO I = 1 TO 100;
108 PUT EDIT(ED1(I)) (SKIP, F(6));
109 END;
110 DO I = 1 TO 100;
111
112

```

```

PRETEXT: PROCEDURE OPTIONS(MAIN);
SIML LEVEL NEST
113 1 1 PUT EDIT(ED2(I)) (SKIP, F(6));
114 1 1 END;
115 1 1 DO I = 1 TO NMAX;
116 1 1 X = 2;
117 1 1 R;
/* A TEXT WORD IS READ IN
GST FILE(STAT) EDIT(LIN, MAT, FREQ, WRD, CH)
(X(2), F(6), X(3), F(5), F(4), A(6), X(1), A(1));
/* AFTER THE INITIAL PROCESSING OF A WORD TYPE, ALL
/* SUBSEQUENT TOKENS ARE PROCESSED SIMILARLY.
118 1 1 IF MAT = REJECT THEN GO TO A;
120 1 1 IF MAT = LSTMAT THEN DO;
121 1 1 LOC(I, X) = LIN; /* LOCATION IS LOADED INTO STRUCTURE
122 1 1 NLOC(I) = NLOC(I) + 1;
123 1 1 X = X + 1;
124 1 1 GO TO A;
125 1 1 END;
126 1 1 END;
127 1 1 IF LSTMAT = 0 THEN DO;
129 1 1 IF FREQ >= CUT THEN DO;
130 1 1 CALL EDOUT;
131 1 1 IF F2 = 1 THEN GO TO A;
132 1 1 IF WORD IS NOT EDITED OUT, LOAD DATA INTO STRUCTURE
134 1 1 MATCH(I) = MAT;
135 1 1 WORD(I) = WRD;
136 1 1 LSTMAT = MAT; /* PROVIDES CHECK FOR SUBSEQUENT RECORDS
137 1 1 NLOC(I) = NLOC(I) + 1;
138 1 1 LOC(I, 1) = LIN;
139 1 1 END;
140 1 1 GO TO A;
141 1 1 END;
142 1 1 IF FREQ >= CUT THEN DO;
144 1 1 CALL EDOUT;
145 1 1 IF F2 = 1 THEN GO TO A;
147 1 1 MATCH(I + 1) = MAT;
148 1 1 WORD(I + 1) = WRD;
149 1 1 LSTMAT = MAT;
150 1 1 NLOC(I + 1) = NLOC(I) + 1;
151 1 1 LOC(I + 1, 1) = LIN;
152 1 1 GO TO ENDI;
153 1 1 END;
/* IF FREQ. IS LESS THAN THRESHOLD, CHECK EDIT IN LIST
154 1 1 CALL EDIN;
155 1 1 IF F1 = 1 THEN GO TO ENDI;
157 1 1 GO TO A;
158 1 1 ENDI: END;

```

PRETEXT: PROCEDURE OPTIONS(MAIN);

STMT LEVEL NEST

```

201 1 3 IF ABS(LOC(I1,J1) - LOC(I2,J2)) <= ENWIR
202 1 3 THEN DO;
203 1 3 COUNT = COUNT + 1;
204 1 3 J2 = J2 + 1;
205 1 3 GJ ID J1END;
206 1 3 END;
207 1 3 IF LOC(I1,J1) < LOC(I2,J2) THEN GO TO J1END;
208 1 3 ELSE DO;
209 1 3 J2 = J2 + 1;
210 1 3 GO TO TEST2;
211 1 3 END;
212 1 3 J1END; END;
213 1 3 IZEND: DATAC(I1,I2) = COUNT;
214 1 2 DATAC(I2,I1) = COUNT;
215 1 2 END;
216 1 2 I1END: END;
217 1 1

```

```

218 1 /* THE TABLE OF CO-OCCURRENCES IS PRINTED OUT */
219 1 PUT PAGE;
220 1 DO K = 1 TO NMAT;
221 1 PUT SKIP(2);
222 1 PUT EDIT(DATAC(K,*)) ((NMAT) (F(3), X(2)));
223 1 END;

```

PRE3 DATAC;

```

/* THE REMAINDER OF THE PROGRAM IS CONCERNED WITH THE
/* COMPUTATION OF THE INPUT DATA FOR THE FACTOR ANALYSIS
/* PROCEDURE. BECAUSE OF STORAGE CONSTRAINTS THIS MATRIX
/* CANNOT BE COMPUTED DIRECTLY. IT IS CONSTRUCTED A COLUMN
/* AT A TIME. SINCE IT IS A SPARSE MATRIX, ACTUAL VALUES
/* ARE STORED IN A SINGLE VECTOR OF LOCATIONS. POSITIONS IN
/* THE MATRIX TO BE PASSED ARE MARKED IN A BIT MATRIX OF THE
/* SAME DIMENSIONS AS THE DATA MATRIX. WHERE THERE IS A NON
/* ZERO ENTRY, THE CORRESPONDING BIT IS SET TO 1; OTHERWISE
/* TO 0. WHEN ALL COLUMNS HAVE BEEN SO COMPUTED THE MATRIX
/* TO BE PASSED IS RECONSTRUCTED A ROW AT A TIME BY
/* EXTRACTING THE NON-ZERO ELEMENTS FROM THE STORAGE VECTOR
/* AND INSERTING THEM IN THEIR PROPER POSITIONS.

```

```

224 1 DCL BIT(NUNIT,NMAT) BIT(1) PACKED CONTROLLED;
225 1 DCL PROF FIXED DEC(6) INITIAL(0);
226 1 DCL COL(NUNIT) FIXED DEC(2) CONTROLLED;
227 1 DCL COLTOT(NMAT) FIXED DEC(4) CONTROLLED;
228 1 DCL UNIT FIXED DEC(4) INITIAL(100);
229 1 UNIT = 50;
230 1 DCL NUNIT FIXED DEC(4) INITIAL(100);
231 1 DCL LFIELD(PTOT) FIXED DEC(2) CONTROLLED;
232 1 DCL X3 FIXED DEC(5) INITIAL(1);
233 1 DCL ROW(NMAT) FIXED DEC(2) CONTROLLED;
234 1 DCL POSIT FIXED DEC(5) INITIAL(0);

```

PRETEXT: PROCEDURE OPTIONS (MAIN);

STMP LEVEL NEST

159 1 OUT1:
 PUT PAGE;

 /* AT THIS POINT ALL OF THE INPUT RECORDS HAVE BEEN CHECKED */
 /* AND THE STRUCTURES OF SELECTED WORDS COMPILED. */

160 DO I = 1 TO NMAP;
161 IF MATCH(I) = 0 THEN GO TO OUT2;
162 END;
163 OUT2: NMAP = I - 1;

 /* EACH VECTOR OF LOCATIONS IS CHECKED FOR SEQUENTIAL ORDER */
165 DO I = 1 TO NMAP;
166 DO J = 1 TO (NLOC(I) - 1);
167 IF LOC(I,J) > LOC(I,J + 1) THEN DO;
168 TEMP = LOC(I,J);
169 LOC(I,J) = LOC(I,J + 1);
170 LOC(I,J + 1) = TEMP;
171 DO J2 = J TO 2 BY -1 WHILE (LOC(I,J2) < LOC(I,J2-1));
172 IF LOC(I,J2) < LOC(I,J2 - 1) THEN DO;
173 TEMP = LOC(I,J2);
174 LOC(I,J2) = LOC(I,J2-1);
175 LOC(I,J2-1) = TEMP;
176 END;
177 END;
178 END;
179 END;
180 END;
181 END;
182 END;

 /* THE MAIN DATA STRUCTURE IS PRINTED FOR MANUAL REFERENCE */
183 DO K = 1 TO NMAP;
184 PUT SKIP (2);
185 PUT EDIT(LOCAT(K)) (A(6), X(2), F(5), X(2), F(3), SKIP(1), (MAX)
 (X(2), F(5)));
186 END;

 /* THE MATRIX OF CO-OCCURRENCES EXPRESSED IN ABSOLUTE TERMS */
 /* IS COMPUTED IN THE NEXT BLOCK OF CODE. THE PROCEDURE */
 /* WORKS BY 'RABBINS', ONE EACH PAIR OF LOCATION VECTORS */
 /* CHECKING PAIRS OF LOCATIONS FOR A DIFFERENCE IN VALUE LESS */
 /* THAN THE SPECIFIED ENVIRONMENT PARAMETER. */
187 DO I1 = 1 TO NMAP;
188 DO I2 = 1 TO NMAP;
189 TEST1: IF I1 = I2
190 THEN DO;
191 COUNT = NLOC(I1);
192 GO TO I2END;
193 END;
194 COUNT = 0;
195 J2 = 1;
196 DO J1 = 1 TO MAX WHILE (LOC(I1,J1) ->= 0);
197 TEST2: IF J2 > MAX THEN GO TO I2END;
198 IF LOC(I2,J2) = 0 THEN GO TO I2END;
199 END;

PRETEXT: PROCEDURE OPTIONS (MAIN);

SIME LEVEL NEST

```

235 NUNIF = 19967/UNIT + 1;
236 PUT PAGE; PUT SKIP; PUT DATA(NUNIF);
239 DO I = 1 TO NMAT;
240 PDEF = PDEF + NLOC(I);
241 END;
242 PUT SKIP; PUT DATA(PDEF);

244 ALLOCATE B13;
245 ALLOCATE COL(NUNIT);
246 ALLOCATE ROW(NMAT);
247 ALLOCATE COLTOT(NMAT);
248 ALLOCATE LFIELD(FTOR);
249 BIG(*,*) = 0;
250 ROW(*) = 0;
251 COLTOT(*) = 0;

252 DO J = 1 TO NMAT;
253 COL(*) = 0;

254 DO I = 1 TO MAX;
255 IF LOC(J,I) = 0
256 THEN IF I > 1 /* JUST FOR PRASGER, CHAPT. U*/
257 THEN GO TO OUT5;
258 COL(LOC(J,I)/UNIT + 1) = COL(LOC(J,I)/UNIT + 1) + 1;
259 END;

260 OUT5: DO I4 = 1 TO NUNIT;
261 IF COL(I4) = 0 THEN DO;
263 B13(I4,J) = '1'B;
264 PUT SKIP; PUT DATA(BIG(I4,J));
266 LFIELD(X3) = COL(I4);
267 PUT DATA(LFIELD(X3));
268 X3 = X3 + 1;
269 COLTOT(J) = COLTOT(J) + 1;
270 END;
271 END;

272 END;

273 DCL CF FIXED DEC(3) INITIAL(0);
274 DO I = 1 TO 15 WHILE (CT <= 0);
275 CT = 80*I - NMAT*2;
276 END;
277 CT2 = 80 - CT;
278 NUMB = I - 2;
279 PUT SKIP; PUT DATA(NMAT, NUMB, CT2);
280 NAMED12 = NMAT/12;
281 NAMED12 = NMAT - 12*NAMED12;
283 PUT FILE(PAPASS) EDIT('CONTEXT: FACTOR ANALYSIS OF CHAPT. 1 OF PORTRA
284 IF--UNIT= ',UNIT, BLANK(A), F(4), X(18), A(1));
285 PUT FILE(PAPASS) EDIT(NMAT,NUNIT, '0', '(', NUMB, '(40(F2.0)/)', '

```

PREFEKT: PROCEDURE OPTIONS(MAIN):

STMT LEVEL NEST

```

286 CT2, '(P2.0)*, BLANK)
287 (F(5), F(5), X(21), A(1), X(4), A, F(2), A, X(19), A(1));
288 DO I = 1 TO NAMED12;
289 DO J = 1 TO 12;
290 PUT FILE(PATPASS) EDIT(WORD(12*(I - 1) + J)) (A(5));
291 END;
292 PUT FILE(PATPASS) EDIT(' ') (A(8));
293 END;
294 DO L = (12*NAMED12 + 1); TO NMAN;
295 PUT FILE(PATPASS) EDIT(WORD(I)) (A(6));
296 CT1 = CT1 + 1;
297 PUT FILE(PATPASS) EDIT((120 ' ') (A(89 - CT1*6)));
298 END;
299 PUT PAGE;
300 DO J = 1 TO NUNIT;
301 ROW(*) = 0;
302 DO I = 1 TO NMAN;
303 IF BIG(J,I) = 1
304 THEN DO;
305 POSIT = 0;
306 DO I3 = 1 TO (I - 1) WHILE ((I-1)>= 1);
307 POSIF = POSIF + COLTOT(I3);
308 END;
309 DO I3 = 1 TO J;
310 IF BIG(J,I) = 1 THEN POSIF = POSIF + 1;
311 END;
312 ROW(I) = LFIELD(POSIT);
313 PUT SKIP; PUT DATA(POSIT, LFIELD(POSIT));
314 END;
315 END;
316 PUT FILE(PATPASS) EDIT(ROW(*) ((NMAN) F(2)));
317 PUT FILE(PATPASS) EDIT((120 ' ') (A(89 - CT1)));
318 PUT EDIT(ROW(*) ((SKIP, (NMAN) X(2), F(2)));
319 END;
320 END PREFEKT;

```

1. ORIGINAL
Unit Checked
3. REPRODUCTION
Aut Per
4. DESCRIPTION
5. AUTHORITY
6. REPORT
8a. CONTINUED
NOOO
b. PRO
c.
d.
10. DISTRIBUTION
11. SUPERVISOR
13. ABSTRACT

This
izi:
a P
pac
ana
of
for
int
nat
has
str
has
rep