F The Interpreter JSP Designs

F.1 Scanning The '.dat'

F.1.1 Operations

```
001 infile=open_input(infile_name);
    eof_marker=fgets(line,100,infile);
002 close_file(infile);
003 poscount++;
004 poscount=0;
005 putc(line[poscount],outfile);
006 outfile=open_output(outfile_name);
007 close_file(outfile);
009 eof_marker=fgets(line,100,infile);
010 poscount=6;
011 putc('\n',outfile);
012 printf(" ");
```

F.1.2 Conditions

```
C01 feof(infile) /*eof_marker==NULL*/
C02 feof(infile) || (is_last_full_stop(line) && (line[area2]!='*'))
C03 line[area2]==' '
C04 line[area2]=='*'
C05 line[area2]=='\\'
C06 line[area2]=='-'
C07 line[poscount]=='\n'
C08 line[poscount]!=' '
C09 line[poscount]=='\n' || poscount>72
C10 is_blank_or_end(line)
C11 line[area2]=='\''
```

F.1.3 Data Definitions

```
#include <stdio.h>
#include "alfile.h"
#include "allinesc.h"

#define area2 6
```

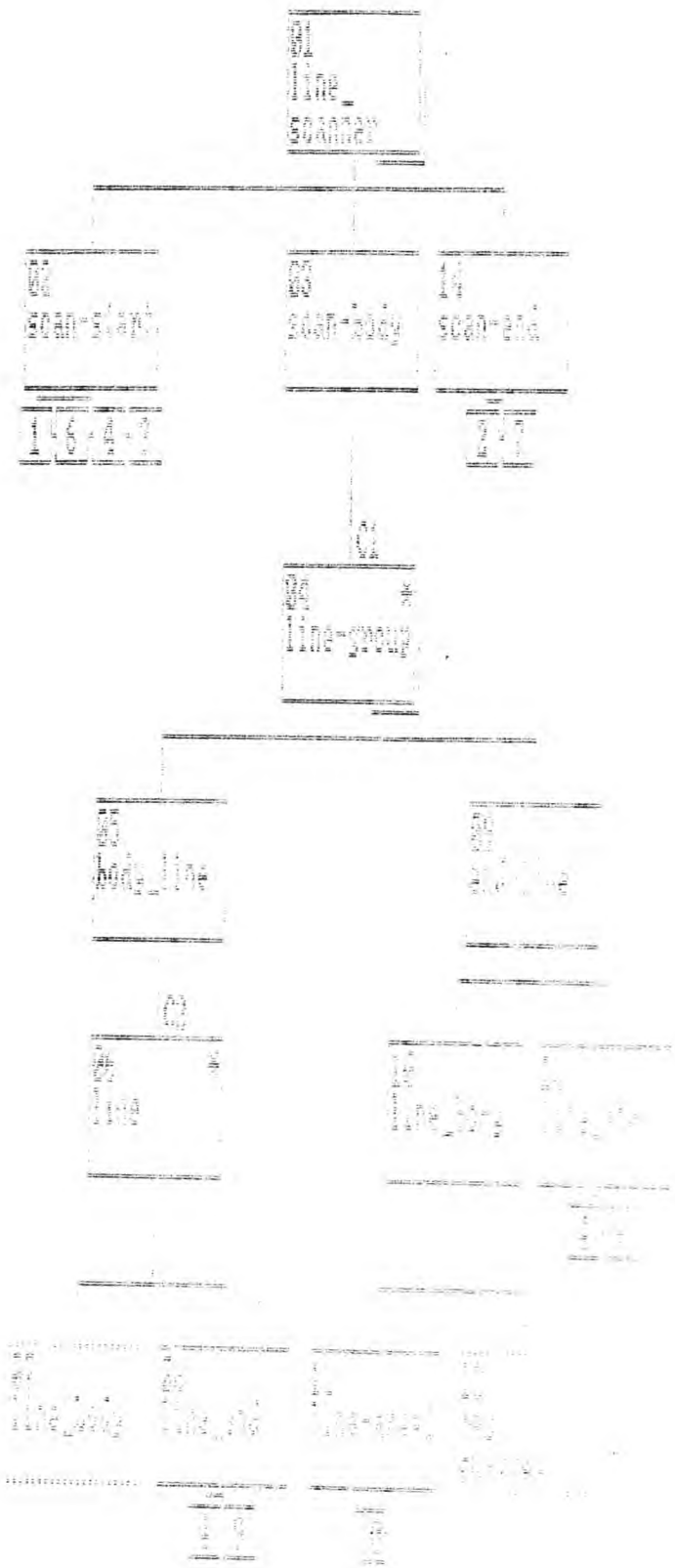
```
#define infile_name "b:altest.dat"
#define outfile_name "b:\\testout.dat"

void main(){

 int poscount=0;
 char line[100];
 FILE *infile,*outfile;
 char *eof_marker;

EOD!
```

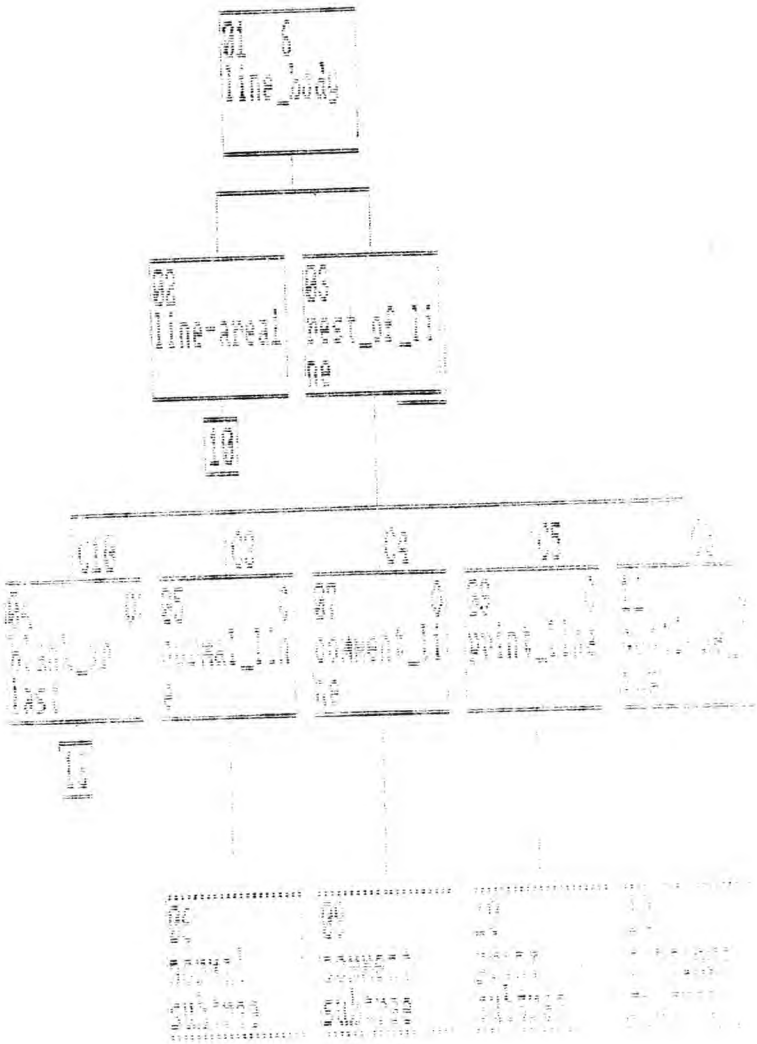
F.1.4 The Header File

```
#ifndef ALLINESC
#define ALLINESC

#include <stdio.h>
#include "alfile.h"

int is_last_full_stop(char *);
int is_blank_or_end(char *);


int is_last_full_stop(char *line)
 {
   int stopflag=0;

   while(*line!='\n')
    {
      if(*line=='.')
          stopflag=1;
      else
          if(*line!=' ')
               stopflag=0;
      line++;
    }
   return stopflag;
 }
```

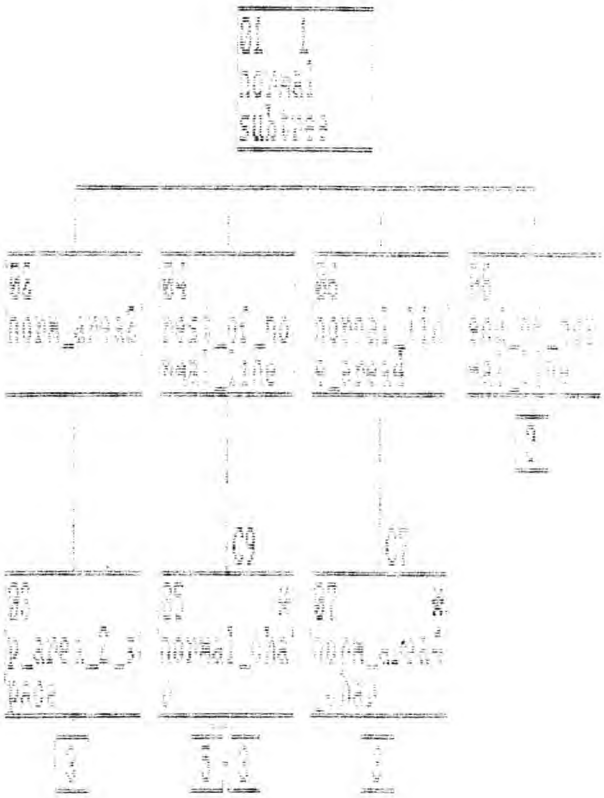
```
int is_blank_or_end(char *line)
  {
   int stopflag=1;

    while(*line!='\n' && *line!=EOF)
     {
      if(*line!=' ')
          stopflag=0;
      line++;
     }
   return(stopflag);
  }

#endif
```

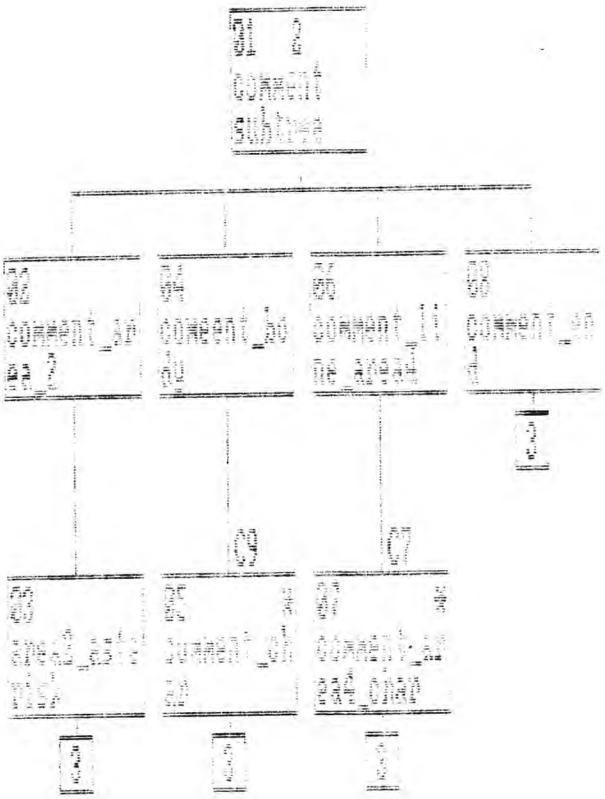
F.1.5 The Diagram

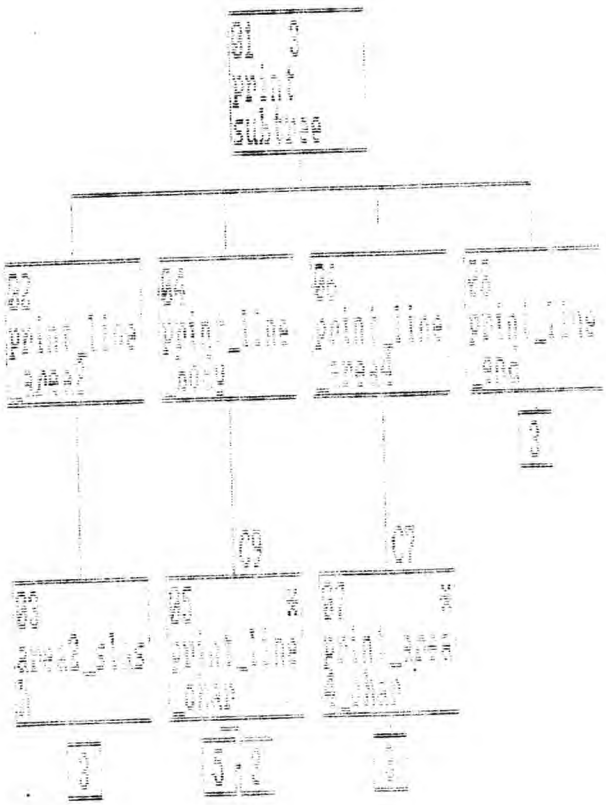


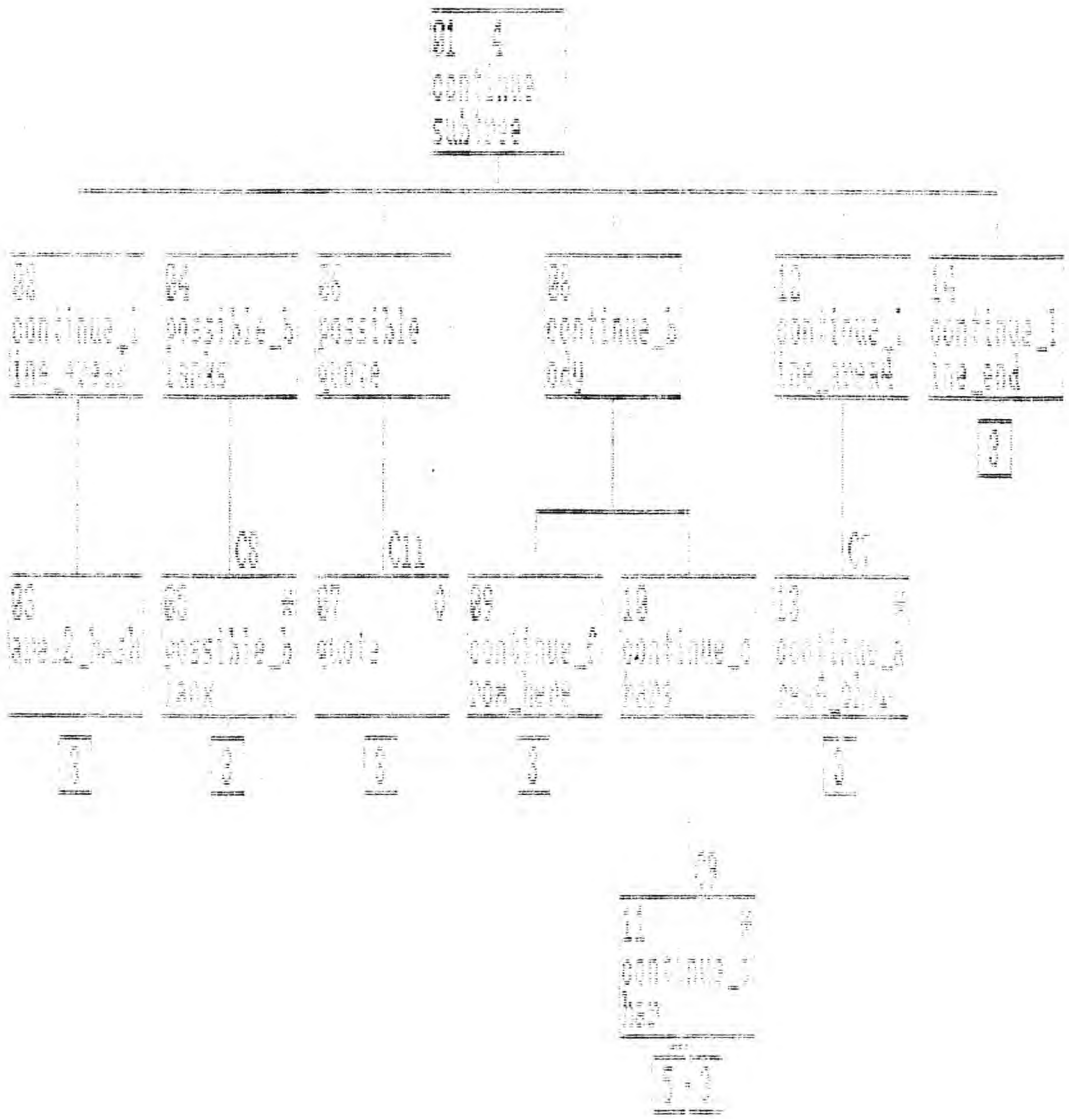

01 0
line_body
02
line-areal
05
next_of_li
ne
10
C10
C2
C3
C5
C6
06
read_ch
last
11
05
convert_li
ne
06
convert_li
ne
08
print_line
12
convert
subtract

C10
C5
C6
blank_or
end_line
print_line
continue_l
ine
normal
subtree
end
comment
print
subtree
end
print
continue
subtree
end
continue




C7
C9


01
print
subtree
02
print_line
_area2
03
area3_size
1
3
04
print_line
_body
09
05
print_line
_char
5,R
06
print_line
_area4
07
area4_char
1
3
08
print_line
_arg
3



F.1.6 The Code

```
#include <stdio.h>
#include "alfile.h"
#include "allinesc.h"

#define area2 6
#define infile_name "b:altest.dat"
#define outfile_name "b:\\testout.dat"

void main(){

 int poscount=0;
 char line[100];
 FILE *infile,*outfile;
 char *eof_marker;

                int g2_001_LINE_BODY_A;
                int g3_001_REST_SUBTREE_A;
                int g4_001_NORMAL_SUBTREE_A;
                int g5_001_COMMENT_SUBTREE_A;
                int g6_001_PRINT_SUBTREE_A;
                int g7_001_CONTINUE_SUBTREE_A;
      /*********************************************************************
      /*********************************************************************
      /****************** DRIVING PROCEDURE    B:\ALLINESC ******************
      /*********************************************************************
       PROCEDURE_DIVISION:
      /**/
      /** line scanner takes an input file*/
      /** reads in the lines from the file and*/
      /** builds up complete cobol statements*/
      /** on a single line*/
      /** it skips comment lines*/
      /** and joins together continuation lines*/
      /** */
      /** NO ERROR HANDLING YET!!!!*/
      /**/
       C1_001_LINE__SCANNER:
```

```
            goto C1_002_SCAN_START;
        C1_002_SCAN_START_EX:
            goto C1_003_SCAN_BODY;
        C1_003_SCAN_BODY_EX:
            goto C1_014_SCAN_END;
        C1_014_SCAN_END_EX:
            goto veryend;
       /**/
        C1_002_SCAN_START:
  /*001*/       infile=open_input(infile_name);
            eof_marker=fgets(line,100,infile);
  /*006*/       outfile=open_output(outfile_name);
  /*004*/       poscount=0;
  /*003*/       poscount++;
            goto C1_002_SCAN_START_EX;
       /**/
        C1_003_SCAN_BODY:
        C1_004_LINE_GROUP_EX:
            if(!(
  /*C01*/         (feof(infile) /*eof_marker==NULL*/)
))
                    goto C1_004_LINE_GROUP;
            goto C1_003_SCAN_BODY_EX;
       /**/
       /** a line group has a number of body lines*/
       /** , these are lines without a full-stop*/
       /** at the end.*/
       /** */
       /** this is followed by a line with a */
       /** fullstop at the end ---the END-LINE*/
       /**/
        C1_004_LINE_GROUP:
            goto C1_005_BODY_LINE;
        C1_005_BODY_LINE_EX:
            goto C1_009_END_LINE;
        C1_009_END_LINE_EX:
            goto C1_004_LINE_GROUP_EX;
       /**/
        C1_005_BODY_LINE:
```

```
      C1_006_LINE_EX:
          if(!(
 /*C02*/          (feof(infile) || (is_last_full_stop(line) && (line[area2]!='
))
                    goto C1_006_LINE;
          goto C1_005_BODY_LINE_EX;
     /**/
      C1_006_LINE:
          goto C1_007_LINE_BODY;
      C1_007_LINE_BODY_EX:
          goto C1_008_LINE_END;
      C1_008_LINE_END_EX:
          goto C1_006_LINE_EX;
     /**/
      C1_007_LINE_BODY:
               g2_001_LINE_BODY_A=0;
          goto C2_001_LINE_BODY;
      C2_001_EXIT01:
          goto C1_007_LINE_BODY_EX;
     /**/
      C1_008_LINE_END:
 /*004*/       poscount=0;
 /*009*/       eof_marker=fgets(line,100,infile);
          goto C1_008_LINE_END_EX;
     /**/
      C1_009_END_LINE:
          goto C1_010_LINE_BODY;
      C1_010_LINE_BODY_EX:
          goto C1_013_LINE_END;
      C1_013_LINE_END_EX:
          goto C1_009_END_LINE_EX;
     /**/
      C1_010_LINE_BODY:
          goto C1_011_LINE_AREA1;
      C1_011_LINE_AREA1_EX:
          goto C1_012_REST_SUBTREE;
      C1_012_REST_SUBTREE_EX:
          goto C1_010_LINE_BODY_EX;
     /**/
```

```
     C1_011_LINE_AREA1:
/*010*/       poscount=6;
          goto C1_011_LINE_AREA1_EX;
    /**/
     C1_012_REST_SUBTREE:
               g3_001_REST_SUBTREE_A=0;
          goto C3_001_REST_SUBTREE;
     C3_001_EXIT01:
          goto C1_012_REST_SUBTREE_EX;
    /**/
     C1_013_LINE_END:
/*004*/       poscount=0;
/*009*/       eof_marker=fgets(line,100,infile);
          goto C1_013_LINE_END_EX;
    /**/
     C1_014_SCAN_END:
/*002*/       close_file(infile);
/*007*/       close_file(outfile);
          goto C1_014_SCAN_END_EX;
    /**/
     C4_001_NORMAL_SUBTREE:
          goto C4_002_NORM_AREA2;
     C4_002_NORM_AREA2_EX:
          goto C4_004_REST_OF_NO_RMAL_LINE;
     C4_004_REST_OF_NO_RMAL_LINE_EX:
          goto C4_006_NORNAL_LIN_E_AREA4;
     C4_006_NORNAL_LIN_E_AREA4_EX:
          goto C4_008_END_OF_NOR_MAL_LINE;
     C4_008_END_OF_NOR_MAL_LINE_EX:
     C4_001_NORMAL_SUBTREE_A:
               switch(g4_001_NORMAL_SUBTREE_A){
               case 0 : goto C4_001_EXIT02;break;
               case 1 : goto C4_001_EXIT03;break;
               }
    /**/
     C4_002_NORM_AREA2:
          goto C4_003_P_AREA_2_S_PACE;
     C4_003_P_AREA_2_S_PACE_EX:
          goto C4_002_NORM_AREA2_EX;
```

```
      /**/
       C4_003_P_AREA_2_S_PACE:
  /*003*/      poscount++;
            goto C4_003_P_AREA_2_S_PACE_EX;
      /**/
       C4_004_REST_OF_NO_RMAL_LINE:
       C4_005_NORMAL_CHA_R_EX:
            if(!(
  /*C09*/         (line[poscount]=='\n' || poscount>72)
))
                   goto C4_005_NORMAL_CHA_R;
            goto C4_004_REST_OF_NO_RMAL_LINE_EX;
      /**/
       C4_005_NORMAL_CHA_R:
  /*005*/      putc(line[poscount],outfile);
  /*003*/      poscount++;
            goto C4_005_NORMAL_CHA_R_EX;
      /**/
       C4_006_NORNAL_LIN_E_AREA4:
       C4_007_NORM_AREA4__CHAR_EX:
            if(!(
  /*C07*/         (line[poscount]=='\n')
))
                   goto C4_007_NORM_AREA4__CHAR;
            goto C4_006_NORNAL_LIN_E_AREA4_EX;
      /**/
       C4_007_NORM_AREA4__CHAR:
  /*003*/      poscount++;
            goto C4_007_NORM_AREA4__CHAR_EX;
      /**/
       C4_008_END_OF_NOR_MAL_LINE:
  /*003*/      poscount++;
            goto C4_008_END_OF_NOR_MAL_LINE_EX;
      /**/
       C5_001_COMMENT_SUBTREE:
            goto C5_002_COMMENT_AR_EA_2;
       C5_002_COMMENT_AR_EA_2_EX:
            goto C5_004_COMEENT_BO_DY;
       C5_004_COMEENT_BO_DY_EX:
```

```
            goto C5_006_COMMENT_LI_NE_AREA4;
        C5_006_COMMENT_LI_NE_AREA4_EX:
            goto C5_008_COMMENT_EN_D;
        C5_008_COMMENT_EN_D_EX:
        C5_001_COMMENT_SUBTREE_A:
                switch(g5_001_COMMENT_SUBTREE_A){
                case 0 : goto C5_001_EXIT02;break;
                case 1 : goto C5_001_EXIT03;break;
                }
       /**/
        C5_002_COMMENT_AR_EA_2:
            goto C5_003_AREA2_ASTE_RISK;
        C5_003_AREA2_ASTE_RISK_EX:
            goto C5_002_COMMENT_AR_EA_2_EX;
       /**/
        C5_003_AREA2_ASTE_RISK:
  /*003*/      poscount++;
            goto C5_003_AREA2_ASTE_RISK_EX;
       /**/
        C5_004_COMEENT_BO_DY:
        C5_005_COMMENT_CH_AR_EX:
            if(!(
  /*C09*/         (line[poscount]=='\n' || poscount>72)
))
                  goto C5_005_COMMENT_CH_AR;
            goto C5_004_COMEENT_BO_DY_EX;
       /**/
        C5_005_COMMENT_CH_AR:
  /*003*/      poscount++;
            goto C5_005_COMMENT_CH_AR_EX;
       /**/
        C5_006_COMMENT_LI_NE_AREA4:
        C5_007_COMMENT_AR_EA4_CHAR_EX:
            if(!(
  /*C07*/         (line[poscount]=='\n')
))
                  goto C5_007_COMMENT_AR_EA4_CHAR;
            goto C5_006_COMMENT_LI_NE_AREA4_EX;
       /**/
```

```
        C5_007_COMMENT_AR_EA4_CHAR:
  /*003*/      poscount++;
            goto C5_007_COMMENT_AR_EA4_CHAR_EX;
       /**/
        C5_008_COMMENT_EN_D:
  /*003*/      poscount++;
            goto C5_008_COMMENT_EN_D_EX;
       /**/
        C6_001_PRINT_SUBTREE:
            goto C6_002_PRINT_LINE__AREA2;
        C6_002_PRINT_LINE__AREA2_EX:
            goto C6_004_PRINT_LINE__BODY;
        C6_004_PRINT_LINE__BODY_EX:
            goto C6_006_PRINT_LINE__AREA4;
        C6_006_PRINT_LINE__AREA4_EX:
            goto C6_008_PRINT_LINE__END;
        C6_008_PRINT_LINE__END_EX:
        C6_001_PRINT_SUBTREE_A:
                 switch(g6_001_PRINT_SUBTREE_A){
                 case 0 : goto C6_001_EXIT02;break;
                 case 1 : goto C6_001_EXIT03;break;
                 }
       /**/
        C6_002_PRINT_LINE__AREA2:
            goto C6_003_AREA2_SLAS_H;
        C6_003_AREA2_SLAS_H_EX:
            goto C6_002_PRINT_LINE__AREA2_EX;
       /**/
        C6_003_AREA2_SLAS_H:
  /*003*/      poscount++;
            goto C6_003_AREA2_SLAS_H_EX;
       /**/
        C6_004_PRINT_LINE__BODY:
        C6_005_PRINT_LINE__CHAR_EX:
            if(!(
  /*C09*/          (line[poscount]=='\n' || poscount>72)
))
                 goto C6_005_PRINT_LINE__CHAR;
            goto C6_004_PRINT_LINE__BODY_EX;
```

```
      /**/
       C6_005_PRINT_LINE__CHAR:
  /*005*/      putc(line[poscount],outfile);
  /*003*/      poscount++;
            goto C6_005_PRINT_LINE__CHAR_EX;
      /**/
       C6_006_PRINT_LINE__AREA4:
       C6_007_PRINT_AREA_4_CHAR_EX:
            if(!(
  /*C07*/          (line[poscount]=='\n')
))
                    goto C6_007_PRINT_AREA_4_CHAR;
            goto C6_006_PRINT_LINE__AREA4_EX;
      /**/
       C6_007_PRINT_AREA_4_CHAR:
  /*003*/      poscount++;
            goto C6_007_PRINT_AREA_4_CHAR_EX;
      /**/
       C6_008_PRINT_LINE__END:
  /*003*/      poscount++;
            goto C6_008_PRINT_LINE__END_EX;
      /**/
       C7_001_CONTINUE_SUBTREE:
            goto C7_002_CONTINUE_L_INE_AREA2;
       C7_002_CONTINUE_L_INE_AREA2_EX:
            goto C7_004_POSSIBLE_B_LANKS;
       C7_004_POSSIBLE_B_LANKS_EX:
            goto C7_006_POSSIBLE_QUOTE;
       C7_006_POSSIBLE_QUOTE_EX:
            goto C7_008_CONTINUE_B_ODY;
       C7_008_CONTINUE_B_ODY_EX:
            goto C7_012_CONTINUE_L_INE_AREA4;
       C7_012_CONTINUE_L_INE_AREA4_EX:
            goto C7_014_CONTINUE_L_INE_END;
       C7_014_CONTINUE_L_INE_END_EX:
       C7_001_CONTINUE_SUBTREE_A:
                switch(g7_001_CONTINUE_SUBTREE_A){
                case 0 : goto C7_001_EXIT02;break;
                case 1 : goto C7_001_EXIT03;break;
```

```
                }
        /**/
         C7_002_CONTINUE_L_INE_AREA2:
             goto C7_003_AREA2_HASH;
         C7_003_AREA2_HASH_EX:
             goto C7_002_CONTINUE_L_INE_AREA2_EX;
        /**/
         C7_003_AREA2_HASH:
   /*003*/      poscount++;
             goto C7_003_AREA2_HASH_EX;
        /**/
         C7_004_POSSIBLE_B_LANKS:
         C7_005_POSSIBLE_B_LANK_EX:
             if(!(
   /*C08*/         (line[poscount]!=' ')
))
                    goto C7_005_POSSIBLE_B_LANK;
             goto C7_004_POSSIBLE_B_LANKS_EX;
        /**/
         C7_005_POSSIBLE_B_LANK:
   /*003*/      poscount++;
             goto C7_005_POSSIBLE_B_LANK_EX;
        /**/
         C7_006_POSSIBLE_QUOTE:
             if((
   /*C11*/         (line[area2]=='\'')
))
                    goto C7_007_QUOTE;
         C7_007_QUOTE_EX:
             goto C7_006_POSSIBLE_QUOTE_EX;
        /**/
         C7_007_QUOTE:
   /*003*/      poscount++;
             goto C7_007_QUOTE_EX;
        /**/
         C7_008_CONTINUE_B_ODY:
             goto C7_009_CONTINUE_F_ROM_HERE;
         C7_009_CONTINUE_F_ROM_HERE_EX:
             goto C7_010_CONTINUE_C_HARS;
```

```
        C7_010_CONTINUE_C_HARS_EX:
            goto C7_008_CONTINUE_B_ODY_EX;
       /**/
        C7_009_CONTINUE_F_ROM_HERE:
  /*003*/      poscount++;
            goto C7_009_CONTINUE_F_ROM_HERE_EX;
       /**/
        C7_010_CONTINUE_C_HARS:
        C7_011_CONTINUE_C_HAR_EX:
            if(!(
  /*C09*/         (line[poscount]=='\n' || poscount>72)
))
                    goto C7_011_CONTINUE_C_HAR;
            goto C7_010_CONTINUE_C_HARS_EX;
       /**/
        C7_011_CONTINUE_C_HAR:
  /*005*/      putc(line[poscount],outfile);
  /*003*/      poscount++;
            goto C7_011_CONTINUE_C_HAR_EX;
       /**/
        C7_012_CONTINUE_L_INE_AREA4:
        C7_013_CONTINUE_A_REA4_CHAR_EX:
            if(!(
  /*C07*/         (line[poscount]=='\n')
))
                    goto C7_013_CONTINUE_A_REA4_CHAR;
            goto C7_012_CONTINUE_L_INE_AREA4_EX;
       /**/
        C7_013_CONTINUE_A_REA4_CHAR:
  /*003*/      poscount++;
            goto C7_013_CONTINUE_A_REA4_CHAR_EX;
       /**/
        C7_014_CONTINUE_L_INE_END:
  /*003*/      poscount++;
            goto C7_014_CONTINUE_L_INE_END_EX;
       /**/
        C3_001_REST_SUBTREE:
            goto C3_002_REST_OF_LI_NE;
        C3_002_REST_OF_LI_NE_EX:
```

```
        C3_001_REST_SUBTREE_A:
                switch(g3_001_REST_SUBTREE_A){
                case 0 : goto C3_001_EXIT01;break;
                }
      /**/
      /** the line is either a normal line*/
      /**                    a comment line*/
      /**                    a print line*/
      /**                 or a continuation line*/
      /**/
       C3_002_REST_OF_LI_NE:
           if((
  /*C10*/         (is_blank_or_end(line))
))
                   goto C3_003_BLANK_OR_END_LINE;
           if((
  /*C03*/         (line[area2]==' ')
))
                   goto C3_004_NORMAL_LIN_E;
           if((
  /*C04*/         (line[area2]=='*')
))
                   goto C3_007_COMMENT_LI_NE;
           if((
  /*C05*/         (line[area2]=='\\')
))
                   goto C3_010_PRINT_LINE;
           if((
  /*C06*/         (line[area2]=='-')
))
                   goto C3_013_CONTINUE_L_INE;
      C3_003_BLANK_OR_END_LINE_EX:
      C3_004_NORMAL_LIN_E_EX:
      C3_007_COMMENT_LI_NE_EX:
      C3_010_PRINT_LINE_EX:
      C3_013_CONTINUE_L_INE_EX:
          goto C3_002_REST_OF_LI_NE_EX;
     /**/
      C3_003_BLANK_OR_END_LINE:
```

```
/*012*/      printf(" ");
          goto C3_003_BLANK_OR_END_LINE_EX;
     /**/
      C3_004_NORMAL_LIN_E:
          goto C3_005_NORMAL_SUBTREE;
      C3_005_NORMAL_SUBTREE_EX:
          goto C3_006_END_NORMAL;
      C3_006_END_NORMAL_EX:
          goto C3_004_NORMAL_LIN_E_EX;
     /**/
      C3_005_NORMAL_SUBTREE:
               g4_001_NORMAL_SUBTREE_A=0;
          goto C4_001_NORMAL_SUBTREE;
      C4_001_EXIT02:
          goto C3_005_NORMAL_SUBTREE_EX;
     /**/
      C3_006_END_NORMAL:
/*011*/      putc('\n',outfile);
          goto C3_006_END_NORMAL_EX;
     /**/
      C3_007_COMMENT_LI_NE:
          goto C3_008_COMMENT_SUBTREE;
      C3_008_COMMENT_SUBTREE_EX:
          goto C3_009_END_COMMENT;
      C3_009_END_COMMENT_EX:
          goto C3_007_COMMENT_LI_NE_EX;
     /**/
      C3_008_COMMENT_SUBTREE:
               g5_001_COMMENT_SUBTREE_A=0;
          goto C5_001_COMMENT_SUBTREE;
      C5_001_EXIT02:
          goto C3_008_COMMENT_SUBTREE_EX;
     /**/
      C3_009_END_COMMENT:
/*011*/      putc('\n',outfile);
          goto C3_009_END_COMMENT_EX;
     /**/
      C3_010_PRINT_LINE:
          goto C3_011_PRINT_SUBTREE;
```

```
      C3_011_PRINT_SUBTREE_EX:
          goto C3_012_END_PRINT;
      C3_012_END_PRINT_EX:
          goto C3_010_PRINT_LINE_EX;
     /**/
      C3_011_PRINT_SUBTREE:
               g6_001_PRINT_SUBTREE_A=0;
          goto C6_001_PRINT_SUBTREE;
      C6_001_EXIT02:
          goto C3_011_PRINT_SUBTREE_EX;
     /**/
      C3_012_END_PRINT:
/*011*/       putc('\n',outfile);
          goto C3_012_END_PRINT_EX;
     /**/
      C3_013_CONTINUE_L_INE:
          goto C3_014_CONTINUE_SUBTREE;
      C3_014_CONTINUE_SUBTREE_EX:
          goto C3_015_END_CONTINUE;
      C3_015_END_CONTINUE_EX:
          goto C3_013_CONTINUE_L_INE_EX;
     /**/
      C3_014_CONTINUE_SUBTREE:
               g7_001_CONTINUE_SUBTREE_A=0;
          goto C7_001_CONTINUE_SUBTREE;
      C7_001_EXIT02:
          goto C3_014_CONTINUE_SUBTREE_EX;
     /**/
      C3_015_END_CONTINUE:
/*011*/       putc('\n',outfile);
          goto C3_015_END_CONTINUE_EX;
     /**/
      C2_001_LINE_BODY:
          goto C2_002_LINE_AREA1;
      C2_002_LINE_AREA1_EX:
          goto C2_003_REST_OF_LI_NE;
      C2_003_REST_OF_LI_NE_EX:
      C2_001_LINE_BODY_A:
               switch(g2_001_LINE_BODY_A){
```

```
                case 0 : goto C2_001_EXIT01;break;
                }
        /**/
         C2_002_LINE_AREA1:
  /*010*/      poscount=6;
            goto C2_002_LINE_AREA1_EX;
        /**/
        /** the line is either a normal line*/
        /**                    a comment line*/
        /**                    a print line*/
        /**                 or a continuation line*/
        /**/
         C2_003_REST_OF_LI_NE:
            if((
  /*C10*/         (is_blank_or_end(line))
))
                   goto C2_004_BLANK_OR_LAST;
            if((
  /*C03*/         (line[area2]==' ')
))
                   goto C2_005_NORMAL_LIN_E;
            if((
  /*C04*/         (line[area2]=='*')
))
                   goto C2_007_COMMENT_LI_NE;
            if((
  /*C05*/         (line[area2]=='\\')
))
                   goto C2_009_PRINT_LINE;
            if((
  /*C06*/         (line[area2]=='-')
))
                   goto C2_011_CONTINUE_L_INE;
        C2_004_BLANK_OR_LAST_EX:
        C2_005_NORMAL_LIN_E_EX:
        C2_007_COMMENT_LI_NE_EX:
        C2_009_PRINT_LINE_EX:
        C2_011_CONTINUE_L_INE_EX:
            goto C2_003_REST_OF_LI_NE_EX;
```

```
    /**/
     C2_004_BLANK_OR_LAST:
/*012*/      printf(" ");
          goto C2_004_BLANK_OR_LAST_EX;
    /**/
     C2_005_NORMAL_LIN_E:
          goto C2_006_NORMAL_SUBTREE;
     C2_006_NORMAL_SUBTREE_EX:
          goto C2_005_NORMAL_LIN_E_EX;
    /**/
     C2_006_NORMAL_SUBTREE:
               g4_001_NORMAL_SUBTREE_A=1;
          goto C4_001_NORMAL_SUBTREE;
     C4_001_EXIT03:
          goto C2_006_NORMAL_SUBTREE_EX;
    /**/
     C2_007_COMMENT_LI_NE:
          goto C2_008_COMMENT_SUBTREE;
     C2_008_COMMENT_SUBTREE_EX:
          goto C2_007_COMMENT_LI_NE_EX;
    /**/
     C2_008_COMMENT_SUBTREE:
               g5_001_COMMENT_SUBTREE_A=1;
          goto C5_001_COMMENT_SUBTREE;
     C5_001_EXIT03:
          goto C2_008_COMMENT_SUBTREE_EX;
    /**/
     C2_009_PRINT_LINE:
          goto C2_010_PRINT_SUBTREE;
     C2_010_PRINT_SUBTREE_EX:
          goto C2_009_PRINT_LINE_EX;
    /**/
     C2_010_PRINT_SUBTREE:
               g6_001_PRINT_SUBTREE_A=1;
          goto C6_001_PRINT_SUBTREE;
     C6_001_EXIT03:
          goto C2_010_PRINT_SUBTREE_EX;
    /**/
     C2_011_CONTINUE_L_INE:
```

```
        goto C2_012_CONTINUE_SUBTREE;
    C2_012_CONTINUE_SUBTREE_EX:
        goto C2_011_CONTINUE_L_INE_EX;
   /**/
    C2_012_CONTINUE_SUBTREE:
            g7_001_CONTINUE_SUBTREE_A=1;
        goto C7_001_CONTINUE_SUBTREE;
    C7_001_EXIT03:
        goto C2_012_CONTINUE_SUBTREE_EX;
   /**/
   /*   Data Analysis Map*/
   /**/
   /*                        ------------- Data Analysis By -------------*
   /**/
   /*                        BOX TYPE                OPERATION  ALLOCATION*
   /*  Tree name: LINE_-SCANNER*/
   /**/
   /*                        Leaf      :   5          Operations:  11*/
   /*                                                 Quits     :   0*/
   /*                        Selections:   0*/
   /*                        Sequences :   5*/
   /*                        Iterations:   2*/
   /*                        Backtracks:   0*/
   /*                        Subtrees  :   2   -->    LINE_BODY*/
   /*                                                 REST-SUBTREE*/
   /**/
   /*  Tree name: NORMAL-SUBTREE*/
   /**/
   /*                        Leaf      :   4          Operations:   5*/
   /*                                                 Quits     :   0*/
   /*                        Selections:   0*/
   /*                        Sequences :   2*/
   /*                        Iterations:   2*/
   /*                        Backtracks:   0*/
   /*                        Subtrees  :   0*/
   /**/
   /*  Tree name: COMMENT-SUBTREE*/
   /**/
   /*                        Leaf      :   4          Operations:   4*/
```

```
/*                                                  Quits      :   0*/
/*                        Selections:   0*/
/*                        Sequences :   2*/
/*                        Iterations:   2*/
/*                        Backtracks:   0*/
/*                        Subtrees  :   0*/
/**/
/*  Tree name: PRINT-SUBTREE*/
/**/
/*                        Leaf      :   4          Operations:   5*/
/*                                                 Quits      :   0*/
/*                        Selections:   0*/
/*                        Sequences :   2*/
/*                        Iterations:   2*/
/*                        Backtracks:   0*/
/*                        Subtrees  :   0*/
/**/
/*  Tree name: CONTINUE-SUBTREE*/
/**/
/*                        Leaf      :   7          Operations:   8*/
/*                                                 Quits      :   0*/
/*                        Selections:   1*/
/*                        Sequences :   3*/
/*                        Iterations:   3*/
/*                        Backtracks:   0*/
/*                        Subtrees  :   0*/
/**/
/*  Tree name: REST-SUBTREE*/
/**/
/*                        Leaf      :   5          Operations:   5*/
/*                                                 Quits      :   0*/
/*                        Selections:   1*/
/*                        Sequences :   5*/
/*                        Iterations:   0*/
/*                        Backtracks:   0*/
/*                        Subtrees  :   4    -->   NORMAL-SUBTREE*/
/*                                                 COMMENT-SUBTREE*/
/*                                                 PRINT-SUBTREE*/
/*                                                 CONTINUE-SUBTREE*/
```

```
        /**/
        /*  Tree name: LINE_BODY*/
        /**/
        /*                          Leaf       :   2          Operations:   2*/
        /*                                                     Quits      :   0*/
        /*                          Selections:   1*/
        /*                          Sequences :   5*/
        /*                          Iterations:   0*/
        /*                          Backtracks:   0*/
        /*                          Subtrees  :   4   -->     NORMAL-SUBTREE*/
        /*                                                    COMMENT-SUBTREE*/
        /*                                                    PRINT-SUBTREE*/
        /*                                                    CONTINUE-SUBTREE*/
        /**/
        /**/
        /**/
veryend: ;
}
```

F.2 Tokenising The '.dat'

F.2.1 Operations

```
001 infile=open_input("b:\\testout.dat");
002 close_file(infile);
003 currch=getc(infile);
    if(currch!=EOF) nextch=getc(infile);
004 currch=nextch;
    if(nextch!=EOF) nextch=getc(infile);
005 ch=chp;
006 *ch='\0';
007 /*printf("string token for %s\n",chp);*/
008 /*printf("open brace\n");*/
009 /*printf("close brace\n");*/
010 /*printf("word token for %s\n",chp);*/
011 length_of=0;
012 length_of++;
013 token->token_type=NON_NUMERIC_LITERAL_LENGTH;
    token->value.keyword_number=length_of;
014 /*printf("fullstop\n");*/
015 /*printf("END LINE TOKEN\n");*/
    /*getchar();*/
016 /*printf("START OF LINE\n");*/
017 listpointer=root;
018 listpointer=check_pointer(listpointer,currch);
019 dat=get_data(listpointer);
020 /*printf("keyword - %s number %d",chp,dat);*/
021 outfile=open_output("b:\\testout2.dat");
022 close_file(outfile);
023 write_token(outfile,token);
024 token->token_type=OPEN_BRACE;
    token->value.keyword_number=OPEN_BRACE;
025 token->token_type=CLOSE_BRACE;
    token->value.keyword_number=CLOSE_BRACE;
026 token->token_type=FULLSTOP;
    token->value.keyword_number=FULLSTOP;
027 token->token_type=KEYWORD;
    token->value.keyword_number=dat;
```

```
028 token->token_type=NON_KEYWORD;
    token->value.string=chp;
029 token->token_type=NON_NUMERIC_LITERAL;
    token->value.string=chp;
030 *ch=currch;
031 ch++;
032 token->token_type=NON_KEYWORD_LENGTH;
    token->value.keyword_number=length_of;
033 token=&t;
```

F.2.2 Conditions

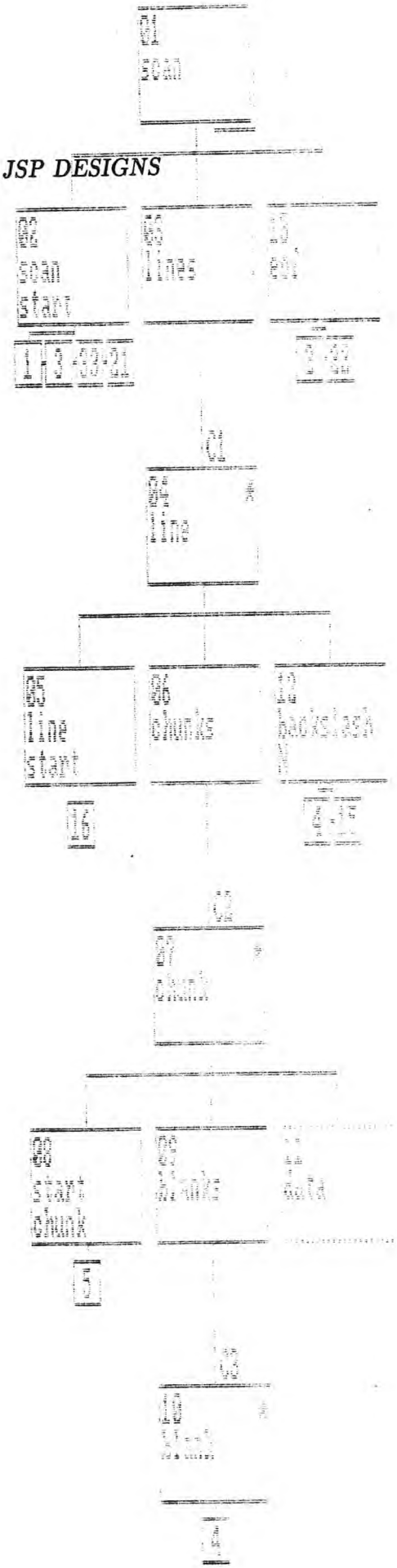
```
C01 currch==EOF
C02 currch=='\n'
C03 currch!=' '
C04 currch=='"'
C05 currch=='\''
C06 currch=='"' || currch=='\''
C07 currch=='('
C08 currch==')'
C09 currch!='"' &&
    currch!='\'' &&
    currch!='(' &&
    currch!=')' &&
    currch!='.' &&
    currch!='\n'
C10 currch==' ' || currch=='\n' ||
    currch=='(' || currch==')' ||
    currch=='.'
C11 currch=='"' && nextch!='"'
C12 currch=='\'' && nextch!='\''
C13 currch=='"' && nextch=='"'
C14 currch=='\'' && nextch=='\''
C15 currch=='.'
```

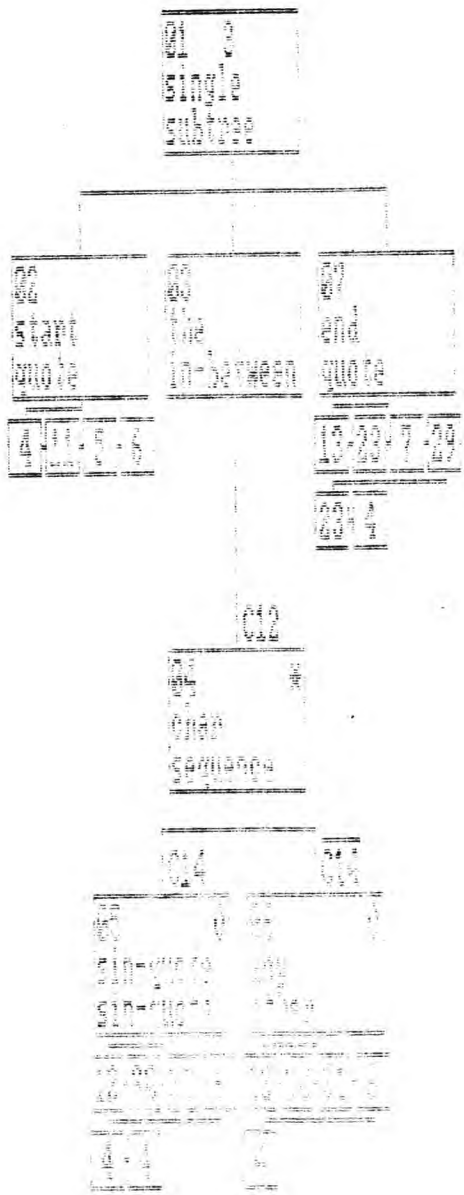
F.2.3 Data Definitions

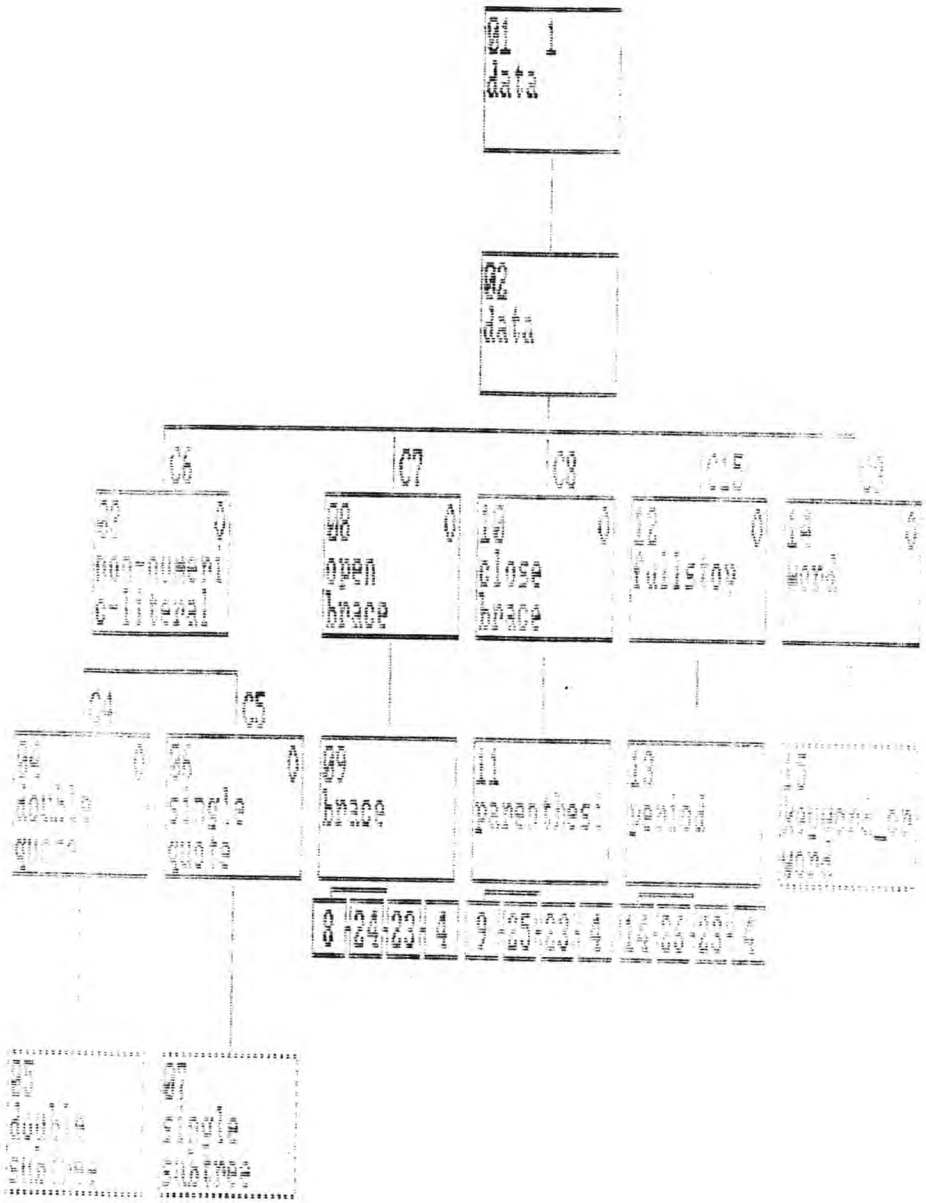
```
#include <stdio.h>
#include "alstring.h"
```

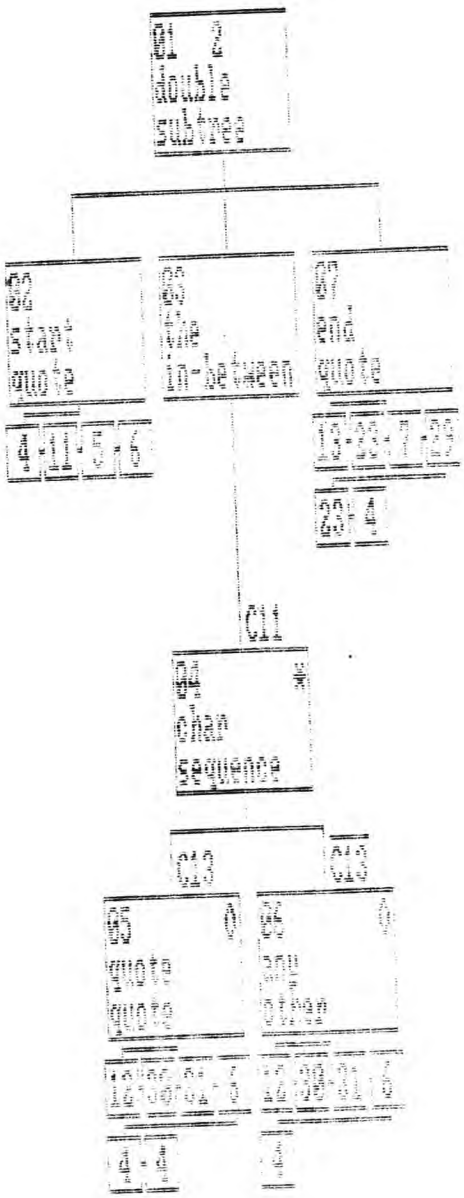
```
#include "alfile.h"
#include "staglist.h"
#include "token.h"


void main(){

 FILE *infile,*outfile;
 char currch,nextch,*ch,chp[100];
 staggered_list
*listpointer,*root=create_list_with_file("B:keywords");
 int dat,length_of;
 TOKEN *token,t;


EOD!
```

F.2.4 The Diagram




C1
single
subtree
C3
start
quote
C4
the
in-between
C5
end
quote
C2
C6
char
sequence
C14
C15


01 1
data
02
data
06
07
08
09
10
11
12
13
14
15
non-numeric-literal
open brace
close brace
fullstop
word
04
05
double quote
single quote
09
brace
11
parenthesis
13
period
15
keyword_or_word
8 124 231 4
9 125 232 4
14 66 23 4
05
double quote
07
single quote


01 2
double
subtree
02
start
quote
03
file
in-between
07
end
quote
C11
04 *
char
sequence
C13
C13
05
quote
quote
06
any
other

F THE INTERPRETER JSP DES

F.2.5 The Code

```
#include <stdio.h>
#include "alstring.h"
#include "alfile.h"
#include "staglist.h"
#include "token.h"



void main(){

 FILE *infile,*outfile;
 char currch,nextch,*ch,chp[100];
 staggered_list *listpointer,*root=create_list_with_file("B:keywords");
 int dat,length_of;
 TOKEN *token,t;



                int g2_001_DATA_A;
                int g3_001_DOUBLE_SUBTREE_A;
                int g4_001_SINGLE_SUBTREE_A;
                int g5_001_KEYWORD_OR_WORD_A;
        /*********************************************************************
        /*********************************************************************
        /******************** DRIVING PROCEDURE    B:\ALSCAN ******************
        /*********************************************************************
         PROCEDURE_DIVISION:
        /**/
        /** takes the output from allinescan*/
        /** which is one line with all the cobol*/
        /** details on it, rather than cobol details*/
        /** */
        /** being spread across a number of lines*/
        /**/
         C1_001_SCAN:
             goto C1_002_SCAN_START;
         C1_002_SCAN_START_EX:
```

```
            goto C1_003_LINES;
        C1_003_LINES_EX:
            goto C1_013_EOF;
        C1_013_EOF_EX:
            goto veryend;
       /**/
        C1_002_SCAN_START:
  /*001*/       infile=open_input("b:\\testout.dat");
  /*003*/       currch=getc(infile);
            if(currch!=EOF) nextch=getc(infile);
  /*033*/       token=&t;
  /*021*/       outfile=open_output("b:\\testout2.dat");
            goto C1_002_SCAN_START_EX;
       /**/
        C1_003_LINES:
        C1_004_LINE_EX:
            if(!(
  /*C01*/         (currch==EOF)
))
                    goto C1_004_LINE;
            goto C1_003_LINES_EX;
       /**/
        C1_004_LINE:
            goto C1_005_LINE_START;
        C1_005_LINE_START_EX:
            goto C1_006_CHUNKS;
        C1_006_CHUNKS_EX:
            goto C1_012_BACKSLASH_N;
        C1_012_BACKSLASH_N_EX:
            goto C1_004_LINE_EX;
       /**/
        C1_005_LINE_START:
  /*016*/       /*printf("START OF LINE\n");*/
            goto C1_005_LINE_START_EX;
       /**/
        C1_006_CHUNKS:
        C1_007_CHUNK_EX:
            if(!(
  /*C02*/         (currch=='\n')
```

```
))
                    goto C1_007_CHUNK;
            goto C1_006_CHUNKS_EX;
        /**/
         C1_007_CHUNK:
            goto C1_008_START_CHUNK;
         C1_008_START_CHUNK_EX:
            goto C1_009_BLANKS;
         C1_009_BLANKS_EX:
            goto C1_011_DATA;
         C1_011_DATA_EX:
            goto C1_007_CHUNK_EX;
        /**/
         C1_008_START_CHUNK:
  /*005*/       ch=chp;
            goto C1_008_START_CHUNK_EX;
        /**/
         C1_009_BLANKS:
         C1_010_BLANK_EX:
            if(!(
  /*C03*/          (currch!=' ')
))
                    goto C1_010_BLANK;
            goto C1_009_BLANKS_EX;
        /**/
         C1_010_BLANK:
  /*004*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C1_010_BLANK_EX;
        /**/
         C1_011_DATA:
                g2_001_DATA_A=0;
            goto C2_001_DATA;
         C2_001_EXIT01:
            goto C1_011_DATA_EX;
        /**/
         C1_012_BACKSLASH_N:
  /*004*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
```

```
  /*015*/      /*printf("END LINE TOKEN\n");*/
            /*getchar();*/
            goto C1_012_BACKSLASH_N_EX;
        /**/
         C1_013_EOF:
  /*002*/      close_file(infile);
  /*022*/      close_file(outfile);
            goto C1_013_EOF_EX;
        /**/
         C2_001_DATA:
            goto C2_002_DATA;
         C2_002_DATA_EX:
         C2_001_DATA_A:
                  switch(g2_001_DATA_A){
                  case 0 : goto C2_001_EXIT01;break;
                  }
        /**/
         C2_002_DATA:
            if((
  /*C06*/         (currch=='"' || currch=='\'')
))
                   goto C2_003_NON_NUMERI_C_LITERAL;
            if((
  /*C07*/         (currch=='(')
))
                   goto C2_008_OPEN_BRACE;
            if((
  /*C08*/         (currch==')')
))
                   goto C2_010_CLOSE_BRACE;
            if((
  /*C15*/         (currch=='.')
))
                   goto C2_012_FULLSTOP;
            if((
  /*C09*/         (currch!='"' &&
               currch!='\'' &&
               currch!='(' &&
               currch!=')' &&
```

```
                currch!='.' &&
                currch!='\n')
))
                    goto C2_014_WORD;
        C2_003_NON_NUMERI_C_LITERAL_EX:
        C2_008_OPEN_BRACE_EX:
        C2_010_CLOSE_BRACE_EX:
        C2_012_FULLSTOP_EX:
        C2_014_WORD_EX:
            goto C2_002_DATA_EX;
       /**/
        C2_003_NON_NUMERI_C_LITERAL:
            if((
  /*C04*/          (currch=='"')
))
                    goto C2_004_DOUBLE_QUOTE;
            if((
  /*C05*/          (currch=='\'')
))
                    goto C2_006_SINGLE_QUOTE;
        C2_004_DOUBLE_QUOTE_EX:
        C2_006_SINGLE_QUOTE_EX:
            goto C2_003_NON_NUMERI_C_LITERAL_EX;
       /**/
        C2_004_DOUBLE_QUOTE:
            goto C2_005_DOUBLE_SUBTREE;
        C2_005_DOUBLE_SUBTREE_EX:
            goto C2_004_DOUBLE_QUOTE_EX;
       /**/
        C2_005_DOUBLE_SUBTREE:
                g3_001_DOUBLE_SUBTREE_A=0;
            goto C3_001_DOUBLE_SUBTREE;
        C3_001_EXIT01:
            goto C2_005_DOUBLE_SUBTREE_EX;
       /**/
        C2_006_SINGLE_QUOTE:
            goto C2_007_SINGLE_SUBTREE;
        C2_007_SINGLE_SUBTREE_EX:
            goto C2_006_SINGLE_QUOTE_EX;
```

```
    /**/
     C2_007_SINGLE_SUBTREE:
              g4_001_SINGLE_SUBTREE_A=0;
         goto C4_001_SINGLE_SUBTREE;
     C4_001_EXIT01:
         goto C2_007_SINGLE_SUBTREE_EX;
    /**/
     C2_008_OPEN_BRACE:
         goto C2_009_BRACE;
     C2_009_BRACE_EX:
         goto C2_008_OPEN_BRACE_EX;
    /**/
     C2_009_BRACE:
/*008*/      /*printf("open brace\n");*/
/*024*/      token->token_type=OPEN_BRACE;
         token->value.keyword_number=OPEN_BRACE;
/*023*/      write_token(outfile,token);
/*004*/      currch=nextch;
         if(nextch!=EOF) nextch=getc(infile);
         goto C2_009_BRACE_EX;
    /**/
     C2_010_CLOSE_BRACE:
         goto C2_011_PARENTHESI;
     C2_011_PARENTHESI_EX:
         goto C2_010_CLOSE_BRACE_EX;
    /**/
     C2_011_PARENTHESI:
/*009*/      /*printf("close brace\n");*/
/*025*/      token->token_type=CLOSE_BRACE;
         token->value.keyword_number=CLOSE_BRACE;
/*023*/      write_token(outfile,token);
/*004*/      currch=nextch;
         if(nextch!=EOF) nextch=getc(infile);
         goto C2_011_PARENTHESI_EX;
    /**/
     C2_012_FULLSTOP:
         goto C2_013_PERIOD;
     C2_013_PERIOD_EX:
         goto C2_012_FULLSTOP_EX;
```

```
    /**/
     C2_013_PERIOD:
/*014*/      /*printf("fullstop\n");*/
/*026*/      token->token_type=FULLSTOP;
         token->value.keyword_number=FULLSTOP;
/*023*/      write_token(outfile,token);
/*004*/      currch=nextch;
         if(nextch!=EOF) nextch=getc(infile);
         goto C2_013_PERIOD_EX;
    /**/
     C2_014_WORD:
         goto C2_015_KEYWORD_OR_WORD;
     C2_015_KEYWORD_OR_WORD_EX:
         goto C2_014_WORD_EX;
    /**/
     C2_015_KEYWORD_OR_WORD:
              g5_001_KEYWORD_OR_WORD_A=0;
         goto C5_001_KEYWORD_OR_WORD;
     C5_001_EXIT01:
         goto C2_015_KEYWORD_OR_WORD_EX;
    /**/
     C3_001_DOUBLE_SUBTREE:
         goto C3_002_START_QUOTE;
     C3_002_START_QUOTE_EX:
         goto C3_003_THE_IN_BETWEEN;
     C3_003_THE_IN_BETWEEN_EX:
         goto C3_007_END_QUOTE;
     C3_007_END_QUOTE_EX:
     C3_001_DOUBLE_SUBTREE_A:
              switch(g3_001_DOUBLE_SUBTREE_A){
              case 0 : goto C3_001_EXIT01;break;
              }
    /**/
     C3_002_START_QUOTE:
/*004*/      currch=nextch;
         if(nextch!=EOF) nextch=getc(infile);
/*011*/      length_of=0;
/*005*/      ch=chp;
/*006*/      *ch='\0';
```

```
            goto C3_002_START_QUOTE_EX;
        /**/
         C3_003_THE_IN_BETWEEN:
         C3_004_CHAR_SEQUENCE_EX:
            if(!(
  /*C11*/          (currch=='"' && nextch!='"')
))
                     goto C3_004_CHAR_SEQUENCE;
            goto C3_003_THE_IN_BETWEEN_EX;
        /**/
         C3_004_CHAR_SEQUENCE:
            if((
  /*C13*/          (currch=='"' && nextch=='"')
))
                     goto C3_005_QUOTE_QUOTE;
            if(!(
  /*C13*/          (currch=='"' && nextch=='"')
))
                     goto C3_006_ANY_OTHER;
         C3_005_QUOTE_QUOTE_EX:
         C3_006_ANY_OTHER_EX:
            goto C3_004_CHAR_SEQUENCE_EX;
        /**/
         C3_005_QUOTE_QUOTE:
  /*012*/       length_of++;
  /*030*/       *ch=currch;
  /*031*/       ch++;
  /*006*/       *ch='\0';
  /*004*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
  /*004*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C3_005_QUOTE_QUOTE_EX;
        /**/
         C3_006_ANY_OTHER:
  /*012*/       length_of++;
  /*030*/       *ch=currch;
  /*031*/       ch++;
  /*006*/       *ch='\0';
```

```
/*004*/       currch=nextch;
          if(nextch!=EOF) nextch=getc(infile);
          goto C3_006_ANY_OTHER_EX;
    /**/
     C3_007_END_QUOTE:
/*013*/       token->token_type=NON_NUMERIC_LITERAL_LENGTH;
          token->value.keyword_number=length_of;
/*023*/       write_token(outfile,token);
/*007*/       /*printf("string token for %s\n",chp);*/
/*029*/       token->token_type=NON_NUMERIC_LITERAL;
          token->value.string=chp;
/*023*/       write_token(outfile,token);
/*004*/       currch=nextch;
          if(nextch!=EOF) nextch=getc(infile);
          goto C3_007_END_QUOTE_EX;
    /**/
     C4_001_SINGLE_SUBTREE:
          goto C4_002_START_QUOTE;
     C4_002_START_QUOTE_EX:
          goto C4_003_THE_IN_BETWEEN;
     C4_003_THE_IN_BETWEEN_EX:
          goto C4_007_END_QUOTE;
     C4_007_END_QUOTE_EX:
     C4_001_SINGLE_SUBTREE_A:
               switch(g4_001_SINGLE_SUBTREE_A){
               case 0 : goto C4_001_EXIT01;break;
               }
    /**/
     C4_002_START_QUOTE:
/*004*/       currch=nextch;
          if(nextch!=EOF) nextch=getc(infile);
/*011*/       length_of=0;
/*005*/       ch=chp;
/*006*/       *ch='\0';
          goto C4_002_START_QUOTE_EX;
    /**/
     C4_003_THE_IN_BETWEEN:
     C4_004_CHAR_SEQUENCE_EX:
          if(!(
```

```
  /*C12*/          (currch=='\'' && nextch!='\'')
))
                    goto C4_004_CHAR_SEQUENCE;
            goto C4_003_THE_IN_BETWEEN_EX;
      /**/
       C4_004_CHAR_SEQUENCE:
            if((
  /*C14*/          (currch=='\'' && nextch=='\'')
))
                    goto C4_005_SIN_QUOTE_SIN_QUOTE;
            if(!(
  /*C14*/          (currch=='\'' && nextch=='\'')
))
                    goto C4_006_ANY_OTHER;
       C4_005_SIN_QUOTE_SIN_QUOTE_EX:
       C4_006_ANY_OTHER_EX:
            goto C4_004_CHAR_SEQUENCE_EX;
      /**/
       C4_005_SIN_QUOTE_SIN_QUOTE:
  /*012*/       length_of++;
  /*030*/       *ch=currch;
  /*031*/       ch++;
  /*006*/       *ch='\0';
  /*004*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
  /*004*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C4_005_SIN_QUOTE_SIN_QUOTE_EX;
      /**/
       C4_006_ANY_OTHER:
  /*012*/       length_of++;
  /*030*/       *ch=currch;
  /*031*/       ch++;
  /*006*/       *ch='\0';
  /*004*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C4_006_ANY_OTHER_EX;
      /**/
       C4_007_END_QUOTE:
```

```
/*013*/      token->token_type=NON_NUMERIC_LITERAL_LENGTH;
         token->value.keyword_number=length_of;
/*023*/      write_token(outfile,token);
/*007*/      /*printf("string token for %s\n",chp);*/
/*029*/      token->token_type=NON_NUMERIC_LITERAL;
         token->value.string=chp;
/*023*/      write_token(outfile,token);
/*004*/      currch=nextch;
         if(nextch!=EOF) nextch=getc(infile);
         goto C4_007_END_QUOTE_EX;
    /**/
     C5_001_KEYWORD_OR_WORD:
         goto C5_002_KEYWORD_WORD_BCKTR;
     C5_002_KEYWORD_WORD_BCKTR_EX:
     C5_001_KEYWORD_OR_WORD_A:
              switch(g5_001_KEYWORD_OR_WORD_A){
              case 0 : goto C5_001_EXIT01;break;
              }
    /**/
     C5_002_KEYWORD_WORD_BCKTR:
         goto C5_003_KEYWORD;
     C5_003_KEYWORD_EX:
     C5_008_NON_KEYWORD_EX:
         goto C5_002_KEYWORD_WORD_BCKTR_EX;
    /**/
     C5_003_KEYWORD:
         goto C5_004_KEYWORD_START;
     C5_004_KEYWORD_START_EX:
         goto C5_005_KEYWORD_CHARS;
     C5_005_KEYWORD_CHARS_EX:
         goto C5_007_KEYWORD_END;
     C5_007_KEYWORD_END_EX:
         goto C5_003_KEYWORD_EX;
    /**/
     C5_004_KEYWORD_START:
/*005*/      ch=chp;
/*011*/      length_of=0;
/*017*/      listpointer=root;
/*030*/      *ch=currch;
```

```
  /*031*/      ch++;
  /*006*/      *ch='\0';
  /*018*/      listpointer=check_pointer(listpointer,currch);
  /*012*/      length_of++;
           if((
  /*Q01*/         (!isupper(currch)) && currch!='-'
))
                 goto C5_008_NON_KEYWORD;
           if((
  /*Q02*/         listpointer==NULL
))
                 goto C5_008_NON_KEYWORD;
  /*004*/      currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
           goto C5_004_KEYWORD_START_EX;
      /**/
       C5_005_KEYWORD_CHARS:
       C5_006_KEYWORD_CHAR_EX:
           if(!(
  /*C10*/         (currch==' ' || currch=='\n' ||
              currch=='(' || currch==')' ||
              currch=='.')
))
                 goto C5_006_KEYWORD_CHAR;
           goto C5_005_KEYWORD_CHARS_EX;
      /**/
       C5_006_KEYWORD_CHAR:
  /*030*/      *ch=currch;
  /*031*/      ch++;
  /*006*/      *ch='\0';
  /*012*/      length_of++;
  /*018*/      listpointer=check_pointer(listpointer,currch);
           if((
  /*Q01*/         (!isupper(currch)) && currch!='-'
))
                 goto C5_008_NON_KEYWORD;
           if((
  /*Q02*/         listpointer==NULL
))
```

```
                    goto C5_008_NON_KEYWORD;
  /*004*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C5_006_KEYWORD_CHAR_EX;
       /**/
        C5_007_KEYWORD_END:
  /*019*/       dat=get_data(listpointer);
            if((
  /*Q03*/          dat==NULL
))
                    goto C5_008_NON_KEYWORD;
  /*020*/       /*printf("keyword - %s number %d",chp,dat);*/
  /*027*/       token->token_type=KEYWORD;
            token->value.keyword_number=dat;
  /*023*/       write_token(outfile,token);
            goto C5_007_KEYWORD_END_EX;
       /**/
        C5_008_NON_KEYWORD:
            goto C5_009_START_NON;
        C5_009_START_NON_EX:
            goto C5_010_CHARS;
        C5_010_CHARS_EX:
            goto C5_012_END_WORD;
        C5_012_END_WORD_EX:
            goto C5_008_NON_KEYWORD_EX;
       /**/
        C5_009_START_NON:
  /*004*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C5_009_START_NON_EX;
       /**/
        C5_010_CHARS:
        C5_011_CHAR_EX:
            if(!(
  /*C10*/          (currch==' ' || currch=='\n' ||
                currch=='(' || currch==')' ||
                currch=='.')
))
                    goto C5_011_CHAR;
```

```
            goto C5_010_CHARS_EX;
        /**/
         C5_011_CHAR:
/*030*/       *ch=currch;
/*031*/       ch++;
/*006*/       *ch='\0';
/*012*/       length_of++;
/*004*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C5_011_CHAR_EX;
        /**/
         C5_012_END_WORD:
/*010*/       /*printf("word token for %s\n",chp);*/
/*032*/       token->token_type=NON_KEYWORD_LENGTH;
            token->value.keyword_number=length_of;
/*023*/       write_token(outfile,token);
/*028*/       token->token_type=NON_KEYWORD;
            token->value.string=chp;
/*023*/       write_token(outfile,token);
/*013*/       token->token_type=NON_NUMERIC_LITERAL_LENGTH;
            token->value.keyword_number=length_of;
            goto C5_012_END_WORD_EX;
        /**/
        /*   Data Analysis Map*/
        /**/
        /*                          ------------- Data Analysis By -------------*
        /**/
        /*                          BOX TYPE                OPERATION  ALLOCATION*
        /*  Tree name: SCAN*/
        /**/
        /*                          Leaf      :   6          Operations:  11*/
        /*                                                   Quits     :   0*/
        /*                          Selections:   0*/
        /*                          Sequences :   3*/
        /*                          Iterations:   3*/
        /*                          Backtracks:   0*/
        /*                          Subtrees  :   1   -->    DATA*/
        /**/
        /*  Tree name: DATA*/
```

```
/**/
/*                          Leaf       :   3          Operations:  12*/
/*                                                    Quits      :   0*/
/*                          Selections:   2*/
/*                          Sequences :   7*/
/*                          Iterations:   0*/
/*                          Backtracks:   0*/
/*                          Subtrees  :   3   -->    DOUBLE-SUBTREE*/
/*                                                   SINGLE-SUBTREE*/
/*                                                   KEYWORD_OR-WORD*/
/**/
/*  Tree name: DOUBLE-SUBTREE*/
/**/
/*                          Leaf       :   4          Operations:  21*/
/*                                                    Quits      :   0*/
/*                          Selections:   1*/
/*                          Sequences :   1*/
/*                          Iterations:   1*/
/*                          Backtracks:   0*/
/*                          Subtrees  :   0*/
/**/
/*  Tree name: SINGLE-SUBTREE*/
/**/
/*                          Leaf       :   4          Operations:  21*/
/*                                                    Quits      :   0*/
/*                          Selections:   1*/
/*                          Sequences :   1*/
/*                          Iterations:   1*/
/*                          Backtracks:   0*/
/*                          Subtrees  :   0*/
/**/
/*  Tree name: KEYWORD_OR-WORD*/
/**/
/*                          Leaf       :   6          Operations:  31*/
/*                                                    Quits      :   5*/
/*                          Selections:   0*/
/*                          Sequences :   3*/
/*                          Iterations:   2*/
/*                          Backtracks:   1*/
```

```
        /*                          Subtrees  :   0*/
        /**/
        /**/
        /**/
veryend: ;
}
```

F.3 Building The Symbol Table

F.3.1 Operations

```
001 infile=open_input("b:testout2.dat");
    outfile=open_output("b:symtable.out");
    token=&t;
002 if(!feof(infile)) alfeof=get_token(infile,token);
003 current=create_node();
    current->level=str_to_int(token->value.string);
004 PARENT(current)=stacktop(stack);
    CHILD(stacktop(stack))=current;
005 stackpop(stack);
006 SIBLING(stacktop(stack))=current;
    PARENT(current)=PARENT(stacktop(stack));
007 NAME(current)=token->value.string;
008 current->info->type=VARIABLE;
009 current->info->type=QUALIFIER;
010 stackpush(stack,current);
011 PARENT(current)=NULL;
    first_01=current;
012 hashval=hash(NAME(current))%SYMTAB_ARRAY_SIZE;
    current->own.hash=hashval;
    add_node_to_sym(current,hashval,symtab);
013 current->own.unique=unique_no;
    unique_no++;
014 close_file(infile);
    save_nodes(symtab,outfile);
    close_file(outfile);
015 number_of=0;
016 pointer=token->value.string;
017 pointer++;
018 number_of++;
019 *pointer2=*pointer;
    pointer++;
    pointer2++;
020 pointer2=count_string;
021 *pointer2='\0';
    number_of+=str_to_int(count_string);
```

```
022 current->info->type=INT;
023 current->info->type=CHAR;
024 current->info->data_c=(char *)malloc(number_of+1);
    current->info->data_c[0]='\0';
025 current->info->data_n=number_of;
026 current->info->data_c=NULL;
    current->info->data_n=0;
```

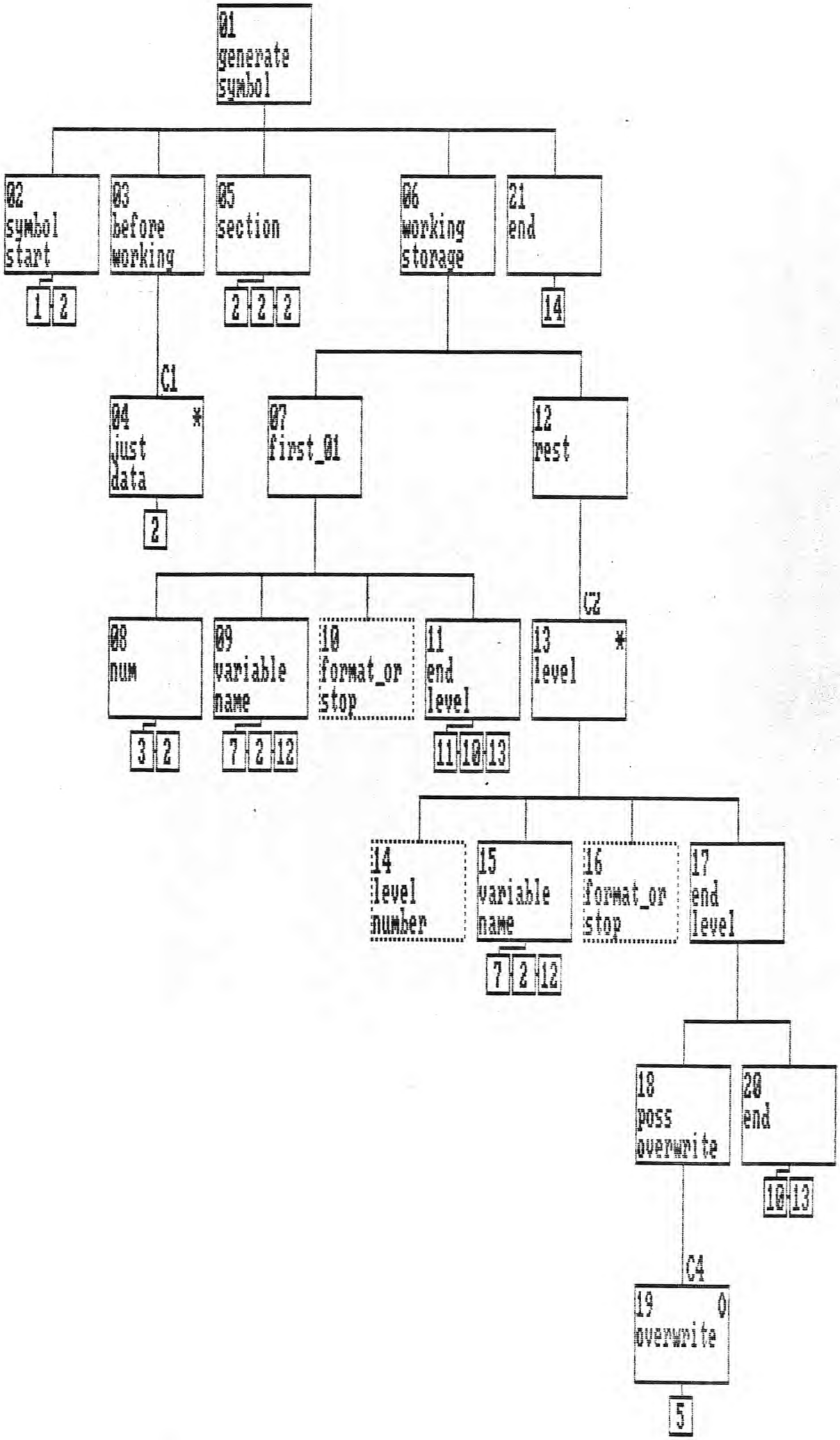
F.3.2 Conditions

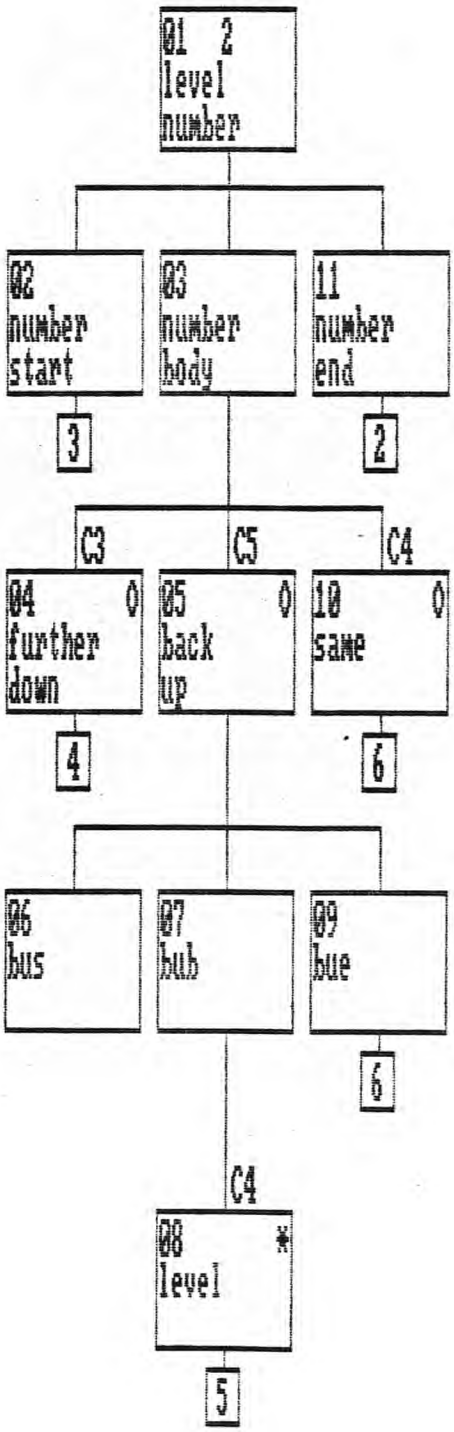
```
C01 token->value.keyword_number==WORKING_STORAGE
C02 alfeof==1
C03 current->level > stacktop(stack)->level
C04 current->level==stacktop(stack)->level
C05 current->level < stacktop(stack)->level
C06 token->token_type==FULLSTOP
C07 *pointer=='\0'
C08 *pointer=='9'
C09 *pointer=='X'
C10 token->token_type==OPEN_BRACE
```

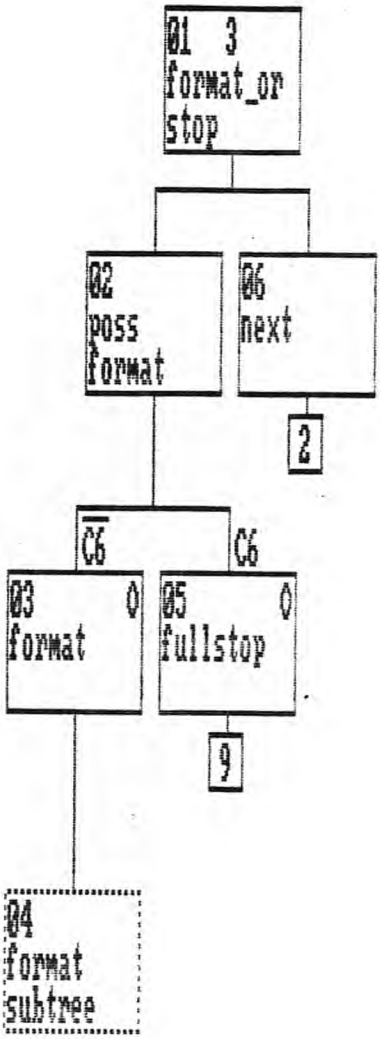
F.3.3 Data Definitions

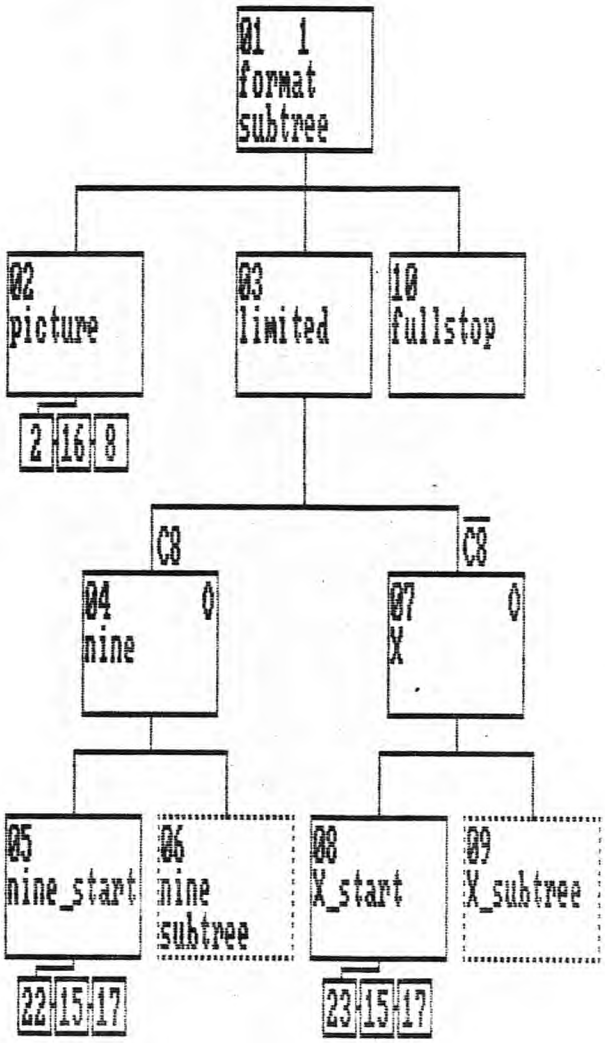
```
#include <stdio.h>
#include "alutil.h"
#include "alstring.h"
#include "alfile.h"
#include "node.h"
#include "stckbyar.h"
#include "symtab.h"
#include "keywords.h"
#include "token.h"


void main(){

 FILE *infile,*outfile;
 NODE *current,*first_01;
 TOKEN *token,t;
```

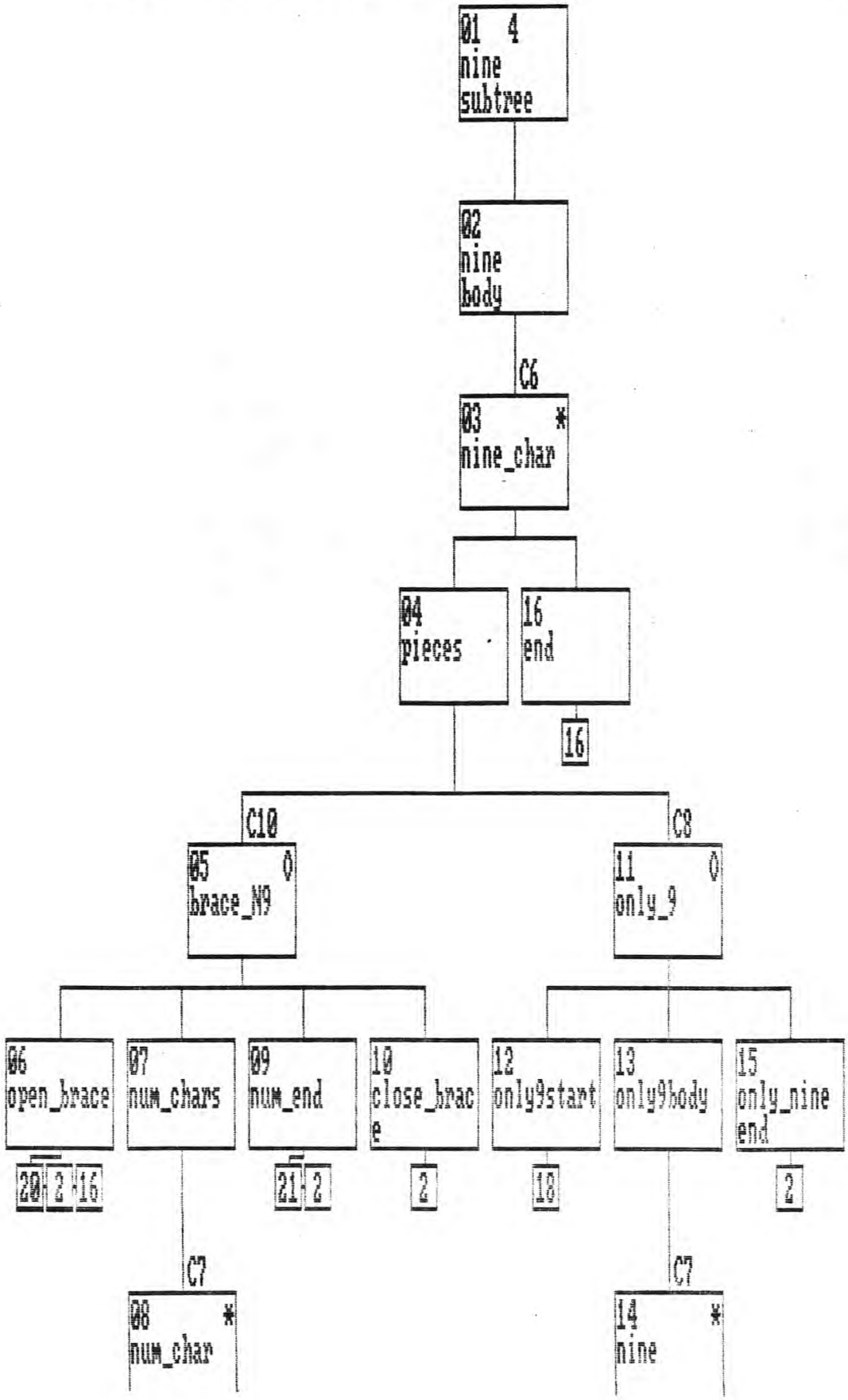
```
 STACKBYARRAY *stack=create_stack_by_array(50);
 SYMTAB_ENTRY **symtab=create_symbol_table_array(SYMTAB_ARRAY_SIZE);
 int unique_no=1;
 int hashval=0;
 int number_of;
 char *pointer,*pointer2,count_string[20];
 int alfeof=0;


EOD!
```

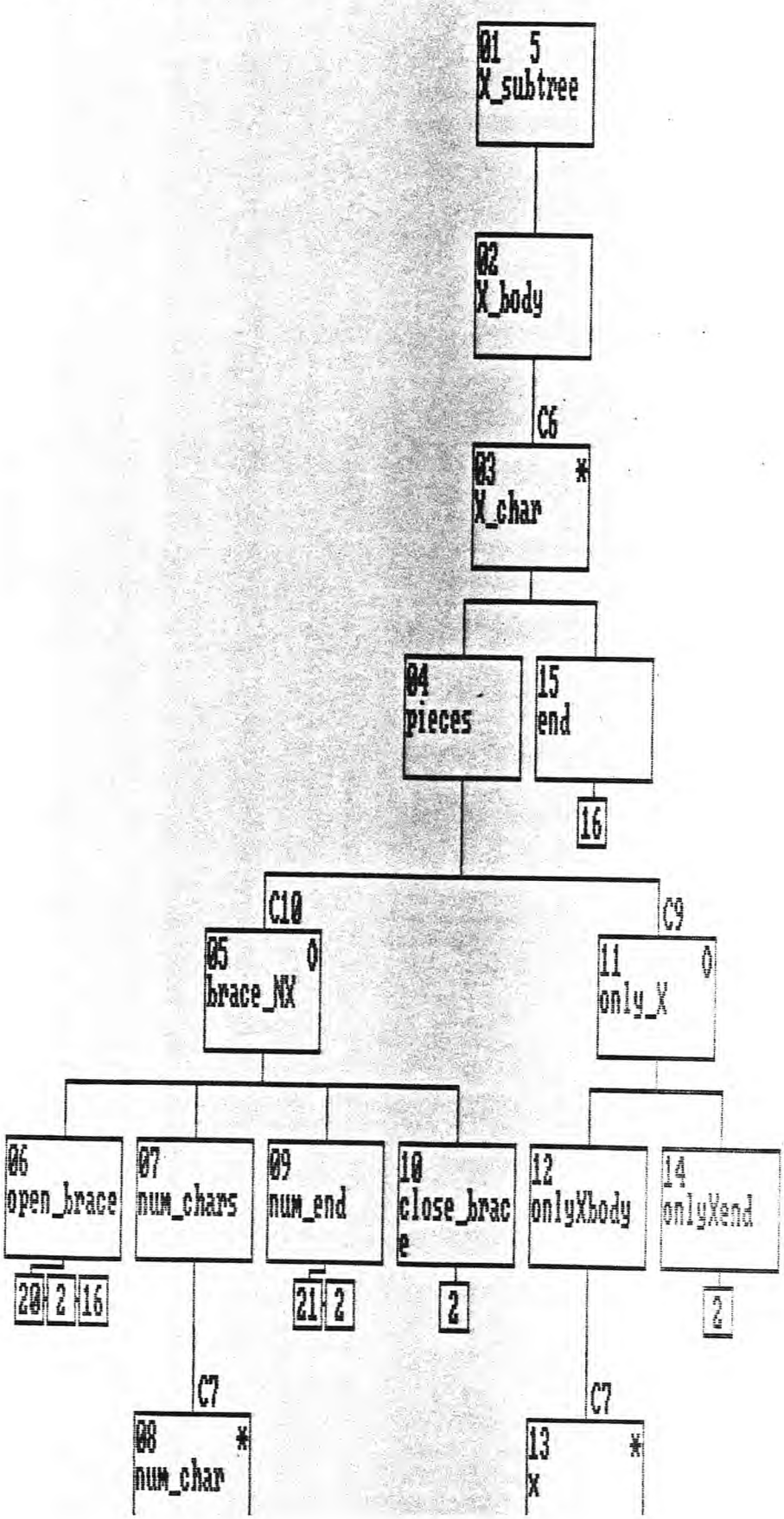

01 generate symbol
02 symbol start
1 2
03 before working
05 section
2 2 2
06 working storage
21 end
14
C1
04 * just data
2
07 first_01
12 rest
C2
08 num
3 2
09 variable name
7 2 12
10 format_or stop
11 end level
11 10 13
13 * level
14 level number
15 variable name
7 2 12
16 format_or stop
17 end level
18 poss overwrite
20 end
10 13
C4
19 0 overwrite
5


01 2
level
number
02
number
start
03
number
body
11
number
end
3
2
C3
C5
C4
04 0
further
down
05 0
back
up
10 0
same
4
6
06
bus
07
bub
09
bue
6
C4
08 *
level
5


01 3
format_or
stop
02
poss
format
06
next
2
$\overline{C6}$
C6
03 0
format
05 0
fullstop
9
04
format
subtree


01 1
format
subtree
02
picture
2 16 8
03
limited
10
fullstop
C8
C8
04 0
nine
07 0
X
05
nine_start
22 15 17
06
nine
subtree
08
X_start
23 15 17
09
X_subtree


01 4
nine
subtree
02
nine
body
C6
03 *
nine_char
04
pieces
16
end
16
C10
C8
05 0
brace_N9
11 0
only_9
06
open_brace
07
num_chars
09
num_end
10
close_brac
e
12
only9start
13
only9body
15
only_nine
end
20h 2 16
21h 2
2
18
2
C7
C7
08 *
num_char
14 *
nine


01 5
X_subtree
02
X_body
C6
03 *
X_char
04
pieces
15
end
16
C10
C9
05 0
brace_NX
11 0
only_X
06
open_brace
07
num_chars
09
num_end
10
close_brac
e
12
onlyXbody
14
onlyXend
20
2
16
21
2
2
2
C7
C7
08 *
num_char
13 *
x

F.3.5 The Code

```
#include <stdio.h>
#include "alutil.h"
#include "alstring.h"
#include "alfile.h"
#include "node.h"
#include "stckbyar.h"
#include "symtab.h"
#include "keywords.h"
#include "token.h"


void main(){

 FILE *infile,*outfile;
 NODE *current,*first_01;
 TOKEN *token,t;
 STACKBYARRAY *stack=create_stack_by_array(50);
 SYMTAB_ENTRY **symtab=create_symbol_table_array(SYMTAB_ARRAY_SIZE);
 int unique_no=1;
 int hashval=0;
 int number_of;
 char *pointer,*pointer2,count_string[20];
 int alfeof=0;


                int g2_001_FORMAT_OR_STOP_A;
                int g3_001_LEVEL_NUMBER_A;
                int g5_001_NINE_SUBTREE_A;
                int g6_001_X_SUBTREE_A;
                int g4_001_FORMAT_SUBTREE_A;
      /*********************************************************************
      /*********************************************************************
      /******************* DRIVING PROCEDURE    B:\SYMCOB ******************
      /*********************************************************************
       PROCEDURE_DIVISION:
      /**/
       C1_001_GENERATE_SYMBOL:
```

```
            goto C1_002_SYMBOL_START;
        C1_002_SYMBOL_START_EX:
            goto C1_003_BEFORE_WORKING;
        C1_003_BEFORE_WORKING_EX:
            goto C1_005_SECTION;
        C1_005_SECTION_EX:
            goto C1_006_WORKING_STORAGE;
        C1_006_WORKING_STORAGE_EX:
            goto C1_021_END;
        C1_021_END_EX:
            goto veryend;
       /**/
        C1_002_SYMBOL_START:
  /*001*/       infile=open_input("b:testout2.dat");
            outfile=open_output("b:symtable.out");
            token=&t;
  /*002*/       if(!feof(infile)) alfeof=get_token(infile,token);
            goto C1_002_SYMBOL_START_EX;
       /**/
        C1_003_BEFORE_WORKING:
        C1_004_JUST_DATA_EX:
            if(!(
  /*C01*/          (token->value.keyword_number==WORKING_STORAGE)
))
                    goto C1_004_JUST_DATA;
            goto C1_003_BEFORE_WORKING_EX;
       /**/
        C1_004_JUST_DATA:
  /*002*/       if(!feof(infile)) alfeof=get_token(infile,token);
            goto C1_004_JUST_DATA_EX;
       /**/
        C1_005_SECTION:
  /*002*/       if(!feof(infile)) alfeof=get_token(infile,token);
  /*002*/       if(!feof(infile)) alfeof=get_token(infile,token);
  /*002*/       if(!feof(infile)) alfeof=get_token(infile,token);
            goto C1_005_SECTION_EX;
       /**/
        C1_006_WORKING_STORAGE:
            goto C1_007_FIRST_01;
```

```
     C1_007_FIRST_01_EX:
         goto C1_012_REST;
     C1_012_REST_EX:
         goto C1_006_WORKING_STORAGE_EX;
    /**/
     C1_007_FIRST_01:
         goto C1_008_NUM;
     C1_008_NUM_EX:
         goto C1_009_VARIABLE_NAME;
     C1_009_VARIABLE_NAME_EX:
         goto C1_010_FORMAT_OR_STOP;
     C1_010_FORMAT_OR_STOP_EX:
         goto C1_011_END_LEVEL;
     C1_011_END_LEVEL_EX:
         goto C1_007_FIRST_01_EX;
    /**/
     C1_008_NUM:
/*003*/        current=create_node();
         current->level=str_to_int(token->value.string);
/*002*/        if(!feof(infile)) alfeof=get_token(infile,token);
         goto C1_008_NUM_EX;
    /**/
     C1_009_VARIABLE_NAME:
/*007*/        NAME(current)=token->value.string;
/*002*/        if(!feof(infile)) alfeof=get_token(infile,token);
/*012*/        hashval=hash(NAME(current))%SYMTAB_ARRAY_SIZE;
         current->own.hash=hashval;
         add_node_to_sym(current,hashval,symtab);
         goto C1_009_VARIABLE_NAME_EX;
    /**/
     C1_010_FORMAT_OR_STOP:
              g2_001_FORMAT_OR_STOP_A=0;
         goto C2_001_FORMAT_OR_STOP;
     C2_001_EXIT01:
         goto C1_010_FORMAT_OR_STOP_EX;
    /**/
     C1_011_END_LEVEL:
/*011*/        PARENT(current)=NULL;
         first_01=current;
```

```
  /*010*/       stackpush(stack,current);
  /*013*/       current->own.unique=unique_no;
            unique_no++;
            goto C1_011_END_LEVEL_EX;
       /**/
        C1_012_REST:
        C1_013_LEVEL_EX:
            if(!(
  /*C02*/          (alfeof==1)
))
                    goto C1_013_LEVEL;
            goto C1_012_REST_EX;
       /**/
        C1_013_LEVEL:
            goto C1_014_LEVEL_NUMBER;
        C1_014_LEVEL_NUMBER_EX:
            goto C1_015_VARIABLE_NAME;
        C1_015_VARIABLE_NAME_EX:
            goto C1_016_FORMAT_OR_STOP;
        C1_016_FORMAT_OR_STOP_EX:
            goto C1_017_END_LEVEL;
        C1_017_END_LEVEL_EX:
            goto C1_013_LEVEL_EX;
       /**/
        C1_014_LEVEL_NUMBER:
                g3_001_LEVEL_NUMBER_A=0;
            goto C3_001_LEVEL_NUMBER;
        C3_001_EXIT01:
            goto C1_014_LEVEL_NUMBER_EX;
       /**/
        C1_015_VARIABLE_NAME:
  /*007*/       NAME(current)=token->value.string;
  /*002*/       if(!feof(infile)) alfeof=get_token(infile,token);
  /*012*/       hashval=hash(NAME(current))%SYMTAB_ARRAY_SIZE;
            current->own.hash=hashval;
            add_node_to_sym(current,hashval,symtab);
            goto C1_015_VARIABLE_NAME_EX;
       /**/
        C1_016_FORMAT_OR_STOP:
```

```
                g2_001_FORMAT_OR_STOP_A=1;
            goto C2_001_FORMAT_OR_STOP;
        C2_001_EXIT02:
            goto C1_016_FORMAT_OR_STOP_EX;
       /**/
        C1_017_END_LEVEL:
            goto C1_018_POSS_OVERWRITE;
        C1_018_POSS_OVERWRITE_EX:
            goto C1_020_END;
        C1_020_END_EX:
            goto C1_017_END_LEVEL_EX;
       /**/
        C1_018_POSS_OVERWRITE:
            if((
  /*C04*/         (current->level==stacktop(stack)->level)
))
                    goto C1_019_OVERWRITE;
        C1_019_OVERWRITE_EX:
            goto C1_018_POSS_OVERWRITE_EX;
       /**/
        C1_019_OVERWRITE:
  /*005*/       stackpop(stack);
            goto C1_019_OVERWRITE_EX;
       /**/
        C1_020_END:
  /*010*/       stackpush(stack,current);
  /*013*/       current->own.unique=unique_no;
            unique_no++;
            goto C1_020_END_EX;
       /**/
        C1_021_END:
  /*014*/       close_file(infile);
            save_nodes(symtab,outfile);
            close_file(outfile);
            goto C1_021_END_EX;
       /**/
        C4_001_FORMAT_SUBTREE:
            goto C4_002_PICTURE;
        C4_002_PICTURE_EX:
```

```
            goto C4_003_LIMITED;
        C4_003_LIMITED_EX:
            goto C4_010_FULLSTOP;
        C4_010_FULLSTOP_EX:
        C4_001_FORMAT_SUBTREE_A:
                switch(g4_001_FORMAT_SUBTREE_A){
                case 0 : goto C4_001_EXIT02;break;
                }
      /**/
        C4_002_PICTURE:
  /*002*/       if(!feof(infile)) alfeof=get_token(infile,token);
  /*016*/       pointer=token->value.string;
  /*008*/       current->info->type=VARIABLE;
            goto C4_002_PICTURE_EX;
      /**/
        C4_003_LIMITED:
            if((
  /*C08*/         (*pointer=='9')
))
                  goto C4_004_NINE;
            if(!(
  /*C08*/         (*pointer=='9')
))
                  goto C4_007_X;
        C4_004_NINE_EX:
        C4_007_X_EX:
            goto C4_003_LIMITED_EX;
      /**/
        C4_004_NINE:
            goto C4_005_NINE_START;
        C4_005_NINE_START_EX:
            goto C4_006_NINE_SUBTREE;
        C4_006_NINE_SUBTREE_EX:
            goto C4_004_NINE_EX;
      /**/
        C4_005_NINE_START:
  /*022*/       current->info->type=INT;
  /*015*/       number_of=0;
  /*017*/       pointer++;
```

```
            goto C4_005_NINE_START_EX;
        /**/
         C4_006_NINE_SUBTREE:
                  g5_001_NINE_SUBTREE_A=0;
            goto C5_001_NINE_SUBTREE;
         C5_001_EXIT01:
            goto C4_006_NINE_SUBTREE_EX;
        /**/
         C4_007_X:
            goto C4_008_X_START;
         C4_008_X_START_EX:
            goto C4_009_X_SUBTREE;
         C4_009_X_SUBTREE_EX:
            goto C4_007_X_EX;
        /**/
         C4_008_X_START:
/*023*/       current->info->type=CHAR;
/*015*/       number_of=0;
/*017*/       pointer++;
            goto C4_008_X_START_EX;
        /**/
         C4_009_X_SUBTREE:
                  g6_001_X_SUBTREE_A=0;
            goto C6_001_X_SUBTREE;
         C6_001_EXIT01:
            goto C4_009_X_SUBTREE_EX;
        /**/
         C4_010_FULLSTOP:
            goto C4_010_FULLSTOP_EX;
        /**/
         C3_001_LEVEL_NUMBER:
            goto C3_002_NUMBER_START;
         C3_002_NUMBER_START_EX:
            goto C3_003_NUMBER_BODY;
         C3_003_NUMBER_BODY_EX:
            goto C3_011_NUMBER_END;
         C3_011_NUMBER_END_EX:
         C3_001_LEVEL_NUMBER_A:
                  switch(g3_001_LEVEL_NUMBER_A){
```

```
                case 0 : goto C3_001_EXIT01;break;
                }
        /**/
         C3_002_NUMBER_START:
  /*003*/       current=create_node();
            current->level=str_to_int(token->value.string);
            goto C3_002_NUMBER_START_EX;
        /**/
         C3_003_NUMBER_BODY:
            if((
  /*C03*/          (current->level > stacktop(stack)->level)
))
                    goto C3_004_FURTHER_DOWN;
            if((
  /*C05*/          (current->level < stacktop(stack)->level)
))
                    goto C3_005_BACK_UP;
            if((
  /*C04*/          (current->level==stacktop(stack)->level)
))
                    goto C3_010_SAME;
         C3_004_FURTHER_DOWN_EX:
         C3_005_BACK_UP_EX:
         C3_010_SAME_EX:
            goto C3_003_NUMBER_BODY_EX;
        /**/
         C3_004_FURTHER_DOWN:
  /*004*/       PARENT(current)=stacktop(stack);
            CHILD(stacktop(stack))=current;
            goto C3_004_FURTHER_DOWN_EX;
        /**/
         C3_005_BACK_UP:
            goto C3_006_BUS;
         C3_006_BUS_EX:
            goto C3_007_BUB;
         C3_007_BUB_EX:
            goto C3_009_BUE;
         C3_009_BUE_EX:
            goto C3_005_BACK_UP_EX;
```

```
      /**/
       C3_006_BUS:
           goto C3_006_BUS_EX;
      /**/
       C3_007_BUB:
       C3_008_LEVEL_EX:
           if(!(
 /*C04*/         (current->level==stacktop(stack)->level)
))
                    goto C3_008_LEVEL;
           goto C3_007_BUB_EX;
      /**/
       C3_008_LEVEL:
 /*005*/       stackpop(stack);
           goto C3_008_LEVEL_EX;
      /**/
       C3_009_BUE:
 /*006*/       SIBLING(stacktop(stack))=current;
           PARENT(current)=PARENT(stacktop(stack));
           goto C3_009_BUE_EX;
      /**/
       C3_010_SAME:
 /*006*/       SIBLING(stacktop(stack))=current;
           PARENT(current)=PARENT(stacktop(stack));
           goto C3_010_SAME_EX;
      /**/
       C3_011_NUMBER_END:
 /*002*/       if(!feof(infile)) alfeof=get_token(infile,token);
           goto C3_011_NUMBER_END_EX;
      /**/
       C2_001_FORMAT_OR_STOP:
           goto C2_002_POSS_FORMAT;
       C2_002_POSS_FORMAT_EX:
           goto C2_006_NEXT;
       C2_006_NEXT_EX:
       C2_001_FORMAT_OR_STOP_A:
                switch(g2_001_FORMAT_OR_STOP_A){
                case 0 : goto C2_001_EXIT01;break;
                case 1 : goto C2_001_EXIT02;break;
```

```
                }
        /**/
         C2_002_POSS_FORMAT:
             if(!(
  /*C06*/          (token->token_type==FULLSTOP)
))
                     goto C2_003_FORMAT;
             if((
  /*C06*/          (token->token_type==FULLSTOP)
))
                     goto C2_005_FULLSTOP;
         C2_003_FORMAT_EX:
         C2_005_FULLSTOP_EX:
             goto C2_002_POSS_FORMAT_EX;
        /**/
         C2_003_FORMAT:
             goto C2_004_FORMAT_SUBTREE;
         C2_004_FORMAT_SUBTREE_EX:
             goto C2_003_FORMAT_EX;
        /**/
         C2_004_FORMAT_SUBTREE:
                 g4_001_FORMAT_SUBTREE_A=0;
             goto C4_001_FORMAT_SUBTREE;
         C4_001_EXIT02:
             goto C2_004_FORMAT_SUBTREE_EX;
        /**/
         C2_005_FULLSTOP:
  /*009*/       current->info->type=QUALIFIER;
             goto C2_005_FULLSTOP_EX;
        /**/
         C2_006_NEXT:
  /*002*/       if(!feof(infile)) alfeof=get_token(infile,token);
             goto C2_006_NEXT_EX;
        /**/
         C5_001_NINE_SUBTREE:
             goto C5_002_NINE_BODY;
         C5_002_NINE_BODY_EX:
         C5_001_NINE_SUBTREE_A:
                 switch(g5_001_NINE_SUBTREE_A){
```

```
                case 0 : goto C5_001_EXIT01;break;
                }
      /**/
       C5_002_NINE_BODY:
       C5_003_NINE_CHAR_EX:
           if(!(
  /*C06*/          (token->token_type==FULLSTOP)
))
                    goto C5_003_NINE_CHAR;
           goto C5_002_NINE_BODY_EX;
      /**/
       C5_003_NINE_CHAR:
           goto C5_004_PIECES;
       C5_004_PIECES_EX:
           goto C5_017_END;
       C5_017_END_EX:
           goto C5_003_NINE_CHAR_EX;
      /**/
       C5_004_PIECES:
           if((
  /*C10*/          (token->token_type==OPEN_BRACE)
))
                    goto C5_005_BRACE_N9;
           if((
  /*C08*/          (*pointer=='9')
))
                    goto C5_012_ONLY_9;
       C5_005_BRACE_N9_EX:
       C5_012_ONLY_9_EX:
           goto C5_004_PIECES_EX;
      /**/
       C5_005_BRACE_N9:
           goto C5_006_OPEN_BRACE;
       C5_006_OPEN_BRACE_EX:
           goto C5_007_N9;
       C5_007_N9_EX:
           goto C5_011_CLOSE_BRAC_E;
       C5_011_CLOSE_BRAC_E_EX:
           goto C5_005_BRACE_N9_EX;
```

```
      /**/
       C5_006_OPEN_BRACE:
  /*020*/      pointer2=count_string;
  /*002*/      if(!feof(infile)) alfeof=get_token(infile,token);
  /*016*/      pointer=token->value.string;
           goto C5_006_OPEN_BRACE_EX;
      /**/
       C5_007_N9:
           goto C5_008_NUM_CHARS;
       C5_008_NUM_CHARS_EX:
           goto C5_010_NUM_END;
       C5_010_NUM_END_EX:
           goto C5_007_N9_EX;
      /**/
       C5_008_NUM_CHARS:
       C5_009_NUM_CHAR_EX:
           if(!(
  /*C07*/         (*pointer=='\0')
))
                    goto C5_009_NUM_CHAR;
           goto C5_008_NUM_CHARS_EX;
      /**/
       C5_009_NUM_CHAR:
  /*019*/      *pointer2=*pointer;
           pointer++;
           pointer2++;
           goto C5_009_NUM_CHAR_EX;
      /**/
       C5_010_NUM_END:
  /*021*/      *pointer2='\0';
           number_of+=str_to_int(count_string);
  /*002*/      if(!feof(infile)) alfeof=get_token(infile,token);
           goto C5_010_NUM_END_EX;
      /**/
       C5_011_CLOSE_BRAC_E:
  /*002*/      if(!feof(infile)) alfeof=get_token(infile,token);
           goto C5_011_CLOSE_BRAC_E_EX;
      /**/
       C5_012_ONLY_9:
```

```
            goto C5_013_ONLY9START;
        C5_013_ONLY9START_EX:
            goto C5_014_ONLY9BODY;
        C5_014_ONLY9BODY_EX:
            goto C5_016_ONLY_NINE_END;
        C5_016_ONLY_NINE_END_EX:
            goto C5_012_ONLY_9_EX;
      /**/
        C5_013_ONLY9START:
  /*018*/      number_of++;
            goto C5_013_ONLY9START_EX;
      /**/
        C5_014_ONLY9BODY:
        C5_015_NINE_EX:
            if(!(
  /*C07*/         (*pointer=='\0')
))
                  goto C5_015_NINE;
            goto C5_014_ONLY9BODY_EX;
      /**/
        C5_015_NINE:
  /*018*/      number_of++;
  /*017*/      pointer++;
  /*026*/      current->info->data_c=NULL;
            current->info->data_n=0;
            goto C5_015_NINE_EX;
      /**/
        C5_016_ONLY_NINE_END:
  /*002*/      if(!feof(infile)) alfeof=get_token(infile,token);
            goto C5_016_ONLY_NINE_END_EX;
      /**/
        C5_017_END:
  /*016*/      pointer=token->value.string;
            goto C5_017_END_EX;
      /**/
        C6_001_X_SUBTREE:
            goto C6_002_X_BODY;
        C6_002_X_BODY_EX:
        C6_001_X_SUBTREE_A:
```

```
                switch(g6_001_X_SUBTREE_A){
                case 0 : goto C6_001_EXIT01;break;
                }
      /**/
       C6_002_X_BODY:
       C6_003_X_CHAR_EX:
           if(!(
  /*C06*/          (token->token_type==FULLSTOP)
))
                    goto C6_003_X_CHAR;
           goto C6_002_X_BODY_EX;
      /**/
       C6_003_X_CHAR:
           goto C6_004_PIECES;
       C6_004_PIECES_EX:
           goto C6_016_END;
       C6_016_END_EX:
           goto C6_003_X_CHAR_EX;
      /**/
       C6_004_PIECES:
           if((
  /*C10*/          (token->token_type==OPEN_BRACE)
))
                    goto C6_005_BRACE_NX;
           if((
  /*C09*/          (*pointer=='X')
))
                    goto C6_012_ONLY_X;
       C6_005_BRACE_NX_EX:
       C6_012_ONLY_X_EX:
           goto C6_004_PIECES_EX;
      /**/
       C6_005_BRACE_NX:
           goto C6_006_OPEN_BRACE;
       C6_006_OPEN_BRACE_EX:
           goto C6_007_NX;
       C6_007_NX_EX:
           goto C6_011_CLOSE_BRAC_E;
       C6_011_CLOSE_BRAC_E_EX:
```

```
            goto C6_005_BRACE_NX_EX;
        /**/
         C6_006_OPEN_BRACE:
  /*020*/       pointer2=count_string;
  /*002*/       if(!feof(infile)) alfeof=get_token(infile,token);
  /*016*/       pointer=token->value.string;
            goto C6_006_OPEN_BRACE_EX;
        /**/
         C6_007_NX:
            goto C6_008_NUM_CHARS;
         C6_008_NUM_CHARS_EX:
            goto C6_010_NUM_END;
         C6_010_NUM_END_EX:
            goto C6_007_NX_EX;
        /**/
         C6_008_NUM_CHARS:
         C6_009_NUM_CHAR_EX:
            if(!(
  /*C07*/         (*pointer=='\0')
))
                    goto C6_009_NUM_CHAR;
            goto C6_008_NUM_CHARS_EX;
        /**/
         C6_009_NUM_CHAR:
  /*019*/       *pointer2=*pointer;
            pointer++;
            pointer2++;
            goto C6_009_NUM_CHAR_EX;
        /**/
         C6_010_NUM_END:
  /*021*/       *pointer2='\0';
            number_of+=str_to_int(count_string);
  /*002*/       if(!feof(infile)) alfeof=get_token(infile,token);
            goto C6_010_NUM_END_EX;
        /**/
         C6_011_CLOSE_BRAC_E:
  /*002*/       if(!feof(infile)) alfeof=get_token(infile,token);
            goto C6_011_CLOSE_BRAC_E_EX;
        /**/
```

```
        C6_012_ONLY_X:
            goto C6_013_ONLYXBODY;
        C6_013_ONLYXBODY_EX:
            goto C6_015_ONLYXEND;
        C6_015_ONLYXEND_EX:
            goto C6_012_ONLY_X_EX;
       /**/
        C6_013_ONLYXBODY:
        C6_014_X_EX:
            if(!(
  /*C07*/         (*pointer=='\0')
))
                    goto C6_014_X;
            goto C6_013_ONLYXBODY_EX;
       /**/
        C6_014_X:
  /*018*/       number_of++;
  /*017*/       pointer++;
  /*024*/       current->info->data_c=(char *)malloc(number_of+1);
            current->info->data_c[0]='\0';
  /*025*/       current->info->data_n=number_of;
            goto C6_014_X_EX;
       /**/
        C6_015_ONLYXEND:
  /*002*/       if(!feof(infile)) alfeof=get_token(infile,token);
            goto C6_015_ONLYXEND_EX;
       /**/
        C6_016_END:
  /*016*/       pointer=token->value.string;
            goto C6_016_END_EX;
       /**/
       /*   Data Analysis Map*/
       /**/
       /*                          ------------- Data Analysis By -------------*
       /**/
       /*                          BOX TYPE                OPERATION  ALLOCATION*
       /*  Tree name: GENERATE-SYMBOL*/
       /**/
       /*                          Leaf      :  10          Operations:  21*/
```

```
/*                                                  Quits      :   0*/
/*                        Selections:   1*/
/*                        Sequences :   5*/
/*                        Iterations:   2*/
/*                        Backtracks:   0*/
/*                        Subtrees  :   3   -->    FORMAT_OR-STOP*/
/*                                                 LEVEL-NUMBER*/
/*                                                 FORMAT_OR-STOP*/
/**/
/*  Tree name: FORMAT-SUBTREE*/
/**/
/*                        Leaf      :   4          Operations:   9*/
/*                                                 Quits      :   0*/
/*                        Selections:   1*/
/*                        Sequences :   3*/
/*                        Iterations:   0*/
/*                        Backtracks:   0*/
/*                        Subtrees  :   2   -->    NINE-SUBTREE*/
/*                                                 X_SUBTREE*/
/**/
/*  Tree name: LEVEL-NUMBER*/
/**/
/*                        Leaf      :   7          Operations:   6*/
/*                                                 Quits      :   0*/
/*                        Selections:   1*/
/*                        Sequences :   2*/
/*                        Iterations:   1*/
/*                        Backtracks:   0*/
/*                        Subtrees  :   0*/
/**/
/*  Tree name: FORMAT_OR-STOP*/
/**/
/*                        Leaf      :   2          Operations:   2*/
/*                                                 Quits      :   0*/
/*                        Selections:   1*/
/*                        Sequences :   2*/
/*                        Iterations:   0*/
/*                        Backtracks:   0*/
/*                        Subtrees  :   1   -->    FORMAT-SUBTREE*/
```

```
        /**/
        /*  Tree name: NINE-SUBTREE*/
        /**/
        /*                          Leaf       :   8          Operations:  13*/
        /*                                                     Quits      :   0*/
        /*                          Selections:   1*/
        /*                          Sequences :   5*/
        /*                          Iterations:   3*/
        /*                          Backtracks:   0*/
        /*                          Subtrees  :   0*/
        /**/
        /*  Tree name: X_SUBTREE*/
        /**/
        /*                          Leaf       :   7          Operations:  13*/
        /*                                                     Quits      :   0*/
        /*                          Selections:   1*/
        /*                          Sequences :   5*/
        /*                          Iterations:   3*/
        /*                          Backtracks:   0*/
        /*                          Subtrees  :   0*/
        /**/
        /**/
        /**/
veryend: ;
}
```

F.4 Tokenising The '.ops'

F.4.1 Operations

```
001 infile=open_input("b:altest.ops");
002 close_file(infile);
003 outfile=open_output("b:opsout.dat");
004 close_file(outfile);
005 currch=getc(infile);
    if(currch!=EOF) nextch=getc(infile);
006 currch=nextch;
    if(nextch!=EOF) nextch=getc(infile);
007 chp=ch;
008 load_symtab(symtab,"b:symtable.out");
    cross_reference(symtab,100);
009 listpointer=root;
010 listpointer=check_pointer(listpointer,currch);
011 dat=get_data(listpointer);
012 write_token(outfile,token);
013 token->token_type=FULLSTOP;token->value.keyword_number=FULLSTOP;
014 token->token_type=KEYWORD;token->value.keyword_number=dat;
015 token->token_type=NON_KEYWORD;token->value.string=ch;
016 token->token_type=NON_NUMERIC_LITERAL;token->value.string=ch;
017 token->token_type=NON_NUMERIC_LITERAL_LENGTH;
    token->value.keyword_number=length_of;
018 dat=str_to_int(ch);
019 *chp='\0';
020 chp++;
021 *chp=currch;
022 token->token_type=OP_NUM;token->value.keyword_number=dat;
023 token->token_type=VARIABLE_NAME;
    token->value.var_values.hash=node->own.hash;
    token->value.var_values.unique=node->own.unique;
024 token->token_type=NUMERIC;token->value.keyword_number=dat;
025 length_of=0;
026 length_of++;
027 node=look_for_by_name(ch,symtab);
028 token=&t;
```

F.4.2 Conditions

```
C01 currch==EOF
C02 currch==' '
C03 currch==EOF || currch=='\n'
C04 currch!=' '
C05 currch=='"' || currch=='\''
C06 currch=='.'
C07 currch!='"' &&
    currch!='\'' &&
    currch!='.' &&
    currch!='\n'
C08 currch=='\''
C09 currch=='"'
C10 currch==' ' || currch=='\n' ||
    currch=='.' || currch==EOF
C11 node==NULL && isdigit(ch[0])
C12 currch==EOF || isdigit(currch)
C13 currch=='\n' || currch==EOF
C14 currch=='\'' && nextch=='\''
C15 currch=='"' && nextch=='"'
C16 currch=='\'' && nextch!='\''
C17 currch=='"' && nextch!='"'
C18 node!=NULL
```

F.4.3 Data Definitions

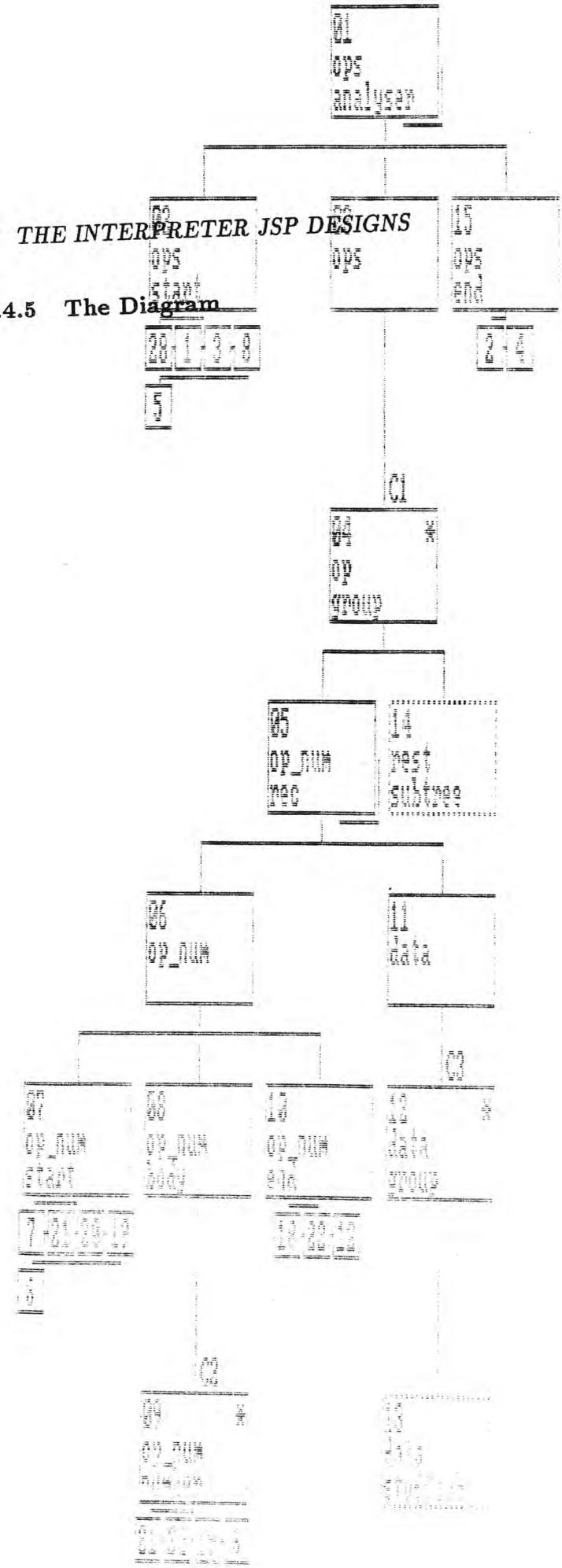
```
#include <stdio.h>
#include "alfile.h"
#include "staglist.h"
#include "node.h"
#include "alutil.h"
#include "symtab.h"
#include "token.h"

void main(){

FILE *infile,*outfile;
char currch,nextch,ch[100],*chp;
```

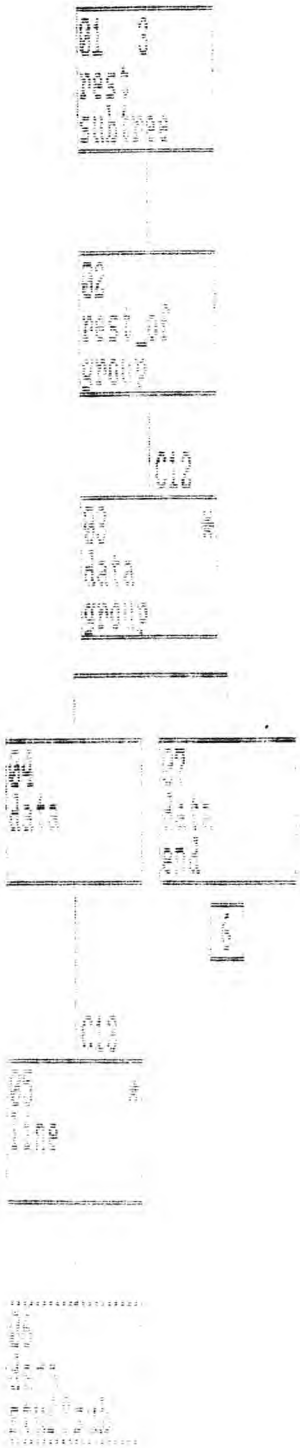
```
staggered_list *listpointer,*root=create_list_with_file("B:keywords");
SYMTAB_ENTRY **symtab=create_symbol_table_array(100);
int dat,length_of;
TOKEN *token,t;
NODE *node;


EOD!
```

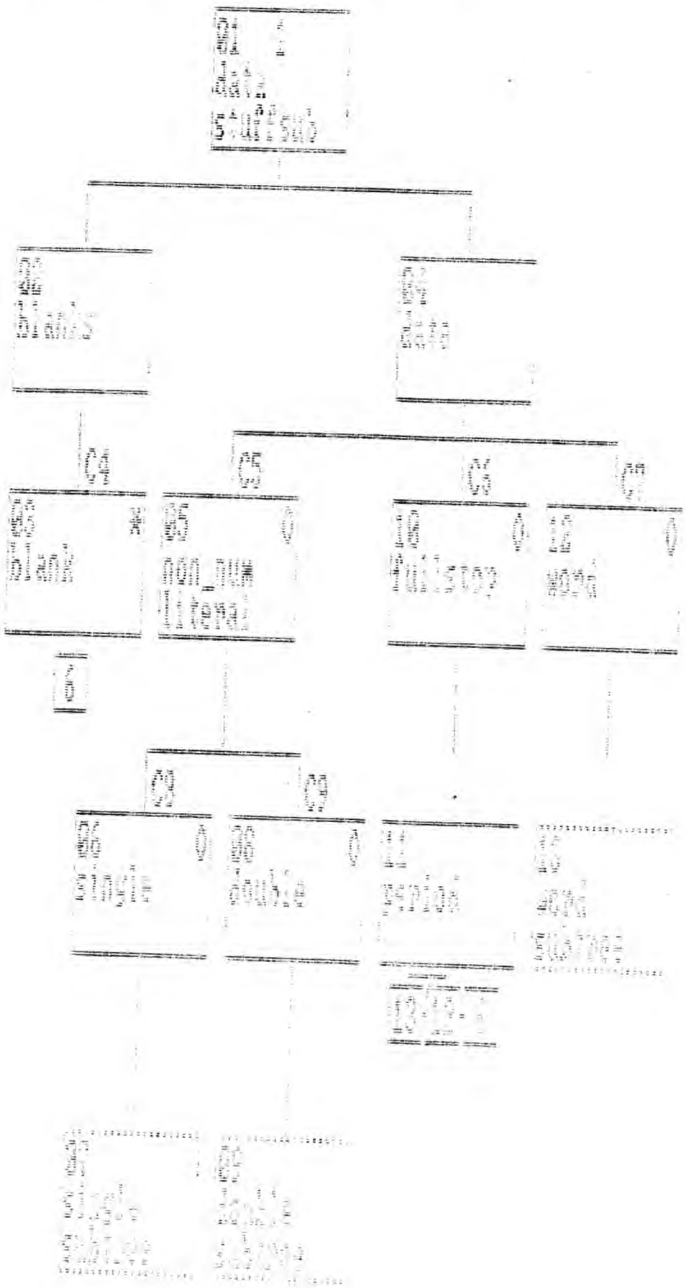
F.4.4 The Quits

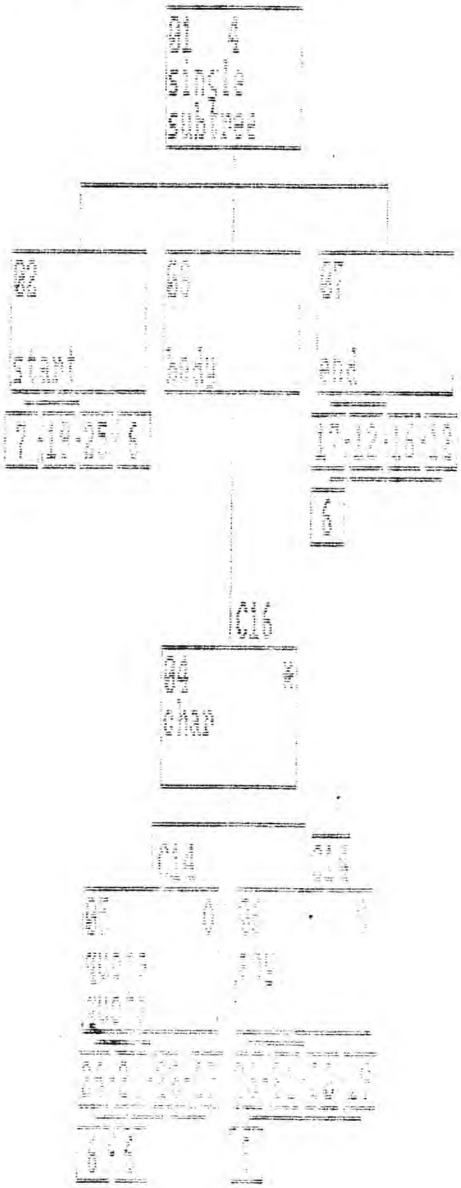
```
Q01 (!isupper(currch)) && currch!='-'
Q02 listpointer==NULL
Q03 dat==NULL
```

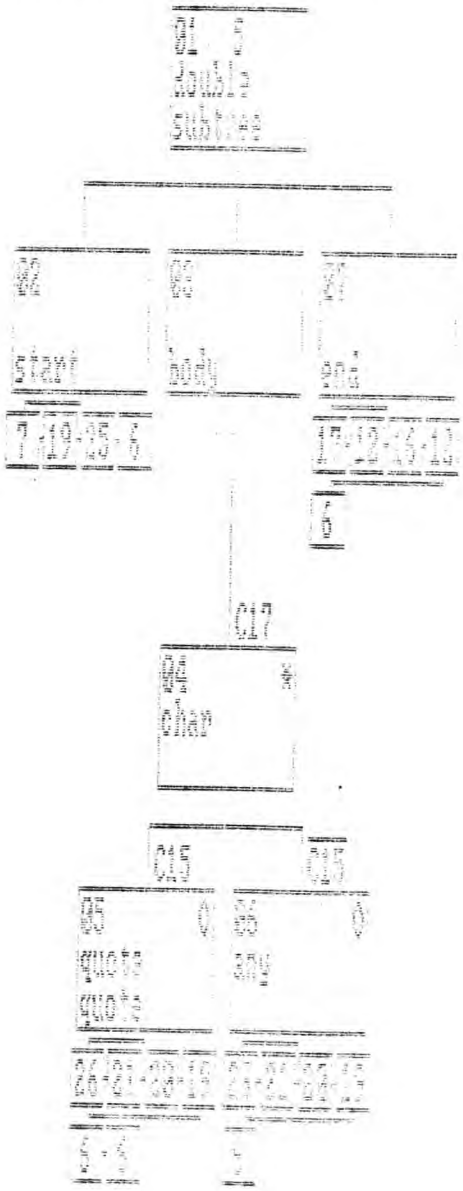
F.4.5 The Diagram

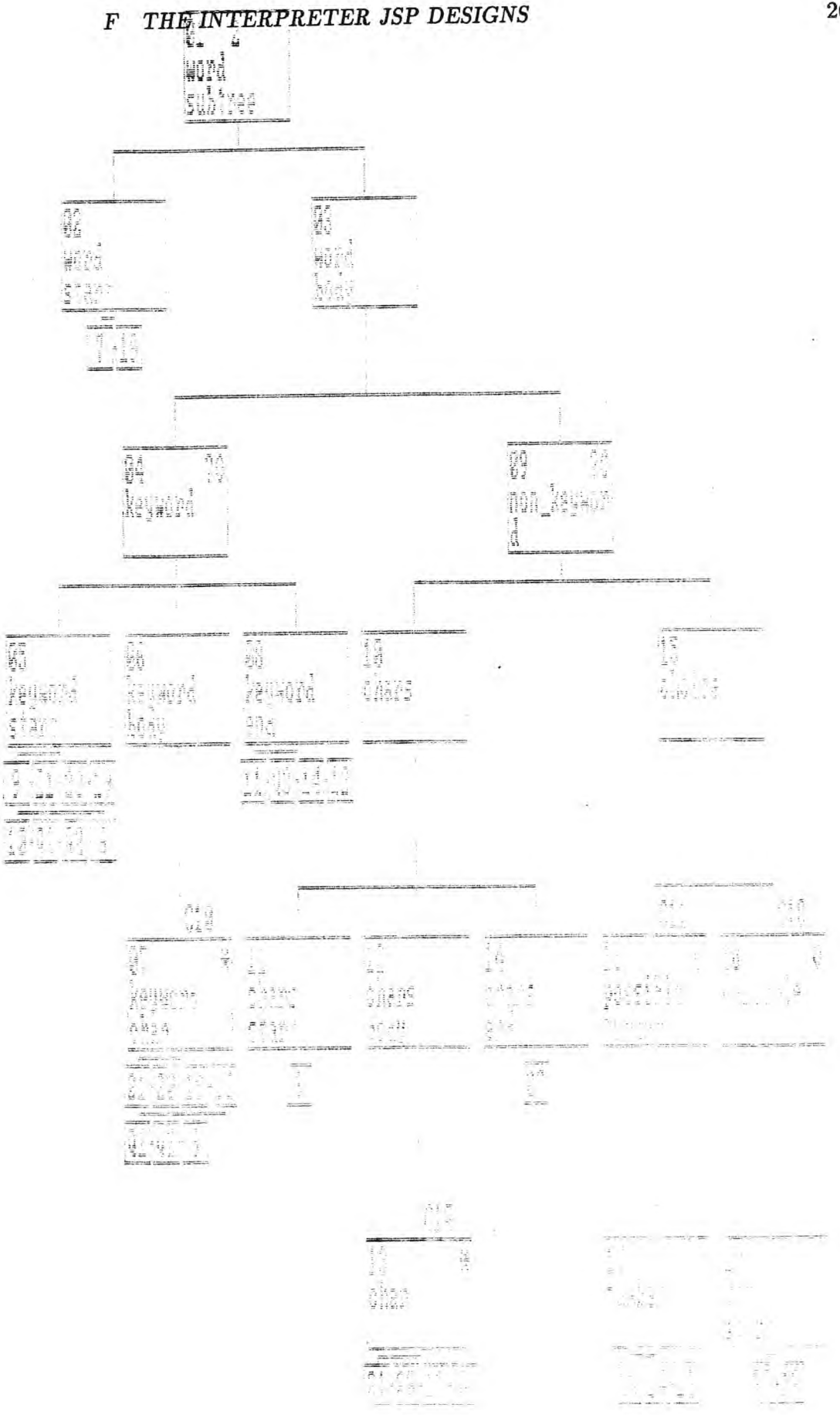



01 3
Test
Subtree
02
Test_of
group
C12
03 *
Data
group
04
Data
05
Data
end
C13
06 *
Type
C10
07
Data
Analysis


01
data
stuff sub
02
blanks
07
rate
C4
C5
C6
C7
03
blank
*
05
non_num
literal
0
10
this stop?
0
12
work
0
C8
C9
06
single
0
08
double
0
11
period
13
add
subtract
14
single
subtract
09
double
subtract




01 S
subtitle
subtitle
02
start
03
body
07
end
17:12:15:10
6
017
04 *
char
015
05 0
quote
quote
015
06 0
any
26:21:20:19
27:24:23:22
5:3
1



F.4.6 The Code

```
#include <stdio.h>
#include "alfile.h"
#include "staglist.h"
#include "node.h"
#include "alutil.h"
#include "symtab.h"
#include "token.h"

void main(){

FILE *infile,*outfile;
char currch,nextch,ch[100],*chp;
staggered_list *listpointer,*root=create_list_with_file("B:keywords");
SYMTAB_ENTRY **symtab=create_symbol_table_array(100);
int dat,length_of;
TOKEN *token,t;
NODE *node;


                int g2_001_DATA_STUFFSUB_A;
                int g3_001_REST_SUBTREE_A;
                int g4_001_SINGLE_SUBTREE_A;
                int g5_001_DOUBLE_SUBTREE_A;
                int g6_001_WORD_SUBTREE_A;
        /*********************************************************************
        /*********************************************************************
        /******************** DRIVING PROCEDURE    B:\OPSANYL *****************
        /*********************************************************************
         PROCEDURE_DIVISION:
        /**/
        /** The ops analyser takes the .ops file*/
        /** and tokenises it for processing by*/
        /** the ops builder*/
        /**/
         C1_001_OPS_ANALYSER:
             goto C1_002_OPS_START;
         C1_002_OPS_START_EX:
```

```
            goto C1_003_OPS;
        C1_003_OPS_EX:
            goto C1_015_OPS_END;
        C1_015_OPS_END_EX:
            goto veryend;
       /**/
        C1_002_OPS_START:
  /*028*/       token=&t;
  /*001*/       infile=open_input("b:altest.ops");
  /*003*/       outfile=open_output("b:opsout.dat");
  /*008*/       load_symtab(symtab,"b:symtable.out");
            cross_reference(symtab,100);
  /*005*/       currch=getc(infile);
            if(currch!=EOF) nextch=getc(infile);
            goto C1_002_OPS_START_EX;
       /**/
        C1_003_OPS:
        C1_004_OP_GROUP_EX:
            if(!(
  /*C01*/         (currch==EOF)
))
                    goto C1_004_OP_GROUP;
            goto C1_003_OPS_EX;
       /**/
        C1_004_OP_GROUP:
            goto C1_005_OP_NUM_REC;
        C1_005_OP_NUM_REC_EX:
            goto C1_014_REST_SUBTREE;
        C1_014_REST_SUBTREE_EX:
            goto C1_004_OP_GROUP_EX;
       /**/
       /** an op consists of the op number*/
       /** followed by the rest of it*/
       /**/
        C1_005_OP_NUM_REC:
            goto C1_006_OP_NUM;
        C1_006_OP_NUM_EX:
            goto C1_011_DATA;
        C1_011_DATA_EX:
```

```
            goto C1_005_OP_NUM_REC_EX;
        /**/
         C1_006_OP_NUM:
            goto C1_007_OP_NUM_START;
         C1_007_OP_NUM_START_EX:
            goto C1_008_OP_NUM_BODY;
         C1_008_OP_NUM_BODY_EX:
            goto C1_010_OP_NUM_END;
         C1_010_OP_NUM_END_EX:
            goto C1_006_OP_NUM_EX;
        /**/
         C1_007_OP_NUM_START:
  /*007*/       chp=ch;
  /*021*/       *chp=currch;
  /*020*/       chp++;
  /*019*/       *chp='\0';
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C1_007_OP_NUM_START_EX;
        /**/
         C1_008_OP_NUM_BODY:
         C1_009_OP_NUM_NUMBER_EX:
            if(!(
  /*C02*/         (currch==' ')
))
                    goto C1_009_OP_NUM_NUMBER;
            goto C1_008_OP_NUM_BODY_EX;
        /**/
         C1_009_OP_NUM_NUMBER:
  /*021*/       *chp=currch;
  /*020*/       chp++;
  /*019*/       *chp='\0';
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C1_009_OP_NUM_NUMBER_EX;
        /**/
         C1_010_OP_NUM_END:
  /*018*/       dat=str_to_int(ch);
  /*022*/       token->token_type=OP_NUM;token->value.keyword_number=dat;
```

```
 /*012*/      write_token(outfile,token);
            goto C1_010_OP_NUM_END_EX;
        /**/
         C1_011_DATA:
         C1_012_DATA_GROUP_EX:
            if(!(
 /*C03*/          (currch==EOF || currch=='\n')
))
                    goto C1_012_DATA_GROUP;
            goto C1_011_DATA_EX;
        /**/
         C1_012_DATA_GROUP:
            goto C1_013_DATA_STUFFSUB;
         C1_013_DATA_STUFFSUB_EX:
            goto C1_012_DATA_GROUP_EX;
        /**/
         C1_013_DATA_STUFFSUB:
                g2_001_DATA_STUFFSUB_A=0;
            goto C2_001_DATA_STUFFSUB;
         C2_001_EXIT01:
            goto C1_013_DATA_STUFFSUB_EX;
        /**/
         C1_014_REST_SUBTREE:
                g3_001_REST_SUBTREE_A=0;
            goto C3_001_REST_SUBTREE;
         C3_001_EXIT01:
            goto C1_014_REST_SUBTREE_EX;
        /**/
         C1_015_OPS_END:
 /*002*/      close_file(infile);
 /*004*/      close_file(outfile);
            goto C1_015_OPS_END_EX;
        /**/
         C2_001_DATA_STUFFSUB:
            goto C2_002_BLANKS;
         C2_002_BLANKS_EX:
            goto C2_004_DATA;
         C2_004_DATA_EX:
         C2_001_DATA_STUFFSUB_A:
```

```
                switch(g2_001_DATA_STUFFSUB_A){
                case 0 : goto C2_001_EXIT01;break;
                case 1 : goto C2_001_EXIT03;break;
                }
        /**/
         C2_002_BLANKS:
         C2_003_BLANK_EX:
             if(!(
  /*C04*/          (currch!=' ')
))
                    goto C2_003_BLANK;
             goto C2_002_BLANKS_EX;
        /**/
         C2_003_BLANK:
  /*006*/       currch=nextch;
             if(nextch!=EOF) nextch=getc(infile);
             goto C2_003_BLANK_EX;
        /**/
         C2_004_DATA:
             if((
  /*C05*/          (currch=='"' || currch=='\'')
))
                    goto C2_005_NON_NUM_LITERAL;
             if((
  /*C06*/          (currch=='.')
))
                    goto C2_010_FULLSTOP;
             if((
  /*C07*/          (currch!='"' &&
                 currch!='\'' &&
                 currch!='.' &&
                 currch!='\n')
))
                    goto C2_012_WORD;
         C2_005_NON_NUM_LITERAL_EX:
         C2_010_FULLSTOP_EX:
         C2_012_WORD_EX:
             goto C2_004_DATA_EX;
        /**/
```

```
      C2_005_NON_NUM_LITERAL:
          if((
  /*C08*/         (currch=='\'')
))
                    goto C2_006_SINGLE;
          if((
  /*C09*/         (currch=='"')
))
                    goto C2_008_DOUBLE;
      C2_006_SINGLE_EX:
      C2_008_DOUBLE_EX:
          goto C2_005_NON_NUM_LITERAL_EX;
     /**/
      C2_006_SINGLE:
          goto C2_007_SINGLE_SUBTREE;
      C2_007_SINGLE_SUBTREE_EX:
          goto C2_006_SINGLE_EX;
     /**/
      C2_007_SINGLE_SUBTREE:
                g4_001_SINGLE_SUBTREE_A=0;
          goto C4_001_SINGLE_SUBTREE;
      C4_001_EXIT01:
          goto C2_007_SINGLE_SUBTREE_EX;
     /**/
      C2_008_DOUBLE:
          goto C2_009_DOUBLE_SUBTREE;
      C2_009_DOUBLE_SUBTREE_EX:
          goto C2_008_DOUBLE_EX;
     /**/
      C2_009_DOUBLE_SUBTREE:
                g5_001_DOUBLE_SUBTREE_A=0;
          goto C5_001_DOUBLE_SUBTREE;
      C5_001_EXIT01:
          goto C2_009_DOUBLE_SUBTREE_EX;
     /**/
      C2_010_FULLSTOP:
          goto C2_011_PERIOD;
      C2_011_PERIOD_EX:
          goto C2_010_FULLSTOP_EX;
```

```
    /**/
     C2_011_PERIOD:
/*013*/      token->token_type=FULLSTOP;token->value.keyword_number=FULLSTOP
/*012*/      write_token(outfile,token);
/*006*/      currch=nextch;
         if(nextch!=EOF) nextch=getc(infile);
         goto C2_011_PERIOD_EX;
    /**/
     C2_012_WORD:
         goto C2_013_WORD_SUBTREE;
     C2_013_WORD_SUBTREE_EX:
         goto C2_012_WORD_EX;
    /**/
     C2_013_WORD_SUBTREE:
              g6_001_WORD_SUBTREE_A=0;
         goto C6_001_WORD_SUBTREE;
     C6_001_EXIT01:
         goto C2_013_WORD_SUBTREE_EX;
    /**/
     C6_001_WORD_SUBTREE:
         goto C6_002_WORD_START;
     C6_002_WORD_START_EX:
         goto C6_003_WORD_BODY;
     C6_003_WORD_BODY_EX:
     C6_001_WORD_SUBTREE_A:
              switch(g6_001_WORD_SUBTREE_A){
              case 0 : goto C6_001_EXIT01;break;
              }
    /**/
     C6_002_WORD_START:
/*007*/      chp=ch;
/*019*/      *chp='\0';
         goto C6_002_WORD_START_EX;
    /**/
     C6_003_WORD_BODY:
         goto C6_004_KEYWORD;
     C6_004_KEYWORD_EX:
     C6_009_NON_KEYWOR_D_EX:
         goto C6_003_WORD_BODY_EX;
```

```
      /**/
       C6_004_KEYWORD:
           goto C6_005_KEYWORD_START;
       C6_005_KEYWORD_START_EX:
           goto C6_006_KEYWORD_BODY;
       C6_006_KEYWORD_BODY_EX:
           goto C6_008_KEYWORD_END;
       C6_008_KEYWORD_END_EX:
           goto C6_004_KEYWORD_EX;
      /**/
       C6_005_KEYWORD_START:
  /*009*/      listpointer=root;
  /*021*/      *chp=currch;
  /*020*/      chp++;
  /*019*/      *chp='\0';
  /*010*/      listpointer=check_pointer(listpointer,currch);
           if((
  /*Q01*/         (!isupper(currch)) && currch!='-'
))
                 goto C6_009_NON_KEYWOR_D;
           if((
  /*Q02*/         listpointer==NULL
))
                 goto C6_009_NON_KEYWOR_D;
  /*006*/      currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
           goto C6_005_KEYWORD_START_EX;
      /**/
       C6_006_KEYWORD_BODY:
       C6_007_KEYWORD_CHAR_EX:
           if(!(
  /*C10*/         (currch==' ' || currch=='\n' ||
              currch=='.' || currch==EOF)
))
                 goto C6_007_KEYWORD_CHAR;
           goto C6_006_KEYWORD_BODY_EX;
      /**/
       C6_007_KEYWORD_CHAR:
  /*021*/      *chp=currch;
```

```
  /*020*/      chp++;
  /*019*/      *chp='\0';
  /*010*/      listpointer=check_pointer(listpointer,currch);
           if((
  /*Q01*/         (!isupper(currch)) && currch!='-'
))
                 goto C6_009_NON_KEYWOR_D;
           if((
  /*Q02*/         listpointer==NULL
))
                 goto C6_009_NON_KEYWOR_D;
  /*006*/      currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
           goto C6_007_KEYWORD_CHAR_EX;
      /**/
       C6_008_KEYWORD_END:
  /*011*/      dat=get_data(listpointer);
           if((
  /*Q03*/         dat==NULL
))
                 goto C6_009_NON_KEYWOR_D;
  /*014*/      token->token_type=KEYWORD;token->value.keyword_number=dat;
  /*012*/      write_token(outfile,token);
           goto C6_008_KEYWORD_END_EX;
      /**/
       C6_009_NON_KEYWOR_D:
           goto C6_010_CHARS;
       C6_010_CHARS_EX:
           goto C6_015_CHOICE;
       C6_015_CHOICE_EX:
           goto C6_009_NON_KEYWOR_D_EX;
      /**/
       C6_010_CHARS:
           goto C6_011_CHARS_START;
       C6_011_CHARS_START_EX:
           goto C6_012_CHARS_BODY;
       C6_012_CHARS_BODY_EX:
           goto C6_014_CHARS_END;
       C6_014_CHARS_END_EX:
```

```
            goto C6_010_CHARS_EX;
        /**/
         C6_011_CHARS_START:
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C6_011_CHARS_START_EX;
        /**/
         C6_012_CHARS_BODY:
         C6_013_CHAR_EX:
            if(!(
  /*C10*/         (currch==' ' || currch=='\n' ||
                currch=='.' || currch==EOF)
))
                    goto C6_013_CHAR;
            goto C6_012_CHARS_BODY_EX;
        /**/
         C6_013_CHAR:
  /*021*/       *chp=currch;
  /*020*/       chp++;
  /*019*/       *chp='\0';
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C6_013_CHAR_EX;
        /**/
         C6_014_CHARS_END:
  /*027*/       node=look_for_by_name(ch,symtab);
            goto C6_014_CHARS_END_EX;
        /**/
         C6_015_CHOICE:
            if((
  /*C11*/         (node==NULL && isdigit(ch[0]))
))
                    goto C6_016_POSSIBLE_NUMBER;
            if((
  /*C18*/         (node!=NULL)
))
                    goto C6_018_VARIABLE;
        C6_016_POSSIBLE_NUMBER_EX:
        C6_018_VARIABLE_EX:
```

```
            goto C6_015_CHOICE_EX;
        /**/
         C6_016_POSSIBLE_NUMBER:
            goto C6_017_NUMBER;
         C6_017_NUMBER_EX:
            goto C6_016_POSSIBLE_NUMBER_EX;
        /**/
         C6_017_NUMBER:
  /*018*/       dat=str_to_int(ch);
  /*024*/       token->token_type=NUMERIC;token->value.keyword_number=dat;
  /*012*/       write_token(outfile,token);
            goto C6_017_NUMBER_EX;
        /**/
         C6_018_VARIABLE:
            goto C6_019_VAR_START;
         C6_019_VAR_START_EX:
            goto C6_018_VARIABLE_EX;
        /**/
         C6_019_VAR_START:
  /*023*/       token->token_type=VARIABLE_NAME;token->value.var_values.hash=nc
            token->value.var_values.unique=node->own.unique;
  /*012*/       write_token(outfile,token);
            goto C6_019_VAR_START_EX;
        /**/
         C3_001_REST_SUBTREE:
            goto C3_002_REST_OF_GROUP;
         C3_002_REST_OF_GROUP_EX:
         C3_001_REST_SUBTREE_A:
                switch(g3_001_REST_SUBTREE_A){
                case 0 : goto C3_001_EXIT01;break;
                }
        /**/
         C3_002_REST_OF_GROUP:
         C3_003_DATA_GROUP_EX:
            if(!(
  /*C12*/          (currch==EOF || isdigit(currch))
))
                  goto C3_003_DATA_GROUP;
            goto C3_002_REST_OF_GROUP_EX;
```

```
      /**/
       C3_003_DATA_GROUP:
           goto C3_004_DATA;
       C3_004_DATA_EX:
           goto C3_007_DATA_END;
       C3_007_DATA_END_EX:
           goto C3_003_DATA_GROUP_EX;
      /**/
       C3_004_DATA:
       C3_005_LINE_EX:
           if(!(
  /*C13*/         (currch=='\n' || currch==EOF)
))
                    goto C3_005_LINE;
           goto C3_004_DATA_EX;
      /**/
       C3_005_LINE:
           goto C3_006_DATA_STUFFSUB;
       C3_006_DATA_STUFFSUB_EX:
           goto C3_005_LINE_EX;
      /**/
       C3_006_DATA_STUFFSUB:
                g2_001_DATA_STUFFSUB_A=1;
           goto C2_001_DATA_STUFFSUB;
       C2_001_EXIT03:
           goto C3_006_DATA_STUFFSUB_EX;
      /**/
       C3_007_DATA_END:
  /*006*/       currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
           goto C3_007_DATA_END_EX;
      /**/
       C4_001_SINGLE_SUBTREE:
           goto C4_002__START;
       C4_002__START_EX:
           goto C4_003__BODY;
       C4_003__BODY_EX:
           goto C4_007__END;
       C4_007__END_EX:
```

```
        C4_001_SINGLE_SUBTREE_A:
                switch(g4_001_SINGLE_SUBTREE_A){
                case 0 : goto C4_001_EXIT01;break;
                }
      /**/
        C4_002__START:
  /*007*/       chp=ch;
  /*019*/       *chp='\0';
  /*025*/       length_of=0;
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C4_002__START_EX;
      /**/
        C4_003__BODY:
        C4_004_CHAR_EX:
            if(!(
  /*C16*/         (currch=='\'' && nextch!='\'')
))
                  goto C4_004_CHAR;
            goto C4_003__BODY_EX;
      /**/
        C4_004_CHAR:
            if((
  /*C14*/         (currch=='\'' && nextch=='\'')
))
                  goto C4_005_QUOTE_QUOTE;
            if(!(
  /*C14*/         (currch=='\'' && nextch=='\'')
))
                  goto C4_006_ANY;
        C4_005_QUOTE_QUOTE_EX:
        C4_006_ANY_EX:
            goto C4_004_CHAR_EX;
      /**/
        C4_005_QUOTE_QUOTE:
  /*026*/       length_of++;
  /*021*/       *chp=currch;
  /*020*/       chp++;
  /*019*/       *chp='\0';
```

```
/*006*/        currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
/*006*/        currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
           goto C4_005_QUOTE_QUOTE_EX;
     /**/
      C4_006_ANY:
/*026*/        length_of++;
/*021*/        *chp=currch;
/*020*/        chp++;
/*019*/        *chp='\0';
/*006*/        currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
           goto C4_006_ANY_EX;
     /**/
      C4_007__END:
/*017*/        token->token_type=NON_NUMERIC_LITERAL_LENGTH;token->value.keywc
                                                                            len
/*012*/        write_token(outfile,token);
/*016*/        token->token_type=NON_NUMERIC_LITERAL;token->value.string=ch;
/*012*/        write_token(outfile,token);
/*006*/        currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
           goto C4_007__END_EX;
     /**/
      C5_001_DOUBLE_SUBTREE:
           goto C5_002__START;
      C5_002__START_EX:
           goto C5_003__BODY;
      C5_003__BODY_EX:
           goto C5_007__END;
      C5_007__END_EX:
      C5_001_DOUBLE_SUBTREE_A:
                switch(g5_001_DOUBLE_SUBTREE_A){
                case 0 : goto C5_001_EXIT01;break;
                }
     /**/
      C5_002__START:
/*007*/        chp=ch;
```

```
  /*019*/       *chp='\0';
  /*025*/       length_of=0;
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C5_002__START_EX;
      /**/
       C5_003__BODY:
       C5_004_CHAR_EX:
            if(!(
  /*C17*/          (currch=='"' && nextch!='"')
))
                    goto C5_004_CHAR;
            goto C5_003__BODY_EX;
      /**/
       C5_004_CHAR:
            if((
  /*C15*/          (currch=='"' && nextch=='"')
))
                    goto C5_005_QUOTE_QUOTE;
            if(!(
  /*C15*/          (currch=='"' && nextch=='"')
))
                    goto C5_006_ANY;
       C5_005_QUOTE_QUOTE_EX:
       C5_006_ANY_EX:
            goto C5_004_CHAR_EX;
      /**/
       C5_005_QUOTE_QUOTE:
  /*026*/       length_of++;
  /*021*/       *chp=currch;
  /*020*/       chp++;
  /*019*/       *chp='\0';
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C5_005_QUOTE_QUOTE_EX;
      /**/
       C5_006_ANY:
```

```
/*026*/      length_of++;
/*021*/      *chp=currch;
/*020*/      chp++;
/*019*/      *chp='\0';
/*006*/      currch=nextch;
         if(nextch!=EOF) nextch=getc(infile);
         goto C5_006_ANY_EX;
    /**/
     C5_007__END:
/*017*/      token->token_type=NON_NUMERIC_LITERAL_LENGTH;token->value.keywc
                                                                          len
/*012*/      write_token(outfile,token);
/*016*/      token->token_type=NON_NUMERIC_LITERAL;token->value.string=ch;
/*012*/      write_token(outfile,token);
/*006*/      currch=nextch;
         if(nextch!=EOF) nextch=getc(infile);
         goto C5_007__END_EX;
    /**/
    /*   Data Analysis Map*/
    /**/
    /*                          ------------- Data Analysis By -------------*
    /**/
    /*                          BOX TYPE                OPERATION  ALLOCATION*
    /*  Tree name: OPS-ANALYSER*/
    /**/
    /*                          Leaf      :   5          Operations:  19*/
    /*                                                   Quits     :   0*/
    /*                          Selections:   0*/
    /*                          Sequences :   5*/
    /*                          Iterations:   3*/
    /*                          Backtracks:   0*/
    /*                          Subtrees  :   2   -->    DATA-STUFFSUB*/
    /*                                                   REST-SUBTREE*/
    /**/
    /*  Tree name: DATA-STUFFSUB*/
    /**/
    /*                          Leaf      :   2          Operations:   4*/
    /*                                                   Quits     :   0*/
    /*                          Selections:   2*/
```

```
/*                          Sequences :   5*/
/*                          Iterations:   1*/
/*                          Backtracks:   0*/
/*                          Subtrees  :   3   -->    SINGLE-SUBTREE*/
/*                                                   DOUBLE-SUBTREE*/
/*                                                   WORD-SUBTREE*/
/**/
/*  Tree name: WORD-SUBTREE*/
/**/
/*                          Leaf      :   9          Operations:  27*/
/*                                                   Quits      :   5*/
/*                          Selections:   1*/
/*                          Sequences :   6*/
/*                          Iterations:   2*/
/*                          Backtracks:   1*/
/*                          Subtrees  :   0*/
/**/
/*  Tree name: REST-SUBTREE*/
/**/
/*                          Leaf      :   1          Operations:   1*/
/*                                                   Quits      :   0*/
/*                          Selections:   0*/
/*                          Sequences :   3*/
/*                          Iterations:   2*/
/*                          Backtracks:   0*/
/*                          Subtrees  :   1   -->    DATA-STUFFSUB*/
/**/
/*  Tree name: SINGLE-SUBTREE*/
/**/
/*                          Leaf      :   4          Operations:  20*/
/*                                                   Quits      :   0*/
/*                          Selections:   1*/
/*                          Sequences :   1*/
/*                          Iterations:   1*/
/*                          Backtracks:   0*/
/*                          Subtrees  :   0*/
/**/
/*  Tree name: DOUBLE-SUBTREE*/
/**/
```

```
      /*                          Leaf       :   4          Operations:  20*/
      /*                                                    Quits      :   0*/
      /*                          Selections:   1*/
      /*                          Sequences :   1*/
      /*                          Iterations:   1*/
      /*                          Backtracks:   0*/
      /*                          Subtrees  :   0*/
      /**/
      /**/
      /**/
veryend: ;
}
```

F.5 Parsing The '.ops'

F.5.1 Operations

```
001 infile=open_input("b:opsout.dat");
002 close_file(infile);
003 token=&t;
004 eof=get_token(infile,token);
005 load_symtab(symtab,"b:symtable.out");
    cross_reference(symtab,100);
006 node=look_for(token->value.var_values.unique,
                  token->value.var_values.hash,
                  symtab);
007 op_list_list=create_op_list_list();
008 op_list=create_op_list();
009 op_array[token->value.keyword_number]=op_list_list;
010 op_list_list->op=op_list;
011 save_ops_array("b:opsarray.out",op_array);
012 op_list->type=OP_VAR;
    op_list->value.var_name=node;
013 op_list->type=OP_VERB;
    op_list->value.keyword_number=token->value.keyword_number;
014 op_list->type=OP_STRING;
    op_list->value.non_numeric_literal=token->value.string;
015 op_list->type=OP_NUM_LIT;
    op_list->value.numeric_literal=token->value.keyword_number;
016 op_list->next=create_op_list();
    op_list=op_list->next;
017 op_list_list->next=create_op_list_list();
    op_list_list=op_list_list->next;
```

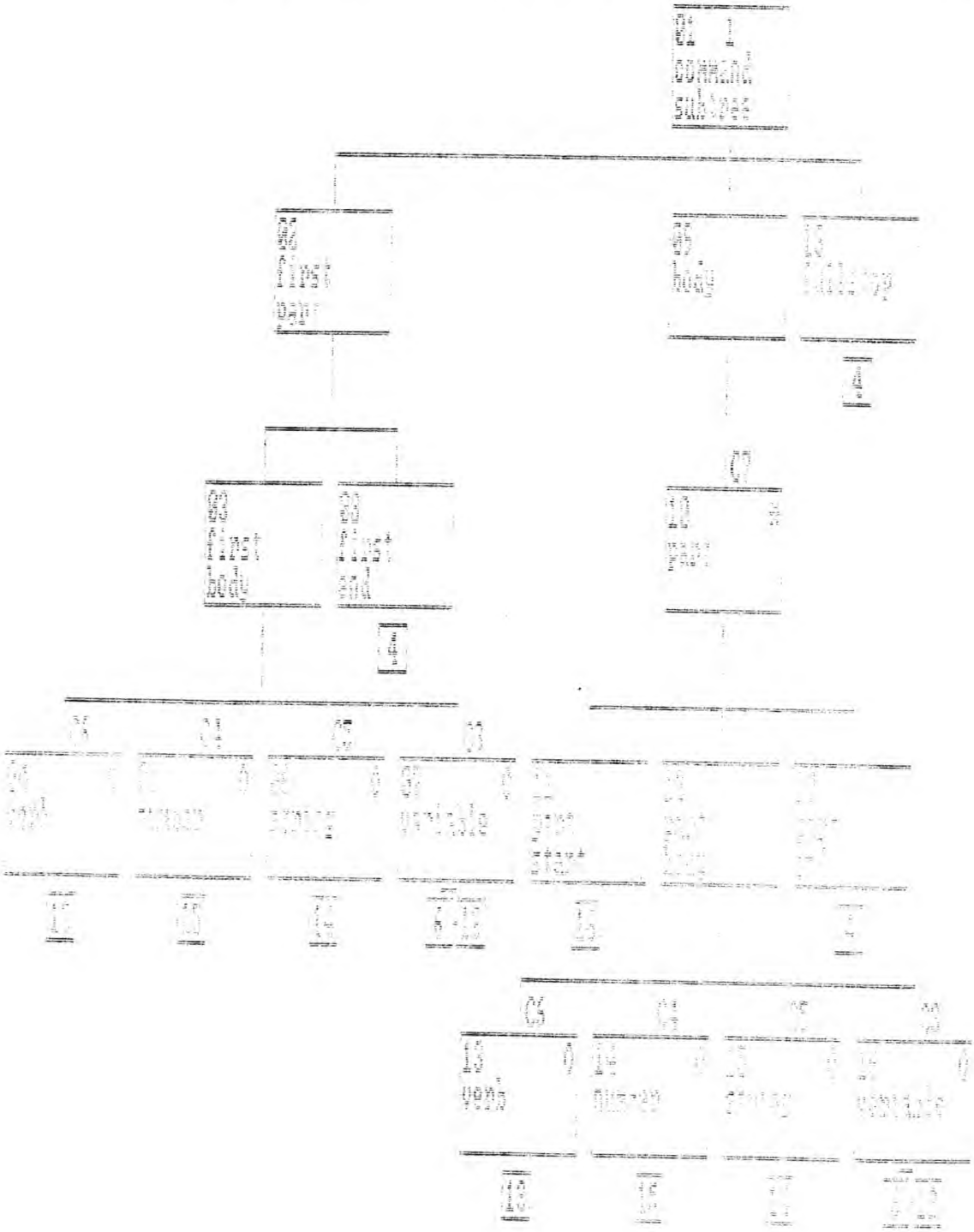
F.5.2 Conditions

```
C01 eof==1
C02 eof==1 || token->token_type==OP_NUM
C03 token->token_type==VARIABLE_NAME
C04 token->token_type==NUMERIC
C05 token->token_type==NON_NUMERIC_LITERAL
C06 token->token_type==KEYWORD
C07 token->token_type==FULLSTOP
```

F.5.3 Data Definitions

```
#include <stdio.h>
#include "alfile.h"
#include "node.h"
#include "alutil.h"
#include "symtab.h"
#include "keywords.h"
#include "token.h"
#include "ops.h"

void main(){

FILE *infile;
SYMTAB_ENTRY **symtab=create_symbol_table_array(100);
int eof;
TOKEN *token,t;
NODE *node;
OP_LIST_LIST **op_array=create_op_array();
OP_LIST_LIST *op_list_list;
OP_LIST *op_list;


EOD!
```

F.5.4 The Diagram



F.5.5 The Code

```
#include <stdio.h>
#include "alfile.h"
#include "node.h"
#include "alutil.h"
#include "symtab.h"
#include "keywords.h"
#include "token.h"
#include "ops.h"

void main(){

FILE *infile;
SYMTAB_ENTRY **symtab=create_symbol_table_array(100);
int eof;
TOKEN *token,t;
NODE *node;
OP_LIST_LIST **op_array=create_op_array();
OP_LIST_LIST *op_list_list;
OP_LIST *op_list;


            int g2_001_COMMAND_SUBTREE_A;
     /*********************************************************************
     /*********************************************************************
     /******************** DRIVING PROCEDURE    B:\OPSBLD ******************
     /*********************************************************************
      PROCEDURE_DIVISION:
     /**/
      C1_001_OPS_BUILDER:
          goto C1_002_OPS_BUILD_START;
      C1_002_OPS_BUILD_START_EX:
          goto C1_003_OPS_BUILD_BODY;
      C1_003_OPS_BUILD_BODY_EX:
          goto C1_013_OPS_BUILD_END;
      C1_013_OPS_BUILD_END_EX:
          goto veryend;
     /**/
```

```
      C1_002_OPS_BUILD_START:
/*001*/       infile=open_input("b:opsout.dat");
/*003*/       token=&t;
/*005*/       load_symtab(symtab,"b:symtable.out");
          cross_reference(symtab,100);
/*004*/       eof=get_token(infile,token);
          goto C1_002_OPS_BUILD_START_EX;
     /**/
      C1_003_OPS_BUILD_BODY:
      C1_004_OP_EX:
          if(!(
/*C01*/         (eof==1)
))
                  goto C1_004_OP;
          goto C1_003_OPS_BUILD_BODY_EX;
     /**/
      C1_004_OP:
          goto C1_005_OPSTART;
      C1_005_OPSTART_EX:
          goto C1_006_FIRST_COMMAND;
      C1_006_FIRST_COMMAND_EX:
          goto C1_008_REST;
      C1_008_REST_EX:
          goto C1_004_OP_EX;
     /**/
      C1_005_OPSTART:
/*007*/       op_list_list=create_op_list_list();
/*009*/       op_array[token->value.keyword_number]=op_list_list;
/*008*/       op_list=create_op_list();
/*010*/       op_list_list->op=op_list;
/*004*/       eof=get_token(infile,token);
          goto C1_005_OPSTART_EX;
     /**/
      C1_006_FIRST_COMMAND:
          goto C1_007_COMMAND_SUBTREE;
      C1_007_COMMAND_SUBTREE_EX:
          goto C1_006_FIRST_COMMAND_EX;
     /**/
      C1_007_COMMAND_SUBTREE:
```

```
                g2_001_COMMAND_SUBTREE_A=0;
            goto C2_001_COMMAND_SUBTREE;
        C2_001_EXIT01:
            goto C1_007_COMMAND_SUBTREE_EX;
       /**/
        C1_008_REST:
        C1_009_COMMAND_EX:
            if(!(
  /*C02*/          (eof==1 || token->token_type==OP_NUM)
))
                    goto C1_009_COMMAND;
            goto C1_008_REST_EX;
       /**/
        C1_009_COMMAND:
            goto C1_010_COMMAND_START;
        C1_010_COMMAND_START_EX:
            goto C1_011_COMMAND_BODY;
        C1_011_COMMAND_BODY_EX:
            goto C1_009_COMMAND_EX;
       /**/
        C1_010_COMMAND_START:
  /*017*/       op_list_list->next=create_op_list_list();
            op_list_list=op_list_list->next;
  /*008*/       op_list=create_op_list();
  /*010*/       op_list_list->op=op_list;
            goto C1_010_COMMAND_START_EX;
       /**/
        C1_011_COMMAND_BODY:
            goto C1_012_COMMAND_SUBTREE;
        C1_012_COMMAND_SUBTREE_EX:
            goto C1_011_COMMAND_BODY_EX;
       /**/
        C1_012_COMMAND_SUBTREE:
                g2_001_COMMAND_SUBTREE_A=1;
            goto C2_001_COMMAND_SUBTREE;
        C2_001_EXIT02:
            goto C1_012_COMMAND_SUBTREE_EX;
       /**/
        C1_013_OPS_BUILD_END:
```

```
  /*002*/      close_file(infile);
  /*011*/      save_ops_array("b:opsarray.out",op_array);
           goto C1_013_OPS_BUILD_END_EX;
      /**/
       C2_001_COMMAND_SUBTREE:
           goto C2_002_FIRST_PART;
       C2_002_FIRST_PART_EX:
           goto C2_009_BODY;
       C2_009_BODY_EX:
           goto C2_018_FULLSTOP;
       C2_018_FULLSTOP_EX:
       C2_001_COMMAND_SUBTREE_A:
                switch(g2_001_COMMAND_SUBTREE_A){
                case 0 : goto C2_001_EXIT01;break;
                case 1 : goto C2_001_EXIT02;break;
                }
      /**/
       C2_002_FIRST_PART:
           goto C2_003_FIRST_BODY;
       C2_003_FIRST_BODY_EX:
           goto C2_008_FIRST_END;
       C2_008_FIRST_END_EX:
           goto C2_002_FIRST_PART_EX;
      /**/
       C2_003_FIRST_BODY:
           if((
  /*C06*/         (token->token_type==KEYWORD)
))
                  goto C2_004_VERB;
           if((
  /*C04*/         (token->token_type==NUMERIC)
))
                  goto C2_005_NUMBER;
           if((
  /*C05*/         (token->token_type==NON_NUMERIC_LITERAL)
))
                  goto C2_006_STRING;
           if((
  /*C03*/         (token->token_type==VARIABLE_NAME)
```

```
))
                    goto C2_007_VARIABLE;
        C2_004_VERB_EX:
        C2_005_NUMBER_EX:
        C2_006_STRING_EX:
        C2_007_VARIABLE_EX:
            goto C2_003_FIRST_BODY_EX;
      /**/
        C2_004_VERB:
 /*013*/      op_list->type=OP_VERB;
            op_list->value.keyword_number=token->value.keyword_number;
            goto C2_004_VERB_EX;
      /**/
        C2_005_NUMBER:
 /*015*/      op_list->type=OP_NUM_LIT;
            op_list->value.numeric_literal=token->value.keyword_number;
            goto C2_005_NUMBER_EX;
      /**/
        C2_006_STRING:
 /*014*/      op_list->type=OP_STRING;
            op_list->value.non_numeric_literal=token->value.string;
            goto C2_006_STRING_EX;
      /**/
        C2_007_VARIABLE:
 /*006*/      node=look_for(token->value.var_values.unique,
                          token->value.var_values.hash,
                          symtab);
 /*012*/      op_list->type=OP_VAR;
            op_list->value.var_name=node;
            goto C2_007_VARIABLE_EX;
      /**/
        C2_008_FIRST_END:
 /*004*/      eof=get_token(infile,token);
            goto C2_008_FIRST_END_EX;
      /**/
        C2_009_BODY:
        C2_010_PART_EX:
            if(!(
 /*C07*/          (token->token_type==FULLSTOP)
```

```
))
                    goto C2_010_PART;
            goto C2_009_BODY_EX;
        /**/
         C2_010_PART:
            goto C2_011_PART_START;
         C2_011_PART_START_EX:
            goto C2_012_PART_BODY;
         C2_012_PART_BODY_EX:
            goto C2_017_PART_END;
         C2_017_PART_END_EX:
            goto C2_010_PART_EX;
        /**/
         C2_011_PART_START:
  /*016*/      op_list->next=create_op_list();
            op_list=op_list->next;
            goto C2_011_PART_START_EX;
        /**/
         C2_012_PART_BODY:
            if((
  /*C06*/          (token->token_type==KEYWORD)
))
                    goto C2_013_VERB;
            if((
  /*C04*/          (token->token_type==NUMERIC)
))
                    goto C2_014_NUMBER;
            if((
  /*C05*/          (token->token_type==NON_NUMERIC_LITERAL)
))
                    goto C2_015_STRING;
            if((
  /*C03*/          (token->token_type==VARIABLE_NAME)
))
                    goto C2_016_VARIABLE;
        C2_013_VERB_EX:
        C2_014_NUMBER_EX:
        C2_015_STRING_EX:
        C2_016_VARIABLE_EX:
```

```
            goto C2_012_PART_BODY_EX;
      /**/
       C2_013_VERB:
 /*013*/       op_list->type=OP_VERB;
            op_list->value.keyword_number=token->value.keyword_number;
            goto C2_013_VERB_EX;
      /**/
       C2_014_NUMBER:
 /*015*/       op_list->type=OP_NUM_LIT;
            op_list->value.numeric_literal=token->value.keyword_number;
            goto C2_014_NUMBER_EX;
      /**/
       C2_015_STRING:
 /*014*/       op_list->type=OP_STRING;
            op_list->value.non_numeric_literal=token->value.string;
            goto C2_015_STRING_EX;
      /**/
       C2_016_VARIABLE:
 /*006*/       node=look_for(token->value.var_values.unique,
                          token->value.var_values.hash,
                          symtab);
 /*012*/       op_list->type=OP_VAR;
            op_list->value.var_name=node;
            goto C2_016_VARIABLE_EX;
      /**/
       C2_017_PART_END:
 /*004*/       eof=get_token(infile,token);
            goto C2_017_PART_END_EX;
      /**/
       C2_018_FULLSTOP:
 /*004*/       eof=get_token(infile,token);
            goto C2_018_FULLSTOP_EX;
      /**/
      /*   Data Analysis Map*/
      /**/
      /*                          ------------- Data Analysis By -------------*
      /**/
      /*                          BOX TYPE                OPERATION  ALLOCATION*
      /*  Tree name: OPS-BUILDER*/
```

```
        /**/
        /*                          Leaf       :  4          Operations:  14*/
        /*                                                   Quits      :   0*/
        /*                          Selections:   0*/
        /*                          Sequences :   5*/
        /*                          Iterations:   2*/
        /*                          Backtracks:   0*/
        /*                          Subtrees  :   2   -->    COMMAND-SUBTREE*/
        /*                                                   COMMAND-SUBTREE*/
        /**/
        /*  Tree name: COMMAND-SUBTREE*/
        /**/
        /*                          Leaf       : 12          Operations:  14*/
        /*                                                   Quits      :   0*/
        /*                          Selections:   2*/
        /*                          Sequences :   3*/
        /*                          Iterations:   1*/
        /*                          Backtracks:   0*/
        /*                          Subtrees  :   0*/
        /**/
        /**/
        /**/
veryend: ;
}
```

F.6 Scanning The '.cnd'

F.6.1 Operations

```
001 infile=open_input("b:altest.cnd");
002 close_file(infile);
003 outfile=open_output("b:cndout.dat");
004 close_file(outfile);
005 currch=getc(infile);
    if(currch!=EOF) nextch=getc(infile);
006 currch=nextch;
    if(nextch!=EOF) nextch=getc(infile);
007 chp=ch;
008 load_symtab(symtab,"b:symtable.out");
    cross_reference(symtab,100);
009 listpointer=root;
010 listpointer=check_pointer(listpointer,currch);
011 dat=get_data(listpointer);
012 write_token(outfile,token);
013 token->token_type=FULLSTOP;
    token->value.keyword_number=FULLSTOP;
014 token->token_type=KEYWORD;
    token->value.keyword_number=dat;
016 token->token_type=NON_NUMERIC_LITERAL;
    token->value.string=ch;
017 token->token_type=NON_NUMERIC_LITERAL_LENGTH;
    token->value.keyword_number=length_of;
018 dat=str_to_int(ch);
019 *chp='\0';
020 chp++;
021 *chp=currch;
022 token->token_type=COND_NUM;token->value.keyword_number=dat;
023 token->token_type=VARIABLE_NAME;
    token->value.var_values.hash=node->own.hash;
    token->value.var_values.unique=node->own.unique;
024 token->token_type=NUMERIC;token->value.keyword_number=dat;
025 length_of=0;
026 length_of++;
027 node=look_for_by_name(ch,symtab);
```

```
028 token=&t;
029 dat=GREATER_THAN;
030 dat=LESS_THAN;
031 dat=EQUAL_TO;
032 token->token_type=RELATIONAL_OP;token->value.keyword_number=dat;
```

F.6.2 Conditions

```
C01 currch==EOF
C02 currch==' '
C03 currch==EOF || currch=='\n'
C04 currch!=' '
C05 currch=='"' || currch=='\''
C06 currch=='.'
C07 currch!='"' &&
    currch!='\'' &&
    currch!='.' &&
    currch!='\n'
C08 currch=='\''
C09 currch=='"'
C10 currch==' ' || currch=='\n' ||
    currch=='.' || currch==EOF
C11 node==NULL && isdigit(ch[0])
C12 currch==EOF || currch=='C'
C13 currch=='\n' || currch==EOF
C14 currch=='\'' && nextch=='\''
C15 currch=='"' && nextch=='"'
C16 currch=='\'' && nextch!='\''
C17 currch=='"' && nextch!='"'
C18 currch=='<' || currch=='>' || currch=='='
C19 currch=='<'
C20 currch=='>'
C21 currch=='='
C22 node!=NULL
```

F.6.3 Data Definitions

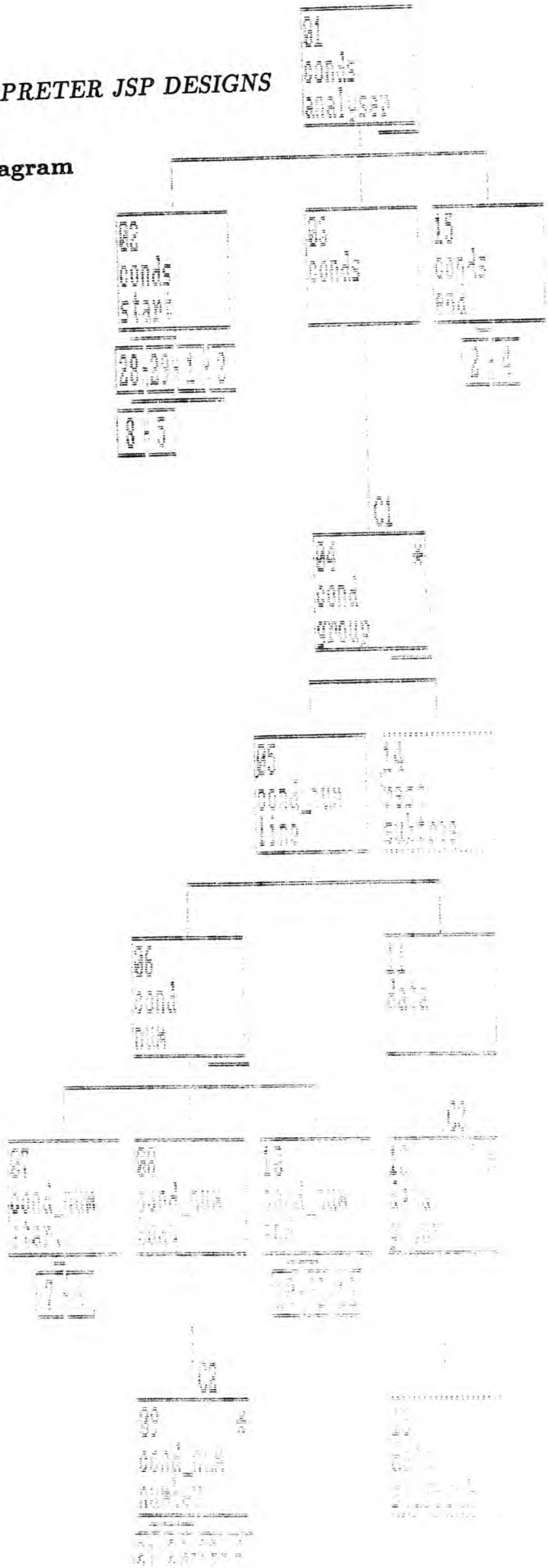
```
#include <stdio.h>
#include "alfile.h"
```

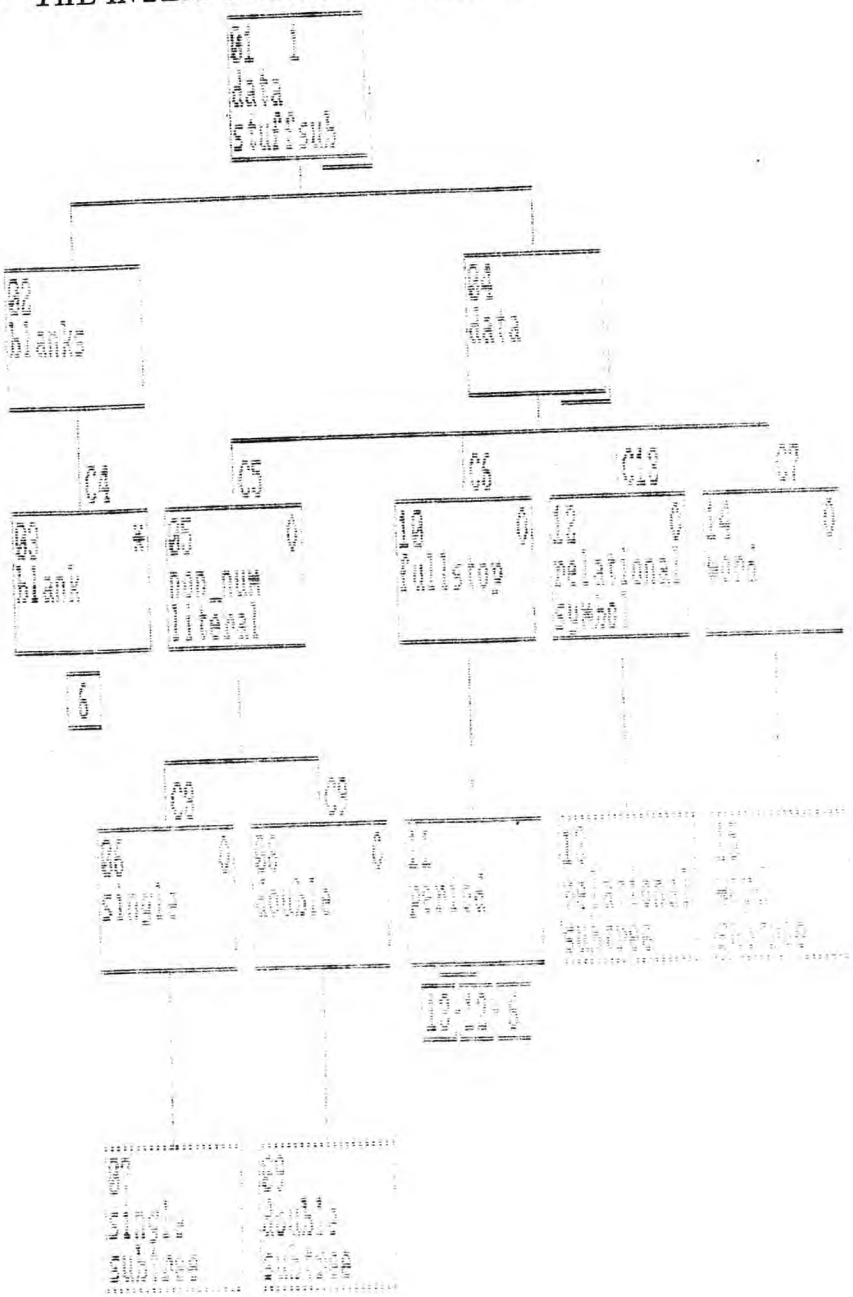
```
#include "staglist.h"
#include "node.h"
#include "alutil.h"
#include "symtab.h"
#include "token.h"
#include "conds.h"

void main(){

FILE *infile,*outfile;
char currch,nextch,ch[100],*chp;
staggered_list *listpointer,*root=create_list_with_file("B:keywords");
SYMTAB_ENTRY **symtab=create_symbol_table_array(100);
int dat,length_of;
TOKEN *token,t;
NODE *node;


EOD!
```

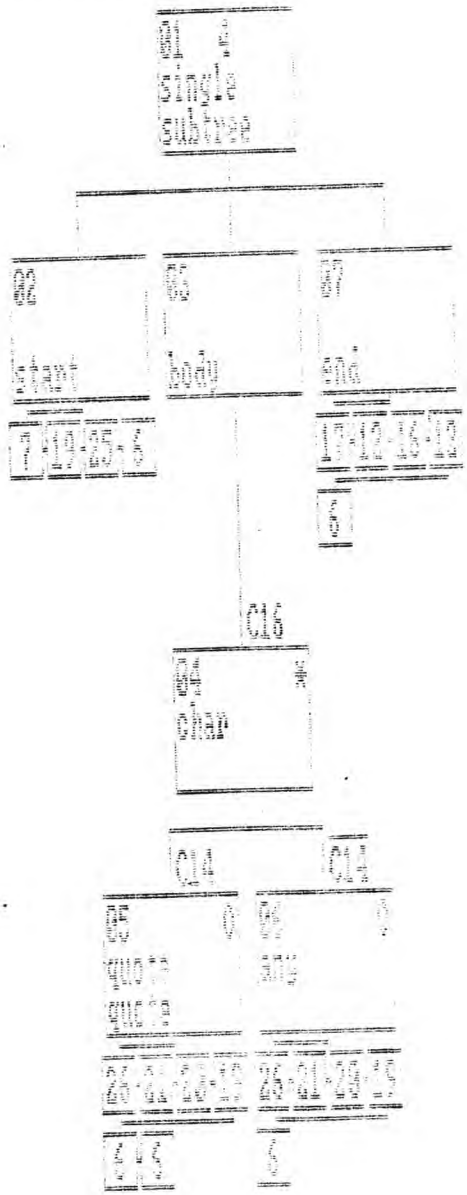
F.6.4 The Quits

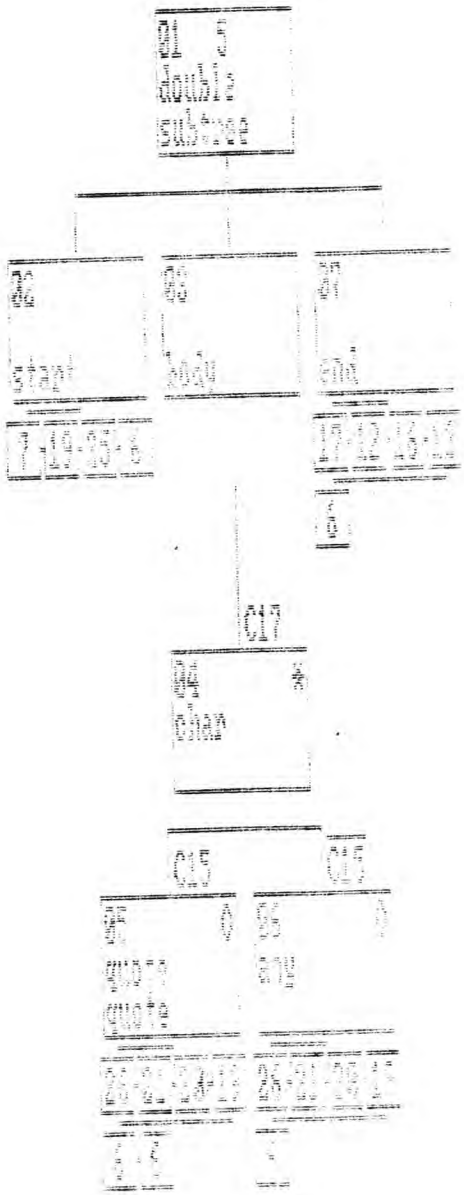
```
Q01 (!isupper(currch)) && currch!='-'
Q02 listpointer==NULL
Q03 dat==NULL
```

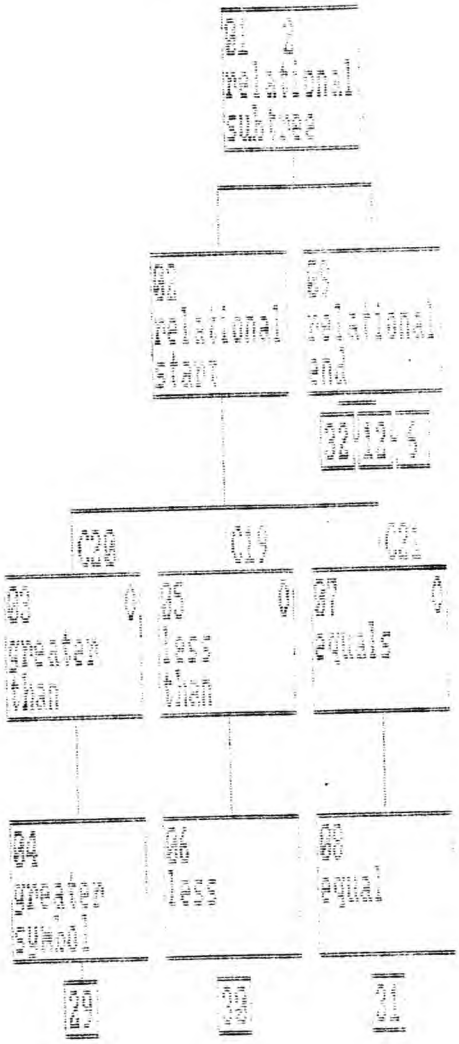
F.6.5 The Diagram

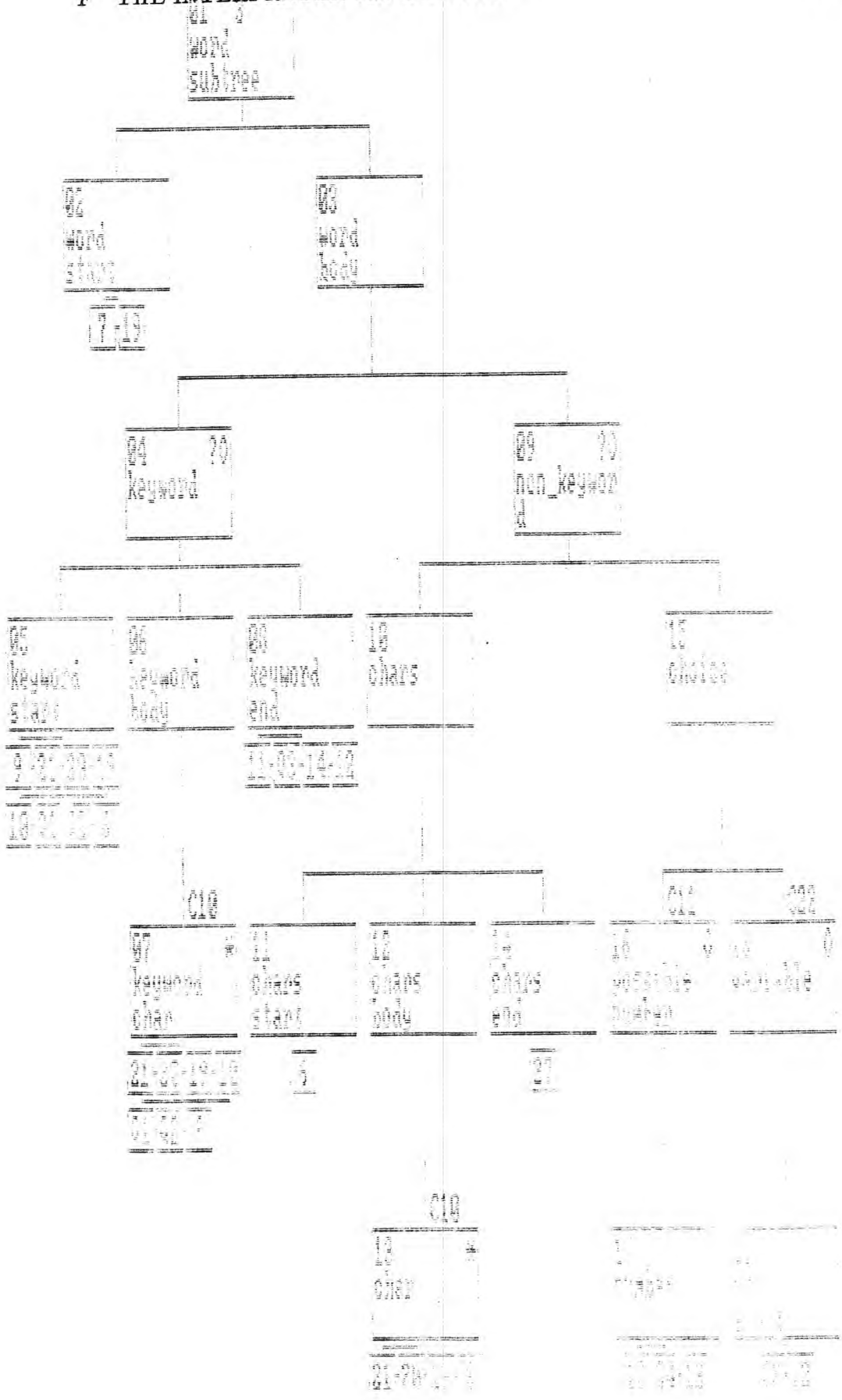



01 1
data
stuffsub
02
blanks
04
03 *
blank
05 0
non_num
literal
06
07
10 0
fullstop
12 0
relational
symbol
14 0
word
08
09
06 0
single
08 0
double
11
period
13
relational
subtree
15
word
subtree
10,11,5
07
single
subtree
09
double
subtree


C1 :
single
subtree
C2
start
7 19 35 6
C3
:
body
C7
end
17 12 16 13
6
C15
C4 *
char
C14
C5 0
quote
quote
26 21 20 15
2 1 5
C13
C6 0
else
26 21 29 15
5


C1 5
double
subtrac
C2
start
C3
body
C7
end
7,19,25,16
17,42,18,13
6
C17
C4 *
blan
C15
C15
C6 0
quote
quote
C5 0
ho
26,21,13,19
26,21,18,15
17
1


B1 2
relational
subtree
B2
relational
start
B3
relational
end
C20
C19
C21
B3 0
greater
than
B5 0
less
than
B7 0
equals
B4
greater
symbol
B6
less
B8
equal
B29
B30
B31



F.6.6 The Code

```
#include <stdio.h>
#include "alfile.h"
#include "staglist.h"
#include "node.h"
#include "alutil.h"
#include "symtab.h"
#include "token.h"

void main(){

FILE *infile,*outfile;
char currch,nextch,ch[100],*chp;
staggered_list *listpointer,*root=create_list_with_file("B:keywords");
SYMTAB_ENTRY **symtab=create_symbol_table_array(100);
int dat,length_of;
TOKEN *token,t;
NODE *node;


                int g2_001_DATA_STUFFSUB_A;
                int g3_001_SINGLE_SUBTREE_A;
                int g4_001_DOUBLE_SUBTREE_A;
                int g5_001_RELATIONAL_SUBTREE_A;
                int g6_001_WORD_SUBTREE_A;
        /*********************************************************************
        /*********************************************************************
        /****************** DRIVING PROCEDURE    B:\CONDSANY ******************
        /*********************************************************************
         PROCEDURE_DIVISION:
        /**/
        /** conds-analyser takes the probench .cnd*/
        /** file and tokenises it, ready for the*/
        /** next stage of conds analysis.*/
        /**/
         C1_001_CONDS_ANALYSER:
             goto C1_002_CONDS_START;
         C1_002_CONDS_START_EX:
```

```
            goto C1_003_CONDS;
        C1_003_CONDS_EX:
            goto C1_021_CONDS_END;
        C1_021_CONDS_END_EX:
            goto veryend;
       /**/
        C1_002_CONDS_START:
  /*028*/       token=&t;
  /*029*/       dat=GREATER_THAN;
  /*001*/       infile=open_input("b:altest.cnd");
  /*003*/       outfile=open_output("b:cndout.dat");
  /*008*/       load_symtab(symtab,"b:symtable.out");
            cross_reference(symtab,100);
  /*005*/       currch=getc(infile);
            if(currch!=EOF) nextch=getc(infile);
            goto C1_002_CONDS_START_EX;
       /**/
        C1_003_CONDS:
        C1_004_COND_GROUP_EX:
            if(!(
  /*C01*/         (currch==EOF)
))
                    goto C1_004_COND_GROUP;
            goto C1_003_CONDS_EX;
       /**/
       /** a condition group consists of a line*/
       /** with a condition number on it, followed*/
       /** by perhaps more lines*/
       /**/
        C1_004_COND_GROUP:
            goto C1_005_COND_NUM_LINE;
        C1_005_COND_NUM_LINE_EX:
            goto C1_015_REST_OF_GROUP;
        C1_015_REST_OF_GROUP_EX:
            goto C1_004_COND_GROUP_EX;
       /**/
        C1_005_COND_NUM_LINE:
            goto C1_006_COND_NUM;
        C1_006_COND_NUM_EX:
```

```
            goto C1_011_DATA;
        C1_011_DATA_EX:
            goto C1_005_COND_NUM_LINE_EX;
       /**/
       /** read the condition number, extract the*/
       /** numeric characters and turn them into an*/
       /** integer*/
       /**/
        C1_006_COND_NUM:
            goto C1_007_COND_NUM_START;
        C1_007_COND_NUM_START_EX:
            goto C1_008_COND_NUM_BODY;
        C1_008_COND_NUM_BODY_EX:
            goto C1_010_COND_NUM_END;
        C1_010_COND_NUM_END_EX:
            goto C1_006_COND_NUM_EX;
       /**/
        C1_007_COND_NUM_START:
  /*007*/       chp=ch;
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C1_007_COND_NUM_START_EX;
       /**/
        C1_008_COND_NUM_BODY:
        C1_009_COND_NUM_NUMBER_EX:
            if(!(
  /*C02*/         (currch==' ')
))
                    goto C1_009_COND_NUM_NUMBER;
            goto C1_008_COND_NUM_BODY_EX;
       /**/
        C1_009_COND_NUM_NUMBER:
  /*021*/       *chp=currch;
  /*020*/       chp++;
  /*019*/       *chp='\0';
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C1_009_COND_NUM_NUMBER_EX;
       /**/
```

```
      C1_010_COND_NUM_END:
  /*018*/      dat=str_to_int(ch);
  /*022*/      token->token_type=COND_NUM;token->value.keyword_number=dat;
  /*012*/      write_token(outfile,token);
          goto C1_010_COND_NUM_END_EX;
      /**/
      C1_011_DATA:
      C1_012_DATA_GROUP_EX:
          if(!(
  /*C03*/         (currch==EOF || currch=='\n')
))
                  goto C1_012_DATA_GROUP;
          goto C1_011_DATA_EX;
      /**/
      C1_012_DATA_GROUP:
          goto C1_013_DATA;
      C1_013_DATA_EX:
          goto C1_012_DATA_GROUP_EX;
      /**/
      C1_013_DATA:
          goto C1_014_DATA_STUFFSUB;
      C1_014_DATA_STUFFSUB_EX:
          goto C1_013_DATA_EX;
      /**/
      C1_014_DATA_STUFFSUB:
                g2_001_DATA_STUFFSUB_A=0;
          goto C2_001_DATA_STUFFSUB;
      C2_001_EXIT01:
          goto C1_014_DATA_STUFFSUB_EX;
      /**/
      C1_015_REST_OF_GROUP:
      C1_016_DATA_GROUP_EX:
          if(!(
  /*C12*/         (currch==EOF || currch=='C')
))
                  goto C1_016_DATA_GROUP;
          goto C1_015_REST_OF_GROUP_EX;
      /**/
      C1_016_DATA_GROUP:
```

```
            goto C1_017_DATA;
        C1_017_DATA_EX:
            goto C1_020_DATA_END;
        C1_020_DATA_END_EX:
            goto C1_016_DATA_GROUP_EX;
       /**/
        C1_017_DATA:
        C1_018_LINE_EX:
            if(!(
  /*C13*/         (currch=='\n' || currch==EOF)
))
                    goto C1_018_LINE;
            goto C1_017_DATA_EX;
       /**/
        C1_018_LINE:
            goto C1_019_DATA_STUFFSUB;
        C1_019_DATA_STUFFSUB_EX:
            goto C1_018_LINE_EX;
       /**/
        C1_019_DATA_STUFFSUB:
                 g2_001_DATA_STUFFSUB_A=1;
            goto C2_001_DATA_STUFFSUB;
        C2_001_EXIT02:
            goto C1_019_DATA_STUFFSUB_EX;
       /**/
        C1_020_DATA_END:
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C1_020_DATA_END_EX;
       /**/
        C1_021_CONDS_END:
  /*002*/       close_file(infile);
  /*004*/       close_file(outfile);
            goto C1_021_CONDS_END_EX;
       /**/
       /** a data segment consists of blanks */
       /** followed by a token of some sort*/
       /**/
        C2_001_DATA_STUFFSUB:
```

```
            goto C2_002_BLANKS;
        C2_002_BLANKS_EX:
            goto C2_004_DATA;
        C2_004_DATA_EX:
        C2_001_DATA_STUFFSUB_A:
                switch(g2_001_DATA_STUFFSUB_A){
                case 0 : goto C2_001_EXIT01;break;
                case 1 : goto C2_001_EXIT02;break;
                }
      /**/
        C2_002_BLANKS:
        C2_003_BLANK_EX:
            if(!(
  /*C04*/          (currch!=' ')
))
                    goto C2_003_BLANK;
            goto C2_002_BLANKS_EX;
      /**/
        C2_003_BLANK:
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C2_003_BLANK_EX;
      /**/
      /** the token can be a string, integer,*/
      /** fullstop, releational operator*/
      /**/
        C2_004_DATA:
            if((
  /*C05*/          (currch=='"' || currch=='\'')
))
                    goto C2_005_NON_NUM_LITERAL;
            if((
  /*C06*/          (currch=='.')
))
                    goto C2_010_FULLSTOP;
            if((
  /*C18*/          (currch=='<' || currch=='>' || currch=='=')
))
                    goto C2_012_RELATIONAL_SYMBOL;
```

```
            if((
  /*C07*/          (currch!='"' &&
                 currch!='\'' &&
                 currch!='.' &&
                 currch!='\n')
))
                    goto C2_014_WORD;
        C2_005_NON_NUM_LITERAL_EX:
        C2_010_FULLSTOP_EX:
        C2_012_RELATIONAL_SYMBOL_EX:
        C2_014_WORD_EX:
            goto C2_004_DATA_EX;
       /**/
        C2_005_NON_NUM_LITERAL:
            if((
  /*C08*/          (currch=='\'')
))
                    goto C2_006_SINGLE;
            if((
  /*C09*/          (currch=='"')
))
                    goto C2_008_DOUBLE;
        C2_006_SINGLE_EX:
        C2_008_DOUBLE_EX:
            goto C2_005_NON_NUM_LITERAL_EX;
       /**/
        C2_006_SINGLE:
            goto C2_007_SINGLE_SUBTREE;
        C2_007_SINGLE_SUBTREE_EX:
            goto C2_006_SINGLE_EX;
       /**/
        C2_007_SINGLE_SUBTREE:
                g3_001_SINGLE_SUBTREE_A=0;
            goto C3_001_SINGLE_SUBTREE;
        C3_001_EXIT01:
            goto C2_007_SINGLE_SUBTREE_EX;
       /**/
        C2_008_DOUBLE:
            goto C2_009_DOUBLE_SUBTREE;
```

```
     C2_009_DOUBLE_SUBTREE_EX:
         goto C2_008_DOUBLE_EX;
    /**/
     C2_009_DOUBLE_SUBTREE:
              g4_001_DOUBLE_SUBTREE_A=0;
         goto C4_001_DOUBLE_SUBTREE;
     C4_001_EXIT01:
         goto C2_009_DOUBLE_SUBTREE_EX;
    /**/
     C2_010_FULLSTOP:
         goto C2_011_PERIOD;
     C2_011_PERIOD_EX:
         goto C2_010_FULLSTOP_EX;
    /**/
     C2_011_PERIOD:
/*013*/      token->token_type=FULLSTOP;
         token->value.keyword_number=FULLSTOP;
/*012*/      write_token(outfile,token);
/*006*/      currch=nextch;
         if(nextch!=EOF) nextch=getc(infile);
         goto C2_011_PERIOD_EX;
    /**/
     C2_012_RELATIONAL_SYMBOL:
         goto C2_013_RELATIONAL_SUBTREE;
     C2_013_RELATIONAL_SUBTREE_EX:
         goto C2_012_RELATIONAL_SYMBOL_EX;
    /**/
     C2_013_RELATIONAL_SUBTREE:
              g5_001_RELATIONAL_SUBTREE_A=0;
         goto C5_001_RELATIONAL_SUBTREE;
     C5_001_EXIT01:
         goto C2_013_RELATIONAL_SUBTREE_EX;
    /**/
     C2_014_WORD:
         goto C2_015_WORD_SUBTREE;
     C2_015_WORD_SUBTREE_EX:
         goto C2_014_WORD_EX;
    /**/
     C2_015_WORD_SUBTREE:
```

```
                g6_001_WORD_SUBTREE_A=0;
            goto C6_001_WORD_SUBTREE;
        C6_001_EXIT01:
            goto C2_015_WORD_SUBTREE_EX;
       /**/
        C5_001_RELATIONAL_SUBTREE:
            goto C5_002_RELATIONAL_START;
        C5_002_RELATIONAL_START_EX:
            goto C5_009_RELATIONAL_END;
        C5_009_RELATIONAL_END_EX:
        C5_001_RELATIONAL_SUBTREE_A:
                switch(g5_001_RELATIONAL_SUBTREE_A){
                case 0 : goto C5_001_EXIT01;break;
                }
       /**/
        C5_002_RELATIONAL_START:
            if((
  /*C20*/          (currch=='>')
))
                    goto C5_003_GREATER_THAN;
            if((
  /*C19*/          (currch=='<')
))
                    goto C5_005_LESS_THAN;
            if((
  /*C21*/          (currch=='=')
))
                    goto C5_007_EQUALS;
        C5_003_GREATER_THAN_EX:
        C5_005_LESS_THAN_EX:
        C5_007_EQUALS_EX:
            goto C5_002_RELATIONAL_START_EX;
       /**/
        C5_003_GREATER_THAN:
            goto C5_004_GREATER_SYMBOL;
        C5_004_GREATER_SYMBOL_EX:
            goto C5_003_GREATER_THAN_EX;
       /**/
        C5_004_GREATER_SYMBOL:
```

```
/*029*/       dat=GREATER_THAN;
          goto C5_004_GREATER_SYMBOL_EX;
    /**/
     C5_005_LESS_THAN:
          goto C5_006_LESS;
     C5_006_LESS_EX:
          goto C5_005_LESS_THAN_EX;
    /**/
     C5_006_LESS:
/*030*/       dat=LESS_THAN;
          goto C5_006_LESS_EX;
    /**/
     C5_007_EQUALS:
          goto C5_008_EQUAL;
     C5_008_EQUAL_EX:
          goto C5_007_EQUALS_EX;
    /**/
     C5_008_EQUAL:
/*031*/       dat=EQUAL_TO;
          goto C5_008_EQUAL_EX;
    /**/
     C5_009_RELATIONAL_END:
/*032*/       token->token_type=RELATIONAL_OP;token->value.keyword_number=dat
/*012*/       write_token(outfile,token);
/*006*/       currch=nextch;
          if(nextch!=EOF) nextch=getc(infile);
          goto C5_009_RELATIONAL_END_EX;
    /**/
     C6_001_WORD_SUBTREE:
          goto C6_002_WORD_START;
     C6_002_WORD_START_EX:
          goto C6_003_WORD_BODY;
     C6_003_WORD_BODY_EX:
     C6_001_WORD_SUBTREE_A:
               switch(g6_001_WORD_SUBTREE_A){
               case 0 : goto C6_001_EXIT01;break;
               }
    /**/
     C6_002_WORD_START:
```

```
/*007*/      chp=ch;
/*019*/      *chp='\0';
         goto C6_002_WORD_START_EX;
     /**/
      C6_003_WORD_BODY:
         goto C6_004_KEYWORD;
      C6_004_KEYWORD_EX:
      C6_009_NON_KEYWOR_D_EX:
         goto C6_003_WORD_BODY_EX;
     /**/
      C6_004_KEYWORD:
         goto C6_005_KEYWORD_START;
      C6_005_KEYWORD_START_EX:
         goto C6_006_KEYWORD_BODY;
      C6_006_KEYWORD_BODY_EX:
         goto C6_008_KEYWORD_END;
      C6_008_KEYWORD_END_EX:
         goto C6_004_KEYWORD_EX;
     /**/
      C6_005_KEYWORD_START:
/*009*/      listpointer=root;
/*021*/      *chp=currch;
/*020*/      chp++;
/*019*/      *chp='\0';
/*010*/      listpointer=check_pointer(listpointer,currch);
         if((
/*Q01*/         (!isupper(currch)) && currch!='-'
))
                  goto C6_009_NON_KEYWOR_D;
         if((
/*Q02*/         listpointer==NULL
))
                  goto C6_009_NON_KEYWOR_D;
/*006*/      currch=nextch;
         if(nextch!=EOF) nextch=getc(infile);
         goto C6_005_KEYWORD_START_EX;
     /**/
      C6_006_KEYWORD_BODY:
      C6_007_KEYWORD_CHAR_EX:
```

```
            if(!(
  /*C10*/          (currch==' ' || currch=='\n' ||
                 currch=='.' || currch==EOF)
))
                    goto C6_007_KEYWORD_CHAR;
            goto C6_006_KEYWORD_BODY_EX;
      /**/
       C6_007_KEYWORD_CHAR:
  /*021*/       *chp=currch;
  /*020*/       chp++;
  /*019*/       *chp='\0';
  /*010*/       listpointer=check_pointer(listpointer,currch);
            if((
  /*Q01*/          (!isupper(currch)) && currch!='-'
))
                    goto C6_009_NON_KEYWOR_D;
            if((
  /*Q02*/          listpointer==NULL
))
                    goto C6_009_NON_KEYWOR_D;
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C6_007_KEYWORD_CHAR_EX;
      /**/
       C6_008_KEYWORD_END:
  /*011*/       dat=get_data(listpointer);
            if((
  /*Q03*/          dat==NULL
))
                    goto C6_009_NON_KEYWOR_D;
  /*014*/       token->token_type=KEYWORD;
            token->value.keyword_number=dat;
  /*012*/       write_token(outfile,token);
            goto C6_008_KEYWORD_END_EX;
      /**/
       C6_009_NON_KEYWOR_D:
            goto C6_010_CHARS;
       C6_010_CHARS_EX:
            goto C6_015_CHOICE;
```

```
        C6_015_CHOICE_EX:
            goto C6_009_NON_KEYWOR_D_EX;
       /**/
        C6_010_CHARS:
            goto C6_011_CHARS_START;
        C6_011_CHARS_START_EX:
            goto C6_012_CHARS_BODY;
        C6_012_CHARS_BODY_EX:
            goto C6_014_CHARS_END;
        C6_014_CHARS_END_EX:
            goto C6_010_CHARS_EX;
       /**/
        C6_011_CHARS_START:
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C6_011_CHARS_START_EX;
       /**/
        C6_012_CHARS_BODY:
        C6_013_CHAR_EX:
            if(!(
  /*C10*/         (currch==' ' || currch=='\n' ||
                currch=='.' || currch==EOF)
))
                    goto C6_013_CHAR;
            goto C6_012_CHARS_BODY_EX;
       /**/
        C6_013_CHAR:
  /*021*/       *chp=currch;
  /*020*/       chp++;
  /*019*/       *chp='\0';
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C6_013_CHAR_EX;
       /**/
        C6_014_CHARS_END:
  /*027*/       node=look_for_by_name(ch,symtab);
            goto C6_014_CHARS_END_EX;
       /**/
        C6_015_CHOICE:
```

```
            if((
  /*C11*/          (node==NULL && isdigit(ch[0]))
))
                    goto C6_016_POSSIBLE_NUMBER;
            if((
  /*C22*/          (node!=NULL)
))
                    goto C6_018_VARIABLE;
        C6_016_POSSIBLE_NUMBER_EX:
        C6_018_VARIABLE_EX:
            goto C6_015_CHOICE_EX;
       /**/
        C6_016_POSSIBLE_NUMBER:
            goto C6_017_NUMBER;
        C6_017_NUMBER_EX:
            goto C6_016_POSSIBLE_NUMBER_EX;
       /**/
        C6_017_NUMBER:
  /*018*/       dat=str_to_int(ch);
  /*024*/       token->token_type=NUMERIC;token->value.keyword_number=dat;
  /*012*/       write_token(outfile,token);
            goto C6_017_NUMBER_EX;
       /**/
        C6_018_VARIABLE:
            goto C6_019_VAR_START;
        C6_019_VAR_START_EX:
            goto C6_018_VARIABLE_EX;
       /**/
        C6_019_VAR_START:
  /*023*/       token->token_type=VARIABLE_NAME;token->value.var_values.hash=nc
            token->value.var_values.unique=node->own.unique;
  /*012*/       write_token(outfile,token);
            goto C6_019_VAR_START_EX;
       /**/
        C3_001_SINGLE_SUBTREE:
            goto C3_002__START;
        C3_002__START_EX:
            goto C3_003__BODY;
        C3_003__BODY_EX:
```

```
            goto C3_007__END;
        C3_007__END_EX:
        C3_001_SINGLE_SUBTREE_A:
                switch(g3_001_SINGLE_SUBTREE_A){
                case 0 : goto C3_001_EXIT01;break;
                }
      /**/
        C3_002__START:
  /*007*/       chp=ch;
  /*019*/       *chp='\0';
  /*025*/       length_of=0;
  /*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C3_002__START_EX;
      /**/
        C3_003__BODY:
        C3_004_CHAR_EX:
            if(!(
  /*C16*/         (currch=='\'' && nextch!='\'')
))
                  goto C3_004_CHAR;
            goto C3_003__BODY_EX;
      /**/
        C3_004_CHAR:
            if((
  /*C14*/         (currch=='\'' && nextch=='\'')
))
                  goto C3_005_QUOTE_QUOTE;
            if(!(
  /*C14*/         (currch=='\'' && nextch=='\'')
))
                  goto C3_006_ANY;
        C3_005_QUOTE_QUOTE_EX:
        C3_006_ANY_EX:
            goto C3_004_CHAR_EX;
      /**/
        C3_005_QUOTE_QUOTE:
  /*026*/       length_of++;
  /*021*/       *chp=currch;
```

```
/*020*/        chp++;
/*019*/        *chp='\0';
/*006*/        currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
/*006*/        currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
           goto C3_005_QUOTE_QUOTE_EX;
     /**/
      C3_006_ANY:
/*026*/        length_of++;
/*021*/        *chp=currch;
/*020*/        chp++;
/*019*/        *chp='\0';
/*006*/        currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
           goto C3_006_ANY_EX;
     /**/
      C3_007__END:
/*017*/        token->token_type=NON_NUMERIC_LITERAL_LENGTH;
           token->value.keyword_number=length_of;
/*012*/        write_token(outfile,token);
/*016*/        token->token_type=NON_NUMERIC_LITERAL;
           token->value.string=ch;
/*012*/        write_token(outfile,token);
/*006*/        currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
           goto C3_007__END_EX;
     /**/
      C4_001_DOUBLE_SUBTREE:
           goto C4_002__START;
      C4_002__START_EX:
           goto C4_003__BODY;
      C4_003__BODY_EX:
           goto C4_007__END;
      C4_007__END_EX:
      C4_001_DOUBLE_SUBTREE_A:
                switch(g4_001_DOUBLE_SUBTREE_A){
                case 0 : goto C4_001_EXIT01;break;
                }
```

```
      /**/
       C4_002__START:
  /*007*/      chp=ch;
  /*019*/      *chp='\0';
  /*025*/      length_of=0;
  /*006*/      currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
           goto C4_002__START_EX;
      /**/
       C4_003__BODY:
       C4_004_CHAR_EX:
           if(!(
  /*C17*/         (currch=='"' && nextch!='"')
))
                  goto C4_004_CHAR;
           goto C4_003__BODY_EX;
      /**/
       C4_004_CHAR:
           if((
  /*C15*/         (currch=='"' && nextch=='"')
))
                  goto C4_005_QUOTE_QUOTE;
           if(!(
  /*C15*/         (currch=='"' && nextch=='"')
))
                  goto C4_006_ANY;
       C4_005_QUOTE_QUOTE_EX:
       C4_006_ANY_EX:
           goto C4_004_CHAR_EX;
      /**/
       C4_005_QUOTE_QUOTE:
  /*026*/      length_of++;
  /*021*/      *chp=currch;
  /*020*/      chp++;
  /*019*/      *chp='\0';
  /*006*/      currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
  /*006*/      currch=nextch;
           if(nextch!=EOF) nextch=getc(infile);
```

```
            goto C4_005_QUOTE_QUOTE_EX;
        /**/
         C4_006_ANY:
/*026*/       length_of++;
/*021*/       *chp=currch;
/*020*/       chp++;
/*019*/       *chp='\0';
/*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C4_006_ANY_EX;
        /**/
         C4_007__END:
/*017*/       token->token_type=NON_NUMERIC_LITERAL_LENGTH;
            token->value.keyword_number=length_of;
/*012*/       write_token(outfile,token);
/*016*/       token->token_type=NON_NUMERIC_LITERAL;
            token->value.string=ch;
/*012*/       write_token(outfile,token);
/*006*/       currch=nextch;
            if(nextch!=EOF) nextch=getc(infile);
            goto C4_007__END_EX;
        /**/
        /*   Data Analysis Map*/
        /**/
        /*                          ------------- Data Analysis By -------------*
        /**/
        /*                          BOX TYPE                OPERATION  ALLOCATION*
        /*  Tree name: CONDS-ANALYSER*/
        /**/
        /*                          Leaf      :   6          Operations:  18*/
        /*                                                   Quits     :   0*/
        /*                          Selections:   0*/
        /*                          Sequences :   8*/
        /*                          Iterations:   5*/
        /*                          Backtracks:   0*/
        /*                          Subtrees  :   2   -->    DATA-STUFFSUB*/
        /*                                                   DATA-STUFFSUB*/
        /**/
        /*  Tree name: DATA-STUFFSUB*/
```

```
/**/
/*                          Leaf       :   2          Operations:   4*/
/*                                                    Quits      :   0*/
/*                          Selections:   2*/
/*                          Sequences :   6*/
/*                          Iterations:   1*/
/*                          Backtracks:   0*/
/*                          Subtrees  :   4    -->   SINGLE-SUBTREE*/
/*                                                   DOUBLE-SUBTREE*/
/*                                                   RELATIONAL-SUBTREE*/
/*                                                   WORD-SUBTREE*/
/**/
/*  Tree name: RELATIONAL-SUBTREE*/
/**/
/*                          Leaf       :   4          Operations:   6*/
/*                                                    Quits      :   0*/
/*                          Selections:   1*/
/*                          Sequences :   4*/
/*                          Iterations:   0*/
/*                          Backtracks:   0*/
/*                          Subtrees  :   0*/
/**/
/*  Tree name: WORD-SUBTREE*/
/**/
/*                          Leaf       :   9          Operations:  27*/
/*                                                    Quits      :   5*/
/*                          Selections:   1*/
/*                          Sequences :   6*/
/*                          Iterations:   2*/
/*                          Backtracks:   1*/
/*                          Subtrees  :   0*/
/**/
/*  Tree name: SINGLE-SUBTREE*/
/**/
/*                          Leaf       :   4          Operations:  20*/
/*                                                    Quits      :   0*/
/*                          Selections:   1*/
/*                          Sequences :   1*/
/*                          Iterations:   1*/
```

```
        /*                          Backtracks:   0*/
        /*                          Subtrees  :   0*/
        /**/
        /*  Tree name: DOUBLE-SUBTREE*/
        /**/
        /*                          Leaf      :   4          Operations:  20*/
        /*                                                   Quits      :   0*/
        /*                          Selections:   1*/
        /*                          Sequences :   1*/
        /*                          Iterations:   1*/
        /*                          Backtracks:   0*/
        /*                          Subtrees  :   0*/
        /**/
        /**/
        /**/
veryend: ;
}
```

F.7 Parsing The '.cnd'

F.7.1 Operations

```
001 infile=open_input("b:cndout.dat");
002 close_file(infile);
003 cond->lhs.type=SIDE_LITERAL;
004 save_cond_array("b:cndout2.dat",cond_array);
005 token=&t;
006 eof=get_token(infile,token);
007 load_symtab(symtab,"b:symtable.out");
    cross_reference(symtab,100);
008 cond->lhs.contents.numeric_literal=token->value.keyword_number;
009 node=look_for(token->value.var_values.unique,
                  token->value.var_values.hash,
                  symtab);
010 cond=create_cond();
011 cond_array[token->value.keyword_number]=cond;
012 cond->lhs.type=SIDE_NON_LITERAL;
    cond->lhs.contents.non_numeric_literal=token->value.string;
013 cond->lhs.contents.variable=node;
    cond->lhs.type=SIDE_VAR;
014 cond->relation.relation=GREATER_THAN;
015 cond->relation.relation=LESS_THAN;
016 cond->relation.relation=EQUAL_TO;
017 cond->relation.not=YES_STATUS;
018 cond->rhs.type=SIDE_LITERAL;
    cond->rhs.contents.numeric_literal=token->value.keyword_number;
019 cond->rhs.type=SIDE_NON_LITERAL;
    cond->rhs.contents.non_numeric_literal=token->value.string;
020 cond->rhs.type=SIDE_VAR;
    cond->rhs.contents.variable=node;
021 cond->relation.not=NO_STATUS;
/*022 printf("right hand side\n");getchar();*/
/*023 printf("relation bit\n");*/
/*024 printf("lefthand side\n");*/
/*025 printf("numeric value \n");*/
/*026 printf("variable\n");*/
/*027 printf("string\n");*/
```

```
/*028 printf("string2\n");*/
```

F.7.2 Conditions

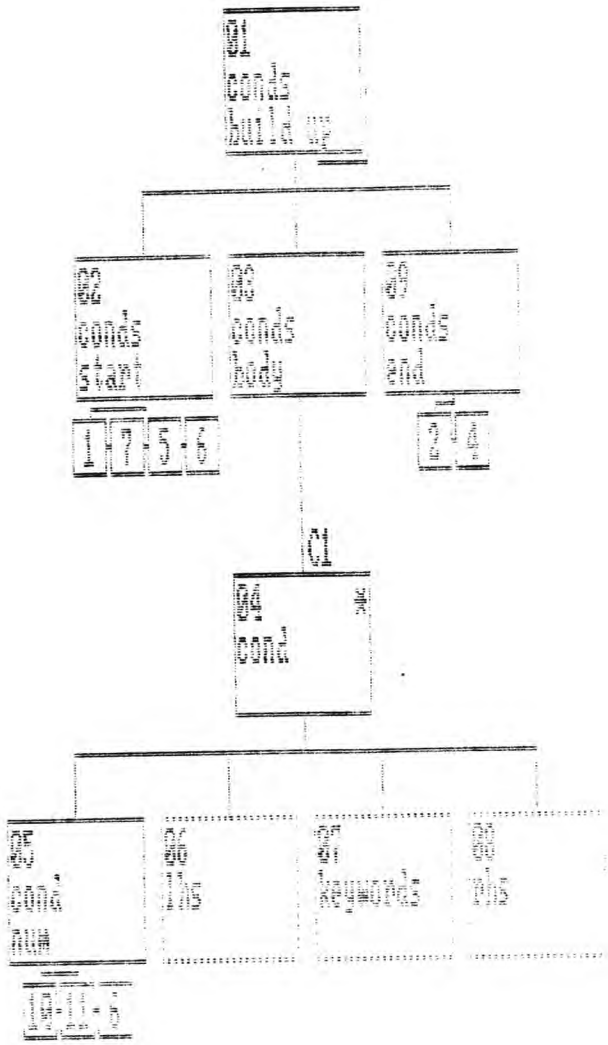
```
C01 eof==1
C02 token->token_type==NUMERIC
C03 token->token_type==VARIABLE_NAME
C04 token->token_type==NON_NUMERIC_LITERAL
C05 token->token_type==KEYWORD &&
    token->value.keyword_number==IS
C06 token->token_type==KEYWORD &&
    token->value.keyword_number==NOT
C07 token->token_type==KEYWORD ||
    token->token_type==RELATIONAL_OP
C08 token->token_type==RELATIONAL_OP
C09 token->token_type==KEYWORD
C10 token->value.keyword_number==GREATER_THAN
C11 token->value.keyword_number==LESS_THAN
C12 token->value.keyword_number==EQUAL_TO
C13 token->value.keyword_number==GREATER
C14 token->value.keyword_number==LESS
C15 token->value.keyword_number==EQUAL
```

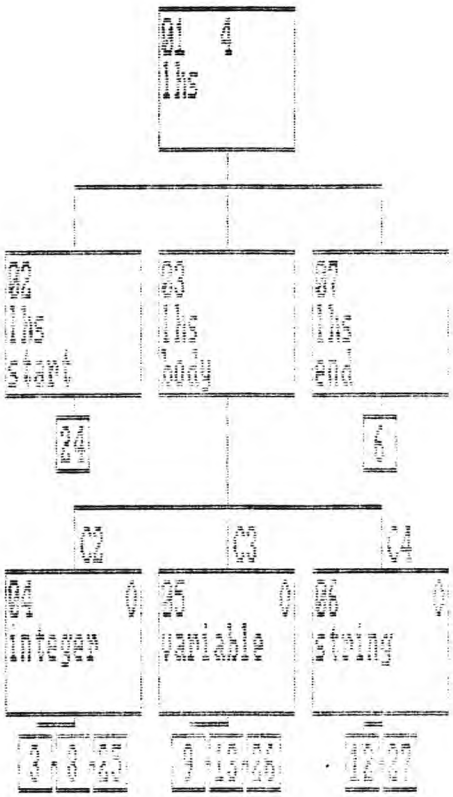
F.7.3 Data Definitions

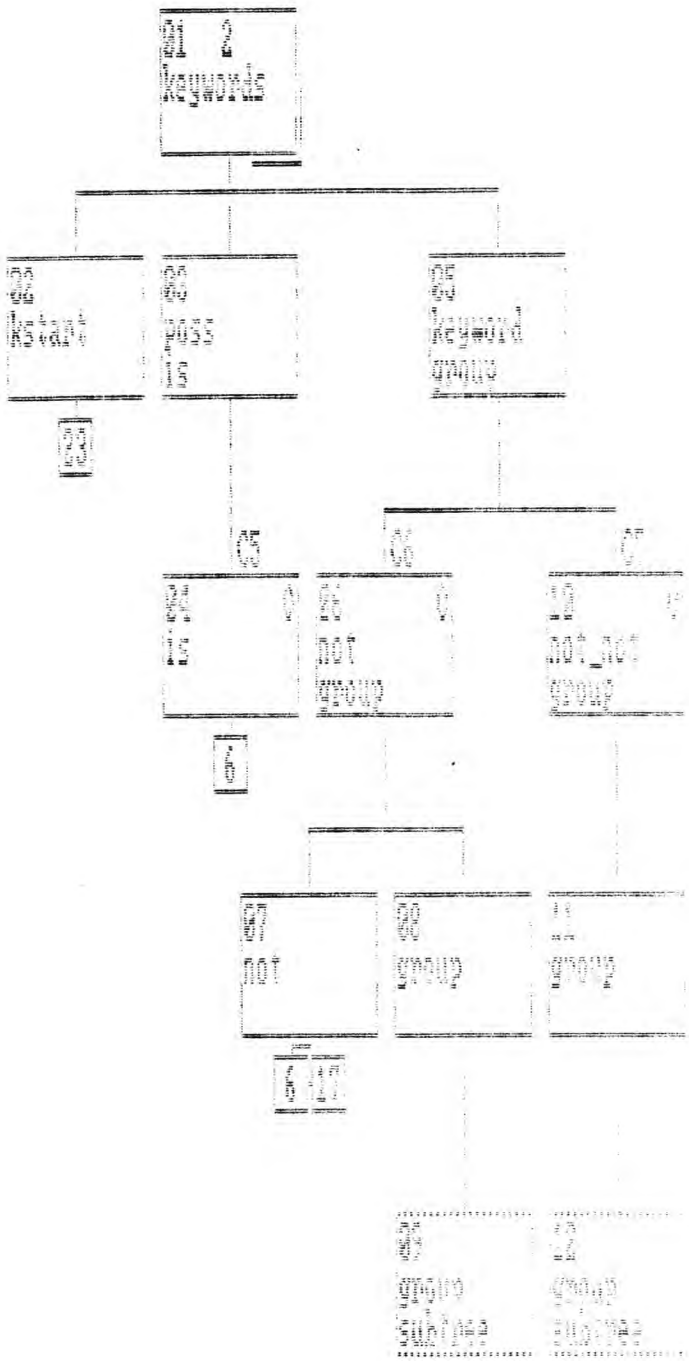
```
#include <stdio.h>
#include "alfile.h"
#include "staglist.h"
#include "node.h"
#include "alutil.h"
#include "symtab.h"
#include "keywords.h"
#include "token.h"
#include "conds.h"

void main(){

FILE *infile;
SYMTAB_ENTRY **symtab=create_symbol_table_array(100);
```

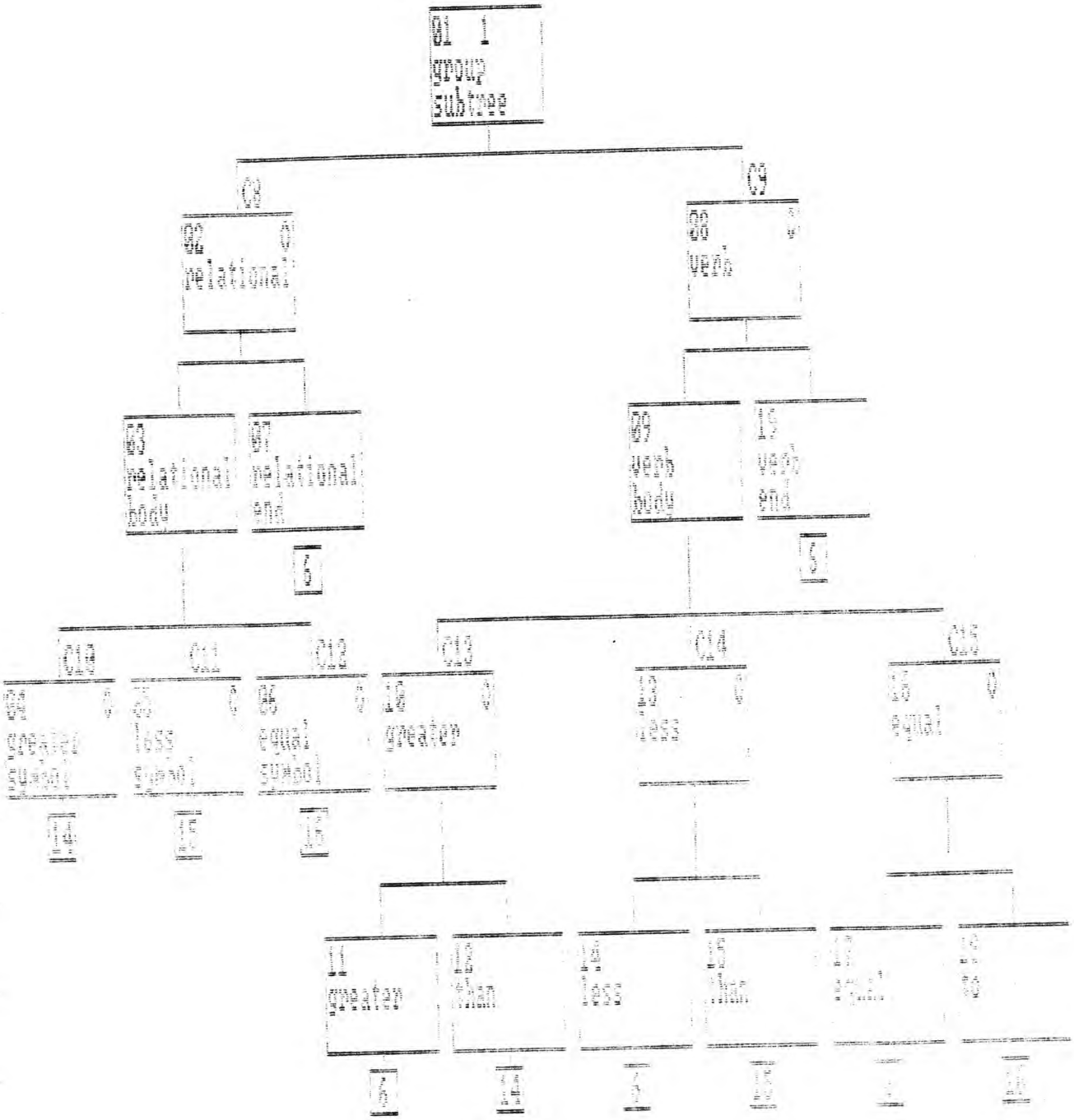
```
int eof;
TOKEN *token,t;
NODE *node;
COND *cond;
COND **cond_array=create_cond_array();


EOD!
```

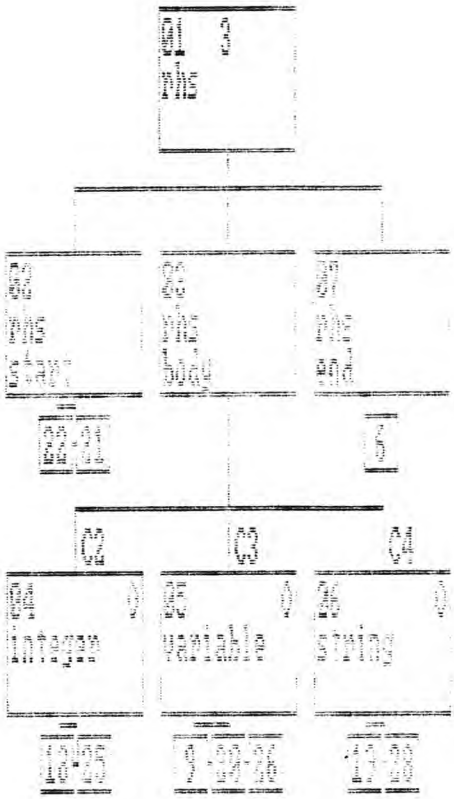
F.7.4 The Diagram




01
lhs
start
body
end
24
6
C2
C3
C4
04
integer
05
variable
06
string


01 2
keywords
02
kstart
03
03
poss
is
05
keyword
group
23
C3
C6
C7
04 ?
is
06 ?
not
group
12 ?
not_not
group
6
07
not
08
group
11
group
6 11
09
group
subtree
12
group
subtree


01 1
group
subtree
C8
C9
02 0
relational
08 0
verb
03
relational
body
07
relational
end
09
verb
body
15
verb
end
5
5
C10
C11
C12
C13
C14
C15
04 0
greater
equal
05 0
less
equal
06 0
equal
equal
10 0
greater
13 0
less
16 0
equal
14
15
16
11
greater
12
than
14
less
15
than
17
equal
18
to
6
14
7
15
1
16


01 3
nhs
02
nhs
start
22-31
03
nhs
body
07
nhs
end
6
C2
C3
C4
04 0
integer
10-25
05 0
variable
9-30-36
06 0
string
17-28

F.7.5 The Code

```
#include <stdio.h>
#include "alfile.h"
#include "staglist.h"
#include "node.h"
#include "alutil.h"
#include "symtab.h"
#include "keywords.h"
#include "token.h"
#include "conds.h"

void main(){

FILE *infile;
SYMTAB_ENTRY **symtab=create_symbol_table_array(100);
int eof;
TOKEN *token,t;
NODE *node;
COND *cond;
COND **cond_array=create_cond_array();


                int g2_001_LHS_A;
                int g3_001_KEYWORDS_A;
                int g4_001_RHS_A;
                int g5_001_GROUP_SUBTREE_A;
        /*********************************************************************
        /*********************************************************************
        /******************* DRIVING PROCEDURE    B:\CONDSBLD *****************
        /*********************************************************************
         PROCEDURE_DIVISION:
        /**/
        /** take the output from the condition*/
        /** tokeniser, it consists of an iteration*/
        /** of condition number, left hand side,*/
        /** a relational operator ie NOT >, or >*/
        /** then the right hand side*/
        /**/
```

```
        C1_001_CONDS_BUILD:
            goto C1_002_CONDS_START;
        C1_002_CONDS_START_EX:
            goto C1_003_CONDS_BODY;
        C1_003_CONDS_BODY_EX:
            goto C1_009_CONDS_END;
        C1_009_CONDS_END_EX:
            goto veryend;
      /**/
        C1_002_CONDS_START:
  /*001*/       infile=open_input("b:cndout.dat");
  /*007*/       load_symtab(symtab,"b:symtable.out");
            cross_reference(symtab,100);
  /*005*/       token=&t;
  /*006*/       eof=get_token(infile,token);
            goto C1_002_CONDS_START_EX;
      /**/
        C1_003_CONDS_BODY:
        C1_004_COND_EX:
            if(!(
  /*C01*/         (eof==1)
))
                  goto C1_004_COND;
            goto C1_003_CONDS_BODY_EX;
      /**/
        C1_004_COND:
            goto C1_005_COND_NUM;
        C1_005_COND_NUM_EX:
            goto C1_006_LHS;
        C1_006_LHS_EX:
            goto C1_007_KEYWORDS;
        C1_007_KEYWORDS_EX:
            goto C1_008_RHS;
        C1_008_RHS_EX:
            goto C1_004_COND_EX;
      /**/
        C1_005_COND_NUM:
  /*010*/       cond=create_cond();
  /*011*/       cond_array[token->value.keyword_number]=cond;
```

```
  /*006*/       eof=get_token(infile,token);
            goto C1_005_COND_NUM_EX;
       /**/
        C1_006_LHS:
                 g2_001_LHS_A=0;
            goto C2_001_LHS;
        C2_001_EXIT01:
            goto C1_006_LHS_EX;
       /**/
        C1_007_KEYWORDS:
                 g3_001_KEYWORDS_A=0;
            goto C3_001_KEYWORDS;
        C3_001_EXIT01:
            goto C1_007_KEYWORDS_EX;
       /**/
        C1_008_RHS:
                 g4_001_RHS_A=0;
            goto C4_001_RHS;
        C4_001_EXIT01:
            goto C1_008_RHS_EX;
       /**/
        C1_009_CONDS_END:
  /*002*/       close_file(infile);
  /*004*/       save_cond_array("b:cndout2.dat",cond_array);
            goto C1_009_CONDS_END_EX;
       /**/
        C5_001_GROUP_SUBTREE:
            if((
  /*C08*/          (token->token_type==RELATIONAL_OP)
))
                    goto C5_002_RELATIONAL;
            if((
  /*C09*/          (token->token_type==KEYWORD)
))
                    goto C5_008_VERB;
        C5_002_RELATIONAL_EX:
        C5_008_VERB_EX:
        C5_001_GROUP_SUBTREE_A:
                 switch(g5_001_GROUP_SUBTREE_A){
```

```
                case 0 : goto C5_001_EXIT02;break;
                case 1 : goto C5_001_EXIT03;break;
                }
        /**/
         C5_002_RELATIONAL:
             goto C5_003_RELATIONAL_BODY;
         C5_003_RELATIONAL_BODY_EX:
             goto C5_007_RELATIONAL_END;
         C5_007_RELATIONAL_END_EX:
             goto C5_002_RELATIONAL_EX;
        /**/
         C5_003_RELATIONAL_BODY:
             if((
  /*C10*/          (token->value.keyword_number==GREATER_THAN)
))
                    goto C5_004_GREATER_SYMBOL;
             if((
  /*C11*/          (token->value.keyword_number==LESS_THAN)
))
                    goto C5_005_LESS_SYMBOL;
             if((
  /*C12*/          (token->value.keyword_number==EQUAL_TO)
))
                    goto C5_006_EQUAL_SYMBOL;
         C5_004_GREATER_SYMBOL_EX:
         C5_005_LESS_SYMBOL_EX:
         C5_006_EQUAL_SYMBOL_EX:
             goto C5_003_RELATIONAL_BODY_EX;
        /**/
         C5_004_GREATER_SYMBOL:
  /*014*/       cond->relation.relation=GREATER_THAN;
             goto C5_004_GREATER_SYMBOL_EX;
        /**/
         C5_005_LESS_SYMBOL:
  /*015*/       cond->relation.relation=LESS_THAN;
             goto C5_005_LESS_SYMBOL_EX;
        /**/
         C5_006_EQUAL_SYMBOL:
  /*016*/       cond->relation.relation=EQUAL_TO;
```

```
            goto C5_006_EQUAL_SYMBOL_EX;
        /**/
         C5_007_RELATIONAL_END:
  /*006*/       eof=get_token(infile,token);
            goto C5_007_RELATIONAL_END_EX;
        /**/
         C5_008_VERB:
            goto C5_009_VERB_BODY;
         C5_009_VERB_BODY_EX:
            goto C5_019_VERB_END;
         C5_019_VERB_END_EX:
            goto C5_008_VERB_EX;
        /**/
         C5_009_VERB_BODY:
            if((
  /*C13*/          (token->value.keyword_number==GREATER)
))
                    goto C5_010_GREATER;
            if((
  /*C14*/          (token->value.keyword_number==LESS)
))
                    goto C5_013_LESS;
            if((
  /*C15*/          (token->value.keyword_number==EQUAL)
))
                    goto C5_016_EQUAL;
         C5_010_GREATER_EX:
         C5_013_LESS_EX:
         C5_016_EQUAL_EX:
            goto C5_009_VERB_BODY_EX;
        /**/
         C5_010_GREATER:
            goto C5_011_GREATER;
         C5_011_GREATER_EX:
            goto C5_012_THAN;
         C5_012_THAN_EX:
            goto C5_010_GREATER_EX;
        /**/
         C5_011_GREATER:
```

```
/*006*/       eof=get_token(infile,token);
          goto C5_011_GREATER_EX;
    /**/
     C5_012_THAN:
/*014*/       cond->relation.relation=GREATER_THAN;
          goto C5_012_THAN_EX;
    /**/
     C5_013_LESS:
          goto C5_014_LESS;
     C5_014_LESS_EX:
          goto C5_015_THAN;
     C5_015_THAN_EX:
          goto C5_013_LESS_EX;
    /**/
     C5_014_LESS:
/*006*/       eof=get_token(infile,token);
          goto C5_014_LESS_EX;
    /**/
     C5_015_THAN:
/*015*/       cond->relation.relation=LESS_THAN;
          goto C5_015_THAN_EX;
    /**/
     C5_016_EQUAL:
          goto C5_017_EQUAL;
     C5_017_EQUAL_EX:
          goto C5_018_TO;
     C5_018_TO_EX:
          goto C5_016_EQUAL_EX;
    /**/
     C5_017_EQUAL:
/*006*/       eof=get_token(infile,token);
          goto C5_017_EQUAL_EX;
    /**/
     C5_018_TO:
/*016*/       cond->relation.relation=EQUAL_TO;
          goto C5_018_TO_EX;
    /**/
     C5_019_VERB_END:
/*006*/       eof=get_token(infile,token);
```

```
            goto C5_019_VERB_END_EX;
        /**/
        /** skip the IS, take the NOT and the*/
        /** relational and build a relational part o*/
        /** f the condition*/
        /**/
         C3_001_KEYWORDS:
             goto C3_002_KSTART;
         C3_002_KSTART_EX:
             goto C3_003_POSS_IS;
         C3_003_POSS_IS_EX:
             goto C3_005_KEYWORD_GROUP;
         C3_005_KEYWORD_GROUP_EX:
         C3_001_KEYWORDS_A:
                 switch(g3_001_KEYWORDS_A){
                 case 0 : goto C3_001_EXIT01;break;
                 }
        /**/
         C3_002_KSTART:
  /*023*/      printf("relation bit\n");
            goto C3_002_KSTART_EX;
        /**/
         C3_003_POSS_IS:
            if((
  /*C05*/          (token->token_type==KEYWORD &&
               token->value.keyword_number==IS)
))
                  goto C3_004_IS;
         C3_004_IS_EX:
            goto C3_003_POSS_IS_EX;
        /**/
         C3_004_IS:
  /*006*/      eof=get_token(infile,token);
            goto C3_004_IS_EX;
        /**/
         C3_005_KEYWORD_GROUP:
            if((
  /*C06*/          (token->token_type==KEYWORD &&
               token->value.keyword_number==NOT)
```

```
))
                    goto C3_006_NOT_GROUP;
          if((
 /*C07*/          (token->token_type==KEYWORD ||
              token->token_type==RELATIONAL_OP)
))
                    goto C3_010_NOT_NOT_GROUP;
      C3_006_NOT_GROUP_EX:
      C3_010_NOT_NOT_GROUP_EX:
          goto C3_005_KEYWORD_GROUP_EX;
     /**/
      C3_006_NOT_GROUP:
          goto C3_007_NOT;
      C3_007_NOT_EX:
          goto C3_008_GROUP;
      C3_008_GROUP_EX:
          goto C3_006_NOT_GROUP_EX;
     /**/
      C3_007_NOT:
 /*006*/       eof=get_token(infile,token);
 /*017*/       cond->relation.not=YES_STATUS;
          goto C3_007_NOT_EX;
     /**/
      C3_008_GROUP:
          goto C3_009_GROUP_SUBTREE;
      C3_009_GROUP_SUBTREE_EX:
          goto C3_008_GROUP_EX;
     /**/
      C3_009_GROUP_SUBTREE:
               g5_001_GROUP_SUBTREE_A=0;
          goto C5_001_GROUP_SUBTREE;
      C5_001_EXIT02:
          goto C3_009_GROUP_SUBTREE_EX;
     /**/
      C3_010_NOT_NOT_GROUP:
          goto C3_011_GROUP;
      C3_011_GROUP_EX:
          goto C3_010_NOT_NOT_GROUP_EX;
     /**/
```

```
        C3_011_GROUP:
            goto C3_012_GROUP_SUBTREE;
        C3_012_GROUP_SUBTREE_EX:
            goto C3_011_GROUP_EX;
       /**/
        C3_012_GROUP_SUBTREE:
                g5_001_GROUP_SUBTREE_A=1;
            goto C5_001_GROUP_SUBTREE;
        C5_001_EXIT03:
            goto C3_012_GROUP_SUBTREE_EX;
       /**/
        C4_001_RHS:
            goto C4_002_RHS_START;
        C4_002_RHS_START_EX:
            goto C4_003_RHS_BODY;
        C4_003_RHS_BODY_EX:
            goto C4_007_RHS_END;
        C4_007_RHS_END_EX:
        C4_001_RHS_A:
                switch(g4_001_RHS_A){
                case 0 : goto C4_001_EXIT01;break;
                }
       /**/
        C4_002_RHS_START:
  /*022*/       printf("right hand side\n");getchar();
  /*021*/       cond->relation.not=NO_STATUS;
            goto C4_002_RHS_START_EX;
       /**/
        C4_003_RHS_BODY:
            if((
  /*C02*/         (token->token_type==NUMERIC)
))
                  goto C4_004_INTEGER;
            if((
  /*C03*/         (token->token_type==VARIABLE_NAME)
))
                  goto C4_005_VARIABLE;
            if((
  /*C04*/         (token->token_type==NON_NUMERIC_LITERAL)
```

```
))
                    goto C4_006_STRING;
        C4_004_INTEGER_EX:
        C4_005_VARIABLE_EX:
        C4_006_STRING_EX:
            goto C4_003_RHS_BODY_EX;
      /**/
        C4_004_INTEGER:
 /*018*/      cond->rhs.type=SIDE_LITERAL;
            cond->rhs.contents.numeric_literal=token->value.keyword_number;
 /*025*/      printf("numeric value \n");
            goto C4_004_INTEGER_EX;
      /**/
        C4_005_VARIABLE:
 /*009*/      node=look_for(token->value.var_values.unique,
                          token->value.var_values.hash,
                          symtab);
 /*020*/      cond->rhs.type=SIDE_VAR;
            cond->rhs.contents.variable=node;
 /*026*/      printf("variable\n");
            goto C4_005_VARIABLE_EX;
      /**/
        C4_006_STRING:
 /*019*/      cond->rhs.type=SIDE_NON_LITERAL;
            cond->rhs.contents.non_numeric_literal=token->value.string;
 /*028*/      printf("string2\n");
            goto C4_006_STRING_EX;
      /**/
        C4_007_RHS_END:
 /*006*/      eof=get_token(infile,token);
            goto C4_007_RHS_END_EX;
      /**/
        C2_001_LHS:
            goto C2_002_LHS_START;
        C2_002_LHS_START_EX:
            goto C2_003_LHS_BODY;
        C2_003_LHS_BODY_EX:
            goto C2_007_LHS_END;
        C2_007_LHS_END_EX:
```

```
      C2_001_LHS_A:
              switch(g2_001_LHS_A){
              case 0 : goto C2_001_EXIT01;break;
              }
     /**/
      C2_002_LHS_START:
 /*024*/      printf("lefthand side\n");
          goto C2_002_LHS_START_EX;
     /**/
      C2_003_LHS_BODY:
          if((
 /*C02*/          (token->token_type==NUMERIC)
))
                  goto C2_004_INTEGER;
          if((
 /*C03*/          (token->token_type==VARIABLE_NAME)
))
                  goto C2_005_VARIABLE;
          if((
 /*C04*/          (token->token_type==NON_NUMERIC_LITERAL)
))
                  goto C2_006_STRING;
      C2_004_INTEGER_EX:
      C2_005_VARIABLE_EX:
      C2_006_STRING_EX:
          goto C2_003_LHS_BODY_EX;
     /**/
      C2_004_INTEGER:
 /*003*/      cond->lhs.type=SIDE_LITERAL;
 /*008*/      cond->lhs.contents.numeric_literal=token->value.keyword_number;
 /*025*/      printf("numeric value \n");
          goto C2_004_INTEGER_EX;
     /**/
      C2_005_VARIABLE:
 /*009*/      node=look_for(token->value.var_values.unique,
                      token->value.var_values.hash,
                      symtab);
 /*013*/      cond->lhs.contents.variable=node;
          cond->lhs.type=SIDE_VAR;
```

```
/*026*/      printf("variable\n");
         goto C2_005_VARIABLE_EX;
    /**/
     C2_006_STRING:
/*012*/      cond->lhs.type=SIDE_NON_LITERAL;
         cond->lhs.contents.non_numeric_literal=token->value.string;
/*027*/      printf("string\n");
         goto C2_006_STRING_EX;
    /**/
     C2_007_LHS_END:
/*006*/      eof=get_token(infile,token);
         goto C2_007_LHS_END_EX;
    /**/
    /*   Data Analysis Map*/
    /**/
    /*                        ------------- Data Analysis By -------------*
    /**/
    /*                        BOX TYPE                OPERATION  ALLOCATION*
    /*  Tree name: CONDS-BUILD*/
    /**/
    /*                        Leaf      :  3          Operations:   9*/
    /*                                                Quits     :   0*/
    /*                        Selections:  0*/
    /*                        Sequences :  2*/
    /*                        Iterations:  1*/
    /*                        Backtracks:  0*/
    /*                        Subtrees  :  3   -->    LHS*/
    /*                                                KEYWORDS*/
    /*                                                RHS*/
    /**/
    /*  Tree name: GROUP-SUBTREE*/
    /**/
    /*                        Leaf      : 11          Operations:  11*/
    /*                                                Quits     :   0*/
    /*                        Selections:  3*/
    /*                        Sequences :  5*/
    /*                        Iterations:  0*/
    /*                        Backtracks:  0*/
    /*                        Subtrees  :  0*/
```

```
        /**/
        /*  Tree name: KEYWORDS*/
        /**/
        /*                          Leaf       :   3          Operations:   4*/
        /*                                                     Quits      :   0*/
        /*                          Selections:   2*/
        /*                          Sequences :   5*/
        /*                          Iterations:   0*/
        /*                          Backtracks:   0*/
        /*                          Subtrees  :   2    -->   GROUP-SUBTREE*/
        /*                                                   GROUP-SUBTREE*/
        /**/
        /*  Tree name: RHS*/
        /**/
        /*                          Leaf       :   5          Operations:  10*/
        /*                                                     Quits      :   0*/
        /*                          Selections:   1*/
        /*                          Sequences :   1*/
        /*                          Iterations:   0*/
        /*                          Backtracks:   0*/
        /*                          Subtrees  :   0*/
        /**/
        /*  Tree name: LHS*/
        /**/
        /*                          Leaf       :   5          Operations:  10*/
        /*                                                     Quits      :   0*/
        /*                          Selections:   1*/
        /*                          Sequences :   1*/
        /*                          Iterations:   0*/
        /*                          Backtracks:   0*/
        /*                          Subtrees  :   0*/
        /**/
        /**/
        /**/
veryend: ;
}
```