United States Department of Interior

Geological Survey

RASO: Program for renumbering FORTRAN source programs

by

George H. Harrach

Open File Report 80-1274

1980

Contents

Abstract

RASO (Renumbering in Ascending Sequential Order, hereafter only refered to by the name RASO) is a FORTRAN computer program that renumbers FORTRAN source programs. RASO has been renumbered many times during its refinements. It is useful in situations where the program statement numbering system has become confusing as a result of modifications, corrections, and program maintenance. RASO processes a program on a subroutine-by-subroutine basis, making the following changes:

1) Deletes blank cards[1].
2) Creates statement numbers in ascending order.
3) Checks for the presence of continuation cards and makes modifications between the first and last continuation cards, so as to make optimum use of the 72 characters per card. This change is made only to the following FORTRAN statements: "do", "go to", "if", "read", "write", "print", "punch", "call".
4) Processes programs written in upper,lower, or mixed case, leaving characters unchanged.
5) Renumbers "format" statements with a different sequence if wanted (optional).
6) Relocates the "format" statements to the end of routines if they are numbered differently (optional).

[1]/ The use of the word card is based on the meaning from standard FORTRAN. Depending on the computer in use, it can actually be a card or card-image.

Introduction

This report describes RASO, a FORTRAN computer program for renumbering and editing FORTRAN source programs whose statement numbering system has become confusing a result of modification and/or corrections.

Commonly source programs with poorly organized statement numbers become complex and difficult for a reader to follow in a logical concise manner. This can cause confusion and results in wasted time. When making many modifications to a program, the programmer sometimes finds it difficult to choose statement numbers that have not been used previously.

RASO provides a means by which a programmer can renumber the program systematically. The other alternative available to a programmer is to make the changes to the statement numbers by the use of a text editor, an extremely laborious and time-consumming process. RASO acts on the old FORTRAN program which is read and processed, subroutine-by-subroutine and a new version written with the statement numbers in ascending order and right justified in columns 1 through 5. If any blank cards are found they are removed.

Operation of the Program

RASO is a multiple-pass renumbering program written in FORTRAN. The program was written to be as machine independent as possible. There are, however, 3 machine-dependent subroutines; these are for opening, closing, and deleting files. RASO was written to process upper and/or lower case characters. The program was written in a modular form in order to make modification of the code straight forward.

RASO requires a compilable FORTRAN program as input data, and produces a compilable output. If the input is not compilable, the output will contain questionable code and may or may not be compilable. An "end" statement is expected to terminate pass 1, i.e. the creation of the statement-number table. If the "end" statement is missing from a subroutine being processed, RASO will treat the next subroutine as if it were part of the preceding one and continue numbering sequentially.

In pass 1, the card being processed is checked for a statement number. If one exists the number is changed and right justified to column 5. The old and the new numbers are placed in the statement-number table for later reference. When an "end" statement is encountered, pass 1 is terminated. In pass 1 "format" statements are handled differently if the "format" options are choosen.

In pass 2, references to old statement numbers are changed to correspond with the new statement numbers. After a statement is processed, it is written in the output file. After completion of pass 2, control is returned to pass 1. Pass 1 will resume if another subroutine follows. This process is repeated until all subroutines of the old program have been processed.

Program Description

RASO is composed of a main program, 14 subroutines and 1 block data subroutine. The function of the main program is to read the name of the program that is to be renumbered and the name to be given to the output program. The names are checked to see that the input and output files names are different. If the names are the same the user will be asked for another output file name. This is done to preserve the integrity of the original program in case of user error. The beginning statement number and the statement number increment to be used in the renumbering is read. If "format" statement changes are to be made the beginning "format" statement number and the "format" statement number increment is read last.

Subroutine "kactio" is the primary routine for changing the statement numbers that are located in the text section. In this subroutine, the text section is scanned for FORTRAN statement types, which contain references to statement numbers. The statement numbers are then checked to see if they occur in the statement-number table, which is created in the main program. If it is found in the table, the statement number is changed.

Subroutine "endec" is for the processing of "encode" and "decode" statements. This was written as a subroutine so that it could more easily be modified for different forms of these statements. The forms processed by RASO are "encode(c,n)list" and "decode(c,n)list", here "n" is the statement number reference to be processed.

Subroutine "kopen", "kclose" and "kremov" are used for file handling, and are machine dependent. The versions listed at the back of this report are for the Honeywell Multics computer.

Restrictions

The following restrictions apply to the input program that is to be renumbered:

1) Statement numbers are limited to a maximum of 4 digits in columns 1 through 5, with no embedded blanks.
2) Statement are located between columns 1 and 72.
3) Comment statements are delineated by either an upper or lower case "c" in column 1.
4) Continuation cards are delineated by a non-blank character in column 6.
5) No multiple statements per card.
6) If the "format" statements are to be numbered differently or numbered differently and relocated. The use of the name "format" should not be used as a variable

Listing of example program

```
c
c.... Input string divider
c
      data kblank,kcomma,kdash/1h ,1h,,1h-/
      dimension input(100),iary1(50),iary2(50)

      write(0,897)
897   format("Enter rows selected  :")
      read(0,10)input
 10   format(100a1)

      istart=1
      ih=1
      ii=0
      do 450 ia=1,100,1
      ic=101-ia
      ili=input(ic)
      if(ili-kblank)340,450,340
  450 continue
340   ia=ic+1
      input(ia)=kcomma
80    do 20 ia=istart,100,1
      ij=input(ia)
      if(ij-kcomma)30,40,30
30    if(ij-kdash )20,130,20
20    continue
      go to 70
130   ii=1
40    ik=ia
50    ik=ik-1
      ij=input(ik)
      if(ij-kblank)60,50,60
60    ix=ik-istart+1
      in=0
      call icon(input(istart),ix,in)
      if(ii)160,170,140
140   iary1(ih)=in
      ii=-1
      go to 150
160   iary2(ih)=in
      ii=0
      ih=ih+1

      go to 150
170   iary1(ih)=in
      iary2(ih)=in
      ih=ih+1
150   istart=ia+1
      ia=0
      go to 80
   70 if(ih)90,120,90
   90 ih=ih-1
```

```
      write(0,943)
943   format(/)
      do 100 ia=1,ih,1
      write(0,110)iary1(ia),iary2(ia)

110   format(1i4,2x,"-",2x,1i4)
100   continue
120   stop
      end
      subroutine icon(ilane,iu,il)
c
c.... Conversion of characters to an integer.
c
      dimension ilane(iu),inum(10)
      data iblank/1h /

      data inum/1h0,1h1,1h2,1h3,1h4,1h5,1h6,1h7,1h8,1h9/
      num=0
      do 20 ia=1,iu,1
      ila=ilane(ia)
      if(ila-iblank)10,20,10
10    do 30 ib=1,10,1
      inu=inum(ib)
      if(ila-inu)30,15,30
15    num=(num*10)+ib-1
30    continue

20    continue
      il=num
      return
      end
```

Example 1

Below is an example execution of the program. The name of the program to renumbered is "qwer.fortran", a compileable program. The statement numbers are not right justified and are out of sequence. In this example the format statements will be treated the same as all other statements.

```
          ****     FORTRAN Renumbering     ****
Program to renumber FORTRAN IV programs. The original
program should be compilable. If not, this program will
give questionable results !! The program requires that
input programs have statement numbers of 1 to 4 digits

Enter the program name to be renumbered      :qwer.fortran

Enter name of output file to be created      :qwer1.fortran

Enter beginning statement number             :10

Enter statement number increment             :10

                    Format Statements
Do you want to renumber format statements, starting at
another value, then the other the statement numbers
Enter your choice please                     :no

Processing of routine number :   1  starting
```

Listing of program from example 1

```
c
c.... Input string divider
c
      data kblank,kcomma,kdash/1h ,1h,,1h-/
      dimension input(100),iary1(50),iary2(50)
      write(0,10)
   10 format("Enter rows selected  :")
      read(0,20)input
   20 format(100a1)
      istart=1
      ih=1
      ii=0
      do 30 ia=1,100,1
      ic=101-ia
      ili=input(ic)
      if(ili-kblank)40,30,40
   30 continue
   40 ia=ic+1
      input(ia)=kcomma
   50 do 70 ia=istart,100,1
      ij=input(ia)
      if(ij-kcomma)60,90,60
   60 if(ij-kdash )70,80,70
   70 continue
      go to 160
   80 ii=1
   90 ik=ia
  100 ik=ik-1
      ij=input(ik)
      if(ij-kblank)110,100,110
  110 ix=ik-istart+1
      in=0
      call icon(input(istart),ix,in)
      if(ii)130,140,120
  120 iary1(ih)=in
      ii=-1
      go to 150
  130 iary2(ih)=in
      ii=0
      ih=ih+1
      go to 150
  140 iary1(ih)=in
      iary2(ih)=in
      ih=ih+1
  150 istart=ia+1
      ia=0
      go to 50
  160 if(ih)170,210,170
  170 ih=ih-1
      write(0,180)
  180 format(/)
      do 200 ia=1,ih,1
```

```
      write(0,190)iary1(ia),iary2(ia)
  190 format(1i4,2x,"-",2x,1i4)
  200 continue
  210 stop
      end
      subroutine icon(ilane,iu,il)
c
c.... Conversion of characters to an integer.
c
      dimension ilane(iu),inum(10)
      data iblank/1h /
      data inum/1h0,1h1,1h2,1h3,1h4,1h5,1h6,1h7,1h8,1h9/
      num=0
      do 40 ia=1,iu,1
      ila=ilane(ia)
      if(ila-iblank)10,40,10
   10 do 30 ib=1,10,1
      inu=inum(ib)
      if(ila-inu)30,20,30
   20 num=(num*10)+ib-1
   30 continue
   40 continue
      il=num
      return
      end
```

Example 2

Below is an example execution of the program. The name of the program to renumbered is "qwer.fortran", a compileable program. The statement numbers are not right justified and are out of sequence. In this example the format statements will be numbered differently.

```
          ****     FORTRAN Renumbering     ****
Program to renumber FORTRAN IV programs. The original
program should be compilable. If not, this program will
give questionable results !! The program requires that
input programs have statement numbers of 1 to 4 digits

Enter the program name to be renumbered      :qwer

Enter name of output file to be created      :qwer2

Enter beginning statement number             :10

Enter statement number increment             :10

                    Format Statements
Do you want to renumber format statements, starting at
another value, then the other the statement numbers
Enter your choice please                     :yes

Enter beginning statement number             :900

Enter statement number increment             :10

Do you want to relocate the format statements to the end
of the routines                              ?no

Processing of routine number :  1  starting
```

Listing of program from example 2

```
c
c.... Input string divider
c
      data kblank,kcomma,kdash/1h ,1h,,1h-/
      dimension input(100),iary1(50),iary2(50)
      write(0,900)
  900 format("Enter rows selected  :")
      read(0,910)input
  910 format(100a1)
      istart=1
      ih=1
      ii=0
      do 10 ia=1,100,1
      ic=101-ia
      ili=input(ic)
      if(ili-kblank)20,10,20
   10 continue
   20 ia=ic+1
      input(ia)=kcomma
   30 do 50 ia=istart,100,1
      ij=input(ia)
      if(ij-kcomma)40,70,40
   40 if(ij-kdash )50,60,50
   50 continue
      go to 140
   60 ii=1
   70 ik=ia
   80 ik=ik-1
      ij=input(ik)
      if(ij-kblank)90,80,90
   90 ix=ik-istart+1
      in=0
      call icon(input(istart),ix,in)
      if(ii)110,120,100
  100 iary1(ih)=in
      ii=-1
      go to 130
  110 iary2(ih)=in
      ii=0
      ih=ih+1
      go to 130
  120 iary1(ih)=in
      iary2(ih)=in
      ih=ih+1
  130 istart=ia+1
      ia=0
      go to 30
  140 if(ih)150,170,150
  150 ih=ih-1
      write(0,920)
  920 format(/)
      do 160 ia=1,ih,1
```

```
      write(0,930)iary1(ia),iary2(ia)
  930 format(1i4,2x,"-",2x,1i4)
  160 continue
  170 stop
      end
      subroutine icon(ilane,iu,il)
c
c.... Conversion of characters to an integer.
c
      dimension ilane(iu),inum(10)
      data iblank/1h /
      data inum/1h0,1h1,1h2,1h3,1h4,1h5,1h6,1h7,1h8,1h9/
      num=0
      do 40 ia=1,iu,1
      ila=ilane(ia)
      if(ila-iblank)10,40,10
   10 do 30 ib=1,10,1
      inu=inum(ib)
      if(ila-inu)30,20,30
   20 num=(num*10)+ib-1
   30 continue
   40 continue
      il=num
      return
      end
```

Example 3

Below is an example execution of the program. The name of the program to renumbered is "qwer.fortran", a compileable program. The statement numbers are not right justified and are out of sequence. In this example the format statements will be numbered differently and relocated to the end of routines.

```
          ****     FORTRAN Renumbering     ****
Program to renumber FORTRAN IV programs. The original
program should be compilable. If not, this program will
give questionable results !! The program requires that
input programs have statement numbers of 1 to 4 digits

Enter the program name to be renumbered      :qwer

Enter name of output file to be created      :qwer3

Enter beginning statement number             :10

Enter statement number increment             :10

                    Format Statements
Do you want to renumber format statements, starting at
another value, then the other the statement numbers
Enter your choice please                     :yes

Enter beginning statement number             :900

Enter statement number increment             :10

Do you want to relocate the format statements to the end
of the routines                              ?yes

Processing of routine number :  1  starting
```

Listing of program from example 3

```
c
c.... Input string divider
c
      data kblank,kcomma,kdash/1h ,1h,,1h-/
      dimension input(100),iary1(50),iary2(50)
      write(0,900)
      read(0,910)input
      istart=1
      ih=1
      ii=0
      do 10 ia=1,100,1
      ic=101-ia
      ili=input(ic)
      if(ili-kblank)20,10,20
   10 continue
   20 ia=ic+1
      input(ia)=kcomma
   30 do 50 ia=istart,100,1
      ij=input(ia)
      if(ij-kcomma)40,70,40
   40 if(ij-kdash )50,60,50
   50 continue
      go to 140
   60 ii=1
   70 ik=ia
   80 ik=ik-1
      ij=input(ik)
      if(ij-kblank)90,80,90
   90 ix=ik-istart+1
      in=0
      call icon(input(istart),ix,in)
      if(ii)110,120,100
  100 iary1(ih)=in
      ii=-1
      go to 130
  110 iary2(ih)=in
      ii=0
      ih=ih+1
      go to 130
  120 iary1(ih)=in
      iary2(ih)=in
      ih=ih+1
  130 istart=ia+1
      ia=0
      go to 30
  140 if(ih)150,170,150
  150 ih=ih-1
      write(0,920)
      do 160 ia=1,ih,1
      write(0,930)iary1(ia),iary2(ia)
  160 continue
  170 stop
```

```
  900 format("Enter rows selected  :")
  910 format(100a1)
  920 format(/)
  930 format(1i4,2x,"-",2x,1i4)
      end
      subroutine icon(ilane,iu,il)
c
c.... Conversion of characters to an integer.
c
      dimension ilane(iu),inum(10)
      data iblank/1h /
      data inum/1h0,1h1,1h2,1h3,1h4,1h5,1h6,1h7,1h8,1h9/
      num=0
      do 40 ia=1,iu,1
      ila=ilane(ia)
      if(ila-iblank)10,40,10
   10 do 30 ib=1,10,1
      inu=inum(ib)
      if(ila-inu)30,20,30
   20 num=(num*10)+ib-1
   30 continue
   40 continue
      il=num
      return
      end
```

```
c     ****************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c                       Program Written August 1980
c                          Revised November 1980
c    ****************************************************************
c....Program to Renumber FORTRAN IV  Programs.
c      Accepts programs written in upper, lower, or mixed case and
c      numbers format differently if wanted.
c    ****************************************************************
c
c....Subroutines necessary for operation:
c
c      kactio  :  Controls the changing of statement numbers.
c      kproc   :  Processes changes to the following types of
c                 FORTRAN IV statements:  "read", "write",
c                 "print", "punch", "if", "go to ".
c      kcheck  :  Checks infile and outfile name for a match.
c      kwrite  :  Writes processed statements into output file.
c      krcont  :  Read continuation lines if any exist.
c      kproio  :  Controls program output.
c      ktabcp  :  Comparison of statement numbers to number table.
c      kintch  :  Converts characters into an integer variable.
c      kchint  :  Converts an integer variable into characters.
c      klinfx  :  Inserts new statement numbers in place of old numbers.
c      kendec  :  Controls processing of "encode" and "decode"
c
c....Machine dependent routines.
c
c      kopen   :  Opens files.
c      kremov  :  Deletes files.
c      kclose  :  Closes files.
c
c....Files used:
c
c      Unit 10 :  Input file.
c      Unit 15 :  Scratch file.
c      Unit 20 :  Output file.
c    ****************************************************************
       common ioldy(1004) ,inewy(1004) ,iwswl
       dimension inline(72) ,ilane(4) ,iline(72),infile(8),outfile(8)
       dimension jline(72)
       common /ispx/cname(8),dname(8)
       external kopen(descriptors)
       common /kchs/khy   ,kly   ,khc   ,klc   ,khe   ,kle   ,
      &             khn   ,kln   ,khd   ,kld   ,kho   ,klo   ,
      &             khp   ,klp   ,kha   ,kla   ,khl   ,kll   ,
      &             kblank,kandpr,kcomma,kdolla,kequal,kastr ,
      &             krpren,klpren
```

```
      common/rt/iskip1(10), iskip2(10), iskip3( 5), iskip4( 5),
     &          iskip5( 6), iskip6( 6), iskip7( 7), iskip8( 7),
     &          iskip9( 6), iskipa( 6), iskipb( 5), iskipc( 5),
     &          iskipd( 9), iskipe( 9), ichars( 4), iskipf( 4),
     &          iskipg( 4), ipctbc(18), iskiph( 2), iskipi( 2),
     &          iskipj( 5), iskipk( 5), iskipl( 5), iskipm( 5),
     &          iskipn( 2), iskipo( 2), iskipp( 4), iskipq( 4),
     &          inumbr(11), iskips( 6), iskipt( 6), iskipu( 5),
     &          iskipv(05), iskipw( 4), iskipx( 4), ispeci( 9),
     &             ien(04),   iend(04),   inum(10)
c
c....Variables:
c      iterm   = Unit number of the terminal.
c      isub    = Counter to indicate the progress of the program.
c      imin    = Lowest statement number to use.
c      incr    = Increment value for the statement number.
c
       iwswl  = 0
       ifld   = 0
       ihf    = 0
       iyrf   = 0
       irec   = 0
       ifi    = 0
       ifm    = 0
       isub   = 0
       iterm  = 0
       ivp    = 0
       istpw  = 0
   10  write (iterm,9000)
 9000  format (//,"           ****    FORTRAN Renumbering    ****",
     &/,"Program  to  renumber FORTRAN IV programs.  The original",
     &/,"program should be compilable.  If not, this program will",
     &/,"give questionable results !!  The program  requires that",
     &/,"input  programs have  statement numbers of 1 to 4 digits")
       write (iterm,9010)
 9010  format (/,"Enter  the  program  name  to  be  renumbered     :",$)
       read (iterm,9020) infile
       numct = 0
 9020  format (8a4)
       iero = 0
       call kopen(10,iero,infile,1)
       if (iero) 40,40,20
   20  write (iterm,9030)
 9030  format (/,"File cannot be opened. Do you want to try another :",$)
       read (iterm,9040) iansl
 9040  format (1a1)
       call kclose(10)
       if (iansl-khy) 30,10,30
   30  if (iansl-kly) 920,10,920
   40  write (iterm,9050)
 9050  format (/,"Enter  name  of  output  file   to  be  created   :",$)
       read (iterm,9020) outfile
```

```
c
c.... If input file name is same as output file name, then flag an error
c     condition.
c
      ieroq = 0
      call kcheck(infile,outfile,ieroq)
      if (ieroq) 50,50,60
   50 write (iterm,9060)
 9060 format (///,"Error! Input file name is the same as Output file",
     &/,"name.----------------------------------------------------")
      go to 40
   60 call kopen(20,iero,outfile,1)
      if (iero) 70,70,100
   70 write (iterm,9070)
c
c.... Check to see if output file name already exists.  If so, ask
c     if the new file is to replace the old.
c
 9070 format (/,"File already exists.  Do  you  want to overwrite  ?",$)
      read (iterm,9040) iansl
      if (iansl-khy) 80,90,80
   80 if (iansl-kly) 40,90,40
   90 call kclose(20)
      call kremov(outfile)
      go to 110
  100 call kclose(20)
  110 write (iterm,9080)
 9080 format (/,"Enter  beginning  statement  number                :",$)
      read (iterm,9090,err=110) imin
 9090 format (v)
  120 write (iterm,9100)
 9100 format (/,"Enter  statement  number  increment                :",$)
      read (iterm,9090,err=120) incr
      if (incr) 130,130,140
  130 write (iterm,9110)
 9110 format (/,"Your  Choice  is not accepted due to an error;  try",
     &/,"again.  Increment is either zero or negative.--------------")
      go to 120
c
c.... Open files for processing of data.
c
  140 write (iterm,9120)
 9120 format (/"                        Format Statements"
     &/,"Do  you  want to renumber format statements, starting at",
     &/,"another  value,  then  the other  the statement  numbers",
     &/,"Enter your choice please                                :",$)
      read (iterm,9040) ifa
      if (ifa-kly) 150,160,150
  150 if (ifa-khy) 220,160,220
  160 write (iterm,9080)
c
c.... Read the format statement base level.
c
```

```
      read (iterm,9090,err=160) ifm
      if (ifm) 160,160,170
  170 write (iterm,9100)
c
c.... Read the format statement increment level.
c
      read (iterm,9090,err=170) ifi
      if (ifi) 170,170,180
c
c.... "irec" = 1 Means format statements have different numbering.
c
  180 irec = 1
      write (iterm,9130)
 9130 format (/,"Do you want to relocate the format statements to the",
     &" end ",/,"of the routines                                    ?",$)
      read (iterm,9040) ifrs
      if (ifrs-kly) 190,200,190
  190 if (ifrs-khy) 210,200,210
  200 iyrf = 1
      call kopen(25,iero,dname,-1)
  210 ifn = ifm
  220 call kopen(20,iero,cname,-1)
      call kopen(15,iero,outfile,-1)
      inc    = 1
      icont  = 0
      numcds = 0
      nbr    = imin
c
c.... The program will read a maximum of 10,000 cards per run, including
c     the main program, subroutines, and/or functions.
c
      do 880 ia = 1,10000,1
c
c.... Read one card(max 72 characters) & process it.
c
      read (10,9140,end=890) inline
c
c.... Count the number of characters in the card. Check card columns
c     from the right to the left for non-blank characters.
c
      do 230 ixy = 1,73,1
      ixe = 73 -ixy
      ilin = inline(ixe)
      ilx = ilin -kblank
      if (ilx) 240,230,240
  230 continue
  240 if (ixe) 250,880,250
  250 num = 0
      inpr= 0
 9140 format (72a1)
      ilin = inline(1)
      if (ilin - khc) 260,290,260
  260 if (ilin - klc) 270,290,270
  270 ilin = inline(6)
```

```
      if (ilin-kblank) 280,300,280
  280 icont = 1
c
c.... kandpr is a "&", if for any reason another continuation card
c     symbol is needed this can be changed.
c
      inline(6) = kandpr
  290 inpr = 1
  300 if (inc) 310,310,360
c
c.... If "iyrf" is 1,  Then move the format statements to the routine
c     end.
c
  310 if (iyrf-1) 320,330,320
  320 write (20,9150) ifc,numchs,icont,iline
      go to 350
c
c.... If "ihf" is 1,  Then there is a format statement in the buffer.
c
  330 if (ihf-1) 320,340,320
  340 write (25,9150) ifc,numchs,icont,iline
c
c.... If "ifld" is 1,  Then unit 25 has been loaded with a format.
c
      ifld  = 0
  350 icont = 0
      ifc   = 0
  360 if (istpw) 370,370,850
 9150 format (3i2,72a1)
  370 do 380 id = 1,72,1
      iline(id) = inline(id)
  380 continue
c
c     Set the character count, "nch" equal to the reverse counter.
c
      numchs = ixe
  390 inc    = 0
      if (inpr) 400,400,880
  400 do 440 ib = 1,5,1
      ilin = iline(ib)
      if (ilin-kblank) 410,440,410
  410 do 430 ic = 1,10,1
      inm = inum(ic)
      if (ilin-inm) 430,420,430
  420 num = (num*10) + ic -1
  430 continue
  440 continue
c
c.... If "num" not zero then a line number exists.
c
      if (num) 450,680,450
```

```
c
c.... If the value of "num" (old statement number) is different from
c     the value of "nbr" or "ifn" (internal counter), then change old
c     statement number to agree with "nbr" or "ifn", creating new
c     statement number.  Otherwise statement number does not need to
c     be checked.
c
  450 if (irec) 590,590,460
c
c.... Find the first non blank position, starting in card column 7.
c
  460 do 470 ic = 7,20,1
      ilin = iline(ic)
      if (ilin-kblank) 480,470,480
  470 continue
  480 ipa  = 0
      isum = 0
      icc = ic + 5
      do 510 id = ic,icc,1
      ilin = iline(id)
      ipa = ipa +1
      if (ilin-iskip5(ipa)) 490,500,490
  490 if (ilin-iskip6(ipa)) 510,500,510
  500 isum = isum +1
  510 continue
c
c.... If there is a "format" statement then "isum" is equal to  6.
c
      if (isum-6) 590,520,590
  520 ilin = iline(ic+6)
      if (ilin-kblank) 530,540,530
  530 if (ilin-klpren) 590,540,590
  540 if (num-ifn) 550,560,550
  550 numct = numct+1
      ioldy(numct) = num
      inewy(numct) = ifn
  560 do 570 ic = 1,4,1
      iline(ic+1) = kblank
      ilane(ic)   = kblank
  570 continue
      iline(1) = kblank
      ixx = ifn
      call kintch(ixx,ilane)
      id = 6
      ie = 5
      do 580 ic = 1,4,1
      id = id-1
      ie = ie-1
      iline(id) = ilane(ie)
  580 continue
      ifn = ifn + ifi
c
c.... Turn the format statement switch on, if a "format" was found.
c
```

```
      ihf = 1
      go to 690
  590 if (num - nbr) 600,610,600
  600 numct = numct + 1
      ihf = 0
c
c.... If "num" & "nbr" are different, then enter these values in
c     tables.
c
      ioldy(numct) = num
      inewy(numct) = nbr
c
c.... Initialize the values of "ilane" and the location of the old
c     statement number in card columns 1 to 5 of "iline".
c
  610 do 620 ic = 1,4,1
      iline(ic + 1) = kblank
      ilane(ic) = kblank
  620 continue
c
c.... Check for a "D" or "d" in card column 1.  If this
c     character is present it is considered a debug feature
c     for certain compilers.
c
      ilin = iline(1)
      if (ilin-khd) 630,650,630
  630 if (ilin-kld) 640,650,640
  640 iline(1) = kblank
  650 ixx = nbr
      call kintch(ixx,ilane)
      id = 6
      ie = 5
c
c.... Move the contents of the buffer "ilane" into "iline".
c
      do 660 ic = 1,4,1
      id = id -1
      ie = ie -1
      iline(id) = ilane(ie)
  660 continue
  670 nbr = nbr + incr
  680 ihf = 0
  690 do 700 iyz = 7,72,1
      ilin = iline(iyz)
      if (ilin-kblank) 710,700,710
  700 continue
  710 if (iyz-72) 730,720,730
  720 inc = 1
      go to 880
  730 ifc = iyz
      ilin = iline(iyz)
```

```
c
c.... Check for an "end" statement.  If found, terminate the
c     pass and resume when control is returned to this routine.
c
      if (ilin-khe) 740,750,740
  740 if (ilin-kle) 880,750,880
  750 ilin = iline(ifc+1)
      if (ilin-khn) 760,770,760
  760 if (ilin-kln) 880,770,880
  770 ilin = iline(ifc+2)
      if (ilin-khd) 780,790,780
  780 if (ilin-kld) 880,790,880
  790 ilin = iline(ifc+3)
      if (ilin-kblank) 880,800,880
c
c.... if "iyrf" is 1,  Then close the format file and open it for
c     reading and writing into the "TMP" file.
c
  800 if (ifld-1) 840,810,840
  810 call kclose(25)
      call kopen(25,iero,dname,1)
      do 820 irth = 1,10000,1
      read (25,9150,end=830) jfc,jnumch,jcont,jline
      write (20,9150) jfc,jnumch,jcont,jline
  820 continue
  830 call kclose(25)
      ifld  = 0
  840 istpw = 1
      go to 880
  850 call kclose(20)
      call kactio(numct)
      call kopen(20,iero,cname,-1)
      if (iyrf-1) 870,860,870
  860 call kopen(25,iero,dname,-1)
      ihf = 0
  870 iwswl  = 0
      ifld   = 0
      ivp    = 0
      numct  = 0
      istpw  = 0
      nbr    = imin
      num    = 0
      ixx    = 0
      isub   = isub + 1
      numcds = 0
      ifn    = ifm
c
c.... Print progress on terminal for user.
c
      write (iterm,9160) isub
 9160 format (/,"Processing  of  routine  number :",li4,2x,"starting")
      go  to 370
  880 continue
  890 write (20,9150) ifc,numchs,icont,iline
```

```
c
c.... Close all scratch files.
c
      call kclose(20)
      if (iyrf-1) 910,900,910
  900 call kclose(25)
  910 call kactio(numct)
      if (ivp) 930,920,930
c
c.... Close the files.
c
  920 call kclose(15)
  930 call kclose(10)
      call kremov(cname)
      if (iyrf-1) 950,940,950
  940 call kremov(dname)
  950 continue
      stop
      end
```

```
      subroutine kactio(numct)
c
c     ****************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c    ****************************************************************
c
c.... Purpose:  Main renumbering subroutine.
c
      common ioldy(1004), inewy(1004),iwswl
      dimension iline(1050),ilane(4)
      common /ispx/cname(8),dname(8)
      external kopen(descriptors)
      data kdq,ksq/1h",1h'/
      common /kchs/khy   ,kly   ,khc   ,klc   ,khe   ,kle   ,
     &             khn   ,kln   ,khd   ,kld   ,kho   ,klo   ,
     &             khp   ,klp   ,kha   ,kla   ,khl   ,kll   ,
     &             kblank,kandpr,kcomma,kdolla,kequal,kastr ,
     &             krpren,klpren
      common/rt/iskip1(10), iskip2(10), iskip3( 5), iskip4( 5),
     &          iskip5( 6), iskip6( 6), iskip7( 7), iskip8( 7),
     &          iskip9( 6), iskipa( 6), iskipb( 5), iskipc( 5),
     &          iskipd( 9), iskipe( 9), ichars( 4), iskipf( 4),
     &          iskipg( 4), ipctbc(18), iskiph( 2), iskipi( 2),
     &          iskipj( 5), iskipk( 5), iskipl( 5), iskipm( 5),
     &          iskipn( 2), iskipo( 2), iskipp( 4), iskipq( 4),
     &          inumbr(11), iskips( 6), iskipt( 6), iskipu( 5),
     &          iskipv(05), iskipw( 4), iskipx( 4), ispeci( 9),
     &             ien(04),   iend(04),   inum(10)
c
c.... Variables:
c     nch       = number of characters in buffer "iline".
c     ictc      = switch, read from scratch file. Indicates
c                 if this card is followed by a continuation
c                 card. 1 = yes  ,  0 = no.
c     isum      = integer counter for character type match.
c     isun      = integer counter for character type match.
c     ixx       = number of characters in buffer "ilane".
c     iterm     = unit number of the user"s terminal.
c     iswv      = statement number change switch from routine "ktabcp"
c                  1 means there is a change to be processed.
c     isw4      = switch for the main buffer status.
c                  1 = buffer not written , 0 = buffer written.
c     kqusw     = quote switch for "if" statement processing.
c     iisw      = processing switch for "if" statement.
c.... Array's;
c     ipctbc    : are the characters for the first match after the card
c                 column number 7 (non blank).
c     iline     : main buffer for the processing of the card.
c
      iwswl     = 0
      ispswc    = 1
```

```
      iterm     = 0
      iisw      = 0
      iunit     = 15
c
c.... Open scratch file for input.
c
      call kopen(20,iero,cname,1)
      do 1490 iru = 1,10000,1
c
c.... Read one line from scratch file.
c
      read (20,9000,end=1500) icpos,nch,ictc,(iline(iss) ,iss = 1,72)
c
c.... If "nch" number of characters in the card is equal to 0 ,
c     skip the card
c
   10 if (ispswc) 20,90,20
 9000 format (3i2,72a1)
c
c.... Check card column 1 for characters, "C" or "c".
c     For "common" and "character"  statements.
c
   20 ilin = iline(1)
      if (ilin-khc) 30,90,30
   30 if (ilin-klc) 40,90,40
c
c.... Check card column number 6 to see if it is blank.
c     If not blank then assume to be a continuation.  All
c     continuations that are important for processing are read
c     in routine "krcont".
c
   40 ilin=iline(6)
      if (ilin-kblank) 90,50,90
   50 ix   = 0
      num  = 0
      isum = 0
      ncha = nch
c
c.... "icpos" Is the first non-blank position starting at or after
c     card column 7. Check to see if the character at that position
c     matches a character in array "ipctbc".
c
      ilin = iline(icpos)
   60 do  70 ic = 1,18,1
      icptc = ipctbc(ic)
      if (ilin - icptc)  70, 80, 70
   70 continue
      go to 1370
c
c.... Process according to character found in "icpos" position of
c     buffer "iline".
c
   80 go to (120,120, 330, 330,1370,1370,570,570,620,620,920,920,980,
     &980,1030,1030,1090,1090) ,ic
```

```
c
c.... This area for the writing of statements onto file 15.
c
   90 write (iunit, 9010) (iline(ki) ,ki = 1,nch)
 9010 format (72a1)
      if (ictc)  100, 110, 100
  100 ispswc = 0
      go to 1490
  110 ispswc = 1
      go to 1490
c
c.... This section for characters "D" or "d".
  120 isum = 0
      ist = icpos -1
c
c.... Check for "DO" statement.
c
      do 150 ic = 1,2,1
      ist = ist + 1
      ilin = iline(ist)
      iskp = iskiph(ic)
      if (ilin-iskp) 130,140,130
  130 iskp = iskipi(ic)
      if (ilin-iskp)150,140,150
  140 isum = isum+1
  150 continue
      if (isum - 2) 160,210,160
c
c.... Search for "decode" statement now process.
c
  160 id = icpos -1
      isum = 0
      do 190 ic = 1,7,1
      id = id + 1
      ilin = iline(id)
      iskp = iskips(ic)
      if (ilin-iskp) 170,180,170
  170 iskp = iskipt(ic)
      if (ilin-iskp) 190,180,190
  180 isum = isum+1
  190 continue
      if (isum - 6) 1370,200,1370
c
c.... Have "decode" statement now process
c
  200 call krcont(nch,ictc,iline,20)
      call kendec(iline,isw4,numct,nch,icpos)
      if (isw4 - 1) 1490,1370,1490
  210 isum = 0
      ixee = icpos + 2
      ixeee = icpos + 13
      do 240 ic = ixee,ixeee,1
      ilin = iline(ic)
      do 230 id = 1,11,1
```

```
      inm = inumbr(id)
      if (ilin-inm) 230,220,230
  220 if (id - 11) 250,240,250
  230 continue
  240 continue
      go to 1370
  250 isum = 0
c
c.... "DO" statement found now process.
c
      do 280 id = ic,ic + 3,1
c
c.... Search for positions where statement number begins and ends.
c
      ilin = iline(id)
      do 270 ie = 1,10,1
      inm = inumbr(ie)
      if (ilin-inm) 270, 260,270
  260 isum = isum + 1
  270 continue
  280 continue
      ista = ic
      istb = ic + isum -1
c
c.... Initialize buffer "ilane" to blanks.
c
      do 290 id = 1,4,1
      ilane(id) = kblank
  290 continue
      iaa = 4
      ido = istb
c
c.... Move buffer "iline" (where statement number is located) to buffer
c     "ilane".
c
      do 300 id = ista,istb,1
      ilane(iaa) = iline(ido)
      iaa = iaa -1
      ido = ido -1
  300 continue
c
c.... Convert characters in "ilane" to an integer variable.
c
      call krcont(nch,ictc,iline,20)
      call kchint(ix,ilane)
c
c.... Compare statement number"ix" against table of statement
c     numbers to be changed.
c
      call ktabcp(numct,ix,iswv,ixx,ilane)
c
c.... If statement number is to be changed switch then "iswv" is 1.
c
      if (iswv-1) 320,310,320
```

```
  310 call klinfx(iline,nch,ixx,ilane,ista,istb)
  320 call kproio(iunit,iline,nch)
      go to 1490
c
c.... Section for characters "C"or"c".
c
c.... Check for "$" return branch in a "call" statement.
c     This symbol can be changed to another one that a different
c     machine might use.  By changing the value of "kdolla" in
c     the block common.
c
  330 ia = icpos -1
  340 ilin = iline(ia + 1)
      if (ilin-khc)  350, 360, 350
  350 if (ilin-klc) 1480, 360,1480
  360 ilin = iline(ia + 2)
      if (ilin-kha)  370, 380, 370
  370 if (ilin-kla) 1480, 380,1480
  380 ilin = iline(ia + 3)
      ilinea = iline(ia + 4)
      if (ilin-ilinea) 1480, 390,1480
  390 if (ilin-khl)  400, 410, 400
  400 if (ilin-kll) 1480, 410,1480
  410 iva = ia + 5
  420 call krcont(nch,ictc,iline,20)
      do  430 ia = iva,nch,1
      ilin = iline(ia)
c
c     Next line of code contains character "kdolla".
c
      if (ilin-kdolla)  430, 440, 430
  430 continue
      go to 1480
  440 iva = ia + 1
      do  470 ic = iva,iva + 10,1
      ilin = iline(ic)
      do  460 id = 1,11,1
      ilina = inumbr(id)
      if (ilin-ilina)  460, 450, 460
  450 if (id - 11)  480, 470, 480
  460 continue
  470 continue
      go to 1480
  480 ista = ic
      isum = 0
      do  510 id = ic,ic + 3,1
      ilin = iline(id)
      do  500 ie = 1,10,1
       inumb = inumbr(ie)
      if (ilin-inumb)  500, 490, 500
  490 isum = isum + 1
  500 continue
  510 continue
      istb = ic + isum -1
```

```
      iixx = istb + 1 -ista
      if (iixx - 4)  520, 520,1480
  520 do  530 id = 1,4,1
      ilane(id) = kblank
  530 continue
      iaa = 4
      ido = istb
      do  540 id = ista,istb,1
      ilane(iaa) = iline(ido)
       iaa = iaa -1
      ido = ido -1
  540 continue
      call kchint(ix,ilane)
      iswv = 0
      call ktabcp(numct,ix,iswv,ixx,ilane)
      if (iswv-1) 560,550,560
  550 call klinfx(iline,nch,ixx,ilane,ista,istb)
  560 iax = istb-ista+1
      iva = istb-(iax-ixx)
      go to  420
c
c.... This section is for characters "E" or "e"
c
  570 isum = 0
      id = icpos -1
c
c.... "Encode (c,n)list"    where "c" is the item being encoded.
c                                 "n" is the format number.
c     Check for "encode" followed by "(" or blank.
c
      do 600 ic = 1,6,1
      id = id + 1
      ilin = iline(id)
      iskp = iskip9(ic)
      if (ilin-iskp) 580,590,580
  580 iskp = iskipa(ic)
      if (ilin-iskp) 600,590,600
  590 isum = isum+1
  600 continue
      if (isum - 6) 1370,610,1370
  610 call krcont(nch,ictc,iline,20)
      call kendec(iline,isw4,numct,nch,icpos)
      if (isw4) 1370,1490,1370
c
c.... This section is for characters "I" or "i"
c
  620 isum = 0
      id = icpos -1
      do 650 ic = 1,2,1
      id = id + 1
      ilin = iline(id)
      iskp = iskipn(ic)
      if (ilin-iskp) 630,640,630
  630 iskp = iskipo(ic)
```

```
      if (ilin-iskp) 650,640,650
  640 isum = isum+1
  650 continue
      if (isum - 2) 1370,660,1370
  660 ixee = icpos + 2
      ixeee = icpos + 13
      do 680 id = ixee,ixeee,1
       ilin = iline(id)
      if (ilin - klpren) 670,690,670
  670 if (ilin - kblank) 1370,680,1370
  680 continue
  690 call krcont(nch,ictc,iline,20)
      isum = 0
c
c.... In an "if" statement, find text following outer set of
c     parentheses.  In order to do this the number and kind of
c     parentheses must be checked.
c          Following is a description of the process:  To find this
c     unique set of starting and stopping parentheses a variable
c     called "isum" is set to 0.  Then move from left to right
c     following the "if".  Add 1 to the value of "isum" for every
c     "(". and subtract 1 for every ")" encountered.  Processing will
c     stop when "isum" is found equal to 0 .  In doing this if a quote
c     is encountered the search for all parentheses is stopped. Until
c     another quote of the same type is found.  At that time the search
c     for the parentheses will resume.
c     "kqusw" is the quote switch for either double or single quotes.
c.... Double quote   = -1.
c     Single quote   = +1.
c
      kqusw = 0
      do 830 ia = id,nch,1
      ilin = iline(ia)
      if (kqusw) 770,700,790
  700 if (ilin - ksq) 710,810,710
  710 if (ilin - kdq) 720,820,720
  720 if (ilin - klpren) 740,730,740
  730 isum = isum + 1
      go to 760
  740 if (ilin - krpren) 830,750,830
  750 isum = isum -1
  760 if (isum) 830,840,830
  770 if (ilin - kdq) 830,780,830
  780 kqusw = 0
      go to 830
  790 if (ilin - ksq) 830,800,830
  800 kqusw = 0
      go to 830
  810 kqusw = 1
      go to 830
  820 kqusw =  -1
  830 continue
      go to 1480
  840 iptr = ia + 1
```

```
c
c.... The first non-blank position following the right parentheses
c     of the "if" statement is assigned to variable "iptr".
c
      do 850 ia = iptr,nch,1
      ilin = iline(ia)
      if (ilin - kblank) 860,850,860
c
c.... If position "ilin" is  equal to blank, check next card column.
c
  850 continue
      go to 1480
c
c.... Check the first non-blank position against array "ispeci".
c    If a match, then the "if" statement must be a logical type.
c
  860 do 870 id = 1,8,1
      ispc = ispeci(id)
      if (ispc - ilin) 870,880,870
  870 continue
      go to 890
  880 ic = id
c
c.... Here the value of "id" is set equal to one less the the
c     value of "ia".  This is so that the pointer is positioned
c     at the non-blank character following the parentheses.
c
      id = ia -1
      go to (990,990,1040,1040,930,930,1100,1100) ,ic
  890 numstat = 3
c
c.... This section is for processing arithmetic "if"
c     statements.
c
      do 900 iee = 1,10,1
c
c.... Check to make sure that the starting position is a
c     number and not a character .
c
      inx = inum(iee)
      if (ilin - inx) 900,910,900
  900 continue
      ia = ia -1
      go to  340
  910 ichn  = 0
      isw4  = 0
      iisw  = 0
      iwswl = 1
      itype = ia
      ia    = nch
      go to 1280
  920 isum = 0
```

```
c
c.... This section is for characters "W" or "w".
c
      isw4 = 0
      id = icpos -1
  930 do 960 ic = 1,5,1
      id = id + 1
      ilin = iline(id)
      iskp = iskipj(ic)
      if (ilin-iskp) 940,950,940
  940 iskp = iskipk(ic)
      if (ilin-iskp) 960,950,960
  950 isum = isum+1
  960 continue
      if (isum - 5) 1370,970,1370
  970 call krcont(nch,ictc,iline,20)
      call kproc(iline,isw4,numct,nch,id + 1)
      if (isw4) 1370,1490,1370
  980 isum = 0
      isun = 0
c
c.... This section is for characters "P" or "p".
c
      isw4 = 0
      id = icpos -1
  990 idd = id
      do 1000 ic = 1,5,1
      id = id + 1
      ilin = iline(id)
      if (ilin.eq.iskipl(ic) ) isum = isum + 1
      if (ilin.eq.iskipm(ic) ) isum = isum + 1
      if (ilin.eq.iskipu(ic) ) isun = isun + 1
      if (ilin.eq.iskipv(ic) ) isun = isun + 1
 1000 continue
      if (isun - 5) 1010,1020,1010
 1010 if (isum - 5) 1370,1020,1370
 1020 call krcont(nch,ictc,iline,20)
      call kproc(iline,isw4,numct,nch,id + 1)
      if (isw4) 1370,1490,1370
 1030 isum = 0
c
c.... This section is for characters "R" or "r".
c
      isw4 = 0
      id = icpos -1
 1040 do 1070 ic = 1,4,1
      id = id + 1
      ilin = iline(id)
      iskp = iskipp(ic)
      if (ilin-iskp) 1050,1060,1050
 1050 iskp = iskipq(ic)
      if (ilin-iskp) 1070,1060,1070
 1060 isum = isum+1
 1070 continue
```

```
      if (isum - 4) 1370,1080,1370
 1080 call krcont(nch,ictc,iline,20)
      call kproc(iline,isw4,numct,nch,id + 1)
      if (isw4) 1370,1490,1370
 1090 id = icpos -1
c
c.... This section is for "go to" statements.
c     Types of "go to" are:
c     .....  Unconditional.
c     .....  Computed.
c     .....  Assigned.
c.... Examples of different types :
c     "go to 10".
c     "go to (10,20,30,40),n".
c     "go to k , (10,20,30,40)".
c.... Special note:  Entry location for "if" statement on next line.
c     This location uses a different "id" value then the first non-blank
c     character in the line as is normally used.
c
 1100 isw4 = 0
      idd  = id
      isum = 0
      do 1130 ic = 1,2,1
c
c.... Check first two non-blank positions for the string "go".
c
      id = id + 1
      ilin = iline(id)
      iskp = iskipw(ic)
      if (ilin-iskp) 1110,1120,1110
 1110 iskp = iskipx(ic)
      if (ilin-iskp) 1130,1120,1130
 1120 isum = isum+1
 1130 continue
      if (isum - 2) 1360,1140,1360
 1140 idd = idd + 3
      inm = inumbr(11)
      do 1150 id = idd,idd + 20,1
      ilin = iline(id)
      if (ilin-inm) 1160,1150,1160
 1150 continue
c
c.... Check first non-blank following the "go" for a "t" or "T".
c
 1160 iskp = iskipx(3)
      if (ilin-iskp) 1170,1180,1170
 1170 iskp = iskipw(3)
      if (ilin-iskp) 1360,1180,1360
c
c.... Check second character for a "O" or "o".
c
 1180 ilin = iline(id+1)
      iskp = iskipw(4)
      if (ilin-iskp) 1190,1200,1190
```

```
 1190 iskp = iskipx(4)
      if (ilin-iskp) 1360,1200,1360
 1200 id = id + 2
c
c     If there are any continuation cards, they should be read now.
c
      call krcont(nch,ictc,iline,20)
      do 1210 ie = id,nch,1
      ilin = iline(ie)
      if (ilin-kequal) 1210,1360,1210
 1210 continue
      do 1220 ie = id,nch,1
c
c.... Start looking for a "(" which is used in an assigned or computed
c     "go to ".
c
      ilin = iline(ie)
      if (ilin-klpren) 1220,1230,1220
 1220 continue
      isw4 = 0
      call kproc(iline,isw4,numct,nch,id)
      if (isw4) 1360,1490,1360
 1230 do 1240 id = 1,nch,1
c
c.... If there was a "(" look for a ")" to stop scanning
c
      ia = nch + 1 -id
      ilin = iline(ia)
      if (ilin-krpren) 1240,1250,1240
 1240 continue
c
c.... Send error message to the user.  Notifying him that there
c     was a starting "(" but  there was no ending ")".
c
      write(iterm,9020)
 9020 format("ERROR: In 'go to' the parentheses do not balance",/)
      write(iterm,9030)(iline(iyy),iyy=1,nch)
 9030 format(1050a1)
      go to 1360
 1250 isum = 0
      do 1270 id = ie,ia,1
      ilin = iline(id)
      if (ilin-kcomma) 1270,1260,1270
 1260 isum = isum +1
 1270 continue
c
c.... In a assigned or computed "go to" statement, the number of
c     statement numbers to be changed is one greater than the
c     number of commas.
c
      numstat = isum + 1
      ichn = 0
```

```
c
c.... "ichn"  is number of statement numbers already checked !
c     "numstat"  is number of statement numbers to be checked.
c
      isw4 = 0
      iwswl = 1
      itype = ie + 1
 1280 idif = nch
      call kproc(iline,isw4,numct,nch,itype)
      if (isw4) 1330,1290,1330
c
c.... Increase the value of "ichn" by 1.  If the value of "ichn" is
c     equal to "numstat" then stop processing this card
c
 1290 ichn = ichn + 1
      ia = ia + nch -idif
      if (numstat - ichn) 1350,1350,1300
 1300 do 1310 id = itype,ia,1
      ilin = iline(id)
      if (ilin-kcomma) 1310,1320,1310
 1310 continue
 1320 itype = id + 1
      isw4 = 0
      go to 1280
 1330 if (ichn) 1340,1340,1350
 1340 isw4 = 0
      iwswl = 0
      go to 1360
 1350 isw4 = 0
      iwswl = 0
      call kproio(iunit,iline,nch)
      go to 1490
 1360 if (iisw) 1370,1370,1480
c
c.... This is the start of the miscellaneous processing section.  This
c     area is used for when there was no match in the first non-blank
c     character following or in card column number 7.
c.... The only item checked here is the Honeywell Multics "open"
c     statement.
c     This area of code can be changed for any additional types
c     of statements.
c
 1370 ilin = iline(icpos)
      if (ilin-kho) 1380,1390,1380
 1380 if (ilin-klo)  330,1390, 330
 1390 ilin = iline(icpos + 1)
      if (ilin-khp) 1400,1410,1400
 1400 if (ilin-klp) 1480,1410,1480
 1410 ilin = iline(icpos + 2)
      if (ilin-khe) 1420,1430,1420
 1420 if (ilin-kle) 1480,1430,1480
 1430 ilin = iline(icpos + 3)
      if (ilin-khn) 1440,1450,1440
 1440 if (ilin-kln) 1480,1450,1480
```

```
1450 ilin = iline(icpos + 4)
     if (ilin-klpren) 1460,1470,1460
1460 if(ilin-kblank) 1480,1470,1480
1470 isw4 = 0
     call krcont(nch,ictc,iline,20)
     call kproc(iline,isw4,numct,nch,11)
     if (isw4) 1480,1490,1480
1480 call kproio(iunit,iline,nch)
1490 continue
1500 continue
     call kclose(20)
     return
     end
```

```
      subroutine kproc(iline,iswitch,numct,nch,itype)
c
c     ****************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c     ****************************************************************
c
c.... Purpose:  To process the following FORTRAN IV statements.
c     Print.
c     Punch.
c     Read.
c     Write.
c     In the following forms:
c     For......... Read Print Punch.
c     .....n,items.
c     .....n.
c     ......,items.
c     For......... Read Write:
c     .....(unit,err=n) ,item.
c     .....(unit,nl) ,item.
c     .....(unit,nl,err=n) ,item.
c     .....(unit,nl).
c     .....(unit,nl,err=n,end=n2).
c     For......... Read.
c.... Plus special processing for "go to" and "if".
c.... Variables and their meaning:
c     iswitch = switch for error in comming here
c     numct   = possible number of statement numbers to be fixed.
c     nch     = number of characters in a string
c     itype   = starting position for searching.
c     ic      = First location of a comma after the starting left
c                parentheses.
c     ia      = The first right parentheses found.
c
      dimension iline(1050),ilane(4),iepos(15)
      common/rt/iskip1(10), iskip2(10), iskip3( 5), iskip4( 5),
     &          iskip5( 6), iskip6( 6), iskip7( 7), iskip8( 7),
     &          iskip9( 6), iskipa( 6), iskipb( 5), iskipc( 5),
     &          iskipd( 9), iskipe( 9), ichars( 4), iskipf( 4),
     &          iskipg( 4), ipctbc(18), iskiph( 2), iskipi( 2),
     &          iskipj( 5), iskipk( 5), iskipl( 5), iskipm( 5),
     &          iskipn( 2), iskipo( 2), iskipp( 4), iskipq( 4),
     &          inumbr(11), iskips( 6), iskipt( 6), iskipu( 5),
     &          iskipv(05), iskipw( 4), iskipx( 4), ispeci( 9),
     &             ien(04),   iend(04),   inum(10)
      common /kchs/khy   ,kly   ,khc   ,klc   ,khe   ,kle   ,
     &             khn   ,kln   ,khd   ,kld   ,kho   ,klo   ,
     &             khp   ,klp   ,kha   ,kla   ,khl   ,kll   ,
     &             kblank,kandpr,kcomma,kdolla,kequal,kastr ,
     &             krpren,klpren
      iunit = 15
      levea = nch
```

```
      m      = 0
      iou    = itype
      iterm = 0
   10 ilin=iline(iou)
      if (ilin-kblank) 20,40,20
   20 if (ilin-klpren) 380,40,380
   30 iswitch = 1
      return
   40 do 50 id = iou,levea,1
c
c.... Check for a "(" in card column number "iou" if not found
c     then there is no unit number in the code being checked.
c     Go to special processing area.
c
      ilin = iline(id)
      if (ilin-klpren) 50,60,50
   50 continue
      go to 380
   60 isw5 = 0
      isw6 = 0
      do 70 ic = iou,levea,1
c
c.... Look for the first leading "," following the "(" If none then
c     unit number,  but no format statement number.
c
      ilin=iline(ic)
      if (ilin-kcomma) 70, 80,70
   70 continue
      go to 380
   80 do  90 ia = iou,levea,1
      ilin=iline(ia)
      if (ilin-krpren)  90,100, 90
   90 continue
      go to 30
  100 continue
      ipl = id -ia
      if (ipl) 110,110,380
  110 continue
      ipl = ic -id
      if (ipl) 380,120,120
  120 ine = 0
  130 do 160 ib = ic,ia,1
      ilin = iline(ib)
      if (ilin-khe) 140,150,140
  140 if (ilin-kle) 160,150,160
  150 ine = ine + 1
      iepos(ine) = ib
  160 continue
      if (ine) 170,360,170
  170 m = 0
  180 ie = 0
      isum = 0
      isun = 0
      m = m + 1
```

```
      ib = iepos(m)
      do 190 id = ib,ib + 2,1
      ie = ie + 1
      ilin = iline(id)
      if (ilin.eq.iskipf(ie) ) isum = isum + 1
      if (ilin.eq.iskipg(ie) )isum = isum + 1
      if (ilin.eq.ien(ie) ) isun = isun + 1
      if (ilin.eq.iend(ie) ) isun = isun + 1
  190 continue
      if (isun - 3) 200,210,200
  200 if (isum - 3) 330,210,330
  210 do 230 ie = id,ia,1
      ilin = iline(ie)
      do 220  if = 1,10,1
      inm = inum(if)
      if (ilin-inm) 220,240,220
  220 continue
  230 continue
      go to 350
  240 istart = ie
c
c....Find the last digit in the statement number.  Assign the card
c    column number to the variable "istop".
c
      do 260 ie = istart,istart + 4,1
      ilin = iline(ie)
      do 250 if = 1,10,1
      inm = inum(if)
      if (ilin-inm) 250,260,250
  250 continue
      istop = ie -1
      go to 270
  260 continue
      go to 30
  270 do 280 ie = 1,4,1
c
c.... Initialize buffer "ilane" equal to blanks.
c
      ilane(ie) = kblank
  280 continue
      id = istop
      ie = 4
c
c.... Move the buffer "ilane" with the characters from the buffer
c     "iline" between positions istart and istop.
c
      do 290 if = istart,istop,1
      ilane(ie) = iline(id)
      id = id -1
      ie = ie -1
  290 continue
      iswv = 0
      call kchint(ix,ilane)
      call ktabcp(numct,ix,iswv,ixx,ilane)
```

```
      if (iswv-1) 310,300,310
  300 call klinfx(iline,nch,ixx,ilane,istart,istop)
c
c.... FORTRAN IV "err" and "end" branches for "read" and "write" are
c     processed in the following section .
c
  310 iix = istop + 1 -istart
  320 iepos(2) = iepos(2) -(ixx-iix)
  330 if (ine - m) 340,340,180
  340 isw5 = 1
      if (isw6 - 1) 360,350,360
  350 iswitch = 0
      call kproio(iunit,iline,nch)
      iepos(1) = 0
      iepos(2) = 0
      m = 0
      return
  360 isw6 = 1
      id = ic
      if (m) 370,210,370
  370 ia = iepos(1)
      go to 210
c
c     $$$$----$$$$----$$$$----$$$$----$$$$----$$$$----$$$$----$$$$
c
c.... ////////        Special Processin Section        ////////
c
c     $$$$----$$$$----$$$$----$$$$----$$$$----$$$$----$$$$----$$$$
c
c.... This section is used only when a format number reference is to
c     be changed but no unit number is used.
c
  380 iswitch = 0
c
c.... Check card column number "iou" to see if it is a blank.
c     If not, check to see if it is a number.
c
  390 ia = iou
  400 ilin = iline(ia)
      if (ilin-kblank) 420,410,420
  410 ia = ia + 1
c
c.... Continue checking until first non-blank position is found.
c     Here variable "ia" is the card column number where the non-blank
c     is located.
c
      go to 400
c
c.... Check to see if the alphanumeric character at position "ia"
c     is a number.
c
  420 ilin = iline(ia)
      do 430 ib = 1,10,1
      inm = inum(ib)
```

```
      if (ilin-inm) 430,450,430
  430 continue
c
c.... If the position was not a number, return to the calling
c     routine.
c
  440 iswitch = 1
      return
  450 istart = ia
      do 480 ib = 1,4,1
      ic = ia + ib
      if (nch-ic) 490,460,460
  460 ilin = iline(ic)
      do 470 id = 1,10,1
      inm = inum(id)
      if (ilin-inm) 470,480,470
  470 continue
      go to 490
  480 continue
c
c.... Send message to the user that statement number has too many
c     digits to be processed.
c
      write(iterm, 9000)
 9000 format ("ERROR.  Statement number has too many digits")
      write(0,9010)ia,ib
 9010 format(2i8)
      write(iterm, 9020)(iline(iyer),iyer=1,nch)
 9020 format(1050a1)
      go to 440
  490 istop = ic -1
c
c.... Variable "istop" can be the same value as "istart".
c     Initalize buffer "ilane" to blanks.
c
      do 500 ia = 1,4,1
      ilane(ia) = kblank
  500 continue
c     now load
      id = istop
      ia = 4
      do 510 ic = istart,istop,1
      ilane(ia) = iline(id)
      id = id -1
      ia = ia -1
  510 continue
      iswv = 0
      call kchint(ix,ilane)
      call ktabcp(numct,ix,iswv,ixx,ilane)
      if (iswv-1) 530,520,530
  520 call klinfx(iline,nch,ixx,ilane,istart,istop)
  530 call kproio(iunit,iline,nch)
      iix = istop-istart+1
      itype = istop-(iix-ixx)
```

```
iswitch = 0
return
end
```

```
      subroutine kcheck(ifil1,ifil2,iswq)
c
c     ***************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c     ***************************************************************
c
c       Purpose:  To see if "infile" and "outfile" are the same.
c
      dimension ifil1(8),ifil2(8)
      do 10 ia=1,8,1
      ilx = ifil1(ia)
      ily = ifil2(ia)
      if (ilx-ily) 20,10,20
   10 continue
      iswq = 0
      return
   20 iswq = 1
      return
      end
```

```
      subroutine kwrite (iunit,ixw,iline)
c
c     ****************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c     ****************************************************************
c
c.... Purpose:  To write final output using dynamic dimensioning,
c      for speed.
c
       dimension iline(ixw)
       write (iunit,9000)iline
 9000  format (72a1)
       return
       end
```

```
      subroutine krcont(nch,ictc,iline,inunit)
c
c     ****************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c     ****************************************************************
c
c.... Purpose:  To pick up continuation cards if any exist and then
c     move them into current buffer.
c.... Variables;
c     inunit  = unit number to be read.
c     ictc    = continuation switch.
c     nch     = number of characters in buffer.
c
      dimension iline(1050),ili(72)
      numbrch = nch
      mct = ictc
   10 if (mct) 40,40,20
   20 read (inunit,9000) icpos,mum,mct,ili
      ijki = 50
 9000 format (3i2,72a1)
      mix = numbrch
      do 30 ix1 = 7,mum,1
      mix = mix + 1
      iline(mix) = ili(ix1)
   30 continue
      numbrch = mix
      go to 10
   40 nch = numbrch
      ictc = mct
      return
      end
```

```
      subroutine kproio(iunit,iline,nch)
c
c     ****************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c     ****************************************************************
c
c.... Purpose:  Outputs the buffer to the file number "iunit".
c     The routine checks to see if the buffer contains less then
c     72 characters.  If the number of the characters is less than
c     72 the buffer is output.  If the buffer contains more than 72
c     characters the output is in sections.  The first section is
c     composed of from the first character to the nearest blank or
c     comma on or before the character position 72.  This is then
c     output in columns 1 through the dividing point.  The next
c     sections have a maximum of 66 characters, and start in the
c     position following the dividing point.  The next dividing point
c     is in another 65 positions or the end of the buffer,
c     which ever is less.  The division is made if needed at the
c     nearest blank or comma at or before the last postion being
c     considered.  This is output in columns 7 through the
c     dividing point,  columns 1 to 5 are blank and column 6 is the
c     continueation symbol.  This is continued  until the buffer has
c     been completely output.
c
      dimension iline(1050),ilix(72)
      common ioldy(1004),inewy(1004),iswsl
      common /kchs/khy   ,kly   ,khc   ,klc   ,khe   ,kle   ,
     &             khn   ,kln   ,khd   ,kld   ,kho   ,klo   ,
     &             khp   ,klp   ,kha   ,kla   ,khl   ,kll   ,
     &             kblank,kandpr,kcomma,kdolla,kequal,kastr ,
     &             krpren,klpren
      if (iswsl) 10,10,150
   10 ixi = nch -72
      if (ixi) 20,20,30
c
c.... Write results to output file.
c
   20 call kwrite (iunit,nch,iline)
      nch    = 0
      return
   30 istart = 1
      ifstl  = 0
      ival   = 71
   40 istop  = istart + ival
      if (istop-nch) 60,60,50
   50 istopl = nch
      go to 100
   60 isum = 0
      do 80 ia = istart,istop,1
      isum = isum + 1
      ib = istop + 1 - isum
```

```
      ilin = iline(ib)
      if (ilin - kblank) 70,90,70
   70 if (ilin - kcomma) 80,90,80
   80 continue
   90 istop1 = ib
  100 if (ifstl) 110,110,120
  110 call kwrite (iunit,istop1,iline)
      ifstl  = 1
      istart = istop1 + 1
      ival   = 65
      go to 40
  120 do 130 ia = 1,5,1
      ilix(ia) = kblank
  130 continue
c
c.... kandpr is a "&", if for any reason another continuation card
c     symbol is needed this can be changed.
c
      ilix(6) = kandpr
      icc = 6
      do 140 ia = istart,istop1,1
      icc = icc + 1
      ilix(icc) = iline(ia)
  140 continue
      call kwrite (iunit,icc,ilix)
      ival = 65
      istart = istop1 + 1
      if (nch - istop1) 150,150,40
  150 return
      end
```

```
      subroutine ktabcp(numct,ix,iswv,ixx,ilane)
c
c     ****************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c     ****************************************************************
c
c.... Purpose:   To compare the statement number "ix" against table
c     of statement numbers which are to be changed.
c
      dimension ilane(4)
      common ioldy(1004), inewy(1004),iswl
      data ibk/1h /
      if (numct) 30,30,10
   10 do 20 ia = 1,numct,1
      iaex = ix -ioldy(ia)
      if (iaex) 20,40,20
   20 continue
   30 iswv = 0
      go to 60
c
c.... Statement number "ix" to be changed.
c
   40 irunn = inewy(ia)
      do 50 ia = 1,4,1
      ilane(ia) = ibk
   50 continue
      call kintch(irunn,ilane)
c
c.... "ilane" buffer contains the new statement number which is to
c      replace the old statement number in the input data
c
c
c.... Count the number of characters in the buffer "ilane" and set
c     variable "ixx" equal to the results.
c
      iswv = 1
   60 ixx=0
      do 70 ia = 1,4,1
      if (ilane(ia) .ne.ibk) ixx = ixx + 1
   70 continue
      return
      end
```

```
      subroutine kintch(ix,ilane)
c
c     ****************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c     ****************************************************************
c
c.... Purpose:  Converts integer variable to characters.
c
      dimension  ilane(4)
      common/rt/iskip1(10), iskip2(10), iskip3( 5), iskip4( 5),
     &          iskip5( 6), iskip6( 6), iskip7( 7), iskip8( 7),
     &          iskip9( 6), iskipa( 6), iskipb( 5), iskipc( 5),
     &          iskipd( 9), iskipe( 9), ichars( 4), iskipf( 4),
     &          iskipg( 4), ipctbc(18), iskiph( 2), iskipi( 2),
     &          iskipj( 5), iskipk( 5), iskipl( 5), iskipm( 5),
     &          iskipn( 2), iskipo( 2), iskipp( 4), iskipq( 4),
     &          inumbr(11), iskips( 6), iskipt( 6), iskipu( 5),
     &          iskipv(05), iskipw( 4), iskipx( 4), ispeci( 9),
     &             ien(04),   iend(04),   inum(10)
      ic = 4
   10 iy = ix/10
      ipos = ix-10*iy + 1
      ilane(ic) = inum(ipos)
      if (iy) 20,30,20
   20 ic = ic -1
      ix = iy
      go to 10
   30 return
      end
```

```
      subroutine kchint(ix,ilane)
c
c     ****************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c     ****************************************************************
c
c.... Purpose:  Converts characters to an integer variable.
c
      dimension  ilane(4)
      data ibk/lh /
      common/rt/iskip1(10), iskip2(10), iskip3( 5), iskip4( 5),
     &          iskip5( 6), iskip6( 6), iskip7( 7), iskip8( 7),
     &          iskip9( 6), iskipa( 6), iskipb( 5), iskipc( 5),
     &          iskipd( 9), iskipe( 9), ichars( 4), iskipf( 4),
     &          iskipg( 4), ipctbc(18), iskiph( 2), iskipi( 2),
     &          iskipj( 5), iskipk( 5), iskipl( 5), iskipm( 5),
     &          iskipn( 2), iskipo( 2), iskipp( 4), iskipq( 4),
     &          inumbr(11), iskips( 6), iskipt( 6), iskipu( 5),
     &          iskipv(05), iskipw( 4), iskipx( 4), ispeci( 9),
     &             ien(04),   iend(04),   inum(10)
      num = 0
      do 20 ia = 1,4,1
      ila = ilane(ia)
      if (ila.eq.ibk) go to 20
      do 10 ib = 1,10,1
      inu = inum(ib)
      if (ila.ne.inu) go to 10
      num = (num*10) + ib -1
   10 continue
   20 continue
      ix = num
      return
      end
```

```
      subroutine klinfx(iline,nchh,ixx,ilane,ista,istb)
c
c     ****************************************************************
c                           Department of Interior
c                           U.S. Geological Survey
c                             George H. Harrach
c                             Geologic Division
c     ****************************************************************
c
c.... Purpose:  To move buffer "ilane" into main buffer "iline".
c.... Variables:
c      nchh   = number of characters in buffer "iline".
c      ixx    = length of buffer ilane.
c      ista   = starting position of the statement number being changed.
c      istb   = ending position of statement number being changed.
c
       dimension iline(1050) , ilane(4)
       data ibk/1h /
       nch = nchh
       ia = (istb + 1) -ista
       if (ia-ixx) 80,10,30
c
c.... Old and new statement numbers have the same number of digits.
c
    10 ib = ista-1
       ih = 4-ixx
       do 20 id=1,ixx,1
       ib=ib+1
       ih=ih+1
       iline(ib)=ilane(ih)
    20 continue
       return
c
c.... New statement number has fewer digits then the old number.
c
    30 ib=ista-1
       ih=4-ixx
       do 40 id=1,ixx,1
       ib=ib+1
       ih=ih+1
       iline(ib)=ilane(ih)
    40 continue
       if (istb-nch) 50,70,70
    50 do 60 id=istb+1,nch,1
       ib=ib+1
       iline(ib)=iline(id)
    60 continue
       nchh=nchh-(nch-ib)
       return
    70 nchh=ib
       istb=ib
       return
c
c.... A new statement number with more digits then the old statement
```

```
c      number has been encountered.  Expand the buffer so that new
c      statement number will fit.
c
   80 ie = nch
   90 iff = ie
c
c.... Expand buffer by "ixx-ia" positions
c
       do 100 id = 1,(ie - istb) ,1
       ifxu = iff + ixx -ia
       iline(ifxu) = iline(iff)
       iff = iff -1
  100 continue
       nct = nct + (ixx - ia)
  110 continue
       ib = istb + 1
       ic = 4
       ic = 4
       ib = istb + ixx-ia
       do 120 id = 1,ixx,1
       iline(ib) = ilane(ic)
       ic = ic -1
       ib = ib -1
  120 continue
c
c.... Increase "nch" by the amount that buffer was expanded.
c
       nch = ixx + ista-istb -1 + nch
       nchh = nch
       return
       end
```

```
      subroutine kendec(iline,iswitch,numct,nch,icp)
c
c     ****************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c     ****************************************************************
c
c.... Purpose:  To process "encode" & "decode" statements.
c     Where form is "encode(c,n) list"  and  "decode(c,n) list.
c     Where "n" is statement number to be changed.
c
      dimension iline(1050) ,ilane(4)
      data ibk/1h /
      data klpren,krpren,kcomma,kblank/1h(,1h),1h,,1h /
      icpa = icp+6
      nchh = nch
      ilin = iline(icpa)
      if (ilin-klpren) 10,30,10
   10 if (ilin-kblank) 20,30,20
   20 iswitch = 1
      return
   30 do 40 ic = icpa,nchh,1
      ilin = iline(ic)
      if (ilin-kcomma) 40,50,40
   40 continue
      go to 20
   50 istart = ic + 1
      do 60 ic = istart,nchh,1
      ilin = iline(ic)
      if (ilin-krpren) 60,70,60
   60 continue
      go to 20
c
c.... Initialize buffer "ilane" to blanks.
c
   70 istop = ic -1
      do 80 ic = 1,4,1
      ilane(ic) = ibk
   80 continue
      id = istop
      iaa = 4
c
c.... Move contents of buffer "iline" into "ilane" (where the statement
c     number is located).
c
      do 90 ic = istart,istop,1
      ilane(iaa) = iline(id)
      id = id -1
      iaa = iaa -1
   90 continue
      iswv = 0
      call kchint(ix,ilane)
```

```
      call ktabcp(numct,ix,iswv,ixx,ilane)
      if (iswv-1) 110,100,110
100   call klinfx(iline,nch,ixx,ilane,istart,istop)
110   call kproio(15,iline,nch)
      iswitch = 0
      return
      end
      subroutine kopen(iunit,ierr,name,inom)
```

```
c
c     ***************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c     ***************************************************************
c
c.... Machine dependent subroutine
c     Purpose:  To open files for the program and subroutines.
c
      character*32 name
c
c.... "inom" is a switch to determine opening mode.
c     "inom" -1       output.
c     "inom"  0       return.
c     "inom" +1       input.
c
      ierr = 0
      if (inom) 20,40,30
   10 ierr=1
      return
   20 open(iunit,file=name,form="formatted",mode="out",err=10)
      go to 40
   30 open(iunit,file=name,form="formatted",mode="in" ,err=10)
   40 ierr=0
      return
      end
```

```
      subroutine kremov(name)
c
c     ***************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c    ***************************************************************
c
c.... Machine dependent subroutine.
c     Purpose:  To delete scratch file.
c
      character*32 name
      external delete(descriptors)
      call delete(name)
      return
      end



      subroutine kclose(iv)
c
c     ***************************************************************
c                          Department of Interior
c                          U.S. Geological Survey
c                            George H. Harrach
c                            Geologic Division
c    ***************************************************************
c
c.... Machine dependent subroutine
c     Purpose:  To close files that are used by the programs.
      external close(descriptors)
      close(iv)
      return
      end
```

```
 block data
 common/rt/iskip1(10), iskip2(10), iskip3( 5), iskip4( 5),
&           iskip5( 6), iskip6( 6), iskip7( 7), iskip8( 7),
&           iskip9( 6), iskipa( 6), iskipb( 5), iskipc( 5),
&           iskipd( 9), iskipe( 9), ichars( 4), iskipf( 4),
&           iskipg( 4), ipctbc(18), iskiph( 2), iskipi( 2),
&           iskipj( 5), iskipk( 5), iskipl( 5), iskipm( 5),
&           iskipn( 2), iskipo( 2), iskipp( 4), iskipq( 4),
&           inumbr(11), iskips( 6), iskipt( 6), iskipu( 5),
&           iskipv(05), iskipw( 4), iskipx( 4), ispeci( 9),
&              ien(04),   iend(04),   inum(10)
 data iskip1/1hD,1hI,1hM,1hE,1hN,1hS,1hI,1hO,1hN,1h /
 data iskip2/1hd,1hi,1hm,1he,1hn,1hs,1hi,1ho,1hn,1h /
 data iskip3/1hD,1hA,1hT,1hA,1h /
 data iskip4/1hd,1ha,1ht,1ha,1h /
 data iskip5/1hF,1hO,1hR,1hM,1hA,1hT/
 data iskip6/1hf,1ho,1hr,1hm,1ha,1ht/
 data iskip7/1hC,1hO,1hM,1hM,1hO,1hN,1h /
 data iskip8/1hc,1ho,1hm,1hm,1ho,1hn,1h /
 data iskip9/1hE,1hN,1hC,1hO,1hD,1hE/
 data iskipa/1he,1hn,1hc,1ho,1hd,1he/
 data iskipb/1hC,1hA,1hL,1hL,1h /
 data iskipc/1hc,1ha,1hl,1hl,1h /
 data iskipd/1hC,1hH,1hA,1hR,1hA,1hC,1hT,1hE,1hR/
 data iskipe/1hc,1hh,1ha,1hr,1ha,1hc,1ht,1he,1hr/
 data iskipf/1hE,1hR,1hR,1h=/
 data iskipg/1he,1hr,1hr,1h=/
 data ichars/1h(,1h),1h ,1h,/
 data ipctbc/1hD,1hd,1hC,1hc,1hF,1hf,1hE,1he,1hI,1hi,
&1hW,1hw,1hP,1hp,1hR,1hr,1hG,1hg/
 data iskiph/1hD,1hO/
 data iskipi/1hd,1ho/
 data iskipj/1hW,1hR,1hI,1hT,1hE/
 data iskipk/1hw,1hr,1hi,1ht,1he/
 data iskipl/1hP,1hR,1hI,1hN,1hT/
 data iskipm/1hp,1hr,1hi,1hn,1ht/
 data iskipn/1hI,1hF/
 data iskipo/1hi,1hf/
 data iskipp/1hR,1hE,1hA,1hD/
 data iskipq/1hr,1he,1ha,1hd/
 data inumbr/1h0,1h1,1h2,1h3,1h4,1h5,1h6,1h7,1h8,1h9,1h /
 data iskips/1hd,1he,1hc,1ho,1hd,1he/
 data iskipt/1hD,1hE,1hC,1hO,1hD,1hE/
 data iskipu/1hP,1hU,1hN,1hC,1hH/
 data iskipv/1hp,1hu,1hn,1hc,1hh/
 data iskipw/1hG,1hO,1hT,1hO/
 data iskipx/1hg,1ho,1ht,1ho/
 data ispeci/1hP,1hp,1hR,1hr,1hW,1hw,1hG,1hg,1h /
 data ien /1hE,1hN,1hD,1h=/
 data iend/1he,1hn,1hd,1h=/
 data inum/1h0,1h1,1h2,1h3,1h4,1h5,1h6,1h7,1h8,1h9/
```

```
c
c.... Start second common block
c
      common /kchs/khy   ,kly   ,khc   ,klc   ,khe   ,kle   ,
     &             khn   ,kln   ,khd   ,kld   ,kho   ,klo   ,
     &             khp   ,klp   ,kha   ,kla   ,khl   ,kll   ,
     &             kblank,kandpr,kcomma,kdolla,kequal,kastr ,
     &             krpren,klpren
      data khy,kly,khc,klc,khe,kle,khn/lhY,lhy,lhC,lhc,lhE,lhe,lhN/
      data kln,khd,kld,kho,klo,khp,klp/lhn,lhD,lhd,lhO,lho,lhP,lhp/
      data kha,kla,khl,kll,kblank,kandpr/lhA,lha,lhL,lhl,lh ,lh&/
      data kcomma,kdolla,kequal,kaser,krpren/",",lh$,lh=,lh*,lh)/
      data klpren/lh(/
      common /ispx/cname(8),dname(8)
      data cname/4hTMP ,4h    ,4h    ,4h    ,4h    ,4h    ,
     &4h    ,4h    /
      data dname/4hTMA ,4h    ,4h    ,4h    ,4h    ,4h    ,
     &4h    ,4h    /
      end
```