Subject: Re: Free source code diagramming programs
Posted by idlwizard-1@yahoo.com on Mon, 10 Apr 2006 19:52:47 GMT
View Forum Message <> Reply to Message

For people who have trouble reading that web page, see the folowing message I just posted this message to alt.sources:

My source code diagramming programs

Last revised 4/9/2006

This post to alt.sources is for anyone who has trouble reading my website

http://www.geocities.com/grunes/diagram.html

INTRODUCTION

These programs diagram source code in the following languages: C and C++
FORTRAN
HTML (very incomplete)
IDL, PV-WAVE, GDL and FL

They do things like draw lines showing the start and end of routines and blocks, put * next to jumps, and = next to commented out sections, and can warn you of certain classes of error. They can help you find problems in your own code, or help you look at other people's long complicated legacy code. For example:

```
+---------- subroutine a(x)       | 1
|+---------- do i=1,5              | 2
||+---------- if(i/2*2.eq.i)then  | 3
||| x=x*i                         | 4
||+---------- else                | 5
||| x=x/i                         | 6
||+---------- endif               | 7
|+---------- enddo                | 8
+---------- end                   | 9
```

The VAX and MS-DOS procedures have not recently been tested. If you like or dislike these programs, send e-mail to username grunes at domain yahoo.com. Bug reports must include sample code on which it failed.

The programs themselves are in FORTRAN. I know that is a problem for users of other programming languages,

but FORTRAN is freely available as g77 or g95 under
Cygwin (under Windows) or Linux, and is available as
f77, f90 or f95 on many other platforms. Compilation
is simple, e.g.

```
g77 diagramf.f -o diagramf
```

The files are at http://www.geocities.com/grunes/diagram.html,
and are also included below. If you request it, I will email you a
diagram.tar.gz archive containing everything.

Included files:

diagramc: Diagrams C, C++
  diagramc.f   Fortran language source code
  Procedures to run diagramc without answering questions:
    diagramc.sh   Unix csh procedure
    diagramc.bat MS-DOS procedure
    diagramc.vax VAX VMS DCL procedure

diagramf: Diagrams FORTRAN
  diagramf.f   Fortran language source code
  Procedures to run diagramf without answering questions on card format
code:
    diagramf.sh   Unix csh procedure
    diagramf.bat MS-DOS procedure
    diagramf.vax VAX VMS DCL procedure
  Procedures to run diagramf without answering questions on free format
code:
    diagram9.sh   Unix csh procedure
    diagram9.bat MS-DOS procedure
    diagram9.vax VAX VMS DCL procedure

diagramh: Diagrams HTML (Very Incomplete)
  diagramh.f   Fortran language source code
  Procedures to run diagramh without answering questions:
    diagramh.sh   Unix csh procedure
    diagramh.bat MS-DOS procedure
    diagramh.vax VAX VMS DCL procedure

diagrami: Diagrams IDL, PV-WAVE, GDL, FL
  diagrami.f   Fortran language source code
  Procedures to run diagrami without answering questions:
    diagrami.sh   Unix csh procedure
    diagrami.bat MS-DOS procedure
    diagrami.vax VAX VMS DCL procedure

undiagram: Try to derive source code from diagram

undiagram.f Fortran language source code

My Home Page: http://www.geocities.com/grunes

----------BEGIN diagramc.f----------
c EXAMPLE OF OUTPUT (looks better if you choose IBM PC line graphics):

```
c    +------ I_Hate_C() {                          |   1
c    |+------- if (You_Like(C)) {                   |   2
c    ||          BoyOrGirl=Bad;                    |   3
c  +-||       #ifdef SMART                         |   4
c  | ||          ReEducate();                      |   5
c  +-||       #endif                              |   6
c    |+------- } else {                            |   7
c    ||          BoyOrGirl=Good;                       |   8
c    |+------- }                                 |   9
c    +------ }                                   |  10
```

c Diagrams C language {} constructs, case and default,
c and puts a * next to goto, break, continue, exit and return. It can
c place = next to comment blocks.
c Up to 2 levels of preprocessor constructs (#if--#elif--#endif) are
c diagrammed separately, on the outside.

c Designed by mitch grunes, in his own time.

c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
c Revision date: 8/25/96.
c If you find it useful, or find a problem, please send me e-mail.

c This program was written in FORTRAN, for historic reasons.
c (For this reason, people who mostly program in C will probably be
c unwilling to use this program, even as a utility.)

c WARNING: The "/*" sequences will confuse compilers like SGI Fortran
c that use a C pre-processor by default on Fortran programs, so you
c must use a compiler switch like "-nocpp" to turn that off.

c It can be confused if an INCLUDE block contains a structure that
c begins inside and ends outside (or vice-versa).

c It also does not diagram IF, FOR, ELSE, WHILE, etc., unless you use
c   { and } to enclose the conditionally executed statement--
c   e.g. it will not draw any lines next to

```
c       if(condition)
c         for (i=0; i<10; i++)
c           a[i]=2;
```

```
c       else
c         b=3;

c I hope this works for you, but bear in mind that nothing short of
c  a full-fledged language parser could really do the job.  Perhaps
c  worth about what you paid for it.    (-:

c Versions: To diagram Fortran:     diagramf.f
c                   IDL/PV-WAVE: diagrami.f
c                   C:           diagramc.f
c MS-DOS procedures to call above programs without asking so many
questions,
c  append output to file diagram.out:
c                   Fortran:     diagramf.bat (card format)
c                                diagram9.bat (free format)
c                   IDL/PV-WAVE: diagrami.bat
c                   C:           diagramc.bat
c Similar Unix csh procedures:
c                   Fortran:     diagramf.sh  (card format)
c                                diagram9.sh  (free format)
c                   IDL/PV-WAVE: diagrami.sh
c                   C:           diagramc.sh
c Similar Vax VMS DCL procedures:
c                   Fortran:     diagramf.vax (card format)
c                                diagram9.vax (free format)
c                   IDL/PV-WAVE: diagrami.vax
c                   C:           diagramc.vax

        program diagramc                        ! Diagrammer
for C
        character*80 filnam,filnam2

        print*,'C source filename?'
        read(*,'(a80)')filnam
        print*,filnam

        print*,'Output file (blank=screen)?'
        read(*,'(a80)')filnam2
        print*,filnam2

        print*,'Column in which to write line #''s ',
     &   '(67 for 80 col screen, 0 for none):'
        LCol=0
        read*,LCol
        print*,LCol

        print*,'Notate comments with = (0=no, 1=yes; 1?):'
        inotate=1
```

```
      read*,inotate
      print*,inotate

      print*,'Use IBM PC graphics characters (0=no):'
      iGraphics=0
      read*,iGraphics
      print*,iGraphics

      call diagram(filnam,filnam2,LCol,inotate,iGraphics)
      end
c------------------------------------------------------------ ------------
      subroutine diagram(filnam,filnam2,LCol,inotate,
     &   iGraphics)
c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
      character*80 filnam,filnam2
      character*160 a,b,bsave
      character*5 form
      character*8 fm
      character*1 c
      logical fout
      logical find
      external find
      common icol

c Symbols which will mark block actions:
      character*1 BlockBegin    (2) /'+','+'/  ! Start of block
      character*1 BlockEnd      (2) /'+','+'/  ! End of block
      character*1 BlockElse     (2) /'+','+'/  ! Else construct
      character*1 BlockContinue (2) /'|','|'/  ! Block continues w/o
change
      character*1 BlockHoriz    (2) /'-','-'/  ! Horizontal to start
of line
c Same, but allows horizontal line to continue through:
      character*1 BlockBeginH   (2) /'+','+'/  ! Start of block
      character*1 BlockEndH     (2) /'+','+'/  ! End of block
      character*1 BlockElseH    (2) /'+','+'/  ! Else construct

      if(iGraphics.ne.0)then
        iGraphics=1

        BlockBegin   (1)=char(218)          ! (1)=normal
        BlockEnd     (1)=char(192)
        BlockElse    (1)=char(195)
        BlockContinue(1)=char(179)
        BlockHoriz   (1)=char(196)
        BlockBeginH  (1)=char(194)
        BlockEndH    (1)=char(193)
        BlockElseH   (1)=char(197)
```

```
        BlockBegin   (2)=char(214)          ! (2)=DO/FOR loops
(doubled)
        BlockEnd     (2)=char(211)          ! (not yet used)
        BlockEnd     (2)=char(211)
        BlockElse    (2)=char(199)
        BlockContinue(2)=char(186)
        BlockHoriz   (2)=char(196)
        BlockBeginH  (2)=char(209)
        BlockEndH    (2)=char(208)
        BlockElseH   (2)=char(215)
      endif

      open(1,file=filnam,status='old')
      fout=filnam2.gt.' '
      if(fout)open(2,file=filnam2,status='unknown')
                                  ! ASCII 12 is a form
feed
      if(fout)write(2,*)char(12),
     &  '=============--',filnam(1:LenA(filnam)),'--=============='

      if(fout)    write(2,'(11x,a50,a49,/)')  ! Write column header
     &  '....,....1....,....2....,....3....,....4....,....5',
     &  '....,....6....,....7....,....8....,....9....,....'
      if(.not.fout)write(*,'(11x,a50,a49,/)')' ',
     &  '....,....1....,....2....,....3....,....4....,....5',
     &  '....,....6....,....7....,....8....,....9....,....'

      i3=0                          ! # nest levels after
                                  !  current line
      i3pp=0                        ! same for
pre-processor
      nline=0
      icomment=0                        ! not inside comment
      iunit=1
10    a=' '
      read(iunit,'(a160)',end=99)a
      nline=nline+1
      fm=' '
      write(fm,'(i5)')nline
      form=fm

      if(a(1:1).eq.char(12))then
        if(fout)write(2,'(a1,:)')char(12)
        if(.not.fout)print*,'------------FORM FEED------------'
        b=a(2:160)
        a=b
      endif
```

```
      b=' '                          ! Turn tabs to spaces
      j=1
      do i=1,LenA(a)
        if(a(i:i).eq.char(9))then
          j=(j-1)/8*8+8+1
     ! Make sure is good ASCII char
        elseif(j.le.160.and.a(i:i).ge.'
'.and.a(i:i).lt.char(128))then
          b(j:j)=a(i:i)
          j=j+1
        endif
      enddo

      a=b
      bsave=b
      b=' '
      i1=i3                           ! # nest levels before
                                     !  current line
      i1pp=i3pp                        ! same for
pre-processor
      i4=0                            ! not 0 to flag start
or end
                                     !  of block
      i4pp=0
      iquote=0                        ! no ' yet
      idquote=0                        ! no " yet
      icomment2=0                        ! anything outside
comment?
      icomment3=icomment                      ! no comment occurred?
      i=1
      j=1
      dowhile(i.le.160)                    ! handle upper case
        c=a(i:i)
        if(c.ge.'A'.and.c.le.'Z')c=char(ichar(c)+32)
        if(c.eq.''''.and.idquote.eq.0.and.icomment.eq.0)then
          iquote=1-iquote
          if(i.gt.1)then
                                     ! char(92) is \
            if(iquote.eq.0.and.a(i-1:i-1).eq.char(92))
     &        iquote=1-iquote
          endif
        endif
        if(c.eq.'"' .and.iquote .eq.0.and.icomment.eq.0)then
          idquote=1-idquote
          if(i.gt.1)then
            if(idquote.eq.0.and.a(i-1:i-1).eq.char(92))
     &        idquote=1-idquote
```

```
          endif
        endif
        if(c.eq.'/'.and.i.lt.160.and.iquote.eq.0.and.idquote.eq.0)
! / *  ?
     &    then
          if(a(i+1:i+1).eq.'/')icomment3=1    ! // is C++ comment
line
          if(a(i+1:i+1).eq.'/')go to 15
          if(a(i+1:i+1).eq.'*')then
            if(icomment.ne.0)then
              PRINT*,'***WARNING--nested comment line',form
              if(fout)print*,a
              print*,char(7)
            endif
            icomment=1
            icomment3=1
            c=' '
            i=i+1
          endif
        endif
        if(c.eq.'*'.and.i.lt.160.and.iquote.eq.0.and.idquote.eq.0)
! * /  ?
     &    then
          if(a(i+1:i+1).eq.'/')then
            if(icomment.eq.0)then
              PRINT*,'***WARNING--*/ without /* clause line',form
              if(fout)print*,a
              print*,char(7)
            endif
            icomment=0
            c=' '
            i=i+1
          endif
        endif
        if(icomment.ne.0)c=' '
        if(c.ne.' ')icomment2=1
        if(c.eq.'{')then
          if(fout.and.i3.eq.0)print*,'Line ',form,' ',a(1:LenA(a))
          i3=i3+1
        elseif(c.eq.'}')then
          i3=i3-1
          i4=max(i4,i1-i3)
          if(i3.lt.0)then
            PRINT*,'***ERROR--INVALID DIAGRAMMING INDEX line',
     &        form
            if(fout)
     &        WRITE(2,*)'***ERROR--INVALID DIAGRAMMING INDEX!***'
            if(fout)print*,a
```

```
            print*,char(7)
            i3=max(i3,0)
          endif
        endif
        if(j.le.160) b(j:j)=c
        if(j.gt.1)then                  ! (kill multiple
spaces)
          if(c.eq.' '.and.b(j-1:j-1).eq.' ')j=j-1
        endif
        j=j+1
        i=i+1
      enddo
      if(iQuote.ne.0.or.idquote.ne.0)then
        PRINT*,'***ERROR--UNCLOSED QUOTE AT LINE ',form
        if(fout)WRITE(2,*)'***ERROR--UNCLOSED QUOTE AT LINE ',form
        if(fout)print*,a
        print*,char(7)
      endif

15    if(find(b,'#if',2).or.find(b,'# if',2))then
        i3pp=i3pp+1
        i4pp=1
      elseif(find(b,'#else',2).or.find(b,'# else',2)
     &  .or.find(b,'#elif',2).or.find(b,'# elif',2))then
        i4pp=1
      elseif(find(b,'#endif',2).or.find(b,'# endif',2))then
        i3pp=i3pp-1
        i4pp=1
      endif

      igoto=0                           ! no goto on line
      if(find(a,'go to',64+512).or.find(a,'goto',64+512)
     &   .or.find(a,'return',32+512)
     &   .or.find(a,'break',32+512).or.find(a,'continue',32+512)
     &   .or.find(a,'exit',32+512))igoto=1

      if(find(b,'case',32+512).or.
     &   find(b,'default ',512).or.find(b,'default:',512))i4=max(1,i4)

20    b=bsave
      a=' '
      if(i1  .lt.0.or.i3  .lt.0.or.i4  .lt.0.or.
     &    i1pp.lt.0.or.i3pp.lt.0.or.i4pp.lt.0)then
        PRINT*,'***ERROR--INVALID DIAGRAMMING INDEX line',form
        if(fout)WRITE(2,*)'***ERROR--INVALID DIAGRAMMING INDEX!***'
        if(fout)print*,b
        print*,char(7)
        i1=max(i1,0)
```

```
        i3=max(i3,0)
        i4=max(i4,0)
        i1pp=max(i1pp,0)
        i3pp=max(i3pp,0)
        i4pp=max(i4pp,0)
      endif

      i2=max(i1,i3)                    ! # of nests on current
line
      i4=max(i4,iabs(i3-i1))             ! not 0, to flag start
or
                                  !  end of block
      i2pp=max(i1pp,i3pp)
      i4pp=max(i4pp,iabs(i3pp-i1pp))

      iBlock=1                         ! For the present
version.

      a=' '                           ! Leave space for
diagram
      a(12:160)=b                      !  (must match column
header)

      LastUse=1                        ! Last usable diagram
col
      dowhile(LastUse.lt.160.and.a(LastUse:LastUse).eq.' ')
        LastUse=LastUse+1
      enddo
      LastUse=LastUse-2

      if(igoto.ne.0)a(1:1)='*'           ! Place * next to jumps
      if(icomment2.eq.0.and.icomment3.ne.0..and.inotate.ne.0)
     &   a(1:1)='='

      if(i2pp.gt.0)then                ! Draw one vertical
line per
        do i=2,min(i2pp+1,3)             !  nest level.
          a(i:i)=BlockContinue(iBlock)
        enddo
      endif

      if(i4pp.ne.0)then                ! Draw horizontal lines
inward
        do i=i2pp+2,3                  !  from above.
          a(i:i)=BlockHoriz(iBlock)
        enddo
      endif
```

```
      do i=0,i4pp-1                    ! May need to replace
some
                                       !  vertical lines with
                                       !       else  symbol
        c=              BlockElse(iBlock) !  or   begin symbol
        if(i1pp+i.lt.i3pp)c=BlockBegin(iBlock)!  or   end   symbol
        if(i1pp+i.gt.i3pp)c=BlockEnd  (iBlock)
        j=max(2,min(3,i2pp+1-i))
        a(j:j)=c
        if(a(j+1:j+1).eq.BlockElse  (iBlock)) ! Continue horizontal
lines
     &     a(j+1:j+1)  = BlockElseH (iBlock)
        if(a(j+1:j+1).eq.BlockBegin (iBlock))
     &     a(j+1:j+1)  = BlockBeginH(iBlock)
        if(a(j+1:j+1).eq.BlockEnd   (iBlock))
     &     a(j+1:j+1)  = BlockEndH  (iBlock)
      enddo

      if(i2.gt.0)then                  ! Same for
non-pre-processor
        do i=4,min(i2+3,LastUse)
          a(i:i)=BlockContinue(iBlock)
        enddo
      endif

      if(i4.ne.0)then
        do i=i2+4,LastUse
          a(i:i)=BlockHoriz(iBlock)
        enddo
      endif

      do i=0,i4-1

        c=            BlockElse(iBlock)
        if(i1+i.lt.i3)c=BlockBegin(iBlock)
        if(i1+i.gt.i3)c=BlockEnd  (iBlock)
        j=max(4,min(LastUse,i2+2+1-i))
        a(j:j)=c
        if(a(j+1:j+1).eq.BlockElse  (iBlock))
     &     a(j+1:j+1)  = BlockElseH (iBlock)
        if(a(j+1:j+1).eq.BlockBegin (iBlock))
     &     a(j+1:j+1)  = BlockBeginH(iBlock)
        if(a(j+1:j+1).eq.BlockEnd   (iBlock))
     &     a(j+1:j+1)  = BlockEndH  (iBlock)
      enddo

      if(LCol.gt.0.and.a(max(1,LCol+11):160).eq.' ')then       ! line
#
```

```
          if(form(1:1).eq.' ')form(1:1)=BlockContinue(iBlock)
          a(LCol+11:160)=form
        endif

        n=LenA(a)                       ! Output diagrammed
line
        if(fout)     write(2,'(80a1,80a1)')(a(i:i),i=1,n)
        if(.not.fout)write(*,'(1x,80a1,80a1)')(a(i:i),i=1,n)

        i1=i3
        i1pp=i3pp
        goto 10
99      if(iunit.eq.3)then
          iunit=1
          i1=i1-1
          i1pp=i1pp-1
          close(3)
          goto 10
        endif
        if(i3.gt.0.or.i3pp.gt.0)then
          PRINT*,'***WARNING--SOME NEST LEVELS LEFT HANGING AT END***'
          print*,char(7)
        endif
        end
 c------------------------------------------------------------ ------------
        logical function find(a,b,icond)        ! find b in a, subject
to
                                        !  conditions:
                                        ! icond=sum of the
following:
                                        ! 2:  Must be first
non-blank
                                        ! 32: Next character
not alphanumeric
                                        ! 64: Next character
not alphabetic
                                        ! 512  Prior character,
if present,
                                        !      must be blank or
) or }
                                        !      or { or ;
c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
c Revision date: 8/25/96.
        character*(*) a,b
        character*1   c,cNext
        common icol
        logical result
```

```
        ii=len(a)
        jj=len(b)
        result=.false.
        do i=1,ii-jj+1
          if(a(i:i+jj-1).eq.b)then
            icol1=i                     ! icol1=column of item
found
            icol =i+jj                  ! icol =column after
item
                                      !  found
            c=' '
            cNext=' '
            if(icol1.gt.1)c=a(icol1-1:icol1-1)
            if(icol .le.ii)cNext=a(icol:icol)

            result=.true.

            if(result.and.iand(icond,2).ne.0.and.icol1.gt.1)then
              result=a(1:icol1-1).eq.' '
            endif

            if(result.and.iand(icond,32).ne.0)
     &        result=(cNext.lt.'0'.or.cNext.gt.'9').and.
     &               (cNext.lt.'a'.or.cNext.gt.'z')

            if(result.and.iand(icond,64).ne.0)
     &        result=(cNext.lt.'a'.or.cNext.gt.'z')

            if(result.and.iand(icond,512).ne.0)result=c.eq.' '
     &        .or.c.eq.';'.or.c.eq.')'.or.c.eq.'{'.or.c.eq.'}'

            find=result
            if(result)return
          endif
        enddo
        find=result
        return
        end
 c------------------------------------------------------------ ------------
        function LenA(a)                ! Length of string, at
least 1
c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
c Revision date: 8/25/96.
        character*(*) a
        n=len(a)
        dowhile(n.gt.1.and.a(n:n).eq.' ')
          n=n-1
        enddo
```

```
      LenA=n
      end
------------------END diagramc.f--------------------
-----------------BEGIN diagramc.sh-------------------
#!/bin/csh
#             ---diagramc.sh---
#Unix csh procedure to diagram a C language program.

#On some unix systems $1 should be replaced by %1.

# by Mitchell R Grunes.
# for his own use, in his own time

#I assume that the executable and this procedure are in the search
path,
#  and that this procedure has execute permission.

#Syntax:
#    diagramc.sh
#to be prompted for input parameters.

#Alternate Syntax:
#    diagramc.sh filename(s)
#to append diagram of file(s) into diagram.out

  if  (${?noclobber}) then
    unset noclobber
    set   noclobbersave
  endif

  if $1a == a  then
    diagramc
    goto quit
  endif

loop:
  echo ========================-- $1 --========================
  #Prompt answers: input from $1, output to diagram2.sc (for now),
  # place numbers in column 67, notate comments with =,
  # don't use IBM PC graphics.

  echo $1          >  diagram.sc
  echo diagram2.sc >> diagram.sc
  echo 67          >> diagram.sc
  echo 1           >> diagram.sc
  echo 0           >> diagram.sc
  diagramc < diagram.sc
  cat diagram2.sc >> diagram.out
```

```
  rm -f diagram.sc
  rm -f diagram2.sc
  shift
  if ! ($1a == a) then
    goto loop
  endif
quit:
  echo Note--This does not delete diagram.out before appending to it.
  if  (${?noclobbersave}) then
    set   noclobber
    unset noclobbersave
  endif
------------------END diagramc.sh--------------------
-----------------BEGIN diagramc.bat--------------------
rem                    ---diagramc.bat---
rem MS-DOS procedure to diagram a C language program.

rem  by Mitchell R Grunes.

rem I assume that the executable is in directory c:\grunes on
rem  your PC.

rem Syntax:
rem     diagramc
rem to be prompted for input parameters.

rem Alternate Syntax:
rem     diagramc filename(s)
rem to append diagram of file(s) into diagram.out

if %1a == a  c:\grunes\diagramc
if %1a == a  goto quit

echo off
:loop
  echo ========================-- %1 --========================
  rem Prompt answers: input from %1, output to diagram2.sc (for now),
  rem  place numbers in column 67, notate comments with =,
  rem  diagram pre-processor blocks, use IBM PC graphics.

  echo %1          >  diagram.sc
  echo diagram2.sc >> diagram.sc
  echo 67          >> diagram.sc
  echo 1           >> diagram.sc
  echo 1           >> diagram.sc
  echo 1           >> diagram.sc
  c:\grunes\diagramc < diagram.sc
  type diagram2.sc >> diagram.out
```

```
  del diagram.sc
  del diagram2.sc
  shift
  if not %1a == a goto loop
:quit
  echo Note--This does not delete diagram.out before appending to it.
-----------------END diagramc.bat-------------------
-----------------BEGIN diagramc.vax-------------------
$!                        ---diagramc.vax---
$!VAX VMS procedure to diagram a C language program
$!
$! by Mitchell R Grunes.
$!
$!I assume that the executable and this procedure are in the search
path,
$!  and that this procedure has execute permission.
$!
$!Syntax:
$!    @diagramc.vax
$!to be prompted for input parameters.
$!
$!Alternate Syntax:
$!    @diagramc.vax filename(s)
$!to append diagram of file(s) into diagram.out
$
$  if P1 .EQS. ""
$  then
$    define/user sys$input sys$command
$    run diagramc
$    goto quit
$  endif
$
$  write sys$output "========================-- "+P1+"
--========================"
$
$! Must pre-create diagram.out if does not exist
$  open/append/error=noSkip diagram.out diagram.out
$  goto Skip
$noSkip:
$  open/write diagram.out diagram.out
$Skip:
$  close diagram.out
$
$! Must pre-create diagram2.sc with same file attributes
$  open/write diagram2.sc diagram2.sc
$  close diagram2.sc
$
$  !Prompt answers: input from P1, output to diagram2.sc (for now),
```

```
$  ! place numbers in column 67, notate comments with =,
$  ! don't use IBM PC graphics.
$
$  open/write diagram.sc diagram.sc
$  write diagram.sc "$Run diagramc"
$  write diagram.sc P1
$  write diagram.sc "diagram2.sc"
$  write diagram.sc "67"
$  write diagram.sc "1"
$  write diagram.sc "0"
$  close diagram.sc
$  @diagram.sc
$  append diagram2.sc diagram.out
$  delete diagram.sc;*
$  delete diagram2.sc;*
$
$  if (P2 .NES. "") then @diagramc.vax 'P2' 'P3' 'P4' 'P5' 'P6' 'P7'
'P8'
$  write sys$output "Note--This does not delete diagram.out before
appending to it."
$quit:
------------------END diagramc.vax--------------------
-----------------BEGIN diagramf.f-------------------
c EXAMPLE OF OUTPUT (looks better if you choose IBM PC line graphics):

c      +---------------- subroutine a(x)            |   1
c      |+---------------- do i=1,5                  |   2
c      ||+---------------- if(i/2*2.eq.i)then       |   3
c      |||                 x=x*i                   |   4
c      ||+---------------- else                     |   5
c      |||                 x=x/i                   |   6
c      ||+---------------- endif                    |   7
c      |+---------------- enddo                     |   8
c      +---------------- end                        |   9

c Diagrams FORTRAN if-else-elseif-endif, do-enddo and case constructs,
c  start and end of routines, type definitions, modules and interfaces;
c  puts a * next to goto, return, cycle, exit, stop, end= and err=.

c Designed by mitch grunes, in his own time.

c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
c Revision date: 8/25/96.
c If you find it useful, or find a problem, please send me e-mail.

c ---------------------------------------------------
c It is VERY IMPORTANT that you select the right FORTRAN
c format.  In CARD format, a C in column 1 marks a
```

```
c comment, and anything in column 6 marks a continuation
c line.  That is not true in FREE format.  Most traditional
c FORTRAN code is in card format.
c ------------------------------------------------
c This program was written in FORTRAN, for historic reasons.
c This was written in Fortran 77 (with common extensions) for
c  portability.  It should also compile under Fortran 90 and Fortran
95,
c  provided you tell the compiler it is in card format.
 c---------------------------------------------------------- ----------

c It can be confused if an INCLUDE block contains a structure that
c  begins inside and ends outside (or vice-versa).

c I hope this works for you, but bear in mind that nothing short of
c  a full-fledged language parser could really do the job.  Perhaps
c  worth about what you paid for it.    (-:

c Versions: To diagram Fortran:     diagramf.f
c                   IDL/PV-WAVE: diagrami.f
c                   C:          diagramc.f
c MS-DOS procedures to call above programs without asking so many
c  questions, append output to file diagram.out:
c                   Fortran:     diagramf.bat (card format)
c                               diagram9.bat (free format)
c                   IDL/PV-WAVE: diagrami.bat
c                   C:          diagramc.bat
c Similar Unix csh procedures:
c                   Fortran:     diagramf.sh  (card format)
c                               diagram9.sh  (free format)
c                   IDL/PV-WAVE: diagrami.sh
c                   C:          diagramc.sh
c Similar Vax VMS DCL procedures:
c                   Fortran:     diagramf.vax (card format)
c                               diagram9.vax (free format)
c                   IDL/PV-WAVE: diagrami.vax
c                   C:          diagramc.vax

      program diagramf                  ! Diagrammer for
Fortran
      character*80 filnam,filnam2

      print*,'FORTRAN source filename?'
      read(*,'(a80)')filnam
      print*,filnam

      print*,'Output file (blank=screen)?'
      read(*,'(a80)')filnam2
```

```
      print*,filnam2

      print*,'Column in which to write line #''s ',
     &   '(0 for none; 67 for 80 col screen; 73 to show card format):'
      LCol=0
      read*,LCol
      print*,LCol

      print*,'Embed include files (0=no; 1?):'
      iembed=1
      read*,iembed
      print*,iembed
      print*,' '
       print*,'-------------------------------------------------- -'
      print*,'It is VERY IMPORTANT that you select the right FORTRAN'
      print*,'format.  In CARD format, a C in column 1 marks a'
      print*,'comment, and anything in column 6 marks a continuation'
      print*,'line.  That is not true in FREE format.'
       print*,'-------------------------------------------------- -'
      print*,'0=Card format (cols 1-6 special, warnings past 72)'
      print*,'1=Free format'
      print*,'2=Card format (same as 0, ignore cols past 72)'
      print*,'Format # (0?):'
      ifree=0
      read*,ifree
      print*,ifree

      print*,'Use IBM PC graphics characters (0=no):'
      igraphics=0
      read*,igraphics
      print*,igraphics

      call diagram(filnam,filnam2,LCol,iembed,ifree,igraphics)
      end
c------------------------------------------------------------ ------------
      subroutine diagram(filnam,filnam2,LCol,iembed,ifree,igraphics)
c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
      character*80 filnam,filnam2
      character*160 a,b,AfterSemi
      character*5 form
      character*8 fm
      character*1 c,c2
      logical find
      external find
      common iCol,iCol1
      character*10 label(100)
      logical fout
```

```
c Symbols which will mark block actions:
      character*1 BlockBegin    (2) /'+','+'/  ! Start of block
      character*1 BlockEnd      (2) /'+','+'/  ! End of block
      character*1 BlockElse     (2) /'+','+'/  ! Else construct
      character*1 BlockContinue (2) /'|','|'/  ! Block continues w/o
change
      character*1 BlockHoriz    (2) /'-','-'/  ! Horizontal to start
of line
c Same, but allows horizontal line to continue through:
      character*1 BlockBeginH   (2) /'+','+'/  ! Start of block
      character*1 BlockEndH     (2) /'+','+'/  ! End of block
      character*1 BlockElseH    (2) /'+','+'/  ! Else construct

      if(iGraphics.ne.0)then
        iGraphics=1

        BlockBegin   (1)=char(218)           ! (1)=normal
        BlockEnd     (1)=char(192)
        BlockElse    (1)=char(195)
        BlockContinue(1)=char(179)
        BlockHoriz   (1)=char(196)
        BlockBeginH  (1)=char(194)
        BlockEndH    (1)=char(193)
        BlockElseH   (1)=char(197)

        BlockBegin   (2)=char(214)           ! (2)=DO/FOR loops
(doubled)
        BlockEnd     (2)=char(211)          ! (not yet used)
        BlockEnd     (2)=char(211)
        BlockElse    (2)=char(199)
        BlockContinue(2)=char(186)
        BlockHoriz   (2)=char(196)
        BlockBeginH  (2)=char(209)
        BlockEndH    (2)=char(208)
        BlockElseH   (2)=char(215)
      endif

      open(1,file=filnam,status='old')
      fout=filnam2.gt.' '
      if(fout)open(2,file=filnam2,status='unknown')
                                  ! ASCII 12 is a form
feed
      if(fout)write(2,*)char(12),
     &   '==============--',filnam(1:LenA(filnam)),'--=============='

      if(fout)     write(2,'(11x,a50,a49,/)')  ! Write column header
     &   '...,....1....,....2....,....3....,....4....,....5',
     &   '...,....6....,....7....,....8....,....9....,....'
```

```
      if(.not.fout)write(*,'(11x,a50,a49,/)')' ',
     &  '....,....1....,....2....,....3....,....4....,....5',
     &  '....,....6....,....7....,....8....,....9....,....'

      i1=0                          ! # of nest levels
before
                                   !  current line
      i2=0                          ! # of nest levels on
                                   !  current line
      i3=0                          ! # of nest levels
after
                                   !  current line
      i4=0                          ! not 0 to flag start
or end
                                   !  of block
      InSub=0                        ! Inside a subroutine,
                                   !  function or mainline
      InMod=0                         ! Inside module or
                                   !   contains
      nMain=0                         ! no mainline program
yet
      InElse=0                       ! Found elseif, but not
then
      nlabel=0                      ! # of labels for do
loop
                                   !  ends
      iAlphaNum=0                          ! Last char of line is
                                   !  alpha-numeric
      iContinueOld=0                       ! next line not
continued line
      nline=0
      iunit=1
10     a=' '
      read(iunit,'(a160)',end=99)a
      nline=nline+1
      fm=' '
      write(fm,'(i5)')nline
      form=fm

      if(a(1:1).eq.char(12))then
        if(fout)write(2,'(a1,:)')char(12)
        if(.not.fout)print*,'------------FORM FEED------------'
        b=a(2:160)
        a=b
      endif

      b=' '                          ! Turn tabs to spaces
      j=1
```

```
      do i=1,LenA(a)
        if(a(i:i).eq.char(9))then
          j=(j-1)/8*8+8+1
      ! Make sure is good ASCII char
        elseif(j.le.160.and.a(i:i).ge.'
'.and.a(i:i).lt.char(128))then
          b(j:j)=a(i:i)
          j=j+1
        endif
      enddo

      a=' '                          ! Pre-processed output
      i=1                            ! Basic pre-processing
      j=1
      i72flag=0                        ! nothing over column
72
                                     ! yet
      iOldAlphaNum=iAlphaNum               ! last line ended in
                                     ! alpha-numeric?
      iAlphaNum=0
      iContinue=iContinueOld               ! This line continued
line?
      if(find(b,'&',2,0))iContinue=1       !  will be changed to 2
after
                                     !  first non/blank.
      if(iContinue.eq.0)then
        iquote=0                       ! no ' yet
        idquote=0                       ! no '' yet
      endif
      j=1
                                     ! comment line
      if((b(1:1).eq.'c'.or.b(1:1).eq.'C').and.ifree.ne.1)goto 15
      if(b(1:1).eq.'*'.or.b(1:2).eq.'??')goto 15

      do i=1,LenA(b)
        c=b(i:i)
                                     ! handle upper case
        if(c.ge.'A'.and.c.le.'Z')c=char(ichar(c)+32)
                                     ! ASCII 33 is '!'
        if(c.eq.char(33).and.iquote.eq.0.and.idquote.eq.0)goto 15

        if(i.gt.72.and.c.ne.' ')then
          if(ifree.eq.0.and.i72flag.eq.0)then
            i72flag=1
            PRINT*,'***WARNING--PAST COLUMN 72 at line',form
            if(fout)print*,b
            print*,char(7)
          elseif(ifree.eq.2)then
```

```
            c=' '
          endif
        endif

        if(c.eq.'"'.and.(i.ne.6.or.ifree.ne.0).and.idquote.eq.0)
     &    iquote=1-iquote
        if(c.eq."'" .and.(i.ne.6.or.ifree.ne.0).and.iquote .eq.0)
     &    idquote=1-idquote
        if(iquote.eq.1)then
          if(find(a,'include ',2,0).and.iembed.ne.0)then
            iquote=0
            idquote=0
          endif
        endif
        if(iquote.ne.0.or.idquote.ne.0)c=' '
        if(j.gt.1)then                    ! (kill multiple
spaces,
                                  !  and spaces around =)
          c2=a(j-1:j-1)
          if(c.eq.' '.and.c2.eq.' ')j=j-1
          if(c.eq.'='.and.c2.eq.' ')j=j-1
          if(c.eq.' '.and.c2.eq.'=')j=j-1
          if(c.eq.' '.and.c2.eq.'=')c='='
        endif
                                  ! Look for
                                  ! identifiers that wrap
                                  ! around lines.
        if((i.gt.6.or.ifree.ne.0).and.c.ne.' '.and.c.ne.'&')then
          iAlphaNum=0
          if((c.ge.'a'.and.c.le.'z').or.
     &       (c.ge.'0'.and.c.le.'9'))then
            iAlphaNum=1
            if(iContinue.eq.1)then
              if(iOldAlphaNum.ne.0)then
                PRINT*,'***POSSIBLE SPLIT IDENTIFIER across
line',form
                print*,char(7)
              endif
            endif
          endif
          iContinue=2
        endif

        if(j.le.160)a(j:j)=c
        j=j+1
      enddo

15    iContinueOld=0
```

```
        if(a(LenA(a):LenA(a)).eq.'&')iContinueOld=1

        i2=i1
        i3=i1
        i4=0
        igoto=0                         ! no goto on line
        Main1=0                          ! (Not mainline)
                                       ! Possible mainline
start

16      AfterSemi=' '                     ! Break line at
semicolons
        if(find(a,';',0,160-1))then
          AfterSemi='      '//a(icol:160)
          a=a(1:icol1-1)
        endif

        if(a.ne.' '.and.InSub.eq.0.and.InMod.eq.0)Main1=1
                                       ! Mark various types of
jump
        if(find(a,'go to',8+64,0).or.find(a,'goto',8+64,0).or.
     &     find(a,'end=',16,0)   .or.find(a,'err=',16,0)  .or.
     &     find(a,'return',8+64,0).or.find(a,'cycle ',8,0).or.
     &     find(a,'exit ',8,0)   .or.find(a,'stop ',8,0))
     &   igoto=1

        if(find(a,')1',64,0).or.find(a,')2',64,0).or.
     &     find(a,')3',64,0).or.find(a,')4',64,0).or.
     &     find(a,')5',64,0).or.find(a,')6',64,0).or.
     &     find(a,')7',64,0).or.find(a,')8',64,0).or.
     &     find(a,')9',64,0))
     &   igoto=1

        if(find(a,') 1',64,0).or.find(a,') 2',64,0).or.
     &     find(a,') 3',64,0).or.find(a,') 4',64,0).or.
     &     find(a,') 5',64,0).or.find(a,') 6',64,0).or.
     &     find(a,') 7',64,0).or.find(a,') 8',64,0).or.
     &     find(a,') 9',64,0))
     &   igoto=1

        if(find(a,'::',0,0))then             ! To distinguish
          iDeclare=iCol                       !  declarations from
                                       !  keywords
        else
          iDeclare=999
        endif

        if(find(a,'include "',2,0).and.iembed.ne.0)then
```

```
        filnam=a(iCol:160)
        if(.not.find(filnam,"",0,0))goto 20
        filnam(iCol-1:80)=' '
        if(fout)print*,'including file ',filnam(1:50)
        close(3)
        open(3,file=filnam,status='old',err=17)
        iunit=3
        nlinesave=nline
        nline=0
        i2=i2+1
        i3=i3+1
        goto 20
17      PRINT*,'***WARNING--Missing include file***'
        print*,char(7)
      elseif(find(a,'end module ',2,0).or.
     &       find(a,'endmodule ',2,0).or.
     &       find(a,'end interface',2,0).or.
     &       find(a,'endinterface',2,0).or.
     &       find(a,'end type ',2,0).or.
     &       find(a,'endtype ',2,0))then
        i3=i3-1
        InMod=InMod-1
        if(find(a,'endmodule ',2,0).or.
     &     find(a,'end module ',2,0))then
          InMod=0
          if(InSub.gt.0.or.i3.ne.0)then
            PRINT*,'***ERROR--INVALID DIAGRAMMING INDEX line',form
            if(fout)WRITE(2,*)
     &        '***ERROR--INVALID DIAGRAMMING INDEX!***'
            if(fout)print*,b
            print*,char(7)
          endif
        endif
        InElse=0
      elseif(find(a,'enddo ',256,0).or.
     &       find(a,'end do ',256,0))then
        i3=i3-1
        nlabel=max(0,nlabel-1)
        InElse=0
      elseif(find(a,'endif  ',256,0).or.
     &       find(a,'end if  ',256,0).or.
     &       find(a,'endselect ',256,0).or.
     &       find(a,'end select ',256,0).or.
     &       find(a,'endforall ',256,0).or.
     &       find(a,'end forall ',256,0).or.
     &       find(a,'endforall ',256,0).or.
     &       find(a,'end where ',256,0).or.
     &       find(a,'endwhere ',256,0))then
```

```
      i3=i3-1
      InElse=0
    elseif(find(a,'end  ',256,0).or.
&         find(a,'end function ',256,0).or.
&         find(a,'endfunction ',256,0).or.
&         find(a,'end subroutine ',256,0).or.
&         find(a,'endsubroutine ',256,0).or.
&         find(a,'end program ',256,0).or.
&         find(a,'endprogram ',256,0).or.
&         find(a,'end block',256,0).or.
&         find(a,'endblock',256,0))then
      i3=i3-1
      InSub=InSub-1
      if(InSub.lt.0.or.(InSub.gt.0.and.InMod.le.0))then
        if(InSub.lt.0.and.InMod.gt.0.and.find(a,'end  ',256,0))then
          InSub=0
          InMod=InMod-1
        else
          PRINT*,'***ERROR--INVALID DIAGRAMMING INDEX line',form
          if(fout)
&           WRITE(2,*)'***ERROR--INVALID DIAGRAMMING INDEX!***'
          if(fout)print*,b
          print*,char(7)
        endif
      endif
      if(i3.eq.0)InSub=0
      InElse=0
    elseif(find(a,'elseif',128+256,0).or.
&         find(a,'else if',128+256,0))then
      i4=max(i4,1)
      InElse=0
      if(.not.find(a,'then  ',8,0))InElse=1
    elseif(find(a,'then  ',8,0))then
      i2=i2+1
      if(InElse.eq.0)i3=i3+1
      InElse=0
    elseif( find(a,'selectcase',256,0).or.
&           find(a,'select case',256,0))then
      i2=i2+1
      i3=i3+1
      i4=max(i4,1)
      InElse=0
    elseif(find(a,'else  ',256,0).or.
&         find(a,'entry ',4,0).or.
&         find(a,'case ',256,0).or.
&         find(a,'case(',256,0).or.
&         find(a,'contains ',2,0).or.
&         find(a,'elsewhere ',256,0).or.
```

```
&         find(a,'else where ',256,0))then
      i4=max(i4,1)
      InElse=0
      if(find(a,'contains ',2,0))then
        if(fout)print*,'Line ',form,' ',b(1:LenA(b))
        InMod=InMod+1
      endif
    elseif( find(a,'selectcase',256,0).or.
&           find(a,'select case',256,0).or.
&           find(a,'for all (',256,0).or.
&           find(a,'forall (',256,0).or.
&           find(a,'for all(',256,0).or.
&           find(a,'forall(',256,0))then
      i2=i2+1
      i3=i3+1
      InElse=0
    elseif( find(a,'where (',256,0).or.
&           find(a,'where(',256,0))then
      if(find(a,'(',0,0))iCol=iCol
      iCntParen=1
      do i=iCol,LenA(a)
        if(a(i:i).eq.'(')iCntParen=iCntParen+1
        if(a(i:i).eq.')')iCntParen=iCntParen-1
        if(iCntParen.eq.0)then
          if(a(i:160).eq.')')then
            i2=i2+1
            i3=i3+1
            InElse=0
          endif
          goto 20
        endif
      enddo
    elseif((find(a,'module ',2,iDeclare).and.
&         .not.find(a,'module procedure',2,iDeclare)).or.
&          find(a,'interface ',2,iDeclare).or.
&         (find(a,'type ',2,iDeclare).and.
&         .not.find(a,'(',0,iDeclare)).or.
&         (find(a,'type,',2,iDeclare).and.
&         .not.find(a,'(',0,iDeclare)))then
      if(fout)print*,'Line ',form,' ',b(1:LenA(b))
      i2=i2+1
      i3=i3+1
      Main1=0
      if(find(a,'module ',2,iDeclare).and.InMod.ne.0)then
        PRINT*,'***ERROR--NESTED MODULES***'
        if(fout)WRITE(2,*)'***NESTED MODULES***'
        if(fout)print*,b
        print*,char(7)
```

```
      endif
      InMod=InMod+1
      InElse=0
    elseif(find(a,'do while',128+256,0).or.
&         find(a,'dowhile',128+256,0))then
      i2=i2+1
      i3=i3+1
      nlabel=min(100,nlabel+1)
      label(nlabel)='####'
      InElse=0
    elseif(find(a,' do ',256,0).or.
&   (ifree.ne.0.and.a(1:3).eq.'do '))then
     if(ifree.ne.0.and.a(1:3).eq.'do ')iCol=4
      if(iCol1.lt.7.or.a(7:max(7,iCol1)).eq.' '.or.
&     (ifree.ne.0.and.a(1:3).eq.'do '))then
        i2=i2+1
        i3=i3+1
        iCol2=iCol
        dowhile(iCol2.lt.160.and.a(iCol2:iCol2).ge.'0'.and.
&         a(iCol2:iCol2).le.'9')
          iCol2=iCol2+1
        enddo
        iCol2=iCol2-1
        nlabel=min(100,nlabel+1)
        if(iCol2.ge.iCol)then
          label(nlabel)=a(iCol:iCol2)
        else
          label(nlabel)='####'
        endif
      endif
      InElse=0
    elseif(find(a,': do ',0,0).or.find(a,':do ',0,0))then
      i2=i2+1
      i3=i3+1
      InElse=0
    elseif(find(a,'function ',4,iDeclare).or.
&         find(a,'subroutine ',4,iDeclare).or.
&         find(a,'program ',2,iDeclare) .or.
&         find(a,'block data ',2,iDeclare).or.
&         find(a,'blockdata ',2,iDeclare))then
      if(fout)print*,'Line ',form,' ',b(1:LenA(b))
      if(InSub.ne.0.and.InMod.eq.0)then
        PRINT*,'***ERROR--ROUTINE INSIDE ROUTINE***'
        if(fout)WRITE(2,*)'***ERROR--ROUTINE INSIDE ROUTINE***'
        if(fout)print*,b
        print*,char(7)
      endif
      Main1=0
```

```
        InSub=InSub+1
        i2=i2+1
        i3=i3+1
        if(InSub.eq.1.and.i3.ne.1.and.InMod.le.0)then
          PRINT*,'***ERROR--INVALID DIAGRAMMING INDEX line',form
          if(fout)
     &      WRITE(2,*)'***ERROR--INVALID DIAGRAMMING INDEX!***'
          if(fout)print*,b
          print*,char(7)
          i3=1
        endif
        InElse=0
      endif

20    if(Main1.ne.0)then                    ! Was start of mainline
        if(fout)print*,'Line ',form,' ',b(1:LenA(b))
        if(nMain.gt.0)then
          PRINT*,'***ERROR--TOO MANY MAINLINES***'
          if(fout)WRITE(2,*)'***ERROR--TOO MANY MAINLINES!***'
          if(fout)print*,b
          print*,char(7)
        endif
        InSub=InSub+1
        nMain=nMain+1
        i2=i2+1
        i3=i3+1
      endif

21    if(b(1:5).ne.' '.or.ifree.ne.0)then     ! Search for DO labels
        istart=1
        dowhile(istart.lt.160.and.b(istart:istart).eq.' ')
          istart=istart+1
        enddo
        iend=istart
        dowhile(iend.lt.160.and.
     &    (b(iend:iend).ge.'0'.and.b(iend:iend).le.'9'))
          iend=iend+1
        enddo
        iend=iend-1
        if(iend.ge.1.and.b(1:max(1,iend)).ne.' ')then
          do i=1,nlabel
            j=nlabel+1-i                  !  (in reverse order)
            if(b(istart:iend).eq.label(j))then
              i3=i3-1
              nlabel=max(0,j-1)
              goto 21
            endif
          enddo
```

```
        endif
      endif

      if(AfterSemi.ne.' ')then
        a=AfterSemi
        goto 16
      endif

      a=' '
      if(i1.lt.0.or.i2.lt.0.or.i3.lt.0.or.i4.lt.0)then
        PRINT*,'***ERROR--INVALID DIAGRAMMING INDEX line',form
        if(fout)WRITE(2,*)'***ERROR--INVALID DIAGRAMMING INDEX!***'
        if(fout)print*,b
        print*,char(7)
        i1=max(i1,0)
        i2=max(i2,0)
        i3=max(i3,0)
        i4=max(i4,0)
      endif

      i2=max(i1,i3)                    ! # of nests on current
line
      i4=max(i4,iabs(i3-i1))              ! not 0, to flag start
or
                                  !  end of block

      iBlock=1                       ! For the present
version.

      a=' '                           ! Leave space for
diagram
      a(12:160)=b                      !  (must match column
header)

      LastUse=1                        ! Last usable diagram
col
      dowhile(LastUse.lt.160.and.a(LastUse:LastUse).eq.' ')
        LastUse=LastUse+1
      enddo
      LastUse=LastUse-2

      if(igoto.ne.0)a(1:1)='*'           ! Place * next to jumps

      if(i2.gt.0)then                  ! Draw one vertical
line per
        do i=2,min(i2+1,LastUse)           !  nest level.
          a(i:i)=BlockContinue(iBlock)
        enddo
```

```
      endif

      if(i4.ne.0)then                    ! Draw horizontal lines
inward
        do i=i2+2,LastUse                !  from above.
          a(i:i)=BlockHoriz(iBlock)
        enddo
      endif

      do i=0,i4-1                        ! May need to replace
some
                                         !  vertical lines with
        c=           BlockElse(iBlock)   !      else  symbol
        if(i1+i.lt.i3)c=BlockBegin(iBlock)   !  or   begin symbol
        if(i1+i.gt.i3)c=BlockEnd   (iBlock)  !  or   end   symbol
        j=max(2,min(LastUse,i2+1-i))
        a(j:j)=c
        if(a(j+1:j+1).eq.BlockElse  (iBlock)) ! Continue horizontal
lines
     &     a(j+1:j+1)  = BlockElseH (iBlock)
        if(a(j+1:j+1).eq.BlockBegin (iBlock))
     &     a(j+1:j+1)  = BlockBeginH(iBlock)
        if(a(j+1:j+1).eq.BlockEnd   (iBlock))
     &     a(j+1:j+1)  = BlockEndH  (iBlock)
      enddo

      if(LCol.gt.0.and.a(max(1,LCol+11):160).eq.' ')then       ! line
#
        if(form(1:1).eq.' ')form(1:1)=BlockContinue(iBlock)
        a(LCol+11:160)=form
      endif

      n=LenA(a)                          ! Output diagrammed
line
      if(fout)     write(2,'(80a1,80a1)')(a(i:i),i=1,n)
      if(.not.fout)write(*,'(1x,80a1,80a1)')(a(i:i),i=1,n)

      i1=i3
      goto 10
99    if(iunit.eq.3)then
        iunit=1
        i1=i1-1
        close(3)
        nline=nlinesave
        goto 10
      endif
      if(i3.gt.0.or.InSub.ne.0)then
        PRINT*,'***WARNING--SOME NEST LEVELS LEFT HANGING AT END***'
```

```
        print*,char(7)
      endif
      end
c------------------------------------------------------------ ------------
      logical function find(a,b,icond,jcol)   ! find b in a, subject
                                    !  to conditions:
                                    ! Column is prior to
jcol
                                    !  (if jcol.ne.0)
                                    ! icond=sum of the
                                    !  following:
                                    ! 1:  Prior, if exists,
must
                                    !     be blank
                                    ! 2:  Must be first
non-blank
                                    ! 4:  Prior character,
if
                                    !  present, must not be
                                    !  alphanumeric.
                                    ! 8:  Prior character,
if
                                    !  present, must be
blank
                                    !  or )
                                    ! 16: Prior character,
if
                                    !  present, must be
blank
                                    !  or ,
                                    ! 32: Next character
not
                                    !  alphanumeric
                                    ! 64: Next character
not
                                    !  alphabetic
                                    ! 128:Next character
must
                                    !  be blank or (
                                    ! 256:1st non-blank,
                                    !  possibly except for
                                    !  numeric labels
                                    ! 512  Prior character,
if present,
                                    !      must be blank or
) or }
                                    !      or { or ;
c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
```

```
c Revision date: 8/25/96.
      character*(*) a,b
      character*1   c,cNext,c2
      common iCol,iCol1
      logical result

      ii=len(a)
      jj=len(b)
      result=.false.
      jjcol=999
      if(jcol.gt.0)jjcol=jcol
      do i=1,min(ii-jj+1,jjcol)
        if(a(i:i+jj-1).eq.b)then            ! Found--Now do tests
          iCol1=i                        ! iCol1=column of item
                                       !  found
          iCol =i+jj                     ! iCol =column after
                                       !  item found

          c=' '
          cNext=' '
          if(iCol1.gt.1)c=a(iCol1-1:iCol1-1)
          if(iCol .le.ii)cNext=a(iCol:iCol)

          result=.true.
          if(result.and.iand(icond,1).ne.0.and.icol1.gt.1)then
            result=c.eq.' '
          endif

          if(result.and.iand(icond,2).ne.0.and.iCol1.gt.1)then
            result=a(1:iCol1-1).eq.' '
          endif

          if(result.and.iand(icond,4).ne.0)
     &      result=(c.lt.'0'.or.c.gt.'9').and.(c.lt.'a'.or.c.gt.'z')
          if(result.and.iand(icond,8).ne.0)result=c.eq.'
'.or.c.eq.')'

          if(result.and.iand(icond,16).ne.0)
     &      result=c.eq.' '.or.c.eq.','

          if(result.and.iand(icond,32).ne.0)
     &      result=(cNext.lt.'0'.or.cNext.gt.'9').and.
     &             (cNext.lt.'a'.or.cNext.gt.'z')

          if(result.and.iand(icond,64).ne.0)
     &      result=(cNext.lt.'a'.or.cNext.gt.'z')

          if(result.and.iand(icond,128).ne.0)
```

```
     &       result=cNext.eq.' '.or.cNext.eq.'('

            if(result.and.iand(icond,256).ne.0.and.iCol1.gt.1)then
              do iii=1,iCol1-1
                c2=a(iii:iii)
                if((c2.lt.'0'.or.c2.gt.'9').and.c2.ne.'
')result=.false.
              enddo
            endif

            if(result.and.iand(icond,512).ne.0)result=c.eq.' '
     &       .or.c.eq.';'.or.c.eq.')'.or.c.eq.'{'.or.c.eq.'}'

            find=result
            if(result)return
          endif
        enddo
        find=result
        end
 c------------------------------------------------------------ ------------
        function LenA(a)                    ! Length of string, at
                                          !  least 1
c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
c Revision date: 8/25/96.
        character*(*) a
        n=len(a)
        dowhile(n.gt.1.and.a(n:n).eq.' ')
          n=n-1
        enddo
        LenA=n
        end
------------------END diagramf.f--------------------
-----------------BEGIN diagramf.sh-------------------
#!/bin/csh
#                       ---diagramf.sh---
#Unix csh procedure to diagram a (card format) Fortran language
program.

#On some unix systems $1 should be replaced by %1.

# by Mitchell R Grunes.
# for his own use, in his own time

#I assume that the executable and this procedure are in the search
path,
#  and that this procedure has execute permission.

#Syntax:
```

```
#    diagramf.sh
#to be prompted for input parameters.

#Alternate Syntax:
#    diagramf.sh filename(s)
#to append diagram of file(s) into diagram.out

  if  (${?noclobber}) then
    unset noclobber
    set   noclobbersave
  endif

  if $1a == a  then
    diagramf
    goto quit
  endif

loop:
  echo ========================-- $1 --========================
  #Prompt answers: input from $1, output to diagram2.sc (for now),
  # place numbers in column 73, embed include files, don't use free
  # format, don't use IBM PC graphics.

  echo $1          >  diagram.sc
  echo diagram2.sc >> diagram.sc
  echo 73          >> diagram.sc
  echo 1           >> diagram.sc
  echo 0           >> diagram.sc
  echo 0           >> diagram.sc
  diagramf < diagram.sc
  cat diagram2.sc >> diagram.out
  rm -f diagram.sc
  rm -f diagram2.sc
  shift
  if ! ($1a == a) then
    goto loop
  endif
quit:
  echo Note--This does not delete diagram.out before appending to it.
  if  (${?noclobbersave}) then
    set   noclobber
    unset noclobbersave
  endif
------------------END diagramf.sh--------------------
-----------------BEGIN diagramf.bat-------------------
rem                    ---diagramf.bat---
rem MS-DOS procedure to diagram a (card format) FORTRAN language
program.
```

```
rem (use diagram9.bat to diagram free format Fortran programs)

rem  by Mitchell R Grunes.

rem I assume that the executable is in directory c:\grunes on
rem  your PC.

rem Syntax:
rem     diagramf
rem to be prompted for input parameters.

rem Alternate Syntax:
rem     diagramf filename(s)
rem to append diagram of file(s) into diagram.out

if %1a == a  c:\grunes\diagramf
if %1a == a  goto quit

echo off
:loop
  echo ========================-- %1 --========================
  rem Prompt answers: input from %1, output to diagram2.sc (for now),
  rem  place numbers in column 73, embed include files, don't use free
  rem  format, use IBM PC graphics.

  echo %1          >  diagram.sc
  echo diagram2.sc >> diagram.sc
  echo 73          >> diagram.sc
  echo 1           >> diagram.sc
  echo 0           >> diagram.sc
  echo 1           >> diagram.sc
  c:\grunes\diagramf < diagram.sc
  type diagram2.sc >> diagram.out
  del diagram.sc
  del diagram2.sc
  shift
  if not %1a == a goto loop
:quit
  echo Note--This does not delete diagram.out before appending to it.
------------------END diagramf.bat--------------------
-----------------BEGIN diagramf.vax--------------------
$!                      ---diagramf.vax---
$!VAX VMS procedure to diagram a (card format) Fortran language program
$!
$! by Mitchell R Grunes.
$!
$!I assume that the executable and this procedure are in the search
path,
```

```
$!  and that this procedure has execute permission.
$!
$!Syntax:
$!    @diagramf.vax
$!to be prompted for input parameters.
$!
$!Alternate Syntax:
$!    @diagramf.vax filename(s)
$!to append diagram of file(s) into diagram.out
$
$  if P1 .EQS. ""
$  then
$    define/user sys$input sys$command
$    run diagramf
$    goto quit
$  endif
$
$  write sys$output "========================-- "+P1+"
--======================="
$  !Prompt answers: input from P1, output to diagram2.sc (for now),
$  ! place numbers in column 73, embed include files, don't use free
$  ! format, don't use IBM PC graphics.
$
$! Must pre-create diagram.out if does not exist
$  open/append/error=noSkip diagram.out diagram.out
$  goto Skip
$noSkip:
$  open/write diagram.out diagram.out
$Skip:
$  close diagram.out
$
$! Must pre-create diagram2.sc with same file attributes
$  open/write diagram2.sc diagram2.sc
$  close diagram2.sc
$
$  open/write diagram.sc diagram.sc
$  write diagram.sc "$Run diagramf"
$  write diagram.sc P1
$  write diagram.sc "diagram2.sc"
$  write diagram.sc "73"
$  write diagram.sc "1"
$  write diagram.sc "0"
$  write diagram.sc "0"
$  close diagram.sc
$  @diagram.sc
$  append diagram2.sc diagram.out
$  delete diagram.sc;*
$  delete diagram2.sc;*
```

```
$
$  if (P2 .NES. "") then @diagramf.vax 'P2' 'P3' 'P4' 'P5' 'P6' 'P7'
'P8'
$quit:
$  write sys$output "Note--This does not delete diagram.out before
appending to it."
------------------END diagramf.vax--------------------
-----------------BEGIN diagram9.sh-------------------
#!/bin/csh
#                       ---diagram9.sh---
#Unix csh procedure to diagram a (free format) FORTRAN language
program.

#On some unix systems $1 should be replaced by %1.

# by Mitchell R Grunes, for his own use, in his own time.

#I assume that the executable and this procedure are in the search
path,
#  and that this procedure has execute permission.

#Syntax:
#    diagram9.sh
#to be prompted for input parameters.

#Alternate Syntax:
#    diagram9.sh filename(s)
#to append diagram of file(s) into diagram.out

  if  (${?noclobber}) then
    unset noclobber
    set   noclobbersave
  endif

  if $1a == a  then
    diagramf
    goto quit
  endif

loop:
  echo ========================-- $1 --========================
  #Prompt answers: input from $1, output to diagram2.sc (for now),
  # place numbers in column 73, embed include files, use free
  # format, don't use IBM PC graphics.

  echo $1          >  diagram.sc
  echo diagram2.sc >> diagram.sc
  echo 73          >> diagram.sc
```

```
  echo 1          >> diagram.sc
  echo 1          >> diagram.sc
  echo 0          >> diagram.sc
  diagramf < diagram.sc
  cat diagram2.sc >> diagram.out
  rm -f diagram.sc
  rm -f diagram2.sc
  shift
  if ! ($1a == a) then
    goto loop
  endif
quit:
  echo Note--This does not delete diagram.out before appending to it.
  if  (${?noclobbersave}) then
    set   noclobber
    unset noclobbersave
  endif
------------------END diagram9.sh--------------------
-----------------BEGIN diagram9.bat-------------------
rem                    ---diagram9.bat---
rem MS-DOS procedure to diagram a (free format) FORTRAN language
program.
rem (use diagramf.bat to diagram card format Fortran programs)

rem  by Mitchell R Grunes.

rem I assume that the executable is in directory c:\grunes on
rem  your PC.

rem Syntax:
rem     diagramf
rem to be prompted for input parameters.

rem Alternate Syntax:
rem     diagramf filename(s)
rem to append diagram of file(s) into diagram.out

if %1a == a  c:\grunes\diagramf
if %1a == a  goto quit

echo off
:loop
  echo ========================-- %1 --========================
  rem Prompt answers: input from %1, output to diagram2.sc (for now),
  rem  place numbers in column 73, embed include files, use free
  rem  format, use IBM PC graphics.

  echo %1         >  diagram.sc
```

```
  echo diagram2.sc >> diagram.sc
  echo 73          >> diagram.sc
  echo 1           >> diagram.sc
  echo 1           >> diagram.sc
  echo 1           >> diagram.sc
  c:\grunes\diagramf < diagram.sc
  type diagram2.sc >> diagram.out
  del diagram.sc
  del diagram2.sc
  shift
  if not %1a == a goto loop
:quit
  echo Note--This does not delete diagram.out before appending to it.
------------------END diagram9.bat--------------------
-----------------BEGIN diagram9.vax-------------------
$!                      ---diagram9.vax---
$!VAX VMS procedure to diagram a (free format) Fortran language program
$!
$! by Mitchell R Grunes.
$!
$!I assume that the executable and this procedure are in the search
path,
$!  and that this procedure has execute permission.
$!
$!Syntax:
$!    @diagram9.vax
$!to be prompted for input parameters.
$!
$!Alternate Syntax:
$!    @diagram9.vax filename(s)
$!to append diagram of file(s) into diagram.out
$
$  if P1 .EQS. ""
$  then
$    define/user sys$input sys$command
$    run diagramf
$    goto quit
$  endif
$
$  write sys$output "=========================-- "+P1+"
--========================"
$  !Prompt answers: input from P1, output to diagram2.sc (for now),
$  ! place numbers in column 73, embed include files, use free
$  ! format, don't use IBM PC graphics.
$
$! Must pre-create diagram.out if does not exist
$  open/append/error=noSkip diagram.out diagram.out
$  goto Skip
```

```
$noSkip:
$  open/write diagram.out diagram.out
$Skip:
$  close diagram.out
$
$! Must pre-create diagram2.sc with same file attributes
$  open/write diagram2.sc diagram2.sc
$  close diagram2.sc
$
$  open/write diagram.sc diagram.sc
$  write diagram.sc "$Run diagramf"
$  write diagram.sc P1
$  write diagram.sc "diagram2.sc"
$  write diagram.sc "73"
$  write diagram.sc "1"
$  write diagram.sc "1"
$  write diagram.sc "0"
$  close diagram.sc
$  @diagram.sc
$  append diagram2.sc diagram.out
$  delete diagram.sc;*
$  delete diagram2.sc;*
$
$  if (P2 .NES. "") then @diagram9.vax 'P2' 'P3' 'P4' 'P5' 'P6' 'P7'
'P8'
$quit:
$  write sys$output "Note--This does not delete diagram.out before
appending to it."
------------------END diagram9.vax--------------------
-----------------BEGIN diagramh.f-------------------
c EXAMPLE OF OUTPUT (looks better if you choose IBM PC line graphics):


c ++-------- <html><head>                        |   1
c ||                                  |   2
c |+-------- <title>My Title</title>             |   3
c ||                                  |   4
c |+-------- </head>                          |   5
c |                                   |   6
c |+-------- <body>                           |   7
c |+-------- <a href="./doc.html">doc.html</a><br>   |   8
c |+-------- </body>                          |   9
c +--------- </html>                         |  10

c Diagrams HTML language constructs,
c  and puts a * next to internal links.  It can
c  place = next to comment blocks.
```

```
c Designed by mitch grunes, in his own time.

c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
c Revision date: 8/25/96.
c If you find it useful, or find a problem, please send me e-mail.

c This program was written in FORTRAN, for historic reasons.
c (For this reason, people who mostly program in C will probably be
c  unwilling to use this program, even as a utility.)

c WARNING: The "/*" sequences will confuse compilers like SGI Fortran
c  that use a C pre-processor by default on Fortran programs, so you
c  must use a compiler switch like "-nocpp" to turn that off.

c It can be confused if an INCLUDE block contains a structure that
c  begins inside and ends outside (or vice-versa).

c  It also does not diagram IF, FOR, ELSE, WHILE, etc., unless you use
c   { and } to enclose the conditionally executed statement--
c   e.g. it will not draw any lines next to

c       if(condition)
c         for (i=0; i<10; i++)
c           a[i]=2;
c       else
c         b=3;

c I hope this works for you, but bear in mind that nothing short of
c  a full-fledged language parser could really do the job.  Perhaps
c  worth about what you paid for it.    (-:

c Versions: To diagram Fortran:     diagramf.f
c                   IDL/PV-WAVE: diagrami.f
c                   C:           diagramc.f
c MS-DOS procedures to call above programs without asking so many
questions,
c  append output to file diagram.out:
c                   Fortran:     diagramf.bat (card format)
c                                diagram9.bat (free format)
c                   IDL/PV-WAVE: diagrami.bat
c                   C:           diagramc.bat
c Similar Unix csh procedures:
c                   Fortran:     diagramf.sh  (card format)
c                                diagram9.sh  (free format)
c                   IDL/PV-WAVE: diagrami.sh
c                   C:           diagramc.sh
c Similar Vax VMS DCL procedures:
c                   Fortran:     diagramf.vax (card format)
```

```
c                           diagram9.vax (free format)
c                  IDL/PV-WAVE: diagrami.vax
c                  C:          diagramc.vax

      program diagramh                          ! Diagrammer
for HTML
      character*80 filnam,filnam2

      print*,'HTML source filename?'
      read(*,'(a80)')filnam
      print*,filnam

      print*,'Output file (blank=screen)?'
      read(*,'(a80)')filnam2
      print*,filnam2

      print*,'Column in which to write line #''s ',
     &   '(67 for 80 col screen, 0 for none):'
      LCol=0
      read*,LCol
      print*,LCol

      print*,'Notate comments with = (0=no, 1=yes; 1?):'
      inotate=1
      read*,inotate
      print*,inotate

      print*,'Use IBM PC graphics characters (0=no):'
      iGraphics=0
      read*,iGraphics
      print*,iGraphics

      call diagram(filnam,filnam2,LCol,inotate,iGraphics)
      end
 c------------------------------------------------------------ ------------
      subroutine diagram(filnam,filnam2,LCol,inotate,
     &   iGraphics)
c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
      character*80 filnam,filnam2
      character*360 a,b,bsave
      character*5 form
      character*8 fm
      character*1 c
      logical fout
      logical find
      external find
      common icol
```

```
c Type of block
      character*16 BlockType(1000)

c Symbols which will mark block actions:
      character*1 BlockBegin    (2) /'+','+'/  ! Start of block
      character*1 BlockEnd      (2) /'+','+'/  ! End of block
      character*1 BlockElse     (2) /'+','+'/  ! Else construct
      character*1 BlockContinue (2) /'|','|'/  ! Block continues w/o
change
      character*1 BlockHoriz    (2) /'-','-'/  ! Horizontal to start
of line
c Same, but allows horizontal line to continue through:
      character*1 BlockBeginH   (2) /'+','+'/  ! Start of block
      character*1 BlockEndH     (2) /'+','+'/  ! End of block
      character*1 BlockElseH    (2) /'+','+'/  ! Else construct

      if(iGraphics.ne.0)then
        iGraphics=1

        BlockBegin   (1)=char(218)          ! (1)=normal
        BlockEnd     (1)=char(192)
        BlockElse    (1)=char(195)
        BlockContinue(1)=char(179)
        BlockHoriz   (1)=char(196)
        BlockBeginH  (1)=char(194)
        BlockEndH    (1)=char(193)
        BlockElseH   (1)=char(197)

        BlockBegin   (2)=char(214)          ! (2)=DO/FOR loops
(doubled)
        BlockEnd     (2)=char(211)          ! (not yet used)
        BlockEnd     (2)=char(211)
        BlockElse    (2)=char(199)
        BlockContinue(2)=char(186)
        BlockHoriz   (2)=char(196)
        BlockBeginH  (2)=char(209)
        BlockEndH    (2)=char(208)
        BlockElseH   (2)=char(215)
      endif

      open(1,file=filnam,status='old')
      fout=filnam2.gt.' '
      if(fout)open(2,file=filnam2,status='unknown')
                                  ! ASCII 12 is a form
feed
      if(fout)write(2,*)char(12),
     &   '==============--',filnam(1:LenA(filnam)),'--=============='
```

```
      if(fout)     write(2,'(11x,a50,a49,/)')  ! Write column header
     &   '...,....1....,....2....,....3....,....4....,....5',
     &   '...,....6....,....7....,....8....,....9....,....'
      if(.not.fout)write(*,'(11x,a50,a49,/)' ',
     &   '...,....1....,....2....,....3....,....4....,....5',
     &   '...,....6....,....7....,....8....,....9....,....'

      i3=0                            ! # nest levels after
                                     !  current line
      ltgt=0                          ! < > nesting
      InHtml=0                           ! 0 Not in <html> block
                                     ! 1 In <html> block
                                     ! 2 <html> block has
already occurred
      inhead=0                           ! same for <head>
      intitle=0                         ! same for <title>
      inbody=0                           ! same for <body>
      infont=0                          ! same for <font>
      inspan=0                           ! same for <span>
      inh1=0                            ! same for <h1>,<h2...>
      ina=0                             ! same for <a ...>
 inb=0     ! same for <b>
 inp=0     ! same for <p>
      nline=0
      icomment=0                            ! not inside comment
      iunit=1
10      a=' '
      read(iunit,'(a360)',end=99)a
      nline=nline+1
      fm=' '
      write(fm,'(i5)')nline
      form=fm

      if(a(1:1).eq.char(12))then
        if(fout)write(2,'(a1,:)')char(12)
        if(.not.fout)print*,'------------FORM FEED------------'
        b=a(2:360)
        a=b
      endif

      b=' '                           ! Turn tabs to spaces
      j=1
      do i=1,LenA(a)
        if(a(i:i).eq.char(9))then
          j=(j-1)/8*8+8+1
     ! Make sure is good ASCII char
        elseif(j.le.360.and.a(i:i).ge.'
'.and.a(i:i).lt.char(128))then
```

```
          b(j:j)=a(i:i)
          j=j+1
        endif
      enddo

      a=b
      bsave=b
      b=' '
      i1=i3                          ! # nest levels before
                                   !  current line
      i4=0                           ! not 0 to flag start
or end
                                   !  of block
      iquote=0                        ! no ' yet
      idquote=0                       ! no " yet
      icomment2=0                        ! anything outside
comment?
      icomment3=icomment                    ! no comment occurred?
      i=1
      j=1
      dowhile(i.le.360)                   ! handle upper case
        c=a(i:i)
        if(c.ge.'A'.and.c.le.'Z')c=char(ichar(c)+32)
        if(c.eq.''''.and.idquote.eq.0.and.icomment.eq.0
     &    .and.ltgt.ne.0)then
          iquote=1-iquote
          if(i.gt.1)then
                                   ! char(92) is \
            if(iquote.eq.0.and.a(i-1:i-1).eq.char(92))
     &        iquote=1-iquote
          endif
        endif
        if(c.eq.'"' .and.iquote .eq.0.and.icomment.eq.0
     &    .and.lgt.ne.0)then
          idquote=1-idquote
          if(i.gt.1)then
            if(idquote.eq.0.and.a(i-1:i-1).eq.char(92))
     &        idquote=1-idquote
          endif
        endif
        if(c.eq.'<'.and.i.lt.359.and.iquote.eq.0.and.idquote.eq.0)
! <!-  ?
     &    then
          if(a(i+1:i+1).eq.'!'.and.a(i+2:i+2).eq.'-')then
            if(icomment.ne.0)then
              if(fout)print*,a
              PRINT*,'***WARNING--nested comment  LINE',form
  if(fout)print*,a
```

```
            if(fout)WRITE(2,*)'***WARNING--nested comment  LINE',
     &   form
              print*,char(7)
            endif
            icomment=1
            icomment3=1
            c=' '
            i=i+2
          endif
        endif
        if(c.eq.'-'.and.i.lt.360.and.iquote.eq.0.and.idquote.eq.0)
! -> ?
     &     then
          if(a(i+1:i+1).eq.'>')then
            if(icomment.eq.0)then
              PRINT*,'***WARNING---> without <!- clause  LINE',form
              if(fout)print*,a
              if(fout)write(2,*)
     &          '***WARNING---> without <!- clause  LINE',form
              print*,char(7)
            endif
            icomment=0
            c=' '
            i=i+2
          endif
        endif
        if(icomment.ne.0)c=' '
        if(c.ne.' ')icomment2=1

        if(c.eq.'<')then
          if(ltgt.ne.0)then
            print*,'***ERROR: nested <  LINE ',form
            if(fout)WRITE(2,*)'***ERROR: nested <  LINE ',form
          endif
          ltgt=1
        elseif(c.eq.'>')then
          if(ltgt.eq.0)then
            PRINT*,'***ERROR-- > without <  LINE ',
     &        form
            if(fout)
     &        WRITE(2,*)'***ERROR-- > without <  LINE ',form
            if(fout)print*,a
            print*,char(7)
            ltgt=max(ltgt,0)
          endif
          ltgt=0
        endif
        if(j.le.360) b(j:j)=c
```

```
        if(j.gt.1)then                  ! (kill multiple
spaces)
          if(c.eq.' '.and.b(j-1:j-1).eq.' ')j=j-1
        endif
        j=j+1
        i=i+1
      enddo
      if(iQuote.ne.0.or.idquote.ne.0)then
        PRINT*,'***ERROR--unclosed quote  LINE ',form
        if(fout)WRITE(2,*)'***ERROR--unclosed quote  LINE ',form
        if(fout)print*,a
        print*,char(7)
      endif

      DO I=1,360
15      if(find(b(i:360),'<html>',1))then
        if(InHtml.eq.1)then
          PRINT*,'***ERROR--nested <html>  LINE ',form
          if(fout)WRITE(2,*)'***ERROR--nested <html>  LINE ',form
          if(fout)print*,a
          print*,char(7)
  else
          i3=i3+1
          i4=1
          BlockType(i3)='<html>'
        endif
        if(InHtml.eq.2)then
          PRINT*,'***ERROR--<html> has already occurred  LINE ',form
          if(fout)
     &      WRITE(2,*)
     &       '***ERROR--<html>  has already occurred LINE ',form
          if(fout)print*,a
          print*,char(7)
        endif
        InHtml=1
      elseif(find(b(i:360),'</html>',1))then
        if(InHtml.ne.1)then
          PRINT*,'***ERROR--</html> without <html>  LINE ',form
          if(fout)
     &      WRITE(2,*)'***ERROR--</html> without <html>  LINE ',form
          if(fout)print*,a
          print*,char(7)
        else
          i3=i3-1
          i4=1
          dowhile(i3.gt.0.and.BlockType(i3+1).ne.'<html>')
            PRINT*,'***WARNING--unclosed ',BlockType(i3+1),
     &        '  LINE ',form
```

```
          if(fout)print*,a
          if(fout)
     &      WRITE(2,*)'***WARNING--unclosed ',BlockType(i3+1),
     &       '  LINE ',form
          print*,char(7)
        i3=i3-1
        enddo
      endif
      InHtml=2
    elseif(find(b(i:360),'<head>',1))then
      if(InHtml.ne.1)then
        PRINT*,'***ERROR--<head> not inside <html>  LINE ',form
        if(fout)WRITE(2,*)
     &    '***ERROR--<head> not inside <html>  LINE ',form
        if(fout)print*,a
        print*,char(7)
      endif
      if(inhead.eq.1)then
        PRINT*,'***ERROR--nested <head>  LINE ',form
        if(fout)WRITE(2,*)'***ERROR--nested <head>  LINE ',form
        if(fout)print*,a
        print*,char(7)
else
        i3=i3+1
        i4=1
        BlockType(i3)='<head>'
      endif
      if(inhead.eq.2)then
        PRINT*,'***ERROR--<head> has already occurred  LINE ',form
        if(fout)WRITE(2,*)
     &    '***ERROR--<head> has already occurred  LINE ',form
        if(fout)print*,a
        print*,char(7)
      endif
      inhead=1
    elseif(find(b(i:360),'</head>',1))then
      if(inhead.ne.1)then
        PRINT*,'***ERROR--</head> without <head>  LINE ',form
        if(fout)
     &    WRITE(2,*)'***ERROR--</head> without <head>  LINE ',form
        if(fout)print*,a
        print*,char(7)
      else
        i3=i3-1
        i4=1
        dowhile(i3.gt.0.and.BlockType(i3+1).ne.'<head>')
          PRINT*,'***WARNING--unclosed ',BlockType(i3+1),
     &      '  LINE ',form
```

```
            if(fout)print*,a
            if(fout)
     &        WRITE(2,*)'***WARNING--unclosed ',BlockType(i3+1),
     &       '  LINE ',form
            print*,char(7)
          i3=i3-1
          enddo
        endif
        inhead=2
      elseif(find(b(i:360),'<title>',1))then
        if(inhead.ne.1)then
          PRINT*,'***ERROR--<title> not inside <head>  LINE ',form
          if(fout)WRITE(2,*)
     &      '***ERROR--<title> not inside <head>  LINE ',form
          if(fout)print*,a
          print*,char(7)
        endif
        if(intitle.eq.1)then
          PRINT*,'***ERROR--nested <title>  LINE ',form
          if(fout)WRITE(2,*)'***ERROR--nested <title>  LINE ',form
          if(fout)print*,a
          print*,char(7)
   else
          i3=i3+1
          i4=1
          BlockType(i3)='<title>'
        endif
        if(intitle.eq.2)then
           PRINT*,'***ERROR--<title> has already occurred  LINE
',form
          if(fout)WRITE(2,*)
     &      '***ERROR--<title>  has already occurred  LINE ',form
          if(fout)print*,a
          print*,char(7)
        endif
        intitle=1
      elseif(find(b(i:360),'</title>',1))then
        if(intitle.ne.1)then
           PRINT*,'***ERROR--</title> without <title>  LINE ',form
          if(fout)
     &      WRITE(2,*)'***ERROR--</title> without <title>  LINE ',
     &       form
          if(fout)print*,a
          print*,char(7)
        else
          i3=i3-1
          i4=1
          dowhile(i3.gt.0.and.BlockType(i3+1).ne.'<title>')
```

```
        PRINT*,'***WARNING--unclosed ',BlockType(i3+1),
     &       '  LINE ',form
        if(fout)
     &    WRITE(2,*)'***WARNING--unclosed ',BlockType(i3+1),
     &     '  LINE ',form
        if(fout)print*,a
        print*,char(7)
      i3=i3-1
       enddo
      endif
      intitle=2
     elseif(find(b(i:360),'<body>',1))then
      if(InHtml.ne.1)then
       PRINT*,'***ERROR--<body> not inside <html>  LINE ',form
       if(fout)WRITE(2,*)
     &    '***ERROR--<body> not inside <html>  LINE ',form
       if(fout)print*,a
       print*,char(7)
      endif
      if(inhead.eq.1)then
       PRINT*,'***ERROR--<body> inside <head>  LINE ',form
       if(fout)WRITE(2,*)
     &    '***ERROR--<body> inside <head>  LINE ',form
       if(fout)print*,a
       print*,char(7)
      endif
      if(inhead.eq.0)then
       PRINT*,'***ERROR--<body> before <head>  LINE ',form
       if(fout)WRITE(2,*)
     &    '***ERROR--<body> before <head>  LINE ',form
       if(fout)print*,a
       print*,char(7)
      endif
      if(inbody.eq.1)then
       PRINT*,'***ERROR--nested <body>  LINE ',form
       if(fout)WRITE(2,*)'***ERROR--nested <body>  LINE ',form
       if(fout)print*,a
       print*,char(7)
    else
       i3=i3+1
       i4=1
       BlockType(i3)='<body>'
      endif
      if(inbody.eq.2)then
        PRINT*,'***ERROR--<body> has already occurred  LINE ',form
       if(fout)
     &    WRITE(2,*)'***ERROR--<body> has already occurred  LINE ',
     &    form
```

```
        if(fout)print*,a
        print*,char(7)
      endif
      inbody=1
    elseif(find(b(i:360),'</body>',1))then
      if(inbody.ne.1)then
        PRINT*,'***ERROR--</body> without <body>  LINE ',form
      if(fout)
     &   WRITE(2,*)'***ERROR--</body> without <body>  LINE ',form
      if(fout)print*,a
      print*,char(7)
      else
      i3=i3-1
      i4=1
      dowhile(i3.gt.0.and.BlockType(i3+1).ne.'<body>')
        PRINT*,'***WARNING--unclosed ',BlockType(i3+1),
     &    '  LINE ',form
        if(fout)print*,a
        if(fout)
     &    WRITE(2,*)'***WARNING--unclosed ',BlockType(i3+1),
     &   '  LINE ',form
        print*,char(7)
  i3=i3-1
      enddo
      endif
      inbody=2
    elseif(find(b(i:360),'<font ',1))then
      if(inbody.ne.1)then
      PRINT*,'***ERROR--<font> not inside <body>  LINE ',form
      if(fout)WRITE(2,*)
     &   '***ERROR--<font> not inside <body>  LINE ',form
      if(fout)print*,a
      print*,char(7)
      endif
      if(infont.eq.1)then
      PRINT*,'***ERROR--nested <font>  LINE ',form
      if(fout)WRITE(2,*)'***ERROR--nested <font>  LINE ',form
      if(fout)print*,a
      print*,char(7)
else
      i3=i3+1
      i4=1
      BlockType(i3)='<font>'
      endif
      infont=1
    elseif(find(b(i:360),'</font>',1))then
      if(infont.ne.1)then
        PRINT*,'***ERROR--</font> without <font>  LINE ',form
```

```
        if(fout)
     &    WRITE(2,*)'***ERROR--</font> without <font>  LINE ',
     &     form
        if(fout)print*,a
        print*,char(7)
      else
        i3=i3-1
        i4=1
        dowhile(i3.gt.0.and.BlockType(i3+1).ne.'<font>')
          PRINT*,'***WARNING--unclosed ',BlockType(i3+1),
     &      '  LINE ',form
          if(fout)print*,a
          if(fout)
     &      WRITE(2,*)'***WARNING--unclosed ',BlockType(i3+1),
     &     '  LINE ',form
          print*,char(7)
    i3=i3-1
        enddo
      endif
      infont=0
    elseif(find(b(i:360),'<span ',1))then
      if(inbody.ne.1)then
        PRINT*,'***ERROR--<span> not inside <body>  LINE ',form
        if(fout)WRITE(2,*)
     &    '***ERROR--<span> not inside <body>  LINE ',form
        if(fout)print*,a
        print*,char(7)
      endif
      if(Inspan.eq.1)then
        PRINT*,'***ERROR--nested <span>  LINE ',form
        if(fout)WRITE(2,*)'***ERROR--nested <span> LINE ',form
        if(fout)print*,a
        print*,char(7)
else
        i3=i3+1
        i4=1
        BlockType(i3)='<span>'
      endif
      inspan=1
    elseif(find(b(i:360),'</span>',1))then
      if(inspan.ne.1)then
        PRINT*,'***ERROR--</span> without <span>  LINE ',form
      if(fout)
     &    WRITE(2,*)'***ERROR--</span> without <span> LINE ',
     &     form
      if(fout)print*,a
      print*,char(7)
      else
```

```
          i3=i3-1
          i4=1
          dowhile(i3.gt.0.and.BlockType(i3+1).ne.'<span>')
            PRINT*,'***WARNING--unclosed ',BlockType(i3+1),
     &        '  LINE ',form
            if(fout)
     &        WRITE(2,*)'***WARNING--unclosed ',BlockType(i3+1),
     &        '  LINE ',form
            if(fout)print*,a
            print*,char(7)
       i3=i3-1
          enddo
        endif
        inspan=0
      elseif(b(i:i+2).ge.'<h1'.and.b(i:i+2).le.'<h9')then
        if(inbody.ne.1)then
          PRINT*,'***ERROR--',b(i:i+4),' not inside <body>
LINE',form
          if(fout)WRITE(2,*)
     &      '***ERROR--',b(i:i+4),' not inside <body> LINE ',form
          if(fout)print*,a
          print*,char(7)
        endif
        if(inh1.ne.0)then
          PRINT*,'***ERROR--nested <h#>  LINE',form
          if(fout)WRITE(2,*)'***ERROR--nested <h#>  LINE ',form
          if(fout)print*,a
          print*,char(7)
   else
          i3=i3+1
          i4=1
          BlockType(i3)='<h#>'
        endif
        inh1=ichar(b(i+2:i+2))
      endif
      if(b(i:i+4).ge.'</h1'.and.b(i:i+4).le.'</h9')then
   if(ichar(b(i+3:i+3)).ne.inh1)then
          PRINT*,'***Incorrect <h#> level***>'
          if(fout)
     &     WRITE(2,*)'***Incorrect <h#> level  LINE ',
     &     form
   endif
        if(inh1.eq.0)then
          PRINT*,'***ERROR--</h#> without <h#>  LINE',form
          if(fout)
     &      WRITE(2,*)'***ERROR--</h#> without <h#>  LINE ',
     &       form
          if(fout)print*,a
```

```
        print*,char(7)
      else
        i3=i3-1
        i4=1
        dowhile(i3.gt.0.and.BlockType(i3+1).ne.'<h#>')
          PRINT*,'***WARNING--unclosed ',BlockType(i3+1),
     &         ' LINE ',form
          if(fout)
     &      WRITE(2,*)'***WARNING--unclosed ',BlockType(i3+1),
     &      ' LINE ',form
          if(fout)print*,a
          print*,char(7)
    i3=i3-1
        enddo
      endif
      inh1=0
    elseif(find(b(i:360),'<a ',1))then
      if(inbody.ne.1)then
        PRINT*,'***ERROR--<a> not inside <body>  LINE',form
        if(fout)WRITE(2,*)
     &    '***ERROR--<a> not inside <body> LINE ',form
        if(fout)print*,a
        print*,char(7)
      endif
      if(ina.eq.1)then
        PRINT*,'***ERROR--nested <a>  LINE',form
        if(fout)WRITE(2,*)'***ERROR--nested <a>  LINE ',form
        if(fout)print*,a
        print*,char(7)
  else
        i3=i3+1
        i4=1
        BlockType(i3)='<a>'
      endif
      ina=1
    elseif(find(b(i:360),'</a>',1))then
      if(ina.ne.1)then
        PRINT*,'***ERROR--</a> without <a>  LINE ',form
        if(fout)
     &    WRITE(2,*)'***ERROR--</a> without <a>  LINE ',
     &     form
        if(fout)print*,a
        print*,char(7)
      else
        i3=i3-1
        i4=1
        dowhile(i3.gt.0.and.BlockType(i3+1).ne.'<a>')
          PRINT*,'***WARNING--unclosed ',BlockType(i3+1),
```

```
&         '  LINE ',form
          if(fout)
&         WRITE(2,*)'***WARNING--unclosed ',BlockType(i3+1),
&         '  LINE ',form
          if(fout)print*,a
          print*,char(7)
       i3=i3-1
         enddo
       endif
       ina=0
     elseif(find(b(i:360),'<b>',1))then
       if(inbody.ne.1)then
         PRINT*,'***ERROR--<b> not inside <body>  LINE',form
         if(fout)WRITE(2,*)
&        '***ERROR--<b> not inside <body>  LINE ',form
         if(fout)print*,a
         print*,char(7)
       endif
       if(inb.eq.1)then
         PRINT*,'***ERROR--nested <b>  LINE',form
         if(fout)WRITE(2,*)'***ERROR--nested <b>  LINE ',form
         if(fout)print*,a
         print*,char(7)
else
         i3=i3+1
         i4=1
         BlockType(i3)='<b>'
       endif
       inb=1
     elseif(find(b(i:360),'</b>',1))then
       if(inb.ne.1)then
          PRINT*,'***ERROR--</b> without <b>  LINE',form
         if(fout)
&        WRITE(2,*)'***ERROR--</b> without <b>  LINE ',
&         form
         if(fout)print*,a
         print*,char(7)
       else
         i3=i3-1
         i4=1
         dowhile(i3.gt.0.and.BlockType(i3+1).ne.'<b>')
           PRINT*,'***WARNING--unclosed ',BlockType(i3+1),
&          '  LINE ',form
           if(fout)
&          WRITE(2,*)'***WARNING--unclosed ',BlockType(i3+1),
&          '  LINE ',form
           if(fout)print*,a
           print*,char(7)
```

```
      i3=i3-1
         enddo
        endif
        inb=0
       elseif(find(b(i:360),'<p>',1))then
        if(inbody.ne.1)then
         PRINT*,'***ERROR--<p> not inside <body>  LINE',form
         if(fout)WRITE(2,*)
     &     '***ERROR--<p> not inside <body>  LINE ',form
         if(fout)print*,a
         print*,char(7)
        endif
        if(inp.eq.1)then
         PRINT*,'***WARNING--prior <p> not closed',form
         if(fout)WRITE(2,*)
     &     '***WARNING--prior <p> not closed LINE ',form
         if(fout)print*,a
         print*,char(7)
        else
         i3=i3+1
      endif
        i4=1
        BlockType(i3)='<p>'
        inp=1
       elseif(find(b(i:360),'</p>',1))then
        if(inp.ne.1)then
          PRINT*,'***ERROR--</p> without <p>  LINE',form
         if(fout)
     &     WRITE(2,*)'***ERROR--</p> without <p>  LINE ',
     &      form
         if(fout)print*,a
         print*,char(7)
        else
         i3=i3-1
         i4=1
         dowhile(i3.gt.0.and.BlockType(i3+1).ne.'<p>')
          PRINT*,'***WARNING--unclosed ',BlockType(i3+1),
     &      '  LINE ',form
          if(fout)
     &      WRITE(2,*)'***WARNING--unclosed ',BlockType(i3+1),
     &      '  LINE ',form
          if(fout)print*,a
          print*,char(7)
      i3=i3-1
         enddo
        endif
        inp=0
c### ADD MORE SEARCH ITEMS HERE
```

```
        endif
        ENDDO

        igoto=0                         ! no goto on line
c       if(find(a,'go to',64+512).or.find(a,'goto',64+512)
c    &   .or.find(a,'return',32+512)
c    &   .or.find(a,'break',32+512).or.find(a,'continue',32+512)
c    &   .or.find(a,'exit',32+512))igoto=1

c       if(find(b,'case',32+512).or.
c    &   find(b,'default ',512).or.find(b,'default:',512))i4=max(1,i4)

20      b=bsave
        a=' '
        if(i1  .lt.0.or.i3  .lt.0.or.i4  .lt.0)then
          PRINT*,'***ERROR--INVALID DIAGRAMMING INDEX  LINE ',form
          if(fout)WRITE(2,*)
     &    '***ERROR--INVALID DIAGRAMMING INDEX  LINE',form
          if(fout)print*,b
          print*,char(7)
          i1=max(i1,0)
          i3=max(i3,0)
          i4=max(i4,0)
        endif

        i2=max(i1,i3)                   ! # of nests on current
line
        i4=max(i4,iabs(i3-i1))              ! not 0, to flag start
or
                                    !  end of block
        iBlock=1                        ! For the present
version.

        a=' '                           ! Leave space for
diagram
        a(12:360)=b                     !  (must match column
header)

        LastUse=1                       ! Last usable diagram
col
        dowhile(LastUse.lt.360.and.a(LastUse:LastUse).eq.' ')
          LastUse=LastUse+1
        enddo
        LastUse=LastUse-2

        if(igoto.ne.0)a(1:1)='*'            ! Place * next to jumps
        if(icomment2.eq.0.and.icomment3.ne.0..and.inotate.ne.0)
     &   a(1:1)='='
```

```
      if(i2.gt.0)then                   ! Same for
non-pre-processor
        do i=1,min(i2,LastUse)
          a(i:i)=BlockContinue(iBlock)
        enddo
      endif

      if(i4.ne.0)then
        do i=i2+1,LastUse
          a(i:i)=BlockHoriz(iBlock)
        enddo
      endif

      do i=0,i4-1

        c=            BlockElse(iBlock)
        if(i1+i.lt.i3)c=BlockBegin(iBlock)
        if(i1+i.gt.i3)c=BlockEnd  (iBlock)
        j=max(1,min(LastUse,i2-i))
        a(j:j)=c
        if(a(j+1:j+1).eq.BlockElse  (iBlock))
     &     a(j+1:j+1)  = BlockElseH (iBlock)
        if(a(j+1:j+1).eq.BlockBegin (iBlock))
     &     a(j+1:j+1)  = BlockBeginH(iBlock)
        if(a(j+1:j+1).eq.BlockEnd   (iBlock))
     &     a(j+1:j+1)  = BlockEndH  (iBlock)
      enddo

      if(LCol.gt.0.and.a(max(1,LCol+11):360).eq.' ')then      ! line
#
        if(form(1:1).eq.' ')form(1:1)=BlockContinue(iBlock)
        a(LCol+11:360)=form
      endif

      n=LenA(a)                     ! Output diagrammed
line
      if(fout)    write(2,'(80a1,80a1,80a1,80a1,80a1)')
     &  (a(i:i),i=1,n)
      if(.not.fout)write(*,'(1x,80a1,80a1,80a1,80a1)')
     &  (a(i:i),i=1,n)

      i1=i3
      goto 10
99    if(iunit.eq.3)then
        iunit=1
        i1=i1-1
```

```
        close(3)
        goto 10
      endif
      if(i3.gt.0)then
        PRINT*,'***WARNING--SOME NEST LEVELS LEFT HANGING AT END***'
        if(fout)write(2,*)
     &   '***WARNING--SOME NEST LEVELS LEFT HANGING AT END***'
        print*,char(7)
      endif
      if(inhead.eq.0)then
        PRINT*,'***ERROR--<head> never occurred***'
        if(fout)WRITE(2,*)'***ERROR--<head> never occurred!***'
        print*,char(7)
      endif
      if(intitle.eq.0)then
        PRINT*,'***ERROR--<title> never occurred***'
        if(fout)WRITE(2,*)'***ERROR--<title> never occurred!***'
        print*,char(7)
      endif
      if(inbody.eq.0)then
        PRINT*,'***ERROR--<body> never occurred***'
        if(fout)WRITE(2,*)'***ERROR--<body> never occurred!***'
        print*,char(7)
      endif
      end
c------------------------------------------------------------ ------------
      logical function find(a,b,icond)       ! find b in a, subject
to
                                     !  conditions:
                                     ! icond=sum of the
following:
                                     ! 1:  Must be first
character
                                     ! 2:  Must be first
non-blank
                                     ! 32: Next character
not alphanumeric
                                     ! 64: Next character
not alphabetic
                                     ! 512  Prior character,
if present,
                                     !      must be blank or
) or }
                                     !      or { or ;
c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
c Revision date: 8/25/96.
      character*(*) a,b
      character*1   c,cNext
```

```
      common icol
      logical result

      ii=len(a)
      jj=len(b)
      result=.false.
      loopend=ii-jj+1
      if(iand(icond,1).ne.0)loopend=min(loopend,1)
      do i=1,loopend
        if(a(i:i+jj-1).eq.b)then
          icol1=i                    ! icol1=column of item
found
          icol =i+jj                 ! icol =column after
item
                                     !  found
          c=' '
          cNext=' '
          if(icol1.gt.1)c=a(icol1-1:icol1-1)
          if(icol .le.ii)cNext=a(icol:icol)

          result=.true.

          if(result.and.iand(icond,2).ne.0.and.icol1.gt.1)then
            result=a(1:icol1-1).eq.' '
          endif

          if(result.and.iand(icond,32).ne.0)
     &      result=(cNext.lt.'0'.or.cNext.gt.'9').and.
     &             (cNext.lt.'a'.or.cNext.gt.'z')

          if(result.and.iand(icond,64).ne.0)
     &      result=(cNext.lt.'a'.or.cNext.gt.'z')

          if(result.and.iand(icond,512).ne.0)result=c.eq.' '
     &      .or.c.eq.';'.or.c.eq.')'.or.c.eq.'{'.or.c.eq.'}'

          find=result
          if(result)return
        endif
      enddo
      find=result
      return
      end
 c------------------------------------------------------------ ------------
      function LenA(a)                 ! Length of string, at
least 1
c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
c Revision date: 8/25/96.
```

```
        character*(*) a
        n=len(a)
        dowhile(n.gt.1.and.a(n:n).eq.' ')
          n=n-1
        enddo
        LenA=n
        end
------------------END diagramh.f--------------------
-----------------BEGIN diagramh.sh-------------------
#!/bin/csh
#                    ---diagramh.sh---
#Unix csh procedure to diagram an HTML language program.

#On some unix systems $1 should be replaced by %1.

# by Mitchell R Grunes.
# for his own use, in his own time

#I assume that the executable and this procedure are in the search
path,
#  and that this procedure has execute permission.

#Syntax:
#    diagramh.sh
#to be prompted for input parameters.

#Alternate Syntax:
#    diagramh.sh filename(s)
#to append diagram of file(s) into diagram.out

  if  (${?noclobber}) then
    unset noclobber
    set   noclobbersave
  endif

  if $1a == a  then
    diagramh
    goto quit
  endif

loop:
  echo ========================-- $1 --========================
  #Prompt answers: input from $1, output to diagram2.sc (for now),
  # Don't place numbers in column 67, don't use IBM PC graphics,
  # warn if 'end' ends if, for, etc.

  echo $1         >  diagram.sc
  echo diagram2.sc >> diagram.sc
```

```
  echo 0          >> diagram.sc
  echo 1          >> diagram.sc
  echo 0          >> diagram.sc
  echo 0          >> diagram.sc

  diagramh < diagram.sc
  cat diagram2.sc >> diagram.out
  rm -f diagram.sc
  rm -f diagram2.sc
  shift
  if ! ($1a == a) then
    goto loop
  endif
quit:
  echo Note--This does not delete diagram.out before appending to it.
  if  (${?noclobbersave}) then
    set   noclobber
    unset noclobbersave
  endif
------------------END diagramh.sh--------------------
-----------------BEGIN diagramh.bat-------------------
rem                    ---diagramh.bat---
rem MS-DOS procedure to diagram an HTML language program.

rem  by Mitchell R Grunes.

rem I assume that the executable is in directory c:\grunes on
rem  your PC.

rem Syntax:
rem     diagramh
rem to be prompted for input parameters.

rem Alternate Syntax:
rem     diagramh filename(s)
rem to append diagram of file(s) into diagram.out

if %1a == a  c:\grunes\diagramh
if %1a == a  goto quit

echo off
:loop
  echo ========================-- %1 --========================
  rem Prompt answers: input from %1, output to diagram2.sc (for now),
  rem  place numbers in column 67, use IBM PC graphics.

  echo %1         >  diagram.sc
  echo diagram2.sc >> diagram.sc
```

```
  echo 0          >> diagram.sc
  echo 1          >> diagram.sc
  echo 0          >> diagram.sc
  echo 0          >> diagram.sc

  c:\grunes\diagramh < diagram.sc
  type diagram2.sc >> diagram.out
  del diagram.sc
  del diagram2.sc
  shift
  if not %1a == a goto loop
:quit
  echo Note--This does not delete diagram.out before appending to it.
------------------END diagramh.bat--------------------
-----------------BEGIN diagramh.vax-------------------
$!                      ---diagramh.vax---
$!VAX VMS procedure to diagram an HTML language program
$!
$! by Mitchell R Grunes.
$!
$!I assume that the executable and this procedure are in the search
path,
$!  and that this procedure has execute permission.
$!
$!Syntax:
$!    @diagramh.vax
$!to be prompted for input parameters.
$!
$!Alternate Syntax:
$!    @diagramh.vax filename(s)
$!to append diagram of file(s) into diagram.out
$
$  if P1 .EQS. ""
$  then
$    define/user sys$input sys$command
$    run diagramh
$    goto quit
$  endif
$
$  write sys$output "========================-- "+P1+"
--========================"
$
$! Must pre-create diagram.out if does not exist
$  open/append/error=noSkip diagram.out diagram.out
$  goto Skip
$noSkip:
$  open/write diagram.out diagram.out
$Skip:
```

```
$  close diagram.out
$
$! Must pre-create diagram2.sc with same file attributes
$  open/write diagram2.sc diagram2.sc
$  close diagram2.sc
$
$  !Prompt answers: input from P1, output to diagram2.sc (for now),
$  ! don't place line numbers anywhere, don't use IBM PC graphics.
$
$  open/write diagram.sc diagram.sc
$  write diagram.sc "$Run diagramh"
$  write diagram.sc P1
$  write diagram.sc "diagram2.sc"
$  write diagram.sc "0"
$  write diagram.sc "1"
$  write diagram.sc "0"
$  write diagram.sc "0"
$  close diagram.sc
$  @diagram.sc
$  append diagram2.sc diagram.out
$  delete diagram.sc;*
$  delete diagram2.sc;*
$
$  if (P2 .NES. "") then @diagramh.vax 'P2' 'P3' 'P4' 'P5' 'P6' 'P7'
'P8'
$  write sys$output "Note--This does not delete diagram.out before
appending to it."
$quit:
------------------END diagramh.vax--------------------
-----------------BEGIN diagrami.f-------------------
c EXAMPLE OF OUTPUT (looks better if you choose IBM PC line graphics):

c     +--------- pro Sample,a,b,c                        |   1
c     |          a=indgen(15)^2                         |   2
c     |+-------- if a eq b then begin                   |   3
c     ||           print,'A equals B'                   |   4
c     ||           c=0                                  |   5
c     |+-------- else begin                             |   6
c     ||           print,'A does not equal B'           |   7
c     ||           c=1                                  |   8
c     |+-------- endif                                  |   9
c     +--------- end                                    |  10

c Diagrams IDL and PV-Wave begin(or case)-end constructs, functions
c  and procedures, places a * next to goto and return statements.

c Designed by mitch grunes, in his own time.
```

```
c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
c Revision date: 8/25/96.
c If you find it useful, or find a problem, please send me e-mail.

c ---------------------------------------------------
c This program was written in FORTRAN, for historic reasons.
c This was written in Fortran 77 (with common extensions) for
c  portability.  It should also compile under Fortran 90 and Fortran
95,
c  provided you tell the compiler it is in card format.
 c---------------------------------------------------------- ----------

c I hope this works for you, but bear in mind that nothing short of
c  a full-fledged language parser could really do the job.  Perhaps
c  worth about what you paid for it.    (-:

c Versions: To diagram Fortran:     diagramf.f
c                   IDL/PV-WAVE: diagrami.f
c                   C:           diagramc.f
c MS-DOS procedures to call above programs without asking so many
questions,
c  append output to file diagram.out:
c                   Fortran:     diagramf.bat (card format)
c                                diagram9.bat (free format)
c                   IDL/PV-WAVE: diagrami.bat
c                   C:           diagramc.bat
c Similar Unix csh procedures:
c                   Fortran:     diagramf.sh  (card format)
c                                diagram9.sh  (free format)
c                   IDL/PV-WAVE: diagrami.sh
c                   C:           diagramc.sh
c Similar Vax VMS DCL procedures:
c                   Fortran:     diagramf.vax (card format)
c                                diagram9.vax (free format)
c                   IDL/PV-WAVE: diagrami.vax
c                   C:           diagramc.vax

      program diagrami                    ! Diagrammer for IDL
and
                                          !  PV-WAVE
      character*80 filnam,filnam2

      print*,'IDL source filename?'
      read(*,'(a80)')filnam
      print*,filnam

      print*,'Output file (blank=screen)?'
      read(*,'(a80)')filnam2
```

```
        print*,filnam2

        print*,'Column in which to write line #''s ',
     &   '(67 for 80 col screen, 0 for none):'
        LCol=0
        read*,LCol
        print*,LCol

        print*,'Use IBM PC graphics characters (0=no):'
        iGraphics=0
        read*,iGraphics
        print*,iGraphics

        print*,'Should I warn if "end" ends if, for... (0=no):'
        iWarnEnd=1                          ! Drop warnings on 'end'
for 'endif...'
        read*,iWarnEnd
        print*,iWarnEnd

        call diagram(filnam,filnam2,LCol,iGraphics,iWarnEnd)
        end
 c------------------------------------------------------------ ------------
        subroutine diagram(filnam,filnam2,LCol,iGraphics,iWarnEnd)
c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
        character*80 filnam,filnam2
        character*160 a,b
        character*5 form
        character*8 fm
        character*1 c
        logical find
        external find
        common icol,icol1
        logical fout

c Symbols which will mark block actions:
        character*1 BlockBegin    (2) /'+','+'/  ! Start of block
        character*1 BlockEnd      (2) /'+','+'/  ! End of block
        character*1 BlockElse     (2) /'+','+'/  ! Else construct
        character*1 BlockContinue (2) /'|','|'/  ! Block continues w/o
change
        character*1 BlockHoriz    (2) /'-','-'/  ! Horizontal to start
of line
c Same, but allows horizontal line to continue through:
        character*1 BlockBeginH   (2) /'+','+'/  ! Start of block
        character*1 BlockEndH     (2) /'+','+'/  ! End of block
        character*1 BlockElseH    (2) /'+','+'/  ! Else construct

        if(iGraphics.ne.0)then
```

```
        iGraphics=1

        BlockBegin   (1)=char(218)          ! (1)=normal
        BlockEnd     (1)=char(192)
        BlockElse    (1)=char(195)
        BlockContinue(1)=char(179)
        BlockHoriz   (1)=char(196)
        BlockBeginH  (1)=char(194)
        BlockEndH    (1)=char(193)
        BlockElseH   (1)=char(197)

        BlockBegin   (2)=char(214)          ! (2)=DO/FOR loops
(doubled)
        BlockEnd     (2)=char(211)          ! (not yet used)
        BlockEnd     (2)=char(211)
        BlockElse    (2)=char(199)
        BlockContinue(2)=char(186)
        BlockHoriz   (2)=char(196)
        BlockBeginH  (2)=char(209)
        BlockEndH    (2)=char(208)
        BlockElseH   (2)=char(215)
      endif

      open(1,file=filnam,status='old')
      fout=filnam2.gt.' '
      if(fout)open(2,file=filnam2,status='unknown')

                                  ! ASCII 12 is a form
feed
      if(fout)write(2,*)char(12),
     &  '==============--',filnam(1:LenA(filnam)),'--=============='

      if(fout)    write(2,'(11x,a50,a49,/)')  ! Write column header
     &  '....,....1....,....2....,....3....,....4....,....5',
     &  '....,....6....,....7....,....8....,....9....,....'
      if(.not.fout)write(*,'(11x,a50,a49,/)' ',
     &  '....,....1....,....2....,....3....,....4....,....5',
     &  '....,....6....,....7....,....8....,....9....,....'

      i1=0                          ! # nest levels before
                                  !  current line
      i2=0                          ! # nest levels on
                                  !  current line
      i3=0                          ! # of nest levels
after
                                  !  current line
      i4=0                          ! not 0 to flag start
or end
```

```
                                  !  of block
      InSub=0                          ! Inside a subroutine
or
                                  !  function?
      nMain=0                          ! no mainline program
yet
      InCase=0                         ! not inside case
      iContinue=0                      ! not continued from
prior line
      nline=0
10     a=' '
      read(1,'(a160)',end=99)a
      nline=nline+1
      fm=' '
      write(fm,'(i5)')nline
      form=fm

      if(a(1:1).eq.char(12))then
        if(fout)write(2,'(a1,:)')char(12)
        if(.not.fout)print*,'------------FORM FEED------------'
        b=a(2:160)
        a=b
      endif

      b=' '                            ! Turn tabs to spaces
      j=1
      do i=1,LenA(a)
        if(a(i:i).eq.char(9))then
          j=(j-1)/8*8+8+1
    ! Make sure is good ASCII char
        elseif(j.le.160.and.a(i:i).ge.'
'.and.a(i:i).lt.char(128))then
          b(j:j)=a(i:i)
          j=j+1
        endif
      enddo
      i=1
      j=1
      a=' '                            ! Pre-processing
      iquote=0                         ! no ' yet
      idquote=0                        ! no " yet
      j=1
      do i=1,LenA(b)
        c=b(i:i)
        if(c.ge.'A'.and.c.le.'Z')c=char(ichar(c)+32)
        if(c.eq.';')goto 15              ! comment
        if(c.eq.'@'.and.i.eq.1)goto 15       ! other procedure
includes
```

```
          if(c.eq.''''.and.idquote.eq.0)then
            iquote=1-iquote
            c=' '
          endif
          if(c.eq.'"' .and.iquote .eq.0)idquote=1-idquote
          if(iquote.ne.0.or.idquote.ne.0)c=' '
          if(j.gt.1)then                    ! (kill multiple
spaces)
            if(c.eq.' '.and.a(j-1:j-1).eq.' ')j=j-1
          endif
          if(c.eq.':')then                  ! (put space after :)
            if(j.le.160) a(j:j)=':'
            j=j+1
            c=' '
          endif
          if(j.le.160) a(j:j)=c
          j=j+1
        enddo

15      i2=i1
        i3=i1
        i4=0
        igoto=0                           ! no goto on line

        if(a.ne.' '.and.InSub.eq.0..and..not.
     &   (find(a,'function ',2).or.find(a,'pro ',2)))then       !
mainline
          InSub=InSub+1
          nMain=nMain+1
          if(fout)print*,'Line ',form,' ',b(1:LenA(b))
          if(nMain.gt.1)then
            PRINT*,'***ERROR--TOO MANY MAINLINES***'
            if(fout)WRITE(2,*)'***ERROR--TOO MANY MAINLINES!***'
            if(fout)print*,b
            print*,char(7)
          endif
          i2=i2+1
          i3=i3+1
        endif

        if(find(a,'goto',8+32).or.find(a,'return',1+128))igoto=1

        if(find(a,'endif ',2).or.find(a,'endfor ',2)
     &  .or.find(a,'endelse ',2).or.find(a,'endwhile ',2)
     &  .or.find(a,'endcase ',2).or.find(a,'endrep ',2))then
          i3=i3-1
          if(find(a,'begin  ',1))i3=i3+1
          i4=max(i4,1)
```

```
        if(i3.lt.InCase)InCase=0
      elseif(find(a,'case ',1).or.find(a,'begin  ',1))then
        InCase=i1
        i2=i2+1
        i3=i3+1
        i4=max(i4,1)
        if(find(a,': begin  ',0))i4=max(i4,2)
        if(find(a,'end ',1))i3=i3-1
      elseif(find(a,'end ',2))then
        if(i3.gt.0.or.Insub.gt.0)then        ! Problem: IDL end may
          i3=i3-1                            !  actually be an
endif,
                                             !  endelse, etc.
          if(i3.eq.0.and.InSub.ne.0)InSub=0
          if(i3.ne.0.and.iWarnEnd.ne.0)then
            print*,'WARNING end at line ',form
            print*,' "end" ends non-program!***'
            if(fout)WRITE(2,*)'***WARNING--"end" ends
non-program!***'
            print*,char(7)
          endif
        endif
        if(i3.lt.InCase)InCase=0
      elseif(find(a,'function ',2).or.find(a,'pro ',2))then
        if(fout)print*,'Line ',form,' ',b(1:LenA(b))
        InSub=InSub+1
        i2=i2+1
        i3=i3+1
        if(InSub.ne.1.or.i3.ne.1)then
          PRINT*,'***ERROR--INVALID DIAGRAMMING INDEX line',form
          if(fout)
     &      WRITE(2,*)'***ERROR--INVALID DIAGRAMMING INDEX!***'
          if(fout)print*,b
          print*,char(7)
          i3=1
          InSub=1
        endif
      elseif((find(a,':  ',0).or.find(a,':',256)).and.
     &  InCase.ne.0)then                     ! simple case instances
        i4=max(i4,1)
      elseif((find(a,':',0).and.InCase.ne.0))then      !other case
instances
        ileft=0
        iright=0
        ileft2=0
        iright2=0
        do i=1,icol1
          if(a(i:i).eq.'(')ileft=ileft+1
```

```
          if(a(i:i).eq.')')iright=iright+1
          if(a(i:i).eq.'[')ileft2=ileft+1
          if(a(i:i).eq.']')iright2=iright+1
        enddo
         if(ileft.eq.iright.and.ileft2.eq.iright2.and.icontinue.eq.0)
     &    i4=max(i4,1)
       endif

       icontinue=0
       if(find(a,'$  ',0))icontinue=1

       a=' '

       if(i1.lt.0.or.i2.lt.0.or.i3.lt.0.or.i4.lt.0)then
         PRINT*,'***ERROR--INVALID DIAGRAMMING INDEX line',form
         if(fout)WRITE(2,*)'***ERROR--INVALID DIAGRAMMING INDEX!***'
         if(fout)print*,b
         print*,char(7)
         i1=max(i1,0)
         i2=max(i2,0)
         i3=max(i3,0)
         i4=max(i4,0)
       endif

       i2=max(i1,i3)                    ! # of nests on current
line
       i4=max(i4,iabs(i3-i1))             ! not 0, to flag start
or
                                  !  end of block

       iBlock=1                      ! For the present
version.

       a=' '                        ! Leave space for
diagram
       a(12:160)=b                    !  (must match column
header)

       LastUse=1                      ! Last usable diagram
col
       dowhile(LastUse.lt.160.and.a(LastUse:LastUse).eq.' ')
         LastUse=LastUse+1
       enddo
       LastUse=LastUse-2

       if(igoto.ne.0)a(1:1)='*'          ! Place * next to jumps

       if(i2.gt.0)then                  ! Draw one vertical
```

```
line per
        do i=2,min(i2+1,LastUse)             !  nest level.
          a(i:i)=BlockContinue(iBlock)
        enddo
      endif

      if(i4.ne.0)then                        ! Draw horizontal lines
inward
        do i=i2+2,LastUse                    !  from above.
          a(i:i)=BlockHoriz(iBlock)
        enddo
      endif

      do i=0,i4-1                            ! May need to replace
some
                                           !  vertical lines with
        c=            BlockElse(iBlock)    !       else  symbol
        if(i1+i.lt.i3)c=BlockBegin(iBlock)   !  or   begin symbol
        if(i1+i.gt.i3)c=BlockEnd   (iBlock)  !  or   end   symbol
        j=max(2,min(LastUse,i2+1-i))
        a(j:j)=c
        if(a(j+1:j+1).eq.BlockElse  (iBlock)) ! Continue horizontal
lines
     &     a(j+1:j+1)  = BlockElseH (iBlock)
        if(a(j+1:j+1).eq.BlockBegin (iBlock))
     &     a(j+1:j+1)  = BlockBeginH(iBlock)
        if(a(j+1:j+1).eq.BlockEnd   (iBlock))
     &     a(j+1:j+1)  = BlockEndH  (iBlock)
      enddo

      if(LCol.gt.0.and.a(max(1,LCol+11):160).eq.' ')then       ! line
#
        if(form(1:1).eq.' ')form(1:1)=BlockContinue(iBlock)
        a(LCol+11:160)=form
      endif

      n=LenA(a)                              ! Output diagrammed
line
      if(fout)     write(2,'(80a1,80a1)')(a(i:i),i=1,n)
      if(.not.fout)write(*,'(1x,80a1,80a1)')(a(i:i),i=1,n)

      i1=i3
      goto 10
99    if(i3.gt.0.or.InSub.ne.0)then
        PRINT*,'***WARNING--SOME NEST LEVELS LEFT HANGING AT END***'
        if(fout)print*,b
        print*,char(7)
      endif
```

```
      end
 c------------------------------------------------------------ ------------
      logical function find(a,b,icond)       ! find b in a, subject
to
                                    !  conditions:
                                    ! icond=sum of the
following:
                                    ! 1:  Prior, if exists,
must
                                    !     be blank
                                    ! 2:  Must be first
non-blank
                                    ! 4:  Prior character,
if
                                    !     present, must not
be
                                    !     alphanumeric.
                                    ! 8:  Prior character,
if
                                    !     present, must be
blank
                                    !     or )
                                    ! 16: Prior character,
if
                                    !     present, must be
blank
                                    !     or ,
                                    ! 32: Next character
not
                                    !     alphanumeric
                                    ! 64: Next character
not
                                    !     alphabetic
                                    ! 128:Next character
must be
                                    !     blank or (
                                    ! 256:1st non-blank,
possibly
                                    !     except for
numeric
                                    !     labels
c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
c Revision date: 8/25/96.
      character*(*) a,b
      character*1   c,cNext,c2
      common icol,icol1
      logical result
```

```
        ii=len(a)
        jj=len(b)
        result=.false.
        do i=1,ii-jj+1
          if(a(i:i+jj-1).eq.b)then
            icol1=i                     ! icol1=column of item
found
            icol =i+jj                  ! icol =column after
item
                                      !  found
            c=' '
            cNext=' '
            if(icol1.gt.1)c=a(icol1-1:icol1-1)
            if(icol .le.ii)cNext=a(icol:icol)

            result=.true.
            if(result.and.iand(icond,1).ne.0.and.icol1.gt.1)then
              result=c.eq.' '
            endif

            if(result.and.iand(icond,2).ne.0.and.icol1.gt.1)then
              result=a(1:icol1-1).eq.' '
            endif

            if(result.and.iand(icond,4).ne.0)
     &        result=(c.lt.'0'.or.c.gt.'9').and.(c.lt.'a'.or.c.gt.'z')

            if(result.and.iand(icond,8).ne.0)result=c.eq.'
'.or.c.eq.')'

            if(result.and.iand(icond,16).ne.0)result=
     &        c.eq.' '.or.c.eq.','

            if(result.and.iand(icond,32).ne.0)
     &        result=(cNext.lt.'0'.or.cNext.gt.'9').and.
     &               (cNext.lt.'a'.or.cNext.gt.'z')

            if(result.and.iand(icond,64).ne.0)
     &        result=(cNext.lt.'a'.or.cNext.gt.'z')

            if(result.and.iand(icond,128).ne.0)
     &        result=cNext.eq.' '.or.cNext.eq.'('

            if(result.and.iand(icond,256).ne.0.and.icol1.gt.1)then
              ii=1
              do iii=1,icol1-1
                c2=a(iii:iii)
                if(c2.ge.'0'.and.c2.le.'9')ii=iii+1
```

```
            if(c2.ne.' '.and.(c2.lt.'0'.or.c2.gt.'9'))goto 20
          enddo
20          if(ii.lt.icol1)then
            result=a(ii:icol1-1).eq.' '
          endif
        endif

        find=result
        if(result)return
      endif
    enddo
    find=result
    return
    end
 c------------------------------------------------------------ ------------
    function LenA(a)                    ! Length of string, at
                                        !  least 1
c Program by Mitchell R Grunes, (grunes at domain yahoo.com).
c Revision date: 8/25/96.
    character*(*) a
    n=len(a)
    dowhile(n.gt.1.and.a(n:n).eq.' ')
      n=n-1
    enddo
    LenA=n
    end
------------------END diagrami.f--------------------
-----------------BEGIN diagrami.sh--------------------
#!/bin/csh
#                    ---diagrami.sh---
#Unix csh procedure to diagram an IDL or PV-WAVE language program.

#On some unix systems $1 should be replaced by %1.

# by Mitchell R Grunes.
# for his own use, in his own time

#I assume that the executable and this procedure are in the search
path,
#  and that this procedure has execute permission.

#Syntax:
#    diagrami.sh
#to be prompted for input parameters.

#Alternate Syntax:
#    diagrami.sh filename(s)
#to append diagram of file(s) into diagram.out
```

```
  if  (${?noclobber}) then
    unset noclobber
    set   noclobbersave
  endif

  if $1a == a  then
    diagrami
    goto quit
  endif

loop:
  echo ========================-- $1 --========================
  #Prompt answers: input from $1, output to diagram2.sc (for now),
  # Don't place line numbers anyware , don't use IBM PC graphics,
  # warn if 'end' ends if, for, etc.

  echo $1          >  diagram.sc
  echo diagram2.sc >> diagram.sc
  echo 0           >> diagram.sc
  echo 0           >> diagram.sc
  echo 0           >> diagram.sc

  diagrami < diagram.sc
  cat diagram2.sc >> diagram.out
  rm -f diagram.sc
  rm -f diagram2.sc
  shift
  if ! ($1a == a) then
    goto loop
  endif
quit:
  echo Note--This does not delete diagram.out before appending to it.
  if  (${?noclobbersave}) then
    set   noclobber
    unset noclobbersave
  endif
------------------END diagrami.sh--------------------
-----------------BEGIN diagrami.bat-------------------
rem                    ---diagrami.bat---
rem MS-DOS procedure to diagram an IDL or PV-WAVE language program.

rem  by Mitchell R Grunes.

rem I assume that the executable is in directory c:\grunes on
rem  your PC.

rem Syntax:
```

```
rem     diagrami
rem to be prompted for input parameters.

rem Alternate Syntax:
rem     diagrami filename(s)
rem to append diagram of file(s) into diagram.out

if %1a == a  c:\grunes\diagrami
if %1a == a  goto quit

echo off
:loop
  echo ========================-- %1 --=======================
  rem Prompt answers: input from %1, output to diagram2.sc (for now),
  rem  does not show line numbers, use IBM PC graphics.

  echo %1          >  diagram.sc
  echo diagram2.sc >> diagram.sc
  echo 0           >> diagram.sc
  echo 0           >> diagram.sc
  echo 0           >> diagram.sc

  c:\grunes\diagrami < diagram.sc
  type diagram2.sc >> diagram.out
  del diagram.sc
  del diagram2.sc
  shift
  if not %1a == a goto loop
:quit
  echo Note--This does not delete diagram.out before appending to it.
------------------END diagrami.bat--------------------
-----------------BEGIN diagrami.vax-------------------
$!                      ---diagrami.vax---
$!VAX VMS procedure to diagram an IDL or PV-WAVE language program
$!
$! by Mitchell R Grunes.
$!
$!I assume that the executable and this procedure are in the search
path,
$!  and that this procedure has execute permission.
$!
$!Syntax:
$!    @diagrami.vax
$!to be prompted for input parameters.
$!
$!Alternate Syntax:
$!    @diagrami.vax filename(s)
$!to append diagram of file(s) into diagram.out
```

```
$
$  if P1 .EQS. ""
$  then
$    define/user sys$input sys$command
$    run diagrami
$    goto quit
$  endif
$
$  write sys$output "========================-- "+P1+"
--========================"
$
$! Must pre-create diagram.out if does not exist
$  open/append/error=noSkip diagram.out diagram.out
$  goto Skip
$noSkip:
$  open/write diagram.out diagram.out
$Skip:
$  close diagram.out
$
$! Must pre-create diagram2.sc with same file attributes
$  open/write diagram2.sc diagram2.sc
$  close diagram2.sc
$
$  !Prompt answers: input from P1, output to diagram2.sc (for now),
$  ! don't place line numbers anywhere, don't use IBM PC graphics.
$
$  open/write diagram.sc diagram.sc
$  write diagram.sc "$Run diagrami"
$  write diagram.sc P1
$  write diagram.sc "diagram2.sc"
$  write diagram.sc "0"
$  write diagram.sc "0"
$  write diagram.sc "0"
$  close diagram.sc
$  @diagram.sc
$  append diagram2.sc diagram.out
$  delete diagram.sc;*
$  delete diagram2.sc;*
$
$  if (P2 .NES. "") then @diagrami.vax 'P2' 'P3' 'P4' 'P5' 'P6' 'P7'
'P8'
$  write sys$output "Note--This does not delete diagram.out before
appending to it."
$quit:
------------------END diagrami.vax--------------------
-----------------BEGIN undiagram.f-------------------
       program UnDiagram
c This attempt to extract source code from diagrammed
```

```
c  programs created by DIAGRAMC, DIAGRAMF and DIAGRAMI.
c Program by Mitchell R Grunes, ATSC/NRL (grunes at domain yahoo.com).
c Revision date: 10/17/95.
c This program was written in FORTRAN, for historic reasons.

c Note that this leaves the headers consisting of the following 3
lines:
c <formfeed>==============--<filenames>--==============
c
 ....,....1....,....2....,....3....,....4....,....5....,....6 ....,....7....,....8....,....9....,....

c  filename lines, (and the following blank lines) but prints a warning
on the screen.

c It also hasn't been extensively tested.

      character*80  FilNam
      character*160 Line,Line2
      character*1   c,c1,c2,c3,c4,c5
      parameter     (nIndent=11)

      print*,'Input file (with diagrammed code):'
      read(*,'(a80)')FilNam
      print*,FilNam
      open(1,file=FilNam,status='old')

      print*,'Output file (with undiagrammed code):'
      read(*,'(a80)')FilNam
      print*,FilNam
      open(2,file=FilNam)

      LCol=0                          ! column of |
      nLine=0

1      read(1,'(a160)',end=99)Line
      nLine=nLine+1

      Line2=' '                       ! Turn tabs to spaces
      j=1
      do i=1,160
        if(Line(i:i).eq.char(9))then
          j=(j-1)/8*8+8+1
     ! Make sure is good ASCII char
        elseif(j.le.160.and.Line(i:i).ge.' '.
     &     and.Line(i:i).lt.char(128))then
          Line2(j:j)=Line(i:i)
          j=j+1
        endif
```

```
      enddo
      Line=Line2

      i=LenA(Line)+1
      dowhile (LCol.eq.0.and.i.gt.14)        ! Find column of ending
| ####
        i=i-1
        c5=Line(i  :i)
        c4=Line(i-1:i-1)
        c3=Line(i-2:i-2)
        c2=Line(i-3:i-3)
        c1=Line(i-4:i-4)
        if( (c1.eq.'|'.or.c1.eq.char(179).or.c1.eq.char(186)).and.
     &       (c2.eq.' '.or.(c2.ge.'0'.and.c2.le.'9')).and.
     &       (c3.eq.' '.or.(c3.ge.'0'.and.c3.le.'9')).and.
     &       (c4.eq.' '.or.(c4.ge.'0'.and.c4.le.'9')).and.
     &       (            (c5.ge.'0'.and.c5.le.'9')))then
          LCol=i-4
          print*,'| column = ',LCol,' from line ',nLine
        endif
      enddo

      if(LCol.gt.0)then                    ! Remove trailing |
#### field.
        i=LenA(Line)
        if(i.eq.LCol+4)then
          c5=Line(i  :i)
          c4=Line(i-1:i-1)
          c3=Line(i-2:i-2)
          c2=Line(i-3:i-3)
          c1=Line(i-4:i-4)
          if( (c1.eq.'|'.or.c1.eq.char(179).or.c1.eq.char(186)).and.
     &         (c2.eq.' '.or.(c2.ge.'0'.and.c2.le.'9')).and.
     &         (c3.eq.' '.or.(c3.ge.'0'.and.c3.le.'9')).and.
     &         (c4.eq.' '.or.(c4.ge.'0'.and.c4.le.'9')).and.
     &         (            (c5.ge.'0'.and.c5.le.'9')))
     &         Line(LCol:160)=' '
        endif
      endif

      i=1                                  ! Remove diagram marks
      iflag=0
      c=Line(1:1)
      dowhile((c.eq.' '.and.iflag.eq.0).or.c.eq.'+'.or.c.eq.'-'.or.
     &   c.eq.'|'.or.c.eq.'*'.or.(c.ge.char(179).and.c.le.char(218)))
        Line(i:i)=' '
        if(c.ne.' ')iflag=1
        i=i+1
```

```
        c=Line(i:i)
      enddo

      if(Line(1:nIndent).eq.' ')then          ! Remove indentation
        Line2=Line(nIndent+1:160)
        Line=Line2
      else
        print*,'Wrong indentation at line ',nLine
        print*,Line(1:LenA(Line))
        print*,char(7)
      endif

      write(2,'(80a1,80a1)')(Line(i:i),i=1,LenA(Line))
      goto 1

99    end
 c------------------------------------------------------------ ------------
      function LenA(a)                    ! Length of string, at
least 1
c Program by Mitchell R Grunes, ATSC/NRL (grunes at domain yahoo.com).
c Revision date: 10/17/95.
      character*(*) a
      n=len(a)
      dowhile(n.gt.1.and.a(n:n).eq.' ')
        n=n-1
      enddo
      LenA=n
      end
-----------------END undiagram.f--------------------
```