

```
c-----  
c   Utility routines used in Phy410 course and available  
c   in 'libp410f.a' on einstein.  
c  
c   Copyright Matthew W. Choptuik, 1988-1997  
c  
c   User routines:  
c  
c       i4arg  
c       r8arg  
c       getu  
c       indlnb  
c  
c   Support routines  
c  
c       onedot  
c       s2i4  
c       s2r8  
c-----
```

```
c-----  
c   Converts argument 'argno' to integer*4 and returns  
c   value or 'defval' if parsing error or if argument is  
c   single dot ('.')  
c-----
```

```
integer function i4arg(argno,defval)
```

```
implicit      none
```

```
real*8        r8_never  
parameter     ( r8_never = -1.0d-60 )
```

```
integer       i4_never  
parameter     ( i4_never = -2 000 000 000 )
```

```
logical       onedot  
integer       iargc  
integer       s2i4
```

```
integer       argno  
integer       defval
```

```
character*32  argi
```

```
if( argno .le. iargc() ) then  
    call getarg(argno,argi)  
    if( onedot(argi) ) then  
        i4arg = defval  
    else  
        i4arg = s2i4(argi)  
        if( i4arg .eq. i4_never ) then  
            i4arg = defval  
        end if  
    end if  
else  
    i4arg = defval  
end if
```

```
return
```

```
end
```

```

c-----
c   Converts argument 'argno' to real*8 and returns
c   value or 'defval' if parsing error or if argument is
c   single dot ('.')
c-----

double precision function r8arg(argno,defval)

    implicit      none

    real*8        r8_never
    parameter     ( r8_never = -1.0d-60 )

    integer       i4_never
    parameter     ( i4_never = -2 000 000 000 )

    logical       onedot
    integer       iargc
    real*8        s2r8

    integer       argno
    real*8        defval

    character*32  argi

    if( argno .le. iargc() ) then
        call getarg(argno,argi)
        if( onedot(argi) ) then
            r8arg = defval
        else
            r8arg = s2r8(argi)
            if( r8arg .eq. r8_never ) then
                r8arg = defval
            end if
        end if
    else
        r8arg = defval
    end if

    return
end

```

```
c-----  
c   Returns index of last non-blank character in S.  
c-----  
integer function indlnb(s)  
  
    character*(*)    s  
    integer          i  
  
    do indlnb = len(s) , 1 , -1  
        if( s(indlnb:indlnb) .ne. ' ' ) RETURN  
    end do  
    indlnb = 0  
  
    return  
  
end
```

```
c-----  
c   Returns first unit number .ge. umin not attached to  
c   a file.  
c-----
```

```
integer function getu()
```

```
implicit none
```

```
integer umin  
parameter ( umin = 10 )
```

```
integer umax  
parameter ( umax = 99 )
```

```
integer u  
logical opened
```

```
getu = -1
```

```
do u = umin , umax
```

```
    inquire(unit=u,opened=opened)
```

```
    if( .not. opened ) then
```

```
        getu = u
```

```
        return
```

```
    end if
```

```
end do
```

```
write(0,*) 'getu: Panic--no available unit number'
```

```
stop
```

```
end
```

```
c-----  
c   Utility routine for parameter handling.  
c-----
```

```
logical function onedot(s)
```

```
character*(*)      s
```

```
integer            indlnb
```

```
onedot = s(1:1) .eq. '.' .and. indlnb(s) .eq. 1
```

```
return
```

```
end
```

```
c-----  
c   Converts string to integer.  
c-----
```

```
integer function s2i4(s)
```

```
implicit          none
```

```
real*8           r8_never
```

```
parameter ( r8_never = -1.0d-60 )
```

```
integer          i4_never
```

```
parameter ( i4_never = -2 000 000 000 )
```

```
integer          indlnb
```

```
character*(*)    s
```

```
character*32     buffer
```

```
integer          lens
```

```
integer          default
```

```
parameter ( default = 0 )
```

```
lens = indlnb(s)
```

```
if( lens .gt. 99 ) then
```

```

        write(*,*) '>>> s2i4:: String too long for conversion.'
        s2i4 = default
    else
        if( lens .le. 9 ) then
            write(buffer,100) lens
        else
            write(buffer,101) lens
        end if
100     format(' (I',i1,')')
101     format(' (I',i2,')')
        read(s,fmt=buffer,end=900,err=900) s2i4
    end if

    return

900     s2i4 = i4_never
    return

end

```

```

c-----
c   Converts string to real*8 value.
c-----

```

```

double precision function s2r8(s)

```

```

    implicit      none

```

```

    real*8        r8_never

```

```

    parameter ( r8_never = -1.0d-60 )

```

```

    integer       i4_never

```

```

    parameter ( i4_never = -2 000 000 000 )

```

```

    integer       indlnb

```

```

    character*(*) s

```

```

    character*32  buffer

```

```

    integer       lens

```

```

double precision default
parameter      ( default = 0.0d0 )

lens = indlnb(s)
if( lens .gt. 99 ) then
    write(*,*) '>>> s2r8:: String too long for conversion.'
    s2r8 = default
else
    if( lens .le. 9 ) then
        write(buffer,100) lens
    else
        write(buffer,101) lens
    end if
100    format(' (G',i1,'.0)')
101    format(' (G',i2,'.0)')
    read(s,fmt=buffer,end=900,err=900) s2r8
end if

return

900    s2r8 = r8_never
return

end

```



```

c-----
c   Driver illustrating use of 'p410f' utility routines.
c-----

      program      p410fsa

      implicit     none

      integer      iargc,  indlnb,  i4arg,  getu
      real*8       r8arg

      integer      arg1
      real*8       arg2
      character*64 arg3

      if( iargc() .ne. 3 ) go to 900

      arg1 = i4arg(1,-1)
      arg2 = r8arg(2,-1.0d0)
      call getarg(3,arg3)
      write(0,*) 'p410fsa: Integer argument ', arg1
      write(0,*) 'p410fsa: Real*8 argument ', arg2
      write(0,*) 'p410fsa: String argument ',
&          arg3(1:indlnb(arg3))
      write(0,*) 'p410fsa: Next available unit number ', getu()
      stop

900  continue
      write(0,*) 'usage: p410fsa <i4arg> <r8arg> <sarg>'
      stop
      end

```

```

#####
# Sample output from 'p410fsa' illustrating use of
# 'p410f' utility routines.
#####
% p410fsa
usage: p410fsa <i4arg> <r8arg> <sarg>

% p410fsa 10 3.14 'A string'
p410fsa: Integer argument          10
p410fsa: Real*8 argument    3.1400000000000000
p410fsa: String argument A string
p410fsa: Next available unit number          10

% p410fsa bad_integer bad_real string
p410fsa: Integer argument          -1
p410fsa: Real*8 argument    -1.0000000000000000
p410fsa: String argument string
p410fsa: Next available unit number          10

#####
# For 'i4arg' and 'r8arg', a single dot ('.') is parsed into
# the default value.
#####
% p410fsa . . .
p410fsa: Integer argument          -1
p410fsa: Real*8 argument    -1.0000000000000000
p410fsa: String argument .
p410fsa: Next available unit number

```