```
      PROGRAM TEST
      USE FMZM

!   FM_FIND_MIN is a multiple precision function minimization routine that uses Brent's method.

!   The function to be minimized or maximized is  F(X,NF).
!   X  is the argument to the function.
!   NF is the function number in case extrema to several functions are needed.

      IMPLICIT NONE
      CHARACTER(80)  :: ST1,ST2

!  declare the multiple precision variables.

      TYPE (FM), SAVE     :: A, B, TOL, XVAL, FVAL
      TYPE (FM), EXTERNAL :: F

!             Set the FM precision to 50 significant digits (plus a few more "guard digits")

      CALL FM_SET(50)

!             Find a minimum of the first function,  X**3 - 9*X + 17.
!             A, B are two endpoints of an interval in which the search takes place.

      A = 1
      B = 2

!             TOL is the error tolerance.  For most functions, the best accuracy we can obtain
!                 corresponds to about half the digits being used for the arithmetic.
!                 EPSILON(A) gives the relative accuracy of full precision, so SQRT(EPSILON(A))
!                 gives the relative accuracy of half precision.

      TOL = SQRT(EPSILON(A))

!             For this call no trace output will be done (KPRT = 0).
!             KW = 6 is used, so any error messages will go to the screen.

      WRITE (*,*) ' '
      WRITE (*,*) ' '
      WRITE (*,*) ' Case 1.  Call FM_FIND_MIN to find a relative minimum between 1 and 2'
      WRITE (*,*) '          for f(x) = X**3 - 9*X + 17.'
      WRITE (*,*) '          Use KPRT = 0, so no output will be done in the routine, then'
      WRITE (*,*) '          write the results from the main program.'

      CALL FM_FIND_MIN(1,A,B,TOL,XVAL,FVAL,F,1,0,6)

!             Write the result, using F32.30 format.

      CALL FM_FORM('F32.30',XVAL,ST1)
      CALL FM_FORM('F32.30',FVAL,ST2)
      WRITE (*,"(/'  A minimum for function 1 is'/'     x  = ',A/'   f(x) = ',A)")  &
               TRIM(ST1),TRIM(ST2)

!             Find a maximum of the first function,  X**3 - 9*X + 17.

!             This time we use FM_FIND_MIN's built-in trace (KPRT = 1) to print the final
!             approximation to the root.  The output will appear on more than one line, to
```

```
!             allow for the possibility that precision could be hundreds or thousands of digits,
!             so the number might not fit on one line.

      WRITE (*,*) ' '
      WRITE (*,*) ' '
      WRITE (*,*) ' Case 2.  Find a relative maximum between -5 and 1.'
      WRITE (*,*) '          Use KPRT = 1, so FM_FIND_MIN will print the results.'

      CALL FM_FIND_MIN(2,-TO_FM('5.0D0'),TO_FM('1.0D0'),TOL,XVAL,FVAL,F,1,1,6)

!             Find a maximum of the first function,  X**3 - 9*X + 17.

!             See what happens when the maximum value is at an endpoint of the search interval.
!             The algorithm still finds a relative maximum in the interior of the interval,
!             not the absolute maximum at x=5.

      WRITE (*,*) ' '
      WRITE (*,*) ' '
      WRITE (*,*) ' Case 3.  Find a relative maximum between -5 and 5.'
      WRITE (*,*) '          Use KPRT = 2, so FM_FIND_MIN will print all iterations,'
      WRITE (*,*) '          as well as the final results.'

      CALL FM_FIND_MIN(2,-TO_FM('5.0D0'),TO_FM('5.0D0'),TOL,XVAL,FVAL,F,1,2,6)

!             Find a minimum of the second function,  gamma(x).

      WRITE (*,*) ' '
      WRITE (*,*) ' '
      WRITE (*,*) ' Case 4.  The gamma function has one minimum for positive x.'
      WRITE (*,*) '          Find it, printing all iterations.'
      WRITE (*,*) '          Fortran did not provide gamma(x) before the Fortran 2008 standard, '
      WRITE (*,*) '          so this case was not included in the original fmin.f95.'

      CALL FM_FIND_MIN(1,TO_FM('0.1D0'),TO_FM('3.0D0'),TOL,XVAL,FVAL,F,2,2,6)


      WRITE (*,*) ' '

      END PROGRAM TEST


      FUNCTION F(X,NF)     RESULT (RETURN_VALUE)
      USE FMZM

!   X  is the argument to the function.
!   NF is the function number.

      IMPLICIT NONE
      INTEGER :: NF
      TYPE (FM) :: RETURN_VALUE,X

      IF      (NF == 1) THEN
          RETURN_VALUE = X**3 - 9*X + 17
      ELSE IF (NF == 2) THEN
          RETURN_VALUE = GAMMA(X)
      ELSE
          RETURN_VALUE = 3*X**2 + X - 2
      ENDIF
```

```
END FUNCTION F
```