

```

PROGRAM ROOTS
USE FMZM
IMPLICIT NONE

! This function was conjectured to have all of its roots on the unit circle:
!
!  $f(x) = (b+1)/2*x^a - b*x^{(a-1)} + b*x^{(a-2)} - \dots + b*x^2 - b*x + (b+1)/2$ 
!     a is an even integer, b is odd.
!     For this example function, a = 110, b = 13.

! This program confirms that in this case, all the roots are within 1e-50 of the unit circle.

! For subroutine ZM_ROOTS, the equation to be solved is written F(X,NF) = 0.
! X is the argument to the function.
! NF is the function number in case roots to several functions are needed.

INTEGER :: J, KPRT, N_FOUND
TYPE (ZM), EXTERNAL :: F
TYPE (FM) :: ERR
TYPE (ZM), ALLOCATABLE :: LIST_OF_ROOTS(:)

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

CALL FM_SET(50)
CALL FM_SETVAR(" KSWIDE = 130 ")

ALLOCATE(LIST_OF_ROOTS(110),STAT=J)
IF (J /= 0) THEN
    WRITE (*, "(/' Error in PROGRAM ROOTS. Unable to allocate arrays with N = ',I8/)" ) 110
    STOP
ENDIF

!     Use KPRT = 1, so ZM_ROOTS will print the results.

KPRT = 1

!     Find all roots of a 110th degree polynomial. (NR = 110)

WRITE (*,*) ' '
WRITE (*,*) ' '
WRITE (*,*) ' Call ZM_ROOTS to find the roots of f(x) = ( 110th degree polynomial )'
WRITE (*,*) ' '

CALL ZM_ROOTS(110,F,1,N_FOUND,LIST_OF_ROOTS,KPRT,6)

WRITE (*,*) ' '
WRITE (*,*) ' Sort these roots by argument and check how close they lie to the unit circle.'
WRITE (*,*) ' '

CALL WRITE_ROOTS(6,N_FOUND,LIST_OF_ROOTS)

!     Check that these roots all lie on the unit circle.

ERR = 0
DO J = 1, N_FOUND

```

```

      ERR = MAX(ERR,ABS(1-ABS(LIST_OF_ROOTS(J))))
ENDDO

WRITE (*,*) ' '
WRITE (*,"(A,ES16.7)") ' All roots lie on the unit circle within error ',TO_DP(ERR)
WRITE (*,*) ' '

END PROGRAM ROOTS

FUNCTION F(X,NF)
USE FMZM
IMPLICIT NONE

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

INTEGER :: J, NF
TYPE (ZM) :: F, X

CALL FM_ENTER_USER_FUNCTION(F)
IF (NF == 1) THEN

!      (b+1)/2*x^a - b*x^(a-1) + b*x^(a-2) - ... + b*x^2 - b*x + (b+1)/2
!      a is an even integer, b is odd.
!      For this example function, a = 110, b = 13.
!      All the roots of this function lie on the unit circle.
!      This function has a simpler form, but is evaluated in this long form
!      to give a stronger check of roundoff control.

      F = (13+1)/2
      DO J = 2, 110
        F = F*X + (-1)**(j-1)*13
      ENDDO
      F = F*X + (13+1)/2
ELSE IF (NF == 2) THEN
      F = SIN(X)
ELSE
      F = X**3 - 2
ENDIF

CALL FM_EXIT_USER_FUNCTION(F)
END FUNCTION F

SUBROUTINE WRITE_ROOTS(KU,N_FOUND,LIST_OF_ROOTS)
USE FMVALS
USE FMZM
IMPLICIT NONE

! Write the list of roots in order of increasing argument on unit KU.

INTEGER :: I, J, JMIN, J_PRINTED(200), KU, KW_SAVE, N_FOUND
TYPE (FM) :: ARG, SIZE
TYPE (ZM) :: LIST_OF_ROOTS(N_FOUND)
CHARACTER(50) :: STR
DOUBLE PRECISION :: AMIN, ARGS(200)

```

```

CALL FM_ENTER_USER_ROUTINE
KW_SAVE = KW
J_PRINTED = 0
DO J = 1, N_FOUND
  IF (LIST_OF_ROOTS(J) == 0) THEN
    ARGS(J) = 0
  ELSE
    CALL ZM_ARG(LIST_OF_ROOTS(J),ARG)
    ARGS(J) = ARG * 180 / ACOS(-1.0D0)
    IF (ARGS(J) < 0) ARGS(J) = 360 + ARGS(J)
  ENDIF
ENDDO

DO I = 1, N_FOUND
  JMIN = 1
  AMIN = 420
  DO J = 1, N_FOUND
    IF (J_PRINTED(J) == 0 .AND. ARGS(J) < AMIN) THEN
      JMIN = J
      AMIN = ARGS(J)
    ENDIF
  ENDDO

  J_PRINTED(JMIN) = 1
  KW = KU
  SIZE = ABS(LIST_OF_ROOTS(JMIN))
  CALL ZM_PRINT(LIST_OF_ROOTS(JMIN))
  IF (SIZE < 1000) THEN
    WRITE (KU,"(9X,'Argument (degrees) = ',F15.10,5X,'Magnitude = ',F15.10)") AMIN, &
      TO_DP(SIZE)
  ELSE
    CALL FM_FORM(" ES27.10 ",SIZE,STR)
    WRITE (KU,"(9X,'Argument (degrees) = ',F15.10,5X,'Magnitude = ',A)") AMIN, TRIM(STR)
  ENDIF
  WRITE (KU,*) ' '
ENDDO

KW = KW_SAVE

CALL FM_EXIT_USER_ROUTINE
END SUBROUTINE WRITE_ROOTS

```