```fortran
      program test

!   One use for FM involves programs that don't need multiple precision results but do need some
!   of the special functions available in FM but not in the Fortran standard.  These include:

!   bernoulli(n)
!   beta(x, y)
!   binomial(n, k) or binomial(x, y)
!   cos_integral(x)
!   cosh_integral(x)
!   exp_integral_ei(x)
!   exp_integral_en(n, x)
!   fresnel_c(x)
!   fresnel_s(x)
!   incomplete_beta(x, a, b)
!   incomplete_gamma1(x, y)
!   incomplete_gamma2(x, y)
!   log_integral(x)
!   pochhammer(x, n)
!   polygamma(n, x)
!   psi(x)
!   sin_integral(x)
!   sinh_integral(x)

!   See the complete list of FM functions in FM_User_Manual.txt.

!   For this application, no type(fm) variables need to be declared.  Just add use fmzm at the top
!   and compile and link the program like SampleFM.f95.

      use fmzm
      implicit none

      integer :: j
      double precision :: a, b, c, c_fm, err, max_err

!              To use with 53-bit double precision, having about 16 significant digits of accuracy,
!              set the FM precision to 16 digits.

      call fm_set(16)


!              1.  Check to see if Fortran's intrinsic gamma function is correctly rounded.

!                  a is the double precision variable, so gamma(a) uses Fortran's intrinsic gamma.

!                  to_fm(a) converts a to an FM number, so gamma( to_fm(a) ) uses FM's gamma,
!                  then the "=" rounds the result back to double precision variable c_fm.

!                  It is possible that different compilers might give different results for this
!                  test.  Some compilers may not give results that are correctly rounded to full
!                  double precision accuray when a is large, but c_fm should be correctly rounded.

      max_err = 0
      do j = 10, 150, 10
         a = j + 0.5d0
         c = gamma(a)
         c_fm = gamma( to_fm(a) )
```

```fortran
          err = abs( (c - c_fm) / c_fm )
          if (err > max_err) then
              max_err = err
              b = a
          endif
      enddo

      write (*, "(///a/)") " Sample 1.  Compare Fortran's built-in gamma function to FM's"
      if (max_err > 0) then
          a = b
          write (*, "(a, es13.7, a, f7.3)") ' Maximum relative error in Fortran gamma was ',  &
                                            max_err, ' for a = ', a
          c = gamma(a)
          write (*, "(es25.15, a)") c,     ' = gamma(a)'
          c_fm = gamma( to_fm(a) )
          write (*, "(es25.15, a)") c_fm, ' = gamma( to_fm(a) )'
      else
          write (*, "(a)") ' All Fortran gamma results were correctly rounded.'
      endif


!           2.  Binomial coefficients.

!               Find the probability of getting exactly 10,000 heads in 20,000 tosses
!               of a fair coin.

!               Here we could not store the results of the binomial and power separately in
!               double precision, since binomial( 20000, 10000 ) = 2.2e+6018 and
!               2**20000 = 4.0e+6020 would both overflow in double precision.

      write (*, "(//a)") " Sample 2.  Binomial coefficients"
      write (*, "(a)")   "            Find the probability of getting exactly 10,000 heads"
      write (*, "(a/)")  "            in 20,000 tosses of a fair coin."

      c_fm = binomial( to_fm(20000), to_fm(10000) ) / to_fm(2)**20000

      write (*, "(a, f20.16)") " binomial( to_fm(20000), to_fm(10000) ) / to_fm(2)**20000 =", c_fm


!           3.  Log Integral function.

!               Estimate the number of primes less than 10**30.

      write (*, "(//a)") " Sample 3.  Log integral"
      write (*, "(a/)")  "            Estimate the number of primes less than 10**30."

      c_fm = log_integral( to_fm('1.0e+30') )

      write (*, "(a, es23.15)") " log_integral(to_fm('1.0e+30')) =", c_fm


!           4.  Psi and polygamma functions.

!               Rational series can often be summed using these functions.
!               Sum (n=1 to infinity) 1/(n**2 * (8n+1)**2) =
!               16*(psi(1) - psi(9/8)) + polygamma(1, 1) + polygamma(1, 9/8)
!               Reference: Abramowitz & Stegun, Handbook of Mathematical Functions,
!               chapter 6, Example 10.
```

```
      write (*, "(//a)") " Sample 4.  Psi and polygamma functions."
      write (*, "(a)")   "            Sum (n=1 to infinity) 1/(n**2 * (8n+1)**2) ="
      write (*, "(a/)")  "            16*(psi(1) - psi(9/8)) + polygamma(1, 1) + polygamma(1, 9/8)"

      c_fm = 16*( psi( to_fm(1) ) - psi( to_fm(9)/8 ) ) +           &
             polygamma( 1, to_fm(1) ) + polygamma( 1, to_fm(9)/8 )

      write (*, "(a, f19.16)") " Sum =", c_fm


!             5.  Incomplete gamma and gamma functions.

!                 Find the probability that an observed chi-square for a correct model should be
!                 less that 2.3 when the number of degrees of freedom is 5.
!                 Reference: Knuth, Volume 2, 3rd ed., Page 56, and Press, Flannery, Teukolsky,
!                 Vetterling, Numerical Recipes, 1st ed., Page 165.

      write (*, "(//a/)") " Sample 5.  Incomplete gamma and gamma functions."

      c_fm = incomplete_gamma1( to_fm(5)/2, to_fm('2.3')/2 ) / gamma( to_fm(5)/2 )

      write (*, "(a, f19.16/)") " Probability =", c_fm


      end program test
```