```
Case 1.  Call fm_secant to find a root between 1 and 2
         for f(x) = x**2 - 3.
         Use kprt = 0, so no output will be done in the routine, then
         write the results from the main program.

A root for function 1 is    1.7320508075688772935274463415 06

 

Case 2.  Find a root between 6 and 7 for  f(x) = x*tan(x) - 1.
         Use kprt = 1, so fm_secant will print the result.

fm_secant.   Function   2     11 iterations.
               Estimated relative error = 7.767234M-58,    Root:
     6.437298179171947120362639851025633245321734171448 0M+0




Case 3.  Find a root between 1 and 5 for  f(x) = gamma(x) - 10.
         Use kprt = 2, so fm_secant will print all iterations,
         as well as the final result.

fm_secant.  Begin trace of all iterations.
j =  0   f(ax) = -9.0000000000M+0             x:
     1.00000000000000000000000000000000000000000000000000M+0
j =  0   f(bx) =  1.4000000000M+1             x:
     5.00000000000000000000000000000000000000000000000000M+0
j =  1    f(x) = -8.6068421859M+0             x:
     2.56521739130434782608695652173913043478260869565220M+0
j =  2    f(x) = -6.7051471157M+0             x:
     3.49218408119506747687260540946434434335867938772 30M+0
j =  3    f(x) =  4.5186929492M+2             x:
     6.76055666353445534279536816831991858866337462956 52M+0
j =  4    f(x) = -6.5259077772M+0             x:
     3.53997331012937087129416838909684762302160227707 89M+0
j =  5    f(x) = -6.3422248053M+0             x:
     3.58582289544246674833732092594485657751743689261 13M+0
j =  6    f(x) =  2.1049716711M+1             x:
     5.16892215824446792825168480436803712195984757261 18M+0
j =  7    f(x) = -4.3466346198M+0             x:
     3.95236761107668848252072092438928858057255320997 91M+0
j =  8    f(x) = -2.6325150260M+0             x:
     4.16058327334707554918538429732193402987456551861 53M+0
j =  9    f(x) =  1.3192340241M+0             x:
     4.48035726200827339813338363806565699240362702631 22M+0
j = 10    f(x) = -2.2137831189M-1             x:
     4.37360536037808730000623281662006304743625678767 79M+0
j = 11    f(x) = -1.5402542083M-2             x:
     4.38894507642629638093258362299539513014111724757 95M+0
j = 12    f(x) =  1.9744111815M-4             x:
     4.39009215611532489098862456931143665216193537977 57M+0
j = 13    f(x) = -1.7323119072M-7             x:
     4.39007763810633208679519149800636292867308255229 73M+0
j = 14    f(x) = -1.9461226903M-12            x:
     4.39007765083299891599584800431293300632856019711 90M+0
j = 15    f(x) =  1.9182672042M-20            x:
     4.39007765083314189217256600429623978032482678996 07M+0
```

```
j = 16    f(x) = -2.1241838752M-33            x:
     4.3900776508331418921711567071874581130384786044923M+0
j = 17    f(x) = -2.3185368313M-54            x:
     4.3900776508331418921711567071874582690963097189549M+0
j = 18    f(x) = -1.0000000000M-77            x:
     4.3900776508331418921711567071874582690963097189549M+0
j = 19    f(x) = -1.0000000000M-77            x:
     4.3900776508331418921711567071874582690963097189549M+0

fm_secant.   Function   3     19 iterations.
               Estimated relative error = 1.138932M-57,    Root:
     4.3900776508331418921711567071874582690963097189549M+0



Case 4.  Find a root between 1 and 2 for  f(x) = polygamma(0, x).
         Use kprt = 1, so fm_secant will print the result.

fm_secant.   Function   4     11 iterations.
               Estimated relative error = 3.420833M-57,    Root:
     1.4616321449683623412626595423257213284681962040064M+0



Case 5.  Find a root near 3.1 for  f(x) = cos(x) + 1.  (Double root)
         Use kprt = 2, so fm_secant will print the iterations.

fm_secant.  Begin trace of all iterations.
j =  0   f(ax) =  8.6484972672M-4             x:
     3.1000000000000000000000000000000000000000000000000M+0
j =  0   f(bx) =  1.7052242052M-3             x:
     3.2000000000000000000000000000000000000000000000000M+0
j =  1    f(x) =  1.0422702438M-2             x:
     2.9970875783570827008391241954412154141991368316547M+0
j =  2    f(x) =  4.8078499092M-3             x:
     3.2396916589497320151463064609294213432799905009368M+0
j =  3    f(x) =  4.6403965569M-2             x:
     3.4474271249802673219623508910459796120747925426733M+0
j =  4    f(x) =  2.7432671392M-3             x:
     3.2156807404865076385409933147079484010368360207527M+0
j =  5    f(x) =  1.7712155432M-3             x:
     3.2011197677104049910687744827986342665736646250163M+0
j =  6    f(x) =  5.4428436058M-4             x:
     3.1745876150836989946718504008020165861243120692951M+0
j =  7    f(x) =  2.2524007549M-4             x:
     3.1628175696799296207951052135294249468006868205820M+0
j =  8    f(x) =  8.3403335932M-5             x:
     3.1545081090839358367174767078372923961650525112318M+0
j =  9    f(x) =  3.2234635163M-5             x:
     3.1496219509866413602496999070622136243836055823282M+0
j = 10    f(x) =  1.2257041924M-5             x:
     3.1465438285944167712048011249784096248313251195505M+0
j = 11    f(x) =  1.5313139121M-3             x:
     3.1446552790200871817410158401485968932236201175305M+0
j = 12    f(x) =  2.4832118180M-3             x:
     3.1465590670175217495090723179227983616801874230198M+0
j = 13    f(x) =  5.0884881537M-9             x:
     3.1415926637667695457830203649682113061534827694136M+0
```

```
j = 14    f(x) =  1.0459100332M-14            x:
     3.1415926535898141566633070137156315280367587416150M+0
j = 15    f(x) =  9.0271675357M-32            x:
     3.1415926535897932384626433832796834275478838739507M+0
j = 16    f(x) =  3.2916898334M-60            x:
     3.1415926535897932384626433832795028841971693993751M+0
j = 17    f(x) =  5.0068598259M-79            x:
     3.1415926535897932384626433832795028841971693993751M+0
j = 18    f(x) =  5.0068598259M-79            x:
     3.1415926535897932384626433832795028841971693993751M+0

fm_secant.   Function   5     18 iterations.
               Estimated relative error = 1.591549M-57,    Root:
     3.1415926535897932384626433832795028841971693993751M+0



Case 6.  Find a root near 3.1 for  f(x) = cos(x) + 1 - 1.0e-40.
         There are two different roots that agree to about 20 digits,
         so here the convergence is slower.
         Use kprt = 1, so fm_secant will print the result.

fm_secant.   Function   6     54 iterations.
               Estimated relative error = 1.591549M-57,    Root:
     3.1415926535897932384767855189032338346851862866172M+0



Case 7.  Find a root near 3.1 for  f(x) = sin(x)**3.  (Triple root)
         Use kprt = 2, so fm_secant will print the iterations.

fm_secant.  Begin trace of all iterations.
j =  0   f(ax) =  7.1890948202M-5             x:
     3.1000000000000000000000000000000000000000000000000M+0
j =  0   f(bx) = -1.9891226494M-4             x:
     3.2000000000000000000000000000000000000000000000000M+0
j =  1    f(x) =  3.4053191807M-6             x:
     3.1265473025109802113579063768052712098830330073810M+0
j =  2    f(x) = -4.6033019682M-3             x:
     3.1277836254965637319671255990161395030696935340751M+0
j =  3    f(x) = -5.0151907695M-3             x:
     3.1265482164119758136089201494691601198372701516572M+0
j =  4    f(x) = -6.6600352394M-7             x:
     3.1415906555792214179684872785638182310056451601383M+0
j =  5    f(x) = -5.0254086750M-11            x:
     3.1415926534390309782133329510918323393079713387829M+0
j =  6    f(x) = -6.6877258692M-23            x:
     3.1415926535897932384624427515034254892529502674081M+0
j =  7    f(x) = -5.0669018059M-43            x:
     3.1415926535897932384626433832795028841971678793046M+0
j =  8    f(x) =  3.3379065506M-79            x:
     3.1415926535897932384626433832795028841971693993751M+0
j =  9    f(x) =  3.3379065506M-79            x:
     3.1415926535897932384626433832795028841971693993751M+0

fm_secant.   Function   7      9 iterations.
               Estimated relative error = 1.591549M-57,    Root:
     3.1415926535897932384626433832795028841971693993751M+0
```