```
 Example 1.   Compute f(x) = x**2 - x + 3 using different formulas.

 x is the interval [ -0.5 , 1.0 ]

x**2 gives  [  0.000 ,  1.000 ]
x*x  gives  [ -0.500 ,  1.000 ]


x**2 - x + 3        gives  [  2.000 ,  4.500 ].  Magnification =  2.500
x*x - x + 3         gives  [  1.500 ,  4.500 ].  Magnification =  3.000
x*(x - 1) + 3       gives  [  1.500 ,  3.750 ].  Magnification =  2.250
(x - 0.5)**2 + 2.75 gives  [  2.750 ,  3.750 ].  Magnification =  1.000



 x is the interval [ 0.1 , 1.0 ]

x**2 - x + 3        gives  [  2.010 ,  3.900 ].  Magnification =  7.560
x*x - x + 3         gives  [  2.010 ,  3.900 ].  Magnification =  7.560
x*(x - 1) + 3       gives  [  2.100 ,  3.000 ].  Magnification =  3.600
(x - 0.5)**2 + 2.75 gives  [  2.750 ,  3.000 ].  Magnification =  1.000



 x is the interval [ 0.9 , 1.0 ]

x**2 - x + 3        gives  [  2.810 ,  3.100 ].  Magnification =  3.222
x*x - x + 3         gives  [  2.810 ,  3.100 ].  Magnification =  3.222
x*(x - 1) + 3       gives  [  2.900 ,  3.000 ].  Magnification =  1.111
(x - 0.5)**2 + 2.75 gives  [  2.910 ,  3.000 ].  Magnification =  1.000



 x is the interval [ 0.99 , 1.0 ]

x**2 - x + 3        gives  [  2.980 ,  3.010 ].  Magnification =  3.020
```

```
x*x - x + 3         gives  [  2.980 ,  3.010 ].   Magnification =  3.020

x*(x - 1) + 3       gives  [  2.990 ,  3.000 ].   Magnification =  1.010

(x - 0.5)**2 + 2.75 gives  [  2.990 ,  3.000 ].   Magnification =  1.000


 Example 2.  Sum  1 / n^7  from 1 to 100,000 =
      1.0083492773819228268397975498496300979331385605652M+0
      1.0083492773819228268397975498496300979331385605653M+0

     The two endpoints agree to about 51 decimal digits.


 Example 3.  Integrate  sin(t) / t  from  t = 0  to  t = 20:
      1.5482417010434398401636433421295136922615733621093M+0
      1.5482417010434398401636433421295136922615733621094M+0

     The two endpoints agree to about 54 decimal digits.


 Example 4.  Step around the unit circle multiple times using a recurrence

  1 trips.  The two endpoints for x agree to about 54 decimal digits.
  2 trips.  The two endpoints for x agree to about 52 decimal digits.
  3 trips.  The two endpoints for x agree to about 49 decimal digits.
  4 trips.  The two endpoints for x agree to about 47 decimal digits.
  5 trips.  The two endpoints for x agree to about 44 decimal digits.
  6 trips.  The two endpoints for x agree to about 42 decimal digits.
  7 trips.  The two endpoints for x agree to about 39 decimal digits.
  8 trips.  The two endpoints for x agree to about 36 decimal digits.
  9 trips.  The two endpoints for x agree to about 34 decimal digits.
 10 trips.  The two endpoints for x agree to about 31 decimal digits.
 11 trips.  The two endpoints for x agree to about 29 decimal digits.
 12 trips.  The two endpoints for x agree to about 26 decimal digits.
 13 trips.  The two endpoints for x agree to about 24 decimal digits.
 14 trips.  The two endpoints for x agree to about 21 decimal digits.
 15 trips.  The two endpoints for x agree to about 18 decimal digits.
 16 trips.  The two endpoints for x agree to about 16 decimal digits.
 17 trips.  The two endpoints for x agree to about 13 decimal digits.
 18 trips.  The two endpoints for x agree to about 11 decimal digits.
 19 trips.  The two endpoints for x agree to about  8 decimal digits.
 20 trips.  The two endpoints for x agree to about  6 decimal digits.
 21 trips.  The two endpoints for x agree to about  3 decimal digits.
 22 trips.  The two endpoints for x agree to about  1 decimal digits.
 23 trips.  The two endpoints for x agree to about  0 decimal digits.

 x =
     -5.8564010117471012542822698126300121646708464338966M+1
      6.0564010117471012542822698126300121646708464338966M+1

 y =
     -5.9564010117471012542822698126300121646708464338966M+1
      5.9564010117471012542822698126300121646708464338966M+1

     The two endpoints for x agree to about  0 decimal digits.
```

```
Compare the same calculation in ordinary FM arithmetic.

 1 trips.  x agrees with 1.0 to about 62 decimal digits.
 2 trips.  x agrees with 1.0 to about 61 decimal digits.
 3 trips.  x agrees with 1.0 to about 61 decimal digits.
 4 trips.  x agrees with 1.0 to about 61 decimal digits.
 5 trips.  x agrees with 1.0 to about 61 decimal digits.
 6 trips.  x agrees with 1.0 to about 61 decimal digits.
 7 trips.  x agrees with 1.0 to about 61 decimal digits.
 8 trips.  x agrees with 1.0 to about 61 decimal digits.
 9 trips.  x agrees with 1.0 to about 61 decimal digits.
10 trips.  x agrees with 1.0 to about 61 decimal digits.
11 trips.  x agrees with 1.0 to about 61 decimal digits.
12 trips.  x agrees with 1.0 to about 60 decimal digits.
13 trips.  x agrees with 1.0 to about 60 decimal digits.
14 trips.  x agrees with 1.0 to about 60 decimal digits.
15 trips.  x agrees with 1.0 to about 60 decimal digits.
16 trips.  x agrees with 1.0 to about 60 decimal digits.
17 trips.  x agrees with 1.0 to about 60 decimal digits.
18 trips.  x agrees with 1.0 to about 60 decimal digits.
19 trips.  x agrees with 1.0 to about 60 decimal digits.
20 trips.  x agrees with 1.0 to about 60 decimal digits.
21 trips.  x agrees with 1.0 to about 60 decimal digits.
22 trips.  x agrees with 1.0 to about 60 decimal digits.
23 trips.  x agrees with 1.0 to about 60 decimal digits.

x =
     1.0000000000000000000000000000000000000000000000000M+0

y =
    -2.2745102100000000000000000000000000000000000000000M-55


Example 4b.  Step back and forth around the unit circle using a recurrence


Starting point for x =
     8.7630668004386358730811590392206258339906423848214M-1
     8.7630668004386358730811590392206258339906423848215M-1

Starting point for y =
     4.8175367410171527498719150287212965352854201015207M-1
     4.8175367410171527498719150287212965352854201015208M-1

 5 trips.  The two endpoints for x agree to about 59 decimal digits.
10 trips.  The two endpoints for x agree to about 56 decimal digits.
15 trips.  The two endpoints for x agree to about 54 decimal digits.
20 trips.  The two endpoints for x agree to about 51 decimal digits.
25 trips.  The two endpoints for x agree to about 48 decimal digits.
30 trips.  The two endpoints for x agree to about 46 decimal digits.
35 trips.  The two endpoints for x agree to about 43 decimal digits.
40 trips.  The two endpoints for x agree to about 41 decimal digits.
45 trips.  The two endpoints for x agree to about 38 decimal digits.
50 trips.  The two endpoints for x agree to about 36 decimal digits.
55 trips.  The two endpoints for x agree to about 33 decimal digits.
60 trips.  The two endpoints for x agree to about 30 decimal digits.
65 trips.  The two endpoints for x agree to about 28 decimal digits.
70 trips.  The two endpoints for x agree to about 25 decimal digits.
```

```
 75 trips.  The two endpoints for x agree to about 23 decimal digits.
 80 trips.  The two endpoints for x agree to about 20 decimal digits.
 85 trips.  The two endpoints for x agree to about 18 decimal digits.
 90 trips.  The two endpoints for x agree to about 15 decimal digits.
 95 trips.  The two endpoints for x agree to about 12 decimal digits.
100 trips.  The two endpoints for x agree to about 10 decimal digits.
105 trips.  The two endpoints for x agree to about  7 decimal digits.
110 trips.  The two endpoints for x agree to about  5 decimal digits.
115 trips.  The two endpoints for x agree to about  2 decimal digits.

 x =
      8.7369369964583434983069574632632545487869869160581M-1
      8.7891966044189282478553606151779971191942978535848M-1

 y =
      4.7914069370368603750977134527639252500817646327574M-1
      4.8436665449974451246461166046786678204890755702842M-1

    The two endpoints for x agree to about  2 decimal digits.


 Example 5.  2 * Product  4*n^2 / (4*n^2-1)  from 1 to 10,000. =
      3.1415141186819220469785580507138775513425133394700M+0
      3.1415141186819220469785580507138775513425133394701M+0

    The two endpoints of y(30) agree to about 52 decimal digits.


 Example 6.  Differential equation.  y'' = -y' / 10  -  2 * y / (x+2)

 x =    3.0000000000   v =    0.831136520960373  -0.452290751829937     Endpoints of v(1) agree to  53
                     digits.
 x =    6.0000000000   v =   -0.765883193156779  -0.352158220002235     Endpoints of v(1) agree to  52
                     digits.
 x =    9.0000000000   v =   -0.821640364088937   0.257782946044846     Endpoints of v(1) agree to  52
                     digits.
 x =   12.0000000000   v =    0.193374374107742   0.325345242516926     Endpoints of v(1) agree to  51
                     digits.
 x =   15.0000000000   v =    0.786966466722527   0.052221392312944     Endpoints of v(1) agree to  51
                     digits.
 x =   18.0000000000   v =    0.572088550349089  -0.166039347343631     Endpoints of v(1) agree to  50
                     digits.
 x =   21.0000000000   v =   -0.008218454426127  -0.190332029943315     Endpoints of v(1) agree to  48
                     digits.
 x =   24.0000000000   v =   -0.440713401930602  -0.086656234642814     Endpoints of v(1) agree to  49
                     digits.
 x =   27.0000000000   v =   -0.515393780043246   0.031589999652254     Endpoints of v(1) agree to  48
                     digits.
 x =   30.0000000000   v =   -0.307766881676032   0.095343042626880     Endpoints of v(1) agree to  48
                     digits.

 y(30) =

    -3.0776688167603169203382987651096461581717811055464M-1
    -3.0776688167603169203382987651096461581717811055418M-1

 y'(30) =
```

```
     9.534304262688028628484324684148983359660244262628 6M-2
```

```
     9.5343042626880286284843246841489833596602442626286M-2
     9.5343042626880286284843246841489833596602442626426M-2
```

The two endpoints of y(30) agree to about 48 decimal digits.

Example 7. Differential equation. y'' = -y' / 10 - 200 * y / (x+2)

```
x =    3.0000000000   v =   -0.102742064000739  -0.215544901796916     Endpoints of v(1) agree to  47
                     digits.
x =    6.0000000000   v =    0.078485157239314  -0.348572728869748     Endpoints of v(1) agree to  39
                     digits.
x =    9.0000000000   v =   -0.037481204238851  -0.383488535010064     Endpoints of v(1) agree to  33
                     digits.
x =   12.0000000000   v =    0.013531679745067  -0.333961311141985     Endpoints of v(1) agree to  27
                     digits.
x =   15.0000000000   v =    0.075496648457680   0.094822963645320     Endpoints of v(1) agree to  23
                     digits.
x =   18.0000000000   v =   -0.072047742909790   0.022294258042798     Endpoints of v(1) agree to  19
                     digits.
x =   21.0000000000   v =    0.063355820821447   0.032461966986868     Endpoints of v(1) agree to  14
                     digits.
x =   24.0000000000   v =   -0.029168578777506  -0.135277237857925     Endpoints of v(1) agree to  10
                     digits.
x =   27.0000000000   v =   -0.036220481301623   0.094265735252438     Endpoints of v(1) agree to   7
                     digits.
x =   30.0000000000   v =    0.025275350395373   0.090890872993315     Endpoints of v(1) agree to   3
                     digits.
```

y(30) =

```
     2.5268274663776318683290452013354524008325842304305M-2
     2.5282426126969610817979865312140233217038151226782M-2
```

y'(30) =

```
     9.0872770822409973135783151703562888452619638318962M-2
     9.0908975164219866493786176735651583524503965546846M-2
```

The two endpoints of y(30) agree to about 3 decimal digits.

Example 8. Solve a "random" n x n linear system.

For n = 10 the solution elements agree to between 51 and 53 significant digits.

For n = 20 the solution elements agree to between 47 and 50 significant digits.

For n = 30 the solution elements agree to between 44 and 48 significant digits.

For n = 40 the solution elements agree to between 36 and 46 significant digits.

For n = 50 the solution elements agree to between 35 and 43 significant digits.

For n = 60 the solution elements agree to between 28 and 40 significant digits.

For n = 70 the solution elements agree to between 27 and 37 significant digits.

```
For n =  80 the solution elements agree to between  23 and  34 significant digits.

For n =  90 the solution elements agree to between  16 and  33 significant digits.

For n = 100 the solution elements agree to between  16 and  30 significant digits.


 Example 9.  Newton's method starting with a single point.


Iteration  1.  x =
      8.6787944117144232159552377016146086744581113103176M-1
      8.6787944117144232159552377016146086744581113103177M-1

Iteration  2.  x =
      8.5278337341640992137798507086554877127484032504254M-1
      8.5278337341640992137798507086554877127484032504255M-1

Iteration  3.  x =
      8.5260552636892205445692227465314123881418241936054M-1
      8.5260552636892205445692227465314123881418241936055M-1

Iteration  4.  x =
      8.5260550201372594802652409548836209726836629454221M-1
      8.5260550201372594802652409548836209726836629454222M-1

Iteration  5.  x =
      8.5260550201372549134647241469547803262670602201255M-1
      8.5260550201372549134647241469547803262670602201256M-1

Iteration  6.  x =
      8.5260550201372549134647241469531746689845330015140M-1
      8.5260550201372549134647241469531746689845330015141M-1

Iteration  7.  x =
      8.5260550201372549134647241469531746689845330015140M-1
      8.5260550201372549134647241469531746689845330015141M-1

Iteration  8.  x =
      8.5260550201372549134647241469531746689845330015140M-1
      8.5260550201372549134647241469531746689845330015141M-1

Iteration  9.  x =
      8.5260550201372549134647241469531746689845330015140M-1
      8.5260550201372549134647241469531746689845330015141M-1

Iteration 10.  x =
      8.5260550201372549134647241469531746689845330015140M-1
      8.5260550201372549134647241469531746689845330015141M-1


 Example 10.   Newton's method starting with an interval containing a root


Iteration  1.  x =
      8.2582914751966549600358492004459062470952152212196M-1
      9.1669483793964048943502242929035570875441708134219M-1
```

```
Iteration  2.  x =
      8.5153653288747945014206990942716504355878954736541M-1
      8.5410378857010420828800408165620227158607956580201M-1

Iteration  3.  x =
      8.5260511282637639422279677005694780814425748090707M-1
      8.5260596120050999165916620170701727472347232648131M-1

Iteration  4.  x =
      8.5260550201370357420197023626581603375273929231293M-1
      8.5260550201374929467463703975692934764436548299921M-1

Iteration  5.  x =
      8.5260550201372549134647241466280563503694077596793M-1
      8.5260550201372549134647241472919881356741988716405M-1

Iteration  6.  x =
      8.5260550201372549134647241469531746689845330015140M-1
      8.5260550201372549134647241469531746689845330015141M-1

Iteration  7.  x =
      8.5260550201372549134647241469531746689845330015140M-1
      8.5260550201372549134647241469531746689845330015141M-1

Iteration  8.  x =
      8.5260550201372549134647241469531746689845330015140M-1
      8.5260550201372549134647241469531746689845330015141M-1

Iteration  9.  x =
      8.5260550201372549134647241469531746689845330015140M-1
      8.5260550201372549134647241469531746689845330015141M-1
```

Example 11.   Count leading digits of powers of 2

```
2^n counts =  301029995 176091267 124938729  96910014  79181253  66946788  57991941  51152528  45757485
```