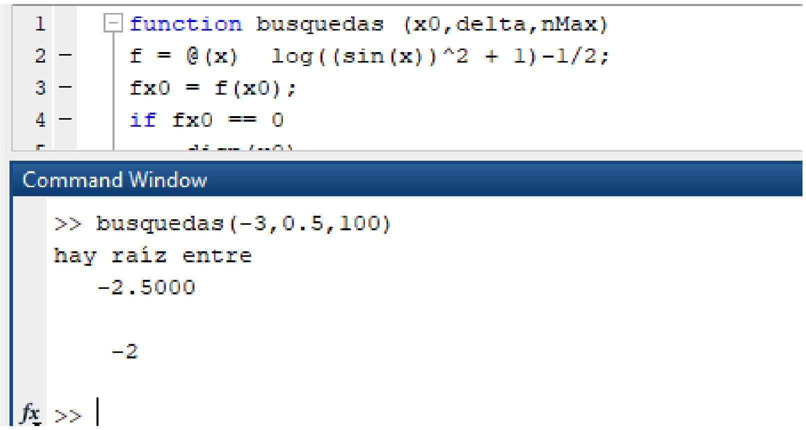
Pruebas de los métodos

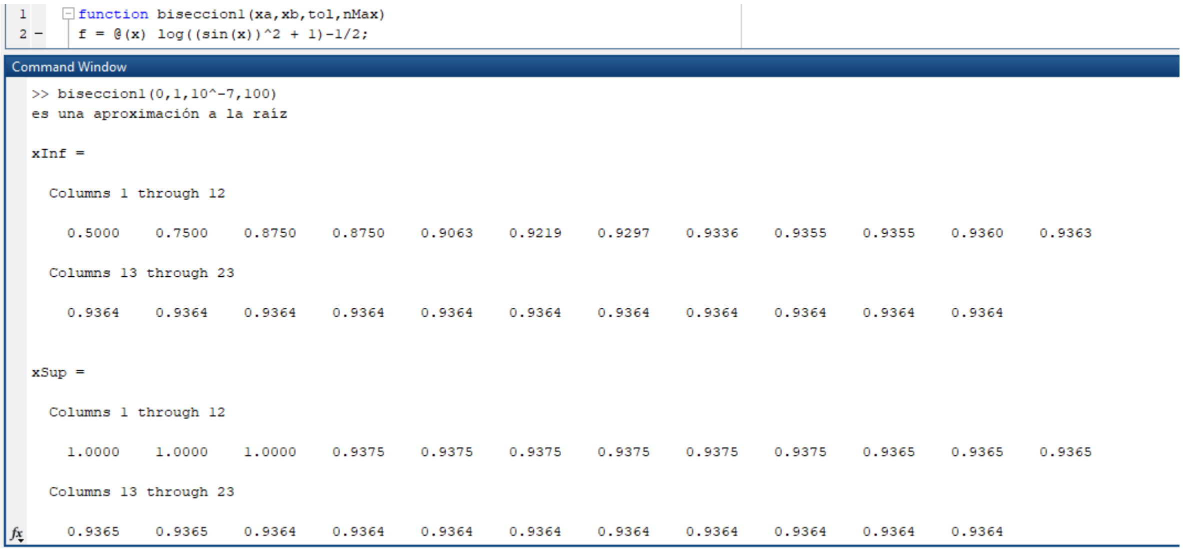
Búsquedas incrementales



```
function busquedas (x0,delta,nMax)
f = @(x)   log((sin(x))^2 + 1)-1/2;
fx0 = f(x0);
if fx0 == 0
```

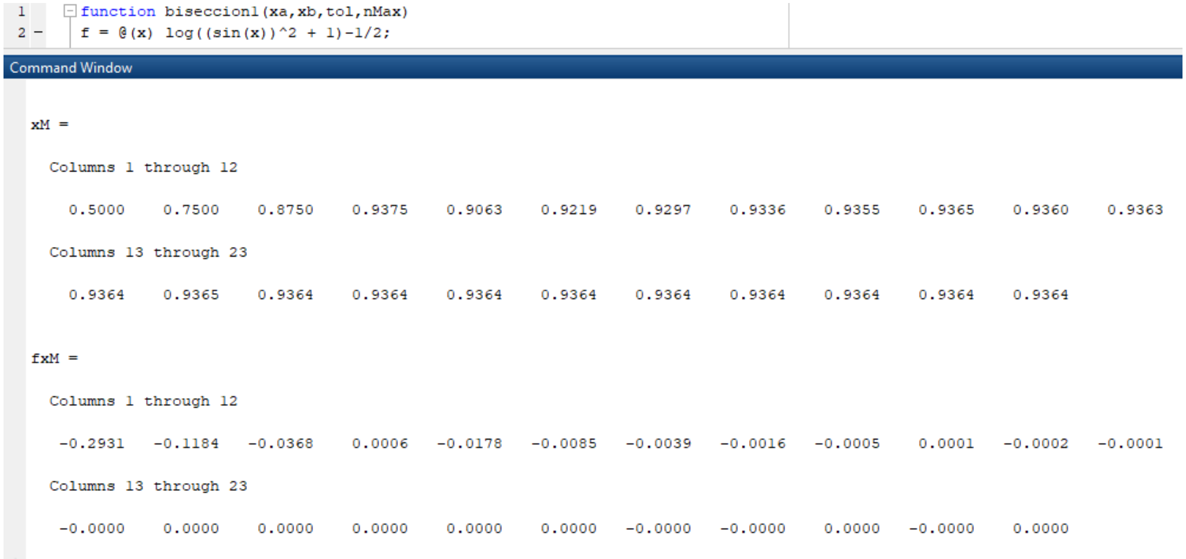
```
>> busquedas(-3,0.5,100)
hay raíz entre
   -2.5000

    -2

>> 
```

Bisección



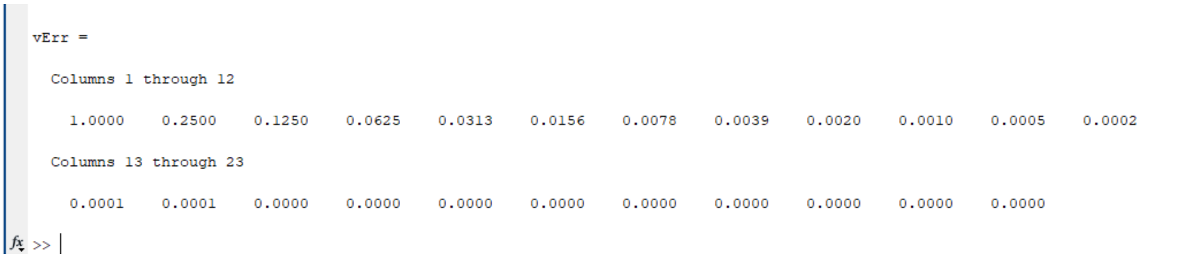
```
function biseccion1(xa,xb,tol,nMax)
f = @(x) log((sin(x))^2 + 1)-1/2;
```

```
>> biseccion1(0,1,10^-7,100)
es una aproximación a la raíz

xInf =

  Columns 1 through 12

    0.5000    0.7500    0.8750    0.8750    0.9063    0.9219    0.9297    0.9336    0.9355    0.9355    0.9360    0.9363

  Columns 13 through 23

    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364


xSup =

  Columns 1 through 12

    1.0000    1.0000    1.0000    0.9375    0.9375    0.9375    0.9375    0.9375    0.9375    0.9365    0.9365    0.9365

  Columns 13 through 23

    0.9365    0.9365    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364
```



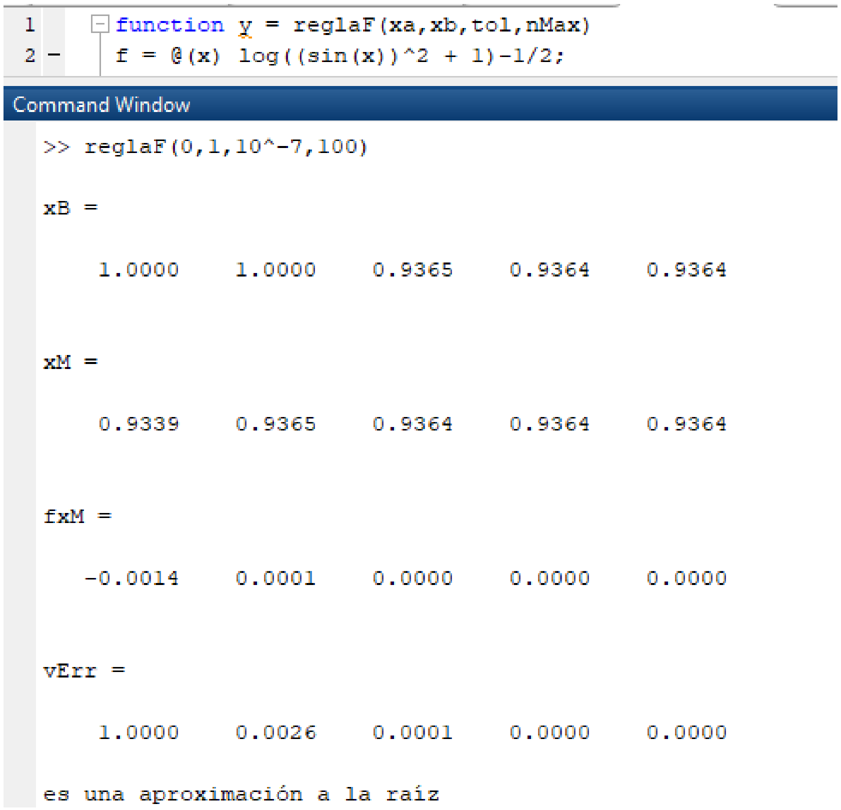
```
function biseccion1(xa,xb,tol,nMax)
f = @(x)  log((sin(x))^2 + 1)-1/2;
```

```
xM =

  Columns 1 through 12

    0.5000    0.7500    0.8750    0.9375    0.9063    0.9219    0.9297    0.9336    0.9355    0.9365    0.9360    0.9363

  Columns 13 through 23

    0.9364    0.9365    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364    0.9364


fxM =

  Columns 1 through 12

   -0.2931   -0.1184   -0.0368    0.0006   -0.0178   -0.0085   -0.0039   -0.0016   -0.0005    0.0001   -0.0002   -0.0001

  Columns 13 through 23

   -0.0000    0.0000    0.0000    0.0000    0.0000    0.0000   -0.0000   -0.0000    0.0000   -0.0000    0.0000
```



```
vErr =

  Columns 1 through 12

    1.0000    0.2500    0.1250    0.0625    0.0313    0.0156    0.0078    0.0039    0.0020    0.0010    0.0005    0.0002

  Columns 13 through 23

    0.0001    0.0001    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

>> 
```

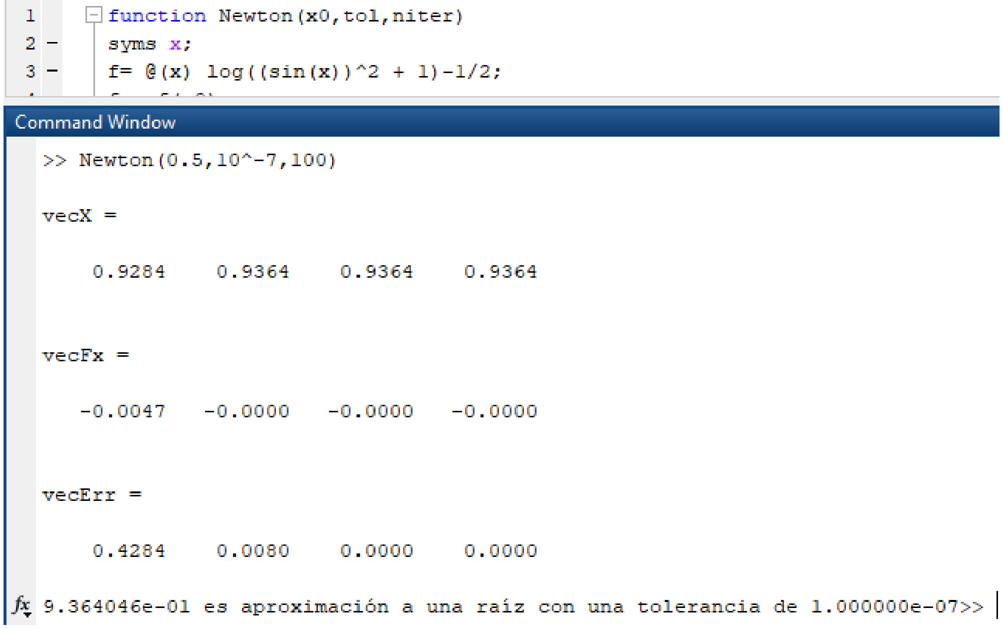
Regla Falsa



```
function y = reglaF(xa,xb,tol,nMax)
f = @(x)  log((sin(x))^2 + 1)-1/2;
```

```
>> reglaF(0,1,10^-7,100)

xB =

    1.0000    1.0000    0.9365    0.9364    0.9364


xM =

    0.9339    0.9365    0.9364    0.9364    0.9364


fxM =

   -0.0014    0.0001    0.0000    0.0000    0.0000


vErr =

    1.0000    0.0026    0.0001    0.0000    0.0000

es una aproximación a la raíz
```

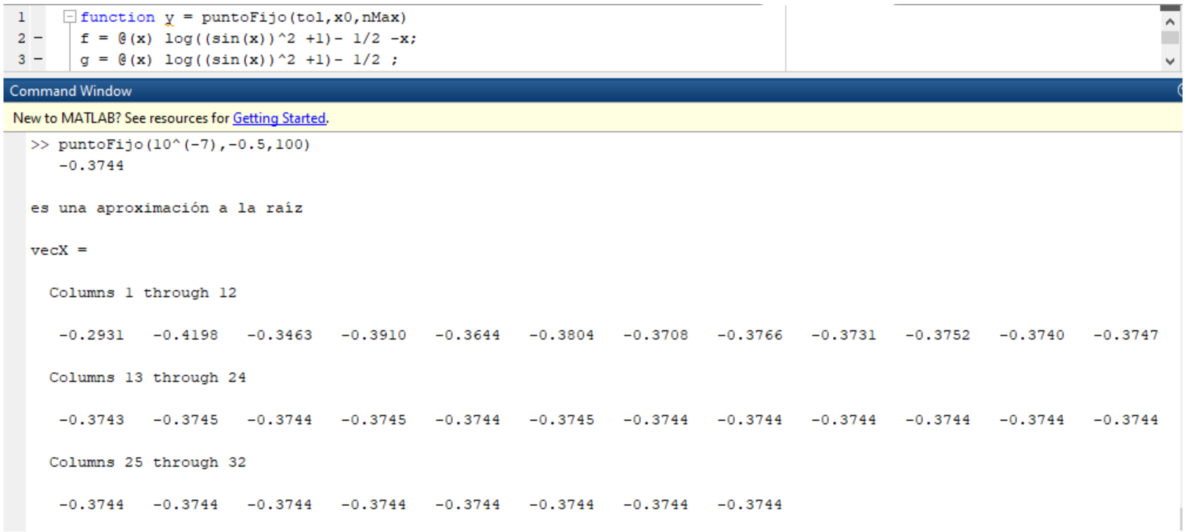
Newton



```
function Newton(x0,tol,niter)
syms x;
f= @(x) log((sin(x))^2 + 1)-1/2;
```

```
>> Newton(0.5,10^-7,100)

vecX =

    0.9284    0.9364    0.9364    0.9364


vecFx =

   -0.0047   -0.0000   -0.0000   -0.0000


vecErr =

    0.4284    0.0080    0.0000    0.0000

9.364046e-01 es aproximación a una raíz con una tolerancia de 1.000000e-07>>
```

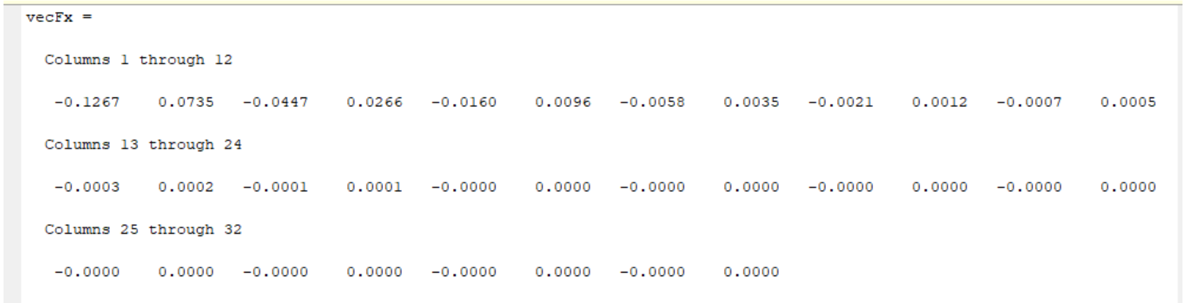
Punto Fijo



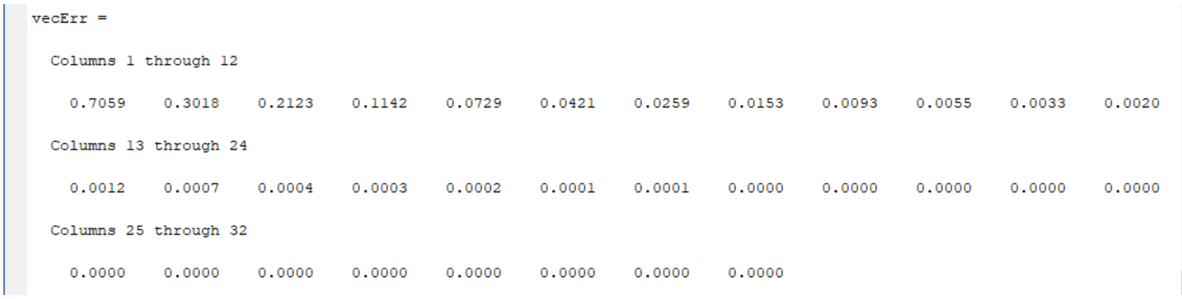
```
function y = puntoFijo(tol,x0,nMax)
f = @(x) log((sin(x))^2 +1)- 1/2 -x;
g = @(x) log((sin(x))^2 +1)- 1/2 ;
```

New to MATLAB? See resources for Getting Started.

```
>> puntoFijo(10^(-7),-0.5,100)
   -0.3744

es una aproximación a la raíz

vecX =

  Columns 1 through 12

   -0.2931   -0.4198   -0.3463   -0.3910   -0.3644   -0.3804   -0.3708   -0.3766   -0.3731   -0.3752   -0.3740   -0.3747

  Columns 13 through 24

   -0.3743   -0.3745   -0.3744   -0.3745   -0.3744   -0.3745   -0.3744   -0.3744   -0.3744   -0.3744   -0.3744   -0.3744

  Columns 25 through 32

   -0.3744   -0.3744   -0.3744   -0.3744   -0.3744   -0.3744   -0.3744   -0.3744
```

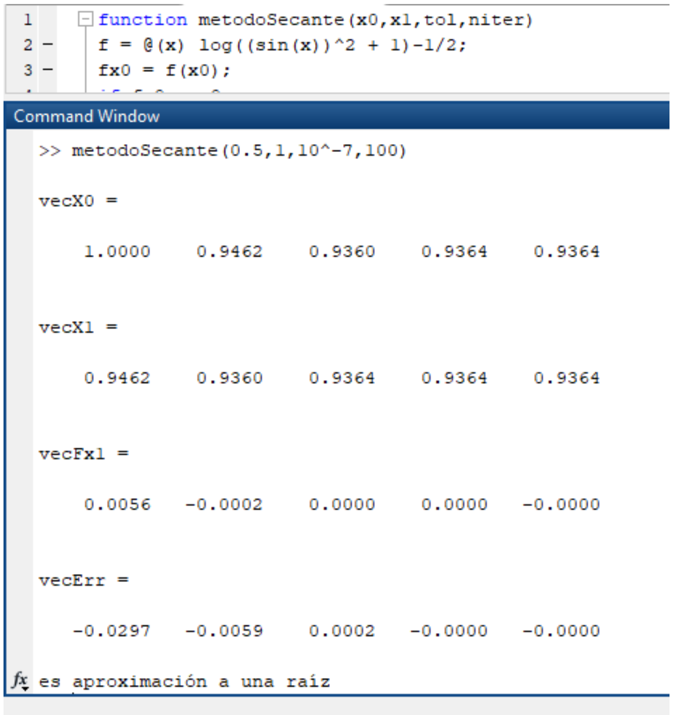


```
vecFx =

  Columns 1 through 12

   -0.1267    0.0735   -0.0447    0.0266   -0.0160    0.0096   -0.0058    0.0035   -0.0021    0.0012   -0.0007    0.0005

  Columns 13 through 24

   -0.0003    0.0002   -0.0001    0.0001   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000

  Columns 25 through 32

   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000
```



```
vecErr =

  Columns 1 through 12

    0.7059    0.3018    0.2123    0.1142    0.0729    0.0421    0.0259    0.0153    0.0093    0.0055    0.0033    0.0020

  Columns 13 through 24

    0.0012    0.0007    0.0004    0.0003    0.0002    0.0001    0.0001    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 25 through 32

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
```

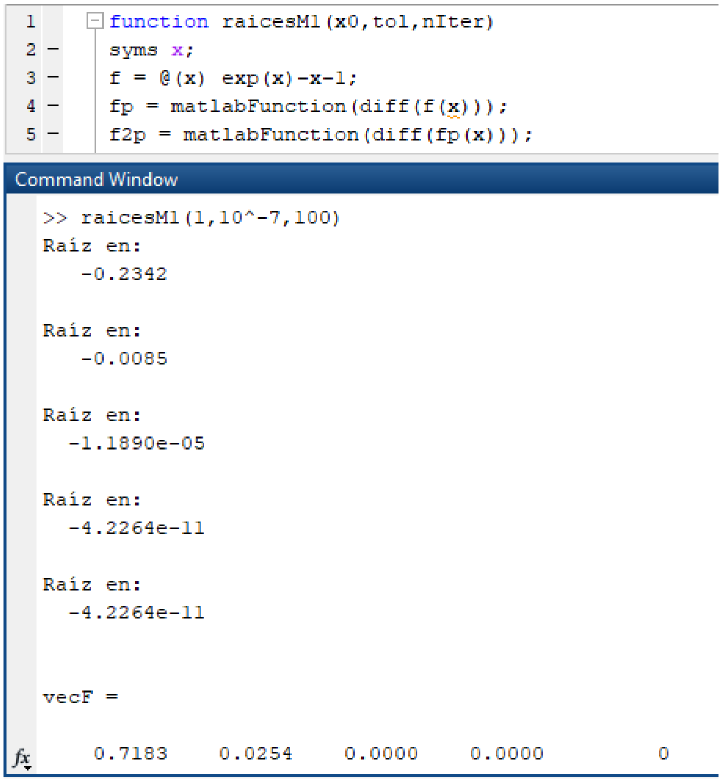
Secante



```
function metodoSecante(x0,x1,tol,niter)
f = @(x) log((sin(x))^2 + 1)-1/2;
fx0 = f(x0);
```

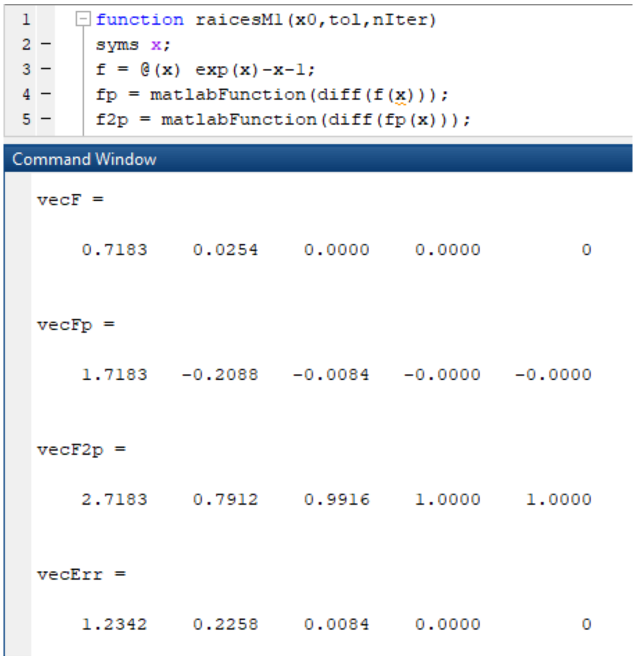
```
>> metodoSecante(0.5,1,10^-7,100)

vecX0 =

    1.0000    0.9462    0.9360    0.9364    0.9364


vecX1 =

    0.9462    0.9360    0.9364    0.9364    0.9364


vecFx1 =

    0.0056   -0.0002    0.0000    0.0000   -0.0000


vecErr =

   -0.0297   -0.0059    0.0002   -0.0000   -0.0000

es aproximación a una raíz
```

Raíces múltiples



```
function raicesM1(x0,tol,nIter)
syms x;
f = @(x) exp(x)-x-1;
fp = matlabFunction(diff(f(x)));
f2p = matlabFunction(diff(fp(x)));
```

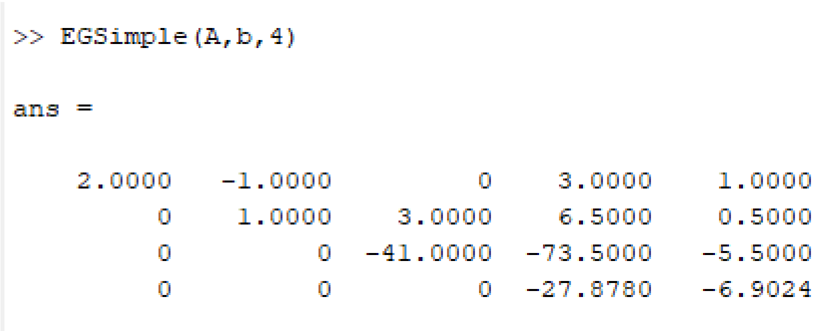
```
>> raicesM1(1,10^-7,100)
Raíz en:
   -0.2342

Raíz en:
   -0.0085

Raíz en:
  -1.1890e-05

Raíz en:
  -4.2264e-11

Raíz en:
  -4.2264e-11


vecF =

    0.7183    0.0254    0.0000    0.0000         0
```



```
function raicesM1(x0,tol,nIter)
syms x;
f = @(x) exp(x)-x-1;
fp = matlabFunction(diff(f(x)));
f2p = matlabFunction(diff(fp(x)));
```

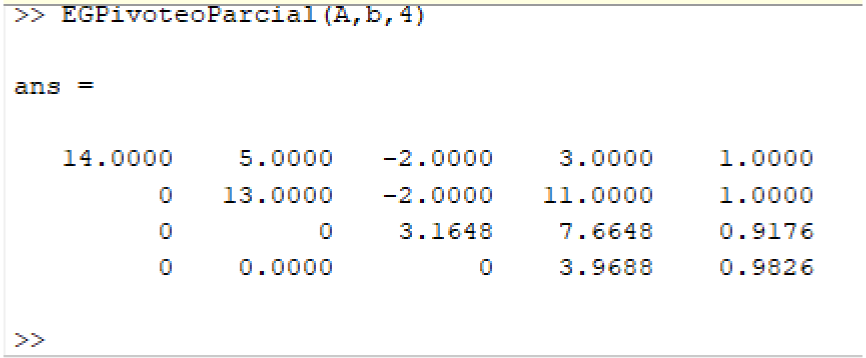
```
vecF =

    0.7183    0.0254    0.0000    0.0000         0


vecFp =

    1.7183   -0.2088   -0.0084   -0.0000   -0.0000


vecF2p =

    2.7183    0.7912    0.9916    1.0000    1.0000


vecErr =

    1.2342    0.2258    0.0084    0.0000         0
```

Eliminación Gaussiana Simple



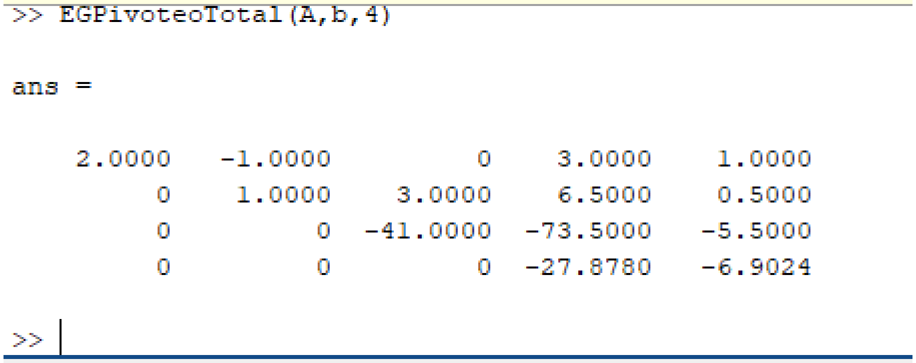
```
>> EGSimple(A,b,4)

ans =

    2.0000   -1.0000         0    3.0000    1.0000
         0    1.0000    3.0000    6.5000    0.5000
         0         0  -41.0000  -73.5000   -5.5000
         0         0         0  -27.8780   -6.9024
```

Eliminación Gaussiana Pivoteo Parcial



```
>> EGPivoteoParcial(A,b,4)

ans =

   14.0000    5.0000   -2.0000    3.0000    1.0000
         0   13.0000   -2.0000   11.0000    1.0000
         0         0    3.1648    7.6648    0.9176
         0    0.0000         0    3.9688    0.9826

>>
```

Eliminación Gaussiana Pivoteo Total



```
>> EGPivoteoTotal(A,b,4)

ans =

    2.0000   -1.0000         0    3.0000    1.0000
         0    1.0000    3.0000    6.5000    0.5000
         0         0  -41.0000  -73.5000   -5.5000
         0         0         0  -27.8780   -6.9024

>>
```