ASSIGNMENT:-3

EECE:-212

NAME: Shaharehar Rahaman Anik

Level: 2

ID No: 201916058

Here are some mathematical problem are solved by MATLAB 2020a.according to the questions. The answers are given bellow:

Write programs to find the real root of the following equations using Newton Raphson Method.

1. xsinx+cosx=0 ; correct to 9 decimal point, near x=3 b) x=e^(-x) ; correct to 9 decimal point, near x=2

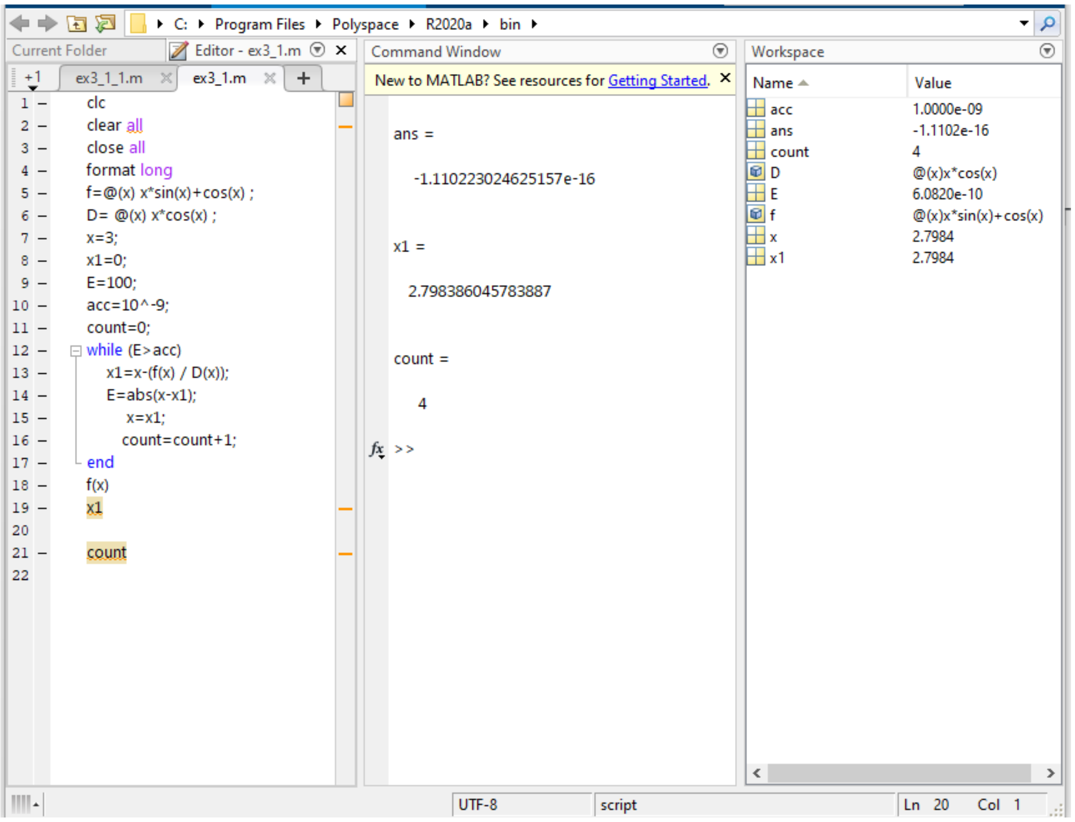
<u>Solution:</u>

Here have to find the real root of the following to equations:

a) xsinx+cosx=0; Near x=3
b) x=e^(-x); Near x=2

Correct to 9 decimal point using Raphson Method. The codes are:

a)

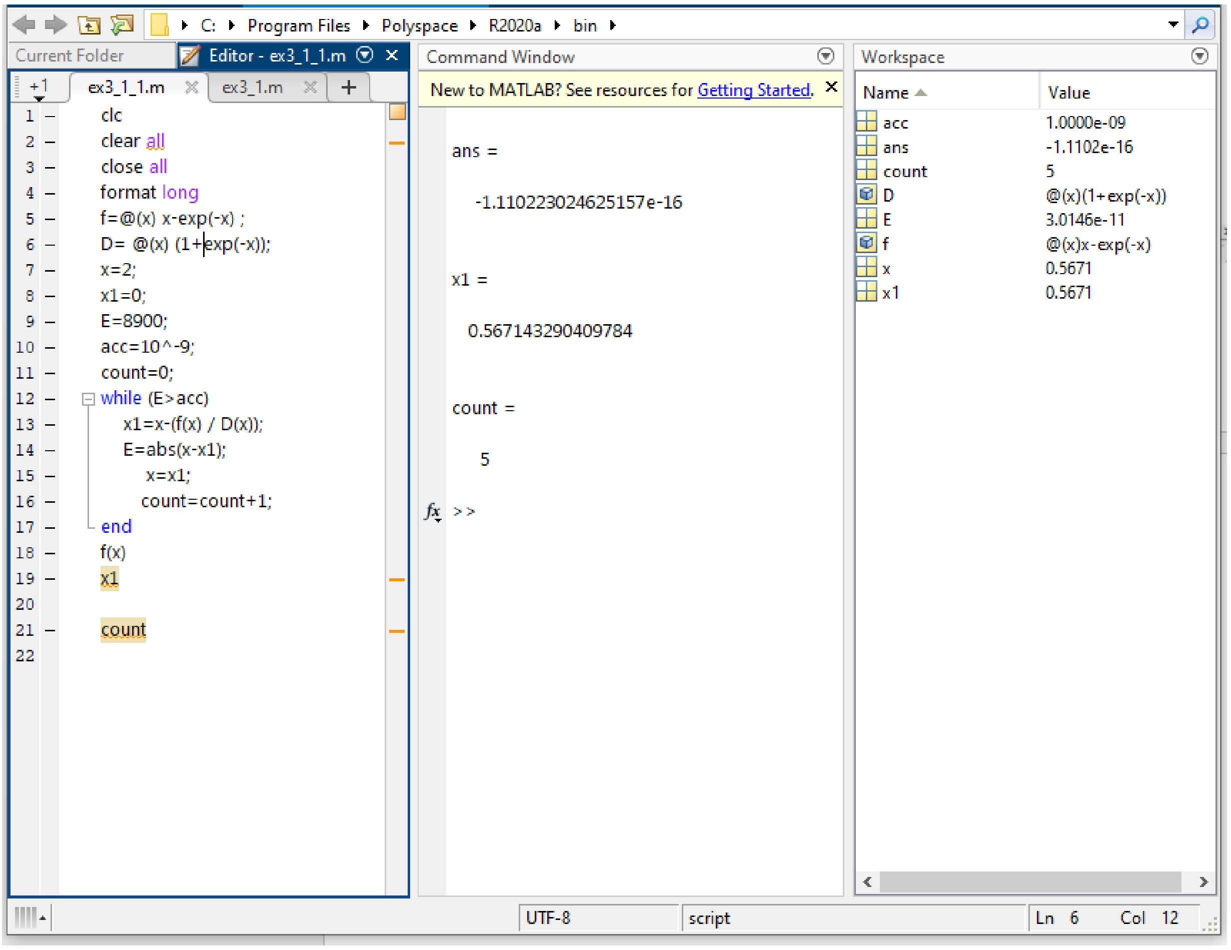

C: ▸ Program Files ▸ Polyspace ▸ R2020a ▸ bin
Current Folder
Editor - ex3_1.m
ex3_1_1.m
ex3_1.m
Command Window
New to MATLAB? See resources for Getting Started.
Workspace
UTF-8
script
Ln 20 Col 1

```
clc
clear all
close all
format long
f=@(x) x*sin(x)+cos(x) ;
D= @(x) x*cos(x) ;
x=3;
x1=0;
E=100;
acc=10^-9;
count=0;
while (E>acc)
   x1=x-(f(x) / D(x));
   E=abs(x-x1);
      x=x1;
     count=count+1;
end
f(x)
x1

count
```

```
ans =

  -1.110223024625157e-16


x1 =

  2.798386045783887


count =

   4

>>
```

Name	Value
acc	1.0000e-09
ans	-1.1102e-16
count	4
D	@(x)x*cos(x)
E	6.0820e-10
f	@(x)x*sin(x)+cos(x)
x	2.7984
x1	2.7984

b)


C: › Program Files › Polyspace › R2020a › bin
Current Folder
Editor - ex3_1_1.m
ex3_1_1.m
ex3_1.m
Command Window
New to MATLAB? See resources for Getting Started.
Workspace
UTF-8
script
Ln 6 Col 12

```
clc
clear all
close all
format long
f=@(x) x-exp(-x) ;
D= @(x) (1+exp(-x));
x=2;
x1=0;
E=8900;
acc=10^-9;
count=0;
while (E>acc)
    x1=x-(f(x) / D(x));
    E=abs(x-x1);
        x=x1;
      count=count+1;
end
f(x)
x1

count
```

```
ans =

  -1.110223024625157e-16


x1 =

   0.567143290409784


count =

     5

>>
```

Name ▲	Value
acc	1.0000e-09
ans	-1.1102e-16
count	5
D	@(x)(1+exp(-x))
E	3.0146e-11
f	@(x)x-exp(-x)
x	0.5671
x1	0.5671

<u>The answer is</u>:

a) 2.798386045783887

b) 0.567143290409784

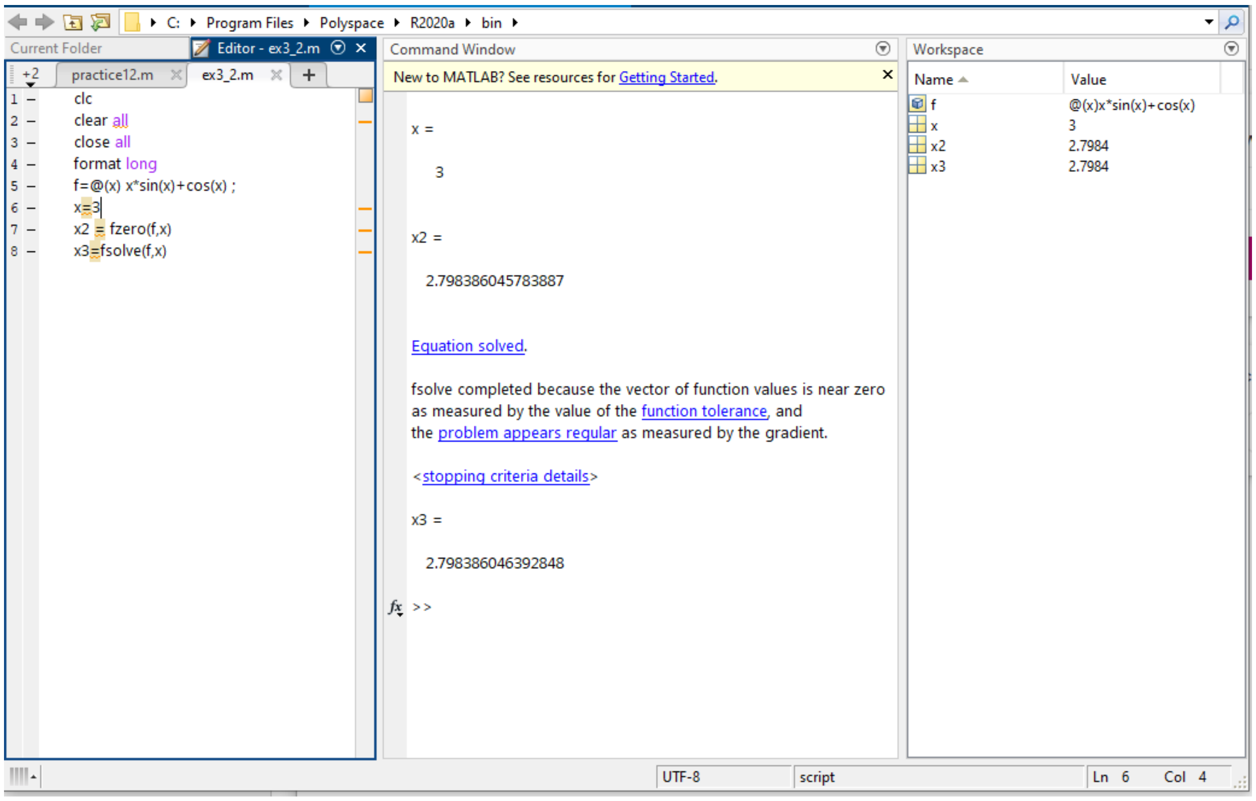
2. Solve 1(a) and 1(b) using 'fzero' and 'fsolve' built in function of MATLAB

<u>Solution:</u>

Here have to use fzero & fsolve built in functions of MATLAB in the equations:

a) xsinx+cosx=0;
b) x=e^(-x);

a)



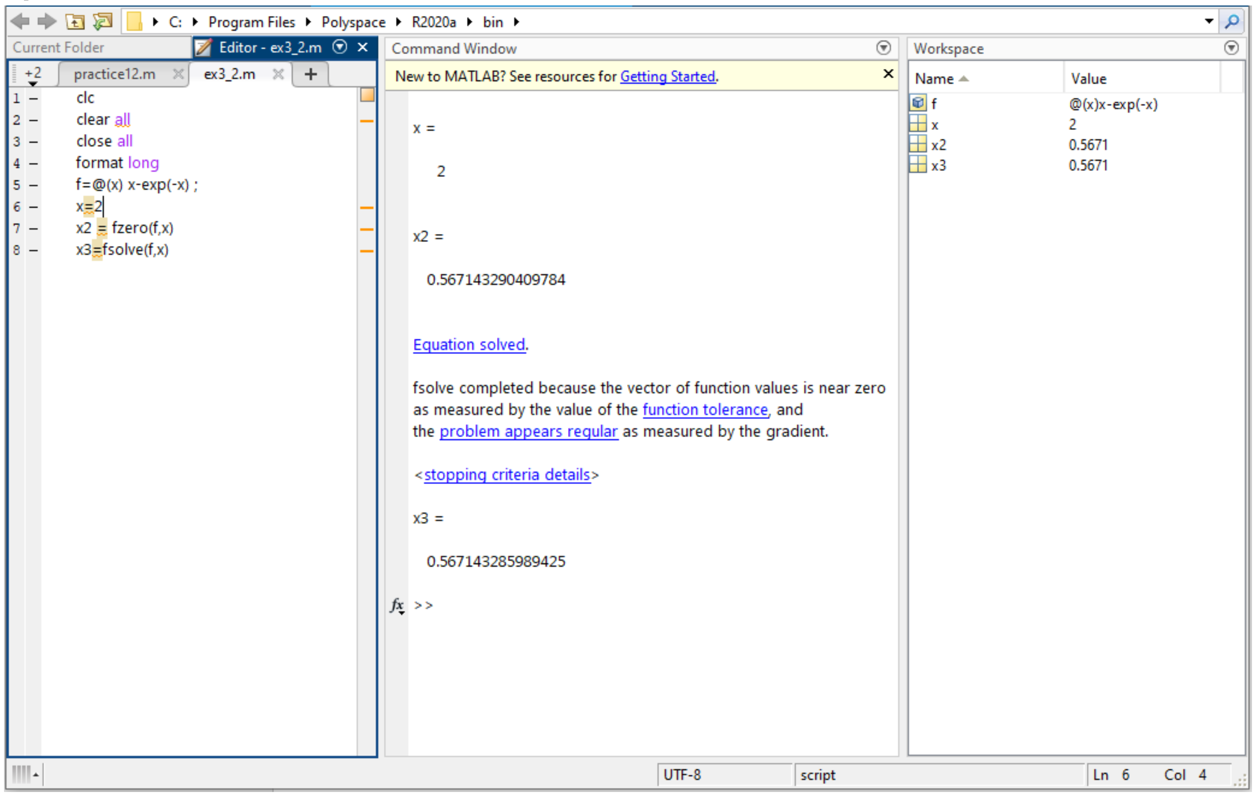
```
clc
clear all
close all
format long
f=@(x) x*sin(x)+cos(x) ;
x=3
x2 = fzero(f,x)
x3=fsolve(f,x)
```

New to MATLAB? See resources for Getting Started.

```
x =

     3


x2 =

   2.798386045783887


Equation solved.

fsolve completed because the vector of function values is near zero
as measured by the value of the function tolerance, and
the problem appears regular as measured by the gradient.

<stopping criteria details>

x3 =

   2.798386046392848

>>
```

Name	Value
f	@(x)x*sin(x)+cos(x)
x	3
x2	2.7984
x3	2.7984

b)



```
clc
clear all
close all
format long
f=@(x) x-exp(-x) ;
x=2
x2 = fzero(f,x)
x3=fsolve(f,x)
```

New to MATLAB? See resources for Getting Started.

```
x =

     2


x2 =

   0.567143290409784


Equation solved.

fsolve completed because the vector of function values is near zero
as measured by the value of the function tolerance, and
the problem appears regular as measured by the gradient.

<stopping criteria details>

x3 =

   0.567143285989425

>>
```

Name	Value
f	@(x)x-exp(-x)
x	2
x2	0.5671
x3	0.5671

<u>The solutions are:</u>

a)

Using fzero: 2.798386045783887

Using fsolve: 2.798386046392848

b)

Using fzero: 0.567143290409784

Using fsolve: 0.567143285989425

3. What is the number of iteration required to find the root of problem 1(a), 1(b). Compare its value with that of obtained from the method of False Position to solve the same equation at same condition. Comment on which method is better from the view of number of iteration.

<u>*Solution:*</u>

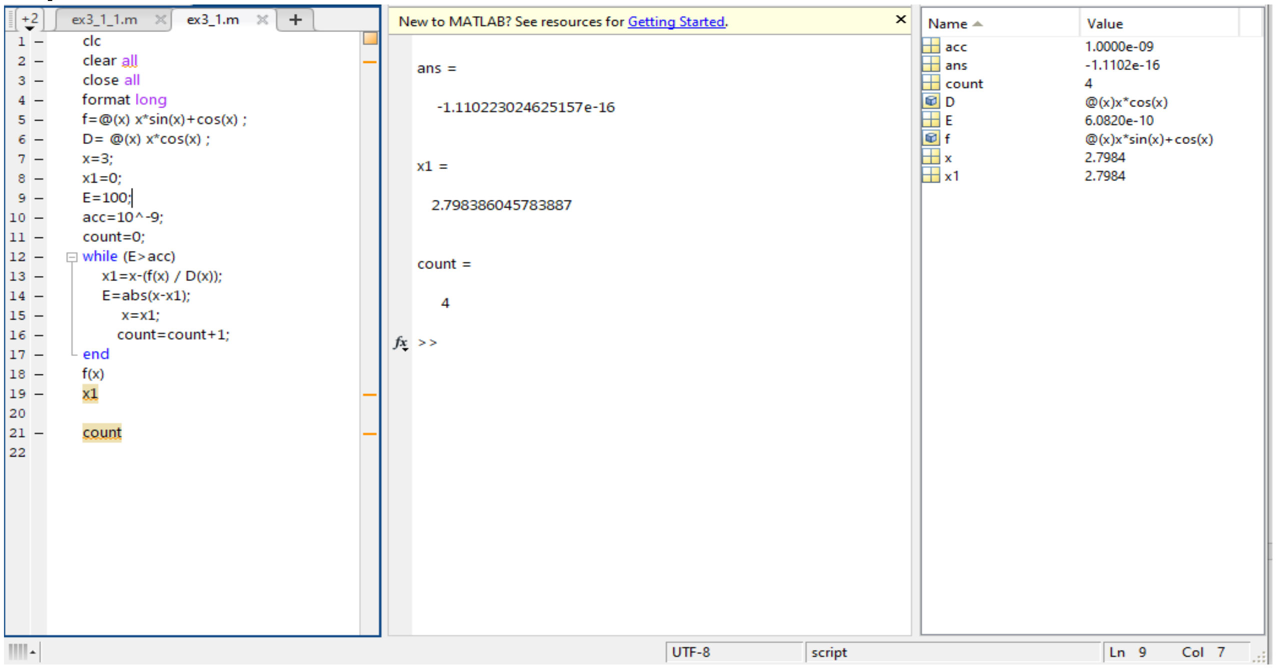
Here it is asked to find out the iteration number of 1(a) & 1(b) .And have to compare with the method of false position of the same equations in the view of iteration number

.

Now the equation 1(a) & 1(b) are solved according to Raphson Method and the method of false position. The programs are given bellow:

1(a) xsinx+cosx=0:

Raphson Method:



ex3_1_1.m ex3_1.m

```matlab
clc
clear all
close all
format long
f=@(x) x*sin(x)+cos(x) ;
D= @(x) x*cos(x) ;
x=3;
x1=0;
E=100;
acc=10^-9;
count=0;
while (E>acc)
    x1=x-(f(x) / D(x));
    E=abs(x-x1);
      x=x1;
      count=count+1;
end
f(x)
x1

count
```

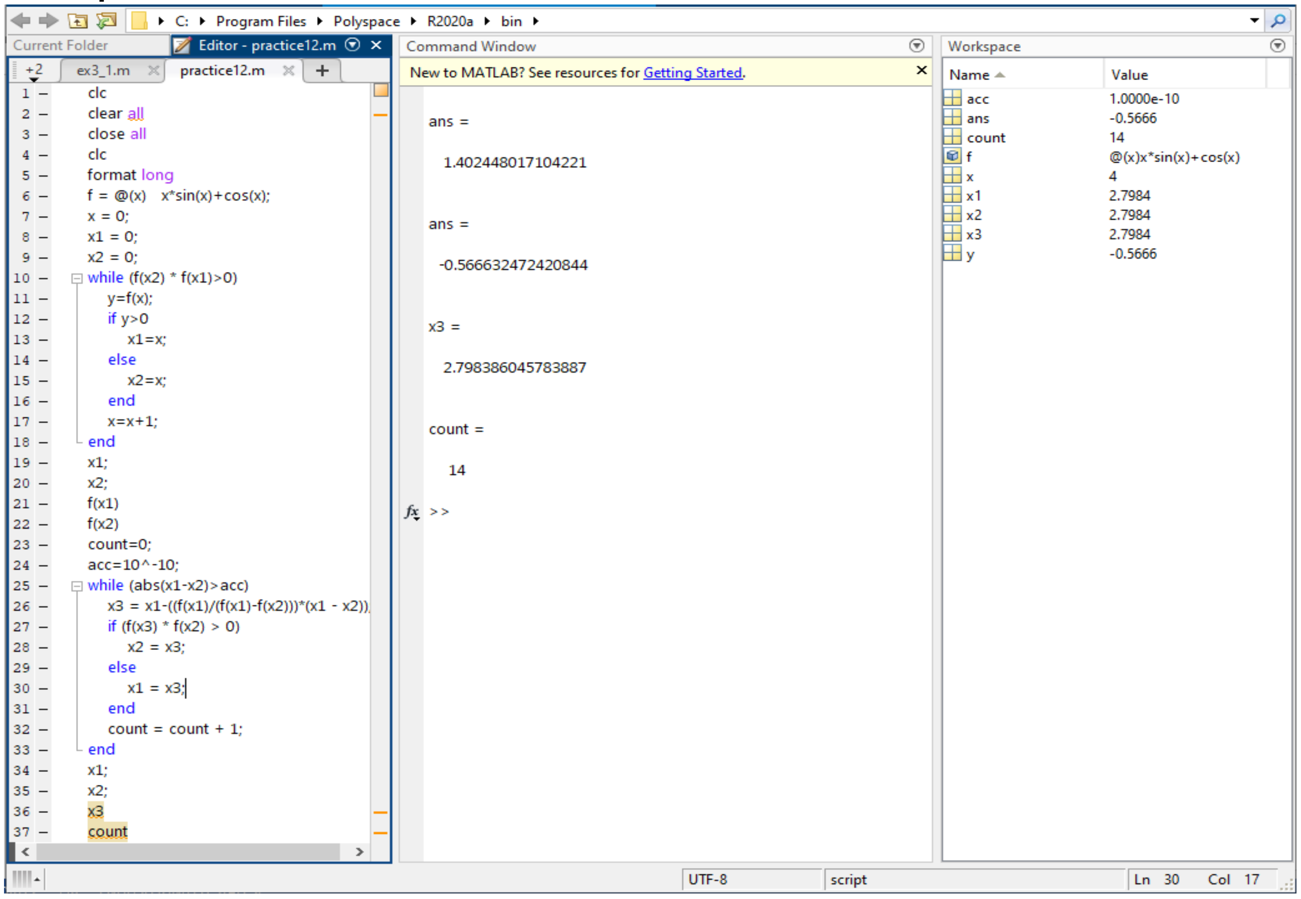
New to MATLAB? See resources for Getting Started.

```
ans =

  -1.110223024625157e-16


x1 =

   2.798386045783887


count =

     4

>>
```

Name	Value
acc	1.0000e-09
ans	-1.1102e-16
count	4
D	@(x)x*cos(x)
E	6.0820e-10
f	@(x)x*sin(x)+cos(x)
x	2.7984
x1	2.7984

UTF-8 script Ln 9 Col 7

False position method:



C: ▸ Program Files ▸ Polyspace ▸ R2020a ▸ bin ▸

Current Folder | Editor - practice12.m | Command Window | Workspace

ex3_1.m practice12.m

```matlab
clc
clear all
close all
clc
format long
f = @(x)   x*sin(x)+cos(x);
x = 0;
x1 = 0;
x2 = 0;
while (f(x2) * f(x1)>0)
    y=f(x);
    if y>0
        x1=x;
    else
        x2=x;
    end
    x=x+1;
end
x1;
x2;
f(x1)
f(x2)
count=0;
acc=10^-10;
while (abs(x1-x2)>acc)
    x3 = x1-((f(x1)/(f(x1)-f(x2)))*(x1 - x2));
    if (f(x3) * f(x2) > 0)
        x2 = x3;
    else
        x1 = x3;
    end
    count = count + 1;
end
x1;
x2;
x3
count
```

New to MATLAB? See resources for Getting Started.

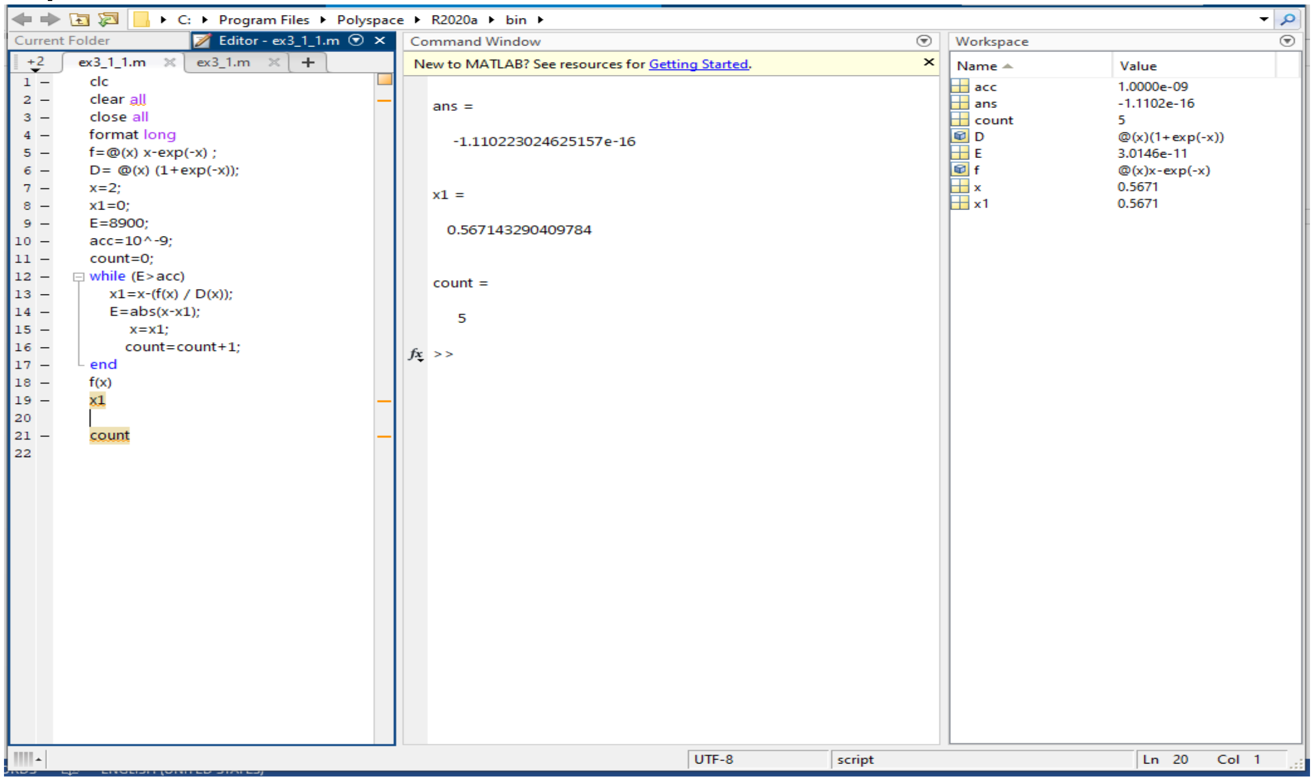
```
ans =

   1.402448017104221


ans =

  -0.566632472420844


x3 =

   2.798386045783887


count =

    14

>>
```

Name	Value
acc	1.0000e-10
ans	-0.5666
count	14
f	@(x)x*sin(x)+cos(x)
x	4
x1	2.7984
x2	2.7984
x3	2.7984
y	-0.5666

UTF-8 script Ln 30 Col 17

1(b) x=e^ (-x):

Raphson Method:



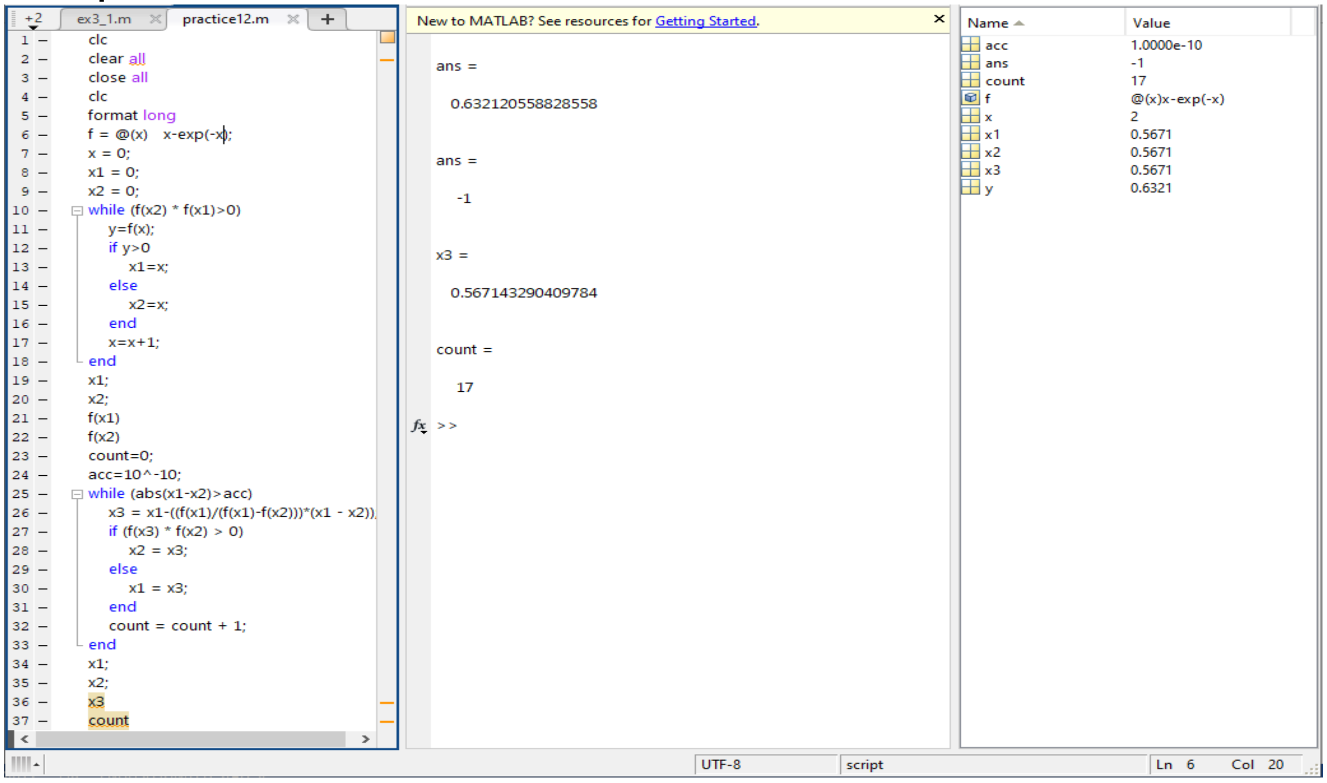
C: ▸ Program Files ▸ Polyspace ▸ R2020a ▸ bin ▸
Current Folder
Editor - ex3_1_1.m
ex3_1_1.m
ex3_1.m
Command Window
Workspace
New to MATLAB? See resources for Getting Started.
UTF-8
script
Ln 20 Col 1

```
clc
clear all
close all
format long
f=@(x) x-exp(-x) ;
D= @(x) (1+exp(-x));
x=2;
x1=0;
E=8900;
acc=10^-9;
count=0;
while (E>acc)
    x1=x-(f(x) / D(x));
    E=abs(x-x1);
       x=x1;
     count=count+1;
end
f(x)
x1

count
```

```
ans =

  -1.110223024625157e-16


x1 =

   0.567143290409784


count =

     5

>>
```

Name	Value
acc	1.0000e-09
ans	-1.1102e-16
count	5
D	@(x)(1+exp(-x))
E	3.0146e-11
f	@(x)x-exp(-x)
x	0.5671
x1	0.5671

False position method:



```
clc
clear all
close all
clc
format long
f = @(x)  x-exp(-x);
x = 0;
x1 = 0;
x2 = 0;
while (f(x2) * f(x1)>0)
    y=f(x);
    if y>0
       x1=x;
    else
       x2=x;
    end
    x=x+1;
end
x1;
x2;
f(x1)
f(x2)
count=0;
acc=10^-10;
while (abs(x1-x2)>acc)
    x3 = x1-((f(x1)/(f(x1)-f(x2)))*(x1 - x2));
    if (f(x3) * f(x2) > 0)
       x2 = x3;
    else
       x1 = x3;
    end
    count = count + 1;
end
x1;
x2;
x3
count
```

```
ans =

   0.632120558828558


ans =

    -1


x3 =

   0.567143290409784


count =

    17

>>
```

Name	Value
acc	1.0000e-10
ans	-1
count	17
f	@(x)x-exp(-x)
x	2
x1	0.5671
x2	0.5671
x3	0.5671
y	0.6321

The iteration numbers are:

1(a)

Raphson Method: 4

False position method: 14

1(b)

Raphson Method: 5

False position method: 17

if we compare these iteration number we can see that for both 1(a) &1(b) every time raphson method iterates less than false position method.

Comment:

In the view of iteration number raphson method is better than false position. Its needs short time to solve. And it covers a few loops to bring out the result.

4. Write the advantage and disadvantage of Newton Raphson method.

Solution:

Here it's wanted to know about the advantages and disadvantages of Newton Raphson method. These are given bellow:

Advantages:

1. One of the fastest methods which converges to root quickly.
2. Converges on the root quadratically that is rate of convergence is 2.
3. As we go near to root, number of significant digits approximately double with each step.

4. It’s make the method useful to get precise results for a root which was previously obtained from some other convergence method.
5. Easy to convert to multiple dimension.

Disadvantages:

1. We must find the derivative to use this method.
2. Poor global convergence properties.
3. The method is very expensive - It needs the function evaluation and then the derivative evaluation
4. If the tangent is parallel or nearly parallel to the x-axis, then the method does not converge
5. Dependent on initial guess.
   - May be too far from local root.
   - May encounter a zero derivative
   - May loop indefinitely

5. For the function f(x) =x^3-3*x-1, find the root near x=2 using Newton Raphson Method by hand calculation. Show three iterations.

Solution:

Here the function is **f(x) =x^3-3*x-1.** Have to find out the root near x=2 using Newton Raphson Method by hand calculation.

First three iterations are shown:

Now,

f(x) =x^3-3x-1,

The differentiation of the function is:

diff (f(x)) = 3x^2-3.

Newton raphson method is:

$$x' = x - \frac{f(x)}{diff(f(x))}$$

Or,

$$x' = x - \frac{x^3 - 3x - 1}{3x^2 - 3}$$

So,

1st iteration is:

Near to x=2.

$$x1' = 2 - \frac{2^3 - 3 * (2) - 1}{3 * (2)^2 - 3}$$

Or,

$$x1' = 1.888888888888889$$

2nd iteration is:

For this 2nd iteration,

x=x1'.

That's mean, x= 1.888888888888889

So,

$$x2' = 1.888888888888889 - \frac{(1.888888888888889)^3 - 3 * (1.888888888888889) - 1}{3 * (1.888888888888889)^2 - 3}$$

Or,

$$x2' = \ 1.879451566951567$$

3rd iteration is:

x=x2'.

That's mean, x= 1.879451566951567

So,

$$x3' = 1.879451566951567 - \frac{(1.879451566951567)^3 - 3*(1.879451566951567) - 1}{3*(1.879451566951567)^2 - 3}$$

Or,

$$x3' = \ 1.879385244836671$$

So, the three iterations are:

X1’= 1.888888888888889

X2’=1.879451566951567

X3’=1.879385244836671