**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**

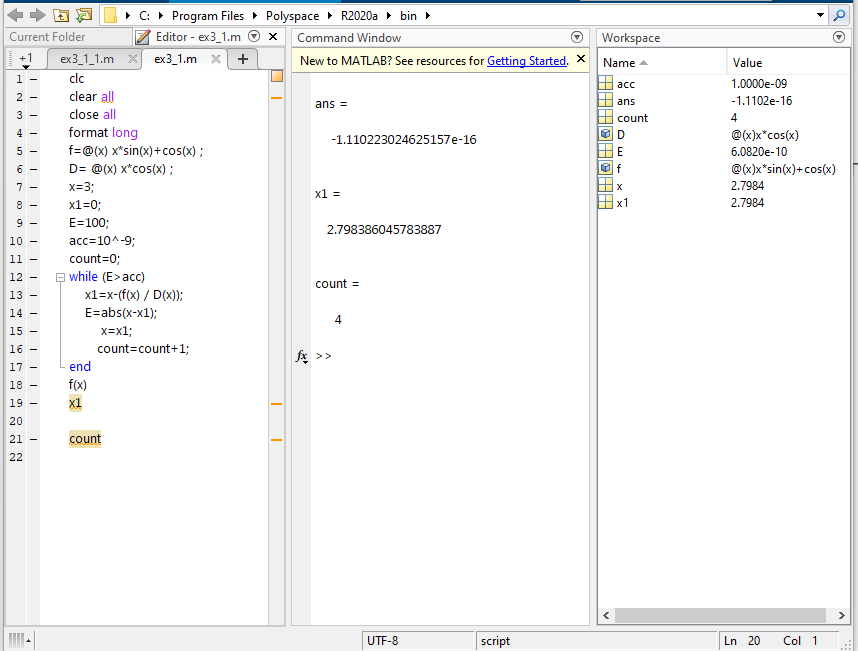
**Solution:**

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

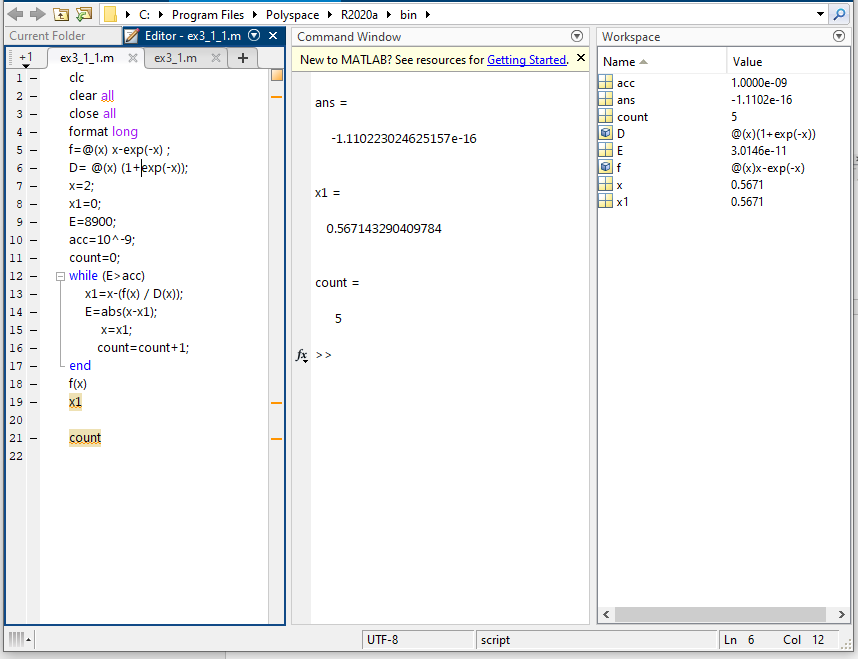
1. **xsinx+cosx=0;** Near x=3
2. **x=e^(-x);** Near x=2

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

**a)**



**b)**



**The answer is:**

**a) 2.798386045783887**

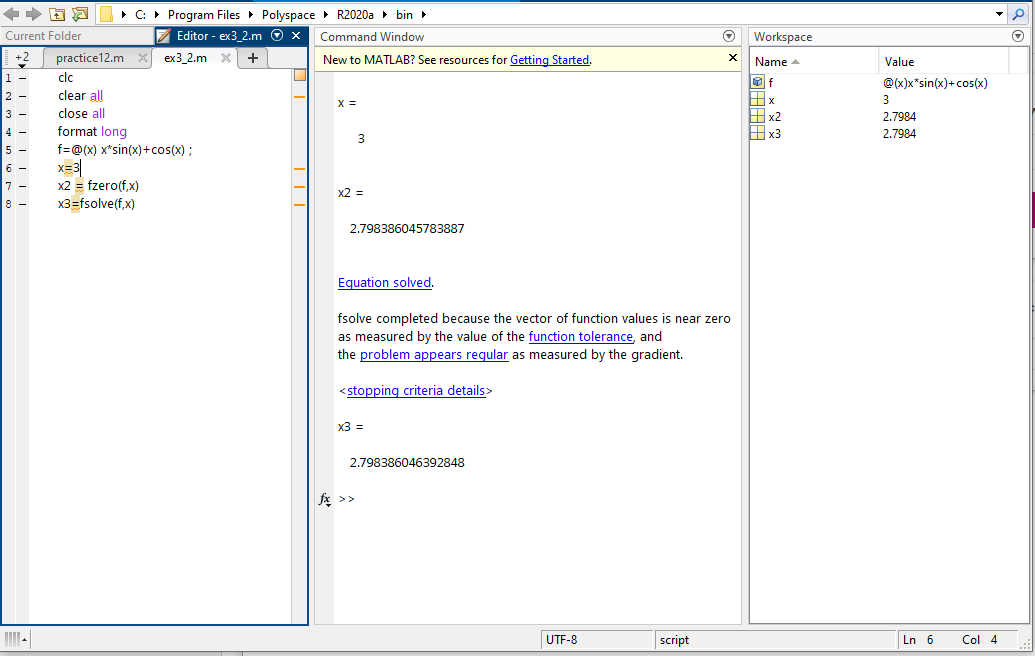
**b) 0.567143290409784**

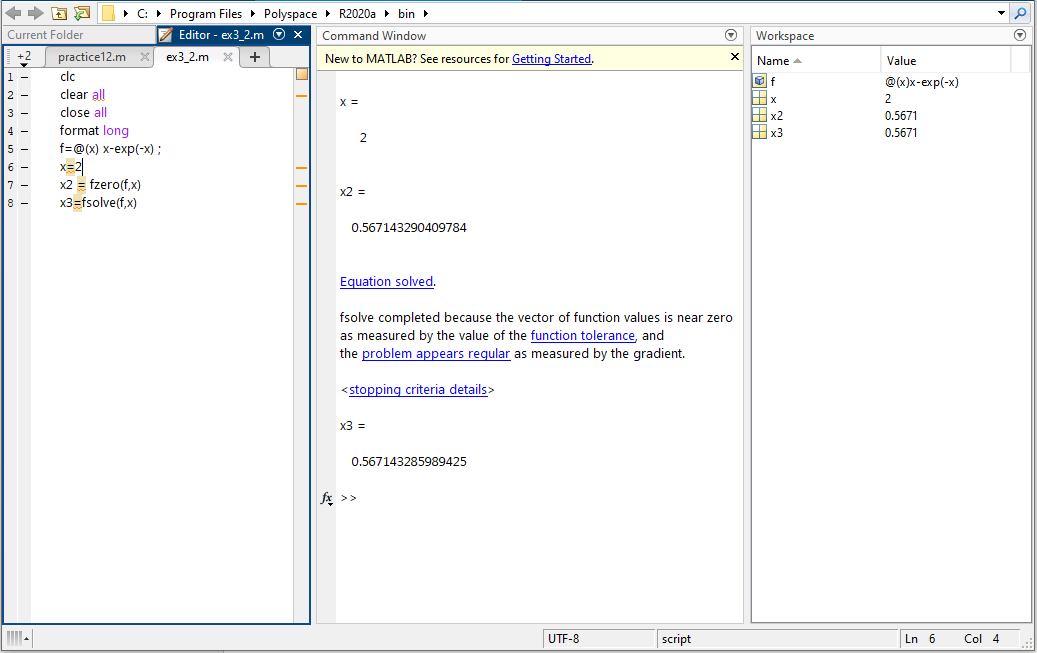
**2. Solve 1(a) and 1(b) using 'fzero' and 'fsolve' built in function of MATLAB**

**Solution:**

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

1. **xsinx+cosx=0;**
2. **x=e^(-x);**

**a)**

**b)** 

**The solutions are:**

**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.**

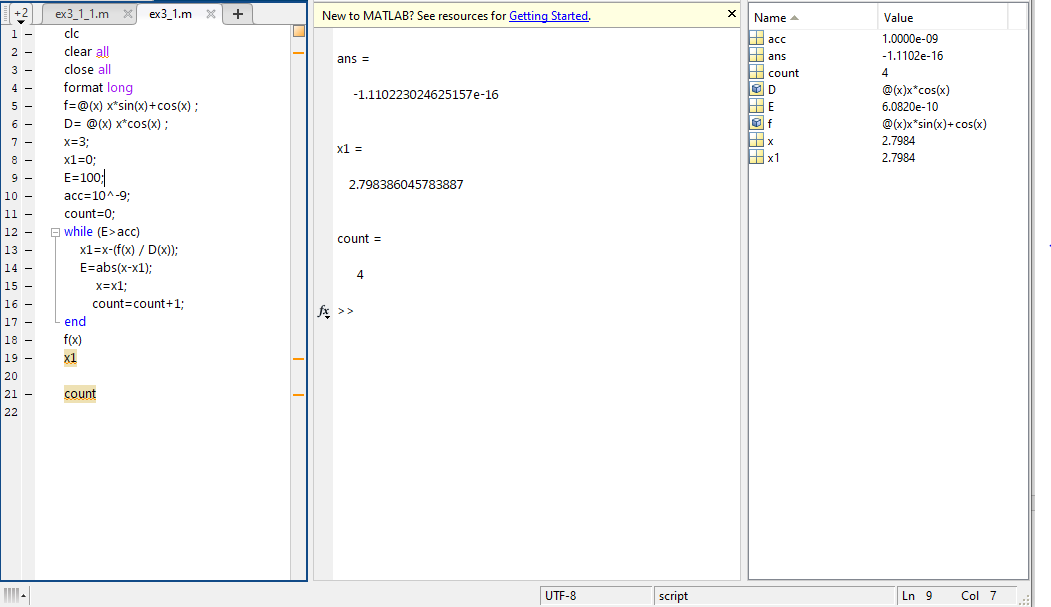
**Solution:**

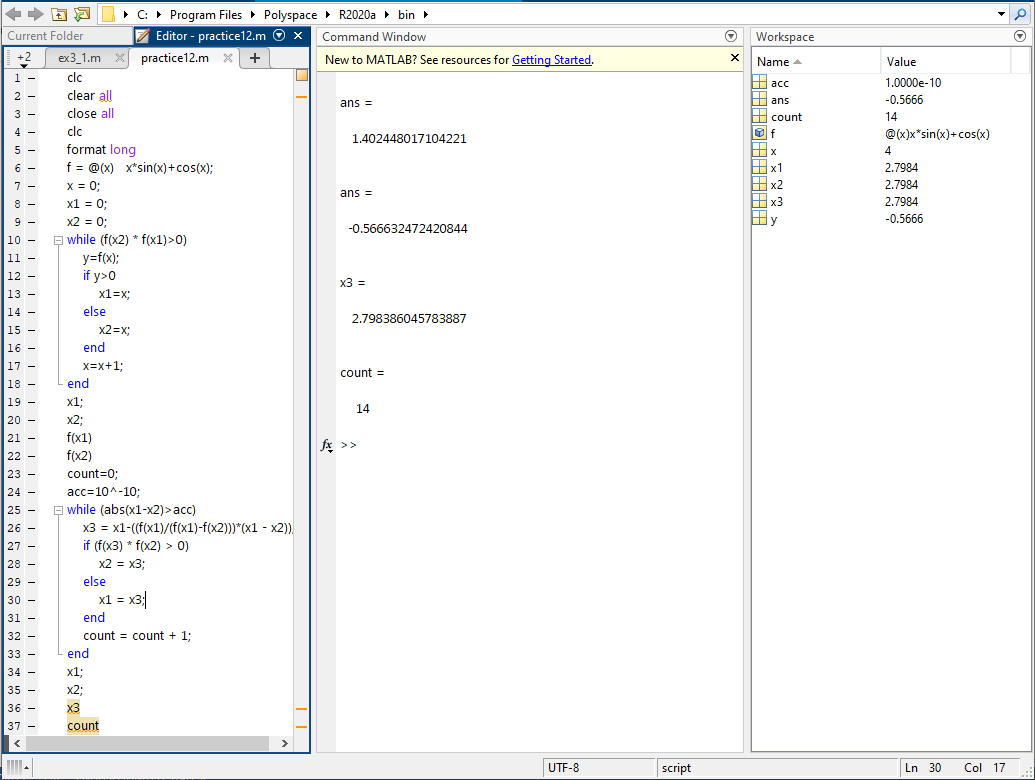
Here it is asked to find out the iteration number of 1(a) & 1(b) .And have to compare with the method of false position of the same equations in the view of iteration number

.

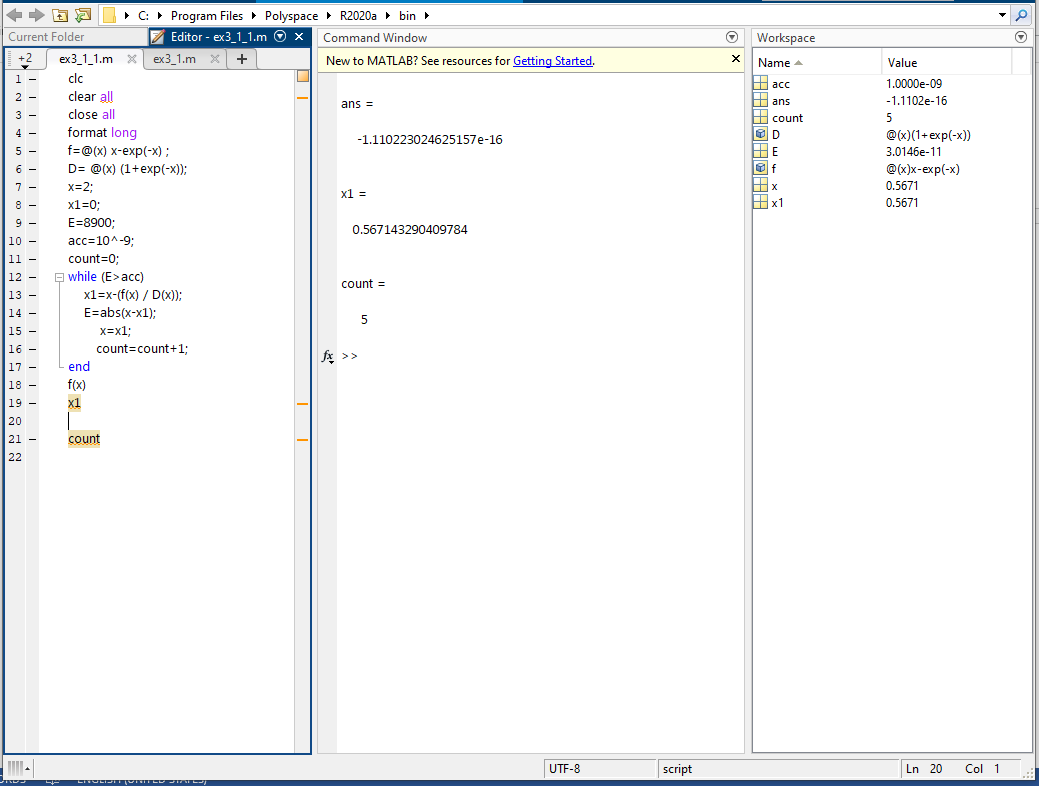
Now the equation 1(a) & 1(b) are solved according to Raphson Method and the method of false position. The programs are given bellow:

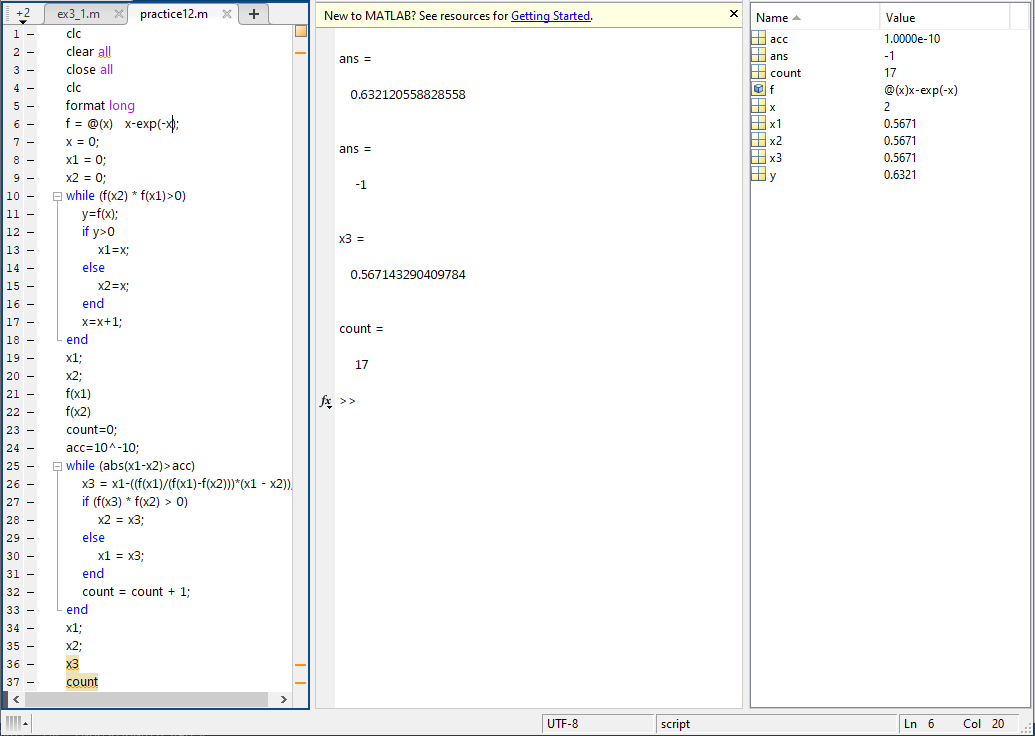
**1(a) xsinx+cosx=0:**

**Raphson Method:**

**False position method:** 

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

**Raphson Method:** 

**False position method:** 

**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) =x3-3\*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) =x­­3-3x-1,

The differentiation of the function is:

diff (f(x)) = 3x2-3.

**Newton raphson method is:**

Or,

So,

**1st iteration is:**

Near to x=2.

Or,

**2nd iteration is:**

For this 2nd iteration,

x=x1’.

That’s mean, x= 1.888888888888889

So,

Or,

**3rd iteration is:**

x=x2’.

That’s mean, x= 1.879451566951567

So,

Or,

**So, the three iterations are:**

**X1’= 1.888888888888889**

**X2’=1.879451566951567**

**X3’=1.879385244836671**