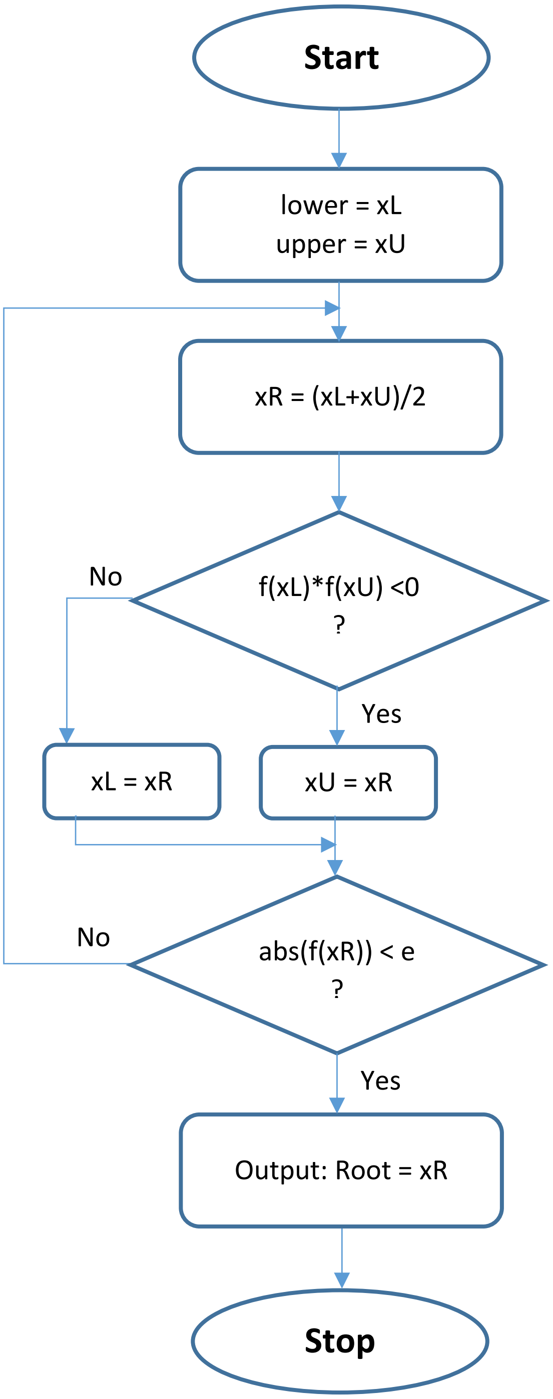
## QUESTION

The equation that gives the depth x to which the ball is submerged under water is given by

$$f(x) = x^3 - 0.165x^2 + 3.993$$

Use the bisection method which is the root-finding method in numerical analysis to find the depth x to which the ball is submerged under water. Conduct 20 iterations to estimate the root of the above equation. Develop flowchart or pseudocode and write a Python program to solve this question. Hence, plot the graph of $f(x)$ by using matplotlib.

## Flowchart for Bisection Method



## Pseudocode for Bisection Method

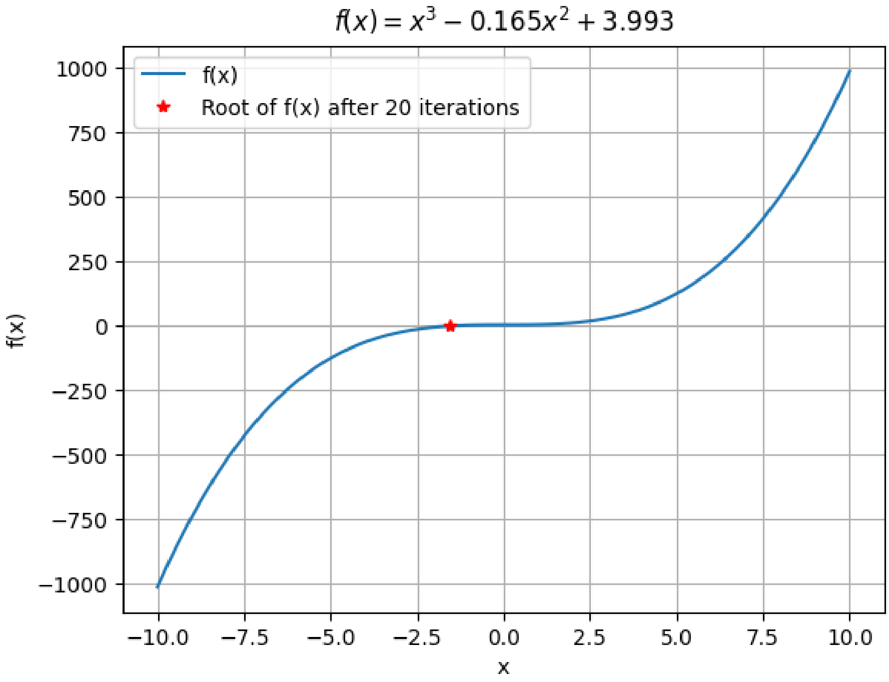
```
1. Start
2. Define function f(x)
3. Input
        a. Lower and Upper guesses xL and xU
        b. Tolerable error as e
        c. Max. number of iterations = 20

4. Check if these values bracket the root or not?
If f(xL)*f(xU) > 0
        print "The given guesses do not bracket the root."
        goto 3
  End If

5. Begin iterations for bisection method

Do
xR = (xL+xU)/2

        If f(xL)*f(xU) < 0:
                xU = xR
        Else
                xL = xR
        End If
  Condition = abs(f(xR)) < e

6. Print root as xR
7. Stop
```

**Codes:**

```
import numpy as np
import matplotlib.pyplot as plt

# Define the function whose roots are required
def f(x):
    return x**3-(0.165*x**2)+3.993

# Implementing Bisection Method
def bisection(xL, xU, e, maxiter):
    step = 1
    print('\n ****BISECTION METHOD IMPLEMENTATION***** ')
    condition = True
    while condition and step <= maxiter:
        xR = (xL + xU) / 2
        print('\nIteration-%d, xR = %0.6f and f(xR) = %0.6f' %
(step, xR, f(xR)))

        if f(xL) * f(xR) < 0:
            xU = xR
        else:
            xL = xR
        condition = abs(f(xR)) > e
        print('Required Root is : %0.8f' % xR)
        step = step + 1
        if step > maxiter:
            print("\nYou have reached the maximum
iterations.")
            print('Required Root after 20 iterations is : ' +
str(xR))
    return xR, step

# Input Initial Guess, Tolerance and Max Iterations
xL = float(input('First Guess: '))
xU = float(input('Second Guess: '))
e = float(input('Tolerable Error: '))
maxiter = 20

# Checking Correctness of initial guess values and bisecting
if f(xL) * f(xU) > 0.0:
    print('Given guess values do not bracket the root.')
    print('Try again with different guess values.')
else:
    xR, step = bisection(xL, xU, e, maxiter)
```

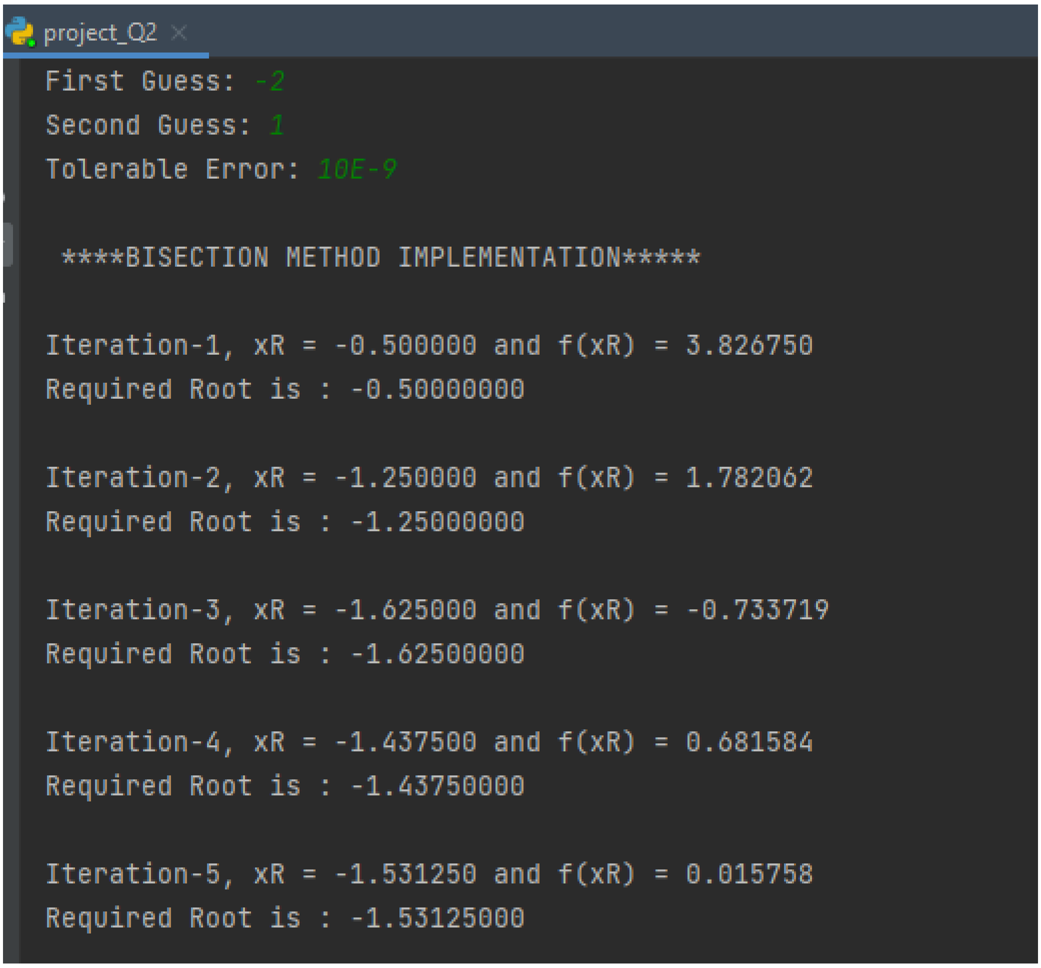
```
# Plot the f(x) and root
x = np.linspace(-10, 10, 80)
plt.figure()
plt.plot(x, f(x), label = 'f(x)')
plt.plot(xR, f(xR), 'r*', label=('Root of f(x) after
'+str(step-1)+' iterations'))
plt.title(r'$f(x) = x^{3}-0.165x^{2}+3.993$')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.legend()
plt.grid()
plt.show()
```

### Graph of $f(x)$



**Output:**

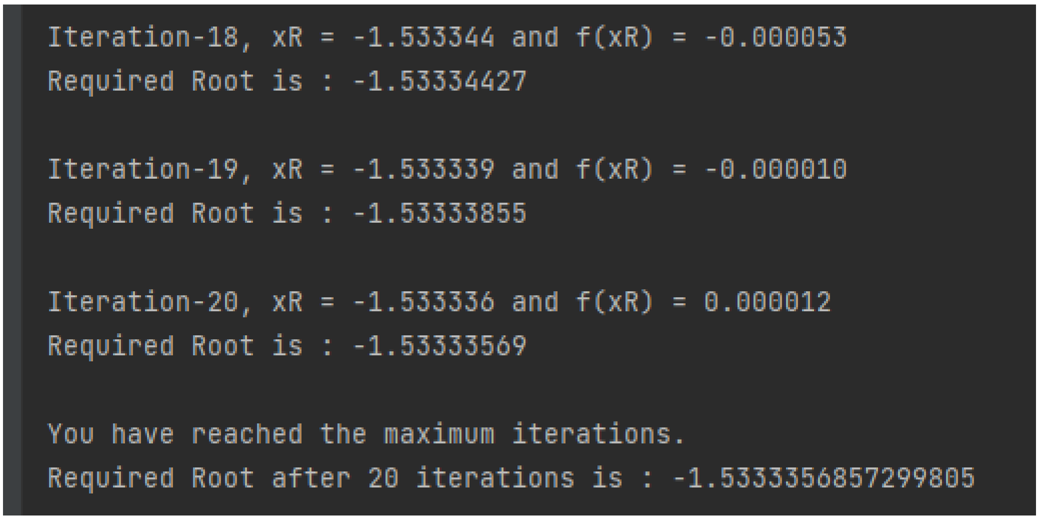
xL = -2
xU =1
e = 10E-9



```
First Guess: -2
Second Guess: 1
Tolerable Error: 10E-9

 ****BISECTION METHOD IMPLEMENTATION*****

Iteration-1, xR = -0.500000 and f(xR) = 3.826750
Required Root is : -0.50000000

Iteration-2, xR = -1.250000 and f(xR) = 1.782062
Required Root is : -1.25000000

Iteration-3, xR = -1.625000 and f(xR) = -0.733719
Required Root is : -1.62500000

Iteration-4, xR = -1.437500 and f(xR) = 0.681584
Required Root is : -1.43750000

Iteration-5, xR = -1.531250 and f(xR) = 0.015758
Required Root is : -1.53125000
```

.

…



```
Iteration-18, xR = -1.533344 and f(xR) = -0.000053
Required Root is : -1.53334427

Iteration-19, xR = -1.533339 and f(xR) = -0.000010
Required Root is : -1.53333855

Iteration-20, xR = -1.533336 and f(xR) = 0.000012
Required Root is : -1.53333569

You have reached the maximum iterations.
Required Root after 20 iterations is : -1.5333356857299805
```