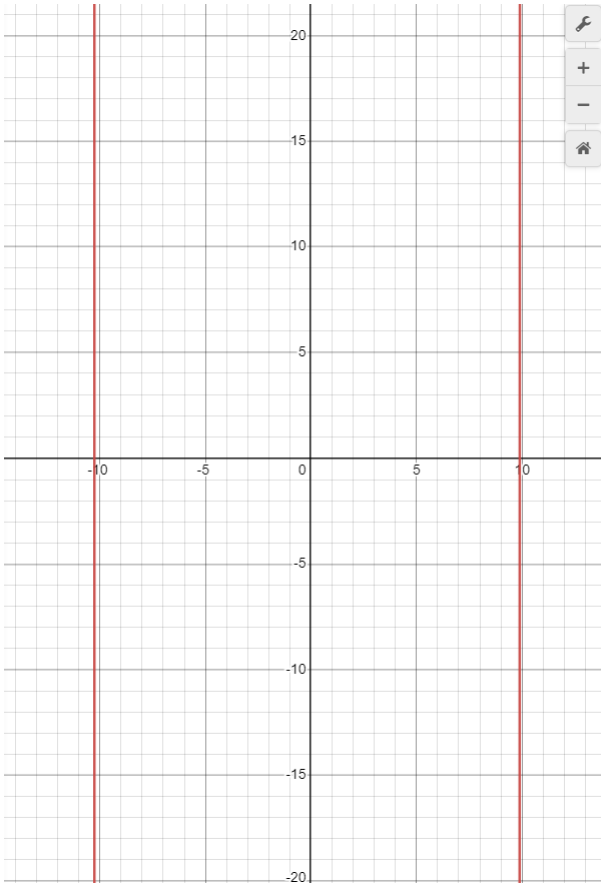
Assignment: Homework Five Name: Cody Strange
Disclaimer: This is my work, not that of others
Total Score: 30 (in points, not percentage)

Problem 1 score: 10
Problem 2 score: 10
Problem 3 score: 10

1. Using the bisection method. The first step is to graph the function and see roughly where the roots are located.



The roots are located close to -10 and 10. Because there are two roots I have to create the program to be able to check for each one individually. To make this easier to implement I'll adjust the given code to take an extra parameter that checks if I am looking for the smaller root or the larger root. To find the -10 root I manually make xr = xm for the first iteration, after that I can treat the problem as if there was only one root and find it. The root for the one that was close to -10 is actually -10.260964380932979. To find the root closer to 10 you do the opposite, you set the xl = xm on the first iteration and then treat the problem as if that was the only root. The root for the one closest to 10 is actually 9.886002700947888.

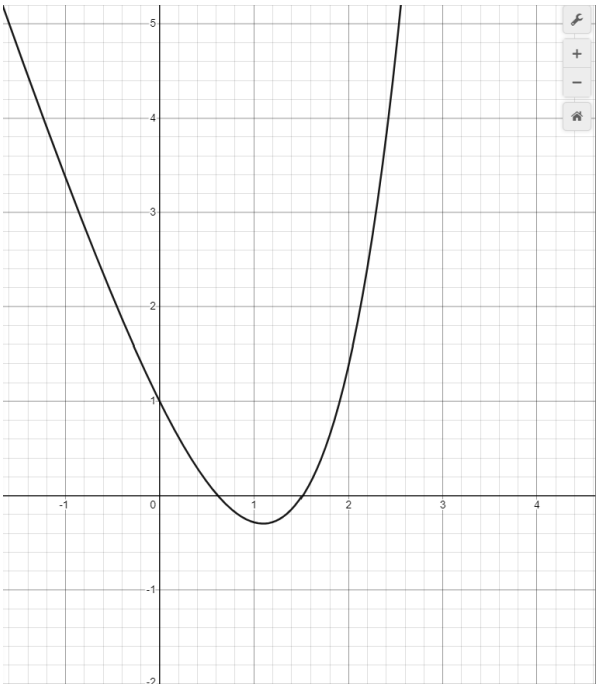
bisection.py > ...

```python
import numpy as np

def myfunc(x):
    return (x**4) - (3*(x**2)) + (75*x) - 10000

def bisect1(func,xl,xu,rootType,maxit=20):
    """
    Uses the bisection method to estimate a root of func(x).
    The method is iterated maxit (default = 20) times.
    Input:
        func = name of the function
        xl = lower guess
        xu = upper guess
    Output:
        xm = root estimate
        or
        error message if initial guesses do not bracket solution
    """
    xm = (xl+xu)/2
    if rootType == "lbr":
        xr = xm
    else:
        xl = xm

    for i in range(maxit-1):
        xm = (xl+xu)/2
        if func(xm)*func(xl)>0:
            xl = xm
        else:
            xu = xm
    return xm


def main():
    print(bisect1(myfunc, -20, 20, "ubr", 1000))
if __name__ == "__main__":
    main()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS D:\School\CS3320\HW\HW-5> & C:/Users/codyl/AppData/Local/Programs/Python/Python39/
9.886002700947888
PS D:\School\CS3320\HW\HW-5>
```

2. Using the bisectional method. The first step is to graph the function to get a rough estimate of where the roots are located.

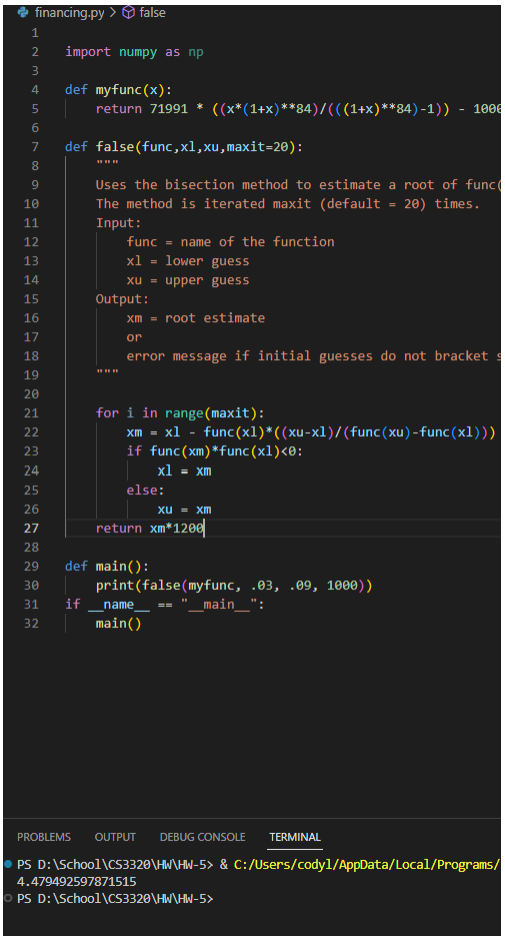



```python
import numpy as np
import math
def myfunc(x):
    return math.e**x-3*x

def bisect1(func,xl,xu,rootType,maxit=20):
    """
    Uses the bisection method to estimate a root of func(x).
    The method is iterated maxit (default = 20) times.
    Input:
        func = name of the function
        xl = lower guess
        xu = upper guess
    Output:
        xm = root estimate
        or
        error message if initial guesses do not bracket solution
    """
    xm = (xl+xu)/2
    if rootType == "ubr":
        xu = xm
    else:
        xl = xm

    for i in range(maxit-1):
        xm = (xl+xu)/2
        if func(xm)*func(xl)>0:
            xl = xm
        else:
            xu = xm
    return xm


def main():
    print(bisect1(myfunc, 0, 1, "lbr", 1000))
if __name__ == "__main__":
    main()
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
1.5121345516578422
PS D:\School\CS3320\HW\HW-5> & C:/Users/cody1/AppData/Local/Programs/Python/Pyth
0.6190612867359451
PS D:\School\CS3320\HW\HW-5>
```

Roots are 1.5121345516578422 and 0.6190612867359451

3. Used the false position method



APR = 4.47%