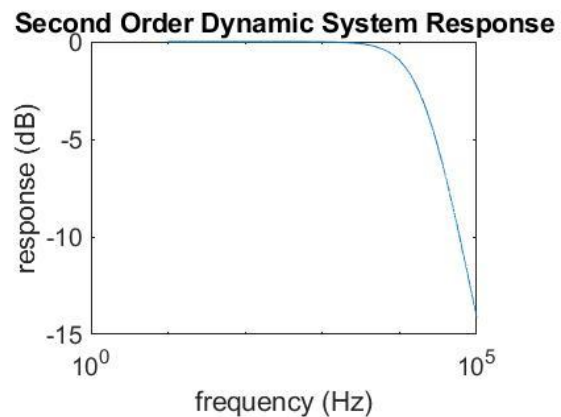
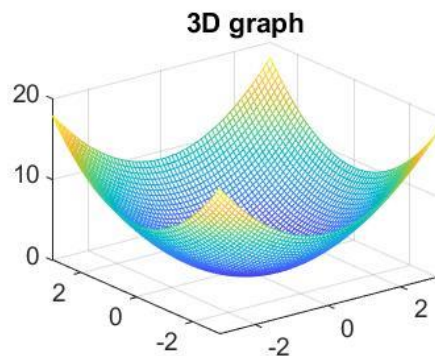
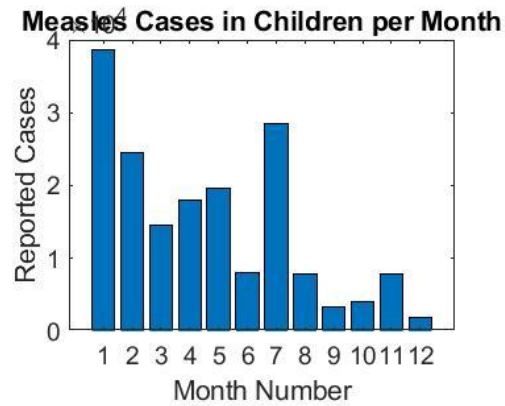
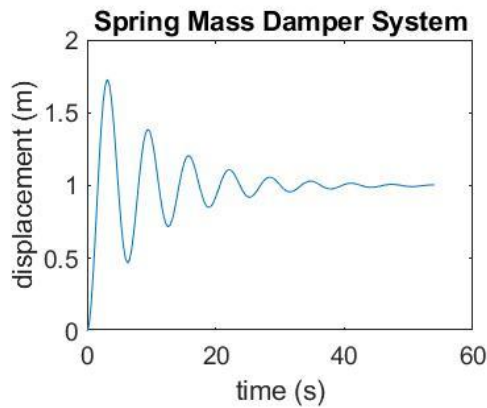


Name: Jingsi Zhou
Date: 10/18/2020
Section: 17460
Assignment: Lab 5

Problem 5.1 Four cases of data are provided to you in the file **data.mat** along with your manual:

1. A spring mass damper system was disturbed from its rest position and its displacement was recorded. The displacement (m) and time (s) data is stored in variables **xdisp** and **time** respectively.
2. The total number of reported cases of measles in children was recorded per month for a period of 12 months. The data is stored in variable **measles**.
3. z is a function of variables x and y in the form of:
$$z = f(x,y) = x^2 + y^2$$

For x and y defined over a range of $[-3, 3]$ in increments of 0.1, z needs to be plotted with respect to both x and y .
4. The frequency response (dB) of a second order dynamic system is recorded for frequencies (Hz) ranging from 0.01 Hz to 1 KHz. The data is provided in the variables **frequency** and **magnitude**. Hint: Because the frequencies span a large range, a standard line plot may not be appropriate.
You need to write a script to:
 1. Import the ".mat" file provided.
 2. Display each type of data using suitable plots. (Hint: Think carefully about the nature of the data you are plotting when selecting a suitable plot.)
 3. Set appropriate title, labels, axis ranges for each plot.
 4. Set all plots in a 2 x 2 form using **subplot**
 5. Export/save the 2 x 2 figure into a ".jpg" format.
 6. Use the jpg file in your solution (Import the jpg file into your solution sheet. **Screenshots of the plot will be given no points.**)



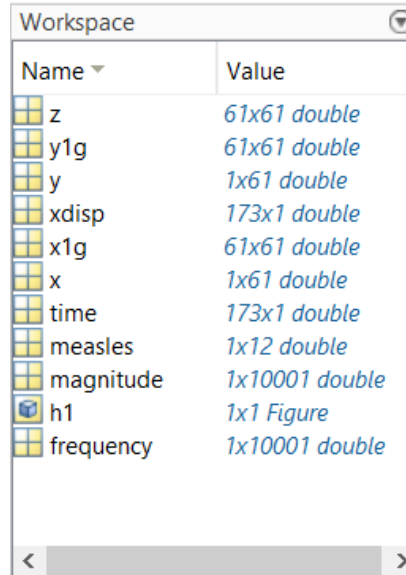
```
load('data.mat');

figure(1);
title('Lab 5 Question 1');
subplot(2, 2, 1);      %first graph with time and xdisp data set
plot(time, xdisp);
title('Spring Mass Damper System');
xlabel('time (s)');
ylabel('displacement (m)');
subplot(2, 2, 2);
bar(measles);
title('Measles Cases in Children per Month'); %bar graph that shows
measles data
xlabel('Month Number');
ylabel('Reported Cases');
subplot(2, 2, 3);
x = -3:0.1:3;
y = -3:0.1:3;
[x1g, y1g] = meshgrid(x, y); %meshgrid so that z can be calculated
and
z = x1g.^2+y1g.^2;           %so 3D graph can be properly graphed
mesh(x1g,y1g,z)
title('3D graph')
subplot(2, 2, 4);
semilogx(frequency, magnitude);
title('Second Order Dynamic System Response');
xlabel('frequency (Hz)'); %used semilogx here because frequency
data set
```

```
ylabel('response (dB)'); %encompasses a large range
```

Command Window:

```
>> h1 = figure(1);
>> saveas(h1, 'Lab5Q1.jpg')
```



Name	Value
z	61x61 double
y1g	61x61 double
y	1x61 double
xdisp	173x1 double
x1g	61x61 double
x	1x61 double
time	173x1 double
measles	1x12 double
magnitude	1x10001 double
h1	1x1 Figure
frequency	1x10001 double

Problem 5.2 Start out by creating two m-files which contain the function definitions for the functions described above. Make sure that your function definitions can accept both vector and scalar input. Test out the functions by calling them in the command line passing vector and scalar as input.

Note: The exponential function in matlab is **exp**. Find out how to use it by typing **doc exp** in the command line.

```
function value = f(x)
value = (3/500)*(x.^3 - 18.535*x.^2 + 25.697*x + 28.099);
```

```
function value = g(x)
value = 2.*x.*exp(cos(3.*x)).*exp(-x) + 70;
```

Command Window:

```
>> vectorg = g([3:.1:5])
```

```
vectorg =
```

```
Columns 1 through 4
```

```
70.1201    70.1036    70.0975    70.1000
```

```
Columns 5 through 8
```

```
70.1111    70.1314    70.1620    70.2031
```

```
Columns 9 through 12
```

```
70.2520    70.3017    70.3407    70.3566
```

```
Columns 13 through 16
```

```

    70.3422    70.3002    70.2419    70.1813

Columns 17 through 20

    70.1287    70.0887    70.0609    70.0428

Column 21

    70.0315

>> scalarg = g(3)

scalarg =

    70.1201

>> vectorf = f([-2:.2:2])

vectorf =

Columns 1 through 4

   -0.6326   -0.5042   -0.3874   -0.2817

Columns 5 through 8

   -0.1869   -0.1028   -0.0290    0.0348

Columns 9 through 12

    0.0887    0.1333    0.1686    0.1950

Columns 13 through 16

    0.2129    0.2224    0.2238    0.2176

Columns 17 through 20

    0.2038    0.1829    0.1552    0.1208

Column 21





    0.0801

>> scalarf = f(3)

scalarf =

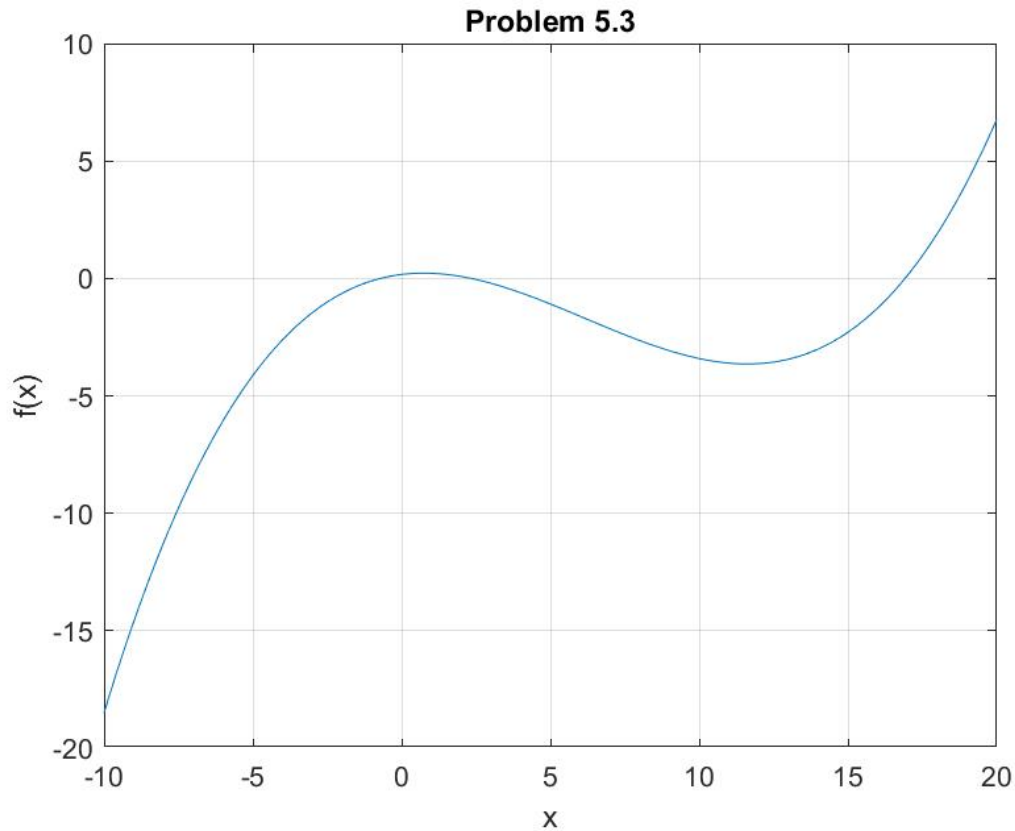
   -0.2078

```

Workspace	
Name ▼	Value
 vectorg	1x21 double
 vectorf	1x21 double
 scalarg	70.1201
 scalarf	-0.2078

Problem 5.3 Create an m-file that calls the function you created corresponding to $f(x)$ with input as a vector $x = -10 : 0.1 : 20$.

(a) Plot the function output corresponding to x and place the plot in the results section of the solution manual with appropriate labeling, title and explanation.



(b) Using the above plot and the **Data Cursor** tool in the figure window (the + sign button), find the value(s) of x where f goes to zero. List the value(s) in the Results section.

$x = -0.7, 2.3, 16.9$

Problem 5.4 Write an m-file that will perform the Bisection algorithm. Since we don't know how many steps it will take, a FOR loop isn't the best thing. Let's use a **while** loop instead. We want to repeat all our steps until either

- $|x_b - x_a| < 1 \times 10^{-4}$, (hint: the absolute function in MATLAB is **abs**) OR
- the number of times the loop has repeated is > 20 .

The first condition is a convergence criteria. The second condition will stop the program if convergence is not met within a maximum allowable number of steps. The last condition can be satisfied by using a counting variable inside the while loop. Use maximum number of iterations as 20.

(a) Using the plot from the previous problem, choose an interval where f crosses 0. The interval does not have to be very precise. Write down the interval that you picked and the solution (the value of x that satisfies the conditions) your code gives.

interval = [16.5, 17.3]

root =

'The root of the equation is located between 16.9179 and 16.918'

numits =

13

Code for Bisection Method:

```
function [Root1, NumIterations] = BisectRoot(xL, xR);

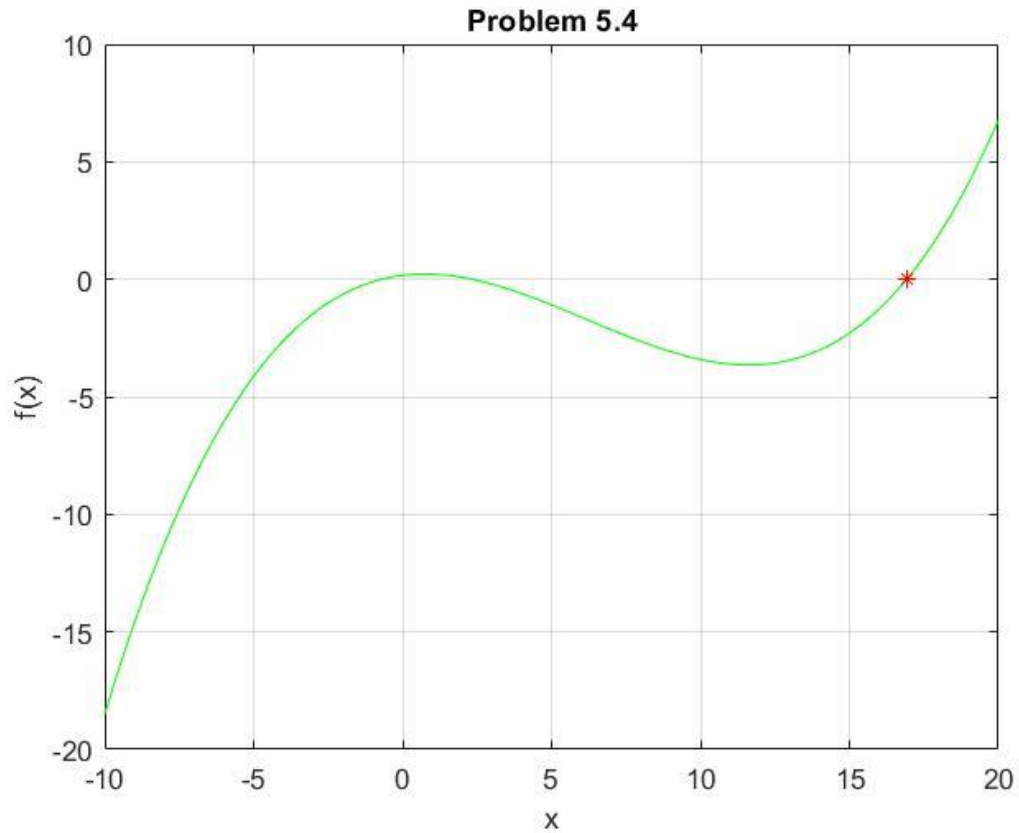
Error = 1*10^(-4);
LeftSol = f(xL);
RightSol = f(xR);

if LeftSol*RightSol < 0
    LeftX = xL;
    RightX = xR;
    count = 0;
    while abs(RightX - LeftX) > Error || count < 20
        MidX = (LeftX + RightX)/2;
        Middle = f(MidX);
        if Middle*RightSol < 0
            LeftX = MidX;
        else
            RightX = MidX;
        end
        count = count + 1;
        disp(['iteration ', num2str(count), ': ', num2str(LeftX), ', ',
num2str(RightX)])
    end
else
    disp(['There is no root between ', num2str(xL), ' and ',
num2str(xR)])
end

Root1 = (['The root of the equation is located between ',
num2str(LeftX), ...
' and ', num2str(RightX)]);
```

```
NumIterations = count;
```

(b) Plot the curve just like you did in the previous problem and overlay the solution as a point on the existing plot. Use **hold** command to do this. Make sure that the overlaid point is a different color than the curve. Include this plot in your results

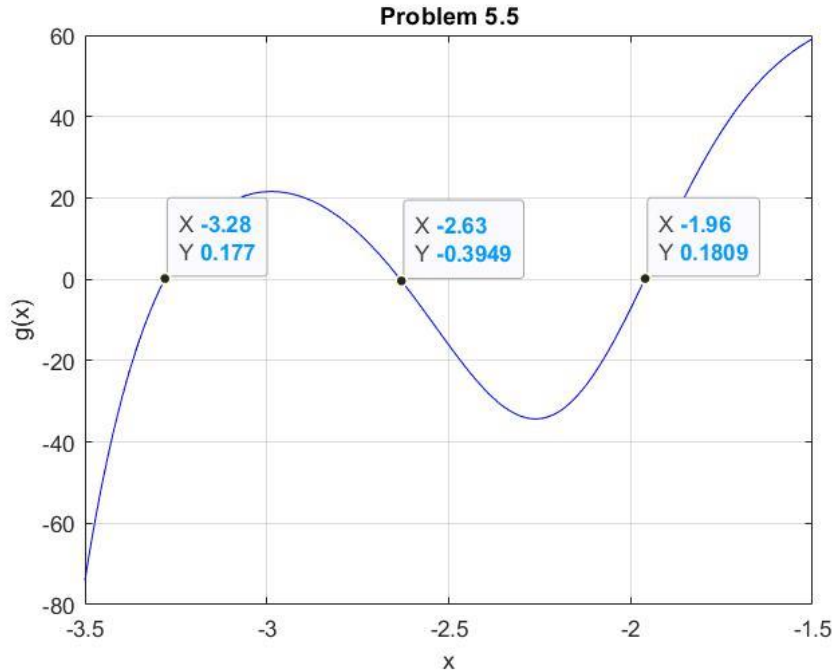


```
x = -10:0.1:20;
y = f(x)
figure(1);
plot(x, y, 'g');
title('Problem 5.4');
xlabel('x');
ylabel('f(x)');
grid on

hold on
plot(16.918, f(16.918), 'r*')
hold off
```

Problem 5.5 Write an m-file for the False-Position method to find all of the roots of $g(x)$ in the interval $[-3.5, -1.5]$ for a desired accuracy $\epsilon = 1 \times 10^{-3}$ and maximum number of iterations as 100.

a) Find all the roots in the given interval and list the initial guesses that gave you each of the unique roots.



```
x = -3.5:0.01:-1.5;
y = g(x)
figure(1);
plot(x, y, 'b');
title('Problem 5.5');
xlabel('x');
ylabel('g(x) ');
grid on
```

intervals: $[-3.35, -3.25]$; $[-2.65, -2.5]$; $[-2, -1.95]$

False-Position Method Code:

```
function [Root1, NumIterations] = FalsePostion(xL, xR);

LeftSol = g(xL);
RightSol = g(xR);
Error = 1*10^(-3);

if LeftSol*RightSol < 0
    LeftX = xL;
    RightX = xR;
    count = 0;
    while abs(RightX - LeftX) > Error
        if count > 100
            break
        end
        XApp = LeftX - ((RightX-LeftX)/(RightSol-LeftSol))*LeftSol;
```



```

XApp_Sol = g(XApp);
if LeftSol*XApp_Sol < 0
    RightX = XApp;
else
    LeftX = XApp;
end
count = count + 1;
disp(['iteration ', num2str(count), ': ', num2str(LeftX), ', ',
num2str(RightX)])
end
else
    disp(['There is no root between ', num2str(xL), ' and ',
num2str(xR)])
end

Root1 = (['The root of the equation is located between ',
num2str(LeftX), ...
' and ', num2str(RightX)]);
NumIterations = count;

```

Roots:

```

>> [root,numits] = FalsePosition(-3.35, -3.25)
iteration 1: -3.35, -3.2752
iteration 2: -3.294, -3.2752
iteration 3: -3.294, -3.2799
iteration 4: -3.2835, -3.2799
iteration 5: -3.2835, -3.2808
iteration 6: -3.2815, -3.2808

```

root =

'The root of the equation is located between -3.2815 and -3.2808'

numits =

6

```

>> [root,numits] = FalsePosition(-2.65, -2.5)
iteration 1: -2.6346, -2.5
iteration 2: -2.6346, -2.6209
iteration 3: -2.6346, -2.6332
iteration 4: -2.6345, -2.6332
iteration 5: -2.6344, -2.6332
iteration 6: -2.6343, -2.6332
iteration 7: -2.6342, -2.6332

```

root =

'The root of the equation is located between -2.6342 and -2.6332'

numits =

7

```
>> [root,numits] = FalsePosition(-2, -1.95)
iteration 1: -1.9611, -1.95
iteration 2: -1.9611, -1.9525
iteration 3: -1.9611, -1.9544
iteration 4: -1.9611, -1.9559
iteration 5: -1.9611, -1.9571
iteration 6: -1.9611, -1.958
iteration 7: -1.9611, -1.9587
iteration 8: -1.9611, -1.9592
iteration 9: -1.9611, -1.9596
iteration 10: -1.9611, -1.96
iteration 11: -1.9611, -1.9602
```

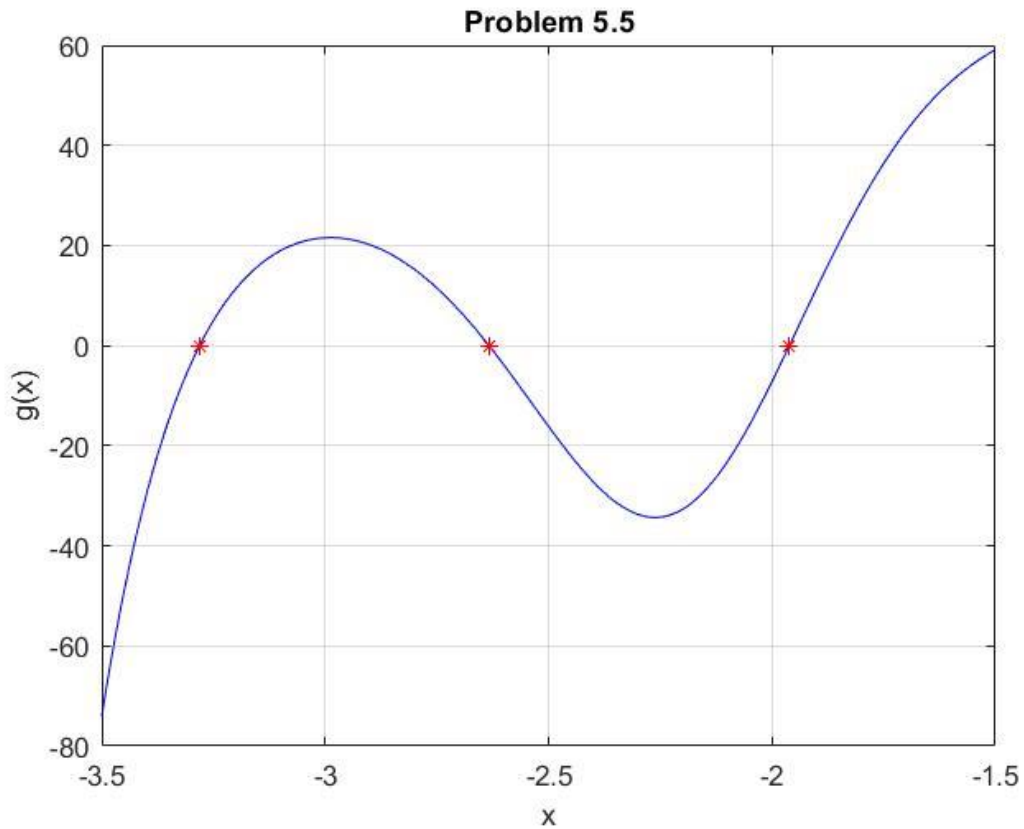
```
root =
```

'The root of the equation is located between -1.9611 and -1.9602'

```
numits =
```

```
11
```

b) Plot $g(x)$ for the given interval and overlay the three points on the same plot. Include this plot in your results. Don't forget to title and label the plot appropriately before saving it.



```
x = -3.5:0.01:-1.5;
y = g(x);
figure(1);
plot(x, y, 'b');
```

```

title('Problem 5.5');
xlabel('x');
ylabel('g(x)');
grid on
hold on
plot(-3.2811, g(-3.2811), 'r*')
plot(-2.6337, g(-2.6337), 'r*')
plot(-1.9608, g(-1.9608), 'r*')
hold off

```

Problem 5.6 Use the bisection program from Problem 5.4, but for $g(x)$ and try to run it with $x_a = 4$ and $x_b = 6$. What do you get for x ? What is the output of the function at the x that you got? Is this value close to 0? Plot $g(x)$ for $x = [-1 : 0.1 : 6]$ and notice the interval $x = [4, 6]$. Can you see why you got a wrong answer? Explain why. (The explanations go right here in the discussions section of your solution template.)

```

>> [root,numits] = BisectRoot(4,6)
There is no root between 4 and 6
Output argument "Root1" (and maybe others)
not assigned during call to "BisectRoot".

```

For my bisection method program, I added a section where, if there is no root between a given interval, the program returns a sentence that states that there is no root. No output x value was displayed; therefore, there was no output at x .

