University of Dhaka

Department of Computer Science and Engineering

CSE-3212: Numerical Methods Lab

3rd Year 2nd Semester

Assignment: 02

Problems on Bisection, False Position, Newton-Raphson and Secant methods

Submitted by:

Name: Md. Musfiqur Rahman, Roll: 05

Date of submission: 16th September, 2018

Submitted to:

Mr. Mubin Ul Haque, Lecturer, Department of CSE, University of Dhaka

Problem 1

The velocity v of a falling parachutist is given by

$$v = \frac{gm}{c}\left(1 - e^{-(c/m)t}\right)$$

where g = 9.8 m/s^2. For a parachutist with a drag coefficient c = 15 kg/s, compute the mass m so that the velocity is v = 35 m/s at t = 9 s.

By using

(a) bisection

and (b) false position.

Solution:

```
#include<stdio.h>
#include<math.h>
#include<bits/stdc++.h>
using namespace std;


void print(double v1, double v2, double v3, double v4, double v5, double v6)
{
    cout<< "|" << setw(15) << v1 << "|" << setw(15) << v2 << "|" << setw(15) << v3 << "|" << setw(15) << v4
<< "|" << setw(15) << v5 << "|" << setw(15) << v6<< "|" <<endl;
    for(int i=0; i<100-4; i++)
        printf("-");
    puts("");
}

void print(string v1, string v2, string v3, string v4, string v5, string v6)
{
    cout<< "|" << setw(15) << v1 << "|" << setw(15) << v2 << "|" << setw(15) << v3 << "|" << setw(15) << v4
<< "|" << setw(15) << v5 << "|" << setw(15) << v6<< "|" <<endl;
    for(int i=0; i<100-4; i++)
        printf("-");
    puts("");
}

double g=9.8, v=35, t=9, c=15;
double f(double m)
{
    double r= g*m * (1 - exp(-(c/m)*t));
    r=r/c -v;
    return r;
}
```

```
void doBisection(double lo, double hi, double prec)
{

    printf("Solving with Bisection method\n\n");
    int iter=0;
    double past=0, cur=0,mid;
    print("iteration", "Upper value", "Lower value", "Xm", "f(Xm)", "Relative error");
    while(1)
    {
        past=mid;
        mid=(lo+hi)/2;
        double r=f(mid);
        if(f(mid)*f(lo)>0)
            lo=mid;
        else if(f(mid)*f(hi)>0)
            hi=mid;

        double rerror= fabs(mid-past)/mid;
        // printf("iteration=%d Upper value=%.4f Lower value=%.4f Xm=%.4f  f(Xm)=%.4f  error=%.4f\n",++iter,
hi,lo,mid,f(mid), rerror);
        print(++iter, hi,lo,mid,f(mid), rerror);
        if(rerror<prec) break;
    }
    printf("root=%.4f\n",mid);
}

double find_point(double x0, double x1)
{
    double r= (f(x0)*(x1-x0))/(f(x0)-f(x1)) + x0;
    return r;
}

void doFalsePosition(double lo, double hi, double prec)
{
    printf("\n\n\nSolving with FalsePosition method\n\n");
    int iter=0;
    double past=0, cur=0,mid;
    print("iteration", "Upper value", "Lower value", "Xm", "f(Xm)", "Relative error");
while(1)
    {
        past=mid;
        mid=find_point(lo,hi);
        double r=f(mid);
        if(f(mid)*f(lo)>0)
            lo=mid;
        else if(f(mid)*f(hi)>0)
            hi=mid;
        double rerror= fabs(mid-past)/mid;
        //  printf("iteration=%d Upper value=%.4f Lower value=%.4f Xm=%.4f  f(Xm)=%.4f  error=%.4f\n",+
+iter, hi,lo,mid,f(mid), rerror);
        print(++iter, hi,lo,mid,f(mid), rerror);
        if(rerror<prec) break;
    }
```

```
    printf("root=%.4f\n",mid);
}

void printLowToHigh(double a, double b)
{
    for(double i=a; i<=b; i+=0.1)
    {
        cout<<i<<" "<<f(i)<<endl;
    }
    puts("");

}

int main()
{
    printf("Maximize the screen\n");
    double lo=-100, hi=0, prec;
    printf("lower limit:");
    cin>>lo;
    printf("higer limit:");
    cin>>hi;
    printf("tolerance:");
    cin>>prec;
    printLowToHigh(lo,hi);
    if(f(lo)*f(hi)>0)
    {
        printf("No root is possible\n");
        return 0;
    }
    doBisection(hi,lo,prec);
    doFalsePosition(hi,lo,prec);
}
```

Sample input:

lower limit:50
higer limit:60
tolerance:0.00001

Sample output:

Snapshot 1 :

```
Terminal
Maximize the screen
lower limit:50
higer limit:60
tolerance:0.00001
50 -4.52871
50.1 -4.47966
50.2 -4.43067
50.3 -4.38174
50.4 -4.33288
50.5 -4.28408
50.6 -4.23534
50.7 -4.18667
50.8 -4.13806
50.9 -4.08951
51 -4.04103
51.1 -3.99261
51.2 -3.94426
51.3 -3.89597
51.4 -3.84774
51.5 -3.79958
51.6 -3.75148
51.7 -3.70344
51.8 -3.65547
51.9 -3.60756
52 -3.55971
52.1 -3.51193
52.2 -3.46421
52.3 -3.41655
52.4 -3.36895
52.5 -3.32142
52.6 -3.27395
52.7 -3.22655
52.8 -3.1792
52.9 -3.13192
53 -3.0847
53.1 -3.03755
53.2 -2.99046
53.3 -2.94343
53.4 -2.89646
53.5 -2.84955
53.6 -2.80271
53.7 -2.75593
53.8 -2.70921
```

```
Terminal
54.1 -2.56943
54.2 -2.52296
54.3 -2.47655
54.4 -2.4302
54.5 -2.38392
54.6 -2.33769
54.7 -2.29153
54.8 -2.24543
54.9 -2.1994
55 -2.15342
55.1 -2.10751
55.2 -2.06165
55.3 -2.01586
55.4 -1.97013
55.5 -1.92446
55.6 -1.87885
55.7 -1.8333
55.8 -1.78782
55.9 -1.74239
56 -1.69703
56.1 -1.65173
56.2 -1.60648
56.3 -1.5613
56.4 -1.51618
56.5 -1.47112
56.6 -1.42612
56.7 -1.38118
56.8 -1.3363
56.9 -1.29148
57 -1.24672
57.1 -1.20202
57.2 -1.15739
57.3 -1.11281
57.4 -1.06829
57.5 -1.02383
57.6 -0.979433
57.7 -0.935095
57.8 -0.890816
57.9 -0.846597
58 -0.802437
58.1 -0.758337
58.2 -0.714296
58.3 -0.670315
```

```
58.4 -0.626393
58.5 -0.58253
58.6 -0.538726
58.7 -0.494982
58.8 -0.451296
58.9 -0.40767
59 -0.364102
59.1 -0.320594
59.2 -0.277144
59.3 -0.233752
59.4 -0.190419
59.5 -0.147145
59.6 -0.103929
59.7 -0.0607721
59.8 -0.017673
59.9 0.0253678
```

Snapshot 2 :

Solving with Bisection method

iteration	Upper value	Lower value	Xm	f(Xm)	Relative error
1	55	60	55	-2.15342	1
2	57.5	60	57.5	-1.02383	0.0434783
3	58.75	60	58.75	-0.473132	0.0212766
4	59.375	60	59.375	-0.201247	0.0105263
5	59.6875	60	59.6875	-0.0661636	0.0052356
6	59.6875	59.8438	59.8438	0.00116448	0.00261097
7	59.7656	59.8438	59.7656	-0.0324818	0.00130719
8	59.8047	59.8438	59.8047	-0.0156542	0.000653168
9	59.8242	59.8438	59.8242	-0.00724375	0.000326477
10	59.834	59.8438	59.834	-0.00303935	0.000163212
11	59.8389	59.8438	59.8389	-0.000937367	8.15993e-05
12	59.8389	59.8413	59.8413	0.000113575	4.0798e-05
13	59.8401	59.8413	59.8401	-0.000411892	2.03994e-05
14	59.8407	59.8413	59.8407	-0.000149157	1.01996e-05
15	59.841	59.8413	59.841	-1.77909e-05	5.09978e-06

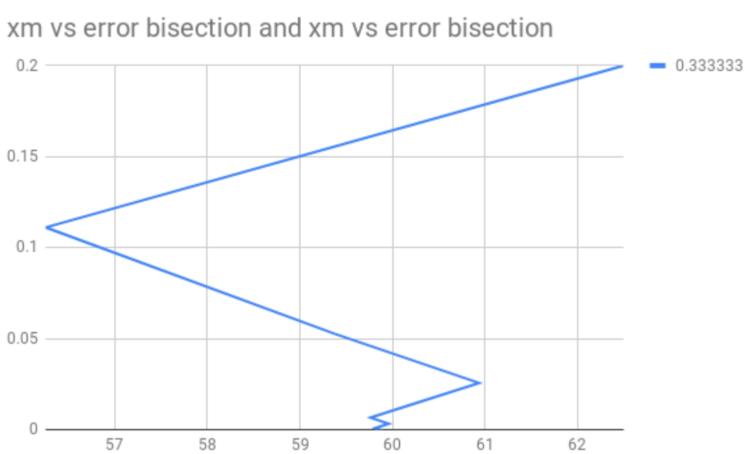
root=59.8410

Snapshot 3:

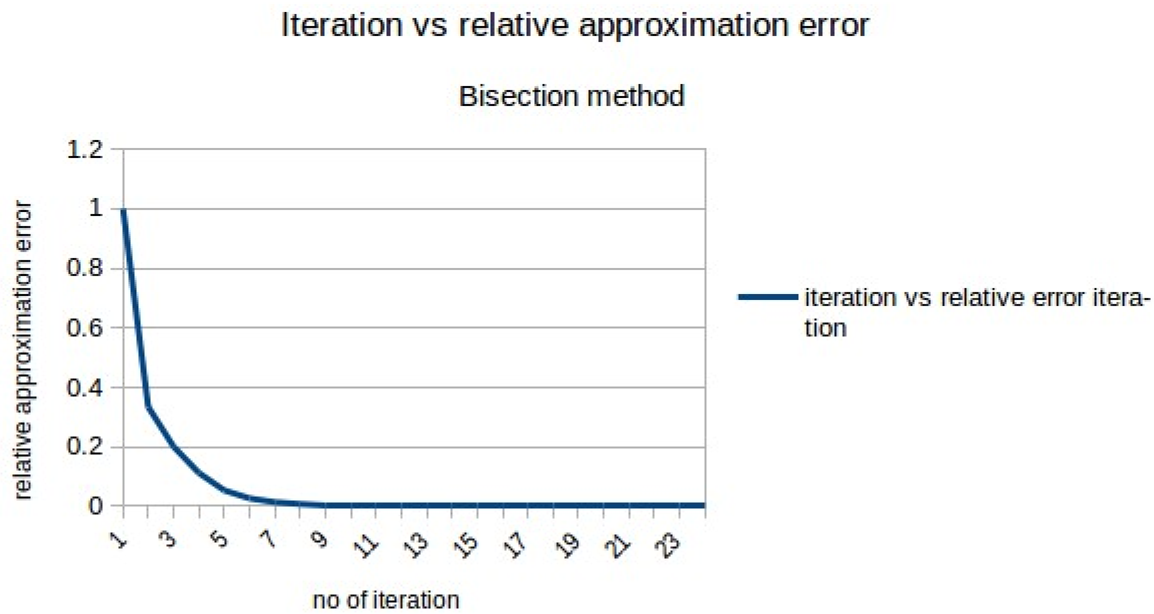
```
Solving with FalsePosition method
|      iteration|     Upper value|     Lower value|              Xm|           f(Xm)| Relative error|
-----------------------------------------------------------------------------------------------------
|              1|              50|         59.8513|         59.8513|      0.00442161|    0.000172325|
-----------------------------------------------------------------------------------------------------
|              2|              50|         59.8417|         59.8417|     0.000285644|    0.000160573|
-----------------------------------------------------------------------------------------------------
|              3|              50|         59.8411|         59.8411|     1.84516e-05|    1.03728e-05|
-----------------------------------------------------------------------------------------------------
|              4|              50|          59.841|          59.841|      1.1919e-06|      6.7004e-07|
-----------------------------------------------------------------------------------------------------
root=59.8410
```

Graphs:

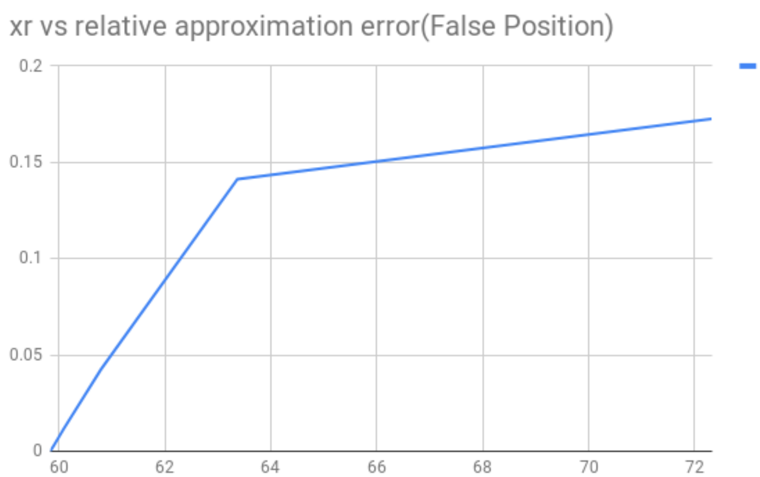
Graph1: The graph of x_m and relative approximation error (bisection).



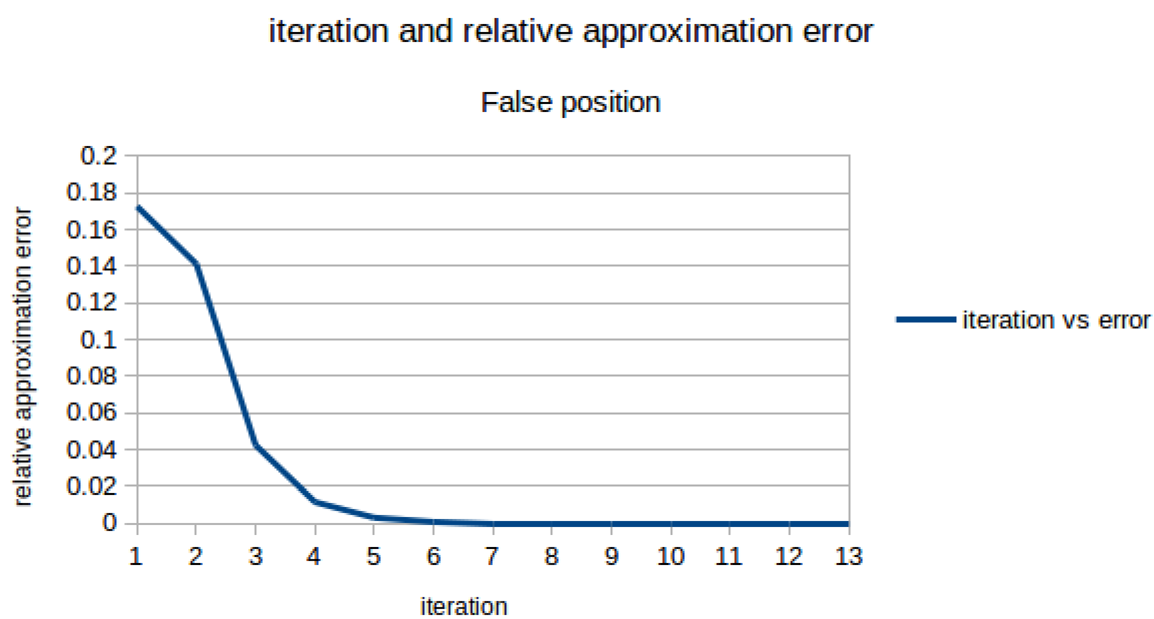
Graph2: The graph of no of iteration and relative approximation error (bisection).



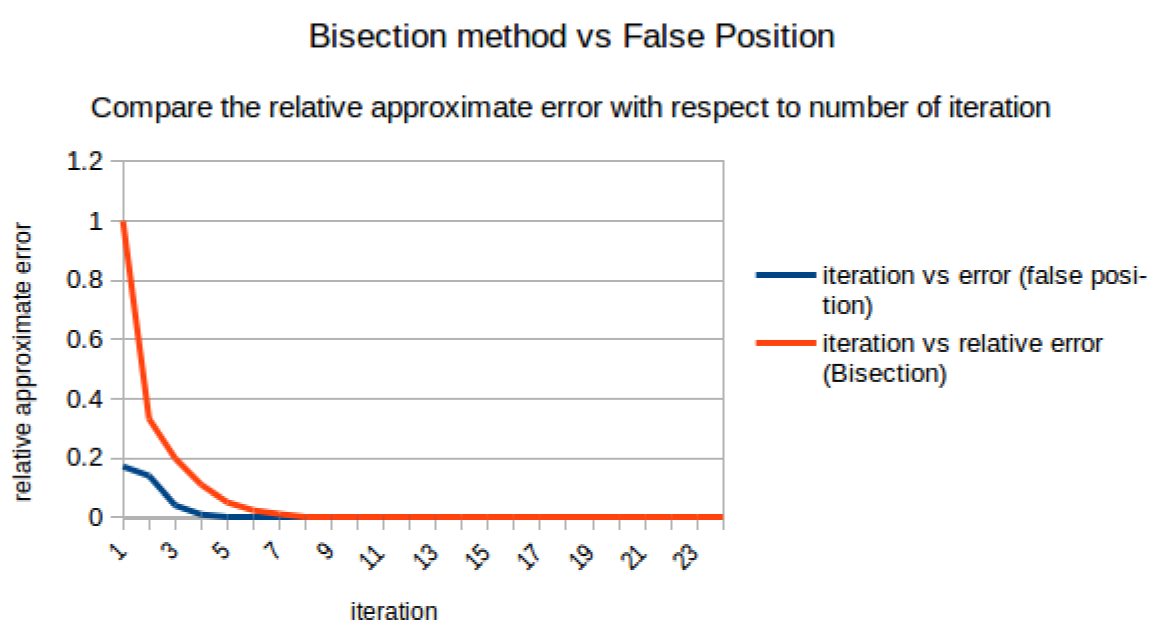
Graph 3: The graph of x_r and relative approximation error (false position).



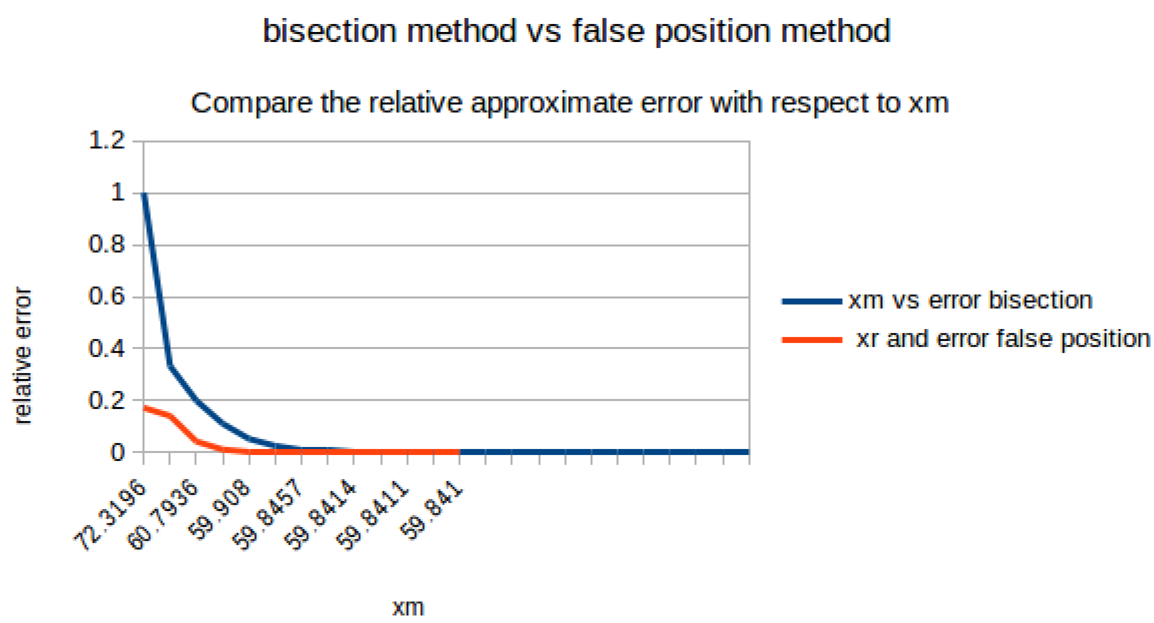
Graph 4: The graph of no of iteration and relative approximation error (false position).



Graph 5: Compare the relative approximate error with respect to number of iteration between the bisection method and false position method.



Graph 6: Compare the relative approximate error with respect to *x*



Problem 2

(a) Use the Newton-Raphson method to determine a root of f (x) = −x2 + 1.8x + 2.5 using x0 = 5.
(b) Use the Newton-Raphson method to find the root of

$f(x) = e^{(-0.5x)}(4 - x) - 2$

Employ initial guesses of (i) 2, (ii) 6, and (iii) 8.

Solution:

```
#include<bits/stdc++.h>
using namespace std;
double f10(double x)
{
   double r= -x*x + 1.8*x +2.5;
   return r;
}
double f11(double x)
{
   double r= -2*x +1.8;
   return r;
}
void print(double v1, double v2, double v3, double v4, double v5)
{
   cout<< "|" << setw(15) << v1 << "|" << setw(15) << v2 << "|" << setw(15) << v3 << "|" << setw(15) << v4
<< "|" << setw(15) << v5 << "|"  <<endl;
```

```
    for(int i=0; i<80; i++)
        printf("-");
    puts("");
}
void print(string v1, string v2, string v3, string v4, string v5)
{
    cout<< "|" << setw(15) << v1 << "|" << setw(15) << v2 << "|" << setw(15) << v3 << "|" << setw(15) << v4
<< "|" << setw(15) << v5 << "|" <<endl;
    for(int i=0; i<80; i++)
        printf("-");
    puts("");
}
double f20(double x)
{
    double r= exp(-0.5*x) * (4-x) -2;
    return r;
}
double f21(double x)
{
    double r= -exp(-0.5*x) - 0.5 * exp(-0.5*x) * (4-x);
    return r;
}
void Newton_Raphson(double initGuess, double input_tolerance, int cs)
{
    double x0, tolerance;;
    x0=initGuess;
    tolerance=input_tolerance;
    double x1=x0,rError=1000;
    print("iteration", "xi", "f(xi)", "f'(xi)", "Relative error");
    int cnt=0;
    while(rError>=tolerance)
    {
        x0=x1;
        double r0,r1;
        if(cs==1)
        {
            r0=f10(x0);
            r1=f11(x0);
        }
```

```
        else
        {
            r0=f20(x0);
            r1=f21(x0);
        }
        print(++cnt,x0,r0, r1,rError);
        if(r1==0)
        {
            printf("Causing division by zero hence terminating\n");
            return ;
        }
        x1= x0 - r0/r1;
        rError=fabs((x1-x0)/x1);
    }
    printf("root=%.6f\n",x1);
}
int main()
{   printf("Maximize the screen\n");
    printf("Newton-Raphson:\n1st equation: root of f (x) = -x^2 + 1.8x + 2.5\n");
    printf("Input tolerance:");
    double tol;
    cin>>tol;
    printf("Initial root: 5 tolerance:%.6f\n\n",tol);
    Newton_Raphson (5,tol,1);
    puts("");
    puts("");
    printf("2nd equation: root of f (x) = e^(-0.5x) (4 - x) - 2\n");
    tol=0.0001;
    printf("Initial root: 2 tolerance:%.6f\n\n",tol);
    Newton_Raphson (2,tol,2);
    puts("");
    puts("");
    printf("Initial root: 6 tolerance:%.6f\n\n",tol);
    Newton_Raphson (6,tol,2);
    puts("");
    puts("");
    printf("Initial root: 8 tolerance:%.6f\n\n",tol);
    Newton_Raphson (8,tol,2);
    puts("");
```

```
    puts("");
}
```

<u>Sample Input:</u>

Input tolerance:0.00001

<u>Sample Output:</u>

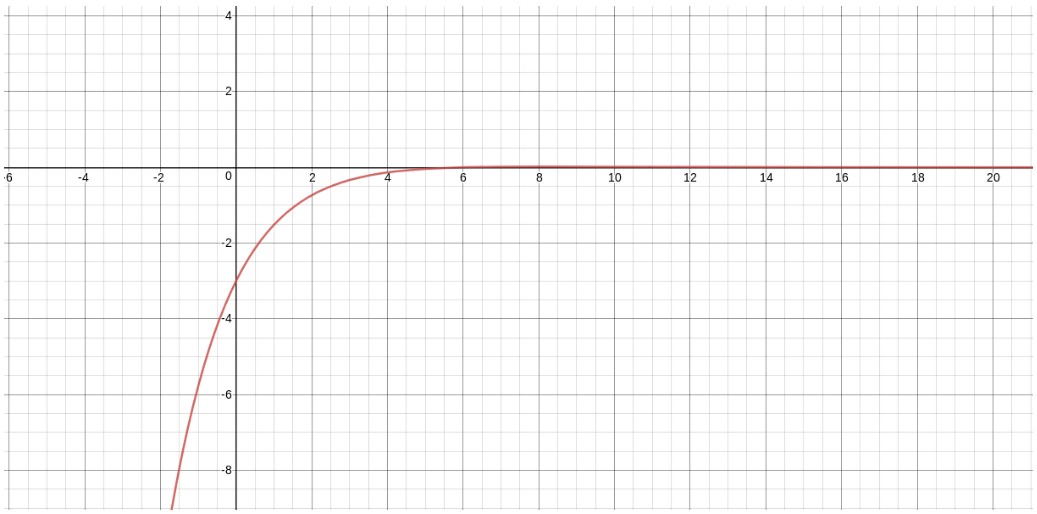
Snapshot 1:

```
Terminal
Maximize the screen
Newton-Raphson:
1st equation: root of f (x) = -x^2 + 1.8x + 2.5
Input tolerance:0.00001
Initial root: 5 tolerance:0.000010

|      iteration|             xi|          f(xi)|         f'(xi)| Relative error|
--------------------------------------------------------------------------------
|              1|              5|          -13.5|           -8.2|           1000|
--------------------------------------------------------------------------------
|              2|        3.35366|       -2.71044|       -4.90732|       0.490909|
--------------------------------------------------------------------------------
|              3|        2.80133|      -0.305064|       -3.80266|       0.197166|
--------------------------------------------------------------------------------
|              4|        2.72111|    -0.00643586|       -3.64222|       0.029482|
--------------------------------------------------------------------------------
|              5|        2.71934|   -3.12235e-06|       -3.63868|    0.000649796|
--------------------------------------------------------------------------------
root=2.719341


2nd equation: root of f (x) = e^(-0.5x) (4 - x) - 2
Initial root: 2 tolerance:0.000100

|      iteration|             xi|          f(xi)|         f'(xi)| Relative error|
--------------------------------------------------------------------------------
|              1|              2|       -1.26424|      -0.735759|           1000|
--------------------------------------------------------------------------------
|              2|       0.281718|        1.22974|       -2.48348|        6.09929|
--------------------------------------------------------------------------------
|              3|       0.776887|        0.18563|       -1.77093|       0.637376|
--------------------------------------------------------------------------------
|              4|       0.881708|     0.00657947|       -1.64678|       0.118884|
--------------------------------------------------------------------------------
|              5|       0.885703|    9.13203e-06|       -1.64221|     0.00451095|
--------------------------------------------------------------------------------
root=0.885709


Initial root: 6 tolerance:0.000100

|      iteration|             xi|          f(xi)|         f'(xi)| Relative error|
--------------------------------------------------------------------------------
|              1|              6|       -2.09957|              0|           1000|
--------------------------------------------------------------------------------
Causing division by zero hence terminating


Initial root: 8 tolerance:0.000100

|      iteration|             xi|          f(xi)|         f'(xi)| Relative error|
--------------------------------------------------------------------------------
|              1|              8|       -2.07326|      0.0183156|           1000|
--------------------------------------------------------------------------------
|              2|        121.196|             -2|    2.77311e-25|       0.933991|
--------------------------------------------------------------------------------
|              3|    7.21213e+24|             -2|              0|              1|
--------------------------------------------------------------------------------
Causing division by zero hence terminating
```

Snapshot 2:

Problem2(b) Discussion:

In this problem , I was asked to find root of the equation where initial guess was 2,6 and 8. for initial guess 6,8 we find the the derivative of the function f’(x) =0 which causes division by zero. So, Newton Raphson can calculate root for 6 and 8.



Problem 3

(a) Consider following easily differentiable function,

f (x) = 8 sin(x)e–x − 1:

Use the secant method, when initial guesses of xi–1 = 0.5 and xi = 0.4

Solution:

```
#include<bits/stdc++.h>

using namespace std;

double f(double x)

{

    double r1=8*sin(x)*exp(-x)-1;
```

```
    return r1;
}
double find_point(double x0, double x1)
{
    double r1,r2;
    r1=x0*f(x1) - x1*f(x0);
    r2= f(x1)-f(x0);
    return r1/r2;
}
void print(double v1, double v2, double v3, double v4, double v5, double v6)
{
    cout<< "|" << setw(15) << v1 << "|" << setw(15) << v2 << "|" << setw(15) << v3 << "|" << setw(15) << v4
<< "|" << setw(15) << v5 << "|" << setw(15) << v6<< "|" <<endl;
    for(int i=0; i<100-4; i++)
        printf("-");
    puts("");
}
void print(string v1, string v2, string v3, string v4, string v5, string v6)
{
    cout<< "|" << setw(15) << v1 << "|" << setw(15) << v2 << "|" << setw(15) << v3 << "|" << setw(15) << v4
<< "|" << setw(15) << v5 << "|" << setw(15) << v6<< "|" <<endl;
    for(int i=0; i<100-4; i++)
        printf("-");
    puts("");
}
int main()
{
    printf("Maximize the screen\n");
    double x0,x1,x2;
    x0=0.5, x1=0.4;
    double rError=1000,tolerance;
```

```
    printf("f(x)=8sin(x)e^(–x)− 1\n");

    printf("Use the secant method, when initial guesses of xi–1 = 0.5 and xi = 0.4\n");

    printf("Input tolerance:");

    cin>>tolerance;

    x2=x1;

    x1=x0;

    print("iteration", "Upper value", "Lower value", "Xm", "f(Xm)", "Relative error");

    int cnt=0;

    while(rError>=tolerance)

    {

        x0=x1;

        x1=x2;

        x2= find_point(x0,x1);

        rError=fabs((x2-x1)/x2);

        //printf("iteration=%d Upper value=%.4f  Lower value=%.4f Xm=%.4f f(Xm)=%.4f rError=%.6f\n",+
+cnt,x0, x1,x2, f(x2),rError);

        print(++cnt,x0, x1,x2, f(x2),rError);

    }

    printf("root=%.4f\n",x2);

}
```

<u>Sample Input:</u>

Input tolerance:0.00001

<u>Sample Output:</u>

Snapshot 1:

```
Terminal
Maximize the screen
f(x)=8sin(x)e^(-x)- 1
Use the secant method, when initial guesses of xi-1 = 0.5 and xi = 0.4
Input tolerance:0.00001
|      iteration|    Upper value|    Lower value|             Xm|         f(Xm)| Relative error|
-----------------------------------------------------------------------------------------------
|              1|            0.5|            0.4|     -0.0572392|      -1.48462|        7.98821|
-----------------------------------------------------------------------------------------------
|              2|            0.4|     -0.0572392|       0.206598|      0.334745|        1.27706|
-----------------------------------------------------------------------------------------------
|              3|     -0.0572392|       0.206598|       0.158055|     0.0750927|        0.30713|
-----------------------------------------------------------------------------------------------
|              4|       0.206598|       0.158055|       0.144016|   -0.00584764|      0.0974821|
-----------------------------------------------------------------------------------------------
|              5|       0.158055|       0.144016|        0.14503|   9.00418e-05|     0.00699346|
-----------------------------------------------------------------------------------------------
|              6|       0.144016|        0.14503|       0.145015|   1.05241e-07|    0.000106063|
-----------------------------------------------------------------------------------------------
|              7|        0.14503|       0.145015|       0.145015|  -1.89782e-12|    1.24112e-07|
-----------------------------------------------------------------------------------------------
root=0.1450
```

End