University of Dhaka

Department of Computer Science and Engineering

CSE-3212: Numerical Methods Lab
3rd Year 2nd Semester
Session: 2017 -18

Name of the assignment: Bracketing Method

Submitted by:
Syeed Abrar Zaoad
Roll: SH-23

Submitted to:
Dr. Md.Haider Ali, Professor, Department of CSE,DU
Mr. Mubin Ul Haque,Lecturer, Department of CSE,DU

problem statement:

Problem 1:

Consider the following function

$$J_n(x) = \left(\frac{x}{2}\right)^n \sum_{k=0}^{\infty} \frac{(-1)^k (\frac{x^2}{4})^k}{k!(n+k)!}$$

In this case, first, you need to draw three graphs as following.
In graph 1: for n=0, and x = 0 to x = 10; increase x by 0.1 with value k starts from 1.
In graph 2: for n=1, and x = 0 to x = 10; increase x by 0.1 with value k starts from 1.
In graph 3: for n=2, and x = 0 to x = 10; increase x by 0.1 with value k starts from 1.
Secondly,
Using the series expansion for Jn(x), find its root for J0(x) to an accuracy of four decimal places
by using bisection method. Consider the initial guesses as 1 and 3. The desired level of accuracy is 0.00001.
At first, print the value of x and J0(x) from 1 to 3, increasing by 0.1. Then, ask the user for upper
bound and lower bound. If the root finding is possible, print the solution, otherwise print no root
is possible. You also need to print the following table in your console view.
iteration
Upper value | Lower value | Xm | f(X m) | Relative approximate error
Lastly,
Draw two graphs from above solution.
In graph 1: the graph of x and relative approximation error.
In graph 2: the graph of no of iteration and relative approximation error.

Problem 2:

Conservation of mass can be used to re-formulate the equilibrium relationship as

$$K = \frac{(c_{c,0} + x)}{(c_{a,0} - 2x)^2 (c_{b,0} - x)}$$

where the subscript 0 designates the initial concentration of each
constituent. If K = 0.016, ca,0 = 42, cb,0 = 28, and cc,0 = 4,
determine the value of x.

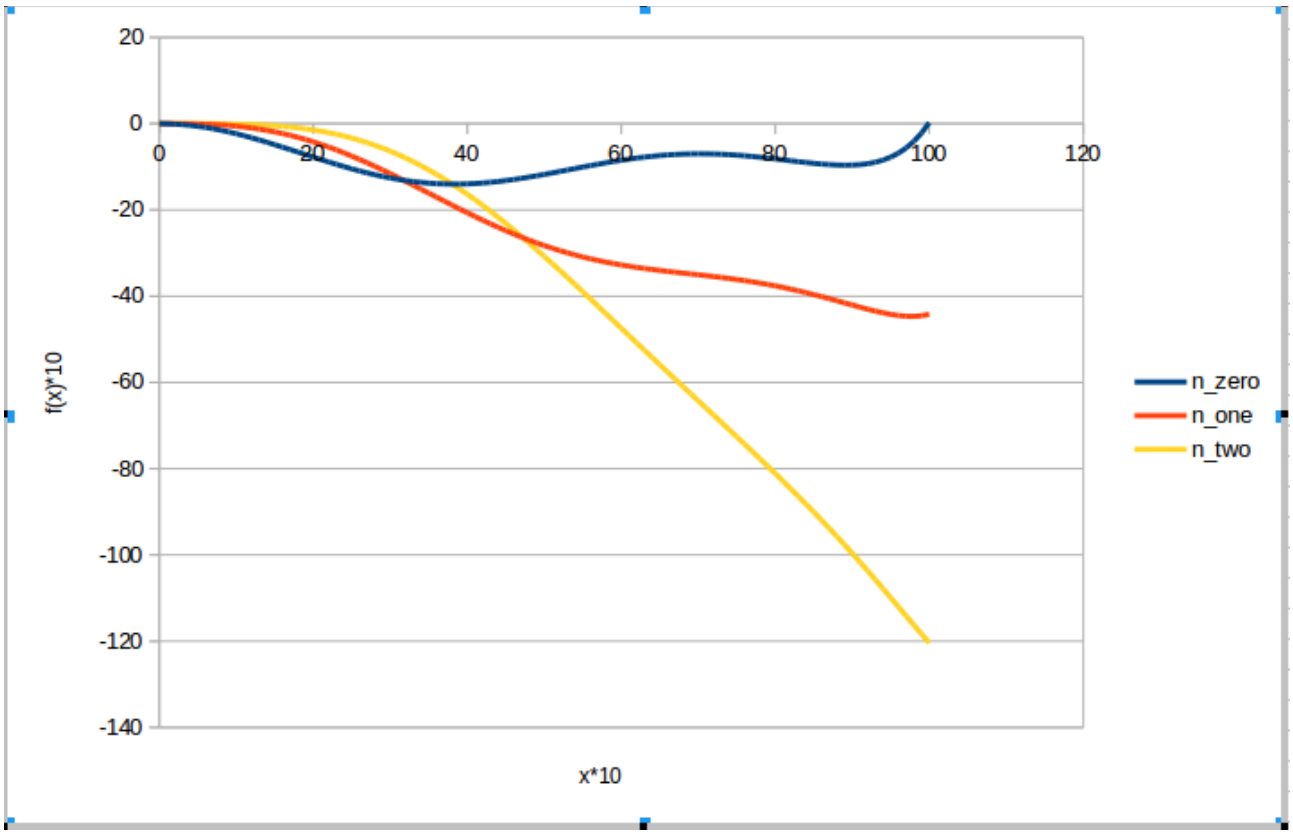
(a) Obtain the solution graphically by plotting the value 0 from 20, with increment of 1.
(b) On the basis of (a), solve for the root with initial guesses of xl = 0 and xu
= 20 with desired accuracy level of 0.00001 Choose false position to obtain your solution. Ask
the user for upper bound and lower bound. Justify your solution if it is not possible.
(c) Compare the relative approximate error between the bisection method and false position method. You need to use the previous problem (Problem 1) solution partially. For comparison,
you need to draw the graph of number of iteration and relative approximation error.
Talking with your classmates may result in deduction of evaluation points.

Solution:

source code of solution:
part1:

```
#include <bits/stdc++.h>
using namespace std;
#define loop(i,n) for(int i=1;i<=n;i++)
double fact[33];
void facto() {
        fact[0] = 1.0;
        loop(i,32) {
                fact[i] = fact[i-1] * (i*1.0);
        }
}
double doublepower(double base, int x) {
        double ret = 1.0;
        loop(i,x) ret *= base;
        return ret;
}
void setup()
{
   for(int i=0;i<33;i++)
   {
      fact[0]=0;
   }
}
double func(int n, double x) {
        double temp1=0.0
   ,sum = 0.0
         ,temp = doublepower(x/2.0, n);
        loop(k,10) {
                temp1= doublepower(-1.0, k);
                temp1*= doublepower((x*x)/4.0, k);
                temp1/= (fact[k] * fact[n + k]);
```

```
                sum += temp1;
        }
        return temp*sum;
}
void bisection(int n,double a,double b)
{
   if(func(n,a)*func(n,b)>=0)
   {
      cout<<"you have to assumed right a and b"<<endl;
   }
   double c;
   int it=1;
   while((b-a)>=0.0001)
   {
      c=(a+b)/2;
      if(func(n,c)==0.0)
         break;
      else if(func(n,c)*func(n,a)<0)
         b=c;
      else
         a=c;
      cout<<"iteration "<<it<<" a "<<a<<" b "<<b<<endl;
      it++;
   }
   cout<<"the value of root is : "<<c<<endl;
}
int main() {
        freopen("output1.csv", "w", stdout);
        facto();
        for(double x = 0.0; x<=10.0; x+=0.1) {
                double res1 = func(0, x);
                double res2= func(1,x);
                double res3= func(2,x);
                cout<<x*10<<" , "<<res1*10<<" , "<<res2*10<<" , "<<res3*10<<endl;
        }
        return 0;
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```



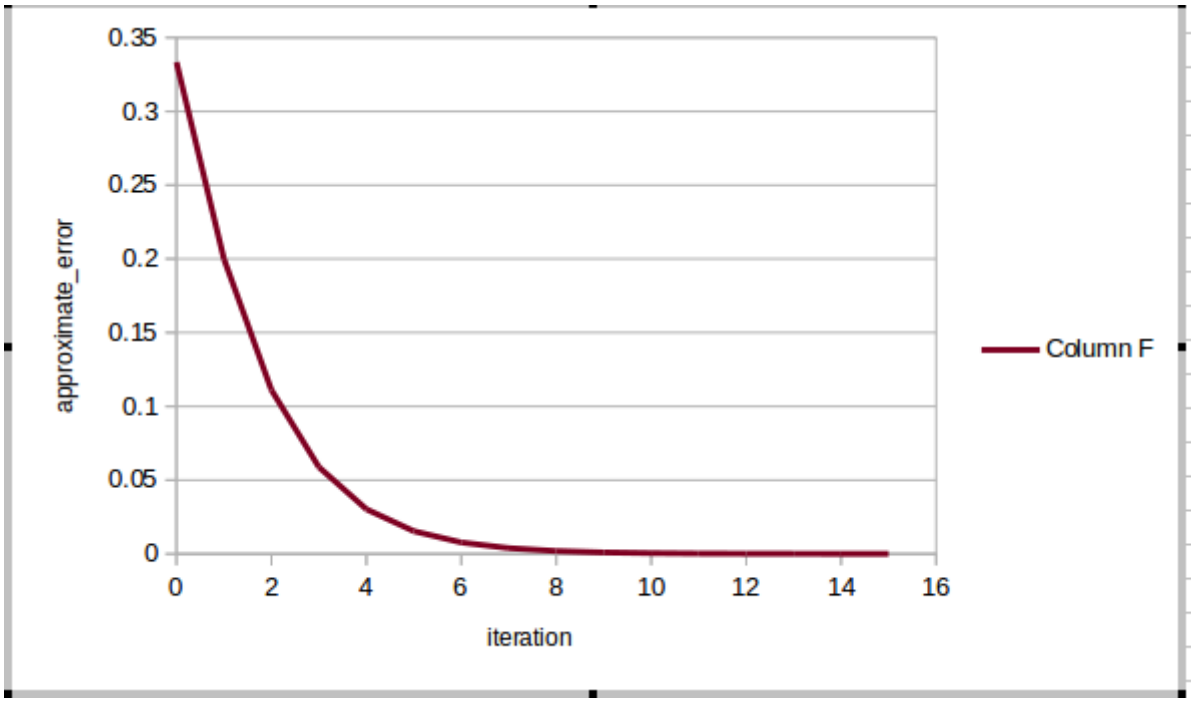
x	j0(x)	j1(x)	j2(x)
0	0	0	0
1	-0.0249844	-0.00062474	-1.04134e-05
2	-0.0997503	-0.00499167	-0.000166458
3	-0.223738	-0.0168118	-0.000841381
4	-0.396018	-0.0397342	-0.00265337
5	-0.615302	-0.0773154	-0.00645977
6	-0.879951	-0.13299	-0.013349
7	-1.18799	-0.210043	-0.0246306
8	-1.53713	-0.31158	-0.0418224
9	-1.92476	-0.440505	-0.066637
10	-2.34802	-0.599494	-0.100965
11	-2.80378	-0.790976	-0.146858
12	-3.28867	-1.01711	-0.20651
13	-3.79914	-1.27977	-0.282233
14	-4.33145	-1.58052	-0.376441
15	-4.88172	-1.92063	-0.491623
16	-5.44598	-2.30104	-0.630322
17	-6.02015	-2.72235	-0.795111
18	-6.60014	-3.18483	-0.988565
19	-7.18181	-3.68843	-1.21324
20	-7.76109	-4.23275	-1.47166
21	-8.33393	-4.81708	-1.76626
22	-8.89638	-5.44037	-2.09941
23	-9.4446	-6.10127	-2.47335
24	-9.97492	-6.79815	-2.8902
25	-10.4838	-7.52906	-3.35191
26	-10.968	-8.29182	-3.86027
27	-11.4245	-9.08399	-4.41688
28	-11.8504	-9.90291	-5.02315
29	-12.2431	-10.7457	-5.68023
30	-12.6005	-11.6094	-6.38909
31	-12.9206	-12.4908	-7.15043
32	-13.2019	-13.3866	-7.96472

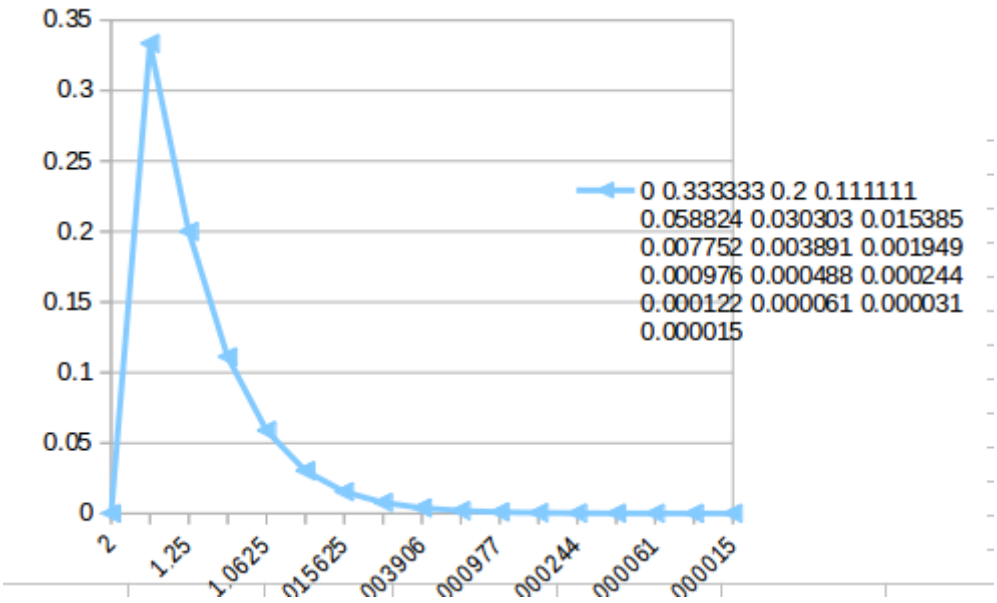
part2:

```
#include <bits/stdc++.h>
using namespace std;
#define loop(i,n) for(int i=1;i<=n;i++)
double fact[33];
void facto() {
        fact[0] = 1.0;
        loop(i,32) {
                fact[i] = fact[i-1] * (i*1.0);
        }
}
double doublepower(double base, int x) {
        double ret = 1.0;
        loop(i,x) ret *= base;
        return ret;
}
void setup()
{
   for(int i=0;i<33;i++)
   {
      fact[0]=0;
   }
}
double func(int n, double x) {
        double temp1= 0.0,
   sum = 0.0,
   temp = doublepower(x/2.0, n);
        loop(k,10) {
                temp1= doublepower(-1.0, k);
                temp1*= doublepower((x*x)/4.0, k);
                temp1/= (fact[k] * fact[n + k]);
                sum += temp1;
        }
        return temp*sum;
}
void bisection(int n,double a,double b)
{
   if(func(n,a)*func(n,b)>=0)
   {
      cout<<"you have to assumed right a and b"<<endl;
   }
   double c;
   int it=1;
   while((b-a)>=0.0001)
```

```
    {
        c=(a+b)/2;
        if(func(n,c)==0.0)
            break;
        else if(func(n,c)*func(n,a)<0)
            b=c;
        else
            a=c;
        cout<<"iteration "<<it<<" a "<<a<<" b "<<b<<endl;
        it++;
    }
    cout<<"the value of root is : "<<c<<endl;
}
void bisect(double lo ,double hi)
{   double px,acceptederror;
    int cnt, iteration = 0;
    while(true) {
                double mid = (hi + lo) / 2.0;
                double relativeerror = 0.0;

                if(iteration>1) {
                        relativeerror = fabs(mid - px) / mid;
                        if(relativeerror <= acceptederror) break;
                }
                cout<<fixed<<setprecision(6)<<iteration<<", "<<hi<<", "<<lo<<",
"<<mid<<", "<<func(0,mid)<<", "<<relativeerror<<endl;
                if(func(0,mid) > 0.0) lo = mid;
                else hi = mid;
                px = mid;
                iteration++;
        }
}
int main() {
        facto();
        double px, lo, hi, acceptederror;
        int cnt, iteration = 0;

        cin>>lo>>hi>>acceptederror;

    freopen("bisection.csv", "w", stdout);
    bisect(lo,hi);
        return 0;
}
```

```
Enter the tolerable error: 0.00001
iteration           hi          lo          xr          function(xr)          approximateError
-------------------------------------------------------------------------------------------------
1         3.000000      1.000000      2.000000      -0.776109         0.000000
2         2.000000      1.000000      1.500000      -0.488172         0.333333
3         1.500000      1.000000      1.250000      -0.354094         0.200000
4         1.250000      1.000000      1.125000      -0.292241         0.111111
5         1.125000      1.000000      1.062500      -0.262927         0.058824
6         1.062500      1.000000      1.031250      -0.248711         0.030303
7         1.031250      1.000000      1.015625      -0.241718         0.015385
8         1.015625      1.000000      1.007812      -0.238250         0.007752
9         1.007812      1.000000      1.003906      -0.236524         0.003891
10         1.003906      1.000000      1.001953      -0.235662         0.001949
11         1.001953      1.000000      1.000977      -0.235232         0.000976
12         1.000977      1.000000      1.000488      -0.235017         0.000488
13         1.000488      1.000000      1.000244      -0.234910         0.000244
14         1.000244      1.000000      1.000122      -0.234856         0.000122
15         1.000122      1.000000      1.000061      -0.234829         0.000061
16         1.000061      1.000000      1.000031      -0.234816         0.000031
17         1.000031      1.000000      1.000015      -0.234809         0.000015

Process returned 0 (0x0)   execution time : 5.987 s
Press ENTER to continue.
```





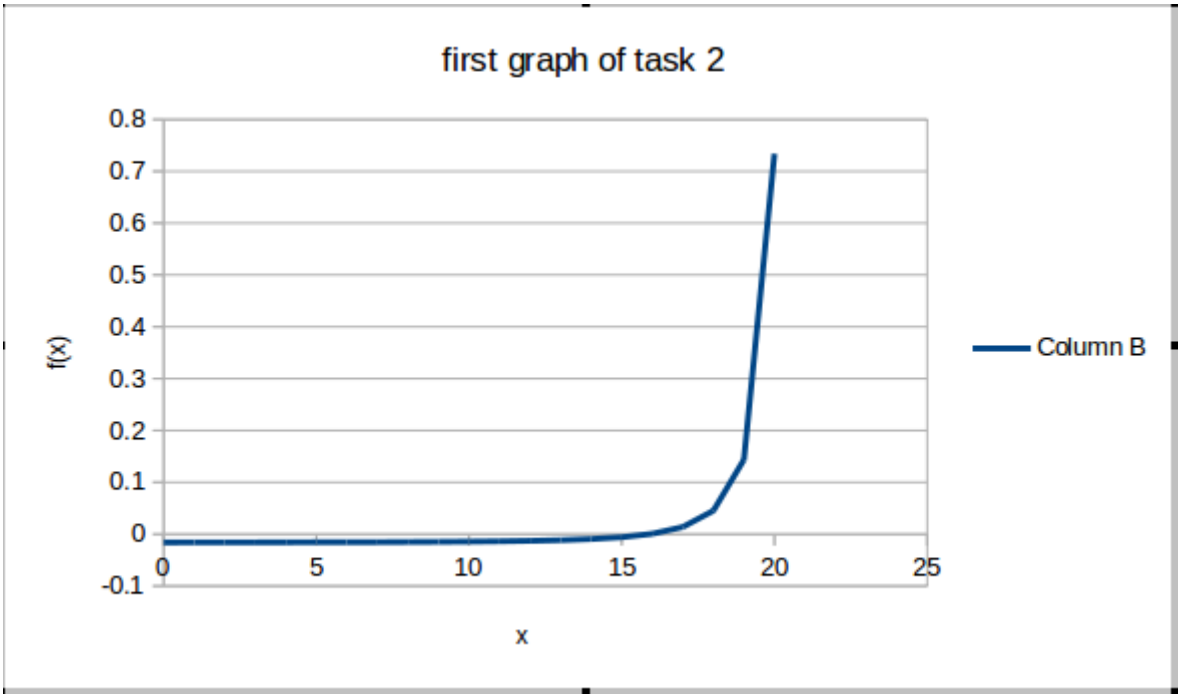
Problem 2:
part1:

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;

double fact[33];

double doublepower(double base, int x) {
        double ret = 1.0;
        for(int i=1; i<=x; i++)
        {
      ret *= base;
   }
        return ret;
}

double _function(double x) {
        double ret = 0.0;
        double ca0 = 42.0, cb0 = 28.0, cc0 = 4.0;
        double k = 0.016;
        ret = cc0 + x;
        ret /= ((ca0-2.0*x)*(ca0-2.0*x)*(cb0 - x));
        return ret-k;
}
void falseMethod(double xl,double xu)
{
   int it=1,e=1000;
   double xm,fu,fl,fm;
   do
```

```
    {
        fu=func(xu);
        fl=func(xl);
        if(fu*fl>0)
        {
            cout<<"x1 and x2 doesn't bracket the root"<<endl;
            return;

        }
        else
        {
            xm=xu-fu*(xu-xl)/(fu-fl);
            fm=func(xm);
            cout<<"xl "<<xl<<" xu "<<xu<<" f(xl) "<<fl<<" f(xu) "<<fu<<" xm "<<xm<<"
f(xm) "<<fm<<endl;
            e=fabs((xu-xl)/xu);
            if(fl*fm<0)
                xu=xm;
            else
                xl=xm;
                  }
        it++;
    }
    while(e>=0.0001&&fl!=0&&it!=100);
    cout<<"the root of the equation is : "<<xm<<endl;

}
int main() {
          freopen("out21.csv", "w", stdout);
          for(int x = 0; x<=20; x++) {
                    cout<<x<<", "<<_function(x)<<endl;
          }
          return 0;
}
```

output:

```
x              k(x)
----------------------
0        -0.015919
1        -0.0158843
2        -0.0158402
3        -0.015784
4        -0.0157116
5        -0.0156179
6        -0.0154949
7        -0.0153319
8        -0.0151124
9        -0.0148121
10        -0.014393
11        -0.0137941
12        -0.0129136
13        -0.0115729
14        -0.00944023
15        -0.00585043
16        0.000666667
17        0.0138295
18        0.0451111
19        0.143722
20        0.734

Process returned 0 (0x0)   execution time : 0.004 s
Press ENTER to continue.
```



part 2:

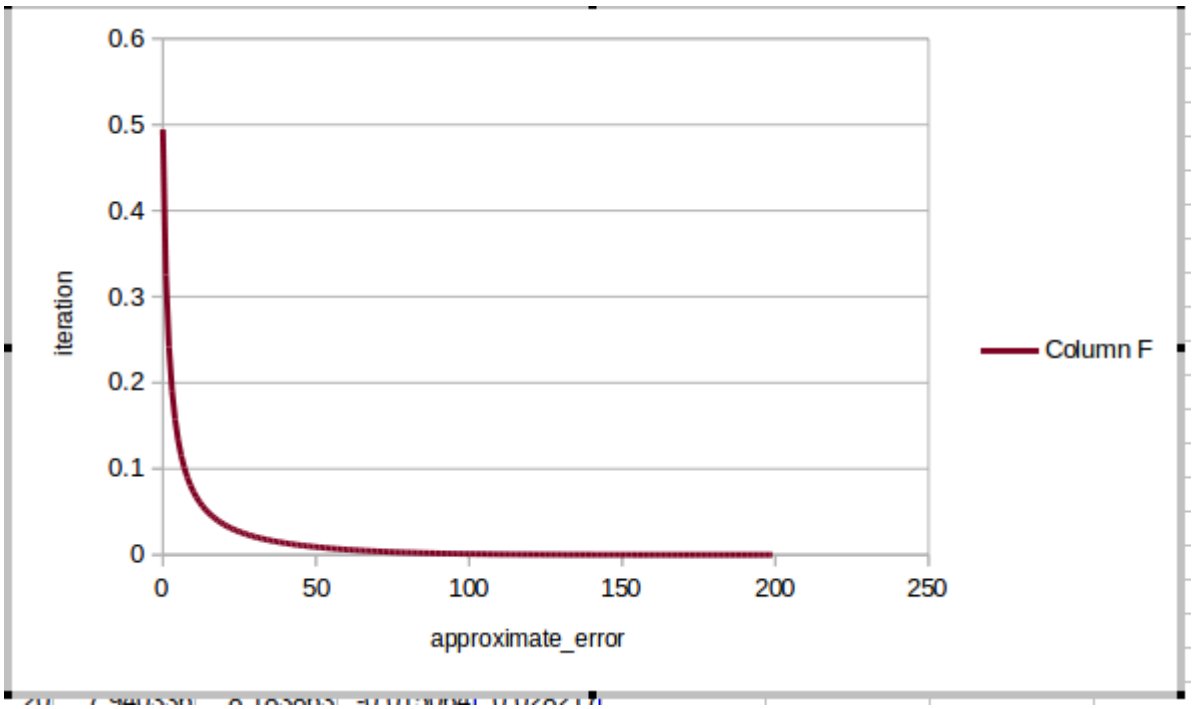
```
#include <bits/stdc++.h>
using namespace std;

typedef long long ll;

const int maxi = 311111;

double _function(double x) {
        double ret = 0.0;
        double ca0 = 42.0, cb0 = 28.0, cc0 = 4.0;
        double k = 0.016;
        ret = cc0 + x;
        ret /= ((ca0-2.0*x)*(ca0-2.0*x)*(cb0 - x));
        return ret-k;
}

double xm(double x0, double x1) {
        double num = _function(x0) * (x0 - x1);
        double den = _function(x0) - _function(x1);
        double ret = num / den;
        return ((-1.0 * ret) + x0);
}
void falseMethod(double xl,double xu)
{
   int it=1,e=1000;
   double xm,fu,fl,fm;
   do
   {
      fu=func(xu);
      fl=func(xl);
      if(fu*fl>0)
      {
         cout<<"x1 and x2 doesn't bracket the root"<<endl;
         return;

      }
      else
      {
         xm=xu-fu*(xu-xl)/(fu-fl);
         fm=func(xm);
         cout<<"xl "<<xl<<" xu "<<xu<<" f(xl) "<<fl<<" f(xu) "<<fu<<" xm "<<xm<<"
f(xm) "<<fm<<endl;
         e=fabs((xu-xl)/xu);
         if(fl*fm<0)
            xu=xm;
```

```
            else
                xl=xm;
                  }
          it++;
      }
      while(e>=0.0001&&fl!=0&&it!=100);
      cout<<"the root of the equation is : "<<xm<<endl;

}
int main() {

          freopen("out22.csv", "w", stdout);

          double prevx, lo, hi, accpErr;
          double inix0, inix1;
          int cnt, caseno = 1;

          cin>>lo>>hi>>accpErr;
          cnt = 3;
      falseMethod(lo,high);
          return 0;
}
```

output:

0 20 0.00001

iteration	hi	lo	xr	f(xr)	approximateerror
1	20.000000	0.000000	0.424553	-0.015905	0.000000
2	20.000000	0.424553	0.839742	-0.015890	0.494424
3	20.000000	0.839742	1.245753	-0.015874	0.325916
4	20.000000	1.245753	1.642769	-0.015857	0.241675
5	20.000000	1.642769	2.030968	-0.015839	0.191140
6	20.000000	2.030968	2.410523	-0.015819	0.157458
7	20.000000	2.410523	2.781605	-0.015797	0.133406
8	20.000000	2.781605	3.144378	-0.015775	0.115372
9	20.000000	3.144378	3.499006	-0.015750	0.101351
10	20.000000	3.499006	3.845646	-0.015724	0.090138
11	20.000000	3.845646	4.184453	-0.015696	0.080968
12	20.000000	4.184453	4.515579	-0.015666	0.073330
13	20.000000	4.515579	4.839169	-0.015635	0.066869
14	20.000000	4.839169	5.155370	-0.015601	0.061334
15	20.000000	5.155370	5.464321	-0.015565	0.056540
16	20.000000	5.464321	5.766159	-0.015527	0.052347
17	20.000000	5.766159	6.061020	-0.015486	0.048649
18	20.000000	6.061020	6.349035	-0.015443	0.045364
19	20.000000	6.349035	6.630331	-0.015398	0.042426
20	20.000000	6.630331	6.905035	-0.015349	0.039783
21	20.000000	6.905035	7.173269	-0.015298	0.037394
22	20.000000	7.173269	7.435153	-0.015245	0.035222
23	20.000000	7.435153	7.690804	-0.015188	0.033241
24	20.000000	7.690804	7.940336	-0.015127	0.031426
25	20.000000	7.940336	8.183863	-0.015064	0.029757
26	20.000000	8.183863	8.421493	-0.014998	0.028217
27	20.000000	8.421493	8.653335	-0.014927	0.026792
28	20.000000	8.653335	8.879493	-0.014854	0.025470
29	20.000000	8.879493	9.100071	-0.014776	0.024239
30	20.000000	9.100071	9.315170	-0.014695	0.023091
31	20.000000	9.315170	9.524889	-0.014610	0.022018
32	20.000000	9.524889	9.729325	-0.014521	0.021012
33	20.000000	9.729325	9.928573	-0.014428	0.020068
34	20.000000	9.928573	10.122728	-0.014331	0.019180
35	20.000000	10.122728	10.311881	-0.014229	0.018343
36	20.000000	10.311881	10.496122	-0.014123	0.017553
37	20.000000	10.496122	10.675541	-0.014013	0.016807
38	20.000000	10.675541	10.850225	-0.013899	0.016100
39	20.000000	10.850225	11.020261	-0.013780	0.015429
40	20.000000	11.020261	11.185733	-0.013656	0.014793



part 3:

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

const int maxi = 311111;

double _function(double x) {
	double ret = 0.0;
	double ca0 = 42.0, cb0 = 28.0, cc0 = 4.0;
	double k = 0.016;
	ret = cc0 + x;
	ret /= ((ca0-2.0*x)*(ca0-2.0*x)*(cb0 - x));
	return ret-k;
}
double xm(double x0, double x1) {
	double num = _function(x0) * (x0 - x1);
	double den = _function(x0) - _function(x1);
	double ret = num / den;
	return ((-1.0 * ret) + x0);
}
int main()
{
	double prevx, lo, hi, accpErr;
	int cnt, caseno = 1;
```

```
        cin>>lo>>hi>>accpErr;
    //freopen("bisection2.csv", "w", stdout);
        //swap(lo, hi);
    //printf("iteration        hi        lo        xr        function(xr)        approximateError\n");

//printf("-------------------------------------------------------------------------------------------\n");
        while(true) {
                double mid = (hi + lo) / 2.0;
                double relErr = 0.0;
                double temp;
                if(caseno>1) {
         if(mid==0.0) break;
                        relErr = fabs(mid - prevx);
                        double temp=relErr/mid;
                        cout<<mid<<" "<<prevx<<" "<<temp<<endl;
                        if(temp <= accpErr) break;
                }

                cout<<fixed<<setprecision(6)<<caseno<<"   ,   "<<hi<<"   ,   "<<lo<<"   ,
"<<mid<<"    ,  "<<_function(mid)<<"    ,    "<<temp<<endl;

                if(_function(mid) > 0.0) lo = mid;
                else hi = mid;
                prevx = mid;
                caseno++;
        }
        return 0;
}
```