Name : Manbir Singh
Roll no. 35551203116
Telco Dt.:
Pg.:

Experiment 1

Aim: To find out the root of the algebraic eqn. using Bisection Method.

Software Used: Turbo C++.

Algorithm

1. Start
2. Read a and b.
3. If $f(a) \cdot f(b) < 0$
   count = 0
   do
   c = (a+b)/2
   if (f(a) . f(c) < 0)
   b = c
   else
   a = c
   end if
   output 'root =', c
   output f(root), f(c)
   while((count <= 20) && (fc > eps) || (fc < -eps));
4. Output 'final root', c.
5. Output 'iterations', count.
6. STOP

$f(x) = x^3 - 9x + 1$

Output :

Enter the values of a & b : 2 3

Root = 2.5

Iteration : 1

f(root) = −5.875, Root = 2.75

Iteration : 2

f(root) = −2.953125, Root = 2.875

Iteration : 3

f(root) = −1.11132, Root = 2.9375

Iteration : 4

f(root) = −0.090088, Root = 2.953125

Iteration : 5

f(root) = 0.446259, Root = 2.953125

Iteration : 6

f(root) = 0.175922, Root = 2.945313

Iteration : 7

f(root) = 0.0423278, Root = 2.941406

Iteration : 8

f(root) = −0.02399, Root = 2.943359

Iteration : 9

f(root) = −0.007423, Root = 2.942383

Final (Root) = −0.007423, Root = 2.942383

final root = 2.942383

Name : Manbir Singh
Roll no. 35551203116

Telco Dt.: Pg.:

Source Code

```
#include <iostream.h>
#include <conio.h>
#include <math.h>
void main()
{
clrscr();
float a, b, count =0, fa, fb, c, fc;
cout << "Enter values of a & b : " <<endl;
cin >> a >> b;
fa = pow(a, 3) - 9 * a + 1;
fb = pow(b, 3) - 9 * b + 1;
if((fa * fb) < 0)
{
    count =0;
    do
        c = (a+b)/2;
        fc = pow(c, 3) - 9 * c + 1;
        if((fa * fc) < 0)
                b = c;
        else
            a = c;
        count = count + 1;
        cout << " Root is : " << c << endl;
        cout << " value of function at the root is : " << fc;
```

Name : Manbir Singh
Roll no. 35351203116

Telco Dt.:
Pg.:

```
    } while ((count <= 20) && ((fc > eps) || (fc < -eps)));
}

cout << "The final root is : " << c << endl;
cout << "No. of iterations are : " << count << endl;
getch();
}
```

15/15

Manbir Singh
23/01/18

Telco Dt.: Pg.:

Experiment 2

Aim : To find out the solution of algebraic & transcedental equation using Newton - Raphson Method.

Software Used: Turbo C.

Source Code / Algorithm

1. Start
2. Read a
3. Count = 0
4. do
   { if derf(a) > 0
   hl = f(a) / derf(a)
   c = a - h
   count = count + 1
   if (((c - a)/c) < eps) && fc == 0)
   break;
   else
   a = c;
   output root;
   output f(root);
   exit while loop
   }

Source Code

Output

Enter the value of a

2

Iteration no. 1 Value of function = 0.703703 root = 2.3333

" " 2 = 0.01935 root = 2.8005

" " 82 = 0.00016 root = 2.7900

" " 82 = 0.000001 root = 2.7900

```
#include <iostream.h>
#include <math.h>
void main()
{ clrscr();
  float a, count=0, fa, derfa, h, fc, c;
  cout << " Enter values of a :" << endl;
  cin >> a;
  fa = pow(a,3) - (3*a) - 5;
  derfa = 3 * pow(a,2) - 3;
  if(derfa > 0)
  do { fa = pow(0,3) - (3*a) - 5;
       derfa = 3 * pow(a,2) - 3;
       h1 = (fa/derfa);
       c = a - h1;
       count = count + 1;
       fc = pow(c,3) - (3*c) - 5;
       if((abs((c-a)/c) < eps) && fc == 0)
       break; else a = c;
       cout << "Root is :" << c << endl;
       cout << "Value of fn at root :" << fc << endl;
     } while((count <= 20) && (fc > eps) || (fc < -eps))
cout << "No. of iterations are :" << count << endl;
getch();
}
```

14/15

Marwadi 06/3/18

Telco Dt.: Pg.:

Experiment 3

Aim : To find out the solution of algebraic or transcendal eqn. using Regula-Falsi method.

Software Used : Turbo C++.

Algorithm

Start
Read a, b.
If $f(a) \cdot f(b) < 0$
  count = 0
  do
    $c = af(b) - bf(a) / f(b) - f(a)$
    if $f(a) \cdot f(c) < 0$
      $b = c$
    else $a = c$.
  end if count = count + 1
  while ((count <= 20) and (f(c) > eps)) or (f(c) <- eps)
  output root = 1, c
  output count, f(c).
else output 'interval incorrect'.
Return.

Source Code

Output

```
Root is 0.66667    iteration is 1    root is -0.37037
Root is 0.756757   iteration is 2    root is -0.53106
Root is 0.769023   iteration is 3    root is -0.007156
Root is 0.770664   iteration is 4    root is -0.009558
Root is 0.770883   iteration is 5    root is -0.000127
Root is 0.770912   iteration is 6    root is -1.713e-05
```

Telco Dt.: Pg.:

```
#include <iostream.h>
#include <math.h>
#define eps 0.0001
float func(float x)
{ float x;
  x = (x*x*x) + 2*x - 2;
  return x;
}
void main()
{
  clrscr();
  float a,b, count=0, fa, fb, fc, c;
  cout << "Enter values of a & b: " << endl;
  cin >> a >> b;
  fa = pow(b,3) + (2*b) - 2;
  if ((fa*fb)<0)
  { count = 0;
    do { fa = func(a);
         fb = func(b);
         c = (a*f(b) - (b*fa))/(fb - fa));
         fc = pow(c,3) + (2*c) - 2;
         if ((fa*fc)<0)   b = c;
         else a = c;
  count = count + 1;
  cout << "Root is: " << c << endl;
  cout << "Value of fn at root is " << fc << endl;
}
```

Telco Dt.: Pg.:

```
while ((count<=20) && (fc>eps)||(fc<-eps));
}
cout << " The final root is : << c << endl;
cout << " No. of iterations are : " << count << endl;
getch();
}
```

13/15

Manveen 06/03/18

Output

1. Enter the value of lower limit & upper limit & no. of iterations.

2. 505

The value of function is 3.03333

Page No.
Date

Experiment 4

- Aim: To find the solution of equation using trapezoidal rule.

- Software Used: Turbo C.

- Algorithm

1. Start
2. Define a fn. float f(c) & return 1/1+x.
3. Enter the lowest limit, upper limit and no. of iterations.
4. h = (b-a)/n
5. for (i = a+h; i <= b-h; i += h)
   p += f(i).
6. d = ((0.5 * h) * (y f(a) + f(b) + (2*p)))
7. Value of the integral is d.

- Source Code

```
#include <iostream.h>
#include <conio.h>
using namespace std;
float f(float x){
        return 1/(1+x);
}
```

```
int main() {
    float a, b, ite, h, p = 0.0, d = 0.0;
    int i;
    cout << " Enter the value of lower limit, upper
            limit & no. of iteration " << endl;
    cin >> a >> b >> ite;
    h = (b-a)/ite;

    for (i = a+h; i <= b-h; i += h) {
                p += f(i)
            }

    d = ((h * 0.5) * f(a) + f(b) + (2 * p)));
     cout << " Value of integration is : " << d << endl;
}
```

14/15

Monisha
06/03/18

Teleco DL.: Pg.:

Experiment 5

- **Aim**: To find the solution of equation using Simpson's 1/3rd Rule.
- **Software Used**: Turbo C
- **Algorithm**

1. START
2. Define a function float func and return $\sqrt{x} \times e^{-x}$.
3. Enter the lower limit, upper limit & no. of iterations.
4. h = (b-a)/n
5. for(i = a+h; i<b; i+=2*h)
   s1 = s1 + func(i)
6. for(i = a+2*h; i<b; i+=2*h)
   s2 = s2 + func(i)
7. d = h*(func(a) + func(b) + 4*s1 + 2*s2)/3
8. Output, 'Value of Integral'; d
9. Stop

Source Code

```
#include <iostream.h>
#include <conio.h>
#include<math.h>
float func(float x)
{
```

```
    float fx;
    fx = pow(x, 0.5) * exp(-x);
    return fx;
}
void main()
{
float a, b, y[20], n, h, i, s1, s2, d;
clrscr();
cout<<" Enter lower limit & upper limit "<<endl;
cin>>a>>b;
cout<<" Enter no. of iterations : "<<endl;
cin>>n;
h = (b-a)/n;
for(i = a+h ; i<b; i+= 2*h)
    s1 = s1 + func(i);
for(i = a+2*h; i<b; i+2*h)
    s2 = s2 + func(i);
d = h * (func(a) + func(b) + 4*s1 + 2*s2)/3
cout<<" Value of integral is : "<<d<<endl;
getch();
}
```

14/15

Manuscript 06/03/18

Telco Dt.: Pg.:

Experiment 6

- **Aim**: To find the integral of the given eqn. using Simpson's 3/8th rule.
- **Software Used**: Turbo C++.
- **Algorithm**

1. Start
2. Define a function float func & return $1/1+x^2$.
3. Enter the lower limit, upper limit & no. of iterations.
4. $h = (b - a)/n$
5. for (i = 1; i < n; i++)
   { if (i % 3 == 0)
   s1 = s1 + func(a + i * h)
   else
   s2 = s2 + func(a + i * h)
6. $d = h \times (func(a) + func(b) + 3 * s2 + 2 * s1) * 3/8$.
7. Output, 'value of integral', d
8. Stop.

- **Source Code**

```
#include <iostream.h>
# include <conio.h>
float f(float x)
```

Output

Enter the lower & upper limits :
0
6
Enter the no. of iterations :
6
The value of integral is :
1.357081

```
{
  return (1/(1+x*x));
}
void main()
{
  int i,n;
  float a,b,h,s=0;
  cout<<f(x) = 1/(1+x^2)"<<endl;
  cout<<" Enter the value of lower limit :"<<endl;
  cin>>a;
  cout<<" Enter the value of upper limit"<<endl;
  cin>>b;
  cout<<" Enter the no. of iterations :"<<endl;
  cin>>n;
  h=(b-a)/n;
  s=f(a)+f(b);
  for(i=1; i<n; i++)
  {
    if(i%3==0)
    { s+=2*f(a+(i*h)); }
    else
    { s+=3*f(a+(i*h)); }
  s=(3*s*h)/8;
  cout<<" Simpson 3/8 is "<<s<<endl;
  getch();
}
```

Manual 13/03/18

Experiment 7

- **Aim**: To find the solution of the system of linear equations using Gauss Jordan Method.
- **Software Used**: Turbo C++.
- **Algorithm**

1. Input variables as a matrix of form by loop

$$\begin{matrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \end{matrix}$$

2. Make $a_1 = 1$, $b_1 = b_1/a$, $c_1 = c_1/a$, $d_1 = d_1/a$.
3. Make $a_2 = 0$, $a_3 = 0$ by elementary row operations.
4. In a similar way, make $b_2 = 1$ & $b_1 = 0$ & $b_3 = 0$ & repeat same for $c_3 = 1$, $c_1 = 0$, $c_2 = 0$.
5. Print matrix obtained after all the transformations.
6. d_1, d_2, d_3 are the required solutions of x, y, z respectively.

- **Source Code**

Output

```
Enter the size of Square matrix

3

Enter the elements of matrix

1
1
1
5
3
3
5
8
4
0
5
2

The solution is :

3
4
-2
```

```
#include <conio.h>
#include <math.h>
void main()
{
  clrscr();
  int i, j, k, n;
  float a[20][20], c, x[10];
  cout << "Enter the size of Square matrix : ";
  cin >> n;
  cout << "Enter the elements of matrix : " << endl;
  for(i=1; j<=n; j++)
  {
    for(j=1; j<=n+1; j++)
    {
      cin >> a[i][j]; }
}
for (j=1; j<n; j++)
{ for (i=1; i<=n; i++)
    { if (i != j)
      { c = a[i][j] / a[j][j];
        for (k=1; k<=n+1; k++)
        { a[i][k] = a[i][k] - c * a[j][k];
          }}
}}
cout << "The Soln. is : " << endl;
for (i=1; i<=n; i++) {
        x[i] = a[i][n+1] / a[i][i];
        cout << x[i] << endl; }
getch();
```

15/15

20/03/18

Experiment 8

- **Aim**: To perform algebra of matrices : addition, subtraction, multiplication & transpose.
- **Software Used** : Turbo C++.
- **Algorithm**: Start
  Enter no. of rows & columns
  Enter element of matrix
  for(i=1; i<=r; i++)
  { for(j=1; j<=c; j++)
  }
  Enter a[i][j] & b[i][j]

Addition

```
for(i=0; i<r; i++)
{ for(j=0; j<c; j++)
  { sum[i][j]=a[i][j]
              +b[i][j]
```

Subtraction

```
for(i=1; i<=r; i++)
{ for(j=1; j<=c; j++)
  { diff[i][j]=a[i][j]
              -b[i][j]
```

Multiplication

```
if(r==c){
  for(i=0; i<r; i++)
  { for(j=0; j<c; j++)
    { mul[i][j]=0;
      mul[i][j]+=a[i][j]*b[j][i];
```

Output

```
Enter no. of rows : 3
Enter no. of columns : 3
Enter elements of 1st matrix :
1 2 3 4 5 6 7 8 9
Enter elements of second matrix :
5 6 7 8 9 10 11 12 13
```

Addition

```
6   8   10
12  14  16
18  20  22
```

Subtraction

```
-4  -4  -4
-4  -4  -4
-4  -4  -4
```

Multiplication

```
54   60   66
126  141  156
198  222  246
```

Transpose

```
1  4  7
2  5  8
3  6  9
```

```
#include <iostream.h>
#include <conio.h>
void main()
{
int a[20][20], b[20][20], i, j, c, k, m, n, x, y, b[20][20];
clrscr();
cout << "Enter dimension of 1st matrix";
cin >> m >> n;
cout << "Enter the first matrix :";
for (i=1; i<=m; j++)
{
  for (j=1; j<=n; j++)
  {
    cin >> a[i][j];
  }
}
cout << "Enter the dimension of 2nd matrix :";
cin >> x >> y;
if (m==x && n==y)
{
  cout << "Enter the second matrix :";
  for (i=1; i<=x; i++)
  {
    for (j=1; j<=y; j++)
    { cin >> b[i][j];
    }
}
```

```
cout << " Enter your choice : ";
cin >> c;
switch (c)
{  case 1 : for (i=1; i<=m; i++)
            { for (j=1; j<=n; j++)
              { cout << a [i][j]+b [i] [j]<< " \+ ";
              }
              cout << endl;
            } break;
  case 2 : for (i=1; i<=m; i++)
           { for(j=1; j<=n; j++)
             { cout << a[i][j]-b [i][j] << " \t";
             }
             cout << endl;
           } break;
  case 3 : cout << " Transpose of first matrix : ";
           for(i=1; i<=m; i++)
           { for(j = 1; j<= n; j++)
             { cout << a [j][i] << " \t";
             } cout << endl; }
           cout << " Transpose of 2nd matrix : ";
           for(i = 1; i<=m; i++)
           { for(j =1; j<=n; j++)
             { cout << b[j] [i] << " \t";
             } cout << endl;
             } break;
```

```
case 4:
        for(i=1; i<=m; i++)
        { for(j=1; j<=y; j++)
          { p[i][j] = 0;
        for(k=1; k<=n; k++)
        {
          p[i][j] = p[i][j] + a[i][k] * b[k][j];
          }}}
        for(i=1; i<=m; i++)
        { for(i=1; i<=m; i++)
          { for(j=1; j<=y; j++)
            {
              cout << p[i][j];
              }
          }
        break;
}}
else
    {
    cout << "Invalid";
    }
    getch();
}
```

15/15

Manuchu
27/03/18

Page No.
Date

Experiment 9

- **Aim** : To solve the given ordinary diff. eqn. using Runge Kutta Method.
- **S/w Used** : Turbo C++
- **Algorithm**

1. Start
2. Define a function float f and return y-x.
3. Enter values for x_0, y_0 & $h = 0.1$.
4. $k_1 = h * f(x_0, y_0)$
   $k_2 = h * f((x_0 + h/2)), (y_0 + (k_1/2))$;
   $k_3 = h * f((x_0 + h/2)), (y_0 + (k_2/2))$;
   $k_4 = h * f((x_0 + h), (y_0 + k_3))$;
   $y_1 = y_0 + (k_1 + 2 * k_2 + 2 * k_3 + k_4) / 6$;
5. Output 'Value of differential eqn.', y1
6. Stop.

Output

```
Enter x0
0
Enter y0
2
Enter h
0.1
Value of differential eqn. is : 2.205171
```

Page No.
Date

Source Code

```
#include <iostream.h>
#include <conio.h>
#include <math.h>
float func(float x, float y)
{
float fx;
fx = y-x;
return fx;
}
void main()
{
 float x0, y0, h, k1, k2, k3, k4, y1;
 clrscr();
 cout << "Enter the value of x0, y0 : ";
 cin >> x0, y0;
 cout << "Enter the value of h : ";
 cin >> h;
 k1 = h * func(x0, y0);
 k2 = h * func(x0+h/2, y0+k1/2);
 k3 = h * func(x0+h/2, y0+k2/2);
 k4 = h * func(x0+h, y0+k3);
 y1 = y0+(k1+2*k2+2*k3+k4)/6;
 cout << y1;
 getch();
}
```

13.5/15

My answer is correct
03/04/18

Experiment 10

- **Aim** : To solve the initial value problem using Euler's method.

- **Software Used** : Turbo C++

- **Algorithm**

1. Start
2. Define a function f & return x^3+y
3. Enter values for x_0, y_0 & h.
4. for(i =0 ; i <=n ; i++)
   { y [i+1] = y [i] + n * fxc[i], y [i]);
   }
5. Output 'Value :', y [n]
6. Stop

- **Source Code**

```
#include < iostream.h >
#include < conio.h >
#include <math.h>

float f(float x, float y)
{
 float fx;
 fx = pow(x,3) + y;
```

Output

Enter value of x_0
0

Enter value of y_0
1

Enter h
0.01

Value = 1.0201

```
    return fx;
}
void main()
{
float x0, x1, y0, y1, y2, h;
clrscr();
cout << "Enter the value of h: ";
cin >> h;
cout << "Enter the value of x0, y0: ";
cin >> x0 >> y0;
x1 = x0 + h;
y1 = y0 + h * func(x0, y0);
y2 = y1 + h * fun(x1, y1);
cout << y2;
getch();
}
```

Manisha
08/04/18

13.5/15