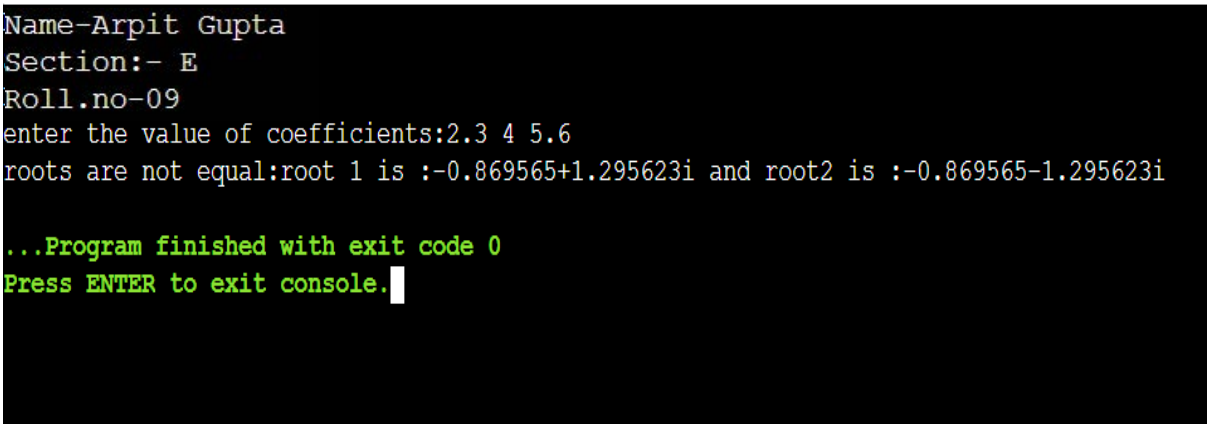
ROOTS OF AN EQUATION:-

```
#include<stdio.h>
#include<math.h>
int main(){
printf(“Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n”);
double a,b,c,d,root1,root2,imag,real;
printf("enter the value of coefficients:");
scanf("%lf%lf%lf",&a,&b,&c);
d=b*b-4*a*c;
if(d>0){
printf("roots are real and both are different:");
root1=(-b+sqrt(d))/(2*a);
root1=(-b-sqrt(d))/(2*a);
printf("roots are:%lf %lf",root1,root2);
}
else if(d==0)
{
printf("roots are real and equal");
root1=root2=-b/(2*a);
printf("roots are:%lf %lf",root1,root2);
}
else
{
printf("roots are not equal:");
real=-b/(2*a);
imag=sqrt(-d)/(2*a);
printf("root 1 is :%lf+%lfi and root2 is :%lf-%lfi",real,imag,real,imag);
}
return 0;
}
```

OUTPUT: -


Name-Arpit Gupta
Section:- E
Roll.no-09
enter the value of coefficients:2.3 4 5.6
roots are not equal:root 1 is :-0.869565+1.295623i and root2 is :-0.869565-1.295623i
...Program finished with exit code 0
Press ENTER to exit console.

BISECTION METHOD:-

```
#include<stdio.h>
#include<math.h>
double equation(float x)
{
return x*x*x-4*x-9;
}
double c;
double e=0.001;
void bisection(double a,double b){
if(equation(a)*equation(b)>=0)
{
printf("invalid");
return;
}
c=a;
while((b-a)>=e){
c=(a+b)/2;
if(equation(c)==0.0)
{
printf("%lf",c);
break;
}
else if(equation(c)*equation(a)<0.0)
{
printf("root is =%lf",c);
b=c;}
```

```
else
{printf("root is =%lf",c);
a=c;}
}
}
int main()
{
printf(“Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n”);
double a,b;
printf("enter a and b");
scanf("%lf%lf",&a,&b);
bisection(a,b);
printf("final root is %lf",c);
return 0;
}
```

OUTPUT:-

```
Name-Arpit Gupta
Section:- E
Roll.no-09
enter a and b2 3
root is =2.500000root is =2.750000root is =2.625000root is =2.687500root is =2.718750root is =2.703125root is =2.710
938root is =2.707031root is =2.705078root is =2.706055final root is 2.706055

...Program finished with exit code 0
Press ENTER to exit console.
```

REGULAR FALSI METHOD:-

```
#include<stdio.h>
#include<math.h>
#define e 0.0001
double fun(double);
void falsi(double,double);
int main(){
printf(“Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n”);
double a,b;
printf("enter the values of a and b:");
scanf("%lf%lf",&a,&b);
falsi(a,b);
return 0;}
double fun(double x){
return x*x*x-4*x-9;}
void falsi(double a,double b){
if(fun(a)*fun(b)>=0){
printf("invalid");
return;}
double c=a;
while((b-a)>=e)
{
c=(a*fun(b)-b*fun(a))/(fun(b)-fun(a));
if(fun(c)==0.0)
break;
else if(fun(c)*fun(a)<0)
b=c;
else
a=c;
printf("value is:%lf\n",c);
}}
```

OUTPUT:-

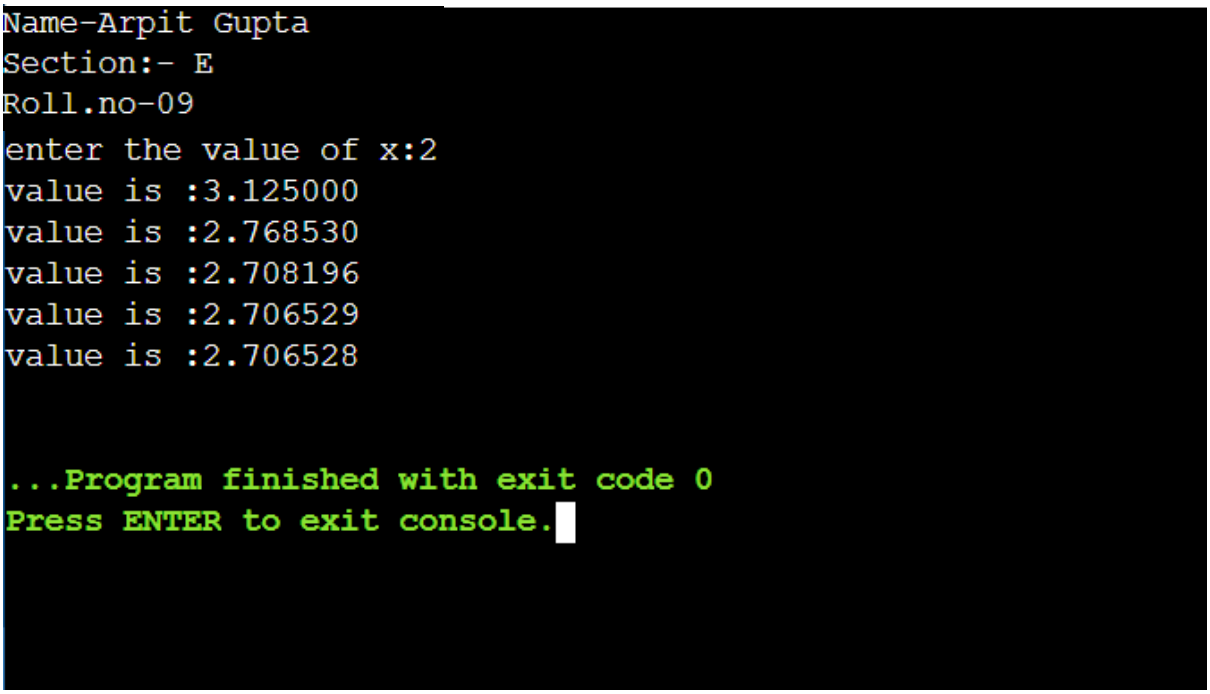
```
Name-Arpit Gupta
Section:- E
Roll.no-09
enter the values of a and b:2 3
value is:2.600000
value is:2.693252
value is:2.704918
value is:2.706333
value is:2.706504
value is:2.706525
value is:2.706528
value is:2.706528
value is:2.706528
value is:2.706528
value is:2.706528
value is:2.706528
value is:2.706528
value is:2.706528
value is:2.706528
value is:2.706528
value is:2.706528
value is:2.706528
value is:2.706528


...Program finished with exit code 0
Press ENTER to exit console.
```

NEWTON RAPHSON METHOD:-

```
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
#define e 0.0001
double fun(double);
double derivative(double);
void newton(double);
int main(){
printf(“Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n”);
   double x1;
printf("enter the value of x:");
scanf("%lf",&x1);
newton(x1);
return 0;}
double fun(double x){
return x*x*x-4*x-9;}
double derivative(double x){
return 3*x*x-4;}
void newton(double x)
{if(derivative(x)==0){
printf("derivative cannot be negative");
exit(0);}
else
{double c=fun(x)/derivative(x);
while(fabs(c)>=e)
{
c=fun(x)/derivative(x);
x=x-c;
printf("value is :%lf\n",x);}
}}
```

OUTPUT:-


Name-Arpit Gupta
Section:- E
Roll.no-09
enter the value of x:2
value is :3.125000
value is :2.768530
value is :2.708196
value is :2.706529
value is :2.706528
...Program finished with exit code 0
Press ENTER to exit console.

GAUSS ELIMINATION METHOD

```
#include<stdio.h>
int main()
{
printf(“Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n”);
  int i,j,k,n;
 float A[20][20],d,x[10],sum=0.0;
 printf("\nEnter the number of equations ");
 scanf("%d",&n);
 printf("\nEnter the elements of augemented matrix:\n\n");
for(i=1; i<=n; i++)
 {
 for(j=1; j<=(n+1); j++)
 {
 printf("A[%d][%d] : ", i,j);
 scanf("%f",&A[i][j]);
 }
 }
 for(j=1; j<=n; j++)
 {
 for(i=1; i<=n; i++)
 {
 if(i>j)
 {
 d=A[i][j]/A[j][j];
 for(k=1; k<=n+1; k++)
 {
 A[i][k]=A[i][k]-d*A[j][k];
 } }
 }
 x[n]=A[n][n+1]/A[n][n];
```

```
 for(i=n-1; i>=1; i--)
 {
 sum=0;
 for(j=i+1; j<=n; j++)
 {
 sum=sum+A[i][j]*x[j];
 }
 x[i]=(A[i][n+1]-sum)/A[i][i];
 }
 printf("\nThe required solution is: \n");
 for(i=1; i<=n; i++)
 {
 printf("\nx%d=%f\t",i,x[i]);
 }
 return(0);
}
```

OUTPUT:-

```
Name-Arpit Gupta
Section:- E
Roll.no-09

Enter the number of equations 3

Enter the elements of augemented matrix:

A[1][1] : 1
A[1][2] : -1
A[1][3] : 2
A[1][4] : 3
A[2][1] : 1
A[2][2] : 2
A[2][3] : 3
A[2][4] : 5
A[3][1] : 3
A[3][2] : -4
A[3][3] : -5
A[3][4] : -13

The required solution is:

x1=-1.000000
x2=0.000000
x3=2.000000

...Program finished with exit code 0
Press ENTER to exit console.
```

GAUSS JORDAN METHOD

```
#include<stdio.h>
int main(){
printf(“Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n”);
  int i,j,k,n;
 float A[20][20],c,x[10];
 printf("\nEnter the number of equations: ");
 scanf("%d",&n);
 printf("\nEnter the elements of augemented matrix:\n");
for(i=1; i<=n; i++){
 for(j=1; j<=(n+1); j++) {
 printf(" A[%d][%d]:", i,j);
 scanf("%f",&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];}
 }
 }
 }
 printf("\nThe solution is:\n");
 for(i=1; i<=n; i++)
 {
 x[i]=A[i][n+1]/A[i][i];
 printf("\n x%d=%f\n",i,x[i]);
 }
 return(0);
}
```

OUTPUT:-

```
Name-Arpit Gupta
Section:- E
Roll.no-09

Enter the number of equations 3

Enter the elements of augemented matrix:

A[1][1] : 1
A[1][2] : -1
A[1][3] : 2
A[1][4] : 3
A[2][1] : 1
A[2][2] : 2
A[2][3] : 3
A[2][4] : 5
A[3][1] : 3
A[3][2] : -4
A[3][3] : -5
A[3][4] : -13

The required solution is:

x1=-1.000000
x2=0.000000
x3=2.000000

...Program finished with exit code 0
Press ENTER to exit console.
```

GAUSS SIEDAL METHOD

```
#include<stdio.h>
#include<math.h>
#include<stdbool.h>
#define EPSILON 0.001
int n;
int flag;
float findSum(int i,float a[][n+1])
{
 float sum=0;
 for(int j=0;j<n;j++)
 {
 if(i!=j)
 sum+=a[i][j];
 }
 return sum;
}
bool isMethodApplicable(float a[][n+1])
{
 for(int i=0;i<n;i++)
 {
 for(int j=0;j<n;j++)
 {
 if(fabs(a[i][i])>findSum(i,a))
 continue;
 else
 return false;
 }}
 return true;
}
void print(int iteration,float values[n])
{
```

```
 printf("Iteration %d ",iteration);
 for(int i=0;i<n;i++)
 printf("value[%d]=%f ",i+1,values[i]);
 printf("\n");}
void findValues(float a[][n+1],int maxIterations,float values_old[n]){
 int i,j,k,iteration;
 float ratio,sum=0;
 float values_new[n];
 for(int i=0;i<n;i++)
 values_new[i]=0;
 for(iteration=1;iteration<=maxIterations;iteration++)
 {
 for(i=0;i<n;i++)
 {
 sum=0;
 for(j=0;j<n;j++)
 {
 if(i!=j)
 sum+=a[i][j]*values_new[j];
 }
 values_new[i]=(a[i][n] - sum)/a[i][i];
 }
 for(k=0;k<n;k++){
 if(fabs(values_old[k]-values_new[k])<EPSILON)
 continue;
 else{
 flag=1;
 break;
 }}
 if(flag==0){
 print(iteration,values_new);
 return ;
 }
```

```
 flag=0;
 print(iteration,values_new);
 for(k=0;k<n;k++)
 values_old[k]=values_new[k];
 }
 print(iteration,values_new) ;
}
int main(){
printf(“Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n”);
 int i,j,k,x,y,maxIterations;
 float ratio;
 printf("Enter no of Unknowns\n");
 scanf("%d",&n);
 printf("Enter no. of iterations\n");
 scanf("%d",&maxIterations);
 float a[n][n+1];
 float values[n];;
 printf("Enter the Augmented Matrix\n");
 for(int i=0;i<n;i++)
 {
 for(int j=0;j<n+1;j++)
 scanf("%f",&a[i][j]);
 }
 if(!isMethodApplicable(a)) {
 printf("Gauss Seidel Method can't be applied");
 return 0;
 }
printf("\n\nGauss Seidel Method is applicable\n");
 for(int i=0;i<n;i++)
 values[i]=0;
 findValues(a,maxIterations,values);
 return 0;
}
```

OUTPUT:-

```
Name-Arpit Gupta
Section:- E
Roll.no-09
Enter no of Unknowns
3
Enter no. of iterations
4
Enter the Augmented Matrix
8
-3
2
20
4
11
-1
33
6
3
12
35


Gauss Seidel Method is applicable
Iteration 1 value[1]=2.500000 value[2]=2.090909 value[3]=1.143939
Iteration 2 value[1]=2.998106 value[2]=2.013774 value[3]=0.914170
Iteration 3 value[1]=3.026623 value[2]=1.982516 value[3]=0.907726
Iteration 4 value[1]=3.016512 value[2]=1.985607 value[3]=0.912009
Iteration 5 value[1]=3.016512 value[2]=1.985607 value[3]=0.912009


...Program finished with exit code 0
Press ENTER to exit console.
```

NEWTON FORWARD INTERPOLATION

```
#include<stdio.h>
#include<math.h>
int main()
{
printf(“Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n”);
float x[10],y[10][10],h,u1,u,fx,fy,fact;
 int i,j,n,ch=30;
 printf("How many terms you want to enter : ");
 scanf("%d",&n);
 printf("Enter the value of X and Y (X,Y)");
 for(i=0;i<n;i++)
 {
 scanf("%f %f",&x[i],&y[i]);
 }
 for(j=1;j<n;j++)
 {
 for(i=0;i<n-j;i++)
 {
 y[i][j]=y[i+1][j-1]-y[i][j-1];
 }
 }
 printf("-------Difference Table-------\n");
 printf("X Y");
 for(i=0;i<n-1;i++)
 printf("\t%c^%d",ch,i+1);
 printf("\n");
 for(i=0;i<n;i++)
 { printf("%.2f",x[i]);
 j=0;
 while(j<n-i)
```

```
{
printf(" %.3f",y[i][j]);
j++;
}
printf("\n");
}
printf("\nEnter the value of x for which you wants to find Y : ");
scanf("%f",&fx);
h=x[1]-x[0];
u=(fx-x[0])/h;
fy=y[0][0];
u1=u;
fact=1;
for(i=1;i<n;i++)
{ fy=fy+(u1*y[0][i])/fact;
u1=u1*(u1-i);
fact=fact*(i+1);
}
printf("\n Y(%f)=%.3f",fx,fy); }
```

OUTPUT:-

```
Name-Arpit Gupta
Section:- E
Roll.no-09
How many terms you want to enter : 5
Enter the value of X and Y (X,Y):
45 114.84
50 96.16
55 83.32
60 74.48
65 68.48
-------Difference Table-------
X Y      ^1       ^2       ^3       ^4
45.00 114.840 -18.680 5.840 -1.840 0.680
50.00 96.160 -12.840 4.000 -1.160
55.00 83.320 -8.840 2.840
60.00 74.480 -6.000
65.00 68.480

Enter the value of x for which you wants to find Y : 46

 Y(46.000000)=110.505

...Program finished with exit code 0
Press ENTER to exit console.
```

NEWTON BACKWARD INTERPOLATION

```
#include<stdio.h>
#include<math.h>
int main()
{
printf(“Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n”);
float x,u1,u,y;
 int i,j,n,fact;
 printf("Enter no. of terms\n");
 scanf("%d",&n);
 float a[n][n+1];
 printf("Enter Values of X \n");
 for(i=0;i<n;i++)
 scanf("%f",&a[i][0]);
 printf("Enter Values of Y\n");
 for(i=0;i<n;i++)
 scanf("%f",&a[i][1]);

 printf("Enter value of x for which you want y\n");
 scanf("%f",&x);

 for(j=2;j<n+1;j++)
 {
 for(i=0;i<n-j+1;i++)
 a[i][j] = a[i+1][j-1]-a[i][j-1];
 }
```

```
printf("The Difference Table is as follows:\n");
for(i=0;i<n;i++)
{
for(j=0;j<=n-i;j++)
printf("%f ",a[i][j]);
printf("\n"); }
u= (x - a[n-1][0])/(a[1][0]-a[0][0]);
y=a[n-1][1];
u1=u;
fact=1;
j=2;
for(i=n-2;i>=0;i--)
{ y=y+(u1*a[i][j])/fact;
fact=fact*j;
u1=u1*(u+(j-1));
j++;
}
printf("\n\nValue at X=%g is = %f", x,y); }
```

OUTPUT:-

```
Name-Arpit Gupta
Section:- E
Roll.no-09
Enter no. of terms
5
Enter Values of X
1891
1901
1911
1921
1931
Enter Values of Y
46
66
81
93
101
Enter value of x for which you want y
1925
The Difference Table is as follows:
1891.000000 46.000000 20.000000 -5.000000 2.000000 -3.000000
1901.000000 66.000000 15.000000 -3.000000 -1.000000
1911.000000 81.000000 12.000000 -4.000000
1921.000000 93.000000 8.000000
1931.000000 101.000000


Value at X=1925 is = 96.836800

...Program finished with exit code 0
Press ENTER to exit console.
```

LAGRANGE METHOD

```
#include<stdio.h>
int main(){
printf(“Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n”);
float x[30],y[30],c,a,b,k=0;
int i,j,n;
printf("enter the number of terms of table:");
scanf("%d",&n);
printf("enter the respective values of x and y:");
for(i=0;i<n;i++){
scanf("%f",&x[i]);
scanf("%f",&y[i]);}
printf("\n\n The table you entered is as follows :\n\n");
 for(i=0; i<n; i++) {
 printf("%0.3f\t%0.3f",x[i],y[i]);
 printf("\n");}
 printf(" \n\n\n Enter the value of the x to find the respective value of y\n\n\n\n");
 scanf("%f",&c);
 for(i=0; i<n; i++){
 a=1;b=1;
 for(j=0; j<n; j++){
 if(j!=i){
 a=a*(c-x[j]);
 b=b*(x[i]-x[j]);}}
 k=k+((a/b)*y[i]);
}
 printf("\n\n The respective value of the variable y is: %f",k);
 }
```

OUTPUT:-

```
Name-Arpit Gupta
Section:- E
Roll.no-09
enter the number of terms of table:4
enter the respective values of x and y:5 12
6 13
9 14
11 16


 The table you entered is as follows :

5.000   12.000
6.000   13.000
9.000   14.000
11.000  16.000



 Enter the value of the x to find the respective value of y


10


 The respective value of the variable y is: 14.666666

...Program finished with exit code 0
Press ENTER to exit console.
```

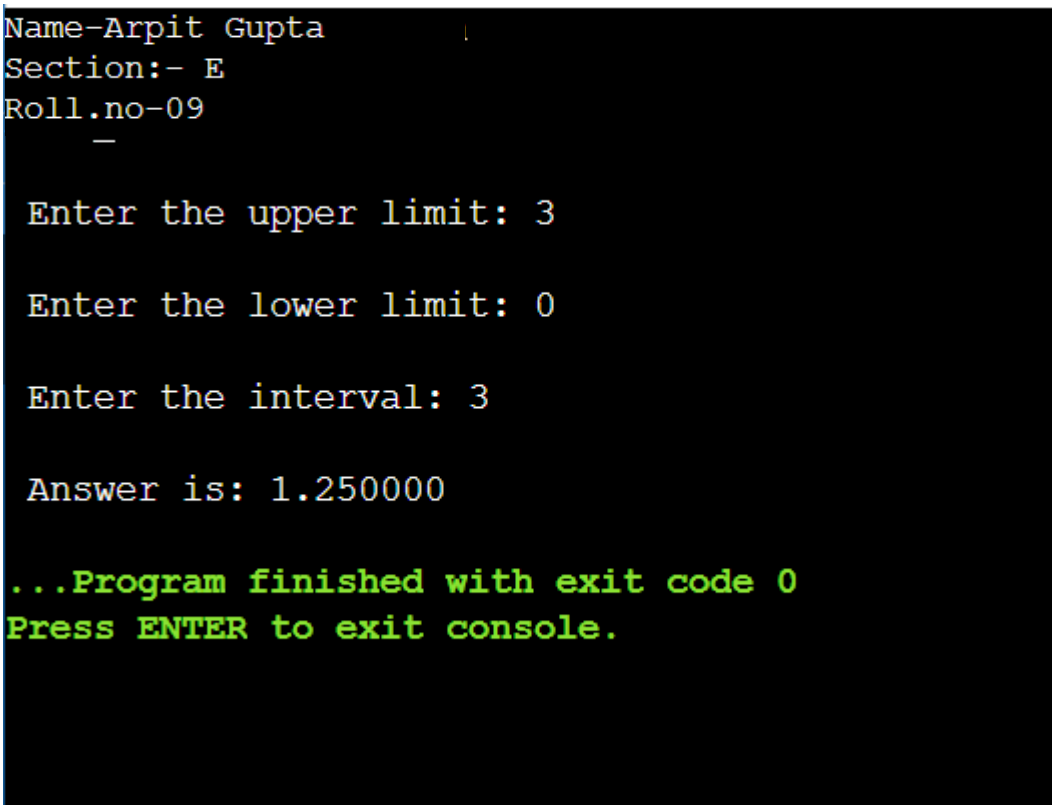
TRAPEZOIDAL METHOD

```
#include<stdio.h>
#include<math.h>
float func(float x)
{
 return(1/(1+pow(x,2)));
}
 int main()
 {
printf(“Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n”);

 int n,i;
 float a,b,h,sum,integral;
 printf("\n Enter the upper limit: ");
 scanf ("%f",&a);
 printf("\n Enter the lower limit: ");
 scanf ("%f",&b);
 printf("\n Enter the interval: ");
 scanf ("%d",&n);
 h=(a-b)/n;
 sum=func(a)+func(b);
 i=2;
 while(i<=n)
 {
 sum=sum+2*func(b+(i-1)*h);
 i++;
 }
 integral=h*sum/2;
 printf("\n Answer is: %f",integral);

}
```

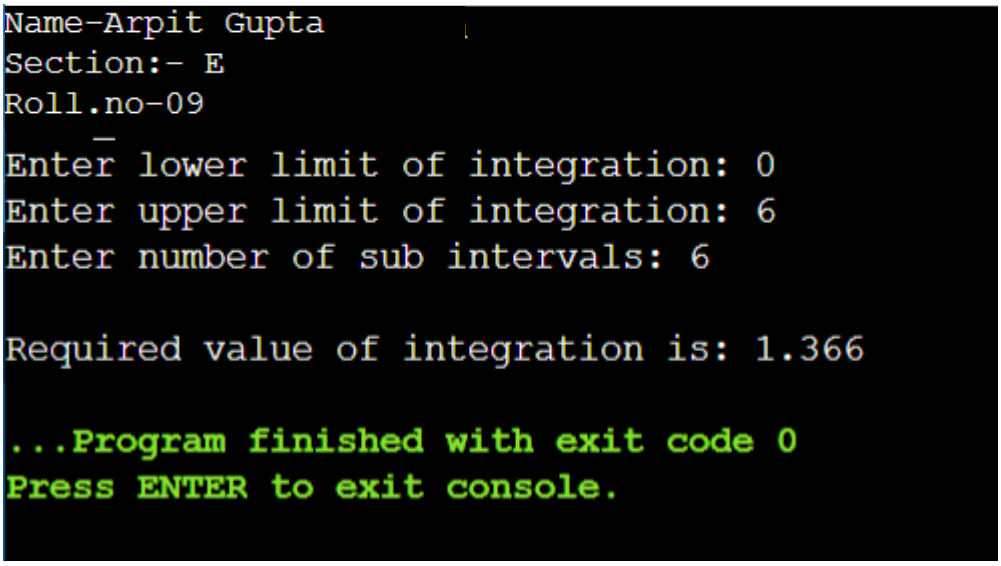
OUTPUT:-


Name-Arpit Gupta
Section:- E
Roll.no-09
Enter the upper limit: 3
Enter the lower limit: 0
Enter the interval: 3
Answer is: 1.250000
...Program finished with exit code 0
Press ENTER to exit console.

SIMPSON'S 1/3 METHOD

```
#include<stdio.h>
#include<math.h>
#define f(x) 1/(1+x*x)
int main(){
printf("Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n");
 float lower, upper, integration=0.0, h, k;
int i, subInterval;
printf("Enter lower limit of integration: ");
scanf("%f", &lower);
printf("Enter upper limit of integration: ");
scanf("%f", &upper);
printf("Enter number of sub intervals: ");
scanf("%d", &subInterval);
h= (upper - lower)/subInterval;
integration = f(lower) + f(upper);
for(i=1; i<= subInterval-1; i++){
 k = lower + i*h;
 if(i%2==0)
 {
 integration = integration + 2 * f(k);
 }
 else
 {
 integration = integration + 4 * f(k);
 }
}
integration = integration * h/3;
printf("\nRequired value of integration is: %.3f", integration);
return 0;
}
```

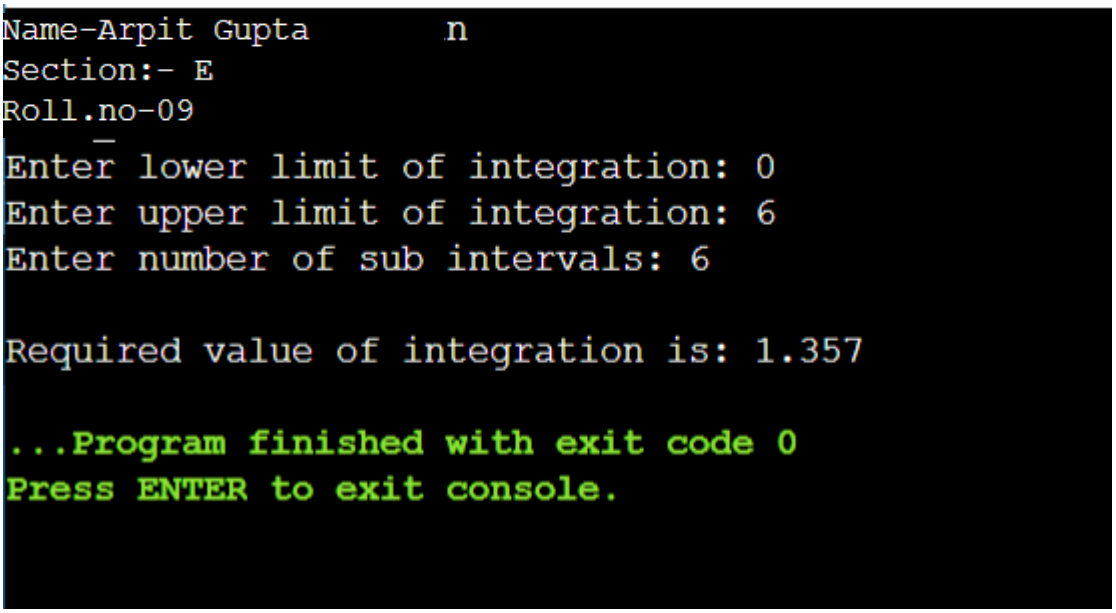
OUTPUT:-


Name-Arpit Gupta
Section:- E
Roll.no-09
Enter lower limit of integration: 0
Enter upper limit of integration: 6
Enter number of sub intervals: 6
Required value of integration is: 1.366
...Program finished with exit code 0
Press ENTER to exit console.

SIMPSON'S 3/8 METHOD

```
#include<stdio.h>
#include<math.h>
#define f(x) 1/(1+x*x)
int main(){
printf("Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n");
 float lower, upper, integration=0.0, h, k;
int i, subInterval;
printf("Enter lower limit of integration: ");
scanf("%f", &lower);
printf("Enter upper limit of integration: ");
scanf("%f", &upper);
printf("Enter number of sub intervals: ");
scanf("%d", &subInterval);
h = (upper - lower)/subInterval;
integration = f(lower) + f(upper);
for(i=1; i<= subInterval-1; i++){
 k = lower + i*h;
 if(i%3 == 0) {
 integration = integration + 2 * f(k);
 }
 else {
 integration = integration + 3 * f(k);
 }
}
integration = integration * h*3/8;
printf("\nRequired value of integration is: %.3f", integration);
return 0;
}
```

OUTPUT:-


Name-Arpit Gupta n
Section:- E
Roll.no-09
Enter lower limit of integration: 0
Enter upper limit of integration: 6
Enter number of sub intervals: 6
Required value of integration is: 1.357
...Program finished with exit code 0
Press ENTER to exit console.

EULER'S METHOD

```
#include<stdio.h>
double f(double x,double y,double h)
{
 double v=(y-x)/(double)(y+x);
 return v;
}
int main()
{
printf("Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n");
double h,x0,xn,y0;
 printf("enter the value of h,x0,y0: ");
 scanf("%lf %lf %lf",&h,&x0,&y0);
 printf("enter the value of xn:");
 scanf("%lf",&xn);
 while(x0 < xn)
 {
 y0=y0+h*(f(x0,y0,h));
 x0=x0+h; }
 printf("xn:%lf,y0:%lf",xn,y0);
}
```

OUTPUT:-

```
Name-Arpit Gupta        n
Section:- E
Roll.no-09
    _
enter the value of h,x0,y0: 0.025 0 1
enter the value of xn:0.1
xn:0.100000,y0:1.093264

...Program finished with exit code 0
Press ENTER to exit console.
```

RK'S METHOD

```
#include<stdio.h>
#include<math.h>
#define f(x,y) x+y
 void main(){
printf("Name-Arpit Gupta \n  Section:- E \n Roll.no-09\n");
  int n,i;
 float x,xf,y,h,s,s1,s2,s3,s4;
 printf("Enter x0 =");
 scanf("%f",&x);
 printf("Enter y0 =");
 scanf("%f",&y);
 printf("Enter the xn =");
 scanf("%f",&xf);
 printf("Enter h =");
 scanf("%f",&h);
 while(x<xf)
 {
 s1=f(x,y);
 s2=f((x+(h/2)),(y+(h/2)*s1));
 s3=f((x+(h/2)),(y+(h/2)*s2));
 s4=f((x+h),(y+h*s3));
 s=(s1+(2*s2)+(2*s3)+s4)/6;
 y=y+(h*s);
 x=x+h;
 }
 printf("Output = %f",y);
 }
```

OUTPUT:-

```
Name-Arpit Gupta        n
Section:- E
Roll.no-09
    _
Enter x0 =0
Enter y0 =1
Enter the xn =0.2
Enter h =0.1
Output = 1.242805

...Program finished with exit code 0
Press ENTER to exit console.
```