```
//BISECTION METHOD
#include<stdio.h>
#include<math.h>
float f(float x)
{
	return (x*x*x-exp(x)+1.7);
}
main()
{
	float x0,x1,x,e;
	printf("\n Enter the given values, x0 and x1:");
	scanf("%f %f",&x0,&x1);
	if(f(x0)*f(x1)<0)
	{
		x=(x0+x1)/2;
		printf("\n Enter the error value:e");
		scanf("%f", &e);
		while(fabs(x0-x1)>e)
		{
			if(f(x0)*f(x)<0)
			{
				x1=x;
			}
			else{
				x0=x;
			}
			x=(x0+x1)/2;
			printf("\n Approx roof is= %f",x);
		}
	} else
	{
		printf("\n Re-enter gass value");
	}
}
```

```
//REGULAR FALSE
#include<stdio.h>
#include<math.h>
float f(float x)
{
	return (x*log10(x)-1.2);
}
main()
{
	float x0,x1,x,e;
	printf("\n Enter the given values, x0 and x1:");
	scanf("%f %f",&x0,&x1);
	if(f(x0)*f(x1)<0)
	{
		x=(x0*f(x1)-x1*f(x0)) / (f(x1)-f(x0));
		printf("\n Enter the error value:e");
		scanf("%f", &e);
		while(fabs(x0-x1)>e)
		{
			if(f(x0)*f(x)<0)
			{
				x1=x;
			}
			else{
				x0=x;
			}
			x=(x0*f(x1)-x1*f(x0)) / (f(x1)-f(x0));
			printf("\n Approximate root is= %f",x);
		}
	} else
	{
		printf("\n Re-enter gass value");
	}
}
```

```
/* programming to implement larangers method */
#include<stdio.h>
#include<math.h>

main(){
	float x[10],y[10],product,sum=0,z;
	int i,j,n;

	printf("\n Enter the number of point,n: ");
	scanf("%d", &n);
	printf("\n Enter the  value of x and y from the data:\n ");
	for(i=1;i<=n;i++)
	{
	scanf("%f%f", &x[i],&y[i]);
	}

printf("\n Enter the value of x for which y=f(x) value required: ");
	scanf("%f",&z);

	for(i=1;i<=n;i++)
	{
		product=1;
		for(j=1;j<=n;j++)
		{
			if(i!=j)
			{
			product=product*(z-x[j])/(x[i]-x[j]);
			}
		}
		sum=sum+product*y[i];
	}
	printf("\nRequired inter polation value=%f:", sum);
}
```

```
//Newton Rapson Method
#include<stdio.h>
#include<math.h>
float f(float x)
{
	return ( pow(x,4)-x-10);
}
float fd(float x)
{
	return ( 4*pow(x,3)-1);
}
float fdd(float x)
{
	return ( 12*pow(x,2));
}

main()
{
	float x0,x1,e;
	printf("\n Enter the initial value of x0:");
	scanf("%f",&x0);
	x1=x0-f(x0)/fd(x0);
	printf("\n Enter the error value e:");
	scanf("%f",&e);


	if(pow(x0*fdd(x0),2)>0)
	{
		while(fabs(x0-x1)>e)
		{
			printf("\n Reqd  approx root = %f ",x1);
			x0=x1;
			x1=x0-f(x0)/fd(x0);
		}

	} else
	{
		printf("\n Re-enter initial value");
	}
}
```

```
//SECANT METHOD
#include<stdio.h>
#include<math.h>
float f(float x)
{
        return (x*x*x-4*x-9);
}
main()
{
        float x0,x1,x2,e;
        printf("\n Enter the given values of x0 and x1:");
        scanf("%f %f",&x0,&x1);
        printf("\n Enter the error values:");
        scanf("%f",&e);
        x2=(x0*f(x1)-x1*f(x0)) / (f(x1)-f(x0));
        while(fabs(x0-x2)>e)
        {
                x0=x1;
                x1=x2;
                x2=(x0*f(x1)-x1*f(x0)) / (f(x1)-f(x0));
                printf("\n Approximate root is= %f",x2);
        }
}
// Eular method
#include<stdio.h>
#include<math.h>
float f(float x, float y)
{
        return ((y-x)/(y+x));
}
main()
{
        float x0,y0,x1,y1,h,xn;
        printf("\n Enter the initial value of x0 and y0:");
        scanf("%f %f",&x0,&y0);
        printf("\n Enter the step length:");
        scanf("%f",&h);
        printf("\n Enter the value of x for which y is required xn:");
        scanf("%f",&xn);

        while(x0<=xn)
        {
                printf("\n when x = %f value of y=%f", x0,y0);
                y1=y0+h*f(x0,y0);
                x1=x0+h;
                y0=y1;
                x0=x1;
   }

}

// Improvement Eular method
#include<stdio.h>
#include<math.h>
float f(float x, float y)
{
        return (x+y);
}
main()
{
        float x0,y0,h,xn,x1,y1,y11;
        printf("\n Enter the initial value of x and y:");
        scanf("%f %f",&x0,&y0);
        printf("\n Enter the step length:");
        scanf("%f",&h);
        printf("\n Enter the last value of x at which y is required xn:");
        scanf("%f",&xn);
        printf("\n The values of y calculated by improved Euler Method
are:");

        while(x0<=xn)
        {
                printf("\n when x = %f value of y=%f", x0,y0);
                y11=y0+h*f(x0,y0);
                x1=x0+h;
                y1=y0+h*(f(x0,y0)+f(x1,y11))/2;
                x0=x1;
                y0=y1;
   }

}

/* Modified method */
#include<stdio.h>
#include<math.h>
float f(float x, float y){
        return (x+y);
}
main(){
        float x0,y0,h,xn,x1,y1,y11,y12,y13,y14,y15,e;
        printf("\n Enter the value of x and y: ");
        scanf("%f%f", &x0,&y0);
        printf("\n Enter the step length: ");
        scanf("%f",&h);
        printf("\n Enter the val of x at which y is required: ");
        scanf("%f",&xn);
        printf("\n Enter the error value: ");
        scanf("%f",&e);
        printf("\n The value of y cal by modified eucler method are:");
        printf("\n when x=%f the val of y =%f", x0,y0);

        while(x0<xn)
        {
                y11=y0+h*f(x0,y0);
                x1=x0+h;
                y12=y0+h*(f(x0,y0)+f(x1,y11))/2;

                while(fabs(y11-y12)>e)
                {
                        y13=y0+h*(f(x0,y0)+f(x1,y12))/2;
                        y11=y12;
                        y12=y13;
                        printf("\n when x=%f the val of y =%f",
x1,y13);
                }
                x0=x1;
                y0=y12;
        }


}

//RK3
#include<stdio.h>
#include<math.h>
float f(float x, float y)
{
        return (x+y);
}
main()
{
        float x0,y0,x1,y1,h,xn,k1,k2,k3,t;
        printf("\n Enter the initial approximations x0,y0: ");
        scanf("%f %f",&x0,&y0);
        printf("\n Enter the step length:");
        scanf("%f",&h);
        printf("\n Enter the value of x at which which y is required xn:");
        scanf("%f",&xn);
        printf("\n The values of y calculated by RK3 for different values of x
are:");

        while(x0<=xn)
        {
                k1=h*f(x0,y0);
                k2=h*f(x0+h/2,y0+k1/2);
                t=y0+h*f(x0+h,y0+k1);
                k3=h*f(x0+h,t);
                y1=y0+(k1+4*k2+k3)/6;
                x1=x0+h;
                y0=y1;
                x0=x1;
                printf("\n when x = %f value of y=%f", x1,y1);
   }

}
```

```
// implement of RK4
#include<stdio.h>
#include<math.h>
float f(float x, float y)
{
	return (3*exp(x)+2*y);
}
main()
{
	float x0,y0,x1,y1,h,xn,k1,k2,k3,k4;
	printf("\n Enter the initial approximations x0,y0: ");
	scanf("%f %f",&x0,&y0);
	printf("\n Enter the step length:");
	scanf("%f",&h);
	printf("\n Enter the value of x at which which y is required
xn:");
	scanf("%f",&xn);
	printf("\n The values of y calculated by RK4 for different
values of x are:");

	while(x0<xn)
	{
		k1=h*f(x0,y0);
		k2=h*f(x0+h/2,y0+k1/2);
		k3=h*f(x0+h/2,y0+k2/2);
		k4=h*f(x0+h,y0+k3);
		y1=y0+(k1+2*k2+2*k3+k4)/6;
		x0=x0+h;
		y0=y1;
		printf("\n when x = %f value of y=%f", x0,y0);
   }

}
```

```
//* A C Programme to Simpson 1/3 Rule */
# include <stdio.h>
# include <math.h>


float f(float x)
{
	return (1/(1+pow(x,2)));

}

main()
{
	float x0,xn,h,sum;
	int i,n;
	printf("\nEnter values of x0 :");
	scanf("%f",&x0);
	printf("\nEnter values of xn: ");
	scanf("%f",&xn);
	printf("\nEnter the number of subintervals:");
	scanf("%d",&n);
	h=(xn-x0)/n;
	sum=f(x0)+f(xn)+4*f(x0+h);
	for(i=3;i<=n-1;i=i+2)
	{
		sum=sum+4*f(x0+i*h)+2*f(x0+(i-1)*h);
	}
	sum=sum*h/3;
	printf("\n Value of integral = %f",sum);
}
```

```
//* A C Programme to Simpson 3/8 Rule */
# include <stdio.h>
# include <math.h>


float f(float x)
{
	return (1/(1+pow(x,2)));
}

main()
{
	float x0,xn,h,sum;
	int i,n;
	printf("\nEnter values of x0 :");
	scanf("%f",&x0);
	printf("\nEnter values of xn: ");
	scanf("%f",&xn);
	printf("\nEnter the number of subintervals:");
	scanf("%d",&n);
	printf("\n x0=%f",x0);
	h=(xn-x0)/n;
	sum=f(x0)+f(xn);
	for(i=1;i<=n-1;i=i+1)
	{
		if(i%3==0){
			sum=sum+2*f(x0+i*h);
		}else{
		sum=sum+3*f(x0+i*h);
		}
	}
	sum=sum*((3*h)/8);
	printf("\n Value of integral = %f",sum);
}
```

```
//* A C Programme to Implement Trapezodial Method */
# include <stdio.h>
# include <math.h>


float f(float x)
{
	return (1/(1+pow(x,2)));

}

main()
{
	float x0, xn, h, sum;
	int i, n;

	printf("\nEnter values of x0 :");
	scanf("%f", &x0);
	printf("\nEnter values of xn: ");
	scanf("%f", &xn);
	printf("\nEnter the number of subintervals:");
	scanf("%d", &n);
	h=(xn-x0)/n;
	sum=f(x0)+f(xn);
	for (i = 1; i <= n-1; i++)
	{
		sum=sum + 2* f(x0+i*h);
	}
	sum = sum *h/2;
	printf("\n Value of integral = %f", sum);
}
```

```
/* A C Programme to Implement Milne Predictor Corrector
Method */
# include <stdio.h>
# include <math.h>

float f(float x, float y)
{
return ((2*x*y+exp(x))/(x*x+x*exp(x)));
}

main() {
        float x0, y0, x1, y1, x2, y2, x3, y3, x4, y4, h, xn, e,
y41, y42;
        printf("\nEnter values of  x0 and y0:");
        scanf("%f%f", &x0, &y0);
        printf("\nEnter values of x1 and y1:");
        scanf("%f%f", &x1, &y1);
        printf("\nEnter values of x2 and y2:");
        scanf("%f%f", &x2, &y2);
        printf("\nEnter values of x3 and y3:");
        scanf("%f%f", &x3, &y3);
        printf("\nEnter the error value : ");
        scanf("%f", &e);
        printf("\nEnter the value of x at which y is
required");
        scanf("%f", &xn);
        h=x1-x0;
        while (x3<xn) {
                x4=x3+h;

y4=y0+4*h*(2*f(x1,y1)-f(x2,y2)+2*f(x3,y3))/3;

y41=y2+h*(f(x2,y2)+4*f(x3,y3)+f(x4,y4))/3;
                printf("\n Predicted value of y at x=%f is
y=%f", x4, y4);
                printf("\n Corrected value of y at x=%f is
y=%f", x4, y41);
                while (fabs(y41-y4)>e) {

y42=y2+h*(f(x2,y2)+4*f(x3,y3)+f(x4,y41))/3;

y42=y2+h*(f(x2,y2)+4*f(x3,y3)+f(x4,y41))/3;
                                y4=y41;
                                y41=y42;
                                printf("\n Corrected value of y
at x=%f is y=%f", x4, y42);
                }
                printf("\nThe value of y at x=%f is y=%f",
x4, y41);
                y0=y1;
                y1=y2;
                y2=y3;
                y3=y4;
                x0=x1;
                x1=x2;
                x2=x3;
                x3=x4;
        }
}
```