```c
// 01 01 Bisection Method
#include <stdio.h>  //input 1, 3
#include <stdlib.h>
#include <math.h>
#define f(x) ((x*x*x)-18)
int main()
{
    float a=0,b=0,error=0,m,mold;
    int i=0;
    printf("Input Interval: ");
    scanf("%f %f",&a,&b);
    if((f(a)*f(b))>0)
    {
        printf("Invalid Interval Exit!");       //to test whether search interval is okay or not
        exit(1);
    }
    else if(f(a)==0 || f(b)==0)
    {
        printf("Root is one of interval bounds. Root is %f\n",f(a)==0?a:b);
        exit(0);
    }

    printf("Ite\ta\t\tb\t\tm\t\tf(m)\t\terror\n");
    do{
        mold=m;
        m=(a+b)/2;

        printf("%2d\t%4.6f\t%4.6f\t%4.6f\t%4.6f\t",i++,a,b,m,f(m));
        if(f(m)==0)
        {
            printf("Root is %4.6f\n",m);
        }
        else if((f(a)*f(m))<0)
        {
            b=m;
        }
        else a=m;
```

```
36                  error=fabs(m-mold);
37              if(i==1)
38              {
39                  printf("----\n");
40              }
41              else
42                  printf("%4.6f\n",error);
43          }while(error>0.00005);
44          printf("Approximate Root is %4.6f",m);
45          return 0;
46      }
47
```

```c
// 01 02 Regula Falsi
#include <stdio.h>  //input 1, 3
#include <stdlib.h>
#include <math.h>
#define f(x) ((x*x*x)-18)
int main()
{
    float a=0,b=0,error=0,c,cold;
    int i=0;
    printf("Input Interval: ");
    scanf("%f %f",&a,&b);
    if((f(a)*f(b))>0)
    {
        printf("Invalid Interval Exit!");
        exit(1);
    }
    else if(f(a)==0 || f(b)==0)
    {
        printf("Root is one of interval bounds. Root is %f\n",f(a)==0?a:b);
        exit(0);
    }

    printf("Ite\ta\t\tb\t\tc\t\tf(c)\t\terror\n");
    do{
        cold=c;

    c=(((a*f(b))-(b*f(a)))/(f(b)-f(a)));

    printf("%2d\t%4.6f\t%4.6f\t%4.6f\t%4.6f\t",i++,a,b,c,f(c));
        if(f(c)==0)
        {
            break;
        }
        else if(f(a)*f(c)<0)
        {
            b=c;
        }
        else a=c;
            error=fabs(c-cold);
```

```
37              if(i==1)
38              {
39                  printf("----\n");
40              }
41              else
42                  printf("%4.6f\n",error);
43
44          }while(error>0.00005);
45          printf(" Root is %4.6f \n",c);
46          return 0;
47      }
48
```

```c
// 01 03 Newton Raphson
#include <stdio.h>  //input 2
#include <stdlib.h>
#include <math.h>
#define f(x) ((x*x*x)-18)
#define fd(x) (3*x*x)
#define fdd(x) 6*x
int main()
{
    float x0,x1,error,errorold,converge,order;
    int i=0;
    printf("Input the approximation : ");
    scanf("%f",&x0);

    converge=(f(x0)*fdd(x0))/(fd(x0)*fd(x0));
    if(converge >1)
        exit(1);

    printf("Ite\tX0\t\tX1\t\tError\t\tOrder\n");
    do{
        errorold=error;
        x1=x0-(f(x0)/fd(x0));
        if(f(x1)==0)
        {
            break;
        }
        error=fabs(x1-x0);

        printf("%2d\t%4.6f\t%4.6f\t%4.6f\t",++i,x0,x1,error);
        if(i==1||error==0||errorold==1)
        {
            printf("-----\n");
        }
        else
        {
            order=log(error)/log(errorold);
            printf("%4.6f\n",order);
        }
        x0=x1;
    }while(error>0.00005);
```

```
38          printf("Root is %4.6f",x0);
39          return 0;
40      }
41
```

```c
// 01 04 Secant Method
#include<stdio.h>
#include<math.h>
/*Function whose root is to be determined*/
double f(double x)
{
    return x*x-4;
}
main()
{
    int iter=1, maxSteps;
    double x1,x2,x3,eps;
    printf("Enter the accuracy desired: \n");
    scanf("%lf",&eps);
    printf("Enter the intial guesses: \nx1 = ");
    scanf("%lf",&x1);
    printf("x2 = ");
    scanf("%lf",&x2);
    printf("Enter the max number of iterations to be performed: ");
    scanf("%d",&maxSteps);

    printf("iter\tx1\t\tx2\t\tx3\t\tf(x3)\n");

    printf("_____\n");
    do
    {
        x3=( x1*f(x2)-x2*f(x1))/(f(x2)-f(x1) );

    printf("%d\t%lf\t%lf\t%lf\t%lf\n",iter,x1,x2,x3,f(x3));
        x1=x2;
        x2=x3;
        iter++;
    }
    while( fabs(f(x3))>eps&&iter<=maxSteps);
    printf("\nOne of the roots is: %lf",x3);
}
```

```c
// 01 05 Gauss Elimination
#include<stdio.h>
#include<conio.h>
#include<math.h>
#define SIZE 10
int main()
{
    float a[SIZE][SIZE], x[SIZE], ratio;
    int i,j,k,n;
    /* Inputs */
    /* 1. Reading number of unknowns */
    printf("Enter number of unknowns: ");
    scanf("%d", &n);
    /* 2. Reading Augmented Matrix */
    for(i=1;i<=n;i++)
    {
        for(j=1;j<=n+1;j++)
        {
        printf("a[%d][%d] = ",i,j);
        scanf("%f", &a[i][j]);
        }
    }
    /* Applying Gauss Elimination */
    for(i=1;i<=n-1;i++)
    {
        if(a[i][i] == 0.0)
        {
        printf("Mathematical Error!");
        exit(0);
        }
        for(j=i+1;j<=n;j++)
        {
            ratio = a[j][i]/a[i][i];
            for(k=1;k<=n+1;k++)
            {
            a[j][k] = a[j][k] -
ratio*a[i][k];
            }
        }
    }
    /* Obtaining Solution by Back
Subsitution */
```

```
41          x[n] = a[n][n+1]/a[n][n];
42          for(i=n-1;i>=1;i--)
43          {
44              x[i] = a[i][n+1];
45              for(j=i+1;j<=n;j++)
46              {
47              x[i] = x[i] - a[i][j]*x[j];
48              }
49              x[i] = x[i]/a[i][i];
50          }
51          /* Displaying Solution */
52          printf("\nSolution:\n");
53          for(i=1;i<=n;i++)
54          {
55              printf("x[%d] = %0.3f\n",i, x[i]);
56          }
57          getch();
58      }
59
```

```c
// 01 06 Gauss Seidal
#include<stdio.h>
#include<conio.h>
#include<math.h>
#define f1(x,y,z)  (17-y+2*z)/20
#define f2(x,y,z)  (-18-3*x+z)/20
#define f3(x,y,z)  (25-2*x+3*y)/20

    // Arrange systems of linear
    // equations to be solved in
    // diagonally dominant form
    // and form equation for each
    // unknown and define here

    // In this example we are solving
    // 3x + 20y - z = -18
    // 2x - 3y + 20z = 25
    // 20x + y - 2z = 17

    // Arranging given system of linear
    // equations in diagonally dominant
    // form:
    // 20x + y - 2z = 17
    // 3x + 20y -z = -18
    // 2x - 3y + 20z = 25

    // Equations:
    // x = (17-y+2z)/20
    // y = (-18-3x+z)/20
    // z = (25-2x+3y)/20

    // Defining function

    // Main function
int main()
{
    float x0=0, y0=0, z0=0, x1, y1, z1,
e1, e2, e3, e;
    int count=1;
    printf("Enter tolerable error:\n");
    scanf("%f", &e);
    printf("\nCount\tx\ty\tz\n");
```

```
42        do
43        {
44            // Calculation
45            x1 = f1(x0,y0,z0);
46            y1 = f2(x1,y0,z0);
47            z1 = f3(x1,y1,z0);
48
      printf("%d\t%0.4f\t%0.4f\t%0.4f\n",count,
      x1,y1,z1);
49
50            // Error
51            e1 = fabs(x0-x1);
52            e2 = fabs(y0-y1);
53            e3 = fabs(z0-z1);
54
55            count++;
56
57            // Set value for next iteration
58            x0 = x1;
59            y0 = y1;
60            z0 = z1;
61        }while(e1>e && e2>e && e3>e);
62        printf("\nSolution: x=%0.3f, y=%0.3f
      and z = %0.3f\n",x1,y1,z1);
63        getch();
64        return 0;
65    }
```

```c
// 01 07 Newton Backward
#include<stdio.h>
#include<conio.h>

int main()
{
    float x[20], y[20][20];
    int i,j, n;
    /* Input Section */
    printf("Enter number of data?\n");
    scanf("%d", &n);
    printf("Enter data:\n");
    for(i = 0; i < n ; i++)
    {
        printf("x[%d]=", i);
        scanf("%f", &x[i]);
        printf("y[%d]=", i);
        scanf("%f", &y[i][0]);
    }
    /* Generating Backward Difference Table */
    for(i = 1; i < n; i++)
    {
        for(j = n-1; j > i-1; j--)
        {
            y[j][i] = y[j][i-1] - y[j-1][i-1];
        }
    }
    /* Displaying Backward Difference Table */
    printf("\nBACKWARD DIFFERENCE TABLE\n\n");
    for(i = 0; i < n; i++)
    {
        printf("%0.2f", x[i]);
        for(j = 0; j <= i ; j++)
        {
            printf("\t%0.2f", y[i][j]);
        }
        printf("\n");
    }
```

```
39          getch(); /* Holding Screen */
40          return 0;
41      }
42
```

```c
// 01 08 Newton Forward
#include<stdio.h>
#include<conio.h>
int main()
{
    float x[20], y[20][20];
    int i,j, n;
    // Input Section
    printf("Enter number of data: ");
    scanf("%d", &n);
    printf("Enter data:\n");
    for(i = 0; i < n ; i++)
    {
        printf("x[%d]=", i);
        scanf("%f", &x[i]);
        printf("y[%d]=", i);
        scanf("%f", &y[i][0]);
    }
    // Generating Forward Difference Table
    for(i = 1; i < n; i++)
    {
        for(j = 0; j < n-i; j++)
        {
            y[j][i] = y[j+1][i-1] -
y[j][i-1];
        }
    }
    // Displaying Forward Difference Table
    printf("\nFORWARD DIFFERENCE
TABLE\n\n");
    for(i = 0; i < n; i++)
    {
        printf("%0.2f", x[i]);
        for(j = 0; j < n-i ; j++)
        {
            printf("\t%0.2f", y[i][j]);
        }
        printf("\n");
    }
    getch();
    return 0;
}
```

```c
// 01 09 Newton divide difference formula
#include<stdio.h>
#include<conio.h>

void main()
{
    int x[10], y[10], p[10];
    int k,f,n,i,j=1,f1=1,f2=0;
    printf("Enter the number of observations: ");
    scanf("%d", &n);

    printf("Enter the different values of x: \n");
    for (i=1;i<=n;i++)
        scanf("%d", &x[i]);

    printf("\nThe corresponding values of y are: \n");
    for (i=1;i<=n;i++)
        scanf("%d", &y[i]);

    f=y[1];
    printf("\nEnter the value of 'k' in f(k) you want to evaluate: ");
    scanf("%d", &k);

    do
    {
        for (i=1;i<=n-1;i++)
        {
            p[i] = ((y[i+1]-y[i])/(x[i+j]-x[i]));
            y[i]=p[i];
        }
        f1=1;
        for(i=1;i<=j;i++)
            {
                f1*=(k-x[i]);
            }
        f2+=(y[1]*f1);
        n--;
```

```
38              j++;
39          }
40
41          while(n!=1);
42          f+=f2;
43          printf("\nf(%d) = %d", k , f);
44          getch();
45      }
46
```

```c
// 01 10 Labrange's Interpolation
#include<stdio.h>
#include<conio.h>
void main()
{
    float x[100], y[100], xp, yp=0, p;
    int i,j,n;
    /* Input Section */
    printf("Enter number of data: ");
    scanf("%d", &n);
    printf("Enter data:\n");
    for(i=1;i<=n;i++)
    {
        printf("x[%d] = ", i);
        scanf("%f", &x[i]);
        printf("y[%d] = ", i);
        scanf("%f", &y[i]);
    }
    printf("Enter interpolation point: ");
    scanf("%f", &xp);
    /* Implementing Lagrange Interpolation
*/
    for(i=1;i<=n;i++)
    {
        p=1;
        for(j=1;j<=n;j++)
        {
            if(i!=j)
            {
            p = p* (xp - x[j])/(x[i] -
x[j]);
            }
        }
          yp = yp + p * y[i];
    }
    printf("Interpolated value at %.3f is
%.3f.", xp, yp);
    getch();
}

```