//Source code (GAUSS JACOBI METHOD)

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
float mat[3][4];
float fun1(float,float);
float fun2(float,float);
float fun3(float,float);
```

```
void main()
{
int i,j;
float x,y,z;
clrscr();
printf("\n\t.....Guass Jacobi Method....\n");
printf("\n\tEnter the elements of the matrix: ");
for(i=0;i<3;i++)
for(j=0;j<4;j++)
      scanf("%f",&mat[i][j]);
      printf("\n\tThe given matrix is:\n\n");
      for(i=0;i<3;i++)
            {
                  for(j=0;j<4;j++)
                  printf("%f\t",mat[i][j]);
                  printf("\n");
             }
             for(i=0;i<5;i++)
             {
             if(i==0)
              {
             x=fun1(0,0);
             y=fun2(0,0);
             z=fun3(0,0);
             }
             else
             {
             x=fun1(y,z);
             y=fun2(x,z);
             z=fun3(x,y);
             }
    }
    printf("\n\t x=%f,y=%f,z=%f\n\n",x,y,z);
    getch();
}
float fun1(float y,float z)
```

```
{
float x1;
x1=1/(mat[0][0])*(mat[0][3]-mat[0][2]*z-mat[0][1]*y);
return(x1);


}
float fun2(float x, float z)

{
float y1;
y1=1/(mat[1][1])*(mat[1][3]-mat[1][2]*z-mat[1][0]*x);
return(y1);

}
float fun3(float x, float y)

{
float z1;
z1=1/(mat[2][2]*(mat[2][3]-mat[2][1]*y-mat[2][0]*x);
return(z1);

}

/**********************************OUTPUT**********************************/
    Enter the elements of the matrix: 8  -3 2 20
411 -1 33
11  4 9
    The given matrix is:
8.000000        -3.000000               2.000000        20.000000
4.000000        11.000000               -1.000000       33.000000
1.000000        1.000000                4.000000        9.000000


x=3.000313, y=1.999982, z=0.999926
```

/*Source Code (Gauss Jordan Method)*/

```
#include<stdio.h>
#include<conio.h>
#include<conio.h>
void main()
{
      float mat[3][4];
      float k,l,m,n,x,y,z;
      int i,j;
      clrscr();
      printf("enter the elements of the matrix:/n");
      for(i=0;i<3;i++)
        for(j=0;j<4;j++)
                scanf("%f",&mat[i][j]);
      printf("\n\nthe given matrix is:/n/n");
      for(i=0;i<3;i++)
      {
        for(j=0;j<4;j++)
        printf("%f\t",mat[i][j]);
      printf("\n");
}
k=mat[1][0]/mat[0][0];
l=mat[2][0]/mat[0][0];
for(i=0;i<4;i++)
{
      mat[1][i]=mat[1][i]-(mat[0][i]*k);
      mat[2][i]=mat[2][i]-(mat[0][i]*l);
}
m=mat[2][1]/mat[1][1];
n=mat[0][1]/mat[1][1];
for(i=0;i<4;i++)
{
      mat[2][i]=mat[2][i]-(mat[1][i]*m);
      mat[0][i]=mat[0][i]-(mat[1][i]*n);
}
z=(mat[2][3]/mat[2][2];
```

```
y=(mat[1][3]-mat[1][2]*z)/mat[1][1];
x=(mat[0][3]-mat[0][2]*z)/mat[0][0];
printf("\n\n x=%f, y=%f,z=%f",x,y,z);
getch();
}
/**********************OUTPUT*****************************
Enter the elements of the matrix:
1 3 2 17
1 2 3 16
2 -1 4 13
the given matrix is:

1.000000    3.000000    2.000000    17.000000
1.000000    2.000000    3.000000    16.000000
2.000000    -1.000000   4.000000    13.000000

x=4.000000, y=3.000000, z=2.000000
```

//Source code (GAUSS SEIDEL METHOD)

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
float mat[3][4];
float fun1(float,float);
float fun2(float,float);
float fun3(float,float);
void main()
```

```
{
int i,j;
float x,y,z;
clrscr();
printf("\n\t.....Guass Seidel Method....\n");
printf("\n\tEnter the elements of the matrix: ");
for(i=0;i<3;i++)
for(j=0;j<4;j++)
    scanf("%f",&mat[i][j]);
    printf("\n\the given matrix is:\n\n");
    for(i=0;i<3;i++)
            {
                    for(j=0;j<4;j++)
                    printf("%f\t",mat[i][j]);
                    printf("\n");

            }
            for(i=0;i<5;i++)
            {
            if(i==0)
              {
             x=fun1(0,0);
            y=fun2(x,0);
            z=fun3(x,y);
            }
            else
            {
            x=fun1(y,z);
            y=fun2(x,z);
            z=fun3(x,y);
            }
    printf("\n\n x=%f,y=%f,z=%f\n\n",x,y,z);
    getch();
}

float fun1(float y,float z)
{
float x1;
```

```
x1=1/(mat[0][0])*(mat[0][3]-mat[0][2]*z-mat[0][1]*y);
return(x1);

}
float fun2(float x, float z)
{
float y1;
y1=1/(mat[1][1])*(mat[1][3]-mat[1][2]*z-mat[1][0]*x);
return(y1);
}
float fun3(float x, float y)
{
float z1;
z1=1/(mat[2][2]*(mat[2][3]-mat[2][1]*y-mat[2][0]*x);
return(z1);
}
```

/**********OUTPUT**********

Enter the elements of the matrix: 8 -3 2 20

4 11 -1 33

1 1 4 9

The given matrix is:

| | | | |
|---|---|---|---|
| 8.000000 | -3.000000 | 2.000000 | 20.000000 |
| 4.000000 | 11.000000 | -1.000000 | 33.000000 |
| 1.000000 | 1.000000 | 4.000000 | 9.000000 |

x=3.000027, y=1.999998, z=0.999994

```
float f(float x,float y)
{
    return(x*x+y); //here the function is f(x,y)=x^2+y
}
void main()
{
    float x0,y0,h,x,x1,y2,y1,y3;
    int count=0;
    clrscr();
    printf("\nEnter the initial value of x: ");
    scanf("%f",&x0);
    printf("\nEnter the initial value of y: ");
    scanf("%f",&y0);
    printf("\nEnter the step length: ");
    scanf("%f",&h);
    printf("\nEnter the value to be find: ");
    scanf("%f",&x);
    x1=x0;
    y1=y0;
    do
    {
    y2=y1+h*f(x1,y1);
    y3=y1+(h/2.0)*(f(x1,y1)+f(x1+h,y2));
    x1=x1+h;
    y1=y3;
    count++;
    }while(x1<x);
    printf("\nThe value of y(%8.6f)=%8.6f",x1,y3);
    getch();
    }
    /**********************OUTPUT*********************/
    Enter the initial value of x:0
    Enter the initial value of y:1
    Enter the step length: 0.01
    Enter the value to be find:0.02

    The value of y(0.020000)=1.020204
```

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#define e 0.00001
float f(float x)
{
    return(pow(x,3)+2*x-2); //only change the function
}
void main()
{
int k=0;
float x,a,b,h;
clrscr();
printf("\n Enter the lower limit of the root:");
scanf("%f",&a);
printf("\n Enter the upper limit of the root:");
scanf("%f",&b);
printf("\niteration\ta\t\tb\t\txn\t\tf(x)\n")'
do
   {
   h=(fabs(f(a))*(b-a)/(fabs(f(a))+fabs(f(b))));
   x=a+h;
   if(f(a)*f(x)<0)
            b=x;
   else
     a=x;
     k=k+1;
      printf("\n\t%d\t%f\t%f\t%f\t%f\n", k,a,b,x,f(x));
   }
   while(fabs(f(x))>=e);
   printf("\n The positive root is:=%f",x);
   getch();
/*******************OUTPUT**********************/
Enter the lower limit of the root:0
Enter the upper limit of the root:1
```