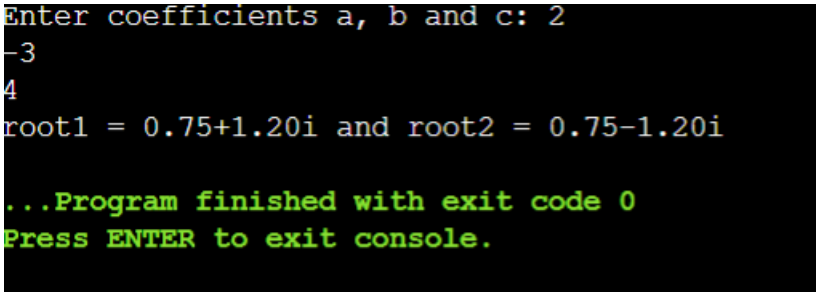
QUADRATIC

```
#include <math.h>
#include <stdio.h>
int main() {
   double a, b, c, discriminant, root1, root2, realPart, imagPart;
   printf("Enter coefficients a, b and c: ");
   scanf("%lf %lf %lf", &a, &b, &c);

   discriminant = b * b - 4 * a * c;

   // condition for real and different roots
   if (discriminant > 0) {
      root1 = (-b + sqrt(discriminant)) / (2 * a);
      root2 = (-b - sqrt(discriminant)) / (2 * a);
      printf("root1 = %.2lf and root2 = %.2lf", root1, root2);
   }

   // condition for real and equal roots
   else if (discriminant == 0) {
      root1 = root2 = -b / (2 * a);
      printf("root1 = root2 = %.2lf;", root1);
   }

   // if roots are not real
   else {
      realPart = -b / (2 * a);
```

```
        imagPart = sqrt(-discriminant) / (2 * a);
        printf("root1 = %.2lf+%.2lfi and root2 = %.2f-%.2fi", realPart, imagPart,
realPart, imagPart);
    }


    return 0;
}
```

OUTPUT


Enter coefficients a, b and c: 2
-3
4
root1 = 0.75+1.20i and root2 = 0.75-1.20i
...Program finished with exit code 0
Press ENTER to exit console.

BISECTION

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
/*
 Defining equation to be solved.
 Change this equation to solve another problem.
*/
#define f(x) cos(x) - x * exp(x)

void main()
{
        float x0, x1, x2, f0, f1, f2, e;
        int step = 1;
        clrscr();
        /* Inputs */
        up:
        printf("\nEnter two initial guesses:\n");
        scanf("%f%f", &x0, &x1);
        printf("Enter tolerable error:\n");
        scanf("%f", &e);
        /* Calculating Functional Value */
        f0 = f(x0);
        f1 = f(x1);
        /* Checking whether given guesses brackets the root or not. */
        if( f0 * f1 > 0.0)
```

```
	{
		printf("Incorrect Initial Guesses.\n");
		goto up;
	}
/* Implementing Bisection Method */
	printf("\nStep\t\tx0\t\tx1\t\tx2\t\tf(x2)\n");
	do
	{
		x2 = (x0 + x1)/2;
		f2 = f(x2);

		printf("%d\t\t%f\t%f\t%f\t%f\n",step, x0, x1, x2, f2);

		if( f0 * f2 < 0)
		{
			x1 = x2;
			f1 = f2;
		}
		else
		{
			x0 = x2;
			f0 = f2;
		}
		step = step + 1;
	}while(fabs(f2)>e);
	printf("\nRoot is: %f", x2);
```

```
        getch();
}
```

OUTPUT

```
Enter two initial guesses:
0
1
Enter tolerable error:
0.0001

Step       x0                  x1                x2                f(x2)
1          0.000000          1.000000          0.500000          0.053222
2          0.500000          1.000000          0.750000          -0.856061
3          0.500000          0.750000          0.625000          -0.356691
4          0.500000          0.625000          0.562500          -0.141294
5          0.500000          0.562500          0.531250          -0.041512
6          0.500000          0.531250          0.515625          0.006475
7          0.515625          0.531250          0.523438          -0.017362
8          0.515625          0.523438          0.519531          -0.005404
9          0.515625          0.519531          0.517578          0.000545
10         0.517578          0.519531          0.518555          -0.002427
11         0.517578          0.518555          0.518066          -0.000940
12         0.517578          0.518066          0.517822          -0.000197
13         0.517578          0.517822          0.517700          0.000174
14         0.517700          0.517822          0.517761          -0.000012

Root is: 0.517761
```

SECANT

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<stdlib.h>

/* Defining equation to be solved.
   Change this equation to solve another problem. */
#define    f(x)    x*x*x - 2*x - 5

void main()
{
        float x0, x1, x2, f0, f1, f2, e;
        int step = 1, N;
        clrscr();
        /* Inputs */
        printf("\nEnter initial guesses:\n");
        scanf("%f%f", &x0, &x1);
        printf("Enter tolerable error:\n");
        scanf("%f", &e);
        printf("Enter maximum iteration:\n");
        scanf("%d", &N);

        /* Implementing Secant Method */
        printf("\nStep\t\tx0\t\tx1\t\tx2\t\tf(x2)\n");
        do
```

```
{
        f0 = f(x0);
        f1 = f(x1);
        if(f0 == f1)
        {
                printf("Mathematical Error.");
                exit(0);
        }

        x2 = x1 - (x1 - x0) * f1/(f1-f0);
        f2 = f(x2);

        printf("%d\t\t%f\t%f\t%f\t%f\n",step,x0,x1,x2, f2);

        x0 = x1;
        f0 = f1;
        x1 = x2;
        f1 = f2;

        step = step + 1;

        if(step > N)
        {
                printf("Not Convergent.");
                exit(0);
        }
```

```
        }while(fabs(f2)>e);

        printf("\nRoot is: %f", x2);
        getch();
}
```

OUTPUT

```
Enter initial guesses:
1
2
Enter tolerable error:
0.00001
Enter maximum iteration:
10

Step       x0                 x1                 x2              f(x2)
1          1.000000           2.000000           2.200000        1.248001
2          2.000000           2.200000           2.088968        -0.062124
3          2.200000           2.088968           2.094233        -0.003554
4          2.088968           2.094233           2.094553        0.000012
5          2.094233           2.094553           2.094552        0.000001

Root is: 2.094552
```

REGULA FALSI

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
/* Defining equation to be solved.
   Change this equation to solve another problem. */
#define   f(x)   x*log10(x) - 1.2

int main()
{

        float x0, x1, x2, f0, f1, f2, e;
        int step = 1;
        clrscr();
        /* Inputs */
        up:
        printf("\nEnter two initial guesses:\n");
        scanf("%f%f", &x0, &x1);
        printf("Enter tolerable error:\n");
        scanf("%f", &e);
        /* Calculating Functional Values */
        f0 = f(x0);
        f1 = f(x1);
        /* Checking whether given guesses brackets the root or not. */
        if( f0*f1 > 0.0)
```

```
{
        printf("Incorrect Initial Guesses.\n");
        goto up;
}
/* Implementing Regula Falsi or False Position Method */
printf("\nStep\t\tx0\t\tx1\t\tx2\t\tf(x2)\n");
do
{
        x2 = x0 - (x0-x1) * f0/(f0-f1);
        f2 = f(x2);
        printf("%d\t\t%f\t%f\t%f\t%f\n",step, x0, x1, x2, f2);

        if(f0*f2 < 0)
        {
                x1 = x2;
                f1 = f2;
        }
        else
        {
                x0 = x2;
                f0 = f2;
        }
        step = step + 1;

}while(fabs(f2)>e);
```

```
        printf("\nRoot is: %f", x2);
        getch();
        return 0;
}
```

OUTPUT

```
Enter two initial guesses:
2
3
Enter tolerable error:
0.000001

Step            x0                 x1             x2                 f(x2)
1               2.000000           3.000000       2.721014           -0.017091
2               2.721014           3.000000       2.740206           -0.000384
3               2.740206           3.000000       2.740636           -0.000009
4               2.740636           3.000000       2.740646           -0.000000

Root is: 2.740646
```

NEWTON RAPHSON

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<stdlib.h>

/* Defining equation to be solved.
   Change this equation to solve another problem. */
#define    f(x)    3*x - cos(x) -1

/* Defining derivative of g(x).
   As you change f(x), change this function also. */
#define   g(x)   3 + sin(x)

void main()
{
        float x0, x1, f0, f1, g0, e;
        int step = 1, N;
        clrscr();
    /* Inputs */
        printf("\nEnter initial guess:\n");
        scanf("%f", &x0);
        printf("Enter tolerable error:\n");
        scanf("%f", &e);
```

```
printf("Enter maximum iteration:\n");
scanf("%d", &N);
/* Implementing Newton Raphson Method */
printf("\nStep\t\tx0\t\tf(x0)\t\tx1\t\tf(x1)\n");
do
{
	g0 = g(x0);
	f0 = f(x0);
	if(g0 == 0.0)
	{
		printf("Mathematical Error.");
		exit(0);
	}


	x1 = x0 - f0/g0;


	printf("%d\t\t%f\t%f\t%f\t%f\n",step,x0,f0,x1,f1);
	x0 = x1;

	step = step+1;

	if(step > N)
	{
		printf("Not Convergent.");
```

```
                exit(0);
            }

            f1 = f(x1);

        }while(fabs(f1)>e);

        printf("\nRoot is: %f", x1);
        getch();
}
```

OUTPUT

```
Enter initial guess:
1
Enter tolerable error:
0.00001
Enter maximum iteration:
10

Step                x0                  f(x0)               x1                  f(x1)
1                    1.000000           1.459698           0.620016           0.000000
2                    0.620016           0.046179           0.607121           0.046179
3                    0.607121           0.000068           0.607102           0.000068

Root is: 0.607102
```

ITERATIVE

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

/* Define function f(x) which
  is to be solved */
#define   f(x)   cos(x)-3*x+1
/* Write f(x) as x = g(x) and
  define g(x) here */
#define   g(x)   (1+cos(x))/3

int main()
{
        int step=1, N;
        float x0, x1, e;
        clrscr();
        /* Inputs */
        printf("Enter initial guess: ");
        scanf("%f", &x0);
        printf("Enter tolerable error: ");
        scanf("%f", &e);
        printf("Enter maximum iteration: ");
```

```
	scanf("%d", &N);
	/* Implementing Fixed Point Iteration */
	printf("\nStep\tx0\t\tf(x0)\t\tx1\t\tf(x1)\n");
	do
	{
		x1 = g(x0);
		printf("%d\t%f\t%f\t%f\t%f\n",step, x0, f(x0), x1, f(x1));

		step = step + 1;

		if(step>N)
		{
			printf("Not Convergent.");
			exit(0);
		}

		x0 = x1;

	}while( fabs(f(x1)) > e);

	printf("\nRoot is %f", x1);

	getch();
	return(0);
}
```

OUTPUT

```
Enter initial guess: 1
Enter tolerable error: 0.000001
Enter maximum iteration: 10

Step     x0                  f(x0)              x1                  f(x1)
1         1.000000          -1.459698         0.513434          0.330761
2         0.513434          0.330761          0.623688          -0.059333
3         0.623688          -0.059333         0.603910          0.011391
4         0.603910          0.011391          0.607707          -0.002162
5         0.607707          -0.002162         0.606986          0.000411
6         0.606986          0.000411          0.607124          -0.000078
7         0.607124          -0.000078         0.607098          0.000015
8         0.607098          0.000015          0.607102          -0.000003
9         0.607102          -0.000003         0.607102          0.000001

Root is 0.607102
```

GAUSS JORDAN

```
#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 */
        printf("Enter coefficients of Augmented 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]);
                }
        }
        /* Applying Gauss Jordan Elimination */
```

```
for(i=1;i<=n;i++)
{
        if(a[i][i] == 0.0)
        {
                printf("Mathematical Error!");
                exit(0);
        }
        for(j=1;j<=n;j++)
        {
                if(i!=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 */
for(i=1;i<=n;i++)
{
        x[i] = a[i][n+1]/a[i][i];
}
/* Displaying Solution */
printf("\nSolution:\n");
for(i=1;i<=n;i++)
{
        printf("x[%d] = %0.3f\n",i, x[i]);
}
```

```
                return(0);
}
```

OUTPUT

```
Enter number of unknowns: 4
Enter Coefficients of Augmented Matrix:
a[1]1]= 1
a[1]2]= 2
a[1]3]= 3
a[1]4]= -1
a[1]5]= 10
a[2]1]= 2
a[2]2]= 3
a[2]3]= -3
a[2]4]= -1
a[2]5]= 1
a[3]1]= 2
a[3]2]= -1
a[3]3]= 2
a[3]4]= 3
a[3]5]= 7
a[4]1]= 3
a[4]2]= 2
a[4]3]= -4
a[4]4]= 3
a[4]5]= 2

Solution:
x[1] = 1.000
x[2] = 2.000
x[3] = 2.000
x[4] = 1.000
```

GAUSS ELIMINATION

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<stdlib.h>

#define   SIZE   10

int main()
{
        float a[SIZE][SIZE], x[SIZE], ratio;
        int i,j,k,n;

        clrscr();

        /* 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 */
x[n] = a[n][n+1]/a[n][n];

for(i=n-1;i>=1;i--)
{
	x[i] = a[i][n+1];
	for(j=i+1;j<=n;j++)
	{
		x[i] = x[i] - a[i][j]*x[j];
	}
	x[i] = x[i]/a[i][i];
}
```

```
        /* Displaying Solution */
        printf("\nSolution:\n");
        for(i=1;i<=n;i++)
        {
                printf("x[%d] = %0.3f\n",i, x[i]);
        }
        getch();
        return(0);
}
```

OUTPUT

```
Enter number of unknowns: 3
a[1][1] = 1
a[1][2] = 1
a[1][3] = 1
a[1][4] = 9
a[2][1] = 2
a[2][2] = -3
a[2][3] = 4
a[2][4] = 13
a[3][1] = 3
a[3][2] = 4
a[3][3] = 5
a[3][4] = 40

Solution:
x[1] = 1.000
x[2] = 3.000
x[3] = 5.000
```