1. ROOTS OF QUADRATIC EQUATION

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
int main()
{
 double a,b,c;
 printf("Enter the values of a,b,c: ");
 scanf("%lf%lf%lf",&a,&b,&c);
 if (a == 0) {
 printf("Invalid Equation");
 }
 int d = b * b - 4 * a * c;
 double sqrt_val = sqrt(abs(d));
 if (d > 0) {
 printf("Roots are real and distinct \n");
 printf("%f\n%f", (double)(-b + sqrt_val) / (2 * a),
 (double)(-b - sqrt_val) / (2 * a));
 }
 else if (d == 0) {
 printf("Roots are real and equal \n");
 printf("%f", -(double)b / (2 * a));
 }
 else
 {
 printf("Roots are imaginary \n");
```

```
 printf("%f + i%f\n%f - i%f", -(double)b / (2 * a),
 sqrt_val / (2 * a), -(double)b / (2 * a),
 sqrt_val / (2 * a));
 }
}
```

OUTPUT:

```
PS C:\Accessable Folder\Academic\CBNST> cd "c:\Accessable Fold
Enter a,b and c: 16 4 60
Roots are imaginary:
-0.125 + i1.93245
-0.125 - i1.93245
PS C:\Accessable Folder\Academic\CBNST>
```

2. BISECTION METHOD

```
#include <stdio.h>
#include <math.h>
double cal(double x, double a, double b, double c, double d)
{
 return pow(x, 3) * a + pow(x, 2) * b + pow(x, 1) * c + d;
}
int main()
{
 double a, b, c, d,x=0,y=1,z=0;
 printf("Enter the value of a,b,c,d:\n");
 scanf("%lf%lf%lf%lf", &a, &b, &c, &d);
 int count = 0, cnt = 0;
 while (count < 3)
 {
 double fx1 = cal(x, a, b, c, d);
 double fx2 = cal(y, a, b, c, d);
 if (fx1 * fx2 < 0)
 {
 z = (x + y) / 2.0;
 double fx3 = cal(z, a, b, c, d);
 if (fx3 == 0.0)
    break;
 if (fx1 * fx3 < 0)
    y=z;
 else if (fx3 * fx2 < 0)
```

```
   x=z;
 count++;
 }
 else
 {
 x = y;
 y++;
 }
 }
 printf("Answer : %.4lf", z);
}
```

OUTPUT:

```
PS C:\Accessable Folder\Academic\CBNST> cd "c:\Ac
Enter the values of a,b,c and d: 0 2 0 -10
Answer : 2.1250
PS C:\Accessable Folder\Academic\CBNST>
```

3. NEWTON RAPHSON METHOD

```
#include<stdio.h>
#include<math.h>
float f(float x)
{
 return x*log10(x) - 1.2;
}
float df (float x)
{
 return log10(x) + 0.43429;
}
int main()
{
 int itr, maxmitr;
 float h, x0, x1, allerr;
 printf("\nEnter x0, allowed error and maximum iterations\n");
 scanf("%f %f %d", &x0, &allerr, &maxmitr);
 for (itr=1; itr<=maxmitr; itr++)
 {
 h=f(x0)/df(x0);
 x1=x0-h;
 printf(" At Iteration no. %3d, x = %9.6f\n", itr, x1);
 if (fabs(h) < allerr)
 {
 printf("After %3d iterations, root = %8.6f\n", itr, x1);
 return 0;
```

```
 }
 x0=x1;
 }
 printf(" The required solution does not converge or iterations are
insufficient\n");
 return 1;
}
```

OUTPUT:

```
PS C:\Accessable Folder\Academic\CBNST> cd "c:\
son }

Enter x0, allowed error and maximum iterations
2 0.0001 16
 At Iteration no.1, x = 2.81317
 At Iteration no.2, x = 2.74111
 At Iteration no.3, x = 2.74065
 At Iteration no.4, x = 2.74065
After 4 iterations, root =2.74065
PS C:\Accessable Folder\Academic\CBNST>
```

4. REGULA FALSI METHOD

```
#include <stdio.h>
#include <math.h>
float f(float x)
{
 return cos(x) - x * exp(x);
}
void regula(float *x, float x0, float x1, float fx0, float fx1, int *itr)
{
 *x = x0 - ((x1 - x0) / (fx1 - fx0)) * fx0;
 ++(*itr);
 printf("Iteration no. %3d X = %7.5f \n", *itr, *x);
}
void main()
{
 int itr = 0, maxmitr;
 float x0, x1, x2, x3, allerr;
 printf("\nEnter the values of x0, x1, allowed error and maximum
iterations:\n");
 scanf("%f %f %f %d", &x0, &x1, &allerr, &maxmitr);
 regula(&x2, x0, x1, f(x0), f(x1), &itr);
 do{
 if (f(x0) * f(x2) < 0)
 x1 = x2;
 else
 x0 = x2;
 regula(&x3, x0, x1, f(x0), f(x1), &itr);
```

```
if (fabs(x3 - x2) < allerr){
printf("After %d iterations, root = %6.4f\n", itr, x3);
return 0;
}
x2 = x3;
} while (itr < maxmitr);
printf("Solution does not converge or iterations not sufficient:\n");
return 1;
}
```

OUTPUT:

```
Enter the values of x0, x1, allowed error and maximum iterations:
0 1 0.0005 16
Iteration no.   1 X = 0.31467
Iteration no.   2 X = 0.44673
Iteration no.   3 X = 0.49402
Iteration no.   4 X = 0.50995
Iteration no.   5 X = 0.51520
Iteration no.   6 X = 0.51692
Iteration no.   7 X = 0.51748
Iteration no.   8 X = 0.51767
After 8 iterations, root = 0.5177
PS C:\Accessable Folder\Academic\CBNST>
```

5. GAUSS ELIMINATION METHOD

```
#include<stdio.h>
int main()
{
 int i,j,k,n;
 float A[20][20],c,x[10],sum=0.0;
 printf("\nEnter the order of matrix: ");
 scanf("%d",&n);
 printf("\nEnter the elements of augmented matrix row-wise:\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){
 c=A[i][j]/A[j][j];
 for(k=1; k<=n+1; k++)
 {
 A[i][k]=A[i][k]-c*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 solution is: \n");
for(i=1; i<=n; i++){
printf("\nx%d=%f\t",i,x[i]);
}
return(0);
}
```

OUTPUT:

```
Enter the order of matrix: 3

Enter the elements of augmented matrix row-wise:

A[1][1] : 3
A[1][2] : -1
A[1][3] : 6
A[1][4] : 4
A[2][1] : 1
A[2][2] : 5
A[2][3] : 8
A[2][4] : 5
A[3][1] : 7
A[3][2] : 6
A[3][3] : 5
A[3][4] : 7

The solution is:

x1=0.520408
x2=0.193878
x3=0.438775
PS C:\Accessable Folder\Academic\CBNST>
```