Newton Raphson

```
#include<stdio.h>
#include<math.h>
/* Defining equation to be solved.
  Change this equation to solve another problem. */
#define    f(x)    x*x-3*x+2
/* Defining derivative of g(x).
  As you change f(x), change this function also. */
#define   g(x)   2*x-3
int main()
{
        float x0, x1, f0, f1, g0, e;
        int step = 1;
    /* Inputs */
        printf("\nEnter initial guess:\n");
        scanf("%f", &x0);
        printf("Enter tolerable error:\n");
        scanf("%f", &e);

        /* 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.");
```

```
                }


                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;
                f1 = f(x1);
        }while(abs(f1)>e);
        printf("\nRoot is: %f", x1);
        // getch();
}
```

Secant method

```
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
/* Defining equation to be solved. */
#define    f(x)    x*x*x - 2*x - 5
int main()
{
        float x0, x1, x2, f0, f1, f2, e;
        int step = 1, N;
        /* 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)
                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;
        }
       while(abs(f2)>e);
        printf("\nRoot is: %f", x2);

}
```

Fixed Point

```
#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;
        /* 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;
                 x0 = x1;
        }while( fabs(f(x1)) > e);
        printf("\nRoot is %f", x1);
        getch();
        return(0);
}
```

Bisection Method

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

```
#include<conio.h>
#include<math.h>
/* Defining equation to be solved.*/
#define f(x) cos(x) - x * exp(x)
int main()
{
        float x0, x1, x2, f0, f1, f2, e;
        int step = 1;
        /* 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();
}
```

False position Method

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
/* Defining equation to be solved. */
#define   f(x)   x*log10(x) - 1.2
int main()
{
        float x0, x1, x2, f0, f1, f2, e;
        int step = 1;
        /* 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;
}
```

Lagarange Interpolation

```
#include<stdio.h>
#include<conio.h>
int 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();
}
```

Forward difference Table

```
#include<stdio.h>
#include<conio.h>
int main()
{
 float x[20] [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 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;
}
```

Trapezoidal Rule

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
/* Define function here */
#define f(x) 1/(1+pow(x,2))
int main()
{
 float lower, upper, integration=0.0, stepSize, k;
 int i, subInterval;
 /* Input */
 printf("Enter lower limit of integration: ");
 scanf("%f", &lower);
 printf("Enter upper limit of integration: ");
 scanf("%f", &upper);
 printf("Enter number of sub intervals: ");
 scanf("%d", &subInterval);
 /* Calculation */
 /* Finding step size */
 stepSize = (upper - lower)/subInterval;
 /* Finding Integration Value */
 integration = f(lower) + f(upper);
 for(i=1; i<= subInterval-1; i++)
 {
  k = lower + i*stepSize;
  integration = integration + 2 * f(k);
 }
```

```
 integration = integration * stepSize/2;
 printf("\nRequired value of integration is: %.3f", integration);
 getch();
 return 0;
}
```

Simpson's 1/3 Interpolation

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
/* Define function here */
#define f(x) 1/(1+x*x)
int main()
{
 float lower, upper, integration=0.0, stepSize, k;
 int i, subInterval;
 /* Input */
 printf("Enter lower limit of integration: ");
 scanf("%f", &lower);
 printf("Enter upper limit of integration: ");
 scanf("%f", &upper);
 printf("Enter number of sub intervals: ");
 scanf("%d", &subInterval);
 /* Calculation */
 /* Finding step size */
 stepSize = (upper - lower)/subInterval;
 /* Finding Integration Value */
 integration = f(lower) + f(upper);
 for(i=1; i<= subInterval-1; i++)
 {
```

```
 k = lower + i*stepSize;
 if(i%2==0)
 {
  integration = integration + 2 * f(k);
 }
 else
 {
  integration = integration + 4 * f(k);
 }
}
 integration = integration * stepSize/3;
 printf("\nRequired value of integration is: %.3f", integration);
 getch();
 return 0;
}
```

Simpsons 3/8

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
/* Define function here */
#define f(x) 1/(1+x*x)
int main()
{
 float lower, upper, integration=0.0, stepSize, k;
 int i, subInterval;
 /* Input */
 printf("Enter lower limit of integration: ");
 scanf("%f", &lower);
```

```
 printf("Enter upper limit of integration: ");
 scanf("%f", &upper);
 printf("Enter number of sub intervals: ");
 scanf("%d", &subInterval);
 /* Calculation */
 /* Finding step size */
 stepSize = (upper - lower)/subInterval;
 /* Finding Integration Value */
 integration = f(lower) + f(upper);
 for(i=1; i<= subInterval-1; i++)
 {
  k = lower + i*stepSize;
  if(i%3 == 0)
  {
   integration = integration + 2 * f(k);
  }
  else
  {
   integration = integration + 3 * f(k);
  }
 }
 integration = integration * stepSize*3/8;
 printf("\nRequired value of integration is: %.3f", integration);
 getch();
 return 0;
}
```

Euler's Method

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

```
#include<conio.h>

#define f(x,y) x+y
int main()
{
 float x0, y0, xn, h, yn, slope;
 int i, n;
 printf("Enter Initial Condition\n");
 printf("x0 = ");
 scanf("%f", &x0);
 printf("y0 = ");
 scanf("%f", &y0);
 printf("Enter calculation point xn = ");
 scanf("%f", &xn);
 printf("Enter number of steps: ");
 scanf("%d", &n);
 /* Calculating step size (h) */
 h = (xn-x0)/n;
 /* Euler's Method */
 printf("\nx0\ty0\tslope\tyn\n");
 printf("------------------------------\n");
 for(i=0; i < n; i++)
 {
  slope = f(x0, y0);
  yn = y0 + h * slope;
  printf("%.4f\t%.4f\t%0.4f\t%.4f\n",x0,y0,slope,yn);
  y0 = yn;
  x0 = x0+h;
 }
```

```
}
```

RK 4 Method

```
#include<stdio.h>
#include<conio.h>
#define f(x,y) (y*y-x*x)/(y*y+x*x)
int main()
{
 float x0, y0, xn, h, yn, k1, k2, k3, k4, k;
 int i, n;
 printf("Enter Initial Condition\n");
 printf("x0 = ");
 scanf("%f", &x0);
 printf("y0 = ");
 scanf("%f", &y0);
 printf("Enter calculation point xn = ");
 scanf("%f", &xn);
 printf("Enter number of steps: ");
 scanf("%d", &n);
 /* Calculating step size (h) */
 h = (xn-x0)/n;
 /* Runge Kutta Method */
 printf("\nx0\ty0\tyn\n");
 for(i=0; i < n; i++)
 {
  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)));
```

```
 k = (k1+2*k2+2*k3+k4)/6;
 yn = y0 + k;
 printf("%0.4f\t%0.4f\t%0.4f\n",x0,y0,yn);
 x0 = x0+h;
 y0 = yn;
}
/* Displaying result */
printf("\nValue of y at x = %0.2f is %0.3f",xn, yn);
getch();
return 0;
}
```