# C-Programming

**Bisection method**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<stdlib.h>
float f ( float x)
{
  float y;
  y= pow(x, 2)+x -2;
  return y;
}
void main()
{
  float x1, x2, x0, error=0.0001;
  int i=0;
  printf("\nEnter two initial guess:");
  scanf("%f%f", &x1, &x2);
  if (f(x1 )*f(x2 )>0)
  {
    printf("\nWrong Input!!");
    exit(0);
  }
else
{
  do
    {
      x0=(x1+x2)/2;
      if(f(x0 )*f(x1 )>0)
            x1=x0;
      else
            x2=x0;
     i++;
    }while(fabs (f(x0))>error);
  }
printf("\nRoot=%f", x0);
printf("\nNumber of iteration=%d",i);
getch();

}
```

**Secant Method**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
float f ( float x)
{
  float y;
  y= pow(x, 2)+x -2;
  return y;
}
void main()
{
  float x1, x2, x0, error=0.0001;
  int i=0;
  printf("\nEnter two initial guess:");
  scanf("%f%f", &x1, &x2);
  do
    {
      x0=x1-(f(x1)*(x2-x1))/(f(x2)-f(x1));
      x2=x1;
      x1=x0;
      i++;
    }while(fabs (f(x0))>error);
  printf("\nRoot=%f", x0);
  printf("\nNumber of iteration=%d",i);
  getch();

}
```

**Newton Raphson Method**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
float f( float x)
{
   float y;
   y= pow(x, 2)+x -2;
   return y;
}
float fd( float x)
{
   float y;
   y= 2*x+1;
   return y;
}
void main()
{
   float x0,x1,error=0.0001;
   int i=0;
   printf("\nGuess initial root:");
   scanf("%f", &x1);
   do
     {
        x0=x1-(f(x1)/fd(x1));
        x1=x0;
        i++;
     }while(fabs (f(x0))>error);
   printf("\nRoot=%f", x0);
  printf("\nNumber of iteration=%d",i);
  getch();
}
```

**Fixed Point Method**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
float g ( float x)
{
   float y;
   y= 2.0-x*x;
   return y;
}
int main()
{
    float x0, x, error, E=0.00001;
   printf("Input initial estimate of a root:\n");
   scanf("%f", &x0);
   while(1)
   {
      x=g(x0);
      error=(x-x0)/x;
      if(fabs(error)<E)
      {
         printf("\nRoot=%f", x0);
         break;
      }
     x0=x;
   }
  getch();
  return 0;

}
```

***Lagranges Interpolation***

```
#include<stdio.h>
#include<conio.h>
int main()
{
  float x[10], f[10], y, sum=0.0, l;
 int n, i, j;
 printf("\nInput number of data:");
 scanf("%d", &n);
 printf("\nInput data points x(i) & f(i):\n");
for(i=0;i<n;i++)
{
   printf("x[%d]=",i);
   scanf("%f", &x[i]);
   printf("f[%d]=",i);
   scanf("%f", &f[i]);
}
printf("\nFunctional value:");
scanf("%f", &y);
for(i=0;i<n;i++)
{
   l=1;
   for(j=0;j<n;j++)
  {
    if(j!=i)
    {
      l=l*(y-x[j])/(x[i]-x[j]);
    }
 }
sum=sum+l*f[i];
}
printf("\nValue at %f=%f", y, sum);
getch();
return 0;
}
```

***Curve Fitting(Fitting Linear Equation)***

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#define error 0.001
int main()
{
  int i, n;
  float x[10], y[10], sumx=0.0,sumy=0.0;
  float sumxx=0.0, sumxy=0.0;
 float meanx, meany, denom, a, b;
 printf("how many element?:");
 scanf("%d", &n);
for(i=0;i<n;i++)
{
 printf("x[%d]=",i);
 scanf("%f",&x[i]);
 printf("y[%d]=",i);
 scanf("%f",&y[i]);
}
for(i=0;i<n;i++)
{
 sumx+=x[i];
 sumy+=y[i];
 sumxx+=x[i]*x[i];
 sumxy+=x[i]*y[i];
}
meanx=sumx/n;
meany=sumy/n;
denom=n*sumxx-sumx*sumx;
if(fabs(denom)>error)
{
 b=(n*sumxy-sumx*sumy)/denom;
 a=meany-b*meanx;
 printf("y=%fx+%f",b,a);
}
else
{
 printf("\nNo Solution");
}
getch();
return 0;
}
```

***Trapezoidal Rule***

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
float f(float x)
{
   return (1-exp(-x/2.0));
}
void main()
{
   float a, b, h, x, sum=0;
  int n;
  printf("Enter initial and final value of
x:\n");
 scanf("%f%f", &a, &b);
 printf("\nNumber of segments:");
 scanf("%d", &n);
 h=(b-a)/n;
for(x=a;x<=b;x=x+h)
{
   if(x==a)
        sum=sum+f(x);
   else if(x==b)
        sum=sum+f(x);
   else
        sum=sum+2*f(x);
}
sum=sum*h/2;
printf("\nIntegral value of f(x)=%f ", sum);
getch();
}
```

***Simpson's 1/3 Rule***

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
float f(float x)
{
   return (1-exp(-x/2.0));
}
void main()
{
   float a, b, h, x,ans,sum=0;
   int n,i;
  printf("Enter initial and final value of
x:\n");
 scanf("%f%f", &a, &b);
 printf("\nNumber of segments:");
 scanf("%d", &n);
 h=(b-a)/n;
 for(i=1;i<n;i++)
 {
    x=a+i*h;
    if(i%2==0)
   {
      sum=sum+2*f(x);
     }
    else{
      sum=sum+4*f(x);
     }
  }
    ans=(h/3)*(f(a)+f(b)+sum);
printf("\nIntegral value of f(x)=%f ", ans);
getch();
}
```

**Simpson's 3/8 Rule**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
float f(float x)
{
   return (1-exp(-x/2.0));
}
void main()
{
   float a, b, h, x,ans,sum=0;
   int n,i;
  printf("Enter initial and final value of
x:\n");
 scanf("%f%f", &a, &b);
 printf("\nNumber of segments:");
 scanf("%d", &n);
 h=(b-a)/n;
 for(i=1;i<n;i++)
 {
    x=a+i*h;
    if(i%3==0)
    {
      sum=sum+2*f(x);
    }
    else{
      sum=sum+3*f(x);
    }
  }
    ans=(3*h/8)*(f(a)+f(b)+sum);
printf("\nIntegral value of f(x)=%f ", ans);
getch();
}
```

**Euler Method**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
float fun( float x, float y)
{
   float f;
   f = x*y;
   return f;
}
int main()
{
  int i, n;
  float x0, y0, xp, h, y;
  printf("Enter initial value of x and y:");
  scanf("%f%f",&x0,&y0);
  printf("Enter x at which y is required:");
  scanf("%f",&xp);
  printf("Enter step-size,h:");
  scanf("%f",&h);
  n=(xp - x0)/h;
  for(i=0; i < n; i++)
  {
    y=y0+h*fun(x0,y0);
    x0=x0+h;
    y0=y;
    printf("%f\t%f\n",x0,y);
   }
  printf("\nValue of y at x=%f id %f",x0,y0);
  getch();
}
```

**Heun's Method**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
float func(float x, float y)
{
   float f;
   f=2.0*y/x;;
   return f;
}
int main()
{
  int i, n;
  float x0, y0, xp, h, m1, m2;
  printf("Enter initial value of x and y:");
  scanf("%f %f", &x0,&y0);
  printf("Enter x at which y is required:");
  scanf("%f", &xp);
  printf("Enter stepsize,h: ");
  scanf("%f", &h);
  n = (xp - x0)/h;
  for(i=1; i<=n; i++)
  {
    m1 = func(x0,y0);
    m2 = func(x0+h, y0+m1*h);
    x0 = x0+h;
    y0 = y0+0.5*h*(m1+m2);
    printf("%f \t%f\n", x0,y0);
  }
  printf("\nValue of y at x=%f is %f",x0,y0);
  getch();
  return 0;
}
```

**4th Order Runge-Kutta Method**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
float func(float x, float y)
{
   float f;
   f=2.0*y/x;;
   return f;
}
int main()
{
  int i,n;
  float x0, y0, xp, h, m1, m2, m3, m4;
  printf("Enter initial value of x and y:");
  scanf("%f %f", &x0,&y0);
  printf("Enter x at which y is required:");
  scanf("%f", &xp);
  printf("Enter stepsize,h: ");
  scanf("%f", &h);
  n = (xp - x0)/h;
  for(i=1; i<=n; i++)
  {
    m1 = func(x0,y0);
    m2 = func(x0+0.5*h, y0+0.5*m1*h);
    m3 = func(x0+0.5*h, y0+0.5*m2*h);
    m4 = func(x0+h, y0+m3*h);
    x0 = x0+h;
    y0 = y0+(m1+2*m2+2*m3+m4)*h/6;
    printf("%f\t%f\n", x0, y0);
  }
  printf("\nValue of y at x=%f is %f",x0, y0);
  getch();
  return 0;
  }
```

**Gauss Elimination Method**

```
#include <stdio.h>
#include<conio.h>
int main()
{
   int i,j,k,n;
   float A[20][20],r,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]);
      }
   }
   /*Generation of upper triangular matrix*/
   for(j=1; j<=n; j++)
   {
      for(i=1; i<=n; i++)
      {
         if(i>j)
         {
            r=A[i][j]/A[j][j];
            for(k=1; k<=n+1; k++)
            {
               A[i][k]=A[i][k]-r*A[j][k];
            }
         }
      }
   }
   x[n]=A[n][n+1]/A[n][n];
   /*backward substitution*/
   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]);
   }
   getch();
   return 0;
}
```

**Gauss Jordan Method**

```
#include<stdio.h>
#include<conio.h>
int main()
{
   int i,j,k,n;
   float A[20][20],r,x[10];
   printf("\nEnter the size of matrix: ");
   scanf("%d",&n);
   printf("\nEnter the elements of augmented
matrix row-wise:\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]);
      }
   }
   /* finding diagonal matrix */
   for(j=1; j<=n; j++)
   {
      for(i=1; i<=n; i++)
      {
         if(i!=j)
         {
            r=A[i][j]/A[j][j];
            for(k=1; k<=n+1; k++)
            {
               A[i][k]=A[i][k]-r*A[j][k];
            }
         }
      }
   }
   printf("\nThe solution is:\n");
   for(i=1; i<=n; i++)
   {
      x[i]=A[i][n+1]/A[i][i];
      printf("\n x%d=%f\n",i,x[i]);
   }
   getch();
   return 0;
}
```

**Gauss Jacobi Iteration Method**

```
/* Arrange system of linear equations in diagonally dominant form and convert the 1st equation in tems of 1st variable (f1), 2nd equation in terms of 2nd variable (f2) and so on  */

#include<stdio.h>
#include<conio.h>
#include<math.h>
#define f1(x,y,z)  (15-y-z)/10
#define f2(x,y,z)  (24-x-z)/10
#define f3(x,y,z)  (33-x-y)/10

int main()
{
 float x0=0, y0=0, z0=0, x1, y1, z1, e1, e2, e3, e;
 int i=1;
 printf("Enter the allowed error:\n");
 scanf("%f", &e);
 printf("\ni\tx\ty\tz\n");
 do
 {
  /* Calculation */
  x1 = f1(x0,y0,z0);
  y1 = f2(x0,y0,z0);
  z1 = f3(x0,y0,z0);
  printf("%d\t%f\t%f\t%f\n",i, x1,y1,z1);

  /* Error */
  e1 = fabs(x0-x1);
  e2 = fabs(y0-y1);
  e3 = fabs(z0-z1);
  i++;

  /* Set value for next iteration */
  x0 = x1;
  y0 = y1;
  z0 = z1;
 }while(e1>e && e2>e && e3>e);

 printf("\nSolution: x=%f, y=%f and z = %f\n",x1,y1,z1);
 getch();
 return 0;
}
```

**Gauss Seidal Iteration Method**

```
/* Arrange system of linear equations in diagonally dominant form and convert the 1st equation in tems of 1st variable (f1), 2nd equation in terms of 2nd variable (f2) and so on  */

#include<stdio.h>
#include<conio.h>
#include<math.h>
#define f1(x,y,z)  (15-y-z)/10
#define f2(x,y,z)  (24-x-z)/10
#define f3(x,y,z)  (33-x-y)/10

int main()
{
 float x0=0, y0=0, z0=0, x1, y1, z1, e1, e2, e3, e;
 int i=1;
 printf("Enter the allowed error:\n");
 scanf("%f", &e);
 printf("\ni\tx\ty\tz\n");
 do
 {
  /* Calculation */
  x1 = f1(x0,y0,z0);
  y1 = f2(x1,y0,z0);
  z1 = f3(x1,y1,z0);
  printf("%d\t%f\t%f\t%f\n",i, x1,y1,z1);

  /* Error */
  e1 = fabs(x0-x1);
  e2 = fabs(y0-y1);
  e3 = fabs(z0-z1);
  i++;

  /* Set value for next iteration */
  x0 = x1;
  y0 = y1;
  z0 = z1;
 }while(e1>e && e2>e && e3>e);

 printf("\nSolution: x=%f, y=%f and z = %f\n",x1,y1,z1);
 getch();
 return 0;
}
```

For more notes visit:

https://collegenote.pythonanywhere.com/