**Lagrange’s Interpolation Formula:**

In the preceding sections, in the interpolation formula, the values of the independent variable to equally spaced. It is therefore desirable to have interpolation formula with unequally spaced values of the argument. **Lagrange’s Interpolation is such formula.**

Let, y(x) be continuous and differentiable (n+1) times in the interval (a, b). Given the (n+1) points (x0, y0), (x1, y1), (x2, y2), … (xn, yn) where the values of x need not necessarily be equally spaced, we wish to find a polynomial of degree n, say Ln(x), such that

Ln (xi) = y(xi) = yi, i = 0,1,2,3, … n

L1 (x) = y0 + 1

=l0(x)y0 + l1(x)y1

= i(x)yi

L2 (x) = 0 + 1 + 2

= i(x)yi

=l0(x)y0 + l1(x)y1 + l2(x)y2

**Source Code:**

#include<bits/stdc++.h>

using namespace std;

int main()

{

float x[20],y[20],Xn,Yn=0,Ynn=1,x\_r=0,y\_r=0;

int n,i,j;

cout<<"Input the number of co-ordinates: "<<endl;

cin>>n;

cout<<"\nEnter the value of x & y :\n"<<endl;

cout<<"X Y"<<endl;

for(i=0;i<n;i++)

{

cin>>x[i]>>y[i];

}

printf("\n\nInput the value of x(n) : ");

scanf("%f",&Xn);

for(i=0;i<n;i++)

{

Ynn=1;

x\_r=x[i];

y\_r=y[i];

for(j=0;j<n;j++)

{

if(i==j)

Ynn=Ynn\*y[j];

else

Ynn=Ynn\*((Xn-x[j])/(x[i]-x[j]));

}

Yn=Yn+Ynn;

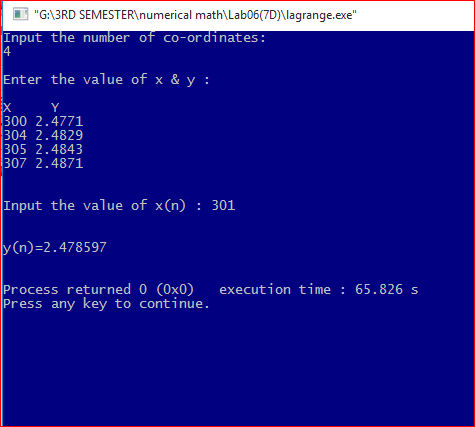
}

printf("\n\ny(n)=%f\n\n",Yn);

return 0;

}

**Input/Output:**



**Least Square Curve Fitting:**

The strategy for least square curve fitting procedure is to derive an approximating function that broadly fits the data without necessarily passing through the given points. The curve drawn is such that the discrepancy between the data points and the curve is least . In this method , the sum of the squares of the errors is minimized.

Let the set of the points be (x0,y0),i=1,2,3….m and let the curve given by Y=*f*(x0 be fitted to this data. At x=xn the given ordinate is yi and the corresponding value on the fitting curve is *f*(xi). If e1 is the error of approximation at x=x0 then we have

ei=yi- *f*(xi).

If we write

S=[ y1- *f*(x1)]2 +[y2- *f*(x2)]2+…………+[ ym- *f*(xm)]2

= e12+ e22+……+ em2

Then the method of least square consists in minimizing S. i.e. the sum of the square of the errors. In the following sections, we shall study the linear and nonlinear least square fitting to given data(x0,y0),i=1,2,3….m.

**Source code:**

#include<bits/stdc++.h>

using namespace std;

int main()

{

int i,m;

float x[10],y[10],sum1=0,sum2=0,sum3=0,sum4=0;

float avr1, avr2,a1,a0;

cout<<"The Itaration number : ";

cin>>m;

cout<<"\nEnter the value of x & y :\n"<<endl;

cout<<"X Y"<<endl;

for(i=0;i<m;i++)

{

cin>>x[i]>>y[i];

sum1+=x[i];

sum2+=y[i];

sum3+=x[i]\*x[i];

sum4+=x[i]\*y[i];

}

avr1=sum1/m;

avr2=sum2/m;

cout<<"\n\t Xi Yi X\*X Xi\*Yi\n"<<endl;

printf("\t-----------------------------------");

for(i=0;i<m;i++){

printf("\n\t %.3f %.3f %.3f %.3f ",x[i],y[i],x[i]\*x[i],x[i]\*y[i]);

}

cout<<"\n\t-----------------------------------"<<endl;

printf("\n\t%.3f %.3f %.3f %.3f \n",sum1,sum2,sum3,sum4);

a1=(m\*sum4-sum1\*sum2)/(m\*sum3-sum1\*sum1);

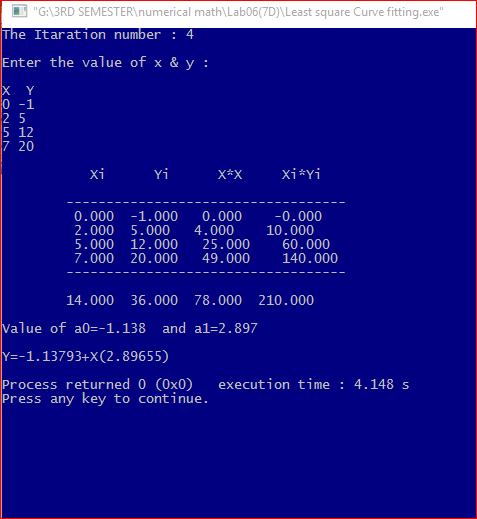
a0=avr2-a1\*avr1;

printf("\nValue of a0=%.3f and a1=%.3f\n",a0,a1);

cout<<"Y="<<a0<<"+X("<<a1<<")\n";

}

**Input/Output:**

****

**Discussion:**

Lagrange Interpolation refers to the process of creating new data points within the given set of data, where the independent set of data are unequally spaced.

The least square curve fitting procedure is a mathematical procedure for finding the best fitting curve to a given set of points by minimizing the sum of the squares of the offsets of the points from the curve. The procedure used here is the linear which is the most commonly applied form.