**Lagrange’s Formula**

**Theory**

This is again an **N**th degree polynomial approximation formula to the function **f(x)**, which is known at discrete points **xi**, i **=** 0, 1, 2 **. . . N**th. The formula can be derived from the Vandermonds determinant but a much simpler way of deriving this is from Newton's divided difference formula. If f(x) is approximated with an Nth degree polynomial then the Nth divided difference of f(x) constant and (N+1)th divided difference is zero. That is

f [x0, x1, . . . xn, x] = 0

From the second property of divided difference we can write

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| f0 | **+** | fn |  | fx | **=** 0 |
|  |  | **+ . . . +** |  |
| (x0- x1) . . . (x0- xn)(x0- x) | (xn- x0) . . . (xn - xn-1)(xn- x) |  | (x- x0) . . . (x- xn) |

or

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | (x- x1) . . . (x- xn) |  | (x- x0) .  .  . (x - xn-1) |  |
| f(x)**=** |  | f0**+ .   .   . +** |  | fn |
|  | (x0- x1) . . . (x0- xn) |  | (xn- x0) .  .  . (xn - xn-1) |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| n | ( | n |  |  | )fi |  |
|  | |  | |  | x - xj |  |
| j = 0 |  | (xi - xj) |  |
| i = 0 | j 1 |  |  |  |

Since Lagrange's interpolation is also an Nth degree polynomial approximation to f(x) and the Nth degree polynomial passing through (N+1)points is unique hence the Lagrange's and Newton's divided difference approximations are one and the same. However, Lagrange's formula is more convenient to use in computer programming and Newton's divided difference formula is more suited for hand calculations.

**Code**

#include<bits/stdc++.h>

using namespace std;

int main()

{

int n,x[10],y[10],X;

double result=0;

cout<<"How many pairs of x and y: ";

cin>>n;

cout<<endl;

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

{

cout<<"Enter the value of x"<<i<<" : ";

cin>>x[i];

cout<<"Enter the value of y"<<i<<" : ";

cin>>y[i];

}

cout<<"\nEnter the value of x for which the value of y will be determined: ";

cin>>X;

int i=0;

do

{

int j=0;

double numerator=1,denominator=1;

do

{

if(i!=j)

{

numerator = numerator\*(X-x[j]);

denominator = denominator\*(x[i]-x[j]);

}

j = j+1;

}while(j<n);

result = result + (numerator/denominator)\*y[i];

i = i+1;

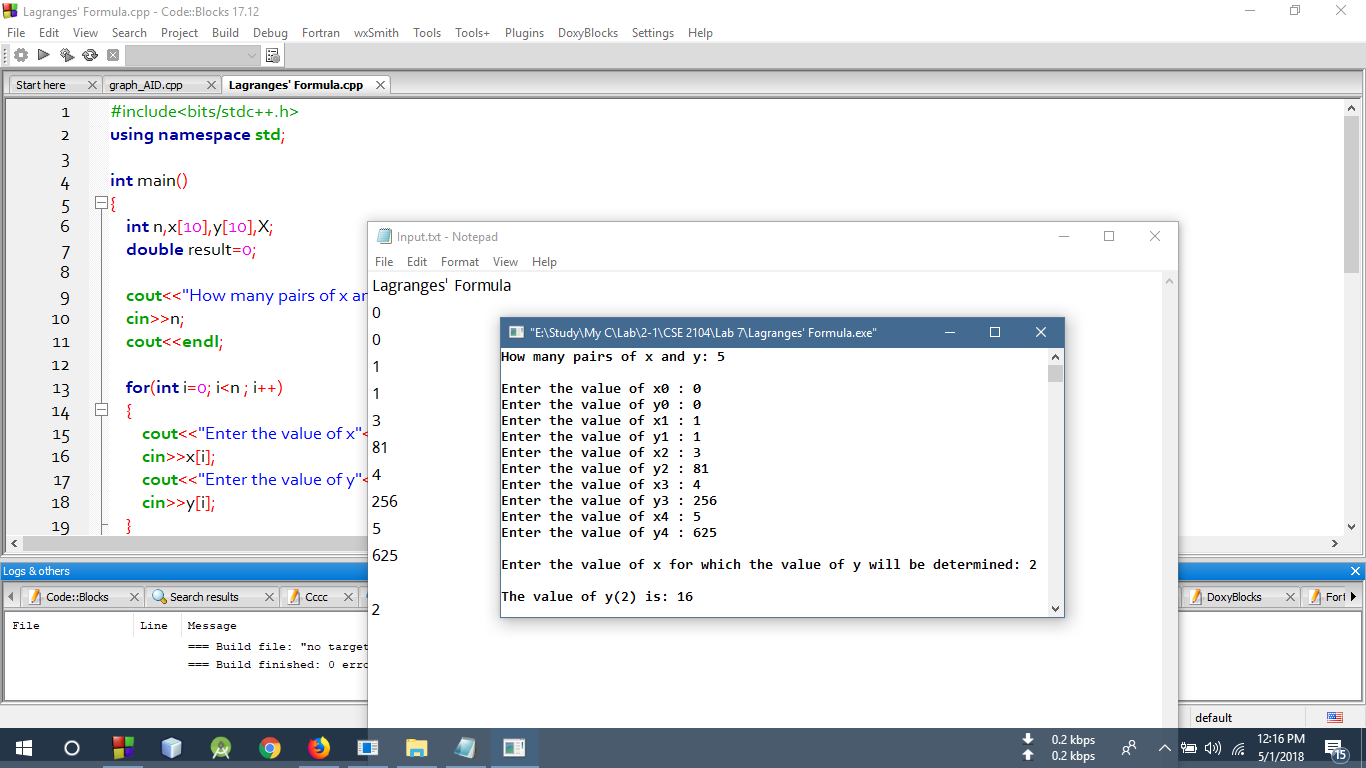
}while(i<n);

cout<<"\nThe value of y("<<X<<") is: "<<result<<endl;

return 0;

}

**Output**



**Discussion**

In this above code, lagrange’s Interpolation formula was applied where the the pairs of the values of x and y were taken as input and for a particular value of x, the value of y was determined using this formula. In this program, a only do while loop was used to execute the main formula of lagrange. Thus in this process, the value of y for x has been calculated.