Jahangirnagar University
Department of Computer Science and Engineering

LAB REPORT
ON
CSE-206 (NUMERICAL METHODS LAB)

EXPERIMENT No.: 06

EXPERIMENT NAME: Determining the interpolation value at a specified point, given a set of data points, using the Lagrange Interpolation representation.

SUBMITTED BY			SUBMITTED TO
Sumaita Binte Shorif			**Dr. Md. Golam Moazzam**
Class Roll: 357			**Professor**
Exam Roll: 191338			**Dept. of Computer Science & Engineering**
Submission Date:			**Jahangirnagar University**
Sept 03, 2021			

Experiment No: 06

Name of the Experiment: Determining the interpolation value at a specified point, given a set of data points, using the Lagrange Interpolation representation.

Objectives:

- Understanding the process of finding interpolation value at a specified point given a set of data points, using Lagrange Interpolation polynomial representation.
- Executing the implementation of the algorithm of Lagrange Interpolation Polynomial.
- To get the value at a specified point as accurate as possible.
- To be able to interpret the advantages and disadvantages of Lagrange Interpolation Polynomial.

Theory:

Let $x_0, x_1, \dots, x_n$ denote n distinct real numbers and let $f_0, f_1, \dots, f_n$ be arbitrary real numbers. The points (x_0, f_0), (x_1, f_1), (x_2, f_2),… ., (x_n, f_n) can be imagined to be data values connected by a curve.

Any function p(x) satisfying the conditions

$p(x_k) = f_k$ for k = 0, 1, …., n

is called *interpolation function*. An interpolation function is, therefore, a curve that passes through the data points as pointed out.

Let us consider a second order polynomial of the form

$$p_2(x) = b_1(x-x_0)(x-x_1) + b_2(x-x_1)(x-x_2) + b_3(x-x_2)(x-x_0)\text{……..(1)}$$

If (x_0, f_0), (x_1, f_1), and (x_2, f_2) are the three interpolating points, then we have

$$p_2(x_0) = f_0 = b_2(x_0 - x_1)(x_0 - x_2)$$

Or,

$$b_2 = \frac{f_0}{(x_0 - x_1)(x_0 - x_2)}$$

Or,

$$p_2(x_1) = f_1 = b_3(x_1 - x_2)(x_1 - x_0)$$

Or,

$$b_3 = \frac{f_1}{(x_1 - x_2)(x_1 - x_0)}$$

$$p_2(x_2) = f_2 = b_1(x_2 - x_0)(x_2 - x_1)$$

Or,

$$b_1 = \frac{f_2}{(x_2 - x_0)(x_2 - x_1)}$$

Substituting $\mathrm{b}_1, \mathrm{b}_2$ and b_3 in equation (1), we get,

$$p_2(x) = \frac{f_2(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} + \frac{f_0(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} + \frac{f_1(x - x_2)(x - x_0)}{(x_1 - x_2)(x_1 - x_0)}$$

Equation (2) may be represented as

$$p_2(x) = f_0 l_0(x) + f_1 l_1(x) + f_2 l_2(x) = \sum_0^n f_0 l_0\,(x)$$

Where,

$$l_i(x) = \prod_{j=0\,,j!=i}^{2} \frac{x - x_j}{x_i - x_j}$$

In general for n+1 points we obtain n-th degree polynomial as,

$$p_n(x) = \sum_0^n f_i l_i\,(x)$$

$$l_i(x) = \prod_{j=0\,,j!=i}^{n} \frac{x - x_j}{x_i - x_j}$$

Algorithm:

Lagrange Interpolation:

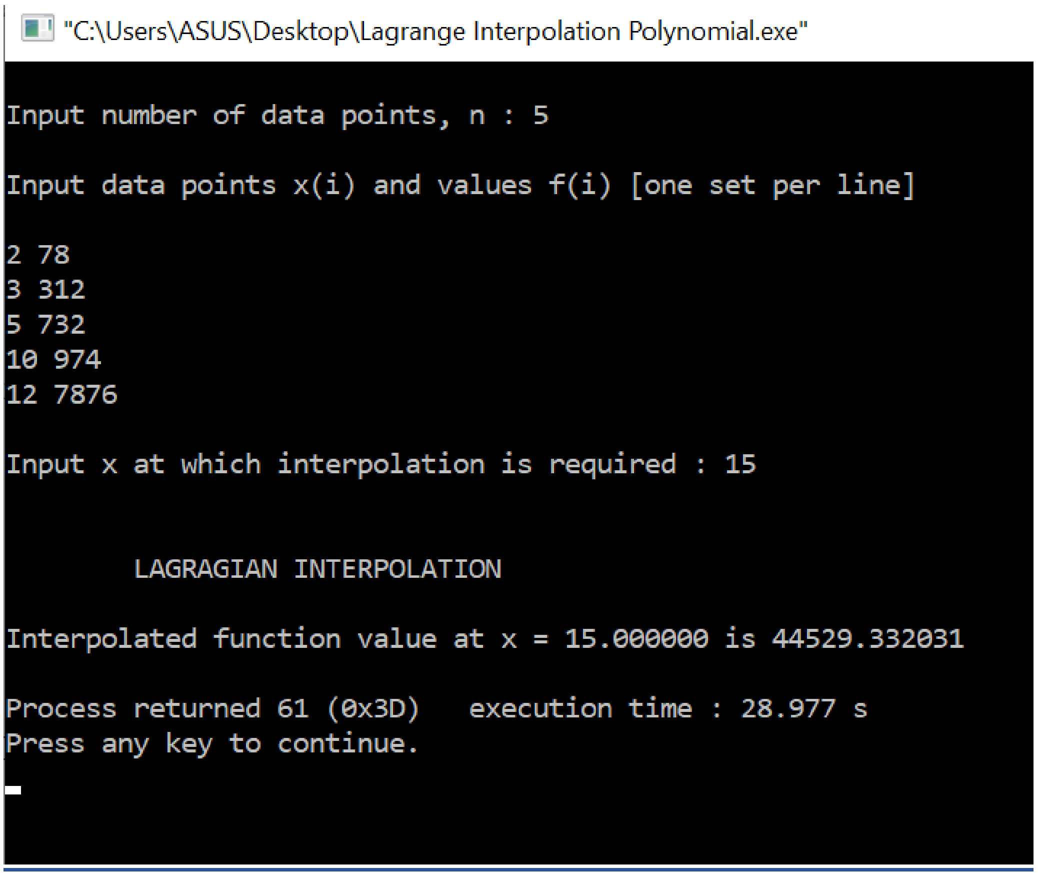
```
1. Start
2. Read number of data n.
3. For I = 1 to n
       Read data xi and fi.
4. Read input xp at which interpolation is required.
5. Set sum = 0.
6. For i = 1 to n
       Set lf = 1.0, where lf is lagrage interpolation factor.
           For j = 1 to n
               if ( i!=j) set lf = lf * (xp - x[j]) / (x[i] - x[j])
                   sum=sum+(lf * fi)
7. fp=sum
8. Print fp
9. End.
```

Coding in C:

```c
#include<stdio.h>
#define MAX 10
int main()
{
    int n,i,j;
    float x[MAX], f[MAX], fp, lf, sum, xp;
    printf("\nInput number of data points, n : ");
    scanf("%d",&n);
    printf("\nInput data points x(i) and values f(i) [one set per 
line]\n\n");
    for(i = 1; i <= n; i++)
    {
        scanf("%f %f",&x[i],&f[i]);
    }
    printf("\nInput x at which interpolation is required : ");
    scanf("%f",&xp);
    printf("\n\n");
    sum = 0.0;
    for(i = 1; i <= n; i++)
    {
        lf = 1.0;
        for(j = 1; j <= n; j++)
        {
            if(i != j) lf = lf * (xp - x[j]) / (x[i] - x[j]);
        }
```

```
            sum += (lf * f[i]);
        }
        fp = sum;
        printf("\tLAGRAGIAN INTERPOLATION\t\n\n");
        printf("Interpolated function value at x = %.6f is %.6f\n",xp,fp);
}
```

Output:



```
"C:\Users\ASUS\Desktop\Lagrange Interpolation Polynomial.exe"

Input number of data points, n : 5

Input data points x(i) and values f(i) [one set per line]

2 78
3 312
5 732
10 974
12 7876

Input x at which interpolation is required : 15


        LAGRAGIAN INTERPOLATION

Interpolated function value at x = 15.000000 is 44529.332031

Process returned 61 (0x3D)   execution time : 28.977 s
Press any key to continue.
```

Fig 01: Output Obtained.

Discussion:

Although it’s an easy method to implement, it has some advantages and disadvantages.

Advantages:

The Lagrange form of the interpolation polynomial shows the linear character of polynomial interpolation and the uniqueness of the interpolation polynomial. Therefore, it is preferred in proofs and theoretical arguments.

Disadvantages:

It requires 2(n+1) multiplications/divisions and 2n+1 additions and subtractions. If we want to add one more data point, we have to compute the polynomial from the beginning. It does not use the polynomial that has already been computed.

As we can observe from the above experiment, Lagrange Interpolation formula requires 2(n+1) multiplications/divisions and 2n+1 additions and subtractions. If we want to add one more data point, we have to compute the polynomial from the beginning. It does not use the polynomial already computed.
