CEE394
Polynomial Interpolation using Divided Differences in C++

contact@thebinarysolutionsllc.com

Introduction

This program performs polynomial interpolation using divided differences, a method commonly used to approximate functions at specified points. It demonstrates the process of constructing an interpolating polynomial and evaluating it at a given point.

contact@thebinarysolutionsllc.com

Problem Statement

Given four data points $(x0, y0), (x1, y1),(x2, y2)$, and $(x3, y3)$, the task is to interpolate the function at a specified point xx using divided differences.

x	y
x_0=3	y_0=7
x_1=4	y_1=3
x_2=2.5	y_2=6.5
x_3=5	y_3=1

Solution Steps

- Define the four data points $(x0, y0)$, $(x1, y1)$,$(x2, y2)$, and $(x3, y3)$,.
- Specify the point xx at which the interpolation is to be performed.
- Calculate the divided differences $f01$, $f02$, $f03$, $f12$, $f13$, and $f23$ using the given data points.
- Compute the interpolated values $p0$, $p1$, $p2$, and $p3$ using the divided differences and the given data points.
- Output the interpolated values at the specified point x.

Pseudo Code

1. Begin main function.

    1.1 Define the given data points:

        x0, y0 = (3, 7)

        x1, y1 = (4, 3)

        x2, y2 = (2.5, 6.5)

        x3, y3 = (5, 1)

    1.2 Define the x value at which interpolation is required:

        x = 3.4

    1.3 Calculate divided differences:

        f_0 = y0

        f_01 = (y1 - y0) / (x1 - x0)

        f_12 = (y2 - y1) / (x2 - x1)

        f_02 = (f_12 - f_01) / (x2 - x0)

        f_23 = (y3 - y2) / (x3 - x2)

        f_13 = (f_23 - f_12) / (x3 - x1)

        f_03 = (f_13 - f_02) / (x3 - x0)

Pseudo Code

```
f_03 = (f_13 - f_02) / (x3 - x0)

    1.4 Calculate interpolated values:
        p0 = f_0
        p1 = p0 + f_01 * (x - x0)
        p2 = p1 + f_02 * (x - x0) * (x - x1)
        p3 = p2 + f_03 * (x - x0) * (x - x1) * (x - x2)

    1.5 Output the interpolated values for x = 3.4:
        "Interpolated value at x = 3.4:"
        "p0: " followed by the value of p0
        "p1: " followed by the value of p1
        "p2: " followed by the value of p2
        "p3: " followed by the value of p3

    1.6 End main function.
```

contact@thebinarysolutionsllc.com

C++ Code

```
#include <iostream>
using namespace std;
int main() {
double x0 = 3, y0 = 7;
    double x1 = 4, y1 = 3;
    double x2 = 2.5, y2 = 6.5;
    double x3 = 5, y3 = 1;
    // x value
    double x = 3.4;
    // Calculate divided differences
    double f_0 = y0;
    double f_01 = (y1 - y0) / (x1 - x0);
    double f_12 = (y2 - y1) / (x2 - x1);
    double f_02 = (f_12 - f_01) / (x2 - x0);
    double f_23 = (y3 - y2) / (x3 - x2);
    double f_13 = (f_23 - f_12) / (x3 - x1);
    double f_03 = (f_13 - f_02) / (x3 - x0);
```

contact@thebinarysolutionsllc.com

C++ Code

```
// Interpolated values
    double p0 = f_0;
    double p1 = p0 + f_01 * (x - x0);
    double p2 = p1 + f_02 * (x - x0) * (x - x1);
    double p3 = p2 + f_03 * (x - x0) * (x - x1) * (x - x2);
cout << "Interpolated value at x = " << x << ":" << endl;
    cout << "p0: " << p0 << endl;
    cout << "p1: " << p1 << endl;
    cout << "p2: " << p2 << endl;
    cout << "p3: " << p3 << endl;
    return 0;
}
```

contact@thebinarysolutionsllc.com

Code Explanation

- **#include <iostream>using namespace std;**

These lines include the necessary header file for input/output stream functionality and use the ‘using namespace std;’ directive to avoid having to prefix standard library elements with ‘std::’.

- **int main() {**

This line marks the beginning of the ‘main’ function, which serves as the entry point of the program.

- **double x0 = 3, y0 = 7; double x1 = 4, y1 = 3; double x2 = 2.5, y2 = 6.5; double x3 = 5, y3 = 1;**

These lines define four data points for polynomial interpolation.

- **double x = 3.4;**

This line specifies the x-value at which interpolation is to be performed.

- **double f_0 = y0;**

This line calculates the zeroth divided difference.

contact@thebinarysolutionsllc.com

Code Explanation

- **double f_01 = (y1 - y0) / (x1 - x0);**

This line calculates the first divided difference.

- **double f_12 = (y2 - y1) / (x2 - x1); double f_02 = (f_12 - f_01) / (x2 - x0);**

These lines calculate the second divided difference.

- **double f_23 = (y3 - y2) / (x3 - x2); double f_13 = (f_23 - f_12) / (x3 - x1); double f_03 = (f_13 - f_02) / (x3 - x0);**

These lines calculate the third divided difference.

- **double p0 = f_0; double p1 = p0 + f_01 * (x - x0); double p2 = p1 + f_02 * (x - x0) * (x - x1); double p3 = p2 + f_03 * (x - x0) * (x - x1) * (x - x2);**

These lines compute the interpolated values using the divided differences.

Code Explanation

- **cout << "Interpolated value at x = " << x << ":" << endl; cout << "p0: " << p0 << endl; cout << "p1: " << p1 << endl; cout << "p2: " << p2 << endl; cout << "p3: " << p3 << endl;**

These lines output the interpolated values for the specified x-value.

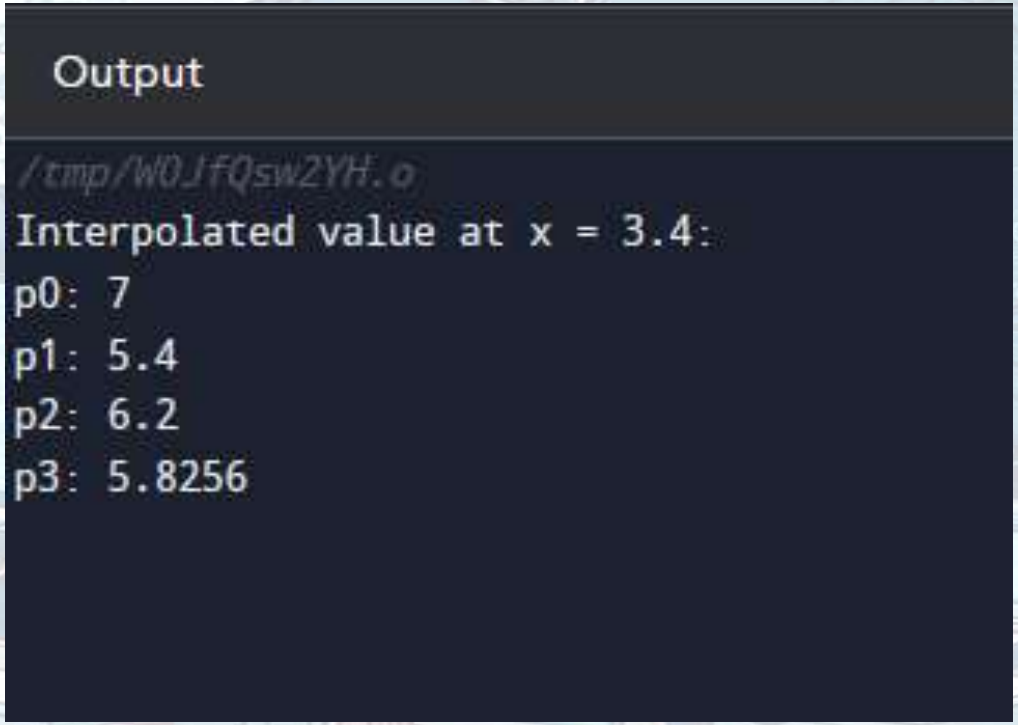
- **return 0;}**

This line indicates the end of the ‘main’ function and returns an integer value of ‘0’ to the operating system, typically indicating successful execution.

contact@thebinarysolutionsllc.com

Final Answer

The program outputs the interpolated values $p0$, $p1$, $p2$, and $p3$ at the specified point x.



Additional Comments/Tips

- Ensure the correctness of the given data points and the specified point xx for accurate interpolation.
- Verify the accuracy of the interpolated values through comparison with known results or by testing against other methods.

contact@thebinarysolutionsllc.com

Conclusion

This program demonstrates the process of polynomial interpolation using divided differences in C++, providing an efficient way to approximate functions at specified points.

contact@thebinarysolutionsllc.com