Nama : Alia Niswah

NIM : 21120122130063

Mata Kuliah : Metode Numerik

Tugas Implementasi Interpolasi

Metode Polinom Lagrange

Source code

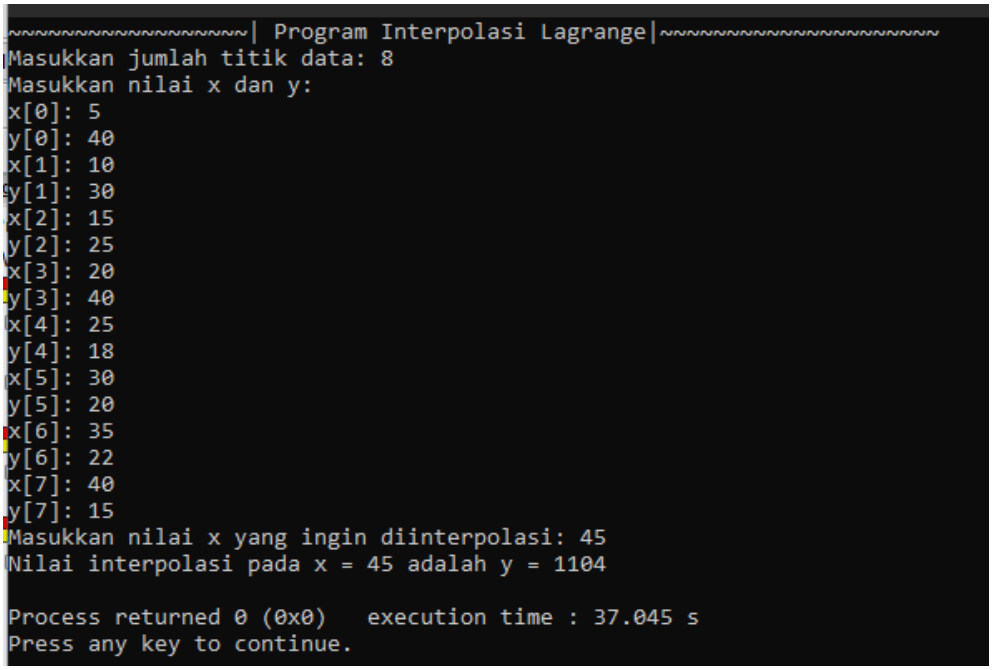
```
#include <iostream>
#include <vector>

// Fungsi untuk menghitung nilai interpolasi polinom Lagrange
double lagrangeInterpolation(const std::vector<double>& x, const
std::vector<double>& y, double xInterp) {
    double result = 0.0;
    int n = x.size();

    for (int i = 0; i < n; ++i) {
        double term = y[i];
        for (int j = 0; j < n; ++j) {
            if (j != i) {
                term *= (xInterp - x[j]) / (x[i] - x[j]);
            }
        }
        result += term;
    }

    return result;
}

int main() {
    // Data input dari pengguna
    int numPoints;
    std::cout<<"~~~~~~~~~~~~~~~~~~";
    std::cout<<"| Program Interpolasi Lagrange|";
    std::cout<<"~~~~~~~~~~~~~~~~~~~~";
    std::cout << "\nMasukkan jumlah titik data: ";
    std::cin >> numPoints;

    std::vector<double> x(numPoints), y(numPoints);

    std::cout << "Masukkan nilai x dan y:\n";
    for (int i = 0; i < numPoints; ++i) {
        std::cout << "x[" << i << "]: ";
        std::cin >> x[i];
        std::cout << "y[" << i << "]: ";
        std::cin >> y[i];
    }
```

```
    // Nilai x yang ingin diinterpolasi
    double xInterp;
    std::cout << "Masukkan nilai x yang ingin diinterpolasi: ";
    std::cin >> xInterp;

    // Menghitung nilai interpolasi
    double yInterp = lagrangeInterpolation(x, y, xInterp);

    std::cout << "Nilai interpolasi pada x = " << xInterp << " adalah y =
" << yInterp << std::endl;

    return 0;
}
```

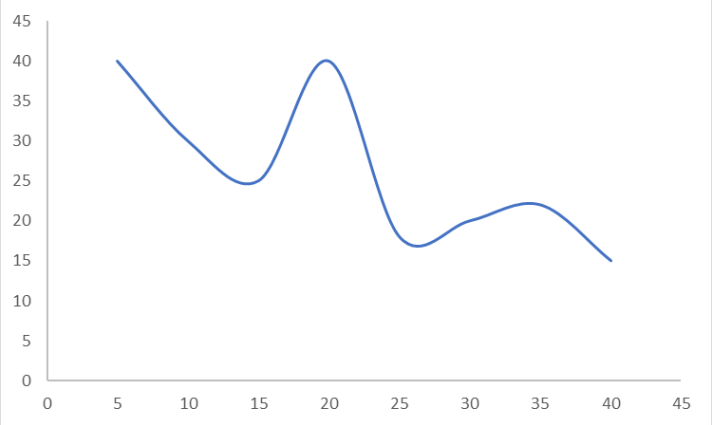
Alur kode;

1. Memasukkan input jumlah titik data.
2. Memasukkan nilai x dan y untuk setiap titik data.
3. Meminta pengguna memasukkan nilai xInterp yang akan diinterpolasi.
4. Menghitung nilai interpolasi pada xInterp menggunakan lagrangeInterpolation.
5. Menampilkan hasil interpolasi.

Output



Grafik Interpolasi



Metode Polinom Newton

Source code

```
#include <iostream>
#include <vector>

// Fungsi untuk menghitung koefisien Newton
std::vector<double> newtonCoefficients(const std::vector<double>& x, const
std::vector<double>& y) {
    int n = x.size();
    std::vector<std::vector<double>>                dividedDifferences(n,
std::vector<double>(n));
    std::vector<double> coefficients(n);

    // Inisialisasi divided differences dengan nilai y
    for (int i = 0; i < n; ++i) {
        dividedDifferences[i][0] = y[i];
    }

    // Hitung divided differences
    for (int j = 1; j < n; ++j) {
        for (int i = 0; i < n - j; ++i) {
            dividedDifferences[i][j] = (dividedDifferences[i + 1][j - 1] -
dividedDifferences[i][j - 1]) / (x[i + j] - x[i]);
        }
    }

    // Ekstrak koefisien
    for (int i = 0; i < n; ++i) {
        coefficients[i] = dividedDifferences[0][i];
    }

    return coefficients;
}

// Fungsi untuk menghitung nilai interpolasi menggunakan polinom Newton
double    newtonInterpolation(const    std::vector<double>&    x,    const
std::vector<double>& coefficients, double xInterp) {
    double result = coefficients[0];
    double term = 1.0;

    for (int i = 1; i < coefficients.size(); ++i) {
        term *= (xInterp - x[i - 1]);
        result += coefficients[i] * term;
    }

    return result;
}

int main() {
    // Data input dari pengguna
    int numPoints;
    std::cout<<"~~~~~~~~~~~~~~~~~~";
```

```
    std::cout<<"| Program Interpolasi Newton |";
    std::cout<<"~~~~~~~~~~~~~~~~~~~~";
    std::cout << "\nMasukkan jumlah titik data: ";
    std::cin >> numPoints;

    std::vector<double> x(numPoints), y(numPoints);

    std::cout << "Masukkan nilai x dan y:\n";
    for (int i = 0; i < numPoints; ++i) {
        std::cout << "x[" << i << "]: ";
        std::cin >> x[i];
        std::cout << "y[" << i << "]: ";
        std::cin >> y[i];
    }

    // Hitung koefisien Newton
    std::vector<double> coefficients = newtonCoefficients(x, y);

    // Nilai x yang ingin diinterpolasi
    double xInterp;
    std::cout << "Masukkan nilai x yang ingin diinterpolasi: ";
    std::cin >> xInterp;

    // Menghitung nilai interpolasi
    double yInterp = newtonInterpolation(x, coefficients, xInterp);

    std::cout << "Nilai interpolasi pada x = " << xInterp << " adalah y =
" << yInterp << std::endl;

    return 0;
}
```

Alur kode;

1. Memasukkan input jumlah titik data.
2. Memasukkan nilai x dan y untuk setiap titik data.
3. Menghitung koefisien Newton menggunakan newtonCoefficients.
4. Meminta pengguna memasukkan nilai xInterp yang akan diinterpolasi.
5. Menghitung nilai interpolasi pada xInterp menggunakan newtonInterpolation.
6. Menampilkan hasil interpolasi.

Output

```
~~~~~~~~~~~~~~~~~~~~| Program Interpolasi Newton |~~~~~~~~~~~~~~~~~~~~
Masukkan jumlah titik data: 8
Masukkan nilai x dan y:
x[0]: 5
y[0]: 40
x[1]: 10
y[1]: 30
x[2]: 15
y[2]: 25
x[3]: 20
y[3]: 40
x[4]: 25
y[4]: 18
x[5]: 30
y[5]: 20
x[6]: 35
y[6]: 22
x[7]: 40
y[7]: 15
Masukkan nilai x yang ingin diinterpolasi: 45
Nilai interpolasi pada x = 45 adalah y = 1104

Process returned 0 (0x0)   execution time : 37.441 s
Press any key to continue.
```

Grafik Interpolasi