

**MINISTERUL EDUCAȚIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Departamentul Informatică şi Ingineria Sistemelor**

**st., gr. Ia-231, Chistol Maxim**

**RAPORT**

**pentru lucrarea de laborator nr.5**

**la cursul *“Metode numerice”***

Verificat:

**Vasile Moraru,** *conf. univ. dr.*

Departamentul Informatică şi IS,

Facultatea CIM, UTM

**Chișinău 2024**

**Tema:**

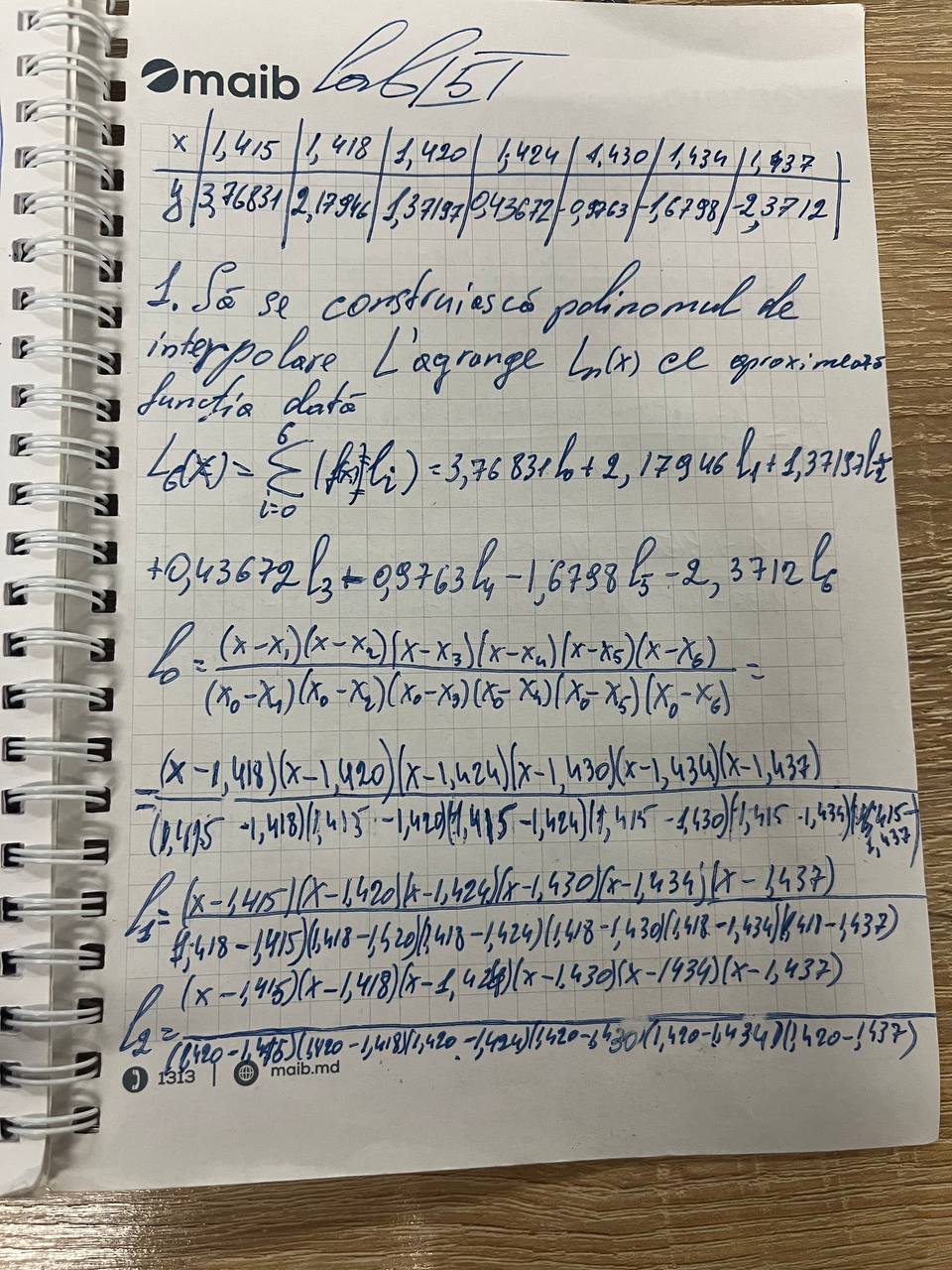
Interpolarea funcțiilor cu ajutorul polinomului L’agrange

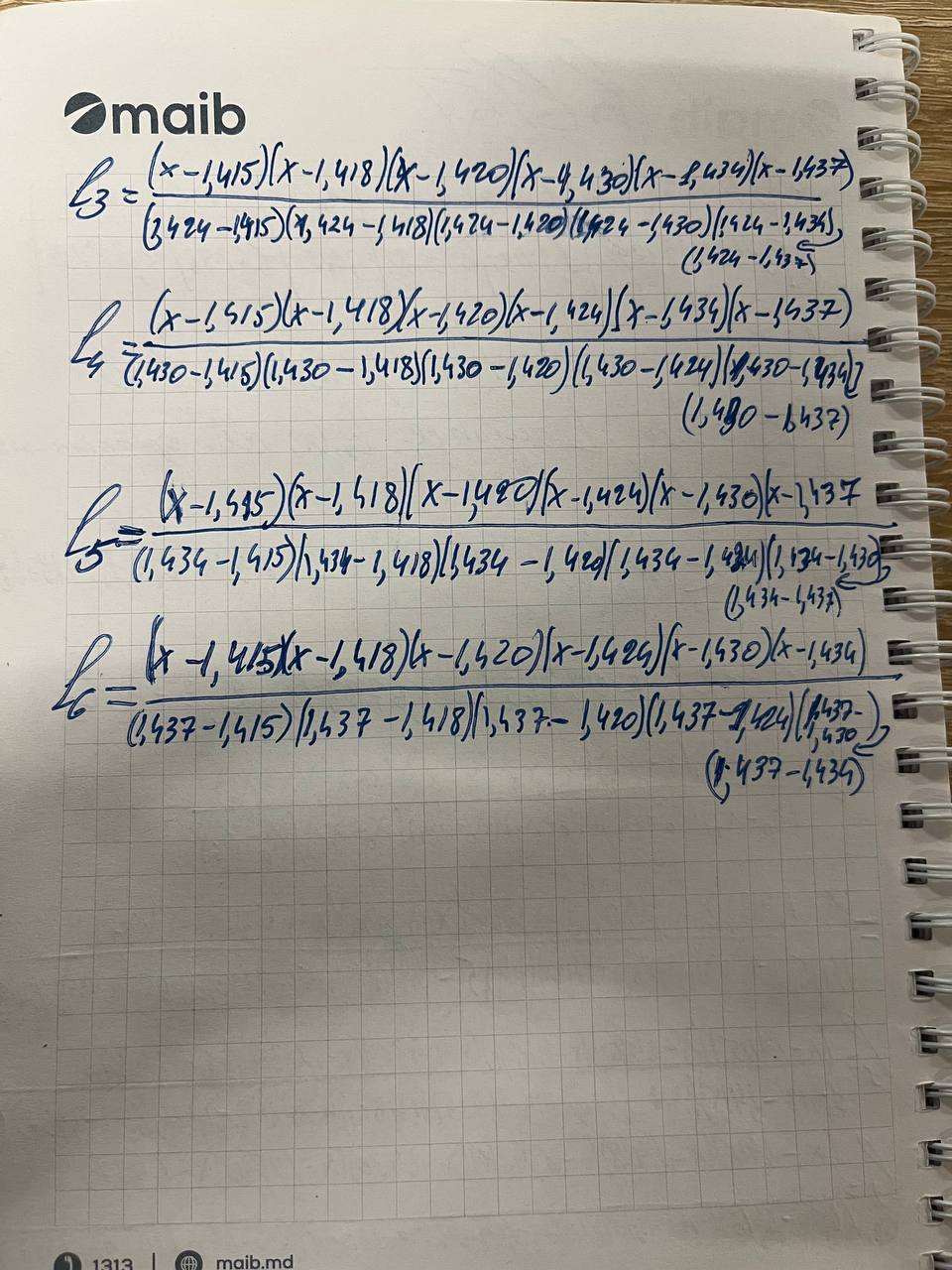
**Sarcina și rezolvarea acesteia:**

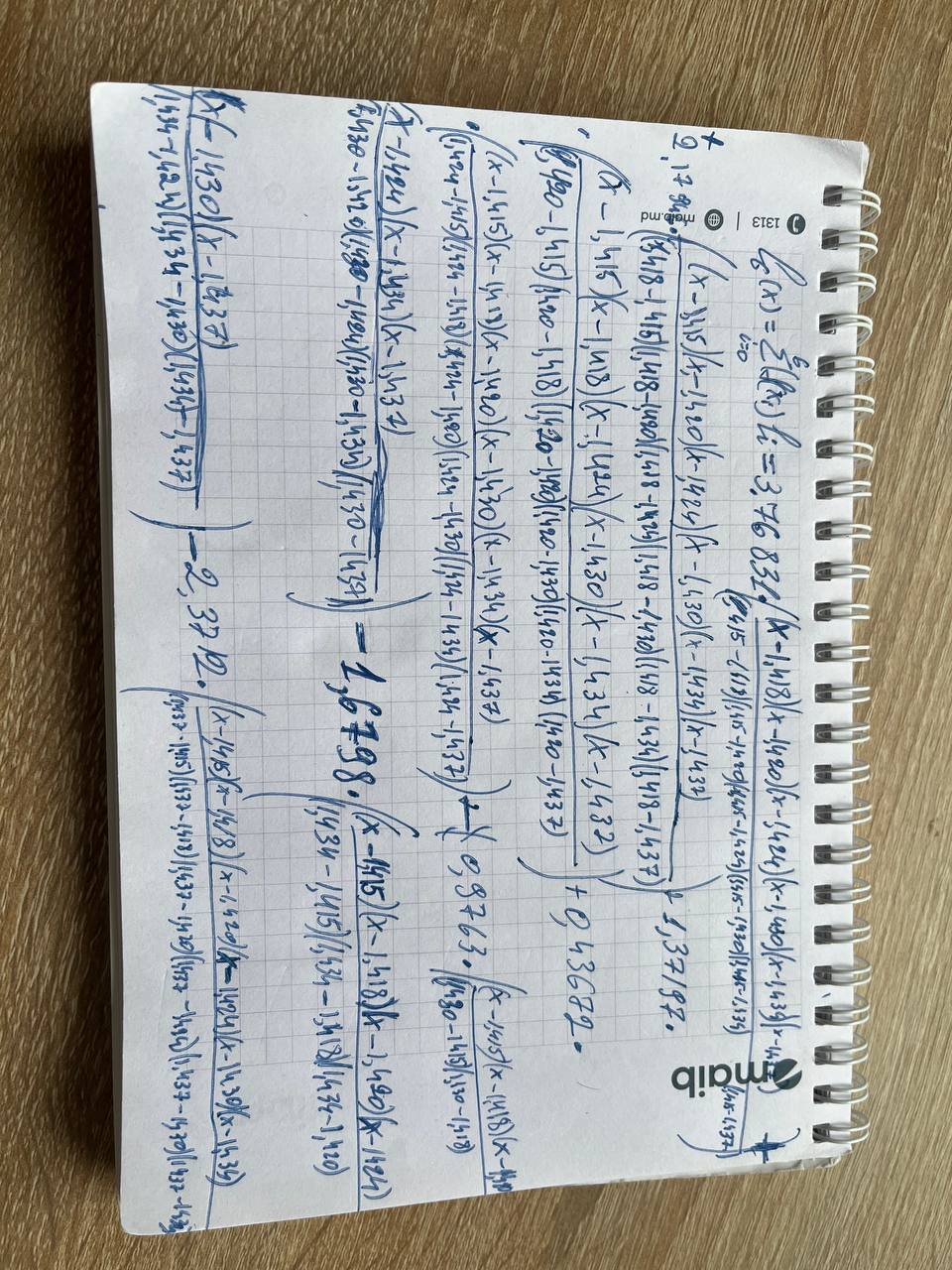
Pentru funcţia se cunosc valorile în nodurile distincte adică .

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| x | 1.415 | 1.418 | 1.420 | 1.424 | 1.430 | 1.434 | 1.437 |
|  | 3.76831 | 2.17946 | 1.37197 | 0.43672 | -0.9763 | -1.6798 | -2.3712 |

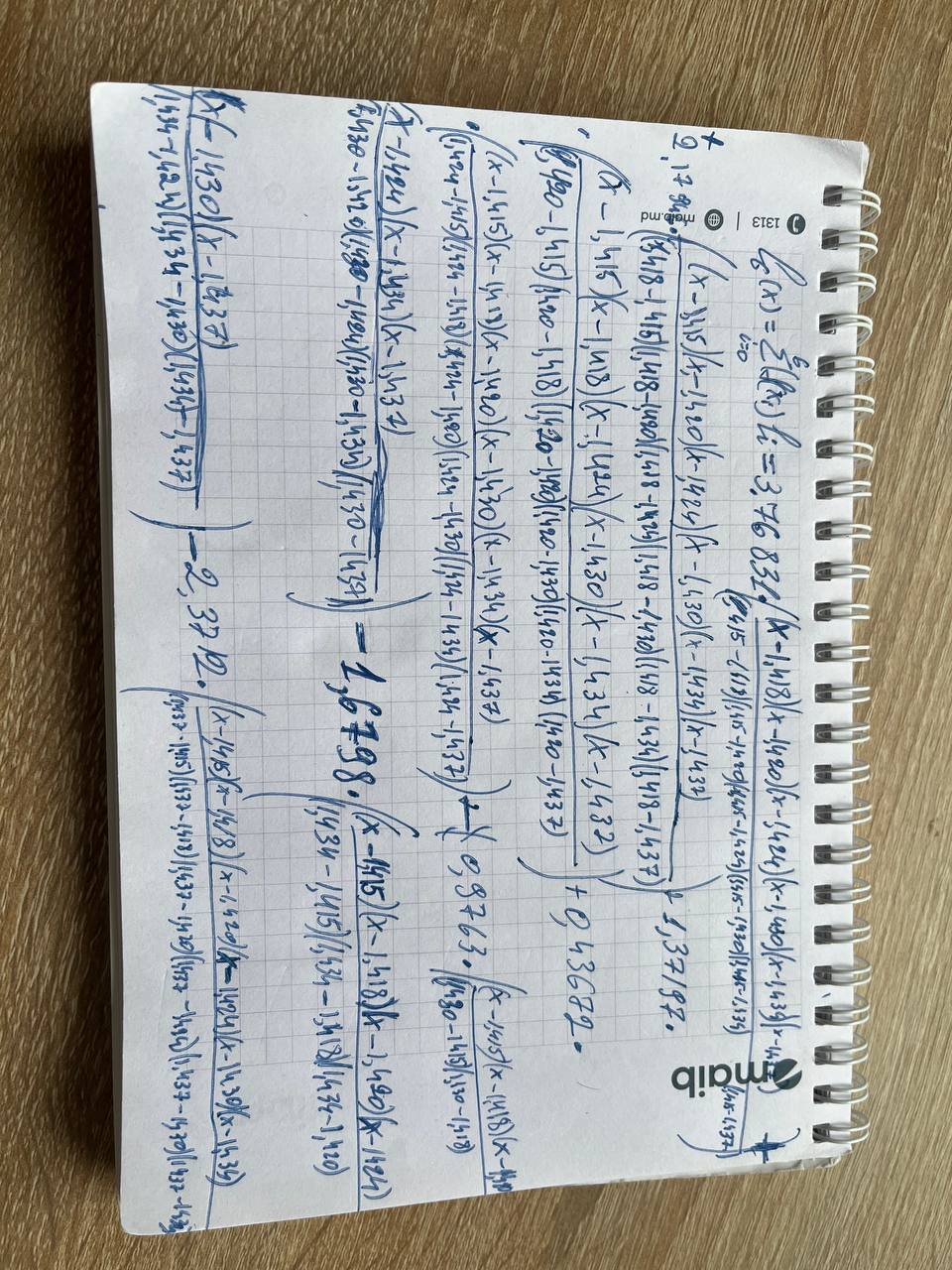
1. Să se construiască polinomul de interpolare L’agrange ce aproximează funcţia dată.







1. Să se calculeze valoarea funcţiei într-un punct utilizând polinomul de interpolare L’agrange



După înlocuirea lui x cu ,vom obține rezultatul -9202412428.710209

1. Să se compare și să se explice rezultatele obținute în sarcinile 2 & 3.

|  |  |
| --- | --- |
| **Sarcina** | **Rezultatul obținut** |
| 2 |  |

Codul pentru sarcina 1:

#include <iostream>

using namespace std;

void generateLagrangePolynomial(double x[], double y[], int n) {

cout << "Polinomul de interpolare Lagrange este:\nL\_n(x) = ";

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

// Afișăm coeficientul curent cu precizie de 4 zecimale

cout << "(";

cout.precision(5);

cout << fixed << y[i];

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

if (i != j) {

// Termenul (x - x[j])

cout << " \* (x - ";

cout.precision(3);

cout << fixed << x[j] << ")";

}

}

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

if (i != j) {

// Termenul / (x[i] - x[j])

cout << " / (";

cout.precision(3);

cout << fixed << x[i] << " - " << x[j] << ")";

}

}

if (i < n - 1) {

cout << " + ";

}

}

cout << endl;

}

int main() {

// Valorile date

double x[] = {1.415, 1.418, 1.420, 1.424, 1.430, 1.434, 1.437};

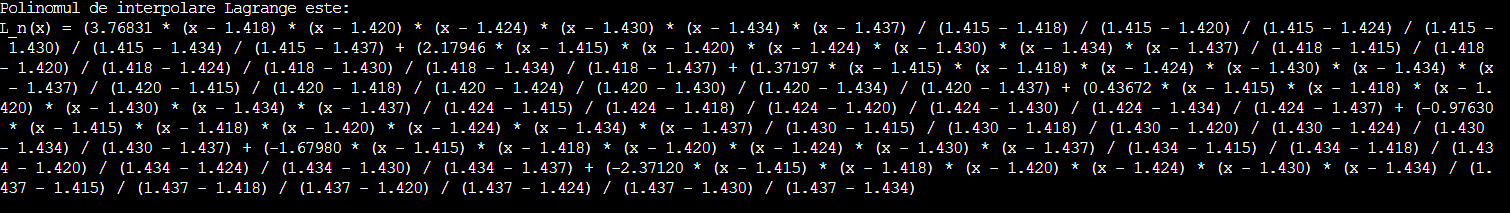
double y[] = {3.76831, 2.17946,1.37197, 0.43672, -0.9763, -1.6798, -2.371};

int n = sizeof(x) / sizeof(x[0]); // Numărul de puncte

// Generăm polinomul

generateLagrangePolynomial(x, y, n);

return 0;

}

**Figura 1.** Outputul pentru sarcina 1

#include <iostream>

using namespace std;

// Funcția pentru calcularea polinomului de interpolare Lagrange

double lagrangeInterpolation(double x[], double y[], int n, double alpha) {

double result = 0.0;

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

double term = y[i]; // Inițial termenul este valoarea y[i]

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

if (i != j) {

term \*= (alpha - x[j]) / (x[i] - x[j]);

}

}

result += term; // Adăugăm termenul calculat la rezultat

}

return result;

}

int main() {

// Valorile date

double x[] = {1.415, 1.418, 1.420, 1.424, 1.430, 1.434, 1.437};

double y[] = {3.76831, 2.17946, 1.37197, 0.43672, -0.9763, -1.6798, -2.3712};

int n = sizeof(x) / sizeof(x[0]); // Numărul de puncte

double alpha = 0.997; // Valoarea pentru care calculăm L\_n(alpha)

// Calcularea rezultatului

double result = lagrangeInterpolation(x, y, n, alpha);

// Afișarea rezultatului

cout << "Valoarea aproximată a funcției la x = " << fixed;

cout.precision(3);

cout << alpha << " este ";

cout.precision(6);

cout << result << endl;

return 0;

}

Codul pentru sarcina 2:



**Figura 2.** Outputul pentru sarcina 2

**Concluzii:**

Laboratorul dedicat interpolării funcțiilor utilizând polinomul Lagrange a evidențiat importanța metodelor numerice în aproximarea funcțiilor pe baza unui număr finit de puncte cunoscute. Polinomul Lagrange oferă o metodă practică și eficientă pentru a construi o funcție polinomială care trece prin toate punctele specificate, având aplicații extinse în domenii precum analiza datelor, modelarea matematică, inginerie și științe.

Pe parcursul activităților, au fost identificate mai multe concluzii esențiale:

1. **Utilitatea interpolării**: Interpolarea este esențială atunci când sunt disponibile doar câteva valori discrete ale unei funcții, dar se dorește o aproximare continuă pe întreg domeniul. Metoda Lagrange garantează că funcția interpolată trece prin toate punctele cunoscute.
2. **Simplitate și accesibilitate**: Polinomul Lagrange poate fi determinat direct, fără necesitatea rezolvării unui sistem de ecuații liniare, ceea ce îl face potrivit pentru seturi mici de puncte.
3. **Limitări**:
   * Creșterea gradului polinomului odată cu numărul de puncte poate genera oscilații nedorite între acestea (Fenomenul Runge), mai ales în cazul punctelor echidistante.
   * Pentru seturi mari de puncte, metode alternative, precum spline-urile cubice, sunt mai eficiente și stabile.
4. **Implementare practică**: Codificarea metodei este simplă în limbaje precum Python sau C++, dar necesită atenție pentru a reduce erorile numerice asociate calculelor în virgulă mobilă.
5. **Aplicații practice**: Metoda Lagrange este utilizată frecvent pentru aproximarea valorilor unei funcții necunoscute între punctele date sau pentru analiza datelor experimentale.

Pentru a maximiza eficiența metodei Lagrange, este recomandată utilizarea nodurilor Chebyshev pentru a diminua erorile de interpolare. Totodată, pentru intervale mari sau cazuri mai complexe, spline-urile cubice reprezintă o alternativă mai precisă și robustă.

Astfel, interpolarea prin polinomul Lagrange constituie o tehnică fundamentală în analiza numerică, oferind o bază solidă pentru învățarea metodelor mai avansate de aproximație și prelucrare a datelor.