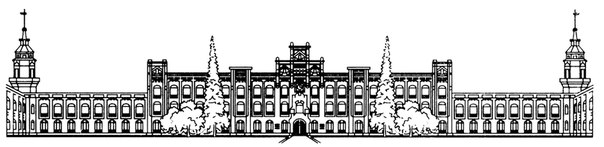
Національний технічний університет України «КПІ ім. Ігоря Сікорського»



Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

Лабораторна робота № 6

з дисципліни «Спеціальні розділи математики-2.  
Чисельні методи»

на тему

***"***[***Інтерполяційні поліноми***](https://do.ipo.kpi.ua/mod/assign/view.php?id=70308)***"***

Виконав:

студент групи ІС-31

Коваль Богдан

Викладач:

доц. Рибачук Л.В.

Київ – 2024

Зміст

Постановку задачі у вигляді заданої функції (із графіком) та значень точок даних у вузлах інтерполяції;

• вигляд поліному **Лагранжа** або Ньютона (за варіантом);

• порівняльний графік функції та інтерполяційного поліному;

• сплайни (коефіцієнти сплайнових інтерполяційних поліномів);

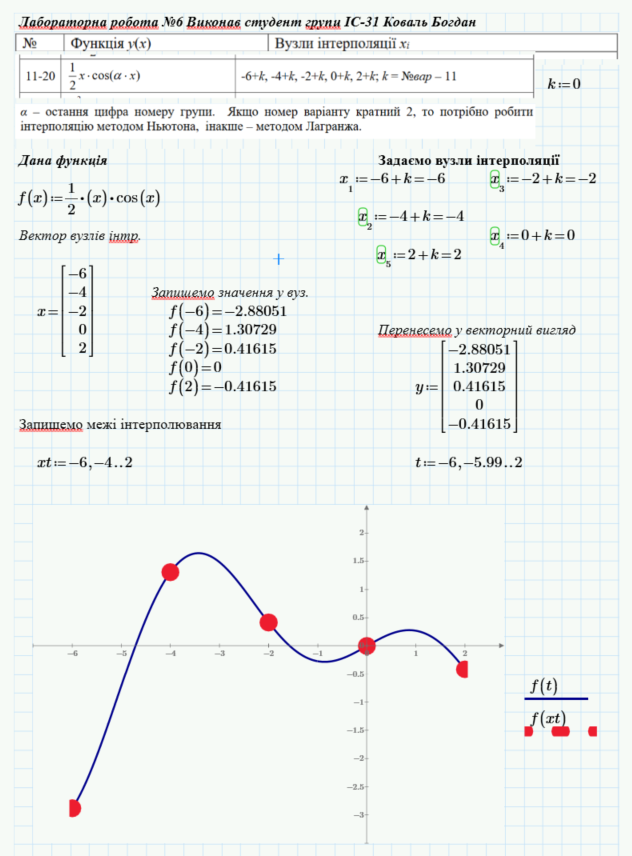
• порівняльний графік функції та сплайн-інтерполяції;

• розв’язок інтерполяції сплайнами у Mathcad (із наведенням порівняльних графіків);

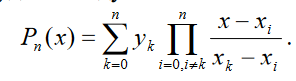
• лістинг програми.

Загальний висновок

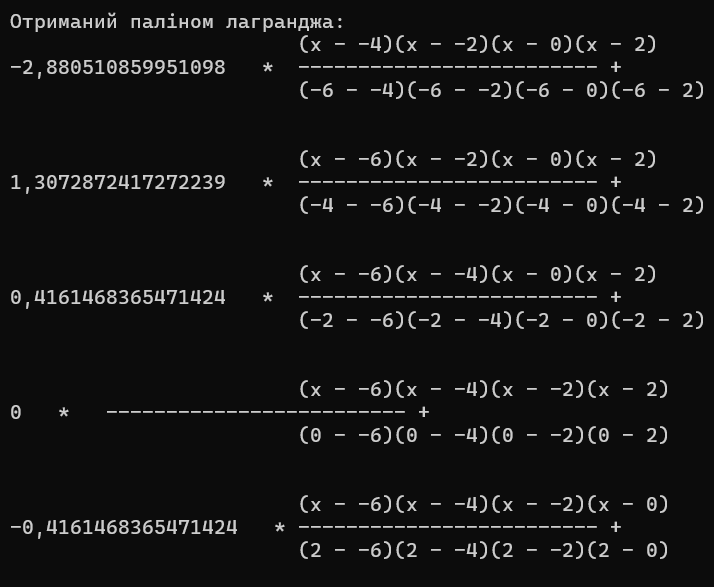
1. Постановку задачі у вигляді заданої функції (із графіком) та значень точок даних у вузлах інтерполяції



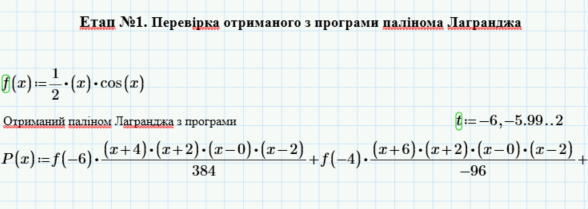
*2. Вигляд поліному* ***Лагранжа*** *(за варіантом);*

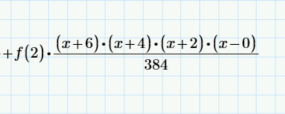
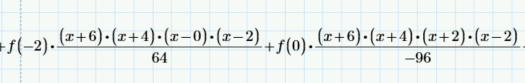


***Результат роботи програми:***

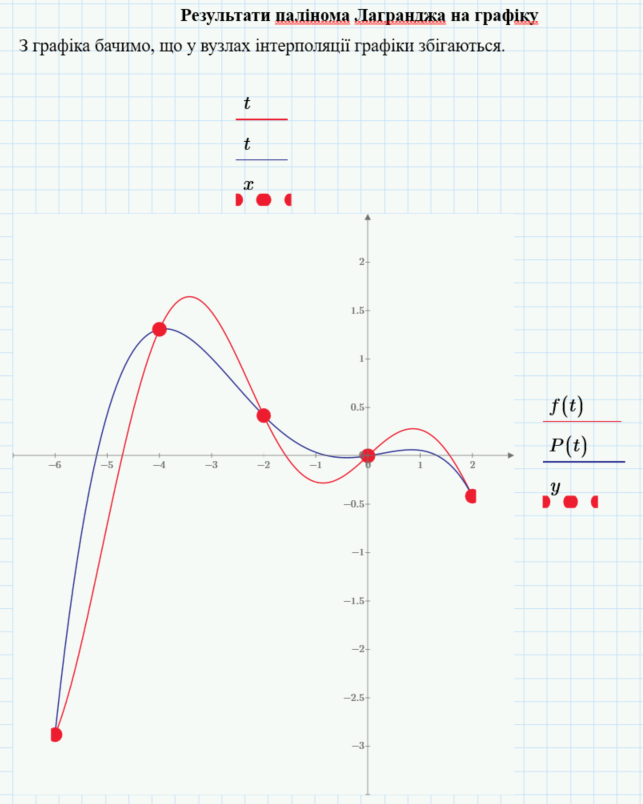
**

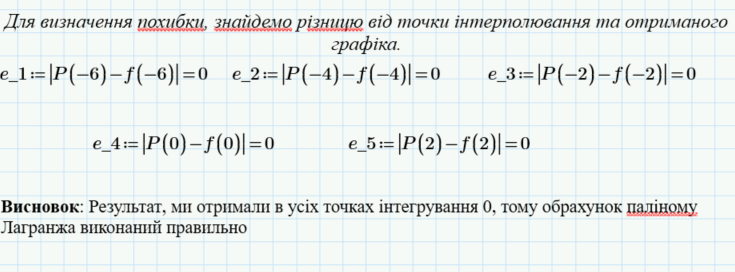
***Перенесемо значення отримані з програми.***

**

**

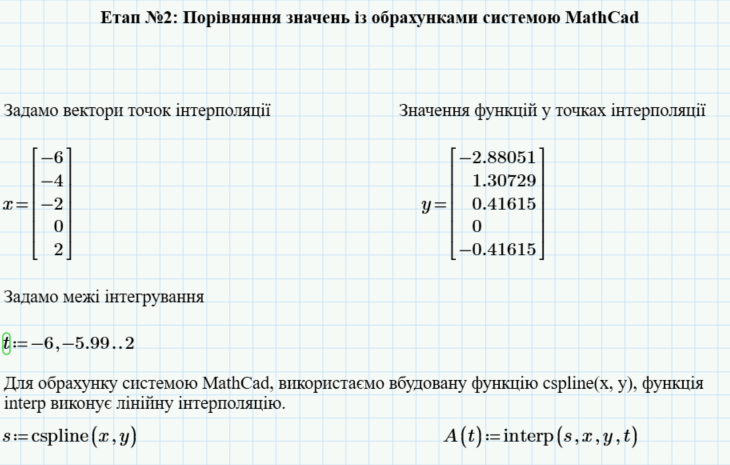
3. Порівняльний графік функції та інтерполяційного поліному;

**

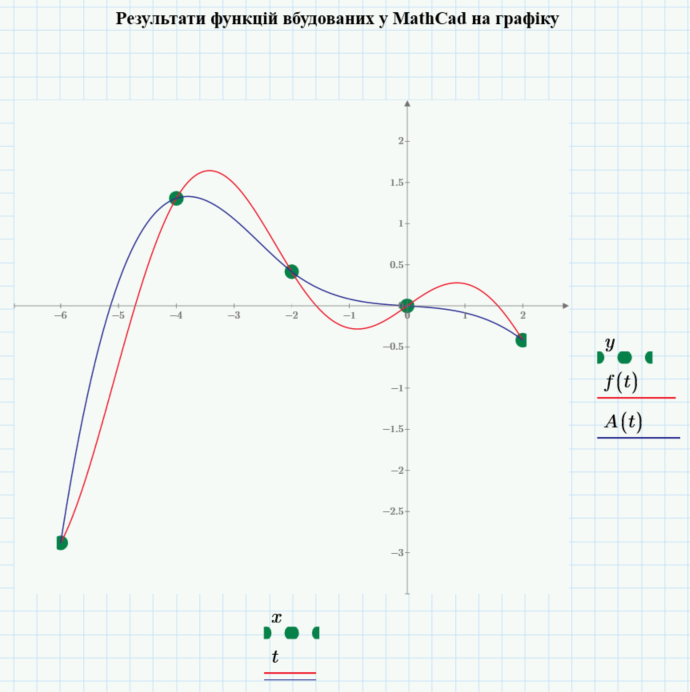
**

4. Сплайни (коефіцієнти сплайнових інтерполяційних поліномів);

№1 Засобами MathCad

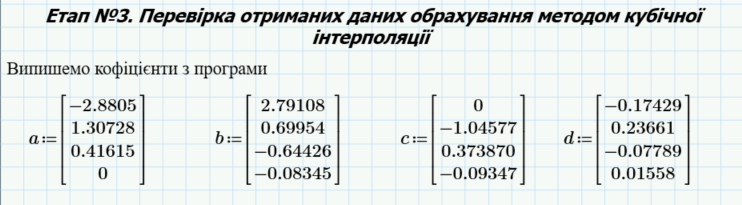
**

5. Порівняльний графік функції та сплайн-інтерполяції;

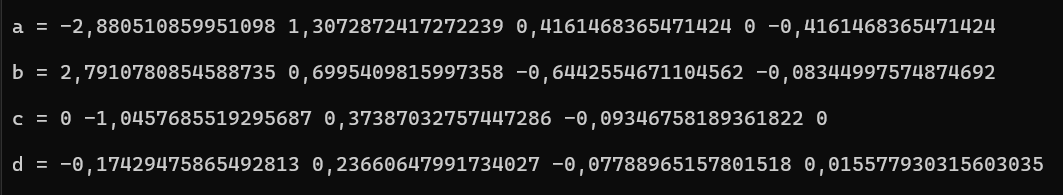
**

5. Розв’язок інтерполяції сплайнами у Mathcad (із наведенням порівняльних графіків);

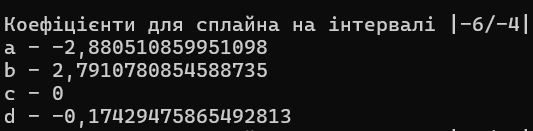
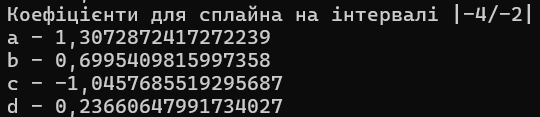
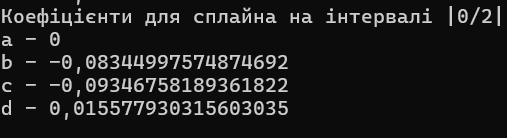
№1 Беремо дані з програми

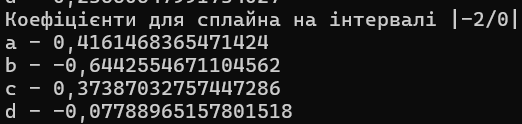
**

*Дані отримані з програми:*

**

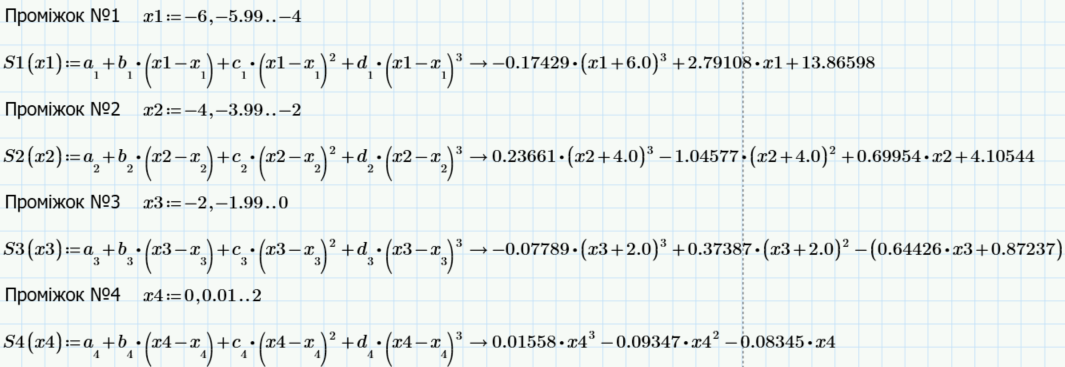
*Запис по інтервалах:*

**

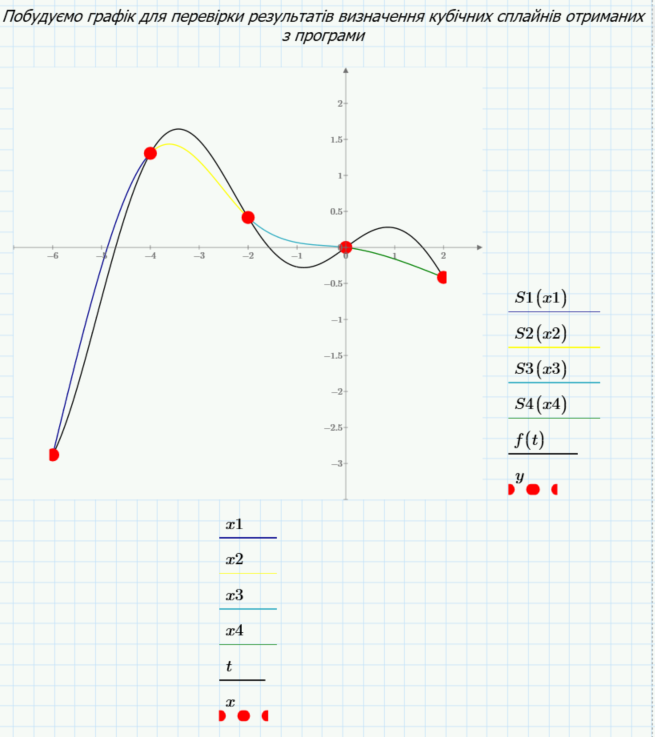
**

Підставимо ці значення у загальну формулу

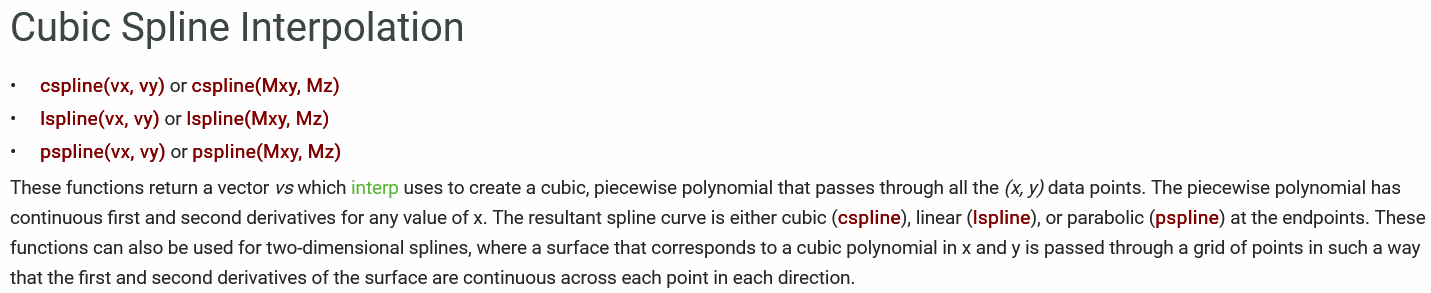


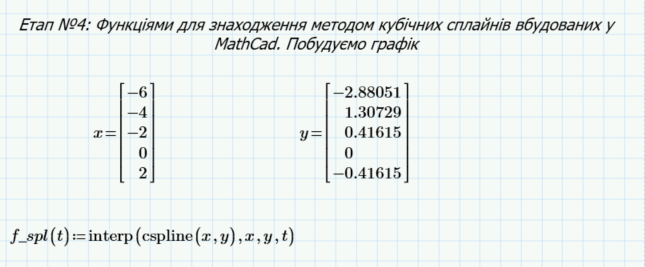
**

*Будуємо графік:*

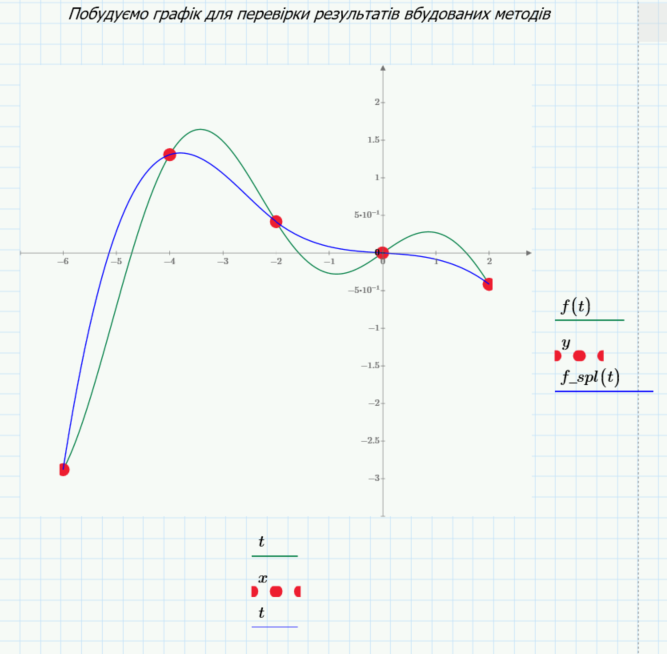
**

*№2 Дані задаються Функцією для знаходження методом кубічних сплайнів вбудованих у MathCad. Побудуємо графік*



**

*Будуємо графік*

**

*6. Лістинг програми*

*Program.cs*

using System;

using System.Linq;

namespace Lab\_6

{

internal class Program

{

static void ViewInfo(double[] x\_i, double[] y\_i)

{

Console.WriteLine("Значення функцiї в узлах iнтерполяцiї");

for (int i = 0; i < x\_i.Length; i++)

{

Console.WriteLine($"X[{i}]: {x\_i[i]}\nY[{i}]: {y\_i[i]}");

PrintSep();

}

}

static double calc\_fun(double argument)

{

double result\_fun = 0.5 \* argument \* Math.Cos(argument);

return result\_fun;

}

static void PrintSep()

{

Console.WriteLine(string.Concat(Enumerable.Repeat("-", 50)));

}

static void Main(string[] args)

{

PrintSep();

Console.WriteLine("\tЛабораторна робота #6");

Console.WriteLine("Виконав студент групи IC-31 Коваль Богдан");

PrintSep();

double[] x\_i = new double[] {-6, -4, -2, 0, 2};

int x\_len = x\_i.Length;

double[] y\_i = new double[x\_len];

//Заповнюємо значення функції в точках x\_i

for (int i = 0; i < x\_len; i++)

{

y\_i[i] = calc\_fun(x\_i[i]);

}

//Перегляд значень у вузлах iнтерполяцiї

ViewInfo(x\_i, y\_i);

//Алгоритм Лагранджа

LagrangeMethod lagra = new LagrangeMethod(x\_i, y\_i);

Console.WriteLine(lagra.LagrangeView());

//Кубічна сплайн-інтерполяція

SplineInterpolation spline = new SplineInterpolation(x\_i, y\_i);

Console.WriteLine(spline.processingSplineCoef());

}

}

}

LagrangeMethod.cs

using System;

using System;

using System.Linq;

namespace Lab\_6

{

internal class LagrangeMethod

{

public double[] \_x\_i { get; private set; }

public double[] \_y\_i { get; private set; }

public LagrangeMethod(double[] x\_i, double[] y\_i)

{

\_x\_i = x\_i;

\_y\_i = y\_i;

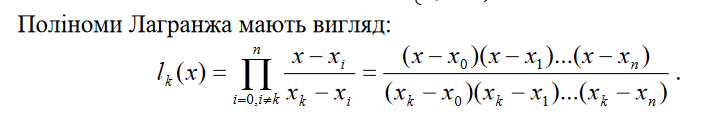
}

public string LagrangeView()

{

int n = \_x\_i.Length;

string resLagrange = "Отриманий палiном лагранджа: ";



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

{

string denomLagrange = ""; //Записати знаменник

string numeratorLagrange = ""; //Записа чисельника

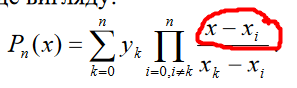
for (int j = 0; j < n; j++)

{

if (i != j)

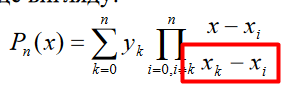
{

string sign = \_x\_i[j] <= 0 ? " - " : " - ";



numeratorLagrange += $"(x{sign}{\_x\_i[j]})";

sign = \_x\_i[j] <= 0 ? " - " : " - ";



denomLagrange += $"({\_x\_i[i]}{sign}{\_x\_i[j]})";

}

}

resLagrange += $"\n\t\t\t{numeratorLagrange}\n{\_y\_i[i]} \*\t" +

$"{string.Concat(Enumerable.Repeat("-", 25))} +" +

$"\n\t\t\t{denomLagrange}\n\n";

}

return resLagrange;

}

}

}

SplineInterpolation.cs

using System;

namespace Lab\_6

{

internal class SplineInterpolation

{

public double[] \_x\_i { get; private set; }

public double[] \_y\_i { get; private set; }

public SplineInterpolation(double[] x\_i, double[] y\_i)

{

\_x\_i = x\_i;

\_y\_i = y\_i;

}

// Метод для обчислення коефiцiєнтiв c за допомогою методу прогону

private double[] tridiagonaAlgo(double[,] matrix, double[] y)

{

int n = y.Length;

// Масиви для коефiцiєнтiв прогону

double[] p = new double[n];

double[] q = new double[n];

// Пряма прогонка

p[1] = matrix[1, 2] / matrix[1, 1];

q[1] = y[1] / matrix[1, 1];

for (int i = 2; i < n - 1; i++)

{

double denom = matrix[i, i] - matrix[i, i - 1] \* p[i - 1];

p[i] = matrix[i, i + 1] / denom;

q[i] = (y[i] - matrix[i, i - 1] \* q[i - 1]) / denom;

}

// Знаходимо зворьотнім ходом

double[] c = new double[n];

c[n - 1] = 0;

for (int i = n - 2; i >= 0; i--)

{

c[i] = q[i] - p[i] \* c[i + 1];

}

return c;

}

public string processingSplineCoef()

{

//Задаєм кiлькiсть наборiв сплайнiв

int n = \_x\_i.Length;

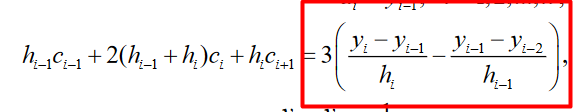
double[] a\_i = new double[n];

//Масив для зберігання довжини частинного відрізка сплайна. (Формула кін - початок)



double[] h\_i = new double[n - 1];

//Допомiжний масив для зберiг. правої частини формули (8)



double[] tmpRight\_i = new double[n - 1];

double[] b\_i = new double[n - 1];

double[] d\_i = new double[n - 1];



//Коефiцiєнтиi a визначають з рiвностi (7)

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

{

a\_i[i] = \_y\_i[i];

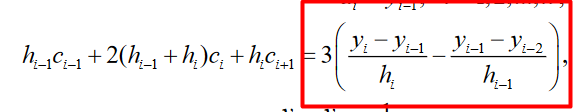
}

//Визначемо довжину частинного вiдрузка h = x(кiнець вiд.) - x(початок вiдрiзка)

for (int i = 1; i < n; i++)

{

h\_i[i - 1] = \_x\_i[i] - \_x\_i[i - 1];

 }

//Заповнюєму змiну. Формула (8) Права частина.

for (int i = 1; i < n - 1; i++)

{

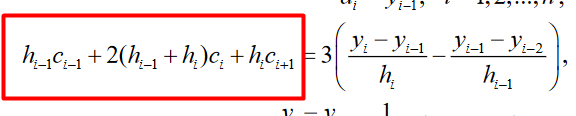
tmpRight\_i[i - 1] = (3 / h\_i[i] \* (a\_i[i + 1] - a\_i[i]) - 3 / h\_i[i - 1] \* (a\_i[i] - a\_i[i - 1]));

}

//Створюємо матрицю для зберiгання даних потрiбних для знаходження c\_i

double[,] matrix = new double[n, n];

double[] y = new double[n];



for (int i = 1; i < n - 1; i++)

{

//Обрахунок доданкiв лiвої частини формули (8)

matrix[i, i - 1] = h\_i[i - 1]; //Найперший iз доданкiв

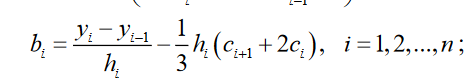
matrix[i, i] = 2 \* (h\_i[i - 1] + h\_i[i]); //Другий iз доданкiв

matrix[i, i + 1] = h\_i[i]; //Третiй iз доданкiв

y[i] = tmpRight\_i[i - 1];

}

double[] c\_i = tridiagonaAlgo(matrix, y);

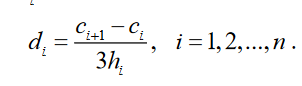


for (int i = 0; i < n - 1; i++)

{

//Знаходимо b\_i за формулою (9)

b\_i[i] = (a\_i[i + 1] - a\_i[i]) / h\_i[i] - h\_i[i] \* (c\_i[i + 1] + 2 \* c\_i[i]) / 3;



//Знаходимо c\_i за формулою (10)

d\_i[i] = (c\_i[i + 1] - c\_i[i]) / (3 \* h\_i[i]);

}

string result = "";

// Виводимо результати для кожного сплайну

for (int i = 0; i < n - 1; i++)

{

result += $"Коефiцiєнти для сплайна на iнтервалi |{\_x\_i[i]}/{\_x\_i[i+1]}|\n" +

$"a - {a\_i[i]}\n" +

$"b - {b\_i[i]}\n" +

$"c - {c\_i[i]}\n" +

$"d - {d\_i[i]}\n";

}

//Вивід в одну строку масив

var str\_a = string.Join(" ", a\_i);

var str\_b = string.Join(" ", b\_i);

var str\_c = string.Join(" ", c\_i);

var str\_d = string.Join(" ", d\_i);

Console.WriteLine($"\na = {str\_a}\n\nb = {str\_b}\n\nc = {str\_c}\n\nd = {str\_d}\n");

return result;

}

}

}

Висновок: В даній лабораторній роботі я дослідив методи інтерполяції, зокрема, інтерполяцію за допомогою поліному Лагранжа та сплайнову інтерполяцію. Мета роботи полягала у створенні інтерполяційного поліному та сплайн-інтерполяції для заданої функції, порівнянні отриманих результатів з оригінальною функцією та реалізації розв'язку за допомогою Mathcad.