Міністерствоосвіти і науки України

Національний університет „Львівська політехніка”

Кафедра ЕОМ



**Лабораторна робота №1**

З дисципліни:”Паралельні та розподіленні обчислення ”

На тему:”ВИКОРИСТАННЯ ФУНКЦІОНАЛЬНОЇ ДЕКОМПОЗИЦІЇ ДЛЯ РОЗВ’ЯЗКУ ОБЧИСЛЮВАЛЬНИХ ЗАДАЧ.”

Виконав: ст.гр. КІ-33

Мельник А.О.

Прийняв: Козак Н.Б.

Львів 2020

Мета роботи: вивчити методи декомпозицій задач. Набути навиків розв’язування задач з використанням функціональної декомпозиції.

**Завдання:** Використовуючи метод функціональної декомпозиції, розробити алгоритм обчислення запропонованого матрично-векторного виразу, який би враховував можливість паралельного виконання і був оптимальним з точки зору часових затрат.

На основі створеного алгоритму написати програму яка дозволяє обчислити вираз та ілюструє проведену декомпозицію.

**Варіант №11**

|  |  |  |  |
| --- | --- | --- | --- |
| 11 | число | | |
| bi=19/(i2+1) для парних і  bi=19 для непарних і | A1(b1+19c1) | A2(B2+C2)  Cij=19/(i+2j)3 |

1.  Аналіз завдання.

Для заданого виразу вхідними даними є:

розмірність матриць – n;

матриці ;

вектори-стовпці .

Ці параметри генеруються випадковим чином (крім розмірності). При чому, елементи всіх матриць та векторів є цілими додатними числами, більшими за нуль.

Вектор-стовпець  та матриця  обраховуються, виходячи з введеної розмірності. При утворенні  враховую, що результатом множення матриці А на вектор-стовпець b є вектор-стовпець, елементи якого будуть раціональними числами(тобто матимуть значущу дробову частину).

При утворенні  враховую, що результатом додавання двох векторів-стовпців є вектор-стовпець. Далі, при множенні цілочисельної додатної матриці А1 на результат додавання, отримую вектор-стовпець з цілочисельними елементами довільного знаку.

При утворенні  враховуємо, що присутні лише операції додавання та множення, а тому вихідний результат завжди буде додатнім і завжди матиме значущу дробову частину.

Таким чином, згідно поставленої задачі, в обчисленні загального виразу приймають участь три різні елементи– два вектори стовпці  та матриця Y3.

Перший множник є рядком. Другий множник загального виразу містить три доданки, перший – вектор-стовпець  помножений транспонований вектор-стовпець y1,(тобто вектор-рядок), результатом є матриця. Другий доданок матриця в квадраті – результат матриця, третій доданок – рядок×стовпець (число) \* матриця. Оскільки, згідно правил матричних обчислень, добуток не є комутативною операцією, всі множення слід виконувати в тій послідовності, яка задана. Результатом множення рядка на стовпець є число, а матриці на матрицю - матриця. Аналогічний аналіз можна застосувати до всіх решти доданків даного виразу. Третій множник є стовпцем. Тобто в нас є добуток рядок \* матриця \* стовпець.

Таким чином, з попереднього випливає, що остаточний результат є числом.

2.  Декомпозиція задачі.

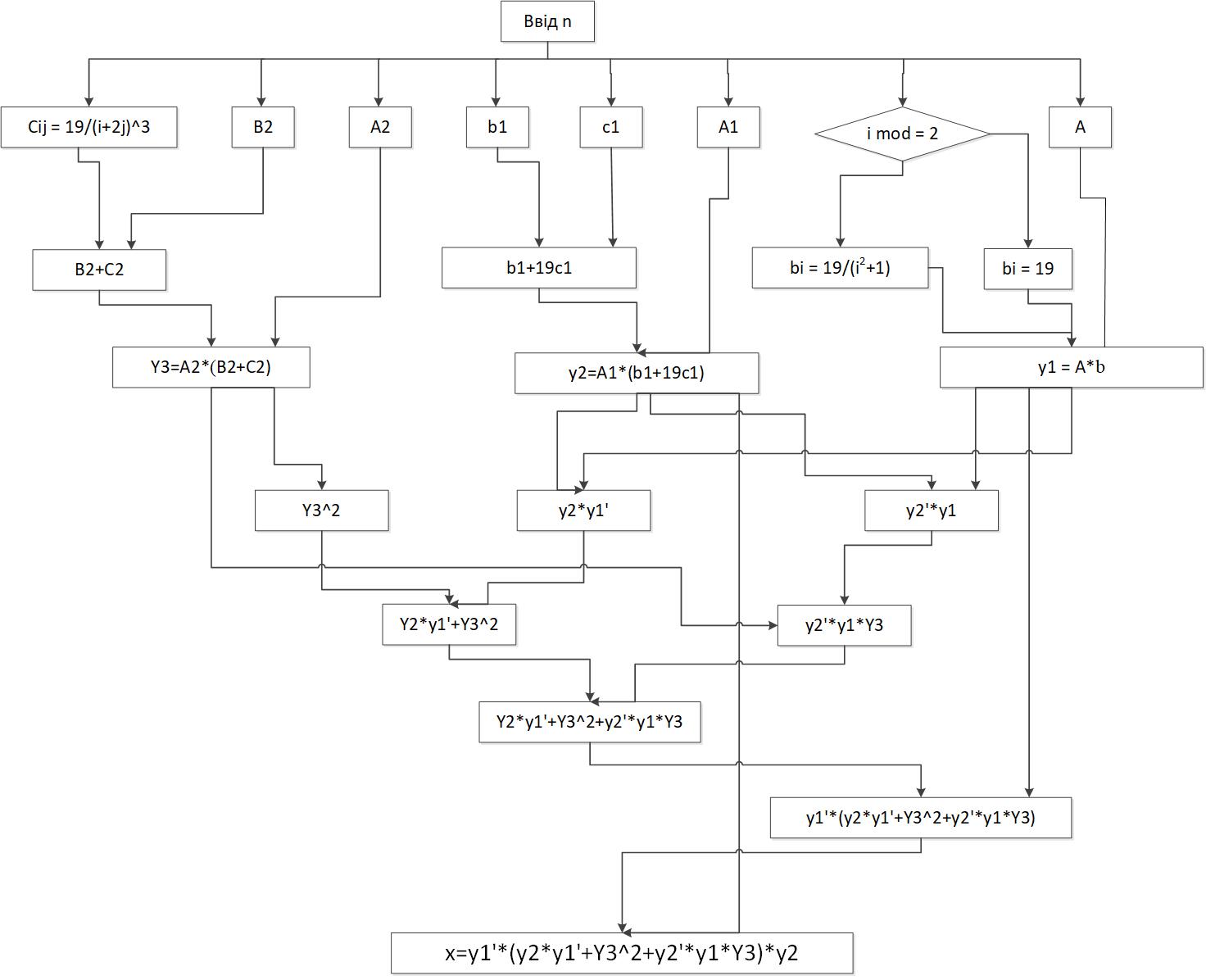
Однозначно, всі обчислення безпосередньо залежать від розмірності даних, тому найперше, забезпечується ввід змінної n, що визначає цю розмірність. Далі, можна паралельно виконувати обчислення значень вектора b та матриці С2, оскільки вони незалежні від інших параметрів. Крім того, на тому ж рівні декомпозиції слід визначати вхідні дані, тобто генерувати випадковим чином матриці  та вектори-стовпці . Наступний рівень декомпозиції – це знаходження елементів виразу. Значення  залежить від введеної матриці А та обрахованого вектора b. Значення  залежить від введеної А1 та суми векторів b1 і c1, тому знайти його можна лише після обчислення (b1 + 19c1). Зауважу, що множення на константу не є окремою операцією, як і транспонування векторів. Аналогічно, знаходимо . Подальша декомпозиція відбувається згідно заданої послідовності операцій та врахування залежностей отриманих на кожному рівні даних. Повна схема декомпозиції обчислення заданого виразу приведена нижче.

3.Об’єднання частин виразу проведено безпосередньо у схемі декомпозиції, оскільки воно однозначно визначається порядком обчислень.

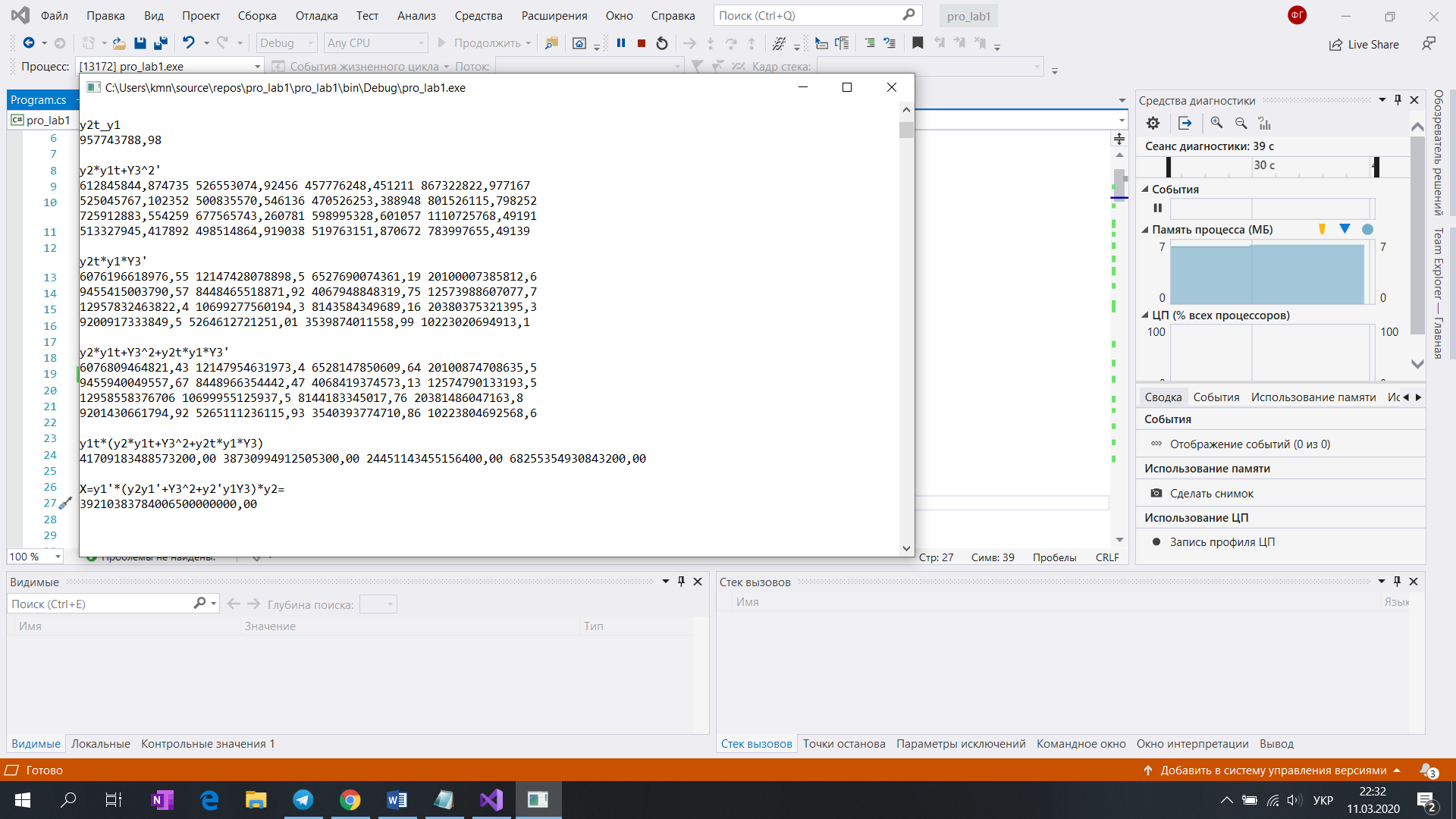
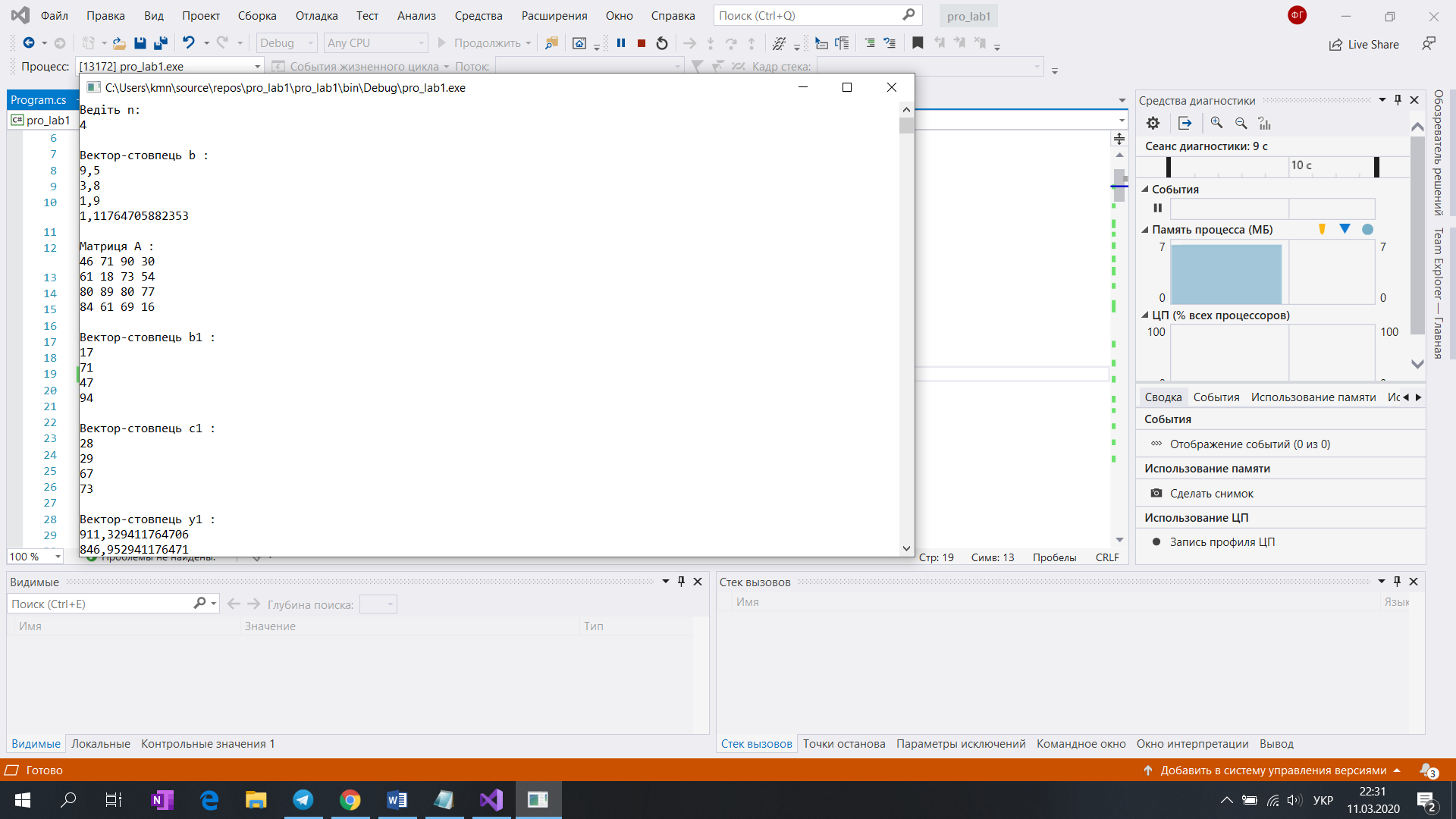
4.   Для написання програми, що ілюструє процес обчислення виразу згідно розробленої схеми декомпозиції, враховано наступні особливості:

Дані, що генеруються випадковим чином є цілими числами, більшими за нуль. Результати проміжних обрахунків будуть містити дробову частину, виняток складає лише елемент y2. Оскільки на окремо взятому рівні декомпозиції обраховуються незалежні частини підвиразів, функції що їх програмно реалізують викликаються псевдо-одночасно.Для того, щоб результат загального виразу був співмірний з вхідними даними, нормуються отримані значення, визначивши тим самим порядок результату.

**Схема декомпозиції обчислення виразу:**



**Результат виконання програми:**



**Код програми:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace PTRO\_lab1

{

class Program

{

static void Main(string[] args)

{

Console.ForegroundColor = ConsoleColor.Black;

Console.BackgroundColor = ConsoleColor.White;

Console.Clear();

Console.WriteLine("Ведiть n: ");

int n = Convert.ToInt32(Console.ReadLine());

double[] b = new double[n];

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

double[,] A1 = new double[n, n];

double[,] C2 = new double[n, n];

double[,] A2 = new double[n, n];

double[,] B2 = new double[n, n];

Random rnd = new Random();

double[] y1 = new double[n];

double[] y2 = new double[n];

double[] b1 = new double[n];

double[] c1 = new double[n];

double[,] Y3 = new double[n, n];

Console.WriteLine();

Console.WriteLine("Вектор-стовпець b : ");

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

{

if ((n%2)==0)

{

b[i] = 19 / (Math.Pow((i + 1), 2)+1);

}

else {

b[i] = 19;

}

Console.WriteLine(b[i]);

}

Console.WriteLine();

Console.WriteLine("Матриця A : ");

A = Arr1(A); Console.WriteLine();

Console.WriteLine("Вектор-стовпець b1 : ");

b1 = Str(b1);

Console.WriteLine();

Console.WriteLine("Вектор-стовпець c1 : ");

c1 = Str(c1);

double[] Str(double[] z)

{

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

{

z[i] = rnd.Next(1, 100);

Console.WriteLine(z[i] + " ");

}

return z;

}

Console.WriteLine();

Console.WriteLine("Вектор-стовпець y1 : ");

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

{

for (int z = 0, m = 0; z < n; z++, m++)

{

y1[i] += b[m] \* A[j, z];

}

Console.WriteLine(y1[i] + " ");

}

Console.WriteLine();

Console.WriteLine("b1 + 19\*c1 :");

double[] l = new double[n];

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

{

l[i] = b1[i] + 19\*c1[i];

Console.WriteLine(l[i] + " ");

}

Console.WriteLine();

Console.WriteLine("Матриця A1 : ");

A1 = Arr1(A1); Console.WriteLine();

Console.WriteLine("Вектор-стовпець y2 : ");

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

{

for (int z = 0, m = 0; z < n; z++, m++)

{

y2[i] += l[m] \* A1[j, z];

}

Console.WriteLine(y2[i] + " ");

}

Console.WriteLine();

Console.WriteLine("Матриця C2 :");

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

{

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

{

double z = Convert.ToDouble(i);

double y = Convert.ToDouble(j);

C2[i, j] = 19 / (Math.Pow(z + 1+2\*(y+1), 3));

Console.Write(C2[i, j].ToString("0.00") + " ");

}

Console.WriteLine();

}

Console.WriteLine();

Console.WriteLine("Матриця A2 :");

A2 = Arr1(A2); Console.WriteLine();

Console.WriteLine("Матриця B2 :");

B2 = Arr1(B2); Console.WriteLine();

double[,] Arr1(double[,] z)

{

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

{

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

{

z[i, j] = rnd.Next(1, 100);

Console.Write(z[i, j] + " ");

}

Console.WriteLine();

}

return z;

}

Console.WriteLine();

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

Console.WriteLine("B2+C2 :");

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

{

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

{

C[i, j] = B2[i, j] + C2[i, j];

Console.Write(C[i, j].ToString("0.00") + " ");

}

Console.WriteLine();

}

Console.WriteLine();

Console.WriteLine("Y3 :");

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

{

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

{

for (int z = 0, m = 0; z < n; z++, m++)

{

Y3[i, j] += A2[i, z] \* C[m, j];

}

Console.Write(Y3[i, j].ToString("0.00") + " ");

}

Console.WriteLine();

}

Console.WriteLine();

Console.WriteLine("y2\_y1t");

double[,] y2\_y1t = new double[n, n];

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

{

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

{

y2\_y1t[i, j] = y2[i] \* y1[j];

Console.Write(y2\_y1t[i, j] + " ");

}

Console.WriteLine();

}

Console.WriteLine();

Console.WriteLine("Y3^2");//

double[,] Y32 = new double[n, n];

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

{

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

{

for (int z = 0, m = 0; z < n; z++, m++)

{

Y32[i, j] += Y3[i, z] \* Y3[m, j];

}

Console.Write(Y32[i, j].ToString("0.00") + " ");//

}

Console.WriteLine();//

}

Console.WriteLine();

Console.WriteLine("y2t\_y1");

double[] y2t\_y1 = new double[n];

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

{

y2t\_y1[0] += y2[i] \* y1[i];

}

Console.WriteLine(y2t\_y1[0].ToString("0.00") + " ");

Console.WriteLine();

Console.WriteLine("y2\*y1t+Y3^2'");

double[,] y1\_y2t\_Y32 = new double[n, n];

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

{

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

{

y1\_y2t\_Y32[i, j] = y2\_y1t[i, j] + Y32[i,j];

Console.Write(y1\_y2t\_Y32[i, j] + " ");

}

Console.WriteLine();

}

Console.WriteLine();

Console.WriteLine("y2t\*y1\*Y3'");

double[,] y2t\_y1\_Y3 = new double[n, n];

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

{

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

{

y2t\_y1\_Y3[i, j] = y2t\_y1[0] \* Y3[i,j];

Console.Write(y2t\_y1\_Y3[i, j] + " ");

}

Console.WriteLine();

}

Console.WriteLine();

Console.WriteLine("y2\*y1t+Y3^2+y2t\*y1\*Y3'");

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

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

{

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

{

yyy[i, j] = y1\_y2t\_Y32[i, j] + y2t\_y1\_Y3[i, j];

Console.Write(yyy[i, j] + " ");

}

Console.WriteLine();

}

Console.WriteLine();

Console.WriteLine("y1t\*(y2\*y1t+Y3^2+y2t\*y1\*Y3)");

double[] y1t\_yyy = new double[n];

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

{

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

{

y1t\_yyy[i] += y1[j] \* yyy[j, i];

}

Console.Write(y1t\_yyy[i].ToString("0.00") + " ");

}

Console.WriteLine();

Console.WriteLine();

Console.WriteLine("X=y1'\*(y2y1'+Y3^2+y2'y1Y3)\*y2=");

double[] x = new double[n];

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

{

x[0] += y1t\_yyy[i] \* y2[i];

}

Console.WriteLine(x[0].ToString("0.00") + " ");

Console.ReadKey();

}

}

}

**Висновок:** виконуючи дану лабораторну роботу я використав паралелізм на рівні підзадач, оскільки передбачається, що кожен блок зі схеми декомпозиції є реалізований у виді функції. Це є середньоблоковий паралелізм. Обмін даними відбувається через використання спільних змінних. Присутня залежність даних між різними рівнями декомпозиції, але в межах одного рівня її немає. Є залежність за керуванням, оскільки послідовність обчислювального процесу наперед однозначно відома. Залежність за ресурсами та вводом/виводом може бути визначена лише у відношенні до певної обчислювальної системи.