Kpi-best

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМ. І.СІКОРСЬКОГО»

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

**Лабораторна робота №1**

з дисципліни «Паралельне програмування»

на тему: «ПОТОКИ В МОВІ JAVA»

Виконав:

студент 3-го курсу

факультету ІОТ

групи ІО-14

Лупащенко А. А.

Перевірив:

доц.

Корочкін О. В.

Київ 2023

**Варіант**

1.26 e = ( (A + SORT(B)) \* ( C\*(MA\*MD) + SORT(Е)) )

2.01 MF = MG + MH\*(MK\*ML)

3.26 s = MAX(V\*MO + P\*(MT\*MS) + R)

**Код програми:**

// Дисципліна: Параленльне програмування

// ЛР 1 "Потоки в мові java"

// F1 = ((A + SORT(B)) \* (C\*(MA\*MD) + SORT(E)))

// F2 = MG + MH\*(MK\*ML)

// F3 = MAX(V\*MO + P\*(MT\*MS) + R)

// Лупащенко Артем Андрійович

// Група ІО-14

// 20.10.2023

import java.util.Arrays;

public class Lab1 {

public static void main(String[] args) {

// Ресурси для функції F1

int[] A = {1, 2, 3};

int[] B = {5, 4, 3};

int[] C = {2, 3, 4};

int[][] MA = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

int[][] MD = {{9, 8, 7}, {6, 5, 4}, {3, 2, 1}};

int[] E = {9, 8, 7};

// Ресурси для функції F2

int[][] MG = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

int[][] MH = {{9, 8, 7}, {6, 5, 4}, {3, 2, 1}};

int[][] MK = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

int[][] ML = {{9, 8, 7}, {6, 5, 4}, {3, 2, 1}};

// Ресурси для функції F3

int[] V = {1, 2, 3};

int[][] MO = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

int[] P = {4, 5, 6};

int[][] MT = {{9, 8, 7}, {6, 5, 4}, {3, 2, 1}};

int[][] MS = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

int[] R = {7, 8, 9};

// Створення групи для потоків

ThreadGroup group1 = new ThreadGroup("GROUP 1");

// Створення потоків

Thread threadF1 = new Thread(group1, new Func1("T1", A, B, C, MA, MD, E), "T1", 1024);

Thread threadF2 = new Thread(group1, new Func2("T2", MG, MH, MK, ML), "T2", 1024);

Thread threadF3 = new Thread(group1, new Func3("T3", V, MO, P, MT, MS, R), "T3", 1024);

// Встановлення пріоритетів потоків

threadF1.setPriority(Thread.MIN\_PRIORITY);

threadF2.setPriority(Thread.MIN\_PRIORITY);

threadF3.setPriority(Thread.MAX\_PRIORITY);

// Запуск потоків

threadF1.start();

threadF2.start();

threadF3.start();

// Очікування завершення виконання потоків

try {

threadF1.join();

threadF2.join();

threadF3.join();

} catch (InterruptedException e) {

// Обробка випадку переривання: виводимо стек виклику в консоль

e.printStackTrace();

}

}

}

class Func1 implements Runnable {

private String name;

private int[] A, B, C, E;

private int[][] MA, MD;

public Func1(String name, int[] A, int[] B, int[] C, int[][] MA, int[][] MD, int[] E) {

this.name = name;

this.A = A;

this.B = B;

this.C = C;

this.MA = MA;

this.MD = MD;

this.E = E;

}

@Override

public void run() {

System.out.println("Початок " + name);

// ((A + SORT(B)) \* (C\*(MA\*MD) + SORT(E)))

// Обчислення C\*(MA\*MD)

int[][] result1 = matrixMultiplication(MA, MD);

int[] result2 = scalarVectorMultiplication(C, result1);

// Обчислення SORT(B), SORT(E)

Arrays.sort(B);

Arrays.sort(E);

// Обчислення фінального результату

int finalResult = 0;

for (int i = 0; i < A.length; i++) {

finalResult += (A[i] + B[i]) \* (result2[i] + E[i]);

}

System.out.println("Result for Func1: " + finalResult);

System.out.println("Кінець " + name);

}

private int[][] matrixMultiplication(int[][] matrix1, int[][] matrix2) {

int rows1 = matrix1.length;

int cols1 = matrix1[0].length;

int rows2 = matrix2.length;

int cols2 = matrix2[0].length;

if (cols1 != rows2) {

throw new IllegalArgumentException("Несумісні розміри матриці");

}

int[][] result = new int[rows1][cols2];

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols2; j++) {

for (int k = 0; k < cols1; k++) {

result[i][j] += matrix1[i][k] \* matrix2[k][j];

}

}

}

return result;

}

private int[] scalarVectorMultiplication(int[] vector, int[][] matrix) {

int rows = matrix.length;

int cols = matrix[0].length;

if (vector.length != rows) {

throw new IllegalArgumentException("Несумісні розміри вектора та матриці");

}

int[] result = new int[cols];

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

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

result[i] += vector[j] \* matrix[j][i];

}

}

return result;

}

}

class Func2 implements Runnable {

private String name;

private int[][] MG, MH, MK, ML;

public Func2(String name, int[][] MG, int[][] MH, int[][] MK, int[][] ML) {

this.name = name;

this.MG = MG;

this.MH = MH;

this.MK = MK;

this.ML = ML;

}

@Override

public void run() {

System.out.println("Початок " + name);

// MG + MH\*(MK\*ML)

// Обчислення MK\*ML

int[][] result1 = matrixMultiplication(MK, ML);

// Обчислення MH\*(MK\*ML)

int[][] result2 = matrixMultiplication(MH, result1);

// Обчислення MG + MH\*(MK\*ML)

int[][] finalResult = matrixAddition(MG, result2);

System.out.println("Result for Func2: " + Arrays.deepToString(finalResult));

System.out.println("Кінець " + name);

}

private int[][] matrixMultiplication(int[][] matrix1, int[][] matrix2) {

int rows1 = matrix1.length;

int cols1 = matrix1[0].length;

int rows2 = matrix2.length;

int cols2 = matrix2[0].length;

if (cols1 != rows2) {

throw new IllegalArgumentException("Несумісні розміри матриці");

}

int[][] result = new int[rows1][cols2];

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols2; j++) {

for (int k = 0; k < cols1; k++) {

result[i][j] += matrix1[i][k] \* matrix2[k][j];

}

}

}

return result;

}

private int[][] matrixAddition(int[][] matrix1, int[][] matrix2) {

int rows = matrix1.length;

int cols = matrix1[0].length;

if (rows != matrix2.length || cols != matrix2[0].length) {

throw new IllegalArgumentException("Несумісні розміри матриці");

}

int[][] result = new int[rows][cols];

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

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

result[i][j] = matrix1[i][j] + matrix2[i][j];

}

}

return result;

}

}

class Func3 implements Runnable {

private String name;

private int[] V, P, R;

private int[][] MO, MT, MS;

public Func3(String name, int[] V, int[][] MO, int[] P, int[][] MT, int[][] MS, int[] R) {

this.name = name;

this.V = V;

this.MO = MO;

this.P = P;

this.MT = MT;

this.MS = MS;

this.R = R;

}

@Override

public void run() {

System.out.println("Початок " + name);

// MAX(V\*MO + P\*(MT\*MS) + R)

// Обчислення V\*MO

int[] result1 = scalarVectorMultiplication(V, MO);

// Обчислення MT\*MS

int[][] result2 = matrixMultiplication(MT, MS);

// Обчислення P\*(MT\*MS)

int[] result3 = scalarVectorMultiplication(P, result2);

// Обчислення V\*MO + P\*(MT\*MS)

int[] result4 = vectorAddition(result1, result3);

// Обчислення V\*MO + P\*(MT\*MS) + R

int[] intermediateResult = vectorAddition(result4, R);

// Обчислення MAX(V\*MO + P\*(MT\*MS) + R)

int finalResult = findMax(intermediateResult);

System.out.println("Result for Func3: " + finalResult);

System.out.println("Кінець " + name);

}

private int[][] matrixMultiplication(int[][] matrix1, int[][] matrix2) {

int rows1 = matrix1.length;

int cols1 = matrix1[0].length;

int rows2 = matrix2.length;

int cols2 = matrix2[0].length;

if (cols1 != rows2) {

throw new IllegalArgumentException("Несумісні розміри матриці");

}

int[][] result = new int[rows1][cols2];

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols2; j++) {

for (int k = 0; k < cols1; k++) {

result[i][j] += matrix1[i][k] \* matrix2[k][j];

}

}

}

return result;

}

private int[] scalarVectorMultiplication(int[] vector, int[][] matrix) {

int rows = matrix.length;

int cols = matrix[0].length;

if (vector.length != rows) {

throw new IllegalArgumentException("Несумісні розміри вектора та матриці");

}

int[] result = new int[cols];

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

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

result[i] += vector[j] \* matrix[j][i];

}

}

return result;

}

private int[] vectorAddition(int[] vector1, int[] vector2) {

int length = vector1.length;

if (length != vector2.length) {

throw new IllegalArgumentException("Несумісні векторні розміри");

}

int[] result = new int[length];

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

result[i] = vector1[i] + vector2[i];

}

return result;

}

private int findMax(int[] vector) {

int max = vector[0];

for (int i = 1; i < vector.length; i++) {

if (vector[i] > max) {

max = vector[i];

}

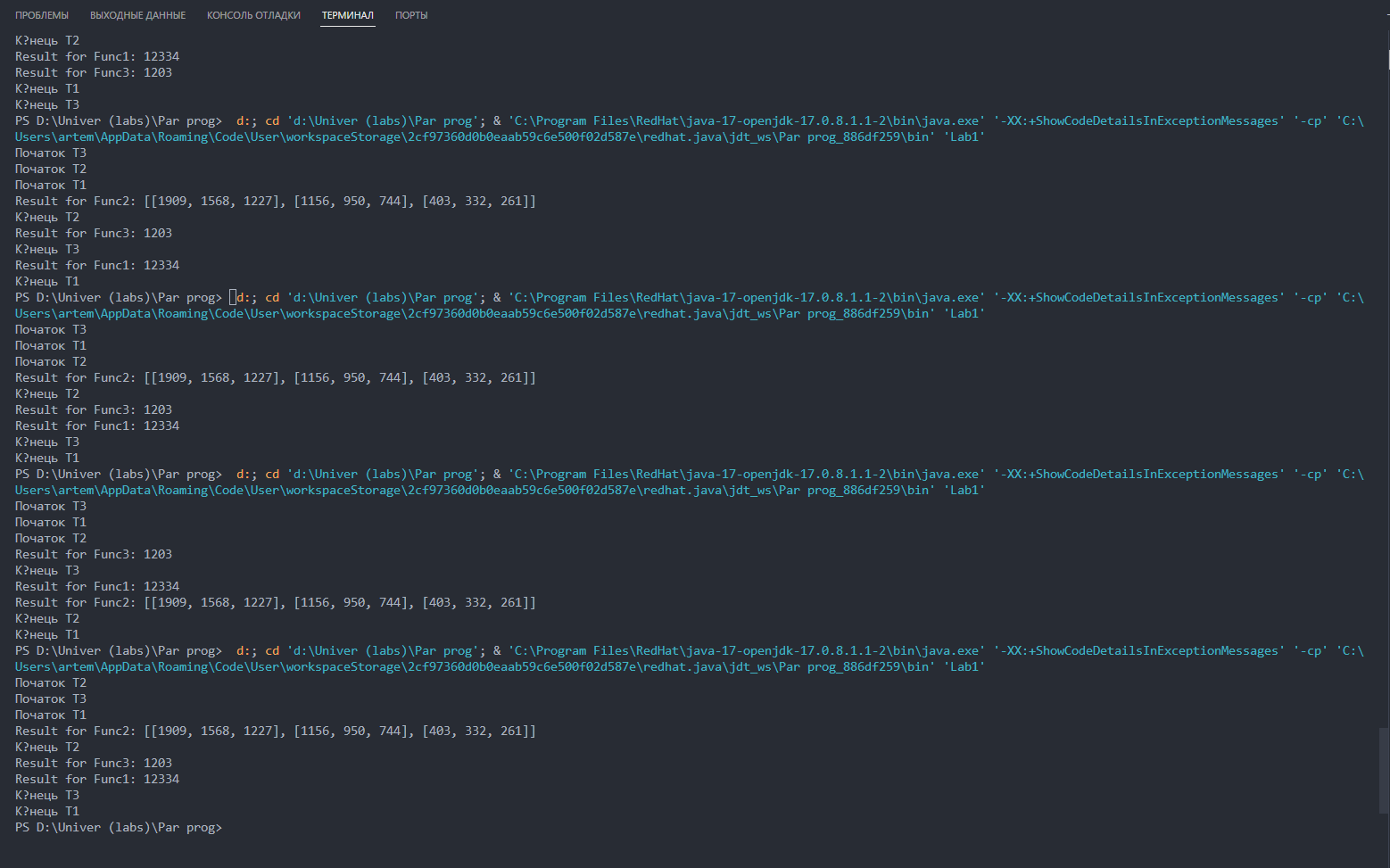
}

return max;

}

}

**Результати виконання програми:**



**Висновок:**

В ході даної лабораторної роботи мені вдалось створити задачі, які виконуються паралельно та вирішують поставленні задачі. З результатів можна побачити залежність пріоритетів задач до черги запуску задач: задача T3 у більшості випадків запускається першою, але високий пріоритет не гарантує цього.