Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

ЛАБОРАТОРНА РОБОТА №2

З ДИСЦИПЛІНИ “ОСНОВИ ПАРАЛЕЛЬНОГО ПРОГРАМУВАННЯ ”

НА ТЕМУ: “ПОТОКИ В МОВІ JAVA”

Виконав:
Студент III курсу ФІОТ
групи ІО-82
Шендріков Євгеній
Номер у списку - 25

Перевірив:
Доцент
Корочкін О. В.

м. Київ – 2020 р.

Мета

Вивчення засобів мови Java для роботи с потоками.

Завдання

Розробити програму яка містить ***паралельні потоки*** (задачі), кожна із яких реалізовує функцію F1, F2, F3 із **Додатку Б** згідно отриманому варіанту.

Вимоги що до створення потоків і завдання дослідження особливості виконання паралельної програми визначені в лабораторної роботі 1.

В потоках використати методи **`sleep()`** та **`join()`**.

Варіант завдання

		F1	*F2*	*F3*	
25	Шендріков Євгеній Олександрович	1.21	2.27	3.30	

1.21 D = SORT(A)+SORT(B)+SORT(C)*(MA*ME)

2.27 MF = (MG*MH)*TRANS(MK)

3.30 S = (MO*MP)*V+t*MR*(O+P)

Аналіз проблем

Проблемою при тестуванні програмної частини, як і в першій лабораторній, стало введення початкових даних з клавіатури, при якому програма поводила себе незвично.

Введення даних можливе з файлу, з клавіатури, генерацією випадкових чисел або заповненням усіх значень одиничками.

В процесі виконання лабораторної роботи було реалізовано останні 3 варіанти введення даних (клавіатура; випадкові значення; усі значення як одинички). Який саме вид даних піде на вхід обирає користувач (даний функціонал прописаний в тілі основної програми).

Продовжуючи аналіз проблеми візьмемо наступну ситуацію: припустимо, що N=3, користувач обрав введення з клавіатури, і завданням є для Func1 ввести всі елементи як 1, для Func2 – як 2, для Func3 – як 3. Якщо з цією проблемою вводу з клавіатури нічого не зробити, то при вводі значень, якщо 3 потоки одночасно запускаються, буде незрозуміло, що куди вводити, буде боротьба за спільні ресурси і все закінчиться хаосом, так само і з виводом.

Описана проблема як для введення, так і для виведення результату є загальною проблемою паралельного програмування. Ця проблема більш відома як *завдання взаємного виключення* і полягає вона в тому, що під час виконання декількох паралельних процесів може виникнути одночасне звернення до спільного ресурсу і таке звернення призводить до конфлікту процесів, як в описаному прикладі.

Для вирішення цієї проблеми необхідно блокувати процес, який звертається до спільного ресурсу, який в цей момент вже використовується іншим процесом. А розблокування процесу відбувається одразу ж після звільнення спільного ресурсу.

Для реалізації цього підходу використовуються спеціальні програмні засоби, такі як семафори, мютекси, критичні секції і т.д. Тому в цій лабораторній була зроблена спроба обробки вхідних даних з консолі за допомогою семафорів, що дозволило виконати необхідне завдання щодо встановлення з клавіатури всіх елементів для Func1 як 1, для Func2 як 2 і для Func3 як 3 (приклад роботи в Додатку А).

Семафори в Java – це конструкція синхронізації потоків, яка контролює доступ до загального ресурсу за допомогою лічильників. Загалом, щоб використовувати семафор, потік, якому потрібен доступ до загального ресурсу, намагається отримати дозвіл. Якщо значення лічильника більше нуля, то потік отримує доступ до ресурсу, при цьому лічильник зменшується на одиницю. Після закінчення роботи з ресурсом потік звільняє семафор, і лічильник збільшується на одиницю. Якщо ж лічильник дорівнює нулю, то потік блокується і чекає, поки не отримає дозвіл від семафора. За замовчуванням всім очікуючим потокам надається дозвіл в невизначеному порядку.

Кожен потік, який хоче використовувати спільний ресурс, повинен спочатку викликати функцію **`acquire()`** перед зверненням до ресурсу для отримання блокування. Коли потік завершує роботу з ресурсом, він повинен викликати **`release()`** для зняття блокування.

При N > 10 можливість вводити дані з клавіатури не передбачена.

При цьому, якщо брати велике N (2000, 3000) і генерувати рандомізовані дані чи «забивати» все одиничками, то в коді поставлена умова, щоб при таких (великих > 10) N в консоль нічого не виводиться, крім того, ясна річ, що задача запустилась і завершилась.

При N < 10 в консоль будуть виводитись дані, але, щоб покращити ввід/вивід, у тілі Func1-3 вказується функція **`sleep()`** (один або декілька разів), під час виконання цього оператора задача блокується і керування передається іншому потоку, яка знаходиться у стані готовності (якщо був обраний ввід не з клавіатури).

Лістинг програми

Lab2.java

```
/*-------------------------------------------------------
  |                    Labwork #2                       |
  -------------------------------------------------------
  |   Author    |    Jack (Yevhenii) Shendrikov         |
  |   Group     |                IO-82                  |
  |   Variant   |                 #25                   |
  |    Date     |             18.09.2020                |
  -------------------------------------------------------
  | Function 1 | D = SORT(A)+SORT(B)+SORT(C)*(MA*ME)     |
  | Function 2 |      MF = (MG*MH)*TRANS(MK)             |
  | Function 3 |     S = (MO*MP)*V+t*MR*(O+P)            |
  -------------------------------------------------------
*/

import java.util.Scanner;
import java.util.concurrent.Semaphore;


public class Lab2 extends Thread{

    @Override
    public void run(){
        setName("Lab2");
        Scanner scanner = new Scanner(System.in);
        int N, val;

        //---------------------------- Input Handler ----------------------------
        // header
        System.out.print("-------------------------------------------------------
\n" +
                "| Function 1 | D = SORT(A)+SORT(B)+SORT(C)*(MA*ME)  |\n" +
                "| Function 2 |      MF = (MG*MH)*TRANS(MK)          |\n" +
                "| Function 3 |     S = (MO*MP)*V+t*MR*(O+P)         |\n" +
                "-------------------------------------------------------\n\n" +
                "!!! Note that if the value of N > 10 -> the result will not be
displayed !!!\n" +
                "!!! If you enter N <= 0 - execution will be terminated !!!\n\n"
+ "Enter N: ");

        // check for int value of N, else N = 3
        if (scanner.hasNextInt()){
            N = scanner.nextInt();
        } else{
            System.out.println("\n!!! You should enter data of type int, N will
be taken as 3 !!!\n");
            N = 3;
        }

        // check for positive value of N
        if (N <= 0) throw new ArithmeticException("Restart the program and enter
N > 0.");

        System.out.print("\nSelect the method according to which the initial
data will be generated:\n" +
                "\t[1] - Create All-Ones Matrices And Vectors.\n"+
                "\t[2] - Create Matrices And Vectors With Random Values.\n" +
                "\t[3] - Enter All Values From The Keyboard.\n" +
                "Your choice [1] / [2] / [3]: ");

       // check for int value of val, else val = 1
        if (scanner.hasNextInt()){
            val = scanner.nextInt();
```

```
        } else{
            System.out.println("\n!!! You should enter data of type int, val
will be taken as 1 !!!\n");
            val = 1;
        }

        if (N > 10 && val == 3){
            throw new IllegalArgumentException("If you want to enter a value
from the keyboard - enter N <= 10.");
        }

        //------------------------------ Main Body ------------------------------
        System.out.println("\nLab2 started!\n");

        Data data = new Data(N);
        Semaphore sem = new Semaphore(1);
        Func1 func1 = new Func1("Func1", Thread.NORM_PRIORITY, data, sem, val);
        Func2 func2 = new Func2("Func2", Thread.MIN_PRIORITY, data, sem, val);
        Func3 func3 = new Func3("Func3", Thread.MAX_PRIORITY, data, sem, val);

        func1.start();
        func2.start();
        func3.start();

        try {
            func1.join();
            func2.join();
            func3.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        System.out.println("Lab2 finished.");
    }

    public static void main(String[] args){
        new Lab2().start();
    }
}
```

Func1.java

```
import java.util.concurrent.Semaphore;

public class Func1 extends Thread {

    private int val;
    private Data data;
    private Semaphore sem;

    Func1(String name, int priority, Data data, Semaphore sem, int val){
        setName(name);
        setPriority(priority);
        this.data = data;
        this.sem = sem;
        this.val = val;
    }

    // D = SORT(A)+SORT(B)+SORT(C)*(MA*ME)
    @Override
    public void run(){
        System.out.println("Func 1 started.");
        try {
            int[] A, B, C;
            int[][] MA, ME;
```

```
            // Generate Input Values
            if (val == 2) {
                sleep(200);
                A = data.randomVector(); B = data.randomVector(); C =
data.randomVector();
                MA = data.randomMatrix(); ME = data.randomMatrix();

                if (data.getN() <= 10){
                    System.out.println("\nFunc 1 Input Values:");
                    data.vectorOutput(A, 'A'); data.vectorOutput(B, 'B');
data.vectorOutput(C, 'C');
                    data.matrixOutput(MA, "MA"); data.matrixOutput(ME, "ME");
                }
            } else if (val == 3){
                sleep(50);
                System.out.println("Func 1 is waiting for a permit.");
                sem.acquire();
                sleep(100);
                System.out.println("Func 1 gets a permit.\n");

                A = data.vectorInput('A'); B = data.vectorInput('B'); C =
data.vectorInput('C');
                MA = data.matrixInput("MA"); ME = data.matrixInput("ME");
            } else {
                sleep(200);
                A = data.allOnesVector(); B = data.allOnesVector(); C =
data.allOnesVector();
                MA = data.allOnesMatrix(); ME = data.allOnesMatrix();
            }

            // Calculate The Result
            int[] result = data.func1(A, B, C, MA, ME);
            sleep(100);
            if (data.getN() <= 10) {
                System.out.print("Func 1 result:\n");
                data.vectorOutput(result, 'D');
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        if (val == 3) {
            System.out.println("Func 1 releases the permit.");
            sem.release();
        }
        System.out.println("Func 1 finished.\n");
    }
}
```

Func2.java

```
import java.util.concurrent.Semaphore;

public class Func2 extends Thread {

    private int val;
    private Data data;
    private Semaphore sem;

    Func2(String name, int priority, Data data, Semaphore sem, int val){
        setName(name);
        setPriority(priority);
        this.data = data;
        this.sem = sem;
```

```
        this.val = val;
    }


    // MF = (MG*MH)*TRANS(MK)
    @Override
    public void run(){
        System.out.println("Func 2 started.");
        try {
            int[][] MG, MH, MK;

            // Generate Input Values
            if (val == 2) {
                sleep(400);
                MG = data.randomMatrix(); MH = data.randomMatrix(); MK =
data.randomMatrix();

                if (data.getN() <= 10){
                    System.out.println("\nFunc 2 Input Values:");
                    data.matrixOutput(MG, "MG"); data.matrixOutput(MH, "MH");
data.matrixOutput(MK, "MK");
                }
            } else if (val == 3){
                sleep(50);
                System.out.println("Func 2 is waiting for a permit.");
                sem.acquire();
                sleep(100);
                System.out.println("Func 2 gets a permit.\n");
                MG = data.matrixInput("MG"); MH = data.matrixInput("MH"); MK =
data.matrixInput("MK");
            } else {
                sleep(400);
                MG = data.allOnesMatrix(); MH = data.allOnesMatrix();  MK =
data.allOnesMatrix();
            }

            // Calculate The Result
            int[][] result = data.func2(MG, MH, MK);
            sleep(100);
            if (data.getN() <= 10) {
                System.out.print("Func 2 result:\n");
                data.matrixOutput(result, "MF");
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        if (val == 3) {
            System.out.println("Func 2 releases the permit.");
            sem.release();
        }
        System.out.println("Func 2 finished.\n");
    }
}
```

Func3.java

```
import java.util.concurrent.Semaphore;

public class Func3 extends Thread {

    private int val;
    private Data data;
    private Semaphore sem;
```

```
    Func3(String name, int priority, Data data, Semaphore sem, int val){
        setName(name);
        setPriority(priority);
        this.data = data;
        this.sem = sem;
        this.val = val;
    }

    // S = (MO*MP)*V+t*MR*(O+P)
    @Override
    public void run() {
        System.out.println("Func 3 started.");
        try {
            int t;
            int[] V, O, P;
            int[][] MO, MP, MR;

            // Generate Input Values
            if (val == 2) {
                t = data.randomNum();
                V = data.randomVector(); O = data.randomVector(); P =
data.randomVector();
                MO = data.randomMatrix(); MP = data.randomMatrix(); MR =
data.randomMatrix();

                if (data.getN() <= 10) {
                    System.out.println("\nFunc 3 Input Values:");
                    data.numOutput(t, 't');
                    data.vectorOutput(V, 'V'); data.vectorOutput(O, 'O');
data.vectorOutput(P, 'P');
                    data.matrixOutput(MO, "MO"); data.matrixOutput(MP, "MP");
data.matrixOutput(MR, "MR");
                }
            } else if (val == 3) {
                sleep(50);
                System.out.println("Func 3 is waiting for a permit.");
                sem.acquire();
                sleep(100);
                System.out.println("Func 3 gets a permit.\n");

                t = data.numInput('t');
                V = data.vectorInput('V'); O = data.vectorInput('O'); P =
data.vectorInput('P');
                MO = data.matrixInput("MO"); MP = data.matrixInput("MP"); MR =
data.matrixInput("MR");
            }
            else {
                t = 1;
                V = data.allOnesVector(); O = data.allOnesVector(); P =
data.allOnesVector();
                MO = data.allOnesMatrix(); MP = data.allOnesMatrix(); MR =
data.allOnesMatrix();
            }

            // Calculate The Result
            int[] result = data.func3(t, V, O, P, MO, MP, MR);
            sleep(100);
            if (data.getN() <= 10) {
                System.out.print("Func 3 result:\n");
                data.vectorOutput(result, 'S');
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        if (val == 3) {
```

```
            System.out.println("Func 3 releases the permit.");
            sem.release();
        }
        System.out.println("Func 3 finished.\n");
    }
}
```

Data.java

```
import java.util.Arrays;
import java.util.Random;
import java.util.Scanner;

class Data {
    private int N;

    Data(int N){
        this.N = N;
    }

    int getN(){
        return N;
    }

    // ------------------- Create All-Ones Matrix And Vector -------------------
    int[][] allOnesMatrix(){
        int[][] MA = new int[N][N];
        for (int i = 0; i < MA.length; i++) {
            for (int j = 0; j < MA[i].length; j++) {
                MA[i][j] = 1;
            }
        }
        return MA;
    }

    int[] allOnesVector(){
        int[] A = new int[N];
        for (int i = 0; i < A.length; i++) {
            A[i] = 1;
        }
        return A;
    }


    // ---------------- Create Random Matrix, Vector And Number ----------------
    int[][] randomMatrix(){
        int[][] MA = new int[N][N];
        Random random = new Random();
        for (int i = 0; i < MA.length; i++) {
            for (int j = 0; j < MA[i].length; j++) {
                MA[i][j] = random.nextInt(10);
            }
        }
        return MA;
    }

    int[] randomVector(){
        int[] A = new int[N];
        Random random = new Random();
        for (int i = 0; i < A.length; i++) {
            A[i] = random.nextInt(10);
        }
        return A;
    }
```

```
    int randomNum(){
        Random random = new Random();
        return random.nextInt(10);
    }


    // ---------- Data Entry Handler For Matrices, Vectors And Numbers ---------
    int[][] matrixInput(String name){
        System.out.println("Enter the " + N*N + " elements of the Matrix " +
name + ":");
        int[][] MA = new int[N][N];
        Scanner scanner = new Scanner(System.in);
        for (int i = 0; i < MA.length; i++) {
            for (int j = 0; j < MA[i].length; j++) {
                System.out.print(name + "[" + i + "][" + j + "] = ");
                MA[i][j] = scanner.nextInt();
            }
        }
        return MA;
    }

    int[] vectorInput(char name){
        System.out.println("Enter the " + N + " elements of the Vector " + name
+ ":");
        int[] input = new int[N];
        Scanner scanner = new Scanner(System.in);
        for (int i = 0; i < input.length; i++) {
            System.out.print(name + "[" + i + "] = ");
            input[i] = scanner.nextInt();
        }
        return input;
    }

    int numInput(char name){
        System.out.print("Enter number " + name + " = ");
        Scanner scanner = new Scanner(System.in);
        return scanner.nextInt();
    }


    // ------------- Print Matrix, Vector And Number Into Console -------------
    void matrixOutput(int[][] MA, String name){
        System.out.println("\tMatrix " + name + ":");
        for (int[] i : MA) {
            System.out.print("\t\t");
            for (int j : i) {
                System.out.print(j + " ");
            }
            System.out.println();
        }
    }

    void vectorOutput(int[] A, char name){
        System.out.print("\tVector " + name + ": ");
        for (int i : A) {
            System.out.print(i + " ");
        }
        System.out.println();
    }

    void numOutput(int a, char name){
        System.out.print("\tNumber " + name + ": " + a + "\n");
    }
```

```java
// Sort Vector
private int[] sortVector(int[] A){
    Arrays.sort(A);
    return A;
}


// Calculates Sum Of 2 Vectors
private int[] sumVectors(int[] A, int[] B){
    int[] C = new int[N];
    for (int i = 0; i < N ; i++){
        C[i] = A[i] + B[i];
    }
    return C;
}


// Transposing Matrix
private int[][] matrixTransp(int[][] MA){
    int buf;
    for (int i = 0; i < MA.length ; i++){
        for (int j = 0; j <=i; j++){
            buf = MA[i][j];
            MA[i][j] = MA[j][i];
            MA[j][i] = buf;
        }
    }
    return MA;
}


// Multiply 2 Matrices
private int[][] matrixMult(int[][] MA, int[][] MB){
    int[][] MC = new int[N][N];
    for (int i = 0; i < N ; i++){
        for (int j = 0; j < N ; j++){
            for (int k = 0; k < N ; k++){
                MC[i][j] += MA[i][k] * MB[k][j];
            }
        }
    }
    return MC;
}


// Multiply Matrix And Vector
private int[] vectorMatrixMult(int[] A, int[][] MA){
    int[] B = new int[N];
    for (int i = 0; i < N ; i++){
        for (int j = 0; j < N ; j++){
            B[i] += A[j] * MA[i][j];
        }
    }
    return B;
}


// Multiply Integer And Matrix
private int[] intVectorMult(int a, int[] A) {
    int[] B = new int[N];
    for (int i = 0; i < N ; i++){
        B[i] = a * A[i];
    }
    return B;
}
```

```
    // F1 -> D = SORT(A)+SORT(B)+SORT(C)*(MA*ME)
    int[] func1(int[] A, int[] B, int[] C, int[][] MA, int[][] ME){
        return sumVectors(sumVectors(sortVector(A), sortVector(B)),
                vectorMatrixMult(sortVector(C), matrixMult(MA, ME)));
    }

    // F2 -> MF = (MG*MH)*TRANS(MK)
    int[][] func2(int[][] MG, int[][] MH, int[][] MK){
        return matrixMult(matrixMult(MG, MH), matrixTransp(MK));
    }

    // F3 -> S = (MO*MP)*V+t*MR*(O+P)
    int[] func3(int t, int[] V, int[] O, int[] P, int[][] MO, int[][] MP,
int[][] MR){
        return sumVectors((vectorMatrixMult(V, matrixMult(MO, MP))),
                intVectorMult(t, vectorMatrixMult(sumVectors(O, P), MR))
        );
    }
}
```

Приклад роботи програми

```
-----------------------------------------------------
| Function 1 | D = SORT(A)+SORT(B)+SORT(C)*(MA*ME) |
| Function 2 |      MF = (MG*MH)*TRANS(MK)         |
| Function 3 |     S = (MO*MP)*V+t*MR*(O+P)        |
-----------------------------------------------------

!!! Note that if the value of N > 10 -> the result will not be displayed !!!
!!! If you enter N <= 0 - execution will be terminated !!!

Enter N: 2

Select the method according to which the initial data will be generated:
      [1] - Create All-Ones Matrices And Vectors.
      [2] - Create Matrices And Vectors With Random Values.
      [3] - Enter All Values From The Keyboard.
Your choice [1] / [2] / [3]: 2

Lab2 started!

Func 1 started.
Func 2 started.
Func 3 started.

Func 3 Input Values:
      Number t: 3
      Vector V: 8 6
      Vector O: 8 6
      Vector P: 5 9
      Matrix MO:
            8 9
            9 2
      Matrix MP:
            6 3
            6 4
```

```
      Matrix MR:
            4 8
            9 9
Func 3 result:
      Vector S: 1773 1407
Func 3 finished.


Func 1 Input Values:
      Vector A: 9 3
      Vector B: 0 1
      Vector C: 7 6
      Matrix MA:
            8 0
            4 6
      Matrix ME:
            5 9
            3 8
Func 1 result:
      Vector D: 509 1030
Func 1 finished.


Func 2 Input Values:
      Matrix MG:
            9 5
            0 3
      Matrix MH:
            9 4
            8 1
      Matrix MK:
            9 3
            2 7
Func 2 result:
      Matrix MF:
            1212 529
            225 69
Func 2 finished.

Lab2 finished.

Process finished with exit code 0
```

Перевірка результатів прикладу

У якості вхідних даних були взяті саме рандомізовані значення для скаляру, векторів та матриць, оскільки на них можна перевірити такі процедури як сортування та транспонування матриць, чого б не було видно, якщо б ми взяли всі значення як одиниці.

Перевірка T1 з обчисленням F1

Умова: $D = SORT(A) + SORT(B) + SORT(C) * (MA * ME)$

$Sort(A) \rightarrow (3 \quad 9)$

$Sort(B) \rightarrow (0 \quad 1)$

$Sort(C) \rightarrow (6 \quad 7)$

$MA \cdot ME = \begin{pmatrix} 8 & 0 \\ 4 & 6 \end{pmatrix} \cdot \begin{pmatrix} 5 & 9 \\ 3 & 8 \end{pmatrix} = \begin{pmatrix} 40 & 72 \\ 38 & 84 \end{pmatrix}$

$SORT(C) \cdot (MA \cdot ME) = (6 \quad 7) \cdot \begin{pmatrix} 40 & 72 \\ 38 & 84 \end{pmatrix} = (506 \quad 1020)$

$D = (3 \quad 9) + (0 \quad 1) + (506 \quad 1020) = (509 \quad 1030)$

Перевірка T2 з обчисленням F2

Умова: $MF = (MG * MH) * TRANS(MK)$

$MG \cdot MH = \begin{pmatrix} 9 & 5 \\ 0 & 3 \end{pmatrix} \cdot \begin{pmatrix} 9 & 4 \\ 8 & 1 \end{pmatrix} = \begin{pmatrix} 121 & 41 \\ 24 & 3 \end{pmatrix}$

$TRANS(MK) \rightarrow \begin{pmatrix} 9 & 2 \\ 3 & 7 \end{pmatrix}$

$MF = \begin{pmatrix} 121 & 41 \\ 24 & 3 \end{pmatrix} \cdot \begin{pmatrix} 9 & 2 \\ 3 & 7 \end{pmatrix} = \begin{pmatrix} 1212 & 529 \\ 225 & 69 \end{pmatrix}$

Перевірка T3 з обчисленням F3

Умова: $S = (MO * MP) * V + t * MR * (O + P)$

$MO \cdot MP = \begin{pmatrix} 8 & 9 \\ 9 & 2 \end{pmatrix} \cdot \begin{pmatrix} 6 & 3 \\ 6 & 4 \end{pmatrix} = \begin{pmatrix} 102 & 60 \\ 66 & 35 \end{pmatrix}$

$O + P = (8 \quad 6) + (5 \quad 9) = (13 \quad 15)$

$t \cdot MR = 3 \cdot \begin{pmatrix} 4 & 8 \\ 9 & 9 \end{pmatrix} = \begin{pmatrix} 12 & 24 \\ 27 & 27 \end{pmatrix}$

$V \cdot (MO \cdot MP) = (8 \quad 6) \cdot \begin{pmatrix} 102 & 60 \\ 66 & 35 \end{pmatrix} = (1212 \quad 690)$

$(O + P) \cdot t \cdot MR = (13 \quad 15) \cdot \begin{pmatrix} 12 & 24 \\ 27 & 27 \end{pmatrix} = (561 \quad 717)$

$S = (1212 \quad 690) + (561 \quad 717) = (1773 \quad 1407)$

Результати програми повністю збігаються з теоретичними даними. Програма працює вірно.

Висновок

В даній лабораторній роботі було розглянуто принцип побудови паралельних програм на основі мови Java. Було використано семафори для обробки вводу даних з клавіатури, при генерації рандомізованих значень чи встановлення всіх вхідних значень як 1 – семафори не використовуються.

Було зроблено аналіз проблем, які виникли під час виконання, а також запропоновано шляхи їх рішення, перевірено працездатність програми і програмним шляхом оброблено виключні ситуації.

Кінцева мета роботи досягнута.

Додаток А

Приклад вводу даних з консолі

```
      ------------------------------------------------
      | Function 1 | D = SORT(A)+SORT(B)+SORT(C)*(MA*ME) |
      | Function 2 |      MF = (MG*MH)*TRANS(MK)         |
      | Function 3 |     S = (MO*MP)*V+t*MR*(O+P)        |
      ------------------------------------------------

      !!! Note that if the value of N > 10 -> the result will not be displayed
!!!
      !!! If you enter N <= 0 - execution will be terminated !!!

      Enter N: 2

      Select the method according to which the initial data will be generated:
            [1] - Create All-Ones Matrices And Vectors.
            [2] - Create Matrices And Vectors With Random Values.
            [3] - Enter All Values From The Keyboard.
      Your choice [1] / [2] / [3]: 3

      Lab2 started!

      Func 1 started.
      Func 2 started.
      Func 3 started.
      Func 2 is waiting for a permit.
      Func 1 is waiting for a permit.
      Func 3 is waiting for a permit.
      Func 2 gets a permit.

      Enter the 4 elements of the Matrix MG:
      MG[0][0] = 2
      MG[0][1] = 2
      MG[1][0] = 2
      MG[1][1] = 2
      Enter the 4 elements of the Matrix MH:
      MH[0][0] = 2
      MH[0][1] = 2
      MH[1][0] = 2
      MH[1][1] = 2
      Enter the 4 elements of the Matrix MK:
      MK[0][0] = 2
      MK[0][1] = 2
      MK[1][0] = 2
      MK[1][1] = 2
      Func 2 result:
            Matrix MF:
                  32 32
                  32 32
      Func 2 releases the permit.
      Func 2 finished.
```

```
Func 1 gets a permit.

Enter the 2 elements of the Vector A:
A[0] = 1
A[1] = 1
Enter the 2 elements of the Vector B:
B[0] = 1
B[1] = 1
Enter the 2 elements of the Vector C:
C[0] = 1
C[1] = 1
Enter the 4 elements of the Matrix MA:
MA[0][0] = 1
MA[0][1] = 1
MA[1][0] = 1
MA[1][1] = 1
Enter the 4 elements of the Matrix ME:
ME[0][0] = 1
ME[0][1] = 1
ME[1][0] = 1
ME[1][1] = 1
Func 1 result:
      Vector D: 6 6
Func 1 releases the permit.
Func 1 finished.

Func 3 gets a permit.

Enter number t = 3
Enter the 2 elements of the Vector V:
V[0] = 3
V[1] = 3
Enter the 2 elements of the Vector O:
O[0] = 3
O[1] = 3
Enter the 2 elements of the Vector P:
P[0] = 3
P[1] = 3
Enter the 4 elements of the Matrix MO:
MO[0][0] = 3
MO[0][1] = 3
MO[1][0] = 3
MO[1][1] = 3
Enter the 4 elements of the Matrix MP:
MP[0][0] = 3
MP[0][1] = 3
MP[1][0] = 3
MP[1][1] = 3
Enter the 4 elements of the Matrix MR:
MR[0][0] = 3
MR[0][1] = 3
MR[1][0] = 3
MR[1][1] = 3
Func 3 result:
```

```
      Vector S: 216 216
Func 3 releases the permit.
Func 3 finished.

Lab2 finished.

Process finished with exit code 0
```