НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №3

з дісципліни **«**Паралельні та розподілені обчислення**»**

Виконав:

студент 3 курсу

ФІОТ гр. ІО-21

Кузьменко Володимир

Перевірив:

Корочкін О. В.

Київ – 2014 р.

ПОТОКИ В ЯЗЫКЕ JAVA

Цель работы: изучение средств языка Java для работы с потоками (процессами).

Выполнение работы: Разработать программу, содержащую параллельные потоки, каждый из которых реализует функцию F1, F2, F3 из лабораторной работы номер 1. Требования к созданию потоков и необходимые исследования программы описаны в

лабораторной работе 2.

В потоках использовать методы sleep() и join().

Лістинг програми:

/\*

\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Laboratory work #3. \*

\* Threads in Java \*

\* \*

\* Task: \*

\* F1: 1.24 E = A + C \* (MA \* MZ) + B \*

\* F2: 2.14 MC = SORT(MA + MB \* MM) \*

\* F3: 3.18 p = MAX(SORT(MS) + MA\* MM) \*

\* \*

\* Autor: Kuzmenko Volodia \*

\* Group: IO-21 \*

\* Date: 24.09.2014 \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

import java.util.Scanner;

public class MainTherad {

// main thread

public static void main(String[] args) throws InterruptedException {

Scanner sc = new Scanner(System.in);

System.out.println("Input N");

int N = sc.nextInt();

Thread t1 = new Thread(null, new Task1\_F1(N, sc), "Task 1", 1000);

Thread t2 = new Thread(null, new Task2\_F2(N, sc), "Task 2", 1000);

Thread t3 = new Thread(null, new Task3\_F3(N, sc), "Task 3", 1000);

t1.start();

t1.join();

t2.start();

t3.start();

}

}

/\*

\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Task F1 \*

\* \*

\* Autor: Kuzmenko Volodia \*

\* Group: IO-21 \*

\* Date: 24.09.2014 \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

**import** java.util.Scanner;

**public** **class** Task1\_F1 **extends** Data **implements** Runnable {

**public** **Task1\_F1**(**int** N, Scanner sc) {

**super**(N, sc);

}

@Override

**public** **void** **run**() {

System.out.println("Task 1 started");

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

**int** [][] MA,MZ;

System.out.println("Input vector A in T1");

A = *vector\_input*();

System.out.println("Output vector A from T1");

*vector\_output*(A);

System.out.println("Input vector B in T1");

B = *vector\_input*();

System.out.println("Output vector B from T1");

*vector\_output*(B);

System.out.println("Input vector C in T1");

C = *vector\_input*();

System.out.println("Output vector C from T1");

*vector\_output*(C);

System.out.println("Input matrix MA in T1 ");

MA = *matrix\_input*();

System.out.println("Output matrix MA from T1");

*matrix\_output*(MA);

System.out.println("Input matrix MZ in T1");

MZ = *matrix\_input*();

System.out.println("Output matrix MZ from T1");

*matrix\_output*(MZ);

System.out.println("Calculation T1...");

**try** {

Thread.*sleep*(10000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

E = func1(A, B, C, MA, MZ);

System.out.println("F1:");

*vector\_output*(E);

System.out.println("Task 1 finished");

}

**private** **int**[] **func1**(**int**[] A, **int**[] B, **int**[] C, **int**[][] MA, **int**[][] MZ) {

**return** *addVec*(*addVec*(*multingMatrixOnVector*(*multMatr*(MA, MZ), C), A), B);

}

}

/\*

\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Utils to work with vectors and matrix \*

\* \*

\* Autor: Kuzmenko Volodia \*

\* Group: IO-21 \*

\* Date: 24.09.2014 \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

import java.util.Arrays;

import java.util.Scanner;

public class Data {

private static int N;

private static Scanner sc;

protected Data(int N, Scanner sc) {

this.N = N;

this.sc = sc;

}

protected synchronized static int[] vector\_input() {

int[] A = new int[N];

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

A[i] = sc.nextInt();

}

return A;

}

protected synchronized static int[][] matrix\_input() {

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] = sc.nextInt();

}

}

return MA;

}

protected static void vector\_output(int[] vector) {

for (int j = 0; j < vector.length; j++) {

System.out.print("|" + vector[j] + "|");

}

System.out.println();

}

protected static void matrix\_output(int[][] matrix) {

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

for (int j = 0; j < matrix[i].length; j++) {

System.out.print("|" + matrix[i][j] + "|");

}

System.out.println();

}

}

protected static int[][] multMatr(final int[][] MA, final int[][] MZ) {

int[][] MT = new int[MA.length][MA.length];

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

for (int j = 0; j < MA.length; j++) {

int s = 0;

for (int k = 0; k < MA.length; k++) {

s += MA[i][k] \* MZ[k][j];

MT[i][j] = s;

}

}

}

return MT;

}

protected static int[] multingMatrixOnVector(final int[][] MA, final int[] A) {

int[] R = new int[A.length];

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

int s = 0;

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

s += MA[i][j] \* A[j];

}

R[i] = s;

}

return R;

}

protected static int[] addVec(final int[] B, final int[] C) {

int[] K = new int[B.length];

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

K[i] = B[i] + C[i];

}

return K;

}

protected static int[][] addMatr(final int[][] MA, final int[][] MB) {

int[][] MM = new int[MA.length][MA.length];

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

for (int j = 0; j < MM.length; j++) {

MM[i][j] = MA[i][j] + MB[i][j];

}

}

return MM;

}

protected static int[] sortVector(final int[] A) {

int[] B = A;

Arrays.sort(B);

return B;

}

protected static int[][] sortMatr(final int[][] MA) {

int[][] MR = MA;

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

MR[i] = sortVector(MR[i]);

}

return MA;

}

protected static int maxElement(final int[][] MD) {

int m = MD[0][0];

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

for (int j = 0; j < MD.length; j++) {

if (m < MD[i][j])

m = MD[i][j];

}

}

return m;

}

}

/\*

\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Task F2 \*

\* \*

\* Autor: Kuzmenko Volodia \*

\* Group: IO-21 \*

\* Date: 24.09.2014 \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

import java.util.Scanner;

public class Task2\_F2 extends Data implements Runnable {

public Task2\_F2(int N, Scanner sc) {

super(N, sc);

}

@Override

public void run() {

System.out.println("Task 2 started");

int[][] MA, MB, MC, MM;

System.out.println("Input matrix MA in T2");

MA = matrix\_input();

System.out.println("Output matrix MA from T2");

matrix\_output(MA);

System.out.println("Input matrix MB in T2");

MB = matrix\_input();

System.out.println("Output matrix MB from T2");

matrix\_output(MB);

System.out.println("Input matrix MM in T2");

MM = matrix\_input();

System.out.println("Output matrix MM from T2");

matrix\_output(MM);

System.out.println("Calculation T2... ");

try {

Thread.sleep(10000);

} catch (InterruptedException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

MC = func2(MA, MB, MM);

System.out.println("F2: ");

matrix\_output(MC);

System.out.println("Task 2 finished");

}

private int[][] func2(int[][] MA, int[][] MB, int[][] MM) {

return sortMatr(addMatr(multMatr(MB, MM), MA));

}

}

/\*

\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Task F3 \*

\* \*

\* Autor: Kuzmenko Volodia \*

\* Group: IO-21 \*

\* Date: 24.09.2014 \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

import java.util.Scanner;

public class Task3\_F3 extends Data implements Runnable {

public Task3\_F3(int N, Scanner sc) {

super(N, sc);

}

@Override

public void run() {

System.out.println("Task 3 started");

int[][] MA, MS, MM;

int P;

System.out.println("Input matrix MA in T3");

MA = matrix\_input();

System.out.println("Output matrix MA from T3");

matrix\_output(MA);

System.out.println("Input matrix MM in T3");

MM = matrix\_input();

System.out.println("Output matrix MM from T3");

matrix\_output(MM);

System.out.println("Input matrix MS in T3");

MS = matrix\_input();

System.out.println("Output matrix MS from T3");

matrix\_output(MS);

System.out.println("Calculation T3...");

try {

Thread.sleep(10000);

} catch (InterruptedException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

P = func3(MA, MM, MS);

System.out.println("F3: ");

System.out.println(P);

System.out.println("Task 3 finished");

}

private int func3(int[][] MA, int[][] MM, int[][] MS) {

return maxElement(addMatr(sortMatr(MS), multMatr(MA, MM)));

}

}