Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Основи паралельного програмування

Лабораторна робота №2

**«Потоки в мові Java»**

Виконала:

студентка групи ІО-64

Бровченко А. В.

Перевірив:

Корочкін О. В.

Київ

2018 р.

**Завдання:**

F1: C = A – B\*(MA\*MС)\*е

F2: MF = MF\*MG\*k

F3: O = SORT(P)\*(MR\*MT)

**Лістинг програми:**

*/\*\*  
 \* Parallel and distributed computing.  
 \* Labwork 2. Java.  
 \*  
 \* Anastasiya Brovchenko  
 \* IO-64  
 \* 02.10.2018  
 \*  
 \* F1: C = A-B\*(MA\*MC)\*e  
 \* F2: MF = MF\*MG\*k  
 \* F3: O = sort(P)\*(MR\*MT)  
 \*/***public class** Lab2 **extends** Thread {  
  
 **public static void** main(String[] args) {  
 (**new** Lab2()).start();  
 }  
  
 @Override  
 **public void** run() {  
 setName(**"Lab 2"**);  
 System.***out***.println(**"Lab 2 start\n"**);  
 **int** n = 1000;  
 Data data = **new** Data(n);  
 F1 f1 = **new** F1(**"F1"**, Thread.***NORM\_PRIORITY***, data);  
 F2 f2 = **new** F2(**"F2"**, Thread.***MIN\_PRIORITY***, data);  
 F3 f3 = **new** F3(**"F3"**, Thread.***MAX\_PRIORITY***, data);  
 f1.start();  
 f2.start();  
 f3.start();  
 **try** {  
 f1.join();  
 f2.join();  
 f3.join();  
 *//System.out.println(f1.getResult());  
 //System.out.println(f2.getResult());  
 //System.out.println(f3.getResult());* } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 System.***out***.println(**"\nLab 2 end"**);  
 }  
}

*/\*\*  
 \* F1: C = A-B\*(MA\*MC)\*e  
 \*/***public class** F1 **extends** Thread {  
  
 **int**[] **A**, **B**;  
 **int e**;  
 **int**[][] **MA**, **MC**;  
 **private int**[] **result**;  
 **private** Data **d**;  
  
 F1(String name, **int** priority, Data d) {  
 setName(name);  
 setPriority(priority);  
 **this**.**d** = d;  
 }  
  
 @Override  
 **public void** run() {  
 **try** {  
 System.***out***.println(**"Task 1 start"**);  
 **int** n = **d**.getN();  
 **int**[] A = **new int**[n], B = **new int**[n];  
 **int** e = 3;  
 **int**[][] MA = **new int** [n][n], MC = **new int**[n][n];  
 **result** = **d**.func1(A, B, MA, MC, e);  
 *sleep*(500);  
 System.***out***.println(**"Task 1 end"**);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
}

*/\*\*  
 \* F2: MF = MF\*MG\*k  
 \*/***public class** F2 **extends** Thread {  
 **int k**;  
 **int**[][] **MF**, **MG**;  
 **private int**[][] **result**;  
 **private** Data **d**;  
  
 F2(String name, **int** priority, Data d) {  
 setName(name);  
 setPriority(priority);  
 **this**.**d** = d;  
 }  
  
 @Override  
 **public void** run() {  
 **try** {  
 System.***out***.println(**"Task 2 start"**);  
 **int** n = **d**.getN();  
 **int** k = 3;  
 **int**[][] MF = **new int** [n][n], MG = **new int**[n][n];  
 **result** = **d**.func2(k,MF,MG);  
 *sleep*(250);  
 System.***out***.println(**"Task 2 end"**);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
}

*/\*\*  
 \* F3: O = sort(P)\*(MR\*MT)  
 \*/***public class** F3 **extends** Thread {  
 **int**[] **P**;  
 **int**[][] **MR**, **MT**;  
 **private int**[] **result**;  
 **private** Data **d**;  
  
 F3(String name, **int** priority, Data d) {  
 setName(name);  
 setPriority(priority);  
 **this**.**d** = d;  
 }  
  
 @Override  
 **public void** run() {  
 **try** {  
 System.***out***.println(**"Task 3 start"**);  
 **int** n = **d**.getN();  
 **int**[] P = **new int**[n];  
 **int**[][] MR = **new int** [n][n], MT = **new int**[n][n];  
 **result** = **d**.func3(P, MR, MT);  
 *sleep*(100);  
 System.***out***.println(**"Task 3 end"**);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
}

**import** java.util.Arrays;  
**import** java.util.Scanner;  
*/\*\*  
 \* F1: C = A-B\*(MA\*MC)\*e  
 \* F2: MF = MF\*MG\*k  
 \* F3: O = sort(P)\*(MR\*MT)  
 \*  
 \*/***public class** Data {  
 **private int n**;  
  
 **public int** getN() {  
 **return n**;  
 }  
  
 **public** Data(**int** n) {  
 **this**.**n** = n;  
 }  
  
 **public int**[] vectorInput() {  
 **int**[] vector = **new int**[**n**];  
 Scanner sc = **new** Scanner(System.***in***);  
 **for** (**int** i = 0; i < **n** ; i++){  
 vector[i] = sc.nextInt();  
 }  
 **return** vector;  
 }  
  
 **public int**[][] matrixInput() {  
 **int**[][] matrix = **new int**[**n**][**n**];  
 Scanner sc = **new** Scanner(System.***in***);  
 **for** (**int** i = 0; i < **n** ; i++){  
 **for** (**int** j = 0; j < **n** ;  
 j++){  
 matrix[i][j] = sc.nextInt();  
 }  
 }  
 **return** matrix;  
 }  
  
 **public int**[] func3(**int**[] p, **int**[][] mr, **int**[][] mt) {  
 **return** vectorMatrixMult(vectorSort(p), matrixMult(mr, mt));  
 }  
  
 **public int**[][] func2(**int** k, **int**[][] mf, **int**[][] mg) {  
 **return** intMatrixMult(k ,matrixMult(mf, mg));  
 }  
  
 **public int**[] func1(**int**[] a, **int**[] b, **int**[][] ma, **int**[][] mc, **int** e) {  
 **return** vectorDiff(a, intVectorMult(e, vectorMatrixMult(b, matrixMult(ma,mc))));  
 }  
  
 **private int**[] vectorAdd(**int**[] a, **int**[] b) {  
 **if** (a.**length** != **n** || b.**length** != **n**) {  
 **return null**;  
 }  
 **int**[] c = **new int**[**n**];  
 **for** (**int** i = 0; i < **n** ; i++){  
 c[i] = a[i] + b[i];  
 }  
 **return** c;  
 }  
  
 **private int**[] vectorDiff(**int**[] a, **int**[] b) {  
 **if** (a.**length** != **n** || b.**length** != **n**) {  
 **return null**;  
 }  
 **int**[] c = **new int**[**n**];  
 **for** (**int** i = 0; i < **n** ; i++){  
 c[i] = a[i] - b[i];  
 }  
 **return** c;  
 }  
  
 **private int**[][] matrixTrans(**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;  
 }  
  
 **private int**[] vectorMatrixMult(**int**[] a, **int**[][] ma) {  
 **if** (a.**length** != **n** || ma.**length** != **n**) {  
 **return null**;  
 }  
 **int**[] c = **new int**[**n**];  
 **for** (**int** i = 0; i < **n** ; i++){  
 **for** (**int** j = 0; j < **n** ; j++){  
 c[i] += a[j] \* ma[j][i];  
 }  
 }  
 **return** c;  
 }  
  
 **private int**[] vectorMult(**int**[] a, **int**[] b) {  
 **if** (b.**length** !=**n** || a.**length** != **n**) {  
 **return null**;  
 }  
 **int**[] c = **new int**[**n**];  
 **for** (**int** i = 0; i < **n** ; i++){  
 c[i] = a[i] \* b[i];  
 }  
 **return** c;  
 }  
  
 **private int**[][] matrixMult(**int**[][] ma, **int**[][] mb) {  
 **if** (ma.**length** != **n** || mb.**length** != **n**) {  
 **return null**;  
 }  
 **int**[][] c = **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++){  
 c[i][j] += ma[i][k] \* mb[k][j];  
 }  
 }  
 }  
 **return** c;  
 }  
  
 **private int**[] intVectorMult(**int** a, **int**[] b) {  
 **int**[] c = **new int**[**n**];  
 **for** (**int** i = 0; i < **n** ; i++){  
 c[i] = a \* b[i];  
 }  
 **return** c;  
 }  
  
 **private int**[][] intMatrixMult(**int** a, **int**[][] b) {  
 **int**[][] c = **new int**[**n**][**n**];  
 **for** (**int** i = 0; i < **n** ; i++){  
 **for** (**int** j = 0; j < **n** ; j++) {  
 c[i][j] = a \* b[i][j];  
 }  
 }  
 **return** c;  
 }  
  
 **private int**[] vectorSort(**int**[] a) {  
 **if** (a.**length** != **n**) {  
 **return null**;  
 }  
 **int**[] c = Arrays.*copyOf*(a, **n**);  
 Arrays.*sort*(c);  
 **return** c;  
 }  
}

**Результат роботи:**

Lab 2 start

Task 1 start

Task 3 start

Task 2 start

Task 2 end

Task 3 end

Task 1 end

Lab 2 end

Process finished with exit code 0