Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Лабораторна робота №4**

*по курсу*

*«Паралельні та розподілені обчислення»*

Виконав:

студент III курсу

гр. IO-82

Куцовол В.В.

Київ 2010

**package** Lab4;

/\*

\* @author Куцовол Віктор

\* ІІІ курс ФІОТ

\* гр. ІО-82

\* Лабораторна робота №4

\* F1: 1.18 d=(A\*B)-(C\*B)

\* F2: 2.19 v=MAX(MA+MB\*MC)

\* F3: 3.20 R=SORT(B-C)\*SORT(MA\*MB)

\*/

**public** **class** Lab4 {

**public** **static** **void** main(String[] args) {

Thread1 t1 = **new** Thread1("Thread1", 6); // Пріоритет 6

Thread2 t2 = **new** Thread2("Thread2", 7); // Пріоритет 7

Thread3 t3 = **new** Thread3("Thread3", 5); // Пріоритет 5

// Запуск потоків

t1.start();

t2.start();

t3.start();

// Синхронізація головного потоку з потоками, який він запускає на

// виконання

**try** {

t1.join();

t2.join();

t3.join();

} **catch** (InterruptedException e) {

System.*out*.println("Interrupt the main thread");

}

}

}

**package** Lab4;

**import** Data.Functions;

**import** Data.InOut;

**public** **class** Thread1 **extends** Thread {

**public** Thread1(String name, **int** newPriority) {

**super**(name);

setPriority(newPriority);

}

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

System.*out*.println(getName() + " started.");

**final** **int** VALUE = 1;

**int**[] A = **new** **int**[Functions.*N*];

**int**[] B = **new** **int**[Functions.*N*];

**int**[] C = **new** **int**[Functions.*N*];

**int** d;

// Введення даних

InOut.*Vector\_Input*(A, VALUE);

InOut.*Vector\_Input*(B, VALUE);

InOut.*Vector\_Input*(C, VALUE);

// Виконання функцій

d = Functions.*f1*(A, B, C);

**try** {

*sleep*(2000);

} **catch** (InterruptedException e) {

System.*out*.println(getName() + " interrupt");

}

// Виведення результату

System.*out*.print("Value d = ");

System.*out*.println(d);

System.*out*.println(getName() + " finished.");

}

}

**package** Lab4;

**import** Data.Functions;

**import** Data.InOut;

**public** **class** Thread2 **extends** Thread {

**public** Thread2(String name, **int** newPriority) {

**super**(name);

setPriority(newPriority);

}

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

System.*out*.println(getName() + " started.");

**final** **int** VALUE = 2;

**int**[][] MA = **new** **int**[Functions.*N*][Functions.*N*];

**int**[][] MB = **new** **int**[Functions.*N*][Functions.*N*];

**int**[][] MC = **new** **int**[Functions.*N*][Functions.*N*];

**int** v;

// Введення даних

InOut.*Matrix\_Input*(MA, VALUE);

InOut.*Matrix\_Input*(MB, VALUE);

InOut.*Matrix\_Input*(MC, VALUE);

// Виконання функцій

v = Functions.*f2*(MA, MB, MC);

**try** {

*sleep*(500);

} **catch** (InterruptedException e) {

System.*out*.println(getName() + " interrupt");

}

// Виведення результату

System.*out*.print("Value v = ");

System.*out*.println(v);

System.*out*.println(getName() + " finished.");

}

}

**package** Lab4;

**import** Data.Functions;

**import** Data.InOut;

**public** **class** Thread3 **extends** Thread {

**public** Thread3(String name, **int** newPriority) {

**super**(name);

setPriority(newPriority);

}

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

System.*out*.println(getName() + " started.");

**final** **int** VALUE = 3;

**int**[] B = **new** **int**[Functions.*N*], C = **new** **int**[Functions.*N*];

**int**[][] MA = **new** **int**[Functions.*N*][Functions.*N*];

**int**[][] MB = **new** **int**[Functions.*N*][Functions.*N*];

**int**[] R;

// Введення даних

InOut.*Vector\_Input*(B, VALUE);

InOut.*Vector\_Input*(C, VALUE);

InOut.*Matrix\_Input*(MA, VALUE);

// Виконання функцій

R = Functions.*f3*(B, C, MA, MB);

**try** {

*sleep*(100);

} **catch** (InterruptedException e) {

System.*out*.println(getName() + " interrupt");

}

// Виведення результату

System.*out*.print("Vector R: ");

InOut.*Vector\_Output*(R);

System.*out*.println(getName() + " finished.");

}

}

**package** Data;

**import** java.util.Arrays;

**public** **class** Functions {

**public** **static** **int** *N* = 5999;

**public** **static** **int** f1(**int**[] A, **int**[] B, **int**[] C) {

**return** *mult*(A, B) - *mult*(C, B);

}

**public** **static** **int** f2(**int**[][] MA, **int**[][] MB, **int**[][] MC) {

**return** *maxElMatr*(*add*(MA, *multMatrix*(MB, MC)));

}

**public** **static** **int**[] f3(**int**[] B, **int**[] C, **int**[][] MA, **int**[][] MB) {

**int**[] arr = *sub*(B, C);

Arrays.*sort*(arr);

**return** *multMonV*(*sortMatrix*(*multMatrix*(MA, MB)), arr);

}

**private** **static** **int** mult(**int**[] A, **int**[] B) {

**int** res = 0;

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

res += A[i] \* B[i];

**return** res;

}

**private** **static** **int**[][] add(**int**[][] MA, **int**[][] MB) {

**int**[][] res = **new** **int**[MA[0].length][MA[0].length];

**for** (**int** i = 0; i < MA[0].length; i++)

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

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

}

**return** res;

}

**private** **static** **int**[][] multMatrix(**int**[][] MB, **int**[][] MC) {

**int**[][] res = **new** **int**[MB[0].length][MB[0].length];

**int** s;

**for** (**int** i = 0; i < MB[0].length; i++)

**for** (**int** j = 0; j < MB[0].length; j++) {

s = 0;

**for** (**int** k = 0; k < MB[0].length; k++)

s = s + MB[i][k] \* MC[k][j];

res[i][j] = s;

}

**return** res;

}

**private** **static** **int** maxElMatr(**int**[][] MA) {

**int** res = MA[0][0];

**for** (**int** i = 0; i < MA[0].length; i++)

**for** (**int** j = 0; i < MA[0].length; i++)

**if** (res < MA[i][j])

res = MA[i][j];

**return** res;

}

**private** **static** **int**[] sub(**int**[] B, **int**[] C) {

**int**[] res = **new** **int**[B.length];

**for** (**int** i = 0; i < B.length; i++)

res[i] = B[i] - C[i];

**return** res;

}

**private** **static** **int**[][] sortMatrix(**int**[][] MA) {

**int**[][] res = **new** **int**[MA[0].length][MA[0].length];

**int** buf, k;

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

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

k = j;

**for** (**int** l = 0; l < MA[0].length; l++)

**if** (MA[i][k] < MA[i][l])

k = l;

buf = MA[i][k];

MA[i][k] = MA[i][j];

MA[i][j] = buf;

}

}

**return** res;

}

**private** **static** **int**[] multMonV(**int**[][] MA, **int**[] A) {

**int**[] res = **new** **int**[A.length];

**int** s;

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

s = 0;

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

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

res[i] = s;

}

**return** res;

}

}

**package** Data;

**import** java.util.Arrays;

**public** **class** InOut {

**public** **static** **void** Vector\_Input(**int**[] v, **int** value) {

Arrays.*fill*(v, value);

}

**public** **static** **void** Vector\_Output(**int**[] v) {

*display*(v);

}

**public** **static** **void** Matrix\_Input(**int**[][] m, **int** value) {

**for** (**int** i = 0; i < m.length; i++)

Arrays.*fill*(m[i], value);

}

**public** **static** **void** Matrix\_Output(**int**[][] m) {

**for** (**int** i = 0; i < m[0].length; i++) {

*display*(m[i]);

}

}

**public** **static** **void** display(**int** array[]) {

**for** (**int** i = 0; i < array.length; i++)

System.*out*.print(array[i] + " ");

System.*out*.println("");

}

}