Kpi-best

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМ. І.СІКОРСЬКОГО»

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

**Лабораторна робота №3**

з дисципліни «Паралельні та розподілені обчислення»

на тему: «Потоки в мові C#»

Виконала:

студентка 3-го курсу

факультету ІОТ

групи ІО-43

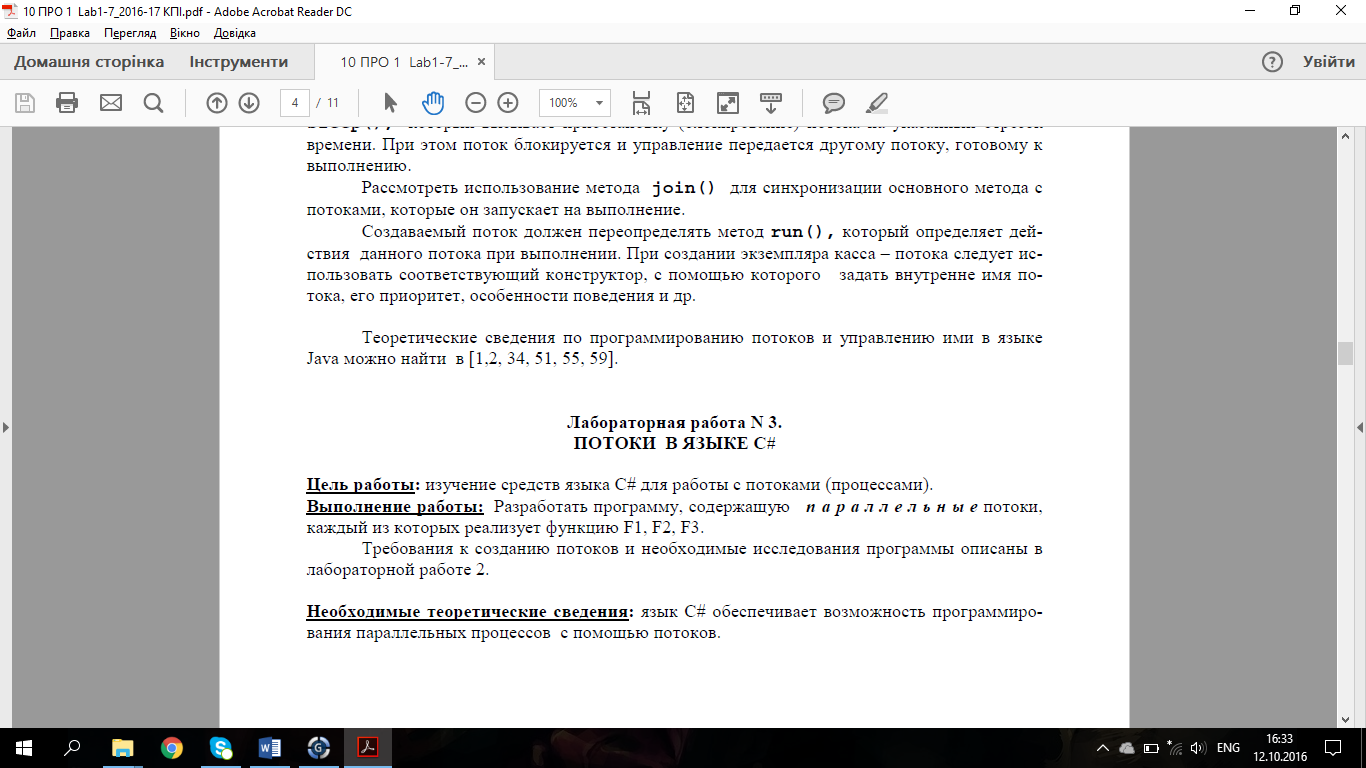
Даніленко Н.

Перевірив:

доц.

Долголенко О. М.

Київ 2016



**Варіант 4**

1.4 C = A + SORT(B) \* (MA \* ME).

2.4 MG = MAX(MH) \* (MK \* ML).

3.4 O = SORT(P) \* SORT (MR \* MS).

**Лістинг:**

**Tools.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication1

{

class Tools

{

/\*\*

\* Додає два вектори

\* @param vectA вектор

\* @param vectB вектор

\* @return вектор суми

\*/

public static double[] addVect(double[] vectA, double[] vectB)

{

for (int i = 0; i < vectB.GetLength(0); i++)

{

vectA[i] += vectB[i];

}

return vectA;

}

/\*\*

\* Перемножає дві матриці

\* @param matrA матриця

\* @param matrB матриця

\* @return повертає добуток двох матриць

\* \*/

public static double[,] MultMatr(double[,] matrA, double[,] matrB)

{

double temp = 0;

double[,] resultMatr = new double[matrA.GetLength(0), matrA.GetLength(1)];

for (int i = 0; i < matrA.GetLength(0); i++)

{

for (int j = 0; j < matrA.GetLength(1); j++)

{

temp = 0;

for (int k = 0; k < matrB.GetLength(0); k++)

{

temp += matrA[i, k] \* matrB[k, j];

}

resultMatr[i, j] = temp;

}

}

return resultMatr;

}

/\*\*

\* Знаходить максимальний елемент в матриці

\* @param matrA матриця для пошуку

\* @return максимальний елемент матриці

\*/

public static double MaxMatr(double[,] matrA)

{

double max = matrA[0, 0];

for (int i = 0; i < matrA.GetLength(0); i++)

{

for (int j = 0; j < matrA.GetLength(1); j++)

{

if (max < matrA[i, j])

{

max = matrA[i, j];

}

}

}

return max;

}

/\*\*

\* Перемножає вектор на матрицю

\* @param vectA вектор

\* @param matrB матрицю

\* @return добуток вектора і матриці

\*/

public static double[] MultVectMatr(double[] vectA, double[,] matrB)

{

double temp = 0;

double[] resultVect = new double[vectA.GetLength(0)];

for (int i = 0; i < matrB.GetLength(0); i++)

{

temp = 0;

for (int j = 0; j < resultVect.GetLength(0); j++)

{

temp += vectA[j] \* matrB[j, i];

}

resultVect[i] = temp;

}

return resultVect;

}

/\*\*

\* Сортує вектор за зростанням

\* @param vectA вектор для сортування

\* @return відсортований вектор

\* \*/

public static double[] sortVect(double[] vectA)

{

double temp = 0;

for (int i = 0; i < vectA.GetLength(0); i++)

{

for (int j = 0; j < (i - 1); j++)

{

if (vectA[j] > vectA[j + 1])

{

temp = vectA[j];

vectA[j] = vectA[j + 1];

vectA[j + 1] = temp;

}

}

}

return vectA;

}

/\*\*

\* Сортує матрицю за спаданням

\* @param matrA матриця для сортування

\* @return відсортовану матрицю

\* \*/

public static double[,] SortMatr(double[,] matrA)

{

double temp = 0;

for (int k = 0; k < matrA.GetLength(0); k++)

{

for (int i = 0; i < matrA.GetLength(0); i++)

{

for (int j = 0; j < (i - 1); j++)

{

if (matrA[k, j] < matrA[k, j + 1])

{

temp = matrA[k, j];

matrA[k, j] = matrA[k, j + 1];

matrA[k, j + 1] = temp;

}

}

}

}

return matrA;

}

/\*\*

\* Множить матрицю на число

\* @param a число

\* @param matrB матриця

\* @return результат множення

\* \*/

public static double[,] MultIntMatr(double a, double[,] matrB)

{

for (int i = 0; i < matrB.GetLength(0); i++)

{

for (int j = 0; j < matrB.GetLength(1); j++)

{

matrB[i, j] = matrB[i, j] \* a;

}

}

return matrB;

}

/\*\*

\* Зчитує матрицю

\* @param n кількість елементів

\* @param message повідомлення для відображення

\* @return зчитану матрицю

\*/

public static double[,] readMatr(int n, String message)

{

double[,] matr = new double[n, n];

Console.WriteLine(message);

for (int i = 0; i < matr.GetLength(0); i++)

{

for (int j = 0; j < matr.GetLength(1); j++)

{

Random rand = new Random();

matr[i, j] = rand.Next(25);

}

}

return matr;

}

/\*\*

\* Відображає матрицю

\* @param matr матриця для відображення

\* @param message повідомлення для відображення

\*/

public static void showMatr(double[,] matr, String message)

{

if (matr.GetLength(0) < 10)

{

Console.WriteLine(message);

Console.WriteLine(message);

for (int i = 0; i < matr.GetLength(0); i++)

{

for (int j = 0; j < matr.GetLength(1); j++)

{

Console.Write(matr[i, j] + " ");

}

Console.WriteLine();

}

}

Console.WriteLine();

}

/\*\*

\* Зчитує вектор

\* @param n кількість елементів вектора

\* @param message повідомлення для відображення

\* @return зчитаний вектор

\*/

public static double[] readVect(int n, String message)

{

double[] vect = new double[n];

Console.WriteLine(message);

for (int i = 0; i < vect.GetLength(0); i++)

{

Random rand = new Random();

vect[i] = rand.Next(25);

}

return vect;

}

/\*\*

\* Відображає вектор

\* @param vect вектор для відображення

\* @param message повідомлення для відображення

\*/

public static void showVect(double[] vect, String message)

{

if (vect.GetLength(0) < 10)

{

Console.WriteLine(message);

for (int i = 0; i < vect.GetLength(0); i++)

{

Console.Write(vect[i] + " ");

}

Console.WriteLine();

}

}

}

}

**Program.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

namespace ConsoleApplication1

{

/\*\*

\* Виконала

\* Студентка гр. ІО-43

\* Даніленко Наталія

\* 1.4 C=A + SORT(B)\*(MA\*ME)

\* 2.4 MG = MAX(MH)\*(MK \* ML)

\* 3.4 O = SORT(P)\* SORT(MR\*MS)

\*/

class Program

{

private static int n;

static void Main(string[] args)

{

Console.WriteLine("Main thread started ");

Console.WriteLine(" Enter n: ");

n = Convert.ToInt32(Console.ReadLine());

Console.WriteLine(" n = " +n);

Thread tf1 = new Thread(MF1);

Thread tf2 = new Thread(MF2);

Thread tf3 = new Thread(MF3);

tf1.Name = "TF1";

tf1.Priority = ThreadPriority.Highest;

tf2.Name = "TF2";

tf2.Priority = ThreadPriority.BelowNormal;

tf3.Name = "TF3";

tf3.Priority = ThreadPriority.Lowest;

tf1.Start();

tf2.Start();

tf3.Start();

tf1.Join();

tf2.Join();

tf3.Join();

Console.WriteLine(" Main thread finished ");

Console.Read();

}

/\*\*

\* F1: C=A + SORT(B)\*(MA\*ME)

\*/

public static void MF1()

{

double[] vectA;

double[] vectB;

double[,] matrA;

double[,] matrE;

double[] vectRes;

Console.WriteLine(Thread.CurrentThread.Name + " started ");

vectA = Tools.readVect(n, " Enter vector A for F1 ") ;

vectB = Tools.readVect(n, " Enter vector B for F1 ");

matrA = Tools.readMatr(n, " Enter matrix A for F1 ") ;

matrE = Tools.readMatr(n, " Enter matrix E for F1 ");

Thread.Sleep(1000);

vectRes = Tools.addVect(vectA, (Tools.MultVectMatr(Tools.sortVect(vectB), Tools.MultMatr(matrA, matrE))));

Tools.showVect(vectRes, "Result of F1");

Console.WriteLine(Thread.CurrentThread.Name + "finished");

}

/\*\*

\* F2: MG = MAX(MH)\*(MK \* ML)

\*/

public static void MF2()

{

double[,] matrH;

double[,] matrK;

double[,] matrL;

double[,] matrRes;

Console.WriteLine(Thread.CurrentThread.Name + "started");

matrH = Tools.readMatr(n, " Enter matrix H for F2 ");

matrK = Tools.readMatr(n, " Enter matrix K for F2 ");

matrL = Tools.readMatr(n, " Enter matrix L for F2 ");

Thread.Sleep(500);

matrRes = Tools.MultIntMatr(Tools.MaxMatr(matrH), Tools.MultMatr(matrK, matrL)) ;

Tools.showMatr(matrRes, "Result of F2");

Console.WriteLine(Thread.CurrentThread.Name + "finished");

}

/\*\*

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

\*/

public static void MF3()

{

double[] vectP;

double[,] matrR;

double[,] matrS;

double[] vectRes;

Console.WriteLine(Thread.CurrentThread.Name + "started");

vectP = Tools.readVect(n, " Enter vector P for F3 ");

matrR = Tools.readMatr(n, " Enter matrix R for F3 ");

matrS = Tools.readMatr(n, " Enter matrix S for F3 ");

Thread.Sleep(100);

vectRes = Tools.MultVectMatr(Tools.sortVect(vectP), Tools.SortMatr(Tools.MultMatr(matrR, matrS)));

Console.WriteLine(" Result of F3:" +vectRes);

Console.WriteLine(Thread.CurrentThread.Name + "finished");

}

}

}

**Результати виконання програми:**

