Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Лабораторна робота №4

з дисципліни «Паралельне програмування -1.

Основи паралельного програмування»

Тема: **Потоки в мові C#**

Виконав

студент групи ІП-31

Кобилинський Дмитро

Київ 2015

**Цель работы:** изучение средств языка C# для работы с потоками.

**Выполнение работы:** Разработать программу, содержащую п а р а л л е л ь н ы е 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.

**Код програми:**

**Program.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

namespace Lab4\_PP

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Input N");

String lineN = Console.ReadLine();

int n;

int.TryParse(lineN, out n);

Console.WriteLine("Input value");

String lineValue = Console.ReadLine();

int value;

int.TryParse(lineValue, out value);

Console.WriteLine();

Tasks tasks = new Tasks(n);

System.Int32 stackSizeT1 = 1000;

System.Int32 stackSizeT2 = 1000;

System.Int32 stackSizeT3 = 1000;

Thread t1 = new Thread(tasks.Task1, stackSizeT1);

Thread t2 = new Thread(tasks.Task2, stackSizeT2);

Thread t3 = new Thread(tasks.Task3, stackSizeT3);

t1.Name = "Thread 1 to F1";

t1.Priority = ThreadPriority.Normal;

t2.Name = "Thread 2 to F2";

t2.Priority = ThreadPriority.Normal;

t3.Name = "Thread 3 to F3";

t3.Priority = ThreadPriority.Normal;

t1.Start();

t2.Start();

t3.Start();

t3.Join();

t1.Join();

Console.ReadLine();

}

}

}

**Tasks.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

namespace Lab4\_PP

{

class Tasks:Data

{

public Tasks(int n) : base(n){}

public void Task1(){

Console.WriteLine("Task 1 started");

int []A,B,C,D;

int [,] MA,MD;

A = inputVector();

B = inputVector();

C = inputVector();

MA = inputMatrix();

MD = inputMatrix();

Console.WriteLine("Calculation T1...");

Thread.Sleep(1000);

D = multiplyVectorOnDigit(multiplyMatrixOnVector(multiplyMatrix(MA, MD), sumVector(B, C)), minVector(sumVector(A, B)));

outputVector(D);

Console.WriteLine("Task 1 finished");

}

public void Task2(){

Console.WriteLine("Task 2 started");

int[,] ML, MK, MO, MM;

ML = inputMatrix();

MK = inputMatrix();

MO = inputMatrix();

Console.WriteLine("Calculation T2... ");

Thread.Sleep(5000);

Console.WriteLine("F2: ");

MM = sumMatrix(ML, multiplyMatrix(MK, MO));

outputMatrix(MM);

Console.WriteLine("Task 2 finished");

}

public void Task3()

{

Console.WriteLine("Task 3 started");

int[] S, T, R;

int[,] MR, MS;

Thread.Sleep(0);

MR = inputMatrix();

MS = inputMatrix();

S = inputVector();

T = inputVector();

Console.WriteLine("Calculation T3...");

Thread.Sleep(6000);

R = multiplyMatrixOnVector(transpose(multiplyMatrix(MS, MR)), sortVector(sumVector(S, T)));

Console.WriteLine("F3: ");

outputVector(R);

Console.WriteLine("Task 3 finished");

}

}

}

**Data.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab4\_PP

{

class Data

{

int size ;

public Data(int n){

this.size = n;

}

/\*\*

\* Method for output matrix

\* @param matrix

\*/

public void outputMatrix(int[,] matrix){

for(int i=0;i<size;i++){

for(int j=0;j<size;j++){

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

}

Console.WriteLine();

}

Console.WriteLine();

}

/\*\*

\* Method for output vector

\* @param vector

\*/

public void outputVector(int[] vector){

for(int i=0;i<size;i++){

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

}

Console.WriteLine();

}

/\*\*

\* Function for input matrix

\* @return matrix

\*/

public int[,] inputMatrix(){

int[,] matrix = new int[size,size];

for(int i=0;i<size;i++){

for(int j=0;j<size;j++){

matrix[i,j] = 1;

}

}

return matrix;

}

/\*\*

\* Function for input vector

\* @return vector

\*/

public int[] inputVector(){

int[] vector = new int[size];

for(int i=0;i<size;i++){

vector[i] = 1;

}

return vector;

}

/\*\*

\* Function for addition

\* @param add1 - addition matrix

\* @param add2 - addition matrix

\* @return result matrix

\*/

public int[,] sumMatrix(int[,] add1,int[,] add2){

int[,] result = new int[size,size];

for (int i = 0; i < size; i++) {

for (int j = 0; j < size; j++) {

result[i,j] = add1[i,j] + add2[i,j];

}

}

return result;

}

/\*\*

\* Function for multiply matrix on vector

\* @param m1 - multiplier matrix

\* @param v1 - multiplier vector

\* @return result vector

\*/

public int[] multiplyMatrixOnVector(int[,] m1, int[] v1){

int[] R = new int[size];

for (int i = 0; i < size; i++) {

int s = 0;

for (int j = 0; j <size; j++) {

s += m1[i,j] \* v1[j];

}

R[i] = s;

}

return R;

}

/\*\*

\* Function for multiply matrix

\* @param m1 - multiplier matrix

\* @param m2 - multiplier matrix

\* @return result matrix

\*/

public int[,] multiplyMatrix(int[,] m1, int[,] m2){

int[,] result = new int[size,size];

for (int i = 0; i < size; i++) {

for (int j = 0; j < size; j++) {

int s = 0;

for (int k = 0; k < size; k++) {

s += m1[i,k] \* m2[k,j];

result[i,j] = s;

}

}

}

return result;

}

/\*\*

\* Function for multiply matrix on digit

\* @param m1 - multiplier matrix

\* @param digit - digit

\* @return

\*/

public int[,] multiplyMatrixOnDigit(int[,] m1,int digit){

int[,] result = new int[size,size];

for(int i=0;i<size;i++){

for(int j=0;j<size;j++) {

result[i,j]=digit\*m1[i,j];

}

}

return result;

}

/\*\*

\* Function for search min element in matrix

\* @param m1 - matrix for search

\* @return min element

\*/

public int minMatrix(int[,] m1){

int result = m1[0,0];

for(int i=0;i<size;i++){

for(int j=0;j<size;j++) {

if(result<m1[i,j]){

result=m1[i,j];

}

}

}

return result;

}

/\*\*

\* Function for transpose matrix

\* @param m1 - matrix

\* @return - transpose matrix

\*/

public int[,] transpose(int[,] m1){

int[,] result = new int[size,size];

for(int i=0;i<size;i++){

for(int j=0;j<size;j++){

result[i,j]=m1[j,i];

}

}

return result;

}

/\*\*

\* Function for addition vector

\* @param add1 - addition vector

\* @param add2 - addition vector

\* @return resutl vector

\*/

public int[] sumVector(int[] add1,int[] add2){

int[] result = new int[size];

for (int i = 0; i < size; i++) {

result[i] = add1[i] + add2[i];

}

return result;

}

/\*\*

\* Function for multiply vector on digit

\* @param v1 vector

\* @param digit digit

\* @return result vector

\*/

public int[] multiplyVectorOnDigit(int[] v1,int digit){

int[] result = new int[size];

for(int i=0;i<size;i++){

result[i]=digit\*v1[i];

}

return result;

}

/\*\*

\* Function for search min element of vector

\* @param v1 vector

\* @return min element of vector

\*/

public int minVector(int[] v1){

int result = v1[0];

for(int i=0;i<size;i++){

for(int j=0;j<size;j++) {

if(result<v1[i]){

result=v1[i];

}

}

}

return result;

}

/\*\*

\* Function for sorting vector

\* @param v - vector

\* @return sorted vector

\*/

public int[] sortVector(int[] v){

int k;

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

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

if(v[j]>v[j+1]){

k=v[j];

v[j]=v[j+1];

v[j+1]=k;

}

}

}

return v;

}

}

}