Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Основи паралельного програмування

Лабораторна робота №3

**«Потоки в мові C#»**

Виконала:

студентка групи ІВ-71

Молчанова В.С.

Перевірив:

Долголенко О.М.

Київ

2019 р.

**Завдання:**

F1: MC = MIN(A) \*(MA\*MD)

F2: MK = TRANS(MA)\*TRANS(MB\*MM) +MХ

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

**Лістинг програми:**

**Program.cs**

using System;

using System.Threading;

namespace Lab3

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Lab3 started");

Console.WriteLine();

var f1 = new F1();

var f2 = new F2();

var f3 = new F3();

var f1Tread = new Thread(f1.Run);

var f2Thread = new Thread(f2.Run);

var f3Thread = new Thread(f3.Run);

f1Tread.Name = "F1";

f2Thread.Name = "F2";

f3Thread.Name = "F3";

f1Tread.Priority = ThreadPriority.Lowest;

f2Thread.Priority = ThreadPriority.Normal;

f3Thread.Priority = ThreadPriority.Highest;

f1Tread.Start();

f2Thread.Start();

f3Thread.Start();

f1Tread.Join();

f2Thread.Join();

f3Thread.Join();

Console.WriteLine();

Console.WriteLine(F1.Function);

Console.WriteLine(f1.Result);

Console.WriteLine(F2.Function);

Console.WriteLine(f2.Result);

Console.WriteLine(F3.Function);

Console.WriteLine(f3.Result);

Console.WriteLine("Lab3 ended");

Console.ReadLine();

}

}

}

**Data.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Lab3

{

public static class Data

{

public static int Func1(Vector A, Vector B, Matrix MA, Matrix MB) => Max(MA \* MB) \* (A \* B);

public static Matrix Func2(Matrix MG, Matrix MH, Matrix MK) => Max(MG) \* (MH \* MK);

public static Vector Func3(Vector O, Vector P, Matrix MR, Matrix MS) => Sort(O + P) \* Transpose(MR \* MS);

public class Vector

{

public int Length;

public List<int> Array { get; }

public Vector(int n)

{

Array = new List<int>(new int[n]);

Length = n;

}

public Vector(List<int> vector)

{

Array = vector;

Length = vector.Count;

}

public void FillByOrder()

{

for (var i = 0; i < Array.Count; i++)

{

Array[i] = i;

}

}

public void FillWithOnes()

{

for (var i = 0; i < Array.Count; i++)

{

Array[i] = 1;

}

}

public override string ToString()

{

var result = "";

foreach (var i in Array)

{

result += $"{i, 3} ";

}

return result+"\n";

}

public int this[int index]

{

get => Array[index];

set => Array[index] = value;

}

public static Vector operator + (Vector vector1, Vector vector2)

{

var result = new Vector(vector1.Length);

for (var i = 0; i < vector1.Length; i++)

{

result[i] = vector1[i] + vector2[i];

}

return result;

}

public static Vector operator \* (int a, Vector vector)

{

return new Vector(vector.Array.Select(t => t \* a).ToList());

}

public static Vector operator \* (Vector vector, int a) => a \* vector;

public static int operator \* (Vector vector1, Vector vector2)

{

var result = 0;

for (var i = 0; i < vector1.Length; i++)

{

result += vector1[i] \* vector2[i];

}

return result;

}

}

public class Matrix

{

public int RowNumber;

public int ColumnNumber;

public List<Vector> Array { get; }

public Matrix(int n, int m)

{

RowNumber = n;

ColumnNumber = m;

Array = new List<Vector>(n);

for (var i = 0; i < n; i++)

{

Array.Add(new Vector(m));

}

}

public Matrix(List<Vector> array)

{

Array = array;

RowNumber = array.Count;

ColumnNumber = array.First().Length;

}

public void FillByOrder()

{

for(var i = 0; i < RowNumber;i++)

for (var j = 0; j < ColumnNumber; j++)

{

Array[i][j] = ColumnNumber \* i + j;

}

}

public void FillWithOnes()

{

for (var i = 0; i < RowNumber; i++)

for (var j = 0; j < ColumnNumber; j++)

{

Array[i][j] = 1;

}

}

public override string ToString()

{

var result = "";

foreach (var vector in Array)

{

result += $"{vector}";

}

return result;

}

public Vector this[int index]

{

get => Array[index];

set => Array[index] = value;

}

public static Matrix operator + (Matrix matrix1, Matrix matrix2)

{

var result = new Matrix(matrix1.RowNumber, matrix1.ColumnNumber);

for (var i = 0; i < matrix1.RowNumber; i++)

{

result[i] = matrix1[i] + matrix2[i];

}

return result;

}

public static Matrix operator -(Matrix matrix1, Matrix matrix2) => matrix1 + (-1 \* matrix2);

public static Matrix operator \* (Matrix matrix1, Matrix matrix2)

{

var result = new Matrix(matrix1.RowNumber, matrix2.ColumnNumber);

var transposedMatrix2 = Transpose(matrix2);

for (var i = 0; i < matrix1.RowNumber; i++)

for (var j = 0; j < transposedMatrix2.RowNumber; j++)

{

result[i][j] = matrix1[i] \* transposedMatrix2[j];

}

return result;

}

public static Vector operator \* (Vector vector, Matrix matrix)

{

var matrix1 = new Matrix(new List<Vector>{vector});

var result = matrix1 \* matrix;

return result[0];

}

public static Matrix operator \* (int a, Matrix matrix)

{

return new Matrix(matrix.Array.Select(v => a \* v).ToList());

}

public static Matrix operator \*(Matrix matrix, int a) => a \* matrix;

}

public static Vector Sort(Vector v)

{

var result = new Vector(v.Array);

result.Array.Sort();

return result;

}

public static Matrix Sort(Matrix m)

{

var result = new Matrix(m.Array);

foreach (var vector in result.Array)

{

vector.Array.Sort();

}

return result;

}

public static int Max(Vector v) => v.Array.Max();

public static int Min(Vector v) => v.Array.Min();

public static int Max(Matrix m) => m.Array.Select(v => v.Array.Max()).Max();

public static int Min(Matrix m) => m.Array.Select(v => v.Array.Min()).Min();

public static Matrix Transpose(Matrix m)

{

var result = new Matrix(m.ColumnNumber, m.RowNumber);

for (var i = 0; i < m.ColumnNumber; i++)

for (var j = 0; j < m.RowNumber; j++)

{

result[i][j] = m.Array[j][i];

}

return result;

}

}

}

**F1.cs**

using System;

using System.Collections.Generic;

using System.Text;

using static Lab3.Data;

namespace Lab3

{

public class F1

{

public string Result { get; private set; }

public void Run()

{

Console.WriteLine("Task 1 started");

var A = new Vector(5);

var B = new Vector(5);

A.FillWithOnes();

B.FillWithOnes();

var MA = new Matrix(5, 3);

var MB = new Matrix(3, 5);

MA.FillWithOnes();

MB.FillWithOnes();

Result = Func1(A, B, MA, MB) + "\n";

Console.WriteLine("Task 1 ended");

}

}

}

**F2.cs**

using System;

using System.Collections.Generic;

using System.Text;

using static Lab3.Data;

namespace Lab3

{

public class F2

{

public Matrix Result { get; private set; }

public void Run()

{

Console.WriteLine("Task 2 started");

var MG = new Matrix(6, 6);

var MH = new Matrix(5, 4);

var MK = new Matrix(4, 5);

MG.FillWithOnes();

MH.FillWithOnes();

MK.FillWithOnes();

Result = Func2(MG, MH, MK);

Console.WriteLine("Task 2 ended");

}

}

}

**F3.cs**

using System;

using System.Collections.Generic;

using System.Text;

using static Lab3.Data;

namespace Lab3

{

public class F3

{

public Vector Result { get; private set; }

public void Run()

{

Console.WriteLine("Task 3 started");

var O = new Vector(5);

var P = new Vector(5);

O.FillWithOnes();

P.FillWithOnes();

var MR = new Matrix(5, 3);

var MS = new Matrix(3, 5);

MR.FillWithOnes();

MS.FillWithOnes();

Result = Func3(O, P, MR, MS);

Console.WriteLine("Task 3 ended");

}

}

}

**Результат роботи:**

Lab3 started

Task 3 started

Task 2 started

Task 1 started

Task 3 ended

Task 2 ended

Task 1 ended

F1:

15

F2:

4 4 4 4 4

4 4 4 4 4

4 4 4 4 4

4 4 4 4 4

4 4 4 4 4

F3:

30 30 30 30 30

Lab3 ended