НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №3

з дисципліни **«**Програмування паралельних комп’ютерних систем**»**

Виконав:

студент 3 курсу

ФІОТ гр. ІО-34

Кривоносов Олексій

Перевірив:

Корочкін О. В.

Київ – 2016 р.

Тема: Програмування для комп’ютерних систем зі спільною пам’яттю. Мова C#

Розробити програму для розв’язання в ПКС зі СП (структура на рис. Рисунок 1) математичної задачі: A=(B\*C)\*Z+α\*E\*(MX\*MZ)

Бібліотека: C#.

Засоби організації взаємодії: примітиви синхронізації (Semaphore, Mutex, Monitor, Events,Lock), volatile-змінні.



Структурна схема ПКС

**Виконання роботи:**

**Етап 1. Побудова паралельного алгоритму**

1. bi = Bн\*Cн  (i = 1..P)
2. b= b+bi (i = 1..P)
3. AН = b\*ZH + α\*E\*(MX\*MZH)

Спільний ресурс: α, β, Z, MO

**Етап 2. Розроблення алгоритмів роботи кожного процесу**

|  |  |
| --- | --- |
| **Задача Т1** | ТС, КД |
| 1. Введення B, MZ |  |
| 2. Сигнал задачам Т2, Т3, Т4, T5, T6 про введення B, MZ | S2,3,4,5,6 – 1 |
| 3. Чекати на введення C, Z у задачі Т3 | W3 – 1 |
| 4. Чекати на введення α, E, MX у задачі T4 | W4 – 2 |
| 5. Обчислення: b1 = Bн\*Cн |  |
| 6. Обчислення: b = b + b1 | КД |
| 7. Сигнал задачам Т2, Т3, Т4, T5,T6 про обчислення b | S2,3,4,5,6 – 1 |
| 8. Чекати на завершення обчислень b в Т2 | W2– 3 |
| 9. Чекати на завершення обчислень b в Т3 | W3 – 4 |
| 10. Чекати на завершення обчислень b в Т4 | W4 – 5 |
| 11. Чекати на завершення обчислень b в Т5 | W5 – 6 |
| 12. Чекати на завершення обчислень b в Т6 | W6 – 7 |
| 13. Копіювання: E1=E | КД |
| 14. Копіювання: МХ1 := МХ | КД |
| 15. Копіювання: α1= α | КД |
| 16. Копіювання: b1 := b | КД |
| 17. Обчислення: AН = b1\*ZH + α1\*E1\*(MX1\*MZH) |  |
| 18. Чекати на завершення обчислень A в T2, T3, T4, Т5, Т6 | W2,3,4,5,6 – 8 |
| 19. Виведення A |  |
| **Задача T2** | ТС, КД |
| 1. Чекати на введення B, MZ у задачі T1 | W1 – 1 |
| 2. Чекати на введення C, Z у задачі T3 | W3 – 2 |
| 3. Чекати на введення α, E, MX у задачі T4 | W4 – 3 |
| 4. Обчислення: b2 = Bн\*Cн |  |
| 5. Обчислення: b = b + b2 | КД |
| 6. Сигнал задачам Т1, Т3, Т4, T5,T6 про обчислення b | S1,3,4,5,6 – 1 |
| 7. Чекати на завершення обчислень b в Т1 | W1– 4 |
| 8. Чекати на завершення обчислень b в Т3 | W3– 5 |
| 9. Чекати на завершення обчислень b в Т4 | W4– 6 |
| 10. Чекати на завершення обчислень b в Т5 | W5– 7 |
| 11. Чекати на завершення обчислень b в Т6 | W6– 8 |
| 12. Копіювання: E2=E | КД |
| 13. Копіювання: МХ2 := МХ | КД |
| 14. Копіювання: α2= α | КД |
| 15. Копіювання: b2 := b | КД |
| 16. Обчислення: AН = b2\*ZH + α2\*E2\*(MX2\*MZH) |  |
| 17. Сигнал T1 про завершення обчислень A | S1 – 2 |
| **Задача Т3** | ТС, КД |
| 1. Введення C, Z |  |
| 2. Сигнал задачам Т1, Т2, Т4, T5,T6 про введення C, Z | S1,2,4,5,6 – 1 |
| 3. Чекати на введення B, MZ у задачі T1 | W1 – 1 |
| 4. Чекати на введення α, E, MX у задачі T4 | W4 – 2 |
| 5. Обчислення: b3 = Bн\*Cн |  |
| 6. Обчислення: b = b + b3 | КД |
| 7. Сигнал задачам Т1, Т2, Т4, T5,T6 про обчислення b | S1,2,4,5,6 – 2 |
| 8. Чекати на завершення обчислень b в Т1 | W1 – 3 |
| 9. Чекати на завершення обчислень b в Т2 | W2– 4 |
| 10. Чекати на завершення обчислень b в Т4 | W4– 5 |
| 11. Чекати на завершення обчислень b в Т5 | W5– 6 |
| 12. Чекати на завершення обчислень b в Т6 | W6– 7 |
| 13. Копіювання:E3=E | КД |
| 14. Копіювання: МХ3 := МХ | КД |
| 15. Копіювання: α3= α | КД |
| 16. Копіювання: b3 := b | КД |
| 17. Обчислення: AН = b3\*Z3 + α3\*E3\*(MX3\*MZH) |  |
| 18. Сигнал T1 про завершення обчислень A | S1 – 3 |
| **Задача Т4** | ТС, КД |
| 1. Введення α, E, MX |  |
| 2. Сигнал задачам T1, T2, T3, T5,T6 про введення α, E, MX | S1,2,3,5,6 – 1 |
| 3. Чекати на введення B, MZ у задачі T1 | W1 – 1 |
| 4. Чекати на введення C, Z у задачі T3 | W3 – 2 |
| 5. Обчислення: b4 = Bн\*Cн |  |
| 6. Обчислення: b = b + b4 | КД |
| 7. Сигнал задачам Т1, Т2, Т3, T5,T6 про обчислення b | S1,2,3,5,6 – 2 |
| 8. Чекати на завершення обчислень b в Т1 | W1– 3 |
| 9. Чекати на завершення обчислень b в Т2 | W2– 4 |
| 10. Чекати на завершення обчислень b в Т3 | W3– 5 |
| 11. Чекати на завершення обчислень b в Т5 | W5–6 |
| 12. Чекати на завершення обчислень b в Т6 | W6– 7 |
| 13. Копіювання: E4=E | КД |
| 14. Копіювання: МХ4 := МХ | КД |
| 15. Копіювання: α4= α | КД |
| 16. Копіювання: b4 := b | КД |
| 17. Обчислення: AН = b4\*ZH+ α4\*E4\*(MX4\*MZH) |  |
| 18. Сигнал T1 про завершення обчислень A | S1 – 3 |
| **Задача T5** | ТС, КД |
| 1. Чекати на введення B, MZ у задачі T1 | W1 – 1 |
| 2. Чекати на введення C, Z у задачі T3 | W3 – 2 |
| 3. Чекати на введення α, E, MX у задачі T4 | W4 – 3 |
| 4. Обчислення: b5 = Bн\*Cн |  |
| 5. Обчислення: b = b + b5 | КД |
| 6. Сигнал задачам Т1,T2, Т3, Т4,T6 про обчислення b | S1,2,3,4,6 – 1 |
| 7. Чекати на завершення обчислень b в Т1 | W1– 4 |
| 8. Чекати на завершення обчислень b в Т2 | W2– 5 |
| 9. Чекати на завершення обчислень b в Т3 | W3– 6 |
| 10. Чекати на завершення обчислень b в Т4 | W4– 7 |
| 11. Чекати на завершення обчислень b в Т6 | W6– 8 |
| 12. Копіювання: E5=E | КД |
| 13. Копіювання: МХ5 := МХ | КД |
| 14. Копіювання: α5= α | КД |
| 15. Копіювання: b5 := b | КД |
| 16. Обчислення: AН = b2\*ZH + α2\*E2\*(MX2\*MZH) |  |
| 17. Сигнал T1 про завершення обчислень A | S1 – 2 |
| **Задача T6** | ТС, КД |
| 1. Чекати на введення B, MZ у задачі T1 | W1 – 1 |
| 2. Чекати на введення C, Z у задачі T3 | W3 – 2 |
| 3. Чекати на введення α, E, MX у задачі T4 | W4 – 3 |
| 4. Обчислення: b6 = Bн\*Cн |  |
| 5. Обчислення: b = b + b6 | КД |
| 6. Сигнал задачам Т1, Т2, Т3, Т4, Т5 про обчислення b | S1,2,3,4,5 – 1 |
| 7. Чекати на завершення обчислень b в Т1 | W1– 4 |
| 8. Чекати на завершення обчислень b в Т2 | W2– 5 |
| 9. Чекати на завершення обчислень b в Т3 | W3– 6 |
| 10. Чекати на завершення обчислень b в Т4 | W4– 7 |
| 11. Чекати на завершення обчислень b в Т5 | W5– 8 |
| 12. Копіювання: E6=E | КД |
| 13. Копіювання: МХ6 := МХ | КД |
| 14. Копіювання: α6= α | КД |
| 15. Копіювання: b6 := b | КД |
| 16. Обчислення: AН = b6\*ZH + α6\*E2\*(MX6\*MZH) |  |
| 17. Сигнал T1 про завершення обчислень A | S1 – 2 |

**Етап 3. Розроблення структурної схеми взаємодії задач**

На структурній схемі взаємодії задач уведено такі умовні позначення

* Evn1\_23456 – подія для синхронізації із завершенням вводу в T1
* Evn3\_12456 – подія для синхронізації із завершенням вводу в T3
* Evn4\_12356 – подія для синхронізації із завершенням вводу в T4
* EvnB1\_23456 – подія для синхронізації із завершенням обчислення b1 в Т1
* EvnB2\_13456 – подія для синхронізації із завершенням обчислення b2 в Т2
* EvnB3\_12456 – подія для синхронізації із завершенням обчислення b3 в Т3
* EvnB4\_12356 – подія для синхронізації із завершенням обчислення b4 в Т4
* EvnB5\_12346– подія для синхронізації із завершенням обчислення b5 в Т5
* EvnB6\_12345– подія для синхронізації із завершенням обчислення b6 в Т6
* Sem\_A2 – для синхронізації обчислень A2
* Sem\_A3 – для синхронізації обчислень A3
* Sem\_A4 – для синхронізації обчислень A4
* Sem\_A5 – для синхронізації обчислень A5
* Sem\_A6 – для синхронізації обчислень A6
* volatile – ключове слово для доступу до спільного ресурсу α
* mutex\_MX – мютекс для доступу до спільного ресурсу MX
* lock\_E – замок для доступу до спільного ресурсу E
* monitor\_b – монітор для доступу до спільного ресурсу b

Етап 4. Розроблення програми

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading;

/\*\*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* \*

\* Parallel and Distributed Computing \*

\* Laboratory work #3. C# \*

\* \*

\* Task: A=(B\*C)\*Z+α\*E\*(MX\*MZ) \*

\* \*

\* \*

\* @author Krivonosov Oleksii \*

\* @group IO-34 \*

\* @date 24.03.16 \*

\* File Program.cs \*

\* \*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\*/

namespace lab3

{

class Program : Operations

{

public const int N = 6;

public const int P = 6;

public const int H = N / P;

public static EventWaitHandle Evn1\_23456;

public static EventWaitHandle Evn3\_12456;

public static EventWaitHandle Evn4\_12356;

public static EventWaitHandle EvnB1\_23456;

public static EventWaitHandle EvnB2\_13456;

public static EventWaitHandle EvnB3\_12456;

public static EventWaitHandle EvnB4\_12356;

public static EventWaitHandle EvnB5\_12346;

public static EventWaitHandle EvnB6\_12345;

public static object lock\_E = new object();

public static object monitor\_b = new object();

public static Mutex mutex\_MX = new Mutex(false);

public static Semaphore Sem\_A2, Sem\_A3, Sem\_A4, Sem\_A5, Sem\_A6;

public static volatile int alpha;

public static int b = 0;

public static Vector Z = new Vector(N);

public static Vector B = new Vector(N);

public static Vector C = new Vector(N);

public static Vector E = new Vector(N);

public static Vector A = new Vector(N);

public static Matrix MX = new Matrix(N);

public static Matrix MZ = new Matrix(N);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Задача Т1 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public static void T1()

{

Console.WriteLine("Task 1 started");

/\* 1. Введення B, MZ\*/

B = inputVector(N, 1);

MZ = inputMatrix(N, 1);

/\* 2. Сигнал задачам Т2, Т3, Т4, T5, T6 про введення B, MZ\*/

Evn1\_23456.Set();

/\* 3. Чекати на введення C, Z у задачі Т3 \*/

Evn3\_12456.WaitOne();

/\* 4. Чекати на введення α, E, MX у задачі T4 \*/

Evn4\_12356.WaitOne();

/\* 5. Обчислення: b1 = Bн\*Cн \*/

int b1 = 0;

for (int i = 0; i < H; i++)

{

b1 += B.get(i) \* C.get(i);

}

/\* 6. Обчислення: b = b + b1 \*/

try

{

Monitor.Enter(monitor\_b);

b = b + b1;

}

finally

{

Monitor.Exit(monitor\_b);

}

/\* 7. Сигнал задачам Т2, Т3, Т4, T5,T6 про обчислення b \*/

EvnB1\_23456.Set();

/\* 8. Чекати на завершення обчислень b в Т2 \*/

EvnB2\_13456.WaitOne();

/\* 9. Чекати на завершення обчислень b в Т3 \*/

EvnB3\_12456.WaitOne();

/\* 10. Чекати на завершення обчислень b в Т4 \*/

EvnB4\_12356.WaitOne();

/\* 11. Чекати на завершення обчислень b в Т5 \*/

EvnB5\_12346.WaitOne();

/\* 12. Чекати на завершення обчислень b в Т6 \*/

EvnB6\_12345.WaitOne();

/\* 13. Копіювання: E1=E \*/

Vector E1 = new Vector(N);

lock (lock\_E) {

E1 = E;

}

/\*14. Копіювання: МХ1 := МХ \*/

Matrix MX1 = new Matrix(N);

mutex\_MX.WaitOne();

MX1 = MX;

mutex\_MX.ReleaseMutex();

/\* 15. Копіювання: α1= α \*/

int alpha1 = alpha;

/\* 16. Копіювання: b1 := b \*/

try

{

Monitor.Enter(monitor\_b);

b1 = b;

}

finally {

Monitor.Exit(monitor\_b);

}

/\* 17. Обчислення: AН = b1\*ZH + α1\*E1\*(MX1\*MZH) \*/

Vector buf = new Vector(H);

buf = add(mult(b1, Z, 0, H), mult(alpha1, mult(E1, mult(MX1, MZ, 0, H), 0, H), 0, H), 0, H);

for (int i = 0; i < H; i++)

{

A.set(i, buf.get(i));

}

/\* 18. Чекати на завершення обчислень A в T2, T3, T4, Т5, Т6 \*/

Sem\_A2.WaitOne();

Sem\_A3.WaitOne();

Sem\_A4.WaitOne();

Sem\_A5.WaitOne();

Sem\_A6.WaitOne();

/\* 19. Виведення A \*/

outputVector(A);

Console.WriteLine("Task 1 finished");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Задача Т2 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public static void T2()

{

Console.WriteLine("Task 2 started");

/\* 1. Чекати на введення B, MZ у задачі T1\*/

Evn1\_23456.WaitOne();

/\* 2. Чекати на введення C, Z у задачі Т3 \*/

Evn3\_12456.WaitOne();

/\* 3. Чекати на введення α, E, MX у задачі T4 \*/

Evn4\_12356.WaitOne();

/\* 4. Обчислення: b2 = Bн\*Cн \*/

int b2 = 0;

for (int i = H; i < H\*2; i++)

{

b2 += B.get(i) \* C.get(i);

}

/\* 5. Обчислення: b = b + b2 \*/

try

{

Monitor.Enter(monitor\_b);

b = b + b2;

}

finally

{

Monitor.Exit(monitor\_b);

}

/\* 6. Сигнал задачам Т1, Т3, Т4, T5,T6 про обчислення b \*/

EvnB2\_13456.Set();

/\* 7. Чекати на завершення обчислень b в Т1 \*/

EvnB1\_23456.WaitOne();

/\* 8. Чекати на завершення обчислень b в Т3 \*/

EvnB3\_12456.WaitOne();

/\* 9. Чекати на завершення обчислень b в Т4 \*/

EvnB4\_12356.WaitOne();

/\* 10. Чекати на завершення обчислень b в Т5 \*/

EvnB5\_12346.WaitOne();

/\* 11. Чекати на завершення обчислень b в Т6 \*/

EvnB6\_12345.WaitOne();

/\* 12. Копіювання: E2=E \*/

Vector E2 = new Vector(N);

lock (lock\_E)

{

E2 = E;

}

/\*13. Копіювання: МХ2 := МХ \*/

Matrix MX2 = new Matrix(N);

mutex\_MX.WaitOne();

MX2 = MX;

mutex\_MX.ReleaseMutex();

/\* 14. Копіювання: α2= α \*/

int alpha2 = alpha;

/\* 15. Копіювання: b1 := b \*/

try

{

Monitor.Enter(monitor\_b);

b2 = b;

}

finally

{

Monitor.Exit(monitor\_b);

}

/\* 16. Обчислення: AН = b2\*ZH + α2\*E2\*(MX2\*MZH) \*/

Vector buf = new Vector(H);

buf = add(mult(b2, Z, H, 2\*H), mult(alpha2, mult(E2, mult(MX2, MZ, H, 2\*H), H, 2\*H), H, 2\*H), H, 2\*H);

for (int i = H; i < H\*2; i++)

{

A.set(i, buf.get(i));

}

/\* 17. Сигнал T1 про завершення обчислень A \*/

Sem\_A2.Release();

Console.WriteLine("Task 2 finished");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Задача Т3 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public static void T3()

{

Console.WriteLine("Task 3 started");

/\* 1. Введення C, Z\*/

C = inputVector(N, 1);

Z = inputVector(N, 1);

/\* 2. Сигнал задачам Т1, Т2, Т4, T5, T6 про введення B, MZ\*/

Evn3\_12456.Set();

/\* 3. Чекати на введення B, MZ у задачі Т1 \*/

Evn1\_23456.WaitOne();

/\* 4. Чекати на введення α, E, MX у задачі T4 \*/

Evn4\_12356.WaitOne();

/\* 5. Обчислення: b3 = Bн\*Cн \*/

int b3 = 0;

for (int i = 2\*H; i < H\*3; i++)

{

b3 += B.get(i) \* C.get(i);

}

/\* 6. Обчислення: b = b + b1 \*/

try

{

Monitor.Enter(monitor\_b);

b = b + b3;

}

finally

{

Monitor.Exit(monitor\_b);

}

/\* 7. Сигнал задачам Т1, Т2, Т4, T5,T6 про обчислення b \*/

EvnB3\_12456.Set();

/\* 8. Чекати на завершення обчислень b в Т1 \*/

EvnB1\_23456.WaitOne();

/\* 9. Чекати на завершення обчислень b в Т2 \*/

EvnB2\_13456.WaitOne();

/\* 10. Чекати на завершення обчислень b в Т4 \*/

EvnB4\_12356.WaitOne();

/\* 11. Чекати на завершення обчислень b в Т5 \*/

EvnB5\_12346.WaitOne();

/\* 12. Чекати на завершення обчислень b в Т6 \*/

EvnB6\_12345.WaitOne();

/\* 13. Копіювання: E3=E \*/

Vector E3 = new Vector(N);

lock (lock\_E)

{

E3 = E;

}

/\*14. Копіювання: МХ3 := МХ \*/

Matrix MX3 = new Matrix(N);

mutex\_MX.WaitOne();

MX3 = MX;

mutex\_MX.ReleaseMutex();

/\* 15. Копіювання: α3= α \*/

int alpha3 = alpha;

/\* 16. Копіювання: b3 := b \*/

try

{

Monitor.Enter(monitor\_b);

b3 = b;

}

finally

{

Monitor.Exit(monitor\_b);

}

/\* 17. Обчислення: AН = b3\*ZH + α3\*E3\*(MX3\*MZH) \*/

Vector buf = new Vector(H);

buf = add(mult(b3, Z, 2 \* H, 3 \* H), mult(alpha3, mult(E3, mult(MX3, MZ, 2 \* H, 3 \* H), 2 \* H, 3 \* H), 2 \* H, 3 \* H), 2 \* H, 3 \* H);

for (int i = 2\*H; i < H\*3; i++)

{

A.set(i, buf.get(i));

}

/\* 18. Сигнал T1 про завершення обчислень A \*/

Sem\_A3.Release();

Console.WriteLine("Task 3 finished");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Задача Т4 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public static void T4()

{

Console.WriteLine("Task 4 started");

/\* 1. Введення α, E, MX\*/

E = inputVector(N, 1);

MX = inputMatrix(N, 1);

alpha = 1;

/\* 2. Сигнал задачам Т1, Т2, Т3, T5, T6 про введення B, MZ\*/

Evn4\_12356.Set();

/\* 3. Чекати на введення B, MZ у задачі Т1 \*/

Evn1\_23456.WaitOne();

/\* 4. Чекати на введення C, Z у задачі T3 \*/

Evn3\_12456.WaitOne();

/\* 5. Обчислення: b4 = Bн\*Cн \*/

int b4 = 0;

for (int i = 3 \* H; i < H \* 4; i++)

{

b4 += B.get(i) \* C.get(i);

}

/\* 6. Обчислення: b = b + b1 \*/

try

{

Monitor.Enter(monitor\_b);

b = b + b4;

}

finally

{

Monitor.Exit(monitor\_b);

}

/\* 7. Сигнал задачам Т1, Т2, Т3, T5,T6 про обчислення b \*/

EvnB4\_12356.Set();

/\* 8. Чекати на завершення обчислень b в Т1 \*/

EvnB1\_23456.WaitOne();

/\* 9. Чекати на завершення обчислень b в Т2 \*/

EvnB2\_13456.WaitOne();

/\* 10. Чекати на завершення обчислень b в Т3 \*/

EvnB3\_12456.WaitOne();

/\* 11. Чекати на завершення обчислень b в Т5 \*/

EvnB5\_12346.WaitOne();

/\* 12. Чекати на завершення обчислень b в Т6 \*/

EvnB6\_12345.WaitOne();

/\* 13. Копіювання: E4=E \*/

Vector E4 = new Vector(N);

lock (lock\_E)

{

E4 = E;

}

/\*14. Копіювання: МХ4 := МХ \*/

Matrix MX4 = new Matrix(N);

mutex\_MX.WaitOne();

MX4 = MX;

mutex\_MX.ReleaseMutex();

/\* 15. Копіювання: α3= α \*/

int alpha3 = alpha;

/\* 16. Копіювання: b3 := b \*/

try

{

Monitor.Enter(monitor\_b);

b4 = b;

}

finally

{

Monitor.Exit(monitor\_b);

}

/\* 17. Обчислення: AН = b4\*ZH + α4\*E4\*(MX4\*MZH) \*/

Vector buf = new Vector(H);

buf = add(mult(b4, Z, 3 \* H, 4 \* H), mult(alpha3, mult(E4, mult(MX4, MZ, 3 \* H, 4 \* H), 3 \* H, 4 \* H), 3 \* H, 4 \* H), 3 \* H, 4 \* H);

for (int i = 3 \* H; i < H \* 4; i++)

{

A.set(i, buf.get(i));

}

/\* 18. Сигнал T1 про завершення обчислень A \*/

Sem\_A4.Release();

Console.WriteLine("Task 4 finished");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Задача Т5 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public static void T5()

{

Console.WriteLine("Task 5 started");

/\* 1. Чекати на введення B, MZ у задачі T1\*/

Evn1\_23456.WaitOne();

/\* 2. Чекати на введення C, Z у задачі Т3 \*/

Evn3\_12456.WaitOne();

/\* 3. Чекати на введення α, E, MX у задачі T4 \*/

Evn4\_12356.WaitOne();

/\* 4. Обчислення: b5 = Bн\*Cн \*/

int b5 = 0;

for (int i = H\*4; i < H \* 5; i++)

{

b5 += B.get(i) \* C.get(i);

}

/\* 5. Обчислення: b = b + b5 \*/

try

{

Monitor.Enter(monitor\_b);

b = b + b5;

}

finally

{

Monitor.Exit(monitor\_b);

}

/\* 6. Сигнал задачам Т1, Т2, Т3, T4,T6 про обчислення b \*/

EvnB5\_12346.Set();

/\* 7. Чекати на завершення обчислень b в Т1 \*/

EvnB1\_23456.WaitOne();

/\* 8. Чекати на завершення обчислень b в Т2 \*/

EvnB2\_13456.WaitOne();

/\* 9. Чекати на завершення обчислень b в Т3 \*/

EvnB3\_12456.WaitOne();

/\* 10. Чекати на завершення обчислень b в Т4 \*/

EvnB4\_12356.WaitOne();

/\* 11. Чекати на завершення обчислень b в Т6 \*/

EvnB6\_12345.WaitOne();

/\* 12. Копіювання: E5=E \*/

Vector E5 = new Vector(N);

lock (lock\_E)

{

E5 = E;

}

/\*13. Копіювання: МХ5 := МХ \*/

Matrix MX5 = new Matrix(N);

mutex\_MX.WaitOne();

MX5 = MX;

mutex\_MX.ReleaseMutex();

/\* 14. Копіювання: α5= α \*/

int alpha5 = alpha;

/\* 15. Копіювання: b5 := b \*/

try

{

Monitor.Enter(monitor\_b);

b5 = b;

}

finally

{

Monitor.Exit(monitor\_b);

}

/\* 16. Обчислення: AН = b5\*ZH + α5\*E5\*(MX5\*MZH) \*/

Vector buf = new Vector(H);

buf = add(mult(b5, Z, 4 \* H, 5 \* H), mult(alpha5, mult(E5, mult(MX5, MZ, 4 \* H, 5 \* H), 4 \* H, 5 \* H), 4 \* H, 5 \* H), 4 \* H, 5 \* H);

for (int i = 4\*H; i < H \* 5; i++)

{

A.set(i, buf.get(i));

}

/\* 17. Сигнал T1 про завершення обчислень A \*/

Sem\_A5.Release();

Console.WriteLine("Task 5 finished");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Задача Т6 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public static void T6()

{

Console.WriteLine("Task 6 started");

/\* 1. Чекати на введення B, MZ у задачі T1\*/

Evn1\_23456.WaitOne();

/\* 2. Чекати на введення C, Z у задачі Т3 \*/

Evn3\_12456.WaitOne();

/\* 3. Чекати на введення α, E, MX у задачі T4 \*/

Evn4\_12356.WaitOne();

/\* 4. Обчислення: b6 = Bн\*Cн \*/

int b6 = 0;

for (int i = H \* 5; i < H \* 6; i++)

{

b6 += B.get(i) \* C.get(i);

}

/\* 5. Обчислення: b = b + b6 \*/

try

{

Monitor.Enter(monitor\_b);

b = b + b6;

}

finally

{

Monitor.Exit(monitor\_b);

}

/\* 6. Сигнал задачам Т1, Т2, Т3, T4,T5 про обчислення b \*/

EvnB6\_12345.Set();

/\* 7. Чекати на завершення обчислень b в Т1 \*/

EvnB1\_23456.WaitOne();

/\* 8. Чекати на завершення обчислень b в Т2 \*/

EvnB2\_13456.WaitOne();

/\* 9. Чекати на завершення обчислень b в Т3 \*/

EvnB3\_12456.WaitOne();

/\* 10. Чекати на завершення обчислень b в Т4 \*/

EvnB4\_12356.WaitOne();

/\* 11. Чекати на завершення обчислень b в Т6 \*/

EvnB5\_12346.WaitOne();

/\* 12. Копіювання: E6=E \*/

Vector E6 = new Vector(N);

lock (lock\_E)

{

E6 = E;

}

/\*13. Копіювання: МХ6 := МХ \*/

Matrix MX6 = new Matrix(N);

mutex\_MX.WaitOne();

MX6 = MX;

mutex\_MX.ReleaseMutex();

/\* 14. Копіювання: α6 = α \*/

int alpha6 = alpha;

/\* 15. Копіювання: b6 := b \*/

try

{

Monitor.Enter(monitor\_b);

b6 = b;

}

finally

{

Monitor.Exit(monitor\_b);

}

/\* 16. Обчислення: AН = b6\*ZH + α6\*E6\*(MX6\*MZH) \*/

Vector buf = new Vector(H);

buf = add(mult(b6, Z, 5 \* H, 6 \* H), mult(alpha6, mult(E6, mult(MX6, MZ, 5 \* H, 6 \* H), 5 \* H, 6 \* H), 5 \* H, 6 \* H), 5 \* H, 6 \* H);

for (int i = 5 \* H; i < H \* 6; i++)

{

A.set(i, buf.get(i));

}

/\* 17. Сигнал T1 про завершення обчислень A \*/

Sem\_A6.Release();

Console.WriteLine("Task 6 finished");

}

static void Main(string[] args)

{

System.Console.WriteLine("Lab3 started");

Evn1\_23456 = new ManualResetEvent(false);

Evn3\_12456 = new ManualResetEvent(false);

Evn4\_12356 = new ManualResetEvent(false);

EvnB1\_23456 = new ManualResetEvent(false);

EvnB2\_13456 = new ManualResetEvent(false);

EvnB3\_12456 = new ManualResetEvent(false);

EvnB4\_12356 = new ManualResetEvent(false);

EvnB5\_12346 = new ManualResetEvent(false);

EvnB6\_12345 = new ManualResetEvent(false);

Sem\_A2 = new Semaphore(0, 1);

Sem\_A3 = new Semaphore(0, 1);

Sem\_A4 = new Semaphore(0, 1);

Sem\_A5 = new Semaphore(0, 1);

Sem\_A6 = new Semaphore(0, 1);

Thread t1 = new Thread(T1);

Thread t2 = new Thread(T2);

Thread t3 = new Thread(T3);

Thread t4 = new Thread(T4);

Thread t5 = new Thread(T5);

Thread t6 = new Thread(T6);

t1.Start();

t2.Start();

t3.Start();

t4.Start();

t5.Start();

t6.Start();

System.Console.WriteLine("Lab3 finished");

System.Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

/\*\*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* \*

\* Parallel and Distributed Computing \*

\* Laboratory work #3. C# \*

\* \*

\* Task: A=(B\*C)\*Z+α\*E\*(MX\*MZ) \*

\* \*

\* \*

\* @author Krivonosov Oleksii \*

\* @group IO-34 \*

\* @date 24.03.16 \*

\* File Operations.cs \*

\* \*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\*/

namespace lab3

{

class Operations

{

public static Vector inputVector(int n, int value)

{

Vector vector = new Vector(n);

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

{

vector.set(i, value);

}

return vector;

}

public static void outputVector(Vector vector)

{

if (vector.size() < 9)

{

Console.WriteLine(vector.toString());

}

}

public static Vector mult(Vector left, Matrix right, int l, int r)

{

Vector result = new Vector(left.size());

for (int i = l; i < r; i++)

{

result.set(i, 0);

for (int j = 0; j < left.size(); j++)

{

result.set(i, result.get(i) + left.get(j) \* right.get(j, i));

}

}

return result;

}

public static Vector mult(int value, Vector vect, int l, int r)

{

Vector result = new Vector(vect.size());

for (int i = l; i < r; i++)

{

result.set(i, value \* vect.get(i));

}

return result;

}

public static Matrix inputMatrix(int n, int value)

{

Matrix matrix = new Matrix(n);

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

{

for (int j = 0; j < n; j++)

{

matrix.set(i, j, value);

}

}

return matrix;

}

public static void outputMatrix(Matrix matrix)

{

if (matrix.size() < 9)

{

Console.WriteLine(matrix.toString());

}

}

public static Matrix mult(Matrix left, Matrix right, int l, int r)

{

Matrix result = new Matrix(left.size());

for (int i = 0; i < left.size(); i++)

{

for (int j = l; j < r; j++)

{

result.set(i, j, 0);

for (int k = 0; k < left.size(); k++)

{

result.set(i, j, result.get(i, j) + left.get(i, k)

\* right.get(k, j));

}

}

}

return result;

}

public static Vector add(Vector left, Vector right, int l, int r)

{

Vector result = new Vector(left.size());

for (int i = l; i < r; i++)

{

result.set(i, left.get(i) + right.get(i));

}

return result;

}

public static Vector sort(Vector vector, int l, int r)

{

int tmp = vector.get(0);

Vector res;

res = vector;

for (int i = l; i < r; i++)

{

for (int k = i + 1; k < r; k++)

{

if (res.get(i) > res.get(k))

{

tmp = res.get(k);

res.set(k, res.get(i));

res.set(i, tmp);

}

}

}

return res;

}

private static int[] merge(int[] left, int[] right)

{

int a = 0, b = 0;

int[] merged = new int[left.Length + right.Length];

for (Int32 i = 0; i < left.Length + right.Length; i++)

{

if (b < right.Length && a < left.Length)

if (left[a] > right[b] && b < right.Length)

merged[i] = right[b++];

else

merged[i] = left[a++];

else

if (b < right.Length)

merged[i] = right[b++];

else

merged[i] = left[a++];

}

return merged;

}

public static void mergeSort(Vector vector, int l, int r)

{

if (vector.size() == 1)

return;

int mid = (r - l) / 2;

int[] merged = new int[r - l];

int[] array1 = new int[mid];

int[] array2 = new int[mid];

for (int i = 0; i < r - l; i++)

{

if (i < mid)

array1[i] = vector.get(i + l);

else

array2[i - mid] = vector.get(i + l);

}

merged = merge(array1, array2);

for (int i = l; i < r; i++)

vector.set(i, merged[i - l]);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

/\*\*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* \*

\* Parallel and Distributed Computing \*

\* Laboratory work #3. C# \*

\* \*

\* Task: A=(B\*C)\*Z+α\*E\*(MX\*MZ) \*

\* \*

\* \*

\* @author Krivonosov Oleksii \*

\* @group IO-34 \*

\* @date 24.03.16 \*

\* File Matrix.cs \*

\* \*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\*/

namespace lab3

{

class Matrix

{

private Vector[] vector;

public Matrix(int n)

{

vector = new Vector[n];

for (int i = 0; i < vector.Length; i++)

{

vector[i] = new Vector(n);

}

}

public void set(int n, int m, int val)

{

vector[n].set(m, val);

}

public int get(int n, int m)

{

return vector[n].get(m);

}

public Vector get(int index)

{

return vector[index];

}

public int size()

{

return vector.Length;

}

public String toString()

{

String res = "";

for (int i = 0; i < vector.Length; i++)

{

res += vector[i].toString();

if (i != vector.Length - 1)

{

res += "\n";

}

}

return res;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

/\*\*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* \*

\* Parallel and Distributed Computing \*

\* Laboratory work #3. C# \*

\* \*

\* Task: A=(B\*C)\*Z+α\*E\*(MX\*MZ) \*

\* \*

\* \*

\* @author Krivonosov Oleksii \*

\* @group IO-34 \*

\* @date 24.03.16 \*

\* File Vector.cs \*

\* \*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\*/

namespace lab3

{

class Vector

{

private int[] array;

public Vector(int n)

{

array = new int[n];

}

public void set(int index, int value)

{

array[index] = value;

}

public int get(int index)

{

return array[index];

}

public int size()

{

return array.Length;

}

public String toString()

{

String res = "";

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

{

res += " " + array[i];

}

return res;

}

}

}