НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №8

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

**“MPI”**

Виконав:

студент 3 курсу

ФІОТ гр. ІО-34

Кривоносов Олексій

Перевірив:

Корочкін О. В.

Київ – 2016 р.

Тема: Програмування для комп’ютерних систем з локальною пам’яттю. Бібліотека MPI.

Розробити програму для розв’язання в ПКС із ЛП математичної задачі:

MA = (В∙С)+ α∙(MT∙MК)

Бібліотека програмування: MPI

Мова програмування: Java



Структурна схема ПКС

**Виконання роботи:**

**Етап 1. Побудова паралельного алгоритму**

1. vi = (BH\*CH ), *i* =
2. v = v + vi, *i* =
3. MAH = v∙MOH + α∙(MTH∙MK)

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

**Задача *Т0***

1. Введення α, B,С, MО, MТ, МК
2. **Передати** α, B H , С H, МТ H , МО H, МК в Т1, Т2
3. **Передати** α, B 4H , С 4H, МТ 4H , МО4H, МК в Т3
4. **Передати** α, B 3H , С 3H, МТ 3H , МО3H, МК в Т4
5. Обчислення : v0 = (BH∙CH )
6. Прийняти v1 від Т1
7. Прийняти v2 від Т2
8. Прийняти v3 від Т3
9. Прийняти v4 від Т4
10. Обчислити v = v0+ v1+ v2+ v3+v4
11. Передати v задачам Т1, Т2, Т3, Т4
12. Обчислити MAH = v∙MOH + α∙(MTH∙MK)
13. **Прийняти** МА H від Т1, Т2
14. Прийняти МА4H від Т3
15. Прийняти МА3H від Т4
16. Вивести МА

**Задача *Тi* , i=(1,2)**

1. **Прийняти** α, B H , С H, МТ H , МОH, МК від Т0
2. Обчислення : vi = (BH∙CH )
3. Передати vi задачі Т0
4. **Прийняти** v від задачі Т0
5. Обчислити MAH = v∙MOH + α∙(MTH∙MK)
6. Передати MAH в Т0

**Задача Т3**

1. **Прийняти** α, B 4H , С 4H, МТ 4H , МО4H, МК від Т0
2. Передати α, B H , С H, МТ H , МОH, МК в Т5, Т6, Т7
3. Обчислення : v3 = (BH∙CH )
4. **Прийняти** v5, v6, v7 від задач Т5, Т6, Т7
5. Обчислення v3 = v3 + v5 + v6 + v7
6. Передати v3 в задачу Т0
7. Прийняти vвід Т0
8. Передати v в Т5, Т6, Т7
9. Обчислити MAH = v∙MOH + α∙(MTH∙MK)
10. Прийняти МАН від Т5, Т6, Т7
11. Передати МА4Н ­задачі Т0

**Задача Т4**

1. **Прийняти** α, B 3H , С 3H, МТ 3H , МО3H, МК від Т0
2. Передати α, B 2H , С 2H, МТ 2H , МО2H, МК в Т8
3. Обчислення : v4 = (BH∙CH )
4. **Прийняти** v8 від задачі Т8
5. Обчислення v4 = v4 + v8
6. Передати v4 в задачу Т0
7. Прийняти vвід Т0
8. Передати v в Т8
9. Обчислити MAH = v∙MOH + α∙(MTH∙MK)
10. Прийняти МА2Н від Т8
11. Передати МА3Н ­задачі Т0

**Задача *Тi* , i=(5..7)**

1. **Прийняти** α, B H , С H, МТ H , МОH, МК від Т3
2. Обчислення : vi = (BH∙CH )
3. Передати vi задачі Т3
4. **Прийняти** v від задачі Т3
5. Обчислити MAH = v∙MOH + α∙(MTH∙MK)
6. Передати MAH в Т3

**Задача Т8**

1. **Прийняти** α, B 2H , С 2H, МТ 2H , МО 2H, МК від Т3
2. Передати α, B H , С H, МТ H , МОH, МК в Т9
3. Обчислення : v8 = (BH∙CH )
4. **Прийняти** v9 від задачі Т9
5. Обчислення v8 = v8 + v9
6. Передати v8 в задачу Т9
7. Прийняти vвід Т0
8. Передати v в Т9
9. Обчислити MAH = v∙MOH + α∙(MTH∙MK)
10. Прийняти МАН від Т9
11. Передати МА2Н ­задачі Т4

**Задача Т9**

1. **Прийняти** α, B H , С H, МТ H , МОH, МК від Т8
2. Обчислення : v9 = (BH∙CH )
3. Передати v9 задачі Т8
4. **Прийняти** v від задачі Т8
5. Обчислити MAH = v∙MOH + α∙(MTH∙MK)
6. Передати MAH в Т8

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

**Етап 4. Код програми**

**package** lab8PPKS;

/\*

-- Parallel and Distributed Computing

-- Laboratory work #8 MPI (Java)

--

-- MA = (B\*C)\*MO + alpha\*(MT\*MK)

--

-- Author: Kryvonosov Oleksii, group IO-34

-- Date: 15.05.2016

------------------------------------------------------------------ \*/

**import** mpi.\*;

**public** **class** Executor {

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

**int** P = 10;

**int** N = 10;

**int** H = N / P;

**int** fill = 1;

**int** alfa = 1;

MPI.*Init*(args);

**int** rank = MPI.*COMM\_WORLD*.Rank();

**int** currentSize = 0;

**if** (rank == 0) {

currentSize = N;

} **else** **if** (rank == 1 || rank == 2 || rank == 5 || rank == 6 || rank == 7 || rank == 9) {

currentSize = H;

} **else** **if** (rank == 3) {

currentSize = 4 \* H;

} **else** **if** (rank == 4) {

currentSize = H \* 3;

} **else** **if** (rank == 8) {

currentSize = 2 \* H;

}

**int**[] B = **new** **int**[currentSize];

**int**[] C = **new** **int**[currentSize];

**int**[][] MT = **new** **int**[currentSize][N];

**int**[][] MO = **new** **int**[N][N];

**int**[][] MK = **new** **int**[N][N];

**int**[][] MA = **new** **int**[currentSize][N];

System.***out***.println("Start task: #" + rank);

**int**[] scalars = **new** **int**[1];

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\* Задача Т0 \*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**if** (rank == 0) {

scalars[0] = alfa;

// 1. Ввод MB, alfa

// ввод данных

**for** (**int** i = 0; i < N; i++) {

B[i] = fill;

C[i] = fill;

**for** (**int** j = 0; j < N; j++) {

MT[i][j] = fill;

MO[i][j] = fill;

MK[i][j] = fill;

}

}

// \*\*\*\*\*\*\*Відправка 1 задачі \*\*\*\*\*\*\*\*\*//

MPI.*COMM\_WORLD*.Send(B, *getStart*(1, H), H, MPI.*INT*, 1, 1);

MPI.*COMM\_WORLD*.Send(C, *getStart*(1, H), H, MPI.*INT*, 1, 2);

*sendRecvMatrixPart*(MT, 1, 0, *getStart*(1, H), *getEnd*(1, H), 3);

*sendRecvMatrixPart*(MO, 1, 0, *getStart*(1, H), *getEnd*(1, H), 4);

*sendRecvMatrix*(MK, 1, 0, 5);

MPI.*COMM\_WORLD*.Send(scalars, 0, 1, MPI.*INT*, 1, 6);

// \*\*\*\*\*\*\*Відправка 2 задачі \*\*\*\*\*\*\*\*\*//

MPI.*COMM\_WORLD*.Send(B, *getStart*(2, H), H, MPI.*INT*, 2, 1);

;

MPI.*COMM\_WORLD*.Send(C, *getStart*(2, H), H, MPI.*INT*, 2, 2);

*sendRecvMatrixPart*(MT, 2, 0, *getStart*(2, H), *getEnd*(2, H), 3);

*sendRecvMatrixPart*(MO, 2, 0, *getStart*(2, H), *getEnd*(2, H), 4);

*sendRecvMatrix*(MK, 2, 0, 5);

MPI.*COMM\_WORLD*.Send(scalars, 0, 1, MPI.*INT*, 2, 6);

// \*\*\*\*\*\*\* Відправка 3 задачі \*\*\*\*\*\*//

MPI.*COMM\_WORLD*.Send(B, *getStart*(3, H), H \* 4, MPI.*INT*, 3, 1);

MPI.*COMM\_WORLD*.Send(C, *getStart*(3, H), H \* 4, MPI.*INT*, 3, 2);

*sendRecvMatrixPart*(MT, 3, 0, *getStart*(3, H), *getEnd*(6, H), 3);

*sendRecvMatrixPart*(MO, 3, 0, *getStart*(3, H), *getEnd*(6, H), 4);

*sendRecvMatrix*(MK, 3, 0, 5);

MPI.*COMM\_WORLD*.Send(scalars, 0, 1, MPI.*INT*, 3, 6);

// \*\*\*\*\*\*\* Відправка 4 задачі \*\*\*\*\*\*//

MPI.*COMM\_WORLD*.Send(B, *getStart*(7, H), H \* 3, MPI.*INT*, 4, 1);

MPI.*COMM\_WORLD*.Send(C, *getStart*(7, H), H \* 3, MPI.*INT*, 4, 2);

*sendRecvMatrixPart*(MT, 4, 0, *getStart*(7, H), *getEnd*(9, H), 3);

*sendRecvMatrixPart*(MO, 4, 0, *getStart*(7, H), *getEnd*(9, H), 4);

*sendRecvMatrix*(MK, 4, 0, 5);

MPI.*COMM\_WORLD*.Send(scalars, 0, 1, MPI.*INT*, 4, 6);

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\* Задачі Т1,Т2 \*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

} **else** **if** (rank == 2 || rank == 1) {

MPI.*COMM\_WORLD*.Recv(B, 0, H, MPI.*INT*, 0, 1);

MPI.*COMM\_WORLD*.Recv(C, 0, H, MPI.*INT*, 0, 2);

*sendRecvMatrixPart*(MT, rank, 0, *getStart*(0, H), H, 3);

*sendRecvMatrixPart*(MO, rank, 0, *getStart*(0, H), H, 4);

*sendRecvMatrix*(MK, rank, 0, 5);

MPI.*COMM\_WORLD*.Recv(scalars, 0, 1, MPI.*INT*, 0, 6);

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

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

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

} **else** **if** (rank == 3) {

// \*\*\* Прийняти з Т0 \*\*\*\* //

MPI.*COMM\_WORLD*.Recv(B, 0, currentSize, MPI.*INT*, 0, 1);

MPI.*COMM\_WORLD*.Recv(C, 0, currentSize, MPI.*INT*, 0, 2);

*sendRecvMatrixPart*(MT, 3, 0, *getStart*(0, H), H \* 4, 3);

*sendRecvMatrixPart*(MO, 3, 0, *getStart*(0, H), H \* 4, 4);

*sendRecvMatrix*(MK, 3, 0, 5);

MPI.*COMM\_WORLD*.Recv(scalars, 0, 1, MPI.*INT*, 0, 6);

// \*\* Передати в Т5\*\* //

MPI.*COMM\_WORLD*.Send(B, *getStart*(1, H), H, MPI.*INT*, 5, 1);

MPI.*COMM\_WORLD*.Send(C, *getStart*(1, H), H, MPI.*INT*, 5, 2);

*sendRecvMatrixPart*(MT, 5, 3, *getStart*(1, H), *getEnd*(1, H), 3);

*sendRecvMatrixPart*(MO, 5, 3, *getStart*(1, H), *getEnd*(1, H), 4);

*sendRecvMatrix*(MK, 5, 3, 5);

MPI.*COMM\_WORLD*.Send(scalars, 0, 1, MPI.*INT*, 5, 6);

// \*\* Передати в Т6\*\* //

MPI.*COMM\_WORLD*.Send(B, *getStart*(2, H), H, MPI.*INT*, 6, 1);

MPI.*COMM\_WORLD*.Send(C, *getStart*(2, H), H, MPI.*INT*, 6, 2);

*sendRecvMatrixPart*(MT, 6, 3, *getStart*(2, H), *getEnd*(2, H), 3);

*sendRecvMatrixPart*(MO, 6, 3, *getStart*(2, H), *getEnd*(2, H), 4);

*sendRecvMatrix*(MK, 6, 3, 5);

MPI.*COMM\_WORLD*.Send(scalars, 0, 1, MPI.*INT*, 6, 6);

// \*\* Передати в Т7\*\* //

MPI.*COMM\_WORLD*.Send(B, *getStart*(3, H), H, MPI.*INT*, 7, 1);

MPI.*COMM\_WORLD*.Send(C, *getStart*(3, H), H, MPI.*INT*, 7, 2);

*sendRecvMatrixPart*(MT, 7, 3, *getStart*(3, H), *getEnd*(3, H), 3);

*sendRecvMatrixPart*(MO, 7, 3, *getStart*(3, H), *getEnd*(3, H), 4);

*sendRecvMatrix*(MK, 7, 3, 5);

MPI.*COMM\_WORLD*.Send(scalars, 0, 1, MPI.*INT*, 7, 6);

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

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

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

} **else** **if** (rank == 4) {

// \*\* Прийняти з Т0 \*\*//

MPI.*COMM\_WORLD*.Recv(B, 0, currentSize, MPI.*INT*, 0, 1);

MPI.*COMM\_WORLD*.Recv(C, 0, currentSize, MPI.*INT*, 0, 2);

*sendRecvMatrixPart*(MT, 4, 0, *getStart*(0, H), H \* 3, 3);

*sendRecvMatrixPart*(MO, 4, 0, *getStart*(0, H), H \* 3, 4);

*sendRecvMatrix*(MK, 4, 0, 5);

MPI.*COMM\_WORLD*.Recv(scalars, 0, 1, MPI.*INT*, 0, 6);

// \*\*\*\* Передати в Т8 \*\*\*\* //

MPI.*COMM\_WORLD*.Send(B, *getStart*(1, H), H \* 2, MPI.*INT*, 8, 1);

MPI.*COMM\_WORLD*.Send(C, *getStart*(1, H), H \* 2, MPI.*INT*, 8, 2);

*sendRecvMatrixPart*(MT, 8, 4, *getStart*(1, H), *getEnd*(2, H), 3);

*sendRecvMatrixPart*(MO, 8, 4, *getStart*(1, H), *getEnd*(2, H), 4);

*sendRecvMatrix*(MK, 8, 4, 5);

MPI.*COMM\_WORLD*.Send(scalars, 0, 1, MPI.*INT*, 8, 6);

} **else**

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\* Задачі Т5, Т6, Т7 \*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**if** (rank == 7 || rank == 6 || rank == 5) {

// \*\* Прийняти з Т3 \*\*//

MPI.*COMM\_WORLD*.Recv(B, 0, currentSize, MPI.*INT*, 3, 1);

MPI.*COMM\_WORLD*.Recv(C, 0, currentSize, MPI.*INT*, 3, 2);

*sendRecvMatrixPart*(MT, rank, 3, *getStart*(0, H), H, 3);

*sendRecvMatrixPart*(MO, rank, 3, *getStart*(0, H), H, 4);

*sendRecvMatrix*(MK, rank, 3, 5);

MPI.*COMM\_WORLD*.Recv(scalars, 0, 1, MPI.*INT*, 3, 6);

} **else**

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\* Задача Т8 \*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**if** (rank == 8) {

// \*\* Прийняти з Т4 \*\*//

MPI.*COMM\_WORLD*.Recv(B, 0, 2 \* H, MPI.*INT*, 4, 1);

MPI.*COMM\_WORLD*.Recv(C, 0, 2 \* H, MPI.*INT*, 4, 2);

*sendRecvMatrixPart*(MT, rank, 4, *getStart*(0, H), *getEnd*(1, H), 3);

*sendRecvMatrixPart*(MO, rank, 4, *getStart*(0, H), *getEnd*(1, H), 4);

*sendRecvMatrix*(MK, rank, 4, 5);

MPI.*COMM\_WORLD*.Recv(scalars, 0, 1, MPI.*INT*, 4, 6);

// \*\*\*\* Передача в задачу Т9 \*\*\*\*\* //

MPI.*COMM\_WORLD*.Send(B, *getStart*(1, H), H, MPI.*INT*, 9, 1);

MPI.*COMM\_WORLD*.Send(C, *getStart*(1, H), H, MPI.*INT*, 9, 2);

*sendRecvMatrixPart*(MT, 9, 8, *getStart*(1, H), *getEnd*(1, H), 3);

*sendRecvMatrixPart*(MO, 9, 8, *getStart*(1, H), *getEnd*(1, H), 4);

*sendRecvMatrix*(MK, 9, 8, 5);

MPI.*COMM\_WORLD*.Send(scalars, 0, 1, MPI.*INT*, 9, 6);

} **else**

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\* Задача Т9 \*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**if** (rank == 9) {

// \*\* Прийняти з Т8 \*\*//

MPI.*COMM\_WORLD*.Recv(B, 0, H, MPI.*INT*, 8, 1);

MPI.*COMM\_WORLD*.Recv(C, 0, H, MPI.*INT*, 8, 2);

*sendRecvMatrixPart*(MT, rank, 8, *getStart*(0, H), *getEnd*(0, H), 3);

*sendRecvMatrixPart*(MO, rank, 8, *getStart*(0, H), *getEnd*(0, H), 4);

*sendRecvMatrix*(MK, rank, 8, 5);

MPI.*COMM\_WORLD*.Recv(scalars, 0, 1, MPI.*INT*, 8, 6);

}

**int**[] Bi = **new** **int**[H];

**int**[] Ci = **new** **int**[H];

**int**[] V = **new** **int**[1];

**int**[][] MTi = **new** **int**[H][N];

**int**[][] MOi = **new** **int**[H][N];

**if** (B.length > H) {

Bi = *sliceVector*(B, 0, H);

Ci = *sliceVector*(C, 0, H);

MTi = *sliceMatrix*(MT, 0, H, N);

MOi = *sliceMatrix*(MO, 0, H, N);

} **else** {

Bi = B;

Ci = C;

MTi = MT;

MOi = MO;

}

/// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// Отримання скалярного добутку (В\*С)//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**int**[] Vi = **new** **int**[1];

Vi[0] = *getScalar*(Bi, Ci);

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// отримання часткового скалярного добутку від задач//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\* Задача Т9 \*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**if** (rank == 9) {

MPI.*COMM\_WORLD*.Send(Vi, 0, 1, MPI.*INT*, 8, 7);

} **else**

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\* Задача Т8 \*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**if** (rank == 8) {

**int**[] Vneibor = **new** **int**[H];

MPI.*COMM\_WORLD*.Recv(Vneibor, 0, 1, MPI.*INT*, 9, 7);

Vi[0] = Vi[0] + Vneibor[0];

MPI.*COMM\_WORLD*.Send(Vi, 0, 1, MPI.*INT*, 4, 7);

} **else**

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\* Задача Т(5,6,7) \*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**if** (rank == 7 || rank == 6 || rank == 5) {

MPI.*COMM\_WORLD*.Send(Vi, 0, 1, MPI.*INT*, 3, 7);

} **else**

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

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

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**if** (rank == 4) {

**int**[] Vneibor = **new** **int**[H];

MPI.*COMM\_WORLD*.Recv(Vneibor, 0, 1, MPI.*INT*, 8, 7);

Vi[0] = Vi[0] + Vneibor[0];

MPI.*COMM\_WORLD*.Send(Vi, 0, 1, MPI.*INT*, 0, 7);

} **else**

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

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

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**if** (rank == 3) {

**int**[] Vneibor5 = **new** **int**[H];

**int**[] Vneibor6 = **new** **int**[H];

**int**[] Vneibor7 = **new** **int**[H];

MPI.*COMM\_WORLD*.Recv(Vneibor5, 0, 1, MPI.*INT*, 5, 7);

Vi[0] = Vi[0] + Vneibor5[0];

MPI.*COMM\_WORLD*.Recv(Vneibor6, 0, 1, MPI.*INT*, 6, 7);

Vi[0] = Vi[0] + Vneibor6[0];

MPI.*COMM\_WORLD*.Recv(Vneibor7, 0, 1, MPI.*INT*, 7, 7);

Vi[0] = Vi[0] + Vneibor7[0];

MPI.*COMM\_WORLD*.Send(Vi, 0, 1, MPI.*INT*, 0, 7);

} **else**

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\* Задача Т(2,1) \*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**if** (rank == 2 || rank == 1) {

MPI.*COMM\_WORLD*.Send(Vi, 0, 1, MPI.*INT*, 0, 7);

} **else**

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\* Задача Т0 \*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**if** (rank == 0) {

**int**[] Vneibor1 = **new** **int**[H];

**int**[] Vneibor2 = **new** **int**[H];

**int**[] Vneibor3 = **new** **int**[H];

**int**[] Vneibor4 = **new** **int**[H];

V = Vi;

MPI.*COMM\_WORLD*.Recv(Vneibor1, 0, 1, MPI.*INT*, 1, 7);

V[0] = V[0] + Vneibor1[0];

MPI.*COMM\_WORLD*.Recv(Vneibor2, 0, 1, MPI.*INT*, 2, 7);

V[0] = V[0] + Vneibor2[0];

MPI.*COMM\_WORLD*.Recv(Vneibor4, 0, 1, MPI.*INT*, 4, 7);

V[0] = V[0] + Vneibor4[0];

MPI.*COMM\_WORLD*.Recv(Vneibor3, 0, 1, MPI.*INT*, 3, 7);

V[0] = V[0] + Vneibor3[0];

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*Розсилка скалярного добутку задачам\*\*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**for** (**int** i = 1; i <= 4; i++) {

MPI.*COMM\_WORLD*.Send(V, 0, 1, MPI.*INT*, i, 8);

}

}

**if** (rank == 1 || rank == 2) {

MPI.*COMM\_WORLD*.Recv(V, 0, 1, MPI.*INT*, 0, 8);

} **else** **if** (rank == 3) {

MPI.*COMM\_WORLD*.Recv(V, 0, 1, MPI.*INT*, 0, 8);

**for** (**int** i = 5; i <= 7; i++) {

MPI.*COMM\_WORLD*.Send(V, 0, 1, MPI.*INT*, i, 8);

}

}

**if** (rank == 4) {

MPI.*COMM\_WORLD*.Recv(V, 0, 1, MPI.*INT*, 0, 8);

MPI.*COMM\_WORLD*.Send(V, 0, 1, MPI.*INT*, 8, 8);

}

**if** (rank >= 5 && rank <= 7) {

MPI.*COMM\_WORLD*.Recv(V, 0, 1, MPI.*INT*, 3, 8);

}

**if** (rank == 8) {

MPI.*COMM\_WORLD*.Recv(V, 0, 1, MPI.*INT*, 4, 8);

MPI.*COMM\_WORLD*.Send(V, 0, 1, MPI.*INT*, 9, 8);

}

**if** (rank == 9) {

MPI.*COMM\_WORLD*.Recv(V, 0, 1, MPI.*INT*, 8, 8);

}

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*MAH = v∙MOH + α∙(MTH∙MK)\*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**int** alphaL = scalars[0];

**int** vL = V[0];

**int** start = 0;

**int** end = H;

**for** (**int** i = start; i < end; i++) {

**for** (**int** j = 0; j < N; j++) {

MA[i][j] = 0;

**for** (**int** k = 0; k < N; k++) {

MA[i][j] += alphaL \* MTi[i][k] \* MK[k][j];

}

MA[i][j] += MOi[i][j] \* vL;

}

}

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// \*\*\*\*Отримання МА від задача \*\*\*//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

**if** (rank == 9) {

*sendRecvMatrixPart*(MA, 8, 9, 0, H, 10);

} **else** **if** (rank == 8) {

*sendRecvMatrixPart*(MA, 8, 9, H, H \* 2, 10);

*sendRecvMatrixPart*(MA, 4, 8, 0, H \* 2, 10);

} **else** **if** (rank == 4) {

*sendRecvMatrixPart*(MA, 4, 8, H, H \* 3, 10);

*sendRecvMatrixPart*(MA, 0, 4, 0, H \* 3, 10);

}

**if** (rank >= 5 && rank <= 7) {

*sendRecvMatrixPart*(MA, 3, rank, 0, H, 10);

}

**if** (rank == 3) {

**int** neibRank = 5;

**for** (**int** i = 0; i < 3; i++) {

*sendRecvMatrixPart*(MA, 3, neibRank, *getStart*(i + 1, H), *getEnd*(i + 1, H), 10);

neibRank++;

}

*sendRecvMatrixPart*(MA, 0, 3, 0, H \* 4, 10);

}

**if** (rank == 1 || rank == 2) {

*sendRecvMatrixPart*(MA, 0, rank, 0, H, 10);

}

**if** (rank == 0) {

*sendRecvMatrixPart*(MA, 0, 1, H, H \* 2, 10);

*sendRecvMatrixPart*(MA, 0, 2, H \* 2, H \* 3, 10);

*sendRecvMatrixPart*(MA, 0, 3, H \* 3, H \* 7, 10);

*sendRecvMatrixPart*(MA, 0, 4, H \* 7, H \* 10, 10);

// \*\*\*\*\* вивід даних \*\*\*\* //

*outputMatrix*(MA);

}

System.***out***.println("End task: #" + rank);

MPI.*Finalize*();

}

**public** **static** **int** getStart(**int** rank, **int** H) {

**return** rank \* H;

}

**public** **static** **int** getEnd(**int** rank, **int** H) {

**return** ((rank \* H) + H);

}

**public** **static** **void** outputMatrix(**int**[][] Matrix) {

**if** (Matrix.length < 100) {

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

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

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

}

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

}

}

}

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

**int** result = 0;

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

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

}

**return** result;

}

**public** **static** **int**[] sliceVector(**int**[] A, **int** start, **int** count) {

**int**[] result = **new** **int**[count];

**for** (**int** i = 0; i < count; i++) {

result[i] = A[start];

start++;

}

**return** result;

}

**public** **static** **int**[][] sliceMatrix(**int** MA[][], **int** start, **int** count, **int** N) {

**int**[][] result = **new** **int**[count][N];

**for** (**int** i = 0; i < count; i++) {

result[i] = MA[start];

start++;

}

**return** result;

}

**public** **static** **void** sendRecvMatrix(**int**[][] buf, **int** dest, **int** source, **int** tag) {

**if** (MPI.*COMM\_WORLD*.Rank() == source) {

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

MPI.*COMM\_WORLD*.Send(buf[i], 0, buf[i].length, MPI.*INT*, dest, tag);

}

} **else** {

**if** (MPI.*COMM\_WORLD*.Rank() == dest) {

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

MPI.*COMM\_WORLD*.Recv(buf[i], 0, buf[i].length, MPI.*INT*, source, tag);

}

}

}

}

**public** **static** **void** sendRecvMatrixPart(**int**[][] M, **int** dest, **int** source, **int** begin, **int** end, **int** tag) {

**if** (MPI.*COMM\_WORLD*.Rank() == source) {

**for** (**int** i = begin; i < end; i++) {

MPI.*COMM\_WORLD*.Send(M[i], 0, M[i].length, MPI.*INT*, dest, tag);

}

} **else** {

**if** (MPI.*COMM\_WORLD*.Rank() == dest) {

**for** (**int** i = begin; i < end; i++) {

MPI.*COMM\_WORLD*.Recv(M[i], 0, M[i].length, MPI.*INT*, source, tag);

}

}

}

}

}