**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**КЕМЕРОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ИНСТИТУТ ЦИФРЫ**

**ДОМАШНЯЯ РАБОТА № 5**

**ПО ДИСЦИПЛИНЕ**

**“Технологии параллельных вычислений”**

**Задача № 1**

студенток 3 курса

**Колесник Полины Олеговны, Малоштановой Ольги Алексеевны**

Направление 02.03.02 – Фундаментальная информатика и информационные технологии

Преподаватель:

к-т физ.-мат.наук, доцент

С.В. Стуколов

Работа защищена

« »

“ ” 2023 г.

Кемерово 2023 г.

СОДЕРЖАНИЕ

[1. Постановка задачи 3](#_Toc145840870)

[2. Описание используемых функций 3](#_Toc145840871)

[3. Описание программы 3](#_Toc145840872)

[4. Реализация 3](#_Toc145840873)

[Заключение 6](#_Toc145840874)

[Литература 6](#_Toc145840875)

# №1

# 1. Постановка задачи

Напишите программу, используя блокирующие коммуникационные функции (MPI\_Send, MPI\_Recv), реализующую построчное распределение двумерного массива, расположенного в памяти 0-го процессора. Все остальные процессы выводят номер процесса и полученную строку от 0-го процесса. В момент запуска параллельного приложения соберите трассу выполнения программы и приведите рисунок трассы в отчете.

# 2. Описание используемых функций

MPI\_Send – процедура передачи сообщений между процессами

MPI\_Recv – процедура приема сообщений между процессами

# 3. Описание программы

На 0-м процессе задается двумерный массив (a[i][j]), который по одной строчке рассылается всем ненулевым процессам. Вспомнили как предоставить рисунок трассы.

# 4. Реализация

Текст программы:

#include <stdio.h>

#include "mpi.h"

#include "iostream"

using namespace std;

int main(int argc, char \*argv[])

{

int rank;

int size;

int n=9;

MPI\_Status stat;

MPI\_Init(&argc, &argv);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);

if(rank==0)

{

int \*\*a=new int\*[size];

    a[0]=new int[size\*n];

    for(int i = 1; i < size; i++)

        a[i]=a[i-1]+n;

    for(int i = 0; i < size; i++)

        for(int j = 0; j < n; j++)

            a[i][j]=10\*(i+1)+j+1;

    for(int i = 1; i < size; i++)

        MPI\_Send(&a[i][0],n,MPI\_INT,i,777,MPI\_COMM\_WORLD);

    delete []a[0];

    delete []a;

}

else {

    int \*b=new int[n];

    // for(int i = 1; i < n; i++)

    //     b[i]=b[i-1]+n;

    for(int i = 0; i < n; i++)

        // for(int j = 0; j < n; j++)

        b[i]=0;

    MPI\_Recv(&b[0],n,MPI\_INT,0,777,MPI\_COMM\_WORLD,&stat);

//sleep(rank);

double s=0;

    for(int i = 0; i < rank\*1000000; i++)

        s+=0.0000000001;

    printf( "rank = %d, b: \n",rank );

    for(int i = 0; i < n; i++) {

        cout<<b[i]<<" ";

    }

    cout<<endl;

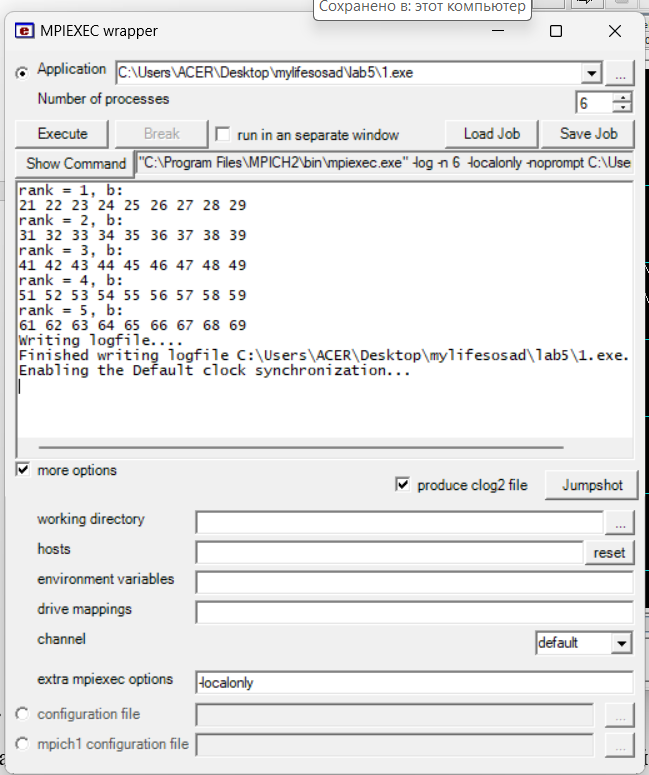
    delete []b;

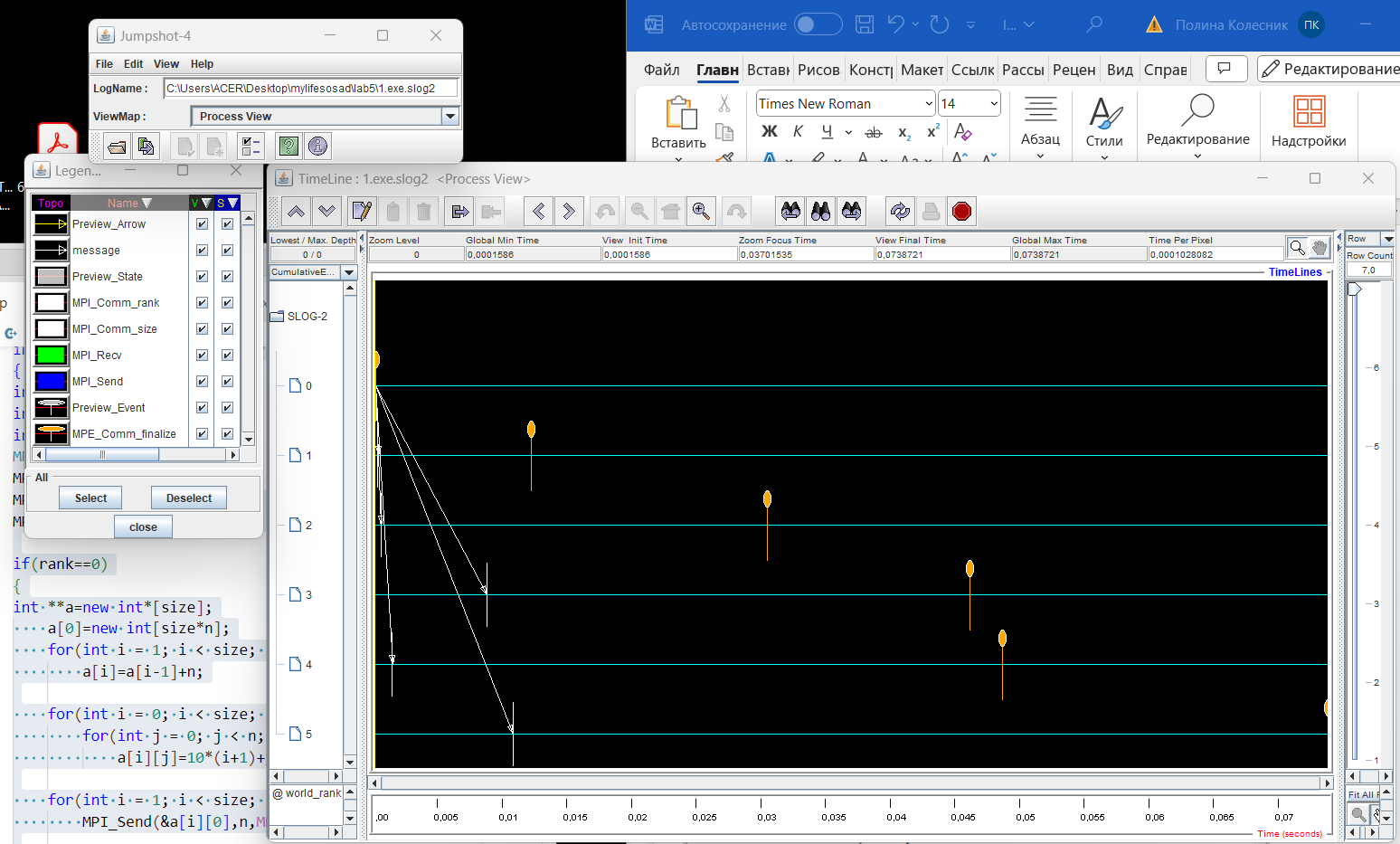
}

MPI\_Finalize();

return 0;

}





# Заключение

По трассе видно, что 0 процесс, разослал данные остальным пяти запущенным процессам

# Литература

**№2**

# 1. Постановка задачи

Напишите программу, используя блокирующие коммуникационные функции (MPI\_Send, MPI\_Recv), реализующую построчную сборку двумерного массива: на каждом процессе задается одномерный массив, который передается на 0-й, 0-й принимает одномерный массив, сразу сохраняя в соответствующую строку двумерного массива.

# 2. Описание используемых функций

MPI\_Send – процедура передачи сообщений между процессами

MPI\_Recv – процедура приема сообщений между процессами

# 3. Описание программы

На всех процессах задается одномерный массив (b[i]), который отправляется на 0-й процесс; 0-й принимает одномерный массив, сразу сохраняя в соответствующую строку двумерного массива.

# 4. Реализация

Текст программы:

#include <stdio.h>

#include "mpi.h"

#include "iostream"

using namespace std;

int main(int argc, char \*argv[])

{

int rank;

int size;

int n=9;

MPI\_Status stat;

MPI\_Init(&argc, &argv);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);

if(rank==0)

{

int \*\*a=new int\*[size];

    a[0]=new int[size\*n];

    for(int i = 1; i < size; i++)

        a[i]=a[i-1]+n;

    for(int i = 0; i < size; i++)

        for(int j = 0; j < n; j++)

            a[i][j]= 0;

    for(int i = 1; i < size; i++)

        MPI\_Recv(&a[i][0],n,MPI\_INT,i,777,MPI\_COMM\_WORLD,&stat);

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

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

                cout << a[i][j] << " ";

            cout << endl;

        }

    delete []a[0];

    delete []a;

}

else {

    int \*b=new int[n];

    // for(int i = 1; i < n; i++)

    //     b[i]=b[i-1]+n;

    for(int i = 0; i < n; i++)

        // for(int j = 0; j < n; j++)

        b[i]=rank;

    MPI\_Send(&b[0],n,MPI\_INT,0,777,MPI\_COMM\_WORLD);

//sleep(rank);

    double s=0;

    for(int i = 0; i < rank\*1000000; i++)

        s+=0.0000000001;

    printf( "rank = %d, b: \n",rank );

    for(int i = 0; i < n; i++) {

        cout<<b[i]<<" ";

    }

    cout<<endl;

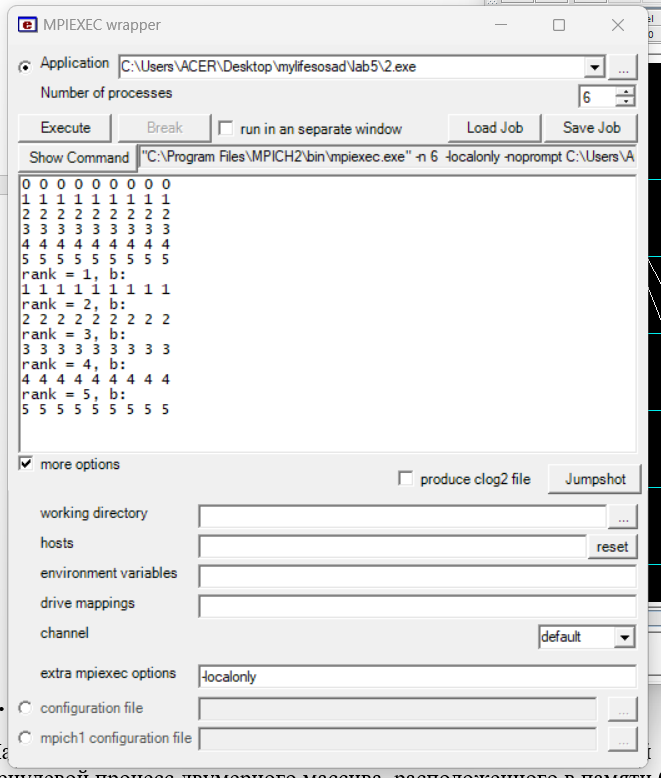
    delete []b;

}

MPI\_Finalize();

return 0;

}



# Заключение

# Литература

**№3**

# 1. Постановка задачи

Напишите программу для распределения по три строки на каждый ненулевой процесс двумерного массива, расположенного в памяти 0-го процесса. На ненулевых процессах для приема объявите трехстрочный массив.

# 2. Описание используемых функций

MPI\_Send – процедура передачи сообщений между процессами

MPI\_Recv – процедура приема сообщений между процессами

# 3. Описание программы

На нулевом процессе задаётся двумерный массив (a[i][j]), который отправляет на другие процессы по 3 строчки; Похожая задача была на паре.

# 4. Реализация

Текст программы:

#include <stdio.h>

#include "mpi.h"

#include "iostream"

using namespace std;

int main(int argc, char \*argv[])

{

int rank;

int size;

int n=9;

MPI\_Status stat;

MPI\_Init(&argc, &argv);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);

if(rank==0)

{

int \*\*a=new int\*[3\*size];

    a[0]=new int[3\*size\*n];

    for(int i = 1; i < 3\*size; i++)

        a[i]=a[i-1]+n;

    for(int i = 0; i < 3\*size; i++)

        for(int j = 0; j < n; j++)

            a[i][j]=10\*(i+1)+j+1;

    for(int i = 1; i < size; i++)

        MPI\_Send(a[3\*i],3\*n,MPI\_INT,i,777,MPI\_COMM\_WORLD);

delete []a[0];

delete []a;

}

else

{

int \*\*b=new int\*[3];

    b[0]=new int[3\*n];

    for(int i = 1; i < 3; i++)

        b[i]=b[i-1]+n;

    for(int i = 0; i < 3; i++)

        for(int j = 0; j < n; j++)

        b[i][j]=0;

    MPI\_Recv(&b[0][0],3\*n,MPI\_INT,0,777,MPI\_COMM\_WORLD,&stat);

//sleep(rank);

double s=0;

    for(int i = 0; i < rank\*1000000; i++)

    s+=0.0000000001;

printf( "rank = %d, b: \n",rank );

    for(int i = 0; i < 3; i++)

    {

        for(int j = 0; j < n; j++)

            cout<<b[i][j]<<" ";

    cout<<endl;

    }

cout<<endl;

delete []b[0];

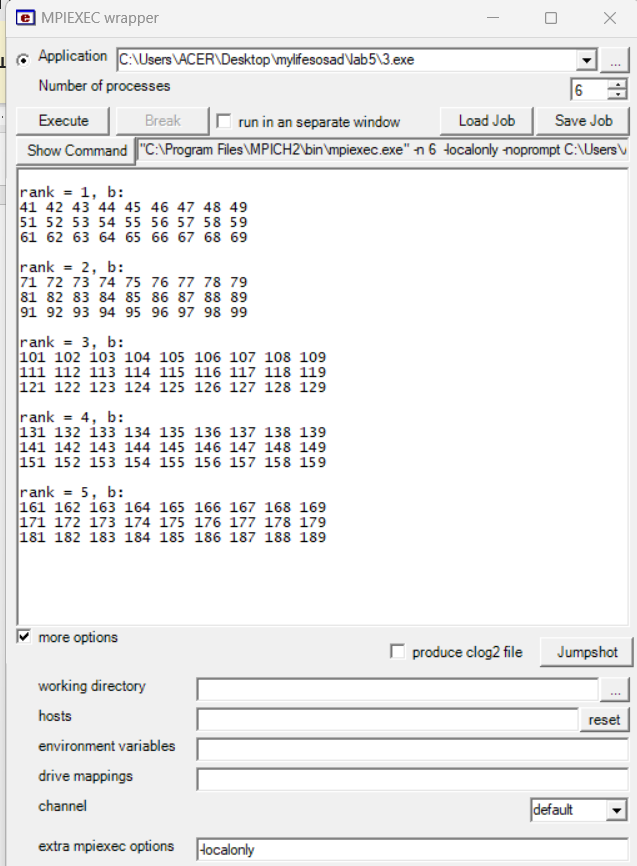
delete []b;

}

MPI\_Finalize();

return 0;

}



# Заключение

# Литература

**№4**

# 1. Постановка задачи

Напишите программу для сборки по три строки с каждого процесса на 0-й, который осуществляет прием сразу в двумерный массив.

# 2. Описание используемых функций

MPI\_Send – процедура передачи сообщений между процессами

MPI\_Recv – процедура приема сообщений между процессами

# 3. Описание программы

На всех процессах задается двумерный массив (b[i][j]), которые состоят из трёх строчек, затем отправляются на 0-й процесс;

# 4. Реализация

Текст программы:

#include <stdio.h>

#include "mpi.h"

#include "iostream"

using namespace std;

int main(int argc, char \*argv[])

{

int rank;

int size;

int n=9;

MPI\_Status stat;

MPI\_Init(&argc, &argv);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);

if(rank==0)

{

int \*\*a=new int\*[3\*size];

    a[0]=new int[3\*size\*n];

    for(int i = 1; i < 3\*size; i++)

        a[i]=a[i-1]+n;

    for(int i = 0; i < 3\*size; i++)

        for(int j = 0; j < n; j++)

            a[i][j]=0;

    for(int i = 1; i < size; i++)

        MPI\_Recv(a[3\*i],3\*n,MPI\_INT,i,777,MPI\_COMM\_WORLD, &stat);

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

        for (int j = 0; j < n; j++)

            cout << a[i][j] << " ";

        cout << endl;

    }

delete []a[0];

delete []a;

}

else

{

int \*\*b=new int\*[3];

    b[0]=new int[3\*n];

    for(int i = 1; i < 3; i++)

        b[i]=b[i-1]+n;

    for(int i = 0; i < 3; i++)

        for(int j = 0; j < n; j++)

        b[i][j]=rank;

    MPI\_Send(&b[0][0],3\*n,MPI\_INT,0,777,MPI\_COMM\_WORLD);

//sleep(rank);

double s=0;

    for(int i = 0; i < rank\*1000000; i++)

    s+=0.0000000001;

printf( "rank = %d, b: \n",rank );

    for(int i = 0; i < 3; i++)

    {

        for(int j = 0; j < n; j++)

            cout<<b[i][j]<<" ";

    cout<<endl;

    }

cout<<endl;

delete []b[0];

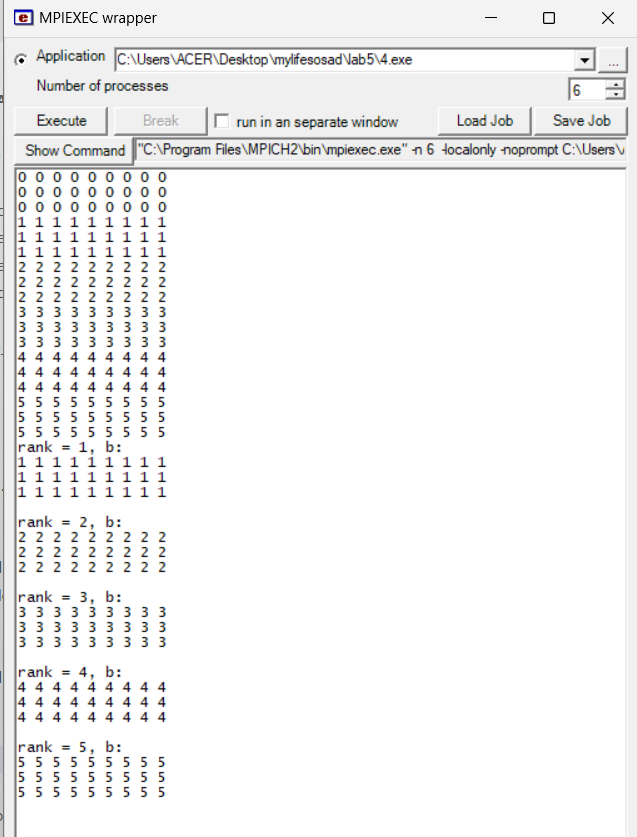
delete []b;

}

MPI\_Finalize();

return 0;

}



# Заключение

# Литература