МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

**Федеральное государственное бюджетное образовательное учреждение высшего образования  
 «Кемеровский государственный университет»**

**Институт фундаментальных наук**

**ДОМАШНЯЯ РАБОТА №4**

**ПО ДИСЦИПЛИНЕ**

**“Технологии параллельных вычислений”**

**Задача № 1**

студента 3 курса

**Сулима Роман Иванович**

Направление 02.03.02 – Фундаментальная информатика и информационные технологии

Преподаватель:

к-т физ.-мат.наук, доцент

С.В. Стуколов

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Работа защищена:

“\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_201\_г.

с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_

Кемерово 2021

СОДЕРЖАНИЕ

[1. Постановка задачи 2](#_Toc84449117)

[2. Реализация 2](#_Toc84449118)

[Заключение 7](#_Toc84449119)

[Литература 7](#_Toc84449120)

# 1. Постановка задачи

Повторите способ выделения памяти под двумерный массив, расположенный непрерывно в ОП. Посмотрите в прилагаемом материале еще раз примеры пересылки двумерных массивов. Выполните упражнения, приведенные в конце параграфа - 1, 2, 3, 4 задачи. Оформляете в одном отчете - номер задачи, текст условия, текст программы с поясняющими комментариями в программе, скрин компиляции и скрин запуска на несколько процессов.

# 2. Реализация

## Задача 1.

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.

Программный код:

#include <stdio.h>

#include "mpi.h"

#include "stdlib.h"

#include <iostream>

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

    int rank;

    int size;

    int n = 10;

    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\*size];

        for (int i = 1; i < size; i++)

            a[i] = a[i-1] + size;

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

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

                a[i][j] = i;

            }

        }

        for (int i = 1; i < size; i++)

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

    }

    else

    {

        int \*b = new int[size];

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

        printf("rank = %d", rank);

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

            printf(" %d, ", b[i]);

        printf("\n");

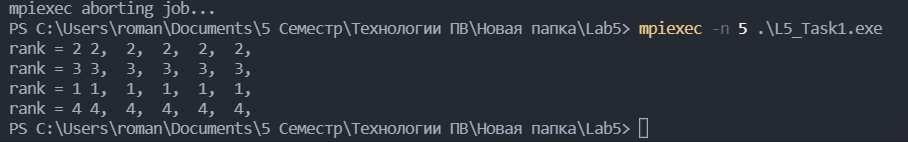
    }

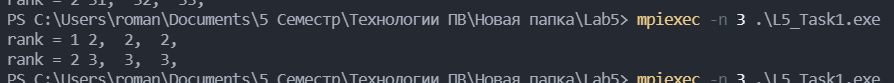
    MPI\_Finalize();

    return 0;

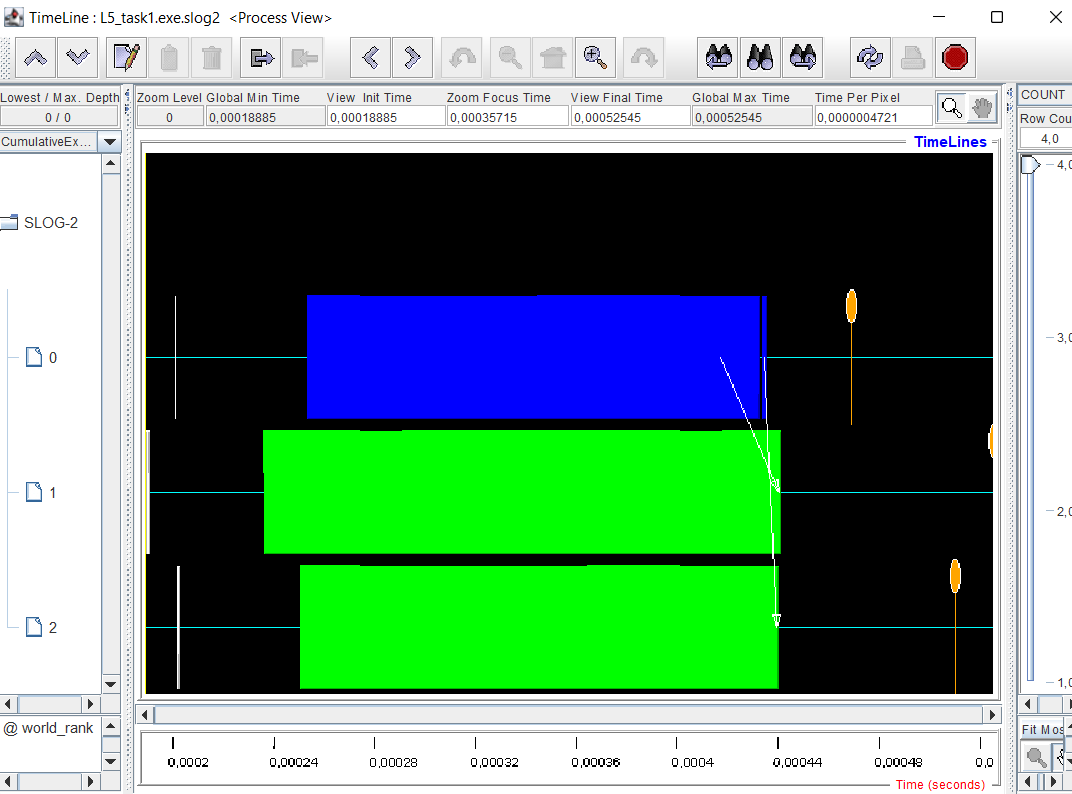
}

Выполнение:





Карта:



## Задача 2.

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.

Программный код:

#include <stdio.h>

#include "mpi.h"

#include "stdlib.h"

#include <iostream>

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

    int rank;

    int size;

    int n = 10;

    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\*size];

        for (int i = 1; i < size; i++)

            a[i] = a[i-1] + size;

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

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

                a[i][j] = i;

            }

        }

        for (int i = 1; i < size; i++)

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

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

        {

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

                printf("%d ",a[i][j]);

            printf("\n");

        }

    }

    else

    {

        int \*b = new int[size];

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

            b[i] = rank;

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

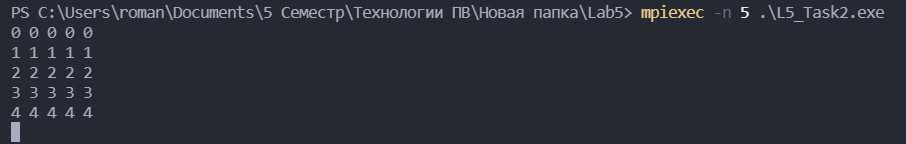
    }

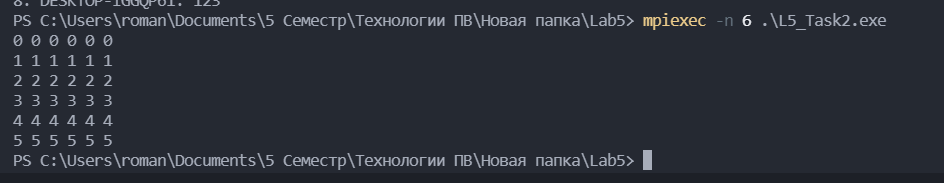
    MPI\_Finalize();

    return 0;

}

Выполнение:





## Задание 3.

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.

Программный код:

#include <stdio.h>

#include "mpi.h"

#include "stdlib.h"

#include <iostream>

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

    int rank;

    int size;

    int n = 10;

    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);

    }

    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;

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

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

        for(int i=0; i<3; i++)

        {

            for(int j=0; j<n; j++)

            printf("%d, ",b[i][j]);

            printf("\n");

        }

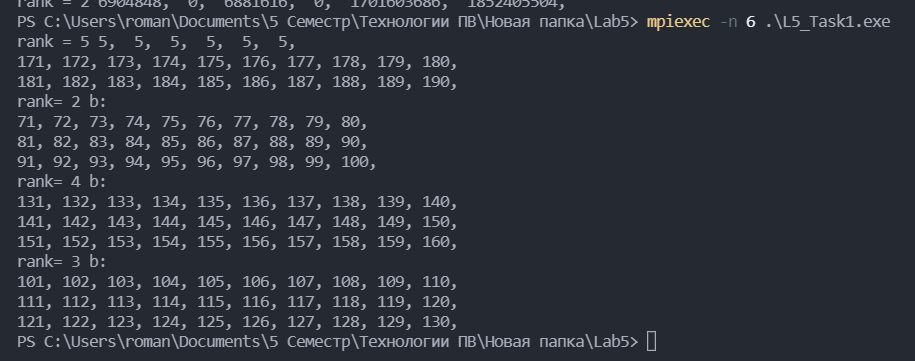
    }

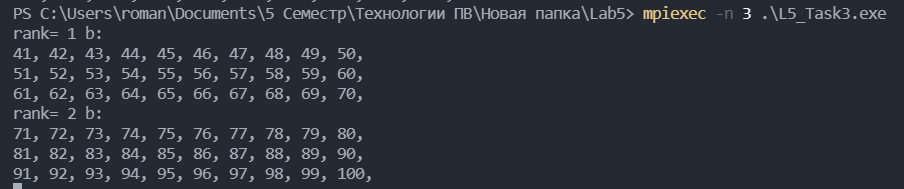
    MPI\_Finalize();

    return 0;

}

Выполнение:





## Задание 4.

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

Программный код:

#include <stdio.h>

#include "mpi.h"

#include "stdlib.h"

#include <iostream>

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

    int rank;

    int size;

    int n = 10;

    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; 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++)

            {

                printf("%d, ",a[i][j]);

            }

            printf("\n");

        }

    }

    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]=10\*(i+1)+j+1+rank;

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

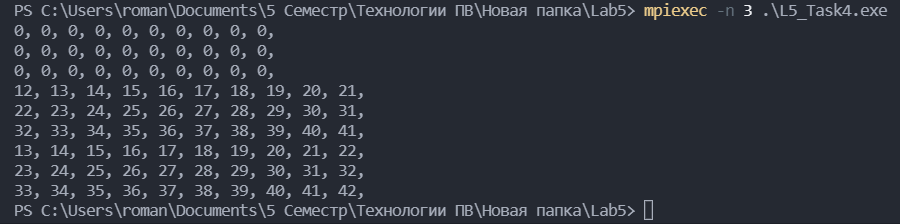
    }

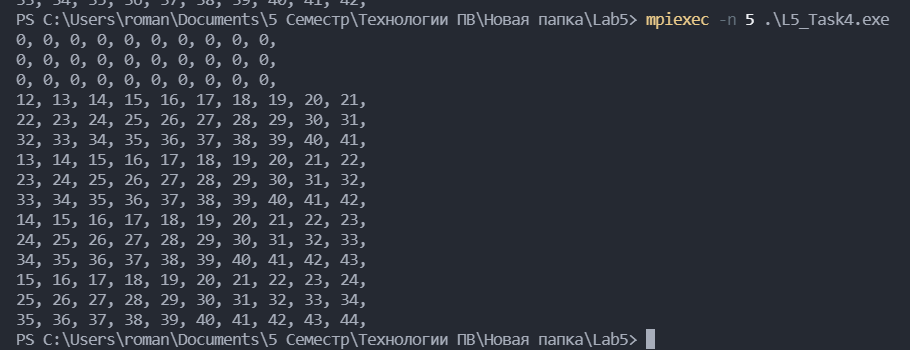
    MPI\_Finalize();

    return 0;

}

Выполнение:





# Заключение

Изучен способ передачи нескольких строк массива с одного потока на другой.

# Литература