МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

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

**Институт фундаментальных наук**

**ДОМАШНЯЯ РАБОТА №10**

**ПО ДИСЦИПЛИНЕ**

**“Технологии параллельных вычислений”**

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](#_Toc87651434)

[2. Описание используемых функций 2](#_Toc87651435)

[3. Реализация 2](#_Toc87651436)

[Заключение 9](#_Toc87651437)

# 1. Постановка задачи

|  |  |
| --- | --- |
|  | Посмотрите в прилагаемом материале еще раз примеры сборки распределенных по процессам данных при помощи коллективных функций MPI\_Gather, MPI\_Allgather, MPI\_Gatherv, MPI\_Allgatherv. Выполните упражнения, приведенные в конце параграфа - 6 задач. Оформляете в одном отчете - номер задачи, текст условия, текст программы с поясняющими комментариями в программе, скрин компиляции и скрин запуска на несколько процессов. |

# 2. Описание используемых функций

MPI\_Scatter – функция разбивает массив данных на процессе 0 на равные части и отправляет в буфер другого процесса..

MPI\_Scatterv– функция разбивает массив данных на процессе 0 на разные части и отправляет в буфер другого процесса.

# 3. Реализация

Задание 1.

Используя функцию MPI\_Scatter, создайте следующую параллельную   
программу: на 0-м процессе задается одномерный массив (a[i]=i,   
i=0...2\*size), который распределяется по процессам по 2 элемента каждому.

Программный код:

#include <stdio.h>

#include "mpi.h"

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

    int rank;

    int size;

    int i,j,s=0;

    MPI\_Init(&argc, &argv);

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

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

    int \*a=new int [2\*size];

    if(rank==0)

        for(i=0;i<2\*size;i++)

            a[i]=i;

    int \*b=new int[2];

    MPI\_Scatter(a,2,MPI\_INT,b,2,MPI\_INT,0,MPI\_COMM\_WORLD);

    for(i=0; i<100000000\*rank; i++)

        s+=1;

    printf("rank= %d  a: ",rank);

    for(i=0; i<2; i++)

        printf(" %d ",b[i]);

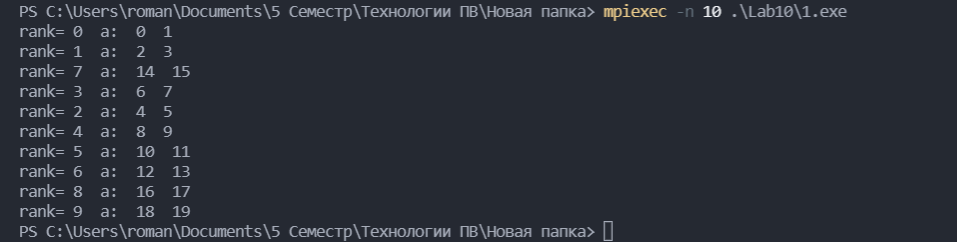
    printf("\n");

    MPI\_Finalize();

    return 0;

}

Выполнение:



Задание 2.

Используя функцию MPI\_Scatterv, создайте следующую параллельную   
программу: на 0-м процессе задается одномерный массив (a[i]=i,   
i=0...size\*(size+1)), который распределяется по процессам следующим   
образом: на 0-й процесс будет отправлено 2 элемента (0,1), на 1-й процесс   
– 4 элемента (2,3,4,5), на 2-й процесс – 6 элементов и т.д..

Программный код:

#include <stdio.h>

#include "mpi.h"

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

    int rank;

    int size;

    int i,s=0;

    MPI\_Status stat;

    MPI\_Init(&argc, &argv);

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

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

    int \*a=new int[size\*(size+1)];

    if(rank==0)

        for(i=0; i<size\*(size+1); i++)

           a[i]=i;

    int \*b=new int [rank + rank + 2];

    int \*RC = new int[size];

    int \*ds = new int [size];

    RC[0] = 2; ds[0] = 0;

    for(int i=1;i<size;i++) {

         RC[i]=RC[i-1] + 2;

         ds[i]=i\*(i+1);

    }

    MPI\_Scatterv(a,RC,ds,MPI\_INT,b,(rank + rank + 2),MPI\_INT,0,MPI\_COMM\_WORLD);

    for(i=0; i<100000000\*rank; i++)

    s+=1;

    printf("rank= %d  b: ",rank);

    for(i=0; i<rank + rank + 2; i++)

        printf("  %d  ",b[i]);

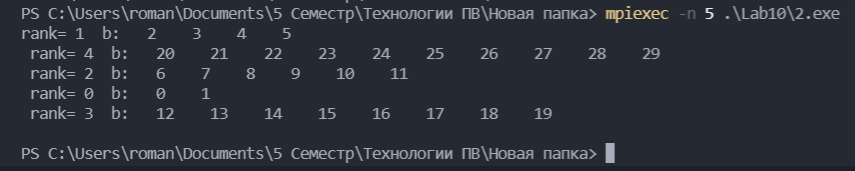
    printf("\n ");

    MPI\_Finalize();

    return 0;

}

Выполнение:



Задание 3.

Напишите программу, которая реализует рассылку по две строки каждому   
процессу из двумерного массива, расположенного в памяти 0-го процесса

Программный код:

#include <stdio.h>

#include "mpi.h"

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

    int rank;

    int size;

    int n=0,i,j,s=0;

    MPI\_Status stat;

    MPI\_Init(&argc, &argv);

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

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

    if(rank==0) scanf("%d",&n);

    MPI\_Bcast(&n,1,MPI\_INT,0,MPI\_COMM\_WORLD);

    int \*\*a=new int \*[2\*size];

    a[0]=new int [2\*n\*size];

        for(i=1; i<2\*size; i++)

            a[i]=a[i-1]+n;

    if(rank==0) {

        for(i=0;i<2\*size;i++)

            for(j=0;j<n;j++)

                a[i][j]=i;

    }

        int \*\*b=new int \*[2];

     b[0]=new int [2\*n];

        for(i=1; i<2; i++)

     b[i]=b[i-1]+n;

    MPI\_Scatter(\*a,2\*n,MPI\_INT,\*b,2\*n,MPI\_INT,0,MPI\_COMM\_WORLD);

    for(i=0; i<100000000\*rank; i++)

    s+=1;

    printf("rank= %d  b: \n",rank);

    for(i=0; i<2; i++)

    {

        for(j=0;j<n;j++)

            printf(" %d ",b[i][j]);

        printf("\n");

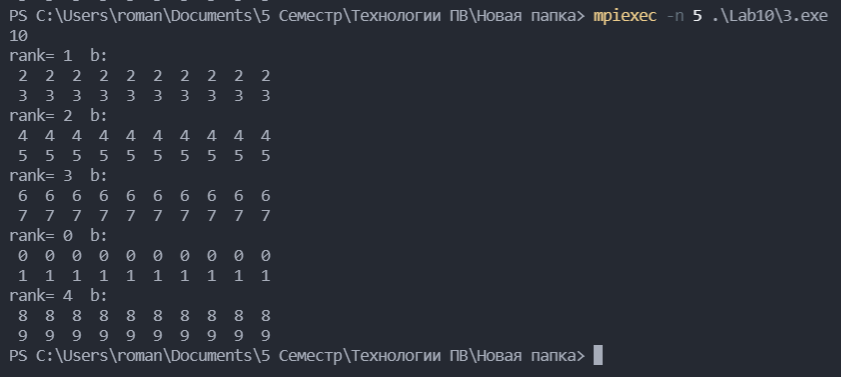
    }

    MPI\_Finalize();

    return 0;

}

Выполнение:



Задание 4.

Используя функцию MPI\_Scatterv, напишите программу, которая   
распределяет двумерный массив (квадратную матрицу произвольного   
размера), расположенный в памяти 0-го процесса, как можно одинаковыми   
по размеру блоками строк по процессам. Возможный алгоритм   
следующий:   
− На 0-м процессе считывается значение переменной n (размер массива)   
и рассылается всем (MPI\_Bcast).   
− Выделяется место под рассылаемый массив a[n][n].   
− На 0-м процессе элементам массива присваиваются значения.   
− Вычисляется оптимальное распределение строк в блоках для рассылки   
каждому процессу (разница в количестве рассылаемых строк не более 1).   
− Выделяется место в памяти под двумерные массивы на каждом   
процессе для сохранения рассылаемых блоков строк.   
− Выделяется место в памяти под вспомогательные массивы и им   
присваиваются соответствующие задаче значения.   
− Выполняется распределение массива по процессам.   
− Производится контрольный вывод полученных массивов.

Программный код:

#include <stdio.h>

#include "mpi.h"

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

    int rank;

    int size;

    int n,i,s=0, j;

    MPI\_Init(&argc, &argv);

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

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

    if(rank==0) scanf("%d",&n);

    MPI\_Bcast(&n,1,MPI\_INT,0,MPI\_COMM\_WORLD);

    int \*\*a=new int \*[n];

    a[0]=new int [n\*n];

        for(i=1; i<n; i++)

            a[i]=a[i-1]+n;

    if(rank==0) {

        for(i=0;i<n;i++)

            for(j=0;j<n;j++)

                a[i][j]=i;

    }

    int cel = n / size;

    int ostat = n % size;

    int rezerv = ostat;

    int \*RC = new int[size];

    int \*ds = new int [size];

    int \*X = new int [size];

    for (int i=0;i<size;i++) {

        if (ostat != 0) {

            ds[i] = n \* i \* (cel + 1);

            ostat--;

            RC[i] = n \* (cel + 1);

            X[i] = cel + 1;

        }

        else {

            if (i == rezerv) {

                RC[i] = n \* cel;

                ds[i] = ds[i-1] + (cel \* n) + n;

                X[i] = cel;

            }

            else {

                RC[i] = n \* cel;

                ds[i] = ds[i-1] + (cel \* n);

                X[i] = cel;

            }

        }

    }

    int \*\*b=new int \*[X[rank]];

    b[0]=new int [X[rank]\*n];

    for(i=1; i<X[rank]; i++)

        b[i]=b[i-1]+n;

    MPI\_Scatterv(\*a,RC,ds,MPI\_INT,\*b,X[rank] \* n,MPI\_INT,0,MPI\_COMM\_WORLD);

    for(i=0; i<100000000\*rank; i++)

    s+=1;

    printf("rank= %d  b: \n",rank);

    for(i=0; i<X[rank]; i++)

    {

        for(j=0;j<n;j++)

            printf(" %d ",b[i][j]);

        printf("\n");

    }

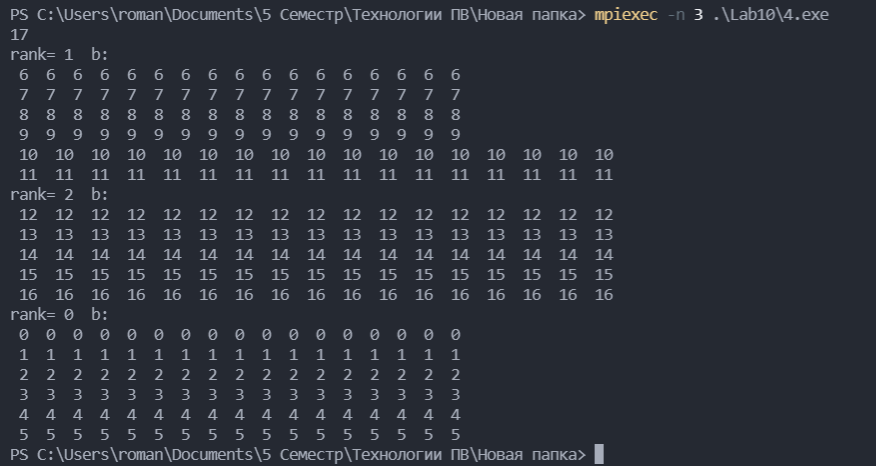
    MPI\_Finalize();

    return 0;

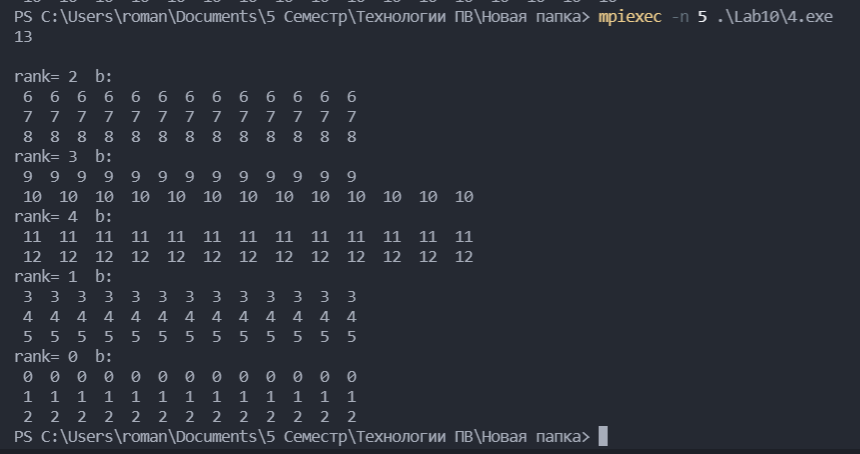
}

Выполнение:

Три процесса и n = 17.



Пять процессов и n = 13.



Задание 5.

На основе примера 6 напишите программу, которая рассылает с 0-го   
процесса двумерный массив так, что в итоге каждому процессу достанется   
по две строки длиной (rank+1) элементов.

Программный код:

#include <stdio.h>

#include "mpi.h"

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

    int rank;

    int size;

    int n,i,s=0, j;

    MPI\_Init(&argc, &argv);

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

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

    if(rank==0) scanf("%d",&n);

    int \*\*a=new int \*[2\*size];

    a[0]=new int [2\*n\*size];

        for(i=1; i<2\*size; i++)

            a[i]=a[i-1]+n;

    if(rank==0) {

        for(i=0;i<2\*size;i++)

            for(j=0;j<n;j++)

                a[i][j]=i;

    }

    int \*\*b=new int \*[2];

    b[0]=new int [(rank+1)\*2];

    for(i=1; i<2; i++)

        b[i]=b[i-1]+(rank+1);

    int \*RC = new int[size];

    int \*ds = new int [size];

        for (int i=0;i<size;i++)

    {

     RC[i]=2\* (i + 1);

     ds[i]=2\*n\*i;

    }

    MPI\_Scatterv(\*a,RC,ds,MPI\_INT,\*b,(rank+1)\*2,MPI\_INT,0,MPI\_COMM\_WORLD);

    for(i=0; i<100000000\*rank; i++)

    s+=1;

    printf("rank= %d  b: \n",rank);

    for(i=0; i<2; i++)

    {

        for(j=0;j<rank+1;j++)

            printf(" %d ",b[i][j]);

        printf("\n");

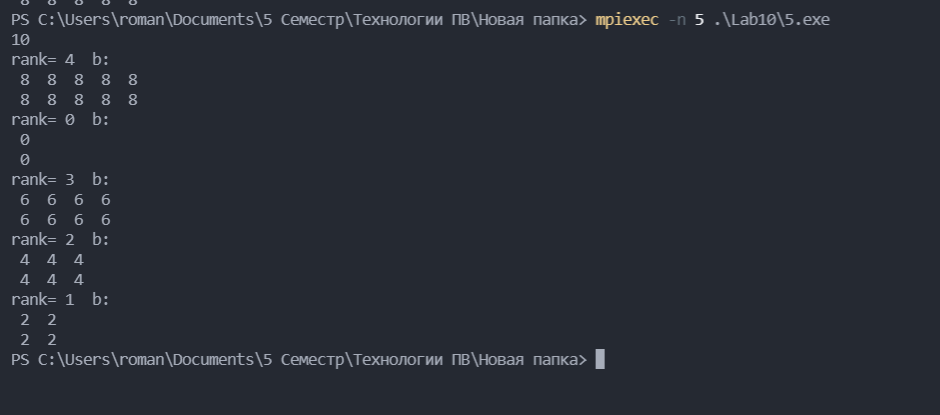
    }

    MPI\_Finalize();

    return 0;

}

Выполнение:



Задание 6.

На основе примера 5 напишите программу, которая осуществляет   
рассылку двумерного массива с 0-го процесса следующим образом (в   
случае запуска на 5-и процессах пример изначального массива и что   
каждый процесс получит):   
0 0 0 0 0 (0-й процесс получит строку целиком)   
1 1 1 1 1 (1-й процесс получит 1 1 1 1)   
2 2 2 2 2 (2-й процесс получит 2 2 2)   
3 3 3 3 3 (3-й процесс получит 3 3)   
4 4 4 4 4 (4-й процесс получит 4).

Программный код:

#include <stdio.h>

#include "mpi.h"

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

    int rank;

    int size;

    int n = 3, i, s = 0, j;

    MPI\_Init(&argc, &argv);

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

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

    int\*\* a = new int\* [size];

    a[0] = new int[size \* size];

    for (i = 1; i < size; i++)

        a[i] = a[i-1] + size;

    if (rank == 0) {

        for (i = 0; i < size; i++)

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

                a[i][j] = i;

    }

    int\* RC = new int[size];

    int\* ds = new int[size];

    for (int i = 0; i < size; i++){

        RC[i] = size - i;

        ds[i] = i \* size;

    }

    int\* b = new int[size - rank];

    MPI\_Scatterv(\*a, RC, ds, MPI\_INT, b, (size - rank), MPI\_INT, 0, MPI\_COMM\_WORLD);

    for (i = 0; i < 100000000 \* rank; i++)

        s += 1;

    printf("rank= %d b: ", rank);

    for (i = 0; i < (size - rank); i++)

        printf(" %d ", b[i]);

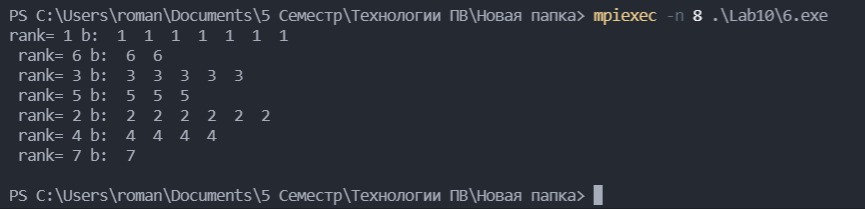
    printf("\n ");

    MPI\_Finalize();

    return 0;

}

Выполнение:



# Заключение

Выполнив лабораторную работу, научился использовать функции для пересылки сообщений с одного процесса на другие как одинаковой длины, так и разной.