**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**КЕМЕРОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ИНСТИТУТ ЦИФРЫ**

**ДОМАШНЯЯ РАБОТА № 3**

**ПО ДИСЦИПЛИНЕ**

**“Технологии параллельных вычислений”**

**Задача № 3**

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. Постановка задачи

Напишите программу, которая реализует следующий алгоритм: на 0-м процессе задается одномерный массив (a[i]=rank, i=0…3\*size), который по одному элементу рассылается всем ненулевым процессам по кругу. Например, Вы запускаете программу на 3-х процессах, 1-й процесс в итоге должен получить 1, 4, 7, 2-й процесс – 2, 5, 8. Не забудьте сделать контрольный вывод пересланных данных.

# 2. Описание используемых функций

# 3. Описание программы

На 0-м процессе задается одномерный массив (a[i]=i, i=0..3\*size), который по одному элементу рассылается всем ненулевым процессам по кругу.

# 4. Реализация

Текст программы:

#include <stdio.h>

#include "mpi.h"

#include "iostream"

using namespace std;

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

{

    int rank, size, n=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)

    {

        n = 3\*size;

        int \*a=new  int[n];

        for(int i = 0; i < n; i++)

            a[i]=i;

        for(int i =1; i< n; i++)

        {

            if(i % size != 0)

                MPI\_Send(&a[i],1,MPI\_INT,i % size,777,MPI\_COMM\_WORLD);

        }

        printf( "rank = %d, a: ",rank );

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

            cout<<a[i]<<" ";

        delete[]a;

    }

    else

    {

        int \*b = new int[3];

        for (int i = 0; i < 3; i++)

            MPI\_Recv(&b[i],1,MPI\_INT,0,777,MPI\_COMM\_WORLD,&stat);

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

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

            cout<<b[i]<<" ";

        cout<<endl;

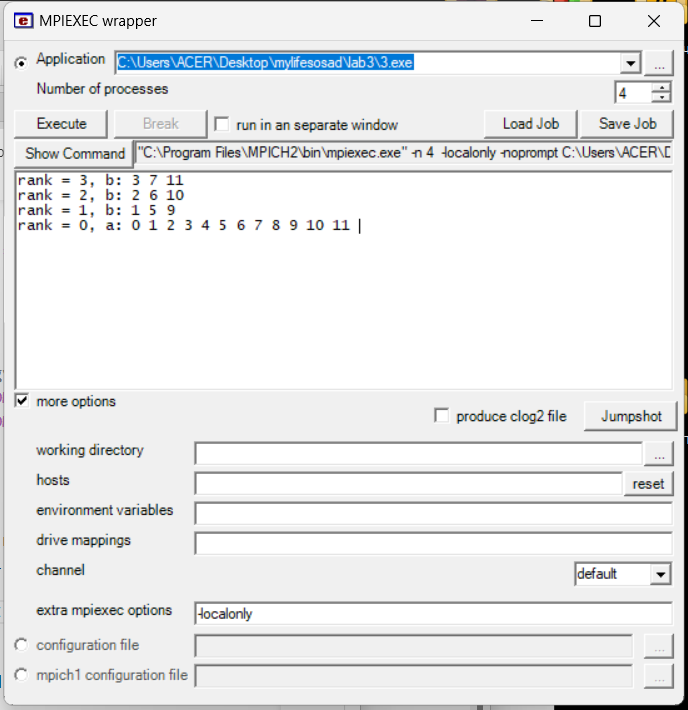
    delete []b;

    }

    MPI\_Finalize();

    return 0;

}

  
Рис.1 результат работы

# Заключение

# Литература

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**КЕМЕРОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ИНСТИТУТ ЦИФРЫ**

**ДОМАШНЯЯ РАБОТА № 3**

**ПО ДИСЦИПЛИНЕ**

**“Технологии параллельных вычислений”**

**Задача № 4**

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. Постановка задачи

Создайте следующую параллельную программу: на всех процессах

задается одномерный массив (a[i]=rank, i=0...rank+1), который отправляется на 0-й процесс; 0-й процесс объявляет необходимый по размеру одномерный массив и принимает от всех остальных пересылаемые данные. Например, Вы запускаете программу на 4-х процессах, 1-й процесс отправляет 0-му следующий массив (1,1), 2-й процесс отправляет 0-му - (2,2,2), 3-й процесс отправляет 0-му (3,3,3,3), 0-й процесс получает данные и, сохраняя в одномерный массив, выводит на экран следующее: 1,1,2,2,2,3,3,3,3.

# 2. Описание используемых функций

# 3. Описание программы

На всех процессах задается одномерный массив (a[i]=rank, i=0...rank+1), который отправляется на 0-й процесс; 0-й процесс объявляет необходимый по размеру одномерный массив и принимает от всех остальных пересылаемые данные.

# 4. Реализация

Текст программы:

#include <stdio.h>

#include "mpi.h"

#include "iostream"

using namespace std;

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

{

    int rank, size;

    int n=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)

    {

        int k = 0;

        n = ((size\*(size+1))/2)-1;

        int \*a=new  int[n];

        for(int i =1; i< size; i++)

        {   if (i ==1)

                MPI\_Recv(&a[i - 1], i+1,MPI\_INT,i,777,MPI\_COMM\_WORLD, &stat);

            else{

                MPI\_Recv(&a[i + k], i+1,MPI\_INT,i,777,MPI\_COMM\_WORLD, &stat);

                k+=i;

            }

        }

        printf( "rank = %d, a: ",rank );

        for(int i = 0; i < n; i++)

            cout<<a[i]<<" ";

        cout<<endl;

        delete[]a;

    }

    else

    {

        int \*b = new int[rank+1];

        for (int i = 0; i < rank + 1; i++)

            b[i] = rank;

        MPI\_Send(&b[0],rank+1,MPI\_INT,0,777,MPI\_COMM\_WORLD);

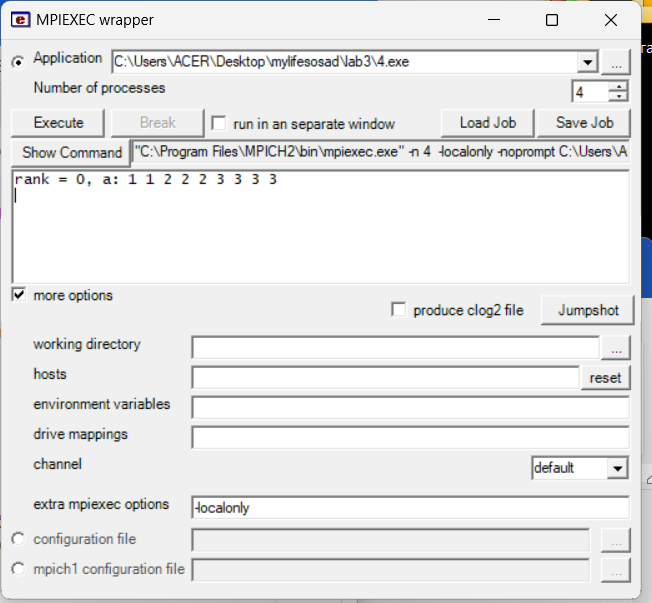
    delete []b;

    }

    MPI\_Finalize();

    return 0;

}

  
Рис.2 результат работы

# Заключение

# Литература