МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

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

**Институт фундаментальных наук**

**ДОМАШНЯЯ РАБОТА №2**

**ПО ДИСЦИПЛИНЕ**

**“Технологии параллельных вычислений”**

**Задача № 3**

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](#_Toc335561691)

[2. Описание используемых функций 2](#_Toc335561692)

[3. Описание программы 2](#_Toc335561693)

[4. Реализация 2](#_Toc335561694)

[Заключение 3](#_Toc335561695)

[Литература 3](#_Toc335561696)

# 1. Постановка задачи

Используя блокирующие коммуникационные функции типа (Точка-Точка), создайте следующую параллельную программу: на всех процессах задается одномерный массив (a(i)=rank, i=0..2), 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 3-х процессах, 1-й процесс отправляет 0-му следующий массив (1,1,1), 2-й процесс отправляет 0-му - (2,2,2), 0-й процесс получает данные и, сохраняя в одномерный массив, выводит на экран следующее: 1,1,1,2,2,2.

# 2. Описание используемых функций

MPI\_Send() – Передача сообщения другим потокам.

MPI\_Recv() – Принимает сообщение из другого потока.

# 3. Описание программы

Программа создает на каждом потоке массив размером size и заполняет его одним и тем же число равным rank. После этого все массивы пересылаются на 0 поток и там собираются в единый массив и выводится на экран.

# 4. Реализация

Программный код:

#include <stdio.h>

#include "mpi.h"

#include "stdlib.h"

int main(int argc, char \*argv[]){

    int rank;

    int size;

    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 = NULL;

        a = (int\*)malloc(sizeof(int)\*(size));

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

            a[i] = rank;

        }

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

    }

    else {

        int\* b = NULL;

        b = (int\*)malloc(sizeof(int)\*(size \* (size - 1)));

        for (int i = 0; i<size-1; i++)

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

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

            printf( "%d, ", b[i] );

    }

    MPI\_Finalize();

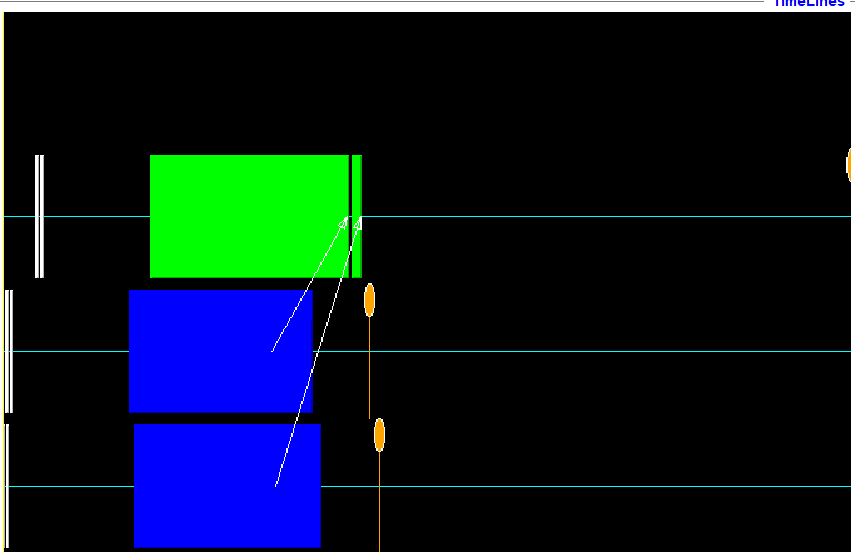
    return 0;

}

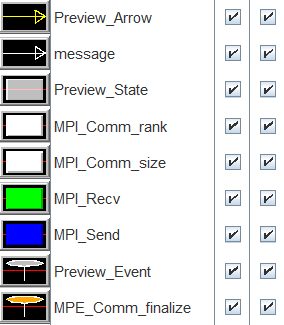
Пример реализации:



Трасса:



Условные обозначения:



# Заключение

Научились передавать массивы на с разных поток в один.

# Литература