

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра вычислительной техники

ОТЧЕТ
по лабораторной работе №4
по дисциплине «Параллельные вычисления»
Тема: «Коллективные функции»

Студентка гр. 1307

Грунская Н.Д.

Преподаватель

Манжиков Л.П.

Санкт-Петербург

2025

Цель работы.

Освоить функции коллективной обработки данных.

Задание 1.

3) Заменить в прямоугольной матрице все положительные элементы на 1, отрицательные на -1;

Задание 2.

В полученной матрице (по результатам выполнения задания 1) найти:

3) Количество элементов, оставшихся неизменными в новой матрице.

Задание 1 (по вариантам).

Решить задание 1 из лаб. работы 2 с применением коллективных функций.

Задание 2 (по вариантам).

В полученной матрице (по результатам выполнения задания 1) найти:

Решить задание 1 или 2 из лаб. работы 3 с применением коллективных функций.

Текст программы task1.cpp.

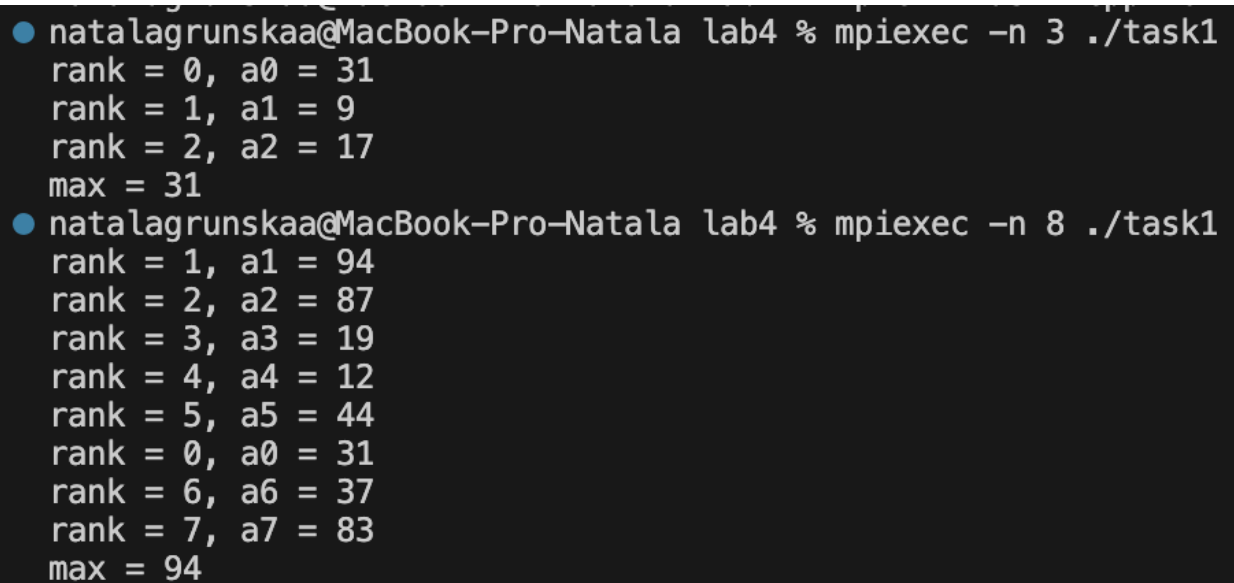
```
#include <cstdlib>
#include <iostream>
#include <ctime>
#include "mpi.h"
using namespace std;

int main(int argc, char *argv[])
{
    int rank, size;
    MPI_Status status;
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    int ai=0, max=0;
```

```

srand(rank* time(NULL));
ai = (rand() % 100) + 1;
printf("rank = %d, a%d = %d\n", rank,rank,ai);
MPI_Reduce(&ai,&max,1,MPI_INT,MPI_MAX,0, MPI_COMM_WORLD);
if(rank==0){
    printf("max = %d\n",max);
}
MPI_Finalize();
return 0;
}

```



```

● natalagrunskaa@MacBook-Pro-Natala lab4 % mpiexec -n 3 ./task1
rank = 0, a0 = 31
rank = 1, a1 = 9
rank = 2, a2 = 17
max = 31
● natalagrunskaa@MacBook-Pro-Natala lab4 % mpiexec -n 8 ./task1
rank = 1, a1 = 94
rank = 2, a2 = 87
rank = 3, a3 = 19
rank = 4, a4 = 12
rank = 5, a5 = 44
rank = 0, a0 = 31
rank = 6, a6 = 37
rank = 7, a7 = 83
max = 94

```

Рисунок 1. Запуск программы на 3-х и 8-и процессах.

Текст программы task2.cpp.

```

#include <stdio.h>
#include <iostream>
#include <vector>
#include <cstdlib>
#include <ctime>
#include "mpi.h"

#define ROWS 5
#define COLUMNS 6

int main(int argc, char **argv)
{

```

```

int rank, size;
int unchanged_count = 0;

MPI_Init(&argc, &argv);
MPI_Comm_rank(MPI_COMM_WORLD, &rank);
MPI_Comm_size(MPI_COMM_WORLD, &size);

int matrix[ROWS][COLUMNS];
int modified_matrix[ROWS][COLUMNS];

if (rank == 0)
{
    srand(time(NULL));
    std::cout << "Old matrix:\n";
    for (int i = 0; i < ROWS; i++)
    {
        for (int j = 0; j < COLUMNS; j++)
        {
            matrix[i][j] = (rand() % 21) - 10;
            std::cout << matrix[i][j] << " ";
        }
        std::cout << "\n";
    }
}

for (int i = 0; i < ROWS; ++i)
{
    MPI_Bcast(matrix[i], COLUMNS, MPI_INT, 0, MPI_COMM_WORLD);
}

for (int i = 0; i < ROWS; i++)
{
    for (int j = 0; j < COLUMNS; j++)
    {
        modified_matrix[i][j] = matrix[i][j];
    }
}

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

```

```

for (int j = 0; j < COLUMNS; j++)
{
    if (matrix[i][j] == 0)
    {
        local_unchanged_count++;
    }
    if (modified_matrix[i][j] > 0)
    {
        if (matrix[i][j] == 1)
        {
            local_unchanged_count++;
        }
        modified_matrix[i][j] = 1;
    }
    else if (modified_matrix[i][j] < 0)
    {
        if (matrix[i][j] == -1)
        {
            local_unchanged_count++;
        }
        modified_matrix[i][j] = -1;
    }
}

for (int i = 0; i < ROWS; ++i)
{
    MPI_Bcast(modified_matrix[i], COLUMNS, MPI_INT, i % size, MPI_COMM_WORLD);
}

int global_unchanged_count = 0;
MPI_Reduce(&local_unchanged_count, &global_unchanged_count, 1, MPI_INT, MPI_SUM, 0, MPI_COMM_WORLD);

if (rank == 0)
{
    std::cout << "\nNew matrix:\n";
    for (int i = 0; i < ROWS; i++)
    {
        for (int j = 0; j < COLUMNS; j++)
        {
            std::cout << modified_matrix[i][j] << " ";

```

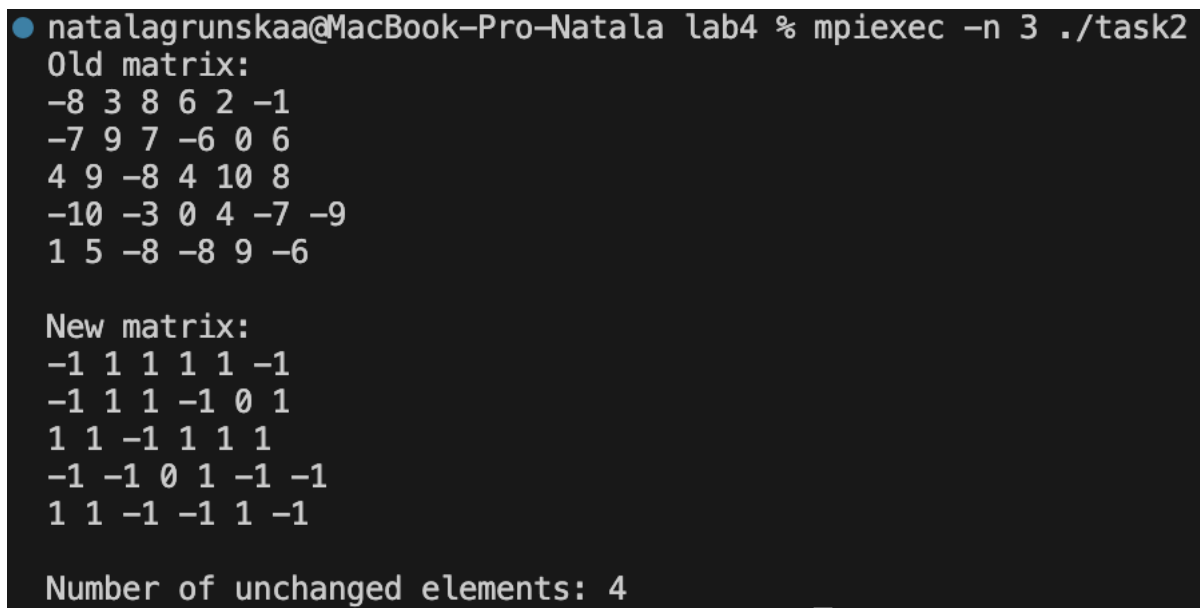
```

    }
    std::cout << "\n";
}

std::cout << "\nNumber of unchanged elements: " << global_unchanged_count << std::endl;
}

MPI_Finalize();
return 0;
}

```



```

natalagrunskaa@MacBook-Pro-Natala lab4 % mpiexec -n 3 ./task2
Old matrix:
-8 3 8 6 2 -1
-7 9 7 -6 0 6
4 9 -8 4 10 8
-10 -3 0 4 -7 -9
1 5 -8 -8 9 -6

New matrix:
-1 1 1 1 1 -1
-1 1 1 -1 0 1
1 1 -1 1 1 1
-1 -1 0 1 -1 -1
1 1 -1 -1 1 -1

Number of unchanged elements: 4

```

Рисунок 2. Запуск программы на 3 процессах

Выводы.

В ходе выполнения лабораторной работы были успешно освоены функции коллективной обработки данных в MPI. Получено понимание принципов их работы и практические навыки применения для реализации параллельных алгоритмов. Приобретен опыт использования коллективных операций для обмена данными между процессами и эффективного распределения вычислительной нагрузки. Также закрепились знания о необходимости синхронизации процессов при работе с коллективными функциями для обеспечения корректного выполнения параллельной программы.