**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра вычислительной техники**

отчет

**по лабораторной работе №1**

**по дисциплине «Параллельные вычисления»**

Тема: «Запуск параллельной **программы»**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 1307 |  | Грунская Н.Д. |
| Преподаватель |  | Манжиков Л.П. |

Санкт-Петербург

2025

**Цель работы.**

Освоить процесс запуска программы на С++ с применением библиотеки MPICH2. Научиться получать сведения о количестве запущенных процессов и номере отдельного процесса.

**Задание 1**. Создать и запустить программу на 2-х процессах с применением функций int MPI\_Init( int\* argc, char\*\*\* argv) и int MPI\_Finalize( void ).

Текст программы task1.cpp.

#include <stdio.h>

#include "mpi.h"

#include <sys/types.h>

#include <unistd.h>

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

{

int rank, size;

MPI\_Init(&argc, &argv);

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

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

printf("Size:%i, Process rank:%i, Start \n", size,rank);

printf("Size:%i, Process rank:%i, End \n", size,rank);

MPI\_Finalize();

return 0;

}

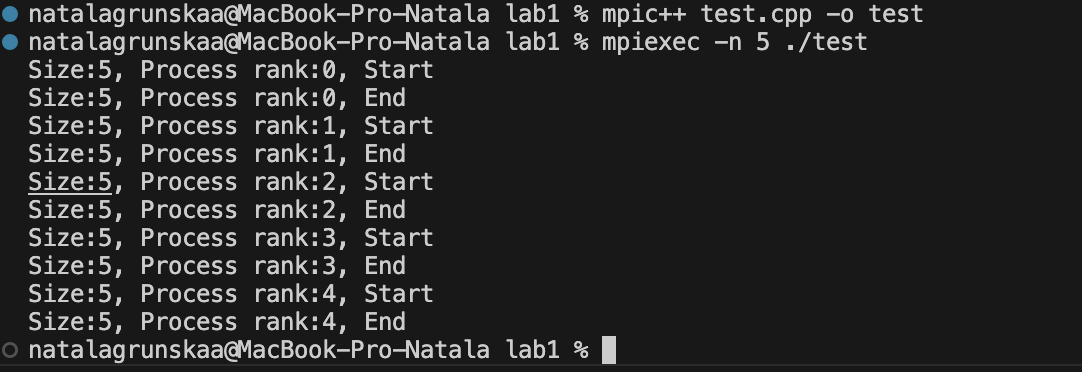


Рисунок 1. Запуск программы на 5-х процессах.

**Задание 2.** Создать и запустить программу на 3-х процессах с применением функций:

1) int MPI\_Init( int\* argc, char\*\*\* argv);

2) int MPI\_Finalize( void );

3) int MPI\_Comm\_size( MPI\_Comm comm, int\* size)

4) int MPI\_Comm\_rank( MPI\_Comm comm, int\* rank)

Программа должна выводить на экран номер процесса и какой-либо идентификатор процесса.

Текст программы task2.cpp

#include <cstdlib>

#include <iostream>

#include <unistd.h>

#include "mpi.h"

#define MAX 10

using namespace std;

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

{

int rank, size, n, i, ibeg, iend;

MPI\_Init(&argc, &argv);

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

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

printf ("process %d, pid: %d\n", rank, getpid());

MPI\_Finalize();

return 0;

}

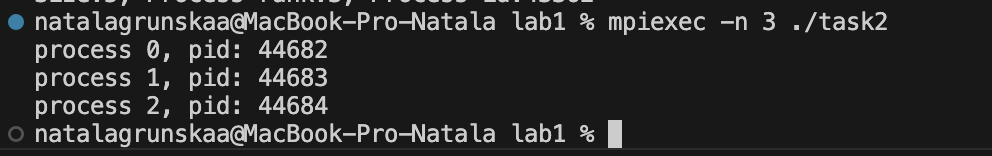


Рисунок 2. Запуск программы на 3-х процессах

**Задание 3.** Создать и запустить программу на n-х процессах печати таблицы умножения.

Текст программы task3.cpp

#include <cstdlib>

#include <iostream>

#include "mpi.h"

#define MAX 10

using namespace std;

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

{

int rank, size, n, i,j, ibeg, iend;

MPI\_Init(&argc, &argv);

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

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

n=(MAX-1)/size+1;

ibeg=rank\*n+1;

iend=(rank+1)\*n;

for(j=1; j<=10; j++){

printf ("process %d, %d ^ %d = %d \n", rank, rank, j,rank\*j);}

printf ("\n");

MPI\_Finalize();

return 0;

}

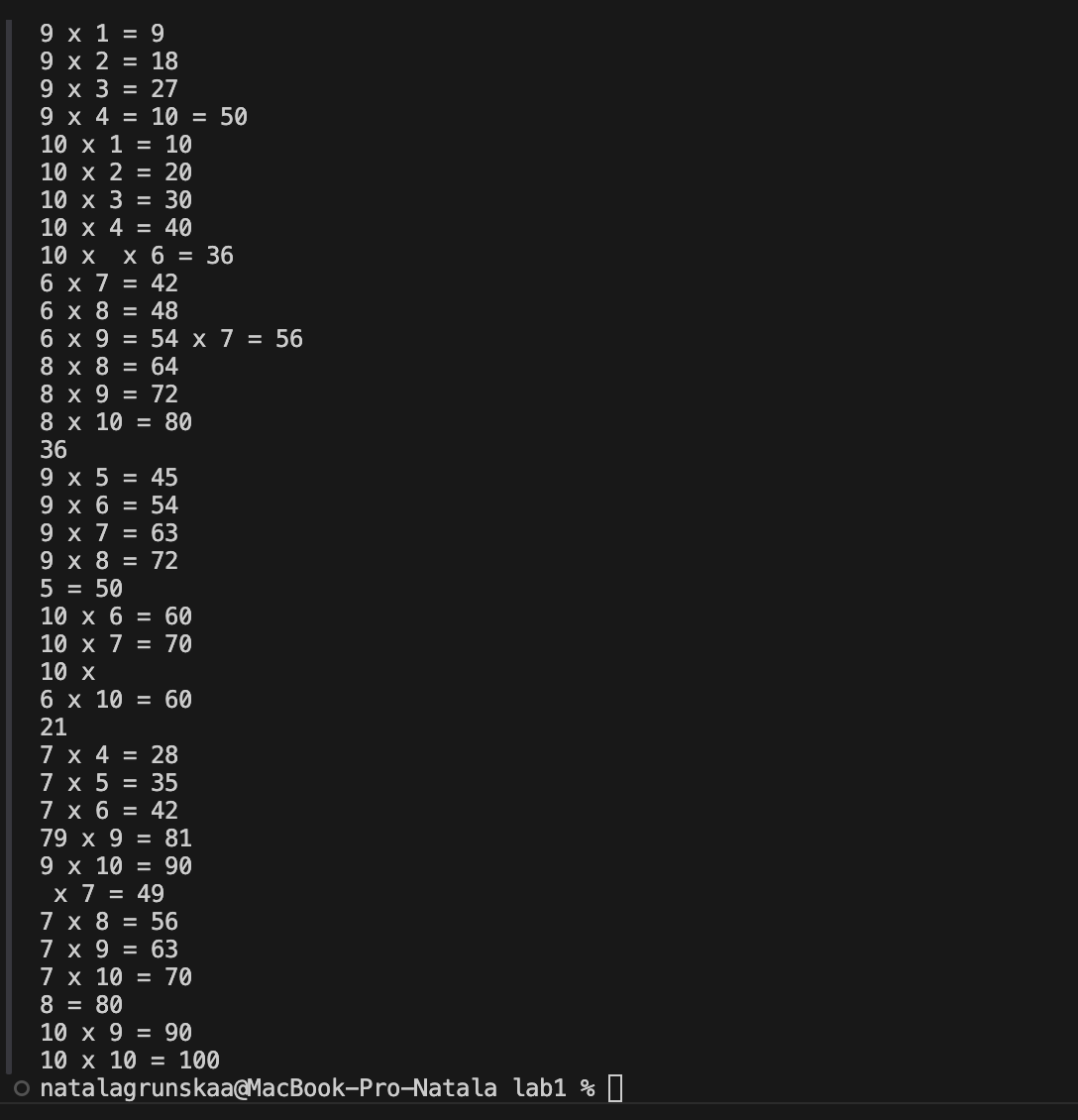
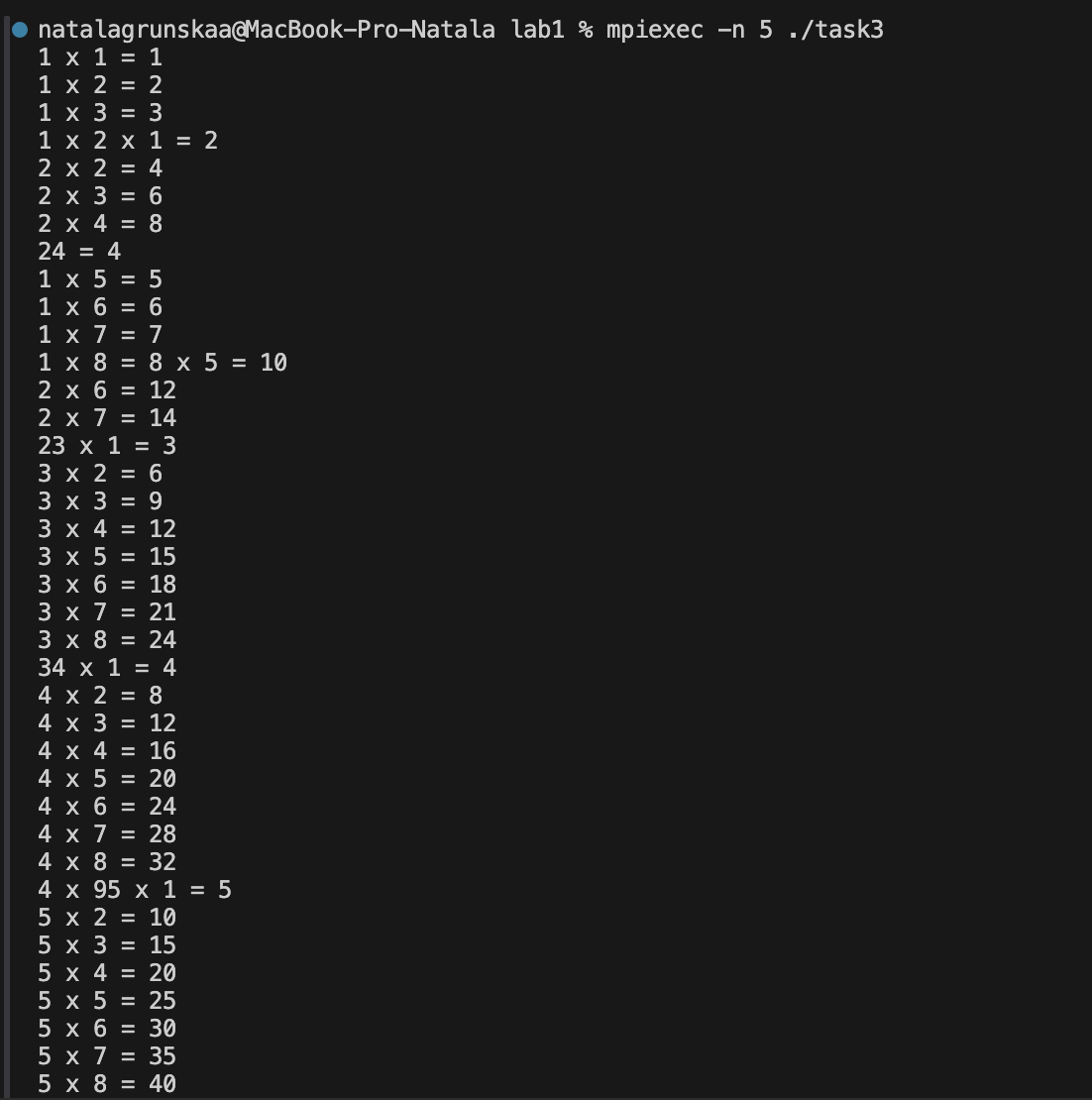


Рисунок 3. Запуск программы на 5-х процессах.

**Выводы.**

Лабораторная работа позволила освоить базовые принципы параллельного программирования с использованием MPI. Полученные навыки инициализации процессов, распределения задач и управления ими являются фундаментом для разработки сложных распределенных приложений, требующих оптимизации вычислений за счет параллелизма.