

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 C++ 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;
}
```

```

● natalagrunskaa@MacBook-Pro-Natala lab1 % mpic++ test.cpp -o test
● natalagrunskaa@MacBook-Pro-Natala lab1 % mpiexec -n 5 ./test
Size:5, Process rank:0, Start
Size:5, Process rank:0, End
Size:5, Process rank:1, Start
Size:5, Process rank:1, End
Size:5, Process rank:2, Start
Size:5, Process rank:2, End
Size:5, Process rank:3, Start
Size:5, Process rank:3, End
Size:5, Process rank:4, Start
Size:5, Process rank:4, End
○ natalagrunskaa@MacBook-Pro-Natala lab1 %

```

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;

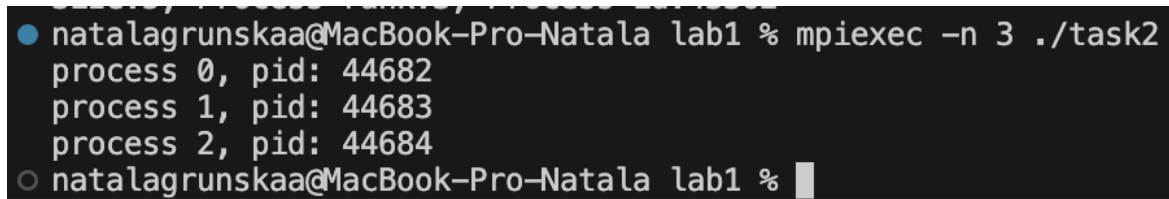
int main(int argc, char *argv[])
{
    int rank, size, n, i, ibeg, iend;
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);

```

```

printf ("process %d, pid: %d\n", rank, getpid());
MPI_Finalize();
return 0;
}

```



```

natalagrunskaa@MacBook-Pro-Natala lab1 % mpiexec -n 3 ./task2
process 0, pid: 44682
process 1, pid: 44683
process 2, pid: 44684
natalagrunskaa@MacBook-Pro-Natala lab1 %

```

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;
}

```

```
● natalagrunskaa@MacBook-Pro-Natala lab1 % mpiexec -n 2 ./task3
process 0, 0 ^ 1 = 0
process 0, 0 ^ 2 = 0
process 0, 0 ^ 3 = 0
process 0, 0 ^ 4 = 0
process 0, 0 ^ 5 = 0
process 0, 0 ^ 6 = 0
process 0, 0 ^ 7 = 0
process 0, 0 ^ 8 = 0
process 0, 0 ^ 9 = 0
process 0, 0 ^ 10 = 0

process 1, 1 ^ 1 = 1
process 1, 1 ^ 2 = 2
process 1, 1 ^ 3 = 3
process 1, 1 ^ 4 = 4
process 1, 1 ^ 5 = 5
process 1, 1 ^ 6 = 6
process 1, 1 ^ 7 = 7
process 1, 1 ^ 8 = 8
process 1, 1 ^ 9 = 9
process 1, 1 ^ 10 = 10

○ natalagrunskaa@MacBook-Pro-Natala lab1 %
```

Рисунок 3. Запуск программы на 2-х процессах.

```
● natalagrunskaa@MacBook-Pro-Natala lab1 % mpiexec -n 3 ./task3
process 0, 0 ^ 1 = 0
process 0, 0 ^ 2 = 0
process 0, 0 ^ 3 = 0
process 0, 0 ^ 4 = 0
process 0, 0 ^ 5 = 0
process 0, 0 ^ 6 = 0
process 0, 0 ^ 7 = 0
process 0, 0 ^ 8 = 0
process 0, 0 ^ 9 = 0
process 0, 0 ^ 10 = 0

process 1, 1 ^ 1 = 1
process 1, 1 ^ 2 = 2
process 1, 1 ^ 3 = 3
process 1, 1 ^ 4 = 4
process 1, 1 ^ 5 = 5
process 1, 1 ^ 6 = 6
process 1, 1 ^ 7 = 7
process 1, 1 ^ 8 = 8
process 1, 1 ^ 9 = 9
process 1, 1 ^ 10 = 10

process 2, 2 ^ 1 = 2
process 2, 2 ^ 2 = 4
process 2, 2 ^ 3 = 6
process 2, 2 ^ 4 = 8
process 2, 2 ^ 5 = 10
process 2, 2 ^ 6 = 12
process 2, 2 ^ 7 = 14
process 2, 2 ^ 8 = 16
process 2, 2 ^ 9 = 18
process 2, 2 ^ 10 = 20

○ natalagrunskaa@MacBook-Pro-Natala lab1 %
```

Рисунок 4 Запуск программы на 3-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.