

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

**ЛАБОРАТОРНАЯ РАБОТА №1**

**по дисциплине «Параллельные алгоритмы и системы»**

**Тема: ЗАПУСК ПАРАЛЛЕЛЬНОЙ ПРОГРАММЫ И ПЕРЕДАЧА**  
**ДАННЫХ ПО ПРОЦЕССАМ**

Студент гр. 1307

Угрюмов М.М.

Преподаватель

Санкт-Петербург

2024

## Введение

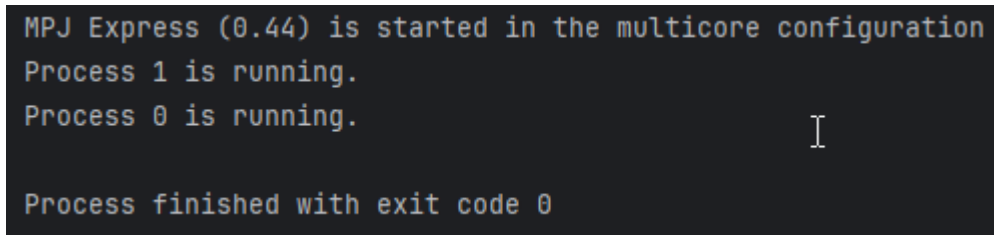
**Тема работы:** запуск параллельной программы и передача данных по процессам.

**Цель работы:** освоить процесс запуска программы с применением библиотеки MPICH2. Научиться получать сведения о количестве запущенных процессов и номере отдельного процесса. Освоить функции передачи данных между процессами.

### Лабораторная работа №1

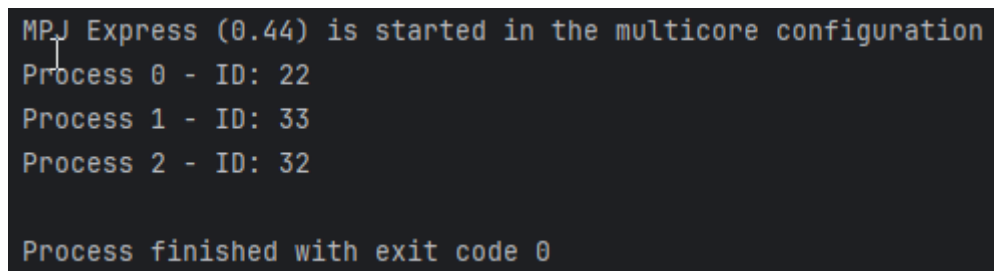
Задания:

- 1) Создать и запустить программу на 2-х процессах с применением функций `int MPI_Init( int* argc, char*** argv)` и `int MPI_Finalize( void )`.
- 2) Создать и запустить программу на 3-х процессах, Программа должна выводить на экран номер процесса и какой-либо идентификатор процесса.
- 3) Создать и запустить программу на n-х процессах печати таблицы умножения.

A screenshot of a terminal window with a dark background and light-colored text. The text shows the output of an MPI program running on two processes. It starts with 'MPJ Express (0.44) is started in the multicore configuration', followed by 'Process 1 is running.' and 'Process 0 is running.' on separate lines. After a cursor is visible, the final line reads 'Process finished with exit code 0'.

```
MPJ Express (0.44) is started in the multicore configuration
Process 1 is running.
Process 0 is running.
Process finished with exit code 0
```

Рис.1 – Результат выполнения задания №1

A screenshot of a terminal window with a dark background and light-colored text. The text shows the output of an MPI program running on three processes. It starts with 'MPJ Express (0.44) is started in the multicore configuration', followed by three lines: 'Process 0 - ID: 22', 'Process 1 - ID: 33', and 'Process 2 - ID: 32'. The final line reads 'Process finished with exit code 0'.

```
MPJ Express (0.44) is started in the multicore configuration
Process 0 - ID: 22
Process 1 - ID: 33
Process 2 - ID: 32
Process finished with exit code 0
```

Рис.2 – Результат выполнения задания №2

Результаты выполнения задания №3 представлены ниже, с учетом того, что параметр -np = 4, то есть 4 процесса.

MPJ Express (0.44) is started in the multicore configuration

Process 1 (ID: 25):  $1 \times 3 = 3$

Process 1 (ID: 25):  $1 \times 7 = 7$

Process 1 (ID: 25):  $2 \times 1 = 2$

Process 1 (ID: 25):  $2 \times 5 = 10$

Process 1 (ID: 25):  $2 \times 9 = 18$

Process 1 (ID: 25):  $3 \times 3 = 9$

Process 1 (ID: 25):  $3 \times 7 = 21$

Process 1 (ID: 25):  $4 \times 1 = 4$

Process 1 (ID: 25):  $4 \times 5 = 20$

Process 1 (ID: 25):  $4 \times 9 = 36$

Process 1 (ID: 25):  $5 \times 3 = 15$

Process 1 (ID: 25):  $5 \times 7 = 35$

Process 1 (ID: 25):  $6 \times 1 = 6$

Process 1 (ID: 25):  $6 \times 5 = 30$

Process 1 (ID: 25):  $6 \times 9 = 54$

Process 1 (ID: 25):  $7 \times 3 = 21$

Process 1 (ID: 25):  $7 \times 7 = 49$

Process 1 (ID: 25):  $8 \times 1 = 8$

Process 1 (ID: 25):  $8 \times 5 = 40$

Process 1 (ID: 25):  $8 \times 9 = 72$

Process 1 (ID: 25):  $9 \times 3 = 27$

Process 1 (ID: 25):  $9 \times 7 = 63$

Process 1 (ID: 25):  $10 \times 1 = 10$

Process 1 (ID: 25):  $10 \times 5 = 50$

Process 1 (ID: 25):  $10 \times 9 = 90$

Process 2 (ID: 24):  $1 \times 4 = 4$

Process 2 (ID: 24):  $1 \times 8 = 8$

Process 2 (ID: 24):  $2 \times 2 = 4$

Process 2 (ID: 24):  $2 \times 6 = 12$

Process 2 (ID: 24):  $2 \times 10 = 20$

Process 2 (ID: 24):  $3 \times 4 = 12$

Process 2 (ID: 24):  $3 \times 8 = 24$

Process 2 (ID: 24):  $4 \times 2 = 8$

Process 3 (ID: 23):  $1 \times 1 = 1$

Process 3 (ID: 23):  $1 \times 5 = 5$

Process 3 (ID: 23):  $1 \times 9 = 9$

Process 3 (ID: 23):  $2 \times 3 = 6$

Process 3 (ID: 23):  $2 \times 7 = 14$

Process 3 (ID: 23):  $3 \times 1 = 3$

Process 3 (ID: 23):  $3 \times 5 = 15$

Process 3 (ID: 23):  $3 \times 9 = 27$

Process 3 (ID: 23):  $4 \times 3 = 12$

Process 3 (ID: 23):  $4 \times 7 = 28$

Process 3 (ID: 23):  $5 \times 1 = 5$

Process 3 (ID: 23):  $5 \times 5 = 25$

Process 3 (ID: 23):  $5 \times 9 = 45$

Process 3 (ID: 23):  $6 \times 3 = 18$

Process 3 (ID: 23):  $6 \times 7 = 42$

Process 3 (ID: 23):  $7 \times 1 = 7$

Process 3 (ID: 23):  $7 \times 5 = 35$

Process 3 (ID: 23):  $7 \times 9 = 63$

Process 3 (ID: 23):  $8 \times 3 = 24$

Process 3 (ID: 23):  $8 \times 7 = 56$

Process 3 (ID: 23):  $9 \times 1 = 9$

Process 3 (ID: 23):  $9 \times 5 = 45$

Process 3 (ID: 23):  $9 \times 9 = 81$

Process 3 (ID: 23):  $10 \times 3 = 30$

Process 3 (ID: 23):  $10 \times 7 = 70$

Process 0 (ID: 22):  $1 \times 2 = 2$

Process 0 (ID: 22):  $1 \times 6 = 6$

Process 0 (ID: 22):  $1 \times 10 = 10$

Process 0 (ID: 22):  $2 \times 4 = 8$

Process 0 (ID: 22):  $2 \times 8 = 16$

Process 0 (ID: 22):  $3 \times 2 = 6$

Process 0 (ID: 22):  $3 \times 6 = 18$

Process 0 (ID: 22):  $3 \times 10 = 30$

Process 2 (ID: 24):  $4 \times 6 = 24$   
Process 2 (ID: 24):  $4 \times 10 = 40$   
Process 2 (ID: 24):  $5 \times 4 = 20$   
Process 2 (ID: 24):  $5 \times 8 = 40$   
Process 2 (ID: 24):  $6 \times 2 = 12$   
Process 2 (ID: 24):  $6 \times 6 = 36$   
Process 2 (ID: 24):  $6 \times 10 = 60$   
Process 2 (ID: 24):  $7 \times 4 = 28$   
Process 2 (ID: 24):  $7 \times 8 = 56$   
Process 2 (ID: 24):  $8 \times 2 = 16$   
Process 2 (ID: 24):  $8 \times 6 = 48$   
Process 2 (ID: 24):  $8 \times 10 = 80$   
Process 2 (ID: 24):  $9 \times 4 = 36$   
Process 2 (ID: 24):  $9 \times 8 = 72$   
Process 2 (ID: 24):  $10 \times 2 = 20$   
Process 2 (ID: 24):  $10 \times 6 = 60$   
Process 2 (ID: 24):  $10 \times 10 = 100$

Process 0 (ID: 22):  $4 \times 4 = 16$   
Process 0 (ID: 22):  $4 \times 8 = 32$   
Process 0 (ID: 22):  $5 \times 2 = 10$   
Process 0 (ID: 22):  $5 \times 6 = 30$   
Process 0 (ID: 22):  $5 \times 10 = 50$   
Process 0 (ID: 22):  $6 \times 4 = 24$   
Process 0 (ID: 22):  $6 \times 8 = 48$   
Process 0 (ID: 22):  $7 \times 2 = 14$   
Process 0 (ID: 22):  $7 \times 6 = 42$   
Process 0 (ID: 22):  $7 \times 10 = 70$   
Process 0 (ID: 22):  $8 \times 4 = 32$   
Process 0 (ID: 22):  $8 \times 8 = 64$   
Process 0 (ID: 22):  $9 \times 2 = 18$   
Process 0 (ID: 22):  $9 \times 6 = 54$   
Process 0 (ID: 22):  $9 \times 10 = 90$   
Process 0 (ID: 22):  $10 \times 4 = 40$   
Process 0 (ID: 22):  $10 \times 8 = 80$

При использовании следующего условия `if ((i * 10 + j) % size == rank)`, вычисления распределяются между процессами на основе их индексов, гарантируя, что каждый процесс получает примерно одинаковое кол-во работы, что позволяет использовать многозадачность более эффективно, а также убирает расчет дубликатов.

## Листинг программ

### Task1.java

```
import mpi.MPI;

public class Task1 {
    public static void main(String[] args) {
        MPI.Init(args);
        int rank = MPI.COMM_WORLD.Rank();
        if (rank < 2) {
            System.out.println("Process " + rank + " is running.");
        }
        MPI.Finalize();
    }
}
```

### Task2.java

```
import mpi.MPI;

public class Task2 {
    public static void main(String[] args) {
        MPI.Init(args);
        int size = MPI.COMM_WORLD.Size();
        int rank = MPI.COMM_WORLD.Rank();
        if (rank < 3) {
            System.out.println("Process " + rank + " of " + size + " - ID: " +
                               Thread.currentThread().getId());
        }
        MPI.Finalize();
    }
}
```

### Task3.java

```
import mpi.MPI;

public class Task3 {
    public static void main(String[] args) {
        MPI.Init(args);
        int rank = MPI.COMM_WORLD.Rank();
        int size = MPI.COMM_WORLD.Size();
        for (int i = 1; i <= 10; i++) {
            for (int j = 1; j <= 10; j++) {
                if ((i * 10 + j) % size == rank) {
                    System.out.printf("Process %d (ID: %d): %2d x %2d = %3d%n",
                                       rank, Thread.currentThread().getId(), i, j, i * j);
                }
            }
        }
        MPI.Finalize();
    }
}
```