

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

**ОТЧЕТ**  
**по лабораторной работе №00**  
**по дисциплине «ПА»**  
**Тема: Запуск параллельной программы на различном числе**  
**одновременно работающих процессов, упорядочение вывода**  
**результатов.**

Студент гр. 8304

\_\_\_\_\_

Сергеев А.Д.

Преподаватель

\_\_\_\_\_

Татаринов Ю.С.

Санкт-Петербург

2020

### **Задание.**

Запустить программу на 1, 2 ... N процессах несколько раз. Проанализировать порядок вывода сообщений на экран. Вывести правило, определяющее порядок вывода сообщений. Модифицировать программу таким образом, чтобы порядок вывода сообщений на экран соответствовал номеру соответствующего процесса.

### **Описание алгоритма.**

Так как процессы выполняются параллельно, а нулевой процесс принимает сообщения в любом порядке, порядок вывода сообщений на экран также случаен.

Для упорядочивания вывода в функцию отправки сообщения ненулевым процессом был добавлен тэг — номер этого процесса, а в функции получения сообщения нулевым процессом джокер ANY\_TAG, любой тэг, заменён на *i* — итератор цикла, проходящего по номерам всех процессов кроме нулевого. Таким образом сообщения отправляются всё так же в случайном порядке по мере завершения ненулевых процессов, а принимаются в порядке возрастания в цикле.

### **Листинг программы.**

```
#include <stdio.h>
#include <stdlib.h>
#include <mpi.h>

int main(int argc, char* argv[]) {
    int ProcNum, ProcRank, RecvRank;
    MPI_Status Status;

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &ProcNum);
    MPI_Comm_rank(MPI_COMM_WORLD, &ProcRank);

    if (ProcRank == 0) {
        printf ("Hello from process %d\n", ProcRank);
        for (int i = 1; i < ProcNum; i++) {
            MPI_Recv(&RecvRank, 1, MPI_INT, MPI_ANY_SOURCE, i, MPI_COMM_WORLD,
                &Status);
```

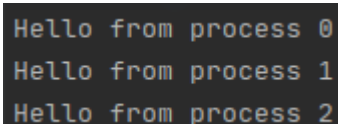
```

printf("Hello from process %d\n", RecvRank);
}
} else {
MPI_Send(&ProcRank,1,MPI_INT, 0, ProcRank, MPI_COMM_WORLD);
}

MPI_Finalize();
return 0;
}

```

### Результаты работы.

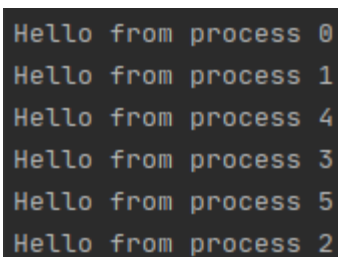


```

Hello from process 0
Hello from process 1
Hello from process 2

```

Рисунок 1 - Результат работы программы для трёх процессов до изменения программы

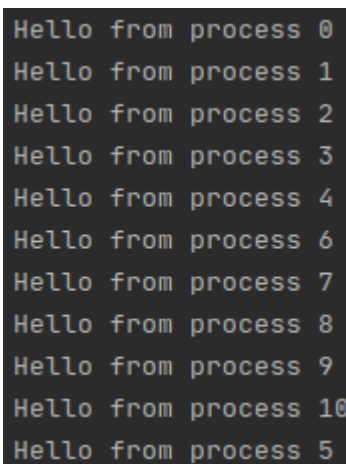


```

Hello from process 0
Hello from process 1
Hello from process 4
Hello from process 3
Hello from process 5
Hello from process 2

```

Рисунок 2 - Результат работы программы для шести процессов до изменения программы



```

Hello from process 0
Hello from process 1
Hello from process 2
Hello from process 3
Hello from process 4
Hello from process 6
Hello from process 7
Hello from process 8
Hello from process 9
Hello from process 10
Hello from process 5

```

Рисунок 3 - Результат работы программы для одиннадцати процессов до изменения программы

```
Hello from process 0  
Hello from process 1  
Hello from process 2
```

Рисунок 4 - Результат работы программы для трёх процессов после изменения программы

```
Hello from process 0  
Hello from process 1  
Hello from process 2  
Hello from process 3  
Hello from process 4  
Hello from process 5
```

Рисунок 5 - Результат работы программы для шести процессов после изменения программы

```
Hello from process 0  
Hello from process 1  
Hello from process 2  
Hello from process 3  
Hello from process 4  
Hello from process 5  
Hello from process 6  
Hello from process 7  
Hello from process 8  
Hello from process 9  
Hello from process 10
```

Рисунок 6 - Результат работы программы для одиннадцати процессов после изменения программы

### **Выводы.**

Программа проанализирована и изменена таким образом, чтобы результат одновременной работы нескольких процессов выводился на экран в желаемом порядке. До внесения изменений результат работы каждого процесса выводился на экран сразу после того, как сообщение было принято, без задержки. После внесения изменений результат работы процесса выводится на

экран после вывода результатов работы всех процессов с рангом ниже. То есть в первом случае время работы процесса зависело только от этого процесса, тогда как во втором — ещё и от всех процессов рангом ниже.