**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Параллельные алгоритмы»**

ТЕМА: **ИСПОЛЬЗОВАНИЕ ФУНКЦИЙ ОБМЕНА ДАННЫМИ «ТОЧКА-ТОЧКА» В БИБЛИОТЕКЕ MPI.**

|  |  |  |
| --- | --- | --- |
| Студент |  | Степаненко Д. В. |
| Преподаватель |  | Татаринов Ю. С. |

Санкт-Петербург

2023 г.

## Цель

Ознакомиться с функциями библиотеки MPI класса точка-точка. Написать программу с их использованием, избегая применения джокеров.

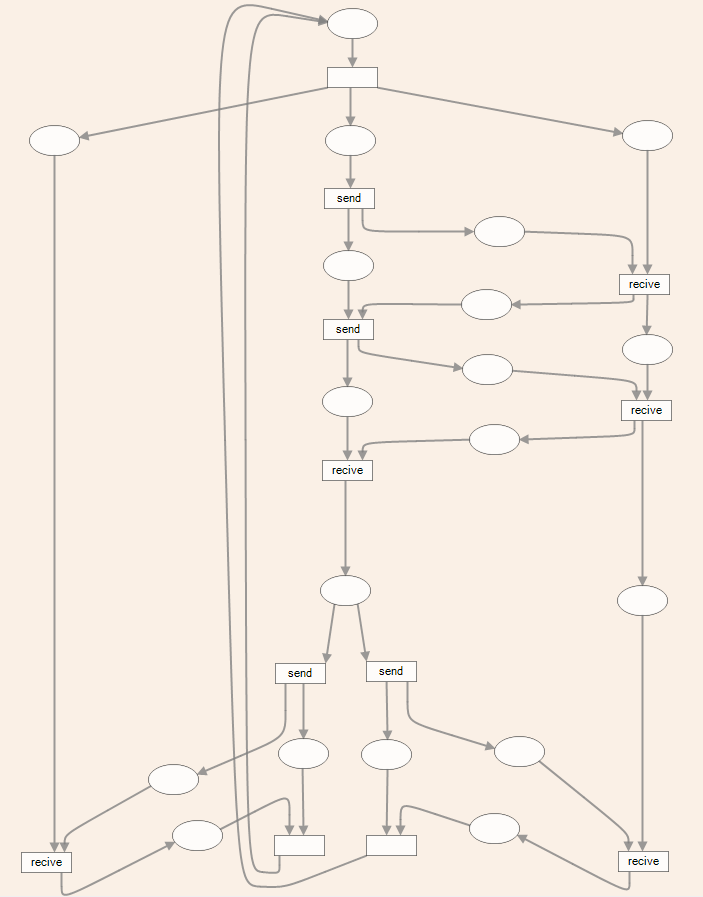
## Постановка задачи (вариант 10)

Морской бой. Несколько процессов поочередно обмениваются сообщениями согласно правилам игры в морской бой. (правила игры с числом игроков больше двух – устанавливаются самостоятельно)

## Выполнение работы

Программа создает несколько процессов, считывает ранг каждого и общее количество процессов. Запускает основной цикл игры, если количество процессов больше или рано 2. Размещает корабли на игровом поле. Далее генерируются псевдослучайные координаты и отправляются на следующий процесс (по кругу). После принятия координат, процесс проверяет на попадание по кораблю (вывод соответствующего сообщения). Далее проверяется условие выхода из игры: все корабли на карте должны быть уничтожены. Если условие выполнено, то процесс выставляет активный флаг завершения и отправляет его другим процессам. После завершения основного цикла завершается параллельная часть программы, освобождаются ресурсы.

Сеть Петри основной части алгоритма для трех процессов:



Листинг программы:

#include <stdio.h>

#include <stdlib.h>

#include <mpi.h>

#include <time.h>

#define FIELD\_SIZE 10

#define EMPTY\_CELL 1

#define SHIP\_CELL 2

#define MISS\_CELL 3

#define HIT\_CELL 4

#define GAME\_END 5

#define HIT\_TARGET 6

#define MISS\_TARGET 7

void place\_ships(int board[FIELD\_SIZE][FIELD\_SIZE]) {

for (int i = 0; i < FIELD\_SIZE; ++i) {

int x = rand() % FIELD\_SIZE;

int y = rand() % FIELD\_SIZE;

board[y][x] = SHIP\_CELL;

}

}

int check\_end(int board[FIELD\_SIZE][FIELD\_SIZE], int rank) {

for (int i = 0; i < FIELD\_SIZE; ++i) {

for (int j = 0; j < FIELD\_SIZE; ++j) {

if (board[i][j] == SHIP\_CELL) {

return 0;

}

}

}

printf("Игрок %d выбыл!", rank);

return 1;

}

void play\_game\_2(int rank, int size) {

int board[FIELD\_SIZE][FIELD\_SIZE];

place\_ships(board);

int end = 0;

int target\_send\_x, target\_send\_y;

int target\_recv\_x, target\_recv\_y;

// int next\_proc\_end;

int hit\_target = 0;

int result;

while (1) {

target\_send\_x = rand() %FIELD\_SIZE;

target\_send\_y = rand() %FIELD\_SIZE;

MPI\_Send(&target\_send\_x, 1, MPI\_INT, (rank + 1) % size, 0, MPI\_COMM\_WORLD);

MPI\_Send(&target\_send\_y, 1, MPI\_INT, (rank + 1) % size, 1, MPI\_COMM\_WORLD);

printf("target:%d %d (from %d to %d)\n",target\_send\_x, target\_send\_y, rank, (rank - 1 + size) % size);

MPI\_Recv(&target\_recv\_x, 1, MPI\_INT, (rank - 1 + size) % size, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

MPI\_Recv(&target\_recv\_y, 1, MPI\_INT, (rank - 1 + size) % size, 1, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

if (board[target\_recv\_x][target\_recv\_y] == SHIP\_CELL) {

hit\_target = HIT\_TARGET;

board[target\_recv\_x][target\_recv\_y] = HIT\_CELL;

}

else if(board[target\_recv\_x][target\_recv\_y] != SHIP\_CELL){

board[target\_recv\_x][target\_recv\_y] = MISS\_CELL;

hit\_target = MISS\_TARGET;

}

if (end != GAME\_END)

MPI\_Send(&hit\_target, 1, MPI\_INT, (rank - 1 + size) % size, 2, MPI\_COMM\_WORLD);

else {

MPI\_Send(&end, 1, MPI\_INT, (rank - 1 + size) % size, 2, MPI\_COMM\_WORLD);

break;

}

MPI\_Recv(&result, 1, MPI\_INT, (rank + 1) % size, 2, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

if(result==HIT\_TARGET){

printf("Hit in proc %d from proc %d!\n", rank, (rank + 1) % size);

}

else if (result==MISS\_TARGET){

printf("Miss in proc %d from proc %d!\n", rank, (rank + 1) % size);

}

if (check\_end(board, rank) || result == GAME\_END) {

end = GAME\_END;

break;

}

}

}

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

int rank, recv, size;

MPI\_Status status;

MPI\_Init(&argc, &argv);

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

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

if(size < 2){

printf("to few processes");

}

else if(size>=2){

play\_game\_2(rank, size);

}

MPI\_Finalize();

return 0;

}

Полученный вывод при запуске на 2-ух процессах:

target:1 8 (from 0 to 1)

Miss in proc 0 from proc 1!

target:7 9 (from 0 to 1)

Miss in proc 0 from proc 1!

target:2 0 (from 0 to 1)

Miss in proc 0 from proc 1!

target:2 3 (from 0 to 1)

Miss in proc 0 from proc 1!

target:7 5 (from 0 to 1)

Miss in proc 0 from proc 1!

target:9 2 (from 0 to 1)

Miss in proc 0 from proc 1!

target:2 8 (from 0 to 1)

target:1 8 (from 1 to 0)

Miss in proc 1 from proc 0!

target:7 9 (from 1 to 0)

Miss in proc 1 from proc 0!

target:2 0 (from 1 to 0)

Miss in proc 1 from proc 0!

target:2 3 (from 1 to 0)

Miss in proc 1 from proc 0!

target:7 5 (from 1 to 0)

Miss in proc 1 from proc 0!

target:9 2 (from 1 to 0)

Miss in proc 1 from proc 0!

target:2 8 (from 1 to 0)

Miss in proc 1 from proc 0!

target:9 7 (from 1 to 0)

Miss in proc 1 from proc 0!

target:3 6 (from 1 to 0)

Miss in proc 1 from proc 0!

target:1 2 (from 1 to 0)

Miss in proc 1 from proc 0!

target:9 3 (from 1 to 0)

Miss in proc 1 from proc 0!

target:1 9 (from 1 to 0)

Hit in proc 1 from proc 0!

target:4 7 (from 1 to 0)

Miss in proc 0 from proc 1!

target:9 7 (from 0 to 1)

Miss in proc 0 from proc 1!

target:3 6 (from 0 to 1)

Miss in proc 0 from proc 1!

target:1 2 (from 0 to 1)

Miss in proc 0 from proc 1!

target:9 3 (from 0 to 1)

Miss in proc 0 from proc 1!

target:1 9 (from 0 to 1)

Hit in proc 0 from proc 1!

target:4 7 (from 0 to 1)

Miss in proc 0 from proc 1!

target:8 4 (from 0 to 1)

Miss in proc 1 from proc 0!

target:8 4 (from 1 to 0)

Miss in proc 0 from proc 1!

target:5 0 (from 0 to 1)

Miss in proc 1 from proc 0!

target:5 0 (from 1 to 0)

Miss in proc 0 from proc 1!

target:3 6 (from 0 to 1)

Miss in proc 1 from proc 0!

target:3 6 (from 1 to 0)

Miss in proc 0 from proc 1!

target:1 0 (from 0 to 1)

Miss in proc 1 from proc 0!

target:1 0 (from 1 to 0)

Miss in proc 0 from proc 1!

target:6 3 (from 0 to 1)

Miss in proc 1 from proc 0!

target:6 3 (from 1 to 0)

Hit in proc 1 from proc 0!

target:2 0 (from 1 to 0)

Hit in proc 0 from proc 1!

target:2 0 (from 0 to 1)

Miss in proc 0 from proc 1!

target:6 1 (from 0 to 1)

Miss in proc 1 from proc 0!

target:6 1 (from 1 to 0)

Miss in proc 0 from proc 1!

Miss in proc 1 from proc 0!

target:5 5 (from 0 to 1)

target:5 5 (from 1 to 0)

Miss in proc 0 from proc 1!

target:4 7 (from 0 to 1)

Miss in proc 0 from proc 1!

target:6 5 (from 0 to 1)

Miss in proc 0 from proc 1!

target:6 9 (from 0 to 1)

Miss in proc 0 from proc 1!

target:3 7 (from 0 to 1)

Miss in proc 0 from proc 1!

target:4 5 (from 0 to 1)

Miss in proc 1 from proc 0!

target:4 7 (from 1 to 0)

Miss in proc 1 from proc 0!

target:6 5 (from 1 to 0)

Miss in proc 1 from proc 0!

target:6 9 (from 1 to 0)

Miss in proc 1 from proc 0!

target:3 7 (from 1 to 0)

Miss in proc 1 from proc 0!

target:4 5 (from 1 to 0)

Miss in proc 1 from proc 0!

target:2 5 (from 1 to 0)

Miss in proc 1 from proc 0!

target:4 7 (from 1 to 0)

Miss in proc 0 from proc 1!

target:2 5 (from 0 to 1)

Miss in proc 0 from proc 1!

target:4 7 (from 0 to 1)

Miss in proc 0 from proc 1!

target:4 4 (from 0 to 1)

Miss in proc 0 from proc 1!

target:3 0 (from 0 to 1)

Miss in proc 1 from proc 0!

target:4 4 (from 1 to 0)

Miss in proc 1 from proc 0!

target:3 0 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:7 8 (from 0 to 1)

Miss in proc 0 from proc 1!

target:6 8 (from 0 to 1)

target:7 8 (from 1 to 0)

Miss in proc 1 from proc 0!

target:6 8 (from 1 to 0)

Miss in proc 0 from proc 1!

target:8 4 (from 0 to 1)

Miss in proc 0 from proc 1!

Miss in proc 1 from proc 0!

target:8 4 (from 1 to 0)

Miss in proc 1 from proc 0!

target:3 1 (from 1 to 0)

target:3 1 (from 0 to 1)

Miss in proc 0 from proc 1!

target:4 9 (from 0 to 1)

Miss in proc 1 from proc 0!

target:4 9 (from 1 to 0)

Miss in proc 0 from proc 1!

target:2 0 (from 0 to 1)

Miss in proc 1 from proc 0!

target:2 0 (from 1 to 0)

Miss in proc 1 from proc 0!

target:6 8 (from 1 to 0)

Miss in proc 0 from proc 1!

target:6 8 (from 0 to 1)

Miss in proc 0 from proc 1!

Miss in proc 1 from proc 0!

target:9 2 (from 1 to 0)

target:9 2 (from 0 to 1)

Miss in proc 0 from proc 1!

target:6 6 (from 0 to 1)

Miss in proc 1 from proc 0!

target:6 6 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:4 9 (from 0 to 1)

Miss in proc 0 from proc 1!

target:4 9 (from 1 to 0)

Miss in proc 1 from proc 0!

target:5 0 (from 1 to 0)

target:5 0 (from 0 to 1)

Miss in proc 0 from proc 1!

target:4 8 (from 0 to 1)

Miss in proc 1 from proc 0!

target:4 8 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:7 1 (from 0 to 1)

target:7 1 (from 1 to 0)

Miss in proc 1 from proc 0!

target:7 2 (from 1 to 0)

Miss in proc 0 from proc 1!

target:7 2 (from 0 to 1)

Hit in proc 0 from proc 1!

Hit in proc 1 from proc 0!

target:7 2 (from 1 to 0)

target:7 2 (from 0 to 1)

Miss in proc 0 from proc 1!

target:2 6 (from 0 to 1)

Miss in proc 1 from proc 0!

target:2 6 (from 1 to 0)

Hit in proc 1 from proc 0!

target:1 0 (from 1 to 0)

Hit in proc 0 from proc 1!

target:1 0 (from 0 to 1)

Miss in proc 0 from proc 1!

Miss in proc 1 from proc 0!

target:6 1 (from 1 to 0)

target:6 1 (from 0 to 1)

Miss in proc 0 from proc 1!

target:5 9 (from 0 to 1)

Miss in proc 1 from proc 0!

target:5 9 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:4 9 (from 0 to 1)

Miss in proc 0 from proc 1!

target:4 9 (from 1 to 0)

Miss in proc 1 from proc 0!

target:0 9 (from 1 to 0)

target:0 9 (from 0 to 1)

Miss in proc 0 from proc 1!

target:1 7 (from 0 to 1)

Miss in proc 1 from proc 0!

target:1 7 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:7 1 (from 0 to 1)

Miss in proc 0 from proc 1!

target:7 1 (from 1 to 0)

Miss in proc 1 from proc 0!

target:1 5 (from 1 to 0)

target:1 5 (from 0 to 1)

Miss in proc 0 from proc 1!

Miss in proc 1 from proc 0!

target:9 7 (from 1 to 0)

target:9 7 (from 0 to 1)

Miss in proc 0 from proc 1!

target:7 6 (from 0 to 1)

Miss in proc 1 from proc 0!

target:7 6 (from 1 to 0)

Hit in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:7 3 (from 0 to 1)

target:7 3 (from 1 to 0)

Miss in proc 1 from proc 0!

target:6 5 (from 1 to 0)

Miss in proc 0 from proc 1!

target:6 5 (from 0 to 1)

Miss in proc 0 from proc 1!

target:6 3 (from 0 to 1)

Miss in proc 1 from proc 0!

target:6 3 (from 1 to 0)

Miss in proc 1 from proc 0!

target:9 4 (from 1 to 0)

Miss in proc 0 from proc 1!

target:9 4 (from 0 to 1)

Miss in proc 0 from proc 1!

Miss in proc 1 from proc 0!

target:8 1 (from 1 to 0)

target:8 1 (from 0 to 1)

Miss in proc 0 from proc 1!

target:2 9 (from 0 to 1)

Miss in proc 1 from proc 0!

target:2 9 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:3 9 (from 0 to 1)

Miss in proc 0 from proc 1!

target:3 9 (from 1 to 0)

Miss in proc 1 from proc 0!

target:0 8 (from 1 to 0)

target:0 8 (from 0 to 1)

Miss in proc 0 from proc 1!

target:8 5 (from 0 to 1)

Miss in proc 1 from proc 0!

target:8 5 (from 1 to 0)

Miss in proc 1 from proc 0!

target:0 9 (from 1 to 0)

Miss in proc 0 from proc 1!

target:0 9 (from 0 to 1)

Miss in proc 0 from proc 1!

target:6 3 (from 0 to 1)

Miss in proc 1 from proc 0!

target:6 3 (from 1 to 0)

Miss in proc 0 from proc 1!

target:8 5 (from 0 to 1)

Miss in proc 1 from proc 0!

target:8 5 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:6 1 (from 0 to 1)

Miss in proc 0 from proc 1!

target:6 1 (from 1 to 0)

Miss in proc 1 from proc 0!

target:1 5 (from 1 to 0)

target:1 5 (from 0 to 1)

Miss in proc 0 from proc 1!

target:9 8 (from 0 to 1)

Miss in proc 1 from proc 0!

target:9 8 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:4 8 (from 0 to 1)

target:4 8 (from 1 to 0)

Miss in proc 1 from proc 0!

target:1 0 (from 1 to 0)

Miss in proc 0 from proc 1!

target:1 0 (from 0 to 1)

Miss in proc 0 from proc 1!

target:3 0 (from 0 to 1)

Miss in proc 1 from proc 0!

target:3 0 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:4 4 (from 0 to 1)

Miss in proc 0 from proc 1!

target:4 4 (from 0 to 1)

Miss in proc 0 from proc 1!

target:7 6 (from 0 to 1)

target:4 4 (from 1 to 0)

Miss in proc 1 from proc 0!

target:4 4 (from 1 to 0)

Miss in proc 1 from proc 0!

target:7 6 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:3 1 (from 0 to 1)

Miss in proc 0 from proc 1!

target:3 1 (from 1 to 0)

Miss in proc 1 from proc 0!

target:7 5 (from 1 to 0)

Miss in proc 1 from proc 0!

target:9 6 (from 1 to 0)

target:7 5 (from 0 to 1)

Miss in proc 0 from proc 1!

target:9 6 (from 0 to 1)

Miss in proc 0 from proc 1!

target:2 1 (from 0 to 1)

Miss in proc 1 from proc 0!

target:2 1 (from 1 to 0)

Miss in proc 1 from proc 0!

target:7 8 (from 1 to 0)

Miss in proc 0 from proc 1!

target:7 8 (from 0 to 1)

Miss in proc 0 from proc 1!

target:5 7 (from 0 to 1)

Miss in proc 1 from proc 0!

target:5 7 (from 1 to 0)

Hit in proc 1 from proc 0!

target:4 1 (from 1 to 0)

Hit in proc 0 from proc 1!

target:4 1 (from 0 to 1)

Miss in proc 0 from proc 1!

target:8 5 (from 0 to 1)

Miss in proc 1 from proc 0!

target:8 5 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:9 7 (from 0 to 1)

Miss in proc 0 from proc 1!

target:5 3 (from 0 to 1)

target:9 7 (from 1 to 0)

Miss in proc 1 from proc 0!

target:5 3 (from 1 to 0)

Hit in proc 0 from proc 1!

target:8 8 (from 0 to 1)

Hit in proc 1 from proc 0!

target:8 8 (from 1 to 0)

Miss in proc 1 from proc 0!

target:3 1 (from 1 to 0)

Miss in proc 0 from proc 1!

target:3 1 (from 0 to 1)

Miss in proc 0 from proc 1!

target:8 9 (from 0 to 1)

Miss in proc 1 from proc 0!

target:8 9 (from 1 to 0)

Miss in proc 1 from proc 0!

target:6 4 (from 1 to 0)

Miss in proc 0 from proc 1!

target:6 4 (from 0 to 1)

Miss in proc 0 from proc 1!

target:3 3 (from 0 to 1)

Miss in proc 0 from proc 1!

Miss in proc 1 from proc 0!

target:3 3 (from 1 to 0)

Miss in proc 1 from proc 0!

target:3 8 (from 1 to 0)

target:3 8 (from 0 to 1)

Miss in proc 0 from proc 1!

target:6 0 (from 0 to 1)

Hit in proc 0 from proc 1!

target:4 8 (from 0 to 1)

Miss in proc 1 from proc 0!

target:6 0 (from 1 to 0)

Hit in proc 1 from proc 0!

target:4 8 (from 1 to 0)

Miss in proc 1 from proc 0!

target:8 8 (from 1 to 0)

Miss in proc 0 from proc 1!

target:8 8 (from 0 to 1)

Miss in proc 0 from proc 1!

target:9 7 (from 0 to 1)

Miss in proc 1 from proc 0!

target:9 7 (from 1 to 0)

Miss in proc 1 from proc 0!

target:7 6 (from 1 to 0)

Miss in proc 0 from proc 1!

target:7 6 (from 0 to 1)

Miss in proc 0 from proc 1!

target:4 3 (from 0 to 1)

Miss in proc 0 from proc 1!

target:0 3 (from 0 to 1)

Miss in proc 1 from proc 0!

target:4 3 (from 1 to 0)

Miss in proc 1 from proc 0!

target:0 3 (from 1 to 0)

Miss in proc 1 from proc 0!

target:0 9 (from 1 to 0)

Miss in proc 0 from proc 1!

target:0 9 (from 0 to 1)

Miss in proc 0 from proc 1!

target:2 5 (from 0 to 1)

Miss in proc 0 from proc 1!

Miss in proc 1 from proc 0!

target:2 5 (from 1 to 0)

Miss in proc 1 from proc 0!

target:4 0 (from 1 to 0)

Miss in proc 1 from proc 0!

target:4 0 (from 0 to 1)

Miss in proc 0 from proc 1!

target:5 9 (from 0 to 1)

Miss in proc 0 from proc 1!

target:5 9 (from 1 to 0)

Miss in proc 1 from proc 0!

target:4 6 (from 1 to 0)

target:4 6 (from 0 to 1)

Miss in proc 0 from proc 1!

target:9 2 (from 0 to 1)

Miss in proc 1 from proc 0!

target:9 2 (from 1 to 0)

Miss in proc 1 from proc 0!

target:2 4 (from 1 to 0)

Miss in proc 0 from proc 1!

target:2 4 (from 0 to 1)

Miss in proc 0 from proc 1!

Miss in proc 1 from proc 0!

target:7 7 (from 1 to 0)

target:7 7 (from 0 to 1)

Miss in proc 0 from proc 1!

target:5 4 (from 0 to 1)

Miss in proc 1 from proc 0!

target:5 4 (from 1 to 0)

Miss in proc 1 from proc 0!

target:8 1 (from 1 to 0)

Miss in proc 0 from proc 1!

target:8 1 (from 0 to 1)

Miss in proc 0 from proc 1!

target:2 8 (from 0 to 1)

Miss in proc 1 from proc 0!

target:2 8 (from 1 to 0)

Miss in proc 1 from proc 0!

target:9 3 (from 1 to 0)

Miss in proc 0 from proc 1!

target:9 3 (from 0 to 1)

Miss in proc 0 from proc 1!

target:6 8 (from 0 to 1)

Miss in proc 1 from proc 0!

target:6 8 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:0 2 (from 0 to 1)

target:0 2 (from 1 to 0)

Miss in proc 0 from proc 1!

target:1 0 (from 0 to 1)

Miss in proc 1 from proc 0!

target:1 0 (from 1 to 0)

Miss in proc 0 from proc 1!

target:5 1 (from 0 to 1)

Miss in proc 1 from proc 0!

target:5 1 (from 1 to 0)

Miss in proc 0 from proc 1!

target:1 0 (from 0 to 1)

Miss in proc 1 from proc 0!

target:1 0 (from 1 to 0)

Miss in proc 0 from proc 1!

target:8 5 (from 0 to 1)

Miss in proc 1 from proc 0!

target:8 5 (from 1 to 0)

Miss in proc 0 from proc 1!

target:0 6 (from 0 to 1)

Miss in proc 0 from proc 1!

Miss in proc 1 from proc 0!

target:0 6 (from 1 to 0)

Miss in proc 1 from proc 0!

target:4 6 (from 1 to 0)

target:4 6 (from 0 to 1)

Miss in proc 0 from proc 1!

Miss in proc 1 from proc 0!

target:2 5 (from 1 to 0)

Miss in proc 1 from proc 0!

target:2 5 (from 0 to 1)

Miss in proc 0 from proc 1!

target:8 6 (from 0 to 1)

Miss in proc 0 from proc 1!

target:8 6 (from 1 to 0)

Miss in proc 1 from proc 0!

target:2 8 (from 0 to 1)

target:2 8 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:4 7 (from 0 to 1)

Miss in proc 0 from proc 1!

target:4 7 (from 1 to 0)

Miss in proc 1 from proc 0!

target:2 4 (from 0 to 1)

target:2 4 (from 1 to 0)

Miss in proc 1 from proc 0!

Miss in proc 0 from proc 1!

target:0 6 (from 0 to 1)

Miss in proc 0 from proc 1!

target:0 6 (from 1 to 0)

Miss in proc 1 from proc 0!

target:2 9 (from 1 to 0)

Miss in proc 1 from proc 0!

target:9 0 (from 1 to 0)

Hit in proc 1 from proc 0!

target:8 1 (from 1 to 0)

Miss in proc 1 from proc 0!

target:3 1 (from 1 to 0)

Miss in proc 1 from proc 0!

target:1 0 (from 1 to 0)

Miss in proc 1 from proc 0!

target:3 4 (from 1 to 0)

Miss in proc 1 from proc 0!

target:0 3 (from 1 to 0)

Miss in proc 1 from proc 0!

target:9 1 (from 1 to 0)

Miss in proc 1 from proc 0!

target:9 6 (from 1 to 0)

Miss in proc 1 from proc 0!

target:9 3 (from 1 to 0)

Miss in proc 1 from proc 0!

target:3 8 (from 1 to 0)

Miss in proc 1 from proc 0!

target:0 5 (from 1 to 0)

Miss in proc 1 from proc 0!

target:6 6 (from 1 to 0)

Miss in proc 1 from proc 0!

target:4 0 (from 1 to 0)

Miss in proc 1 from proc 0!

target:0 4 (from 1 to 0)

Miss in proc 1 from proc 0!

target:6 2 (from 1 to 0)

Hit in proc 1 from proc 0!

Игрок 1 выбыл!target:2 9 (from 0 to 1)

Miss in proc 0 from proc 1!

target:9 0 (from 0 to 1)

Hit in proc 0 from proc 1!

target:8 1 (from 0 to 1)

Miss in proc 0 from proc 1!

target:3 1 (from 0 to 1)

Miss in proc 0 from proc 1!

target:1 0 (from 0 to 1)

Miss in proc 0 from proc 1!

target:3 4 (from 0 to 1)

Miss in proc 0 from proc 1!

target:0 3 (from 0 to 1)

Miss in proc 0 from proc 1!

target:9 1 (from 0 to 1)

Miss in proc 0 from proc 1!

target:9 6 (from 0 to 1)

Miss in proc 0 from proc 1!

target:9 3 (from 0 to 1)

Miss in proc 0 from proc 1!

target:3 8 (from 0 to 1)

Miss in proc 0 from proc 1!

target:0 5 (from 0 to 1)

Miss in proc 0 from proc 1!

target:6 6 (from 0 to 1)

Miss in proc 0 from proc 1!

target:4 0 (from 0 to 1)

Miss in proc 0 from proc 1!

target:0 4 (from 0 to 1)

Miss in proc 0 from proc 1!

target:6 2 (from 0 to 1)

Hit in proc 0 from proc 1!

Игрок 0 выбыл!

|  |  |
| --- | --- |
| Количество процессов (шт) | Среднее затрачиваемое время (мс) |
| 2 | 0,3115 |
| 4 | 0,4583 |
| 8 | 0,9082 |
| 12 | 1,1001 |
| 13 | 804,4069 |
| 14 | 1009,7896 |
| 15 | 1619,9837 |
| 16 | 2289,4688 |
| 24 | 5419,8323 |
| 32 | 4302,5810 |

Табл. 1 – Результаты работы программы на разном количестве процессов.

Расчеты ускорения программы выполним по формуле:

|  |  |
| --- | --- |
| Количество процессов P (шт) | Ускорение |
| 2 | 1 |
| 4 | 0,679686 |
| 8 | 0,342986 |
| 12 | 0,283156 |
| 13 | 0,000387 |
| 14 | 0,000308 |
| 15 | 0,000192 |
| 16 | 0,000136 |
| 24 | 5,75E-05 |
| 32 | 7,24E-05 |

Табл. 2 – Результаты расчетов ускорения программы.

## Вывод

В ходе выполнения лабораторной работы были изучены основные функции класса точка-точка из библиотеки MPI, способы пересылки и принятия сообщений без использования джокеров. Также были улучшены навыки по построению сетей Петри.

Результатом работы является реализованный алгоритм автономной игры в морской бой, где процессы обмениваются псевдослучайными координатами.

Скорость выполнения программного кода сильно зависит от количества процессов, т.к. при увеличении числа игроков (процессов) увеличивается и количество ходов не более чем на 100 (размеры доски). Следовательно растет трафик пересылаемых сообщений. Таким образом, время выполнения программы растет с увеличением числа процессов, а ускорение программы падает.

.