

Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

**Курсовая работа по курсу**  
**«Операционные системы»**

Группа: М80-206Б-20

Студент: Шипилова Т.П.

Преподаватель: Миронов Е.С.

Оценка: \_\_\_\_\_

Дата: 06.03.24

Москва, 2024

# **Постановка задачи**

## **Цель курсового проекта**

- Приобретение практических навыков в использовании знаний, полученных в течении курса
- Проведение исследования в выбранной предметной области

## **Задание**

Необходимо спроектировать и реализовать программный прототип в соответствии с выбранным вариантом. Произвести анализ и сделать вывод на основании данных, полученных при работе программного прототипа.

### **Вариант 1.**

Консоль-серверная игра. Необходимо написать консоль-серверную игру. Необходимо написать 2 программы: сервер и клиент. Сначала запускается сервер, а далее клиенты соединяются с сервером. Сервер координирует клиентов между собой.

### **Описание задания варианта.**

Морской бой. Общение между сервером и клиентом необходимо организовать при помощи pipe'ов. Каждый игрок должен при запуске ввести свой логин. Для каждого игрока должна вестись статистика игр (сколько побед/поражений). Игрок может посмотреть свою статистику.

## Общий метод и алгоритм решения

*Сценарий игры* (по сути, каждый этап - есть состояние):

1. Игроки по очереди вписывают ники (для статистики)
2. Игроки расставляют корабли, дают программе файлы с полем (считаем игроков умными, они не делают ошибок в расстановке)
3. Протекание самой игры с графическим интерфейсом
4. Если завершение боя, то обновляем статистику
5. Завершение работы программы

Интерфейс выводим после каждого хода

```
1 2 3 4 5 6 7 8 9 10  1 2 3 4 5 6 7 8 9 10
A ----- A -----
B ----- B -- O -----
C -- X ----- C -----
D ----- D -----
E ----- E -----
F ----- O -- F -----
G ----- G ----- X ---
H ----- H -----
J ----- J -----
K ----- K -----
```

X - ранение, O - попадание в пустоту

Со стороны сервера идет вся обработка и игра

Общение между сервером и клиентом организуется с помощью pipe`ов.

Формат: клиент<=>клиент, клиент<=>сервер.

Расстановку кораблей храним в массиве.

Корабли:

##### x1

### x2

## x3

# x4

Всего 20 координат.

Для каждого игрока по счетчику убитых. Если счетчик равен 20 у одного из игроков, то завершаем игру.

*Формат статистики* (храним в файле):

username\_1 1 2

username\_2 2 1

Первое число - количество побед, второе - поражений.

Если игрок есть в списке, то его строку редактируем. Если нет, то добавляем нового в конец (не по алфавиту).

## Код программы

Код представлен в репозитории на github: <https://github.com/TPShipilova/OS3sem/tree/main/cp>

Ниже представлен код файла **server.cpp**

```
#include <sys/wait.h>

#include "lib/inc/GameManager.hpp"
#include "lib/inc/utils.hpp"

#define CLIENT1_IN "in_0"
#define CLIENT1_OUT "out_0"

#define CLIENT2_IN "in_1"
#define CLIENT2_OUT "out_1"

int main(void) {

    std::cout << "=== BATTLESHIPS GAME SERVER ===" << std::endl;

    std::cout << "PID [" << getpid() << "]" << std::endl;

    bool is_running = true;

    pid_t clients[2] = {-1};

    role_t roles[2] = {ROLE_DEFEND, ROLE_ATTACK};

    int input_fd[2] = {0, 0};

    int pipe1_send[2];

    create_pipe(pipe1_send);

    int pipe1_recv[2];

    create_pipe(pipe1_recv);
```

```

int pipe2_send[2];

create_pipe(pipe2_send);

int pipe2_recv[2];

create_pipe(pipe2_recv);


clients[0] = create_child();

if(clients[0] == 0) {

    close(pipe1_recv[READ]);

    close(pipe1_send[WRITE]);


    close(pipe2_recv[READ]);

    close(pipe2_recv[WRITE]);

    close(pipe2_send[READ]);

    close(pipe2_send[WRITE]);


    int fd[2];

    create_pipe(fd);

    input_fd[0] = fd[WRITE];


    bind_fd(fd[READ], STDIN_FILENO);

    std::cout << "[INFO]: Use fd " << input_fd[0] << " for input PID " <<
getpid() << std::endl;

    bind_fd(open_file(CLIENT1_OUT), STDOUT_FILENO);


    const char* pr = std::to_string(pipe1_send[READ]).c_str();

    const char* pw = std::to_string(pipe1_recv[WRITE]).c_str();


    execl("./client", "./client", pr, pw, "0", NULL);

}

else {

```

```
clients[1] = create_child();

if(clients[1] == 0) {

    close(pipe2_recv[READ]);

    close(pipe2_send[WRITE]);


    close(pipe1_recv[READ]);

    close(pipe1_recv[WRITE]);

    close(pipe1_send[READ]);

    close(pipe1_send[WRITE]);


    int fd[2];

    create_pipe(fd);

    input_fd[1] = fd[WRITE];


    bind_fd(fd[READ], STDIN_FILENO);

    std::cout << "[INFO]: Use fd " << input_fd[1] << " for input PID " <<
getpid() << std::endl;

    bind_fd(open_file(CLIENT2_OUT), STDOUT_FILENO);


    const char* pr = std::to_string(pipe2_send[READ]).c_str();

    const char* pw = std::to_string(pipe2_recv[WRITE]).c_str();


    execl("./client", "./client", pr, pw, "1", NULL);

}

else {

    close(pipe1_send[READ]);

    close(pipe1_recv[WRITE]);

    close(pipe2_send[READ]);

    close(pipe2_recv[WRITE]);

}
```

```

}

std::cout << "[INFO]: Created clients with PIDs " << clients[0] << " and " <<
clients[1] << std::endl;

char* buf = new char[2];

read(pipe1_recv[READ], buf, 2);

if(buf[1] == 'r')

    std::cout << "[INFO]: Client 1 is ready -- OK" << std::endl;

read(pipe2_recv[READ], buf, 2);

if(buf[1] == 'r')

    std::cout << "[INFO]: Client 2 is ready -- OK" << std::endl;

delete[] buf;

while(true) {

    char* buf = new char[2];

    if(roles[0] == ROLE_ATTACK) {

        read(pipe1_recv[READ], buf, 2);

        write(pipe2_send[WRITE], buf, 2);

        read(pipe2_recv[READ], buf, 2);

        if(buf[1] == 'h') {

            std::cout << "Client 1 hit" << std::endl;

        }

        else if(buf[1] == 'm') {

            std::cout << "Client 1 missed" << std::endl;

            roles[0] = ROLE_DEFEND;

            roles[1] = ROLE_ATTACK;

```

```

    }

    else if(buf[1] == 'c') {

        std::cout << "Client 1 won" << std::endl;

        is_running = false;

    }

    write(pipe1_send[WRITE], buf, 2);

}

else {

    read(pipe2_recv[READ], buf, 2);

    write(pipe1_send[WRITE], buf, 2);

    read(pipe1_recv[READ], buf, 2);

    if(buf[1] == 'h') {

        std::cout << "Client 2 hit" << std::endl;

    }

    else if(buf[1] == 'm') {

        std::cout << "Client 2 missed" << std::endl;

        roles[1] = ROLE_DEFEND;

        roles[0] = ROLE_ATTACK;

    }

    else if(buf[1] == 'c') {

        std::cout << "Client 2 won" << std::endl;

        is_running = false;

    }

    write(pipe2_send[WRITE], buf, 2);

}

```



```
        delete[] buf;

        if(!is_running)
            break;
    }

    waitpid(clients[0], NULL, 0);
    waitpid(clients[1], NULL, 0);

    close(pipe1_recv[READ]);
    close(pipe1_send[WRITE]);
    close(pipe2_recv[READ]);
    close(pipe2_send[WRITE]);

    return EXIT_SUCCESS;
}
```

# Протокол работы программы

Для запуска:

```
cd build
```

```
cmake ..
```

```
cmake --build .
```

```
./battleships
```

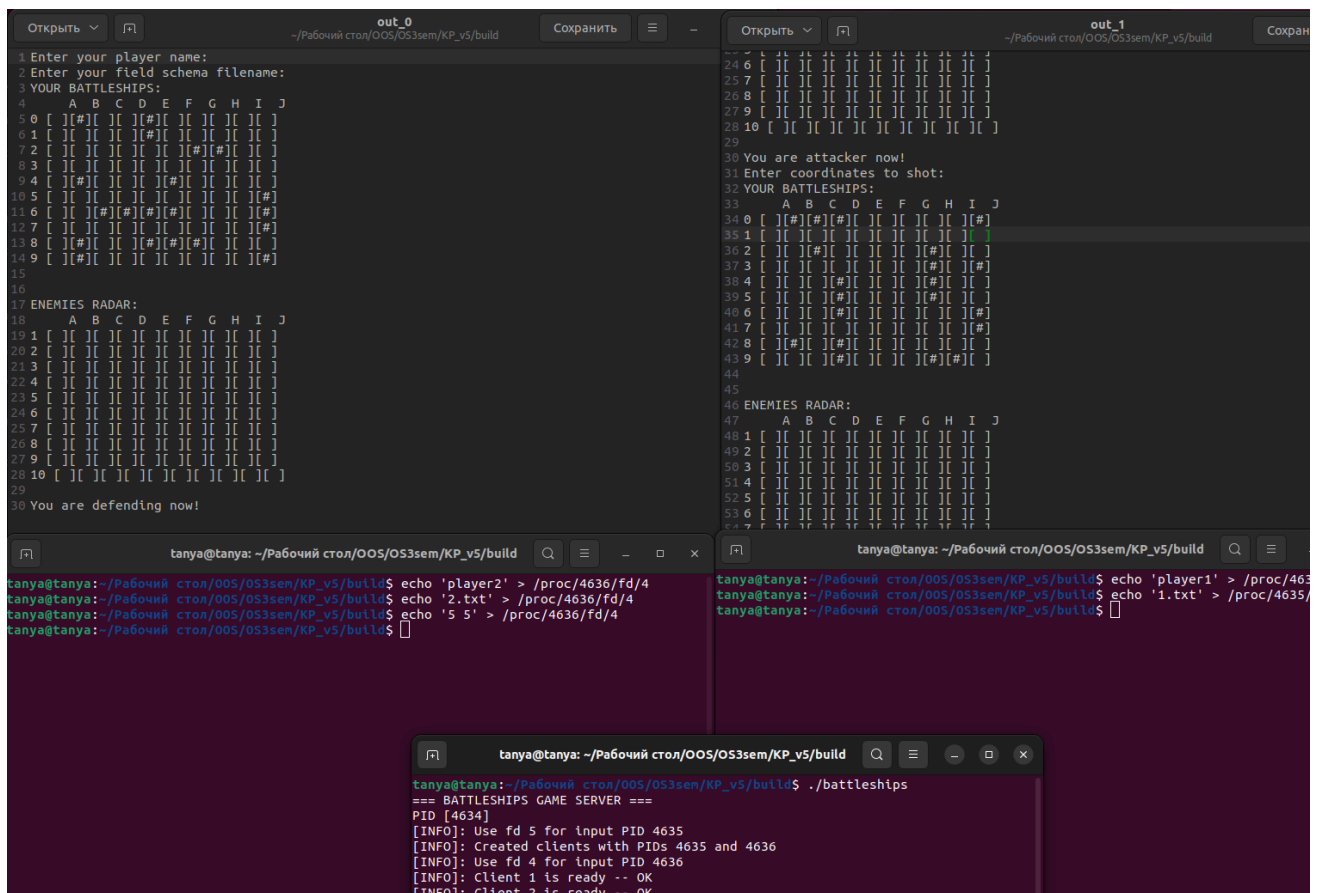
Для игры:

```
echo [сообщение] > /proc/[pid]/fd/[fd]
```

Пример:

```
echo '4 3' > /proc/4276/fd/4
```

## Демонстрация работы программы



```
1 Enter your player name:
2 Enter your field schema filename:
3 YOUR BATTLESHIPS:
4   A B C D E F G H I J
5 0 [ ][#][ ][ ][#][ ][ ][ ][ ][ ]
6 1 [ ][ ][ ][ ][#][ ][ ][ ][ ][ ]
7 2 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
8 3 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
9 4 [ ][#][ ][ ][ ][ ][ ][ ][ ][ ][ ]
10 5 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][#]
11 6 [ ][ ][#][ ][ ][ ][ ][ ][ ][ ][ ]
12 7 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][#]
13 8 [ ][#][ ][ ][ ][ ][ ][ ][ ][ ][ ]
14 9 [ ][#][ ][ ][ ][ ][ ][ ][ ][ ][#]
15
16
17 ENEMIES RADAR:
18   A B C D E F G H I J
19 1 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
20 2 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
21 3 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
22 4 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
23 5 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
24 6 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
25 7 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
26 8 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
27 9 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
28 10 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
29
30 You are defending now!

tanya@tanya: ~/Рабочий стол/OOS/OS3sem/KP_v5/build
tanya@tanya:~/Рабочий стол/OOS/OS3sem/KP_v5/build$ echo 'player2' > /proc/4636/fd/4
tanya@tanya:~/Рабочий стол/OOS/OS3sem/KP_v5/build$ echo '2.txt' > /proc/4636/fd/4
tanya@tanya:~/Рабочий стол/OOS/OS3sem/KP_v5/build$ echo '5 5' > /proc/4636/fd/4
tanya@tanya:~/Рабочий стол/OOS/OS3sem/KP_v5/build$

tanya@tanya:~/Рабочий стол/OOS/OS3sem/KP_v5/build
tanya@tanya:~/Рабочий стол/OOS/OS3sem/KP_v5/build$ echo 'player1' > /proc/4635/fd/4
tanya@tanya:~/Рабочий стол/OOS/OS3sem/KP_v5/build$ echo '1.txt' > /proc/4635/fd/4
tanya@tanya:~/Рабочий стол/OOS/OS3sem/KP_v5/build$

tanya@tanya:~/Рабочий стол/OOS/OS3sem/KP_v5/build$ ./battleships
=== BATTLESHIPS GAME SERVER ===
PID [4634]
[INFO]: Use fd 5 for input PID 4635
[INFO]: Created clients with PIDs 4635 and 4636
[INFO]: Use fd 4 for input PID 4636
[INFO]: client 1 is ready -- OK
[INFO]: client 2 is ready -- OK
```

Strace:

```
strace ./battleships
```

```
execve("./battleships", ["/battleships"], 0x7ffe14f529b0 /* 46 vars */) = 0
```

```
brk(NULL) = 0x55a4212fa000
```

```
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffc7cf3b540) = -1 EINVAL (Недопустимый аргумент)
```

```

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x7fdbc432000
access("/etc/ld.so.preload", R_OK) = -1 ENOENT (Нет такого файла или каталога)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=71527, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 71527, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fdbc420000
close(3) = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libstdc++.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2260296, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 2275520, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7fdbc000000
mprotect(0x7fdbc09a000, 1576960, PROT_NONE) = 0
mmap(0x7fdbc09a000, 1118208, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x9a000) = 0x7fdbc09a000
mmap(0x7fdbc1ab000, 454656, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1ab000) = 0x7fdbc1ab000
mmap(0x7fdbc21b000, 57344, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x21a000) = 0x7fdbc21b000
mmap(0x7fdbc229000, 10432, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7fdbc229000
close(3) = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libgcc_s.so.1", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=125488, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 127720, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7fdbc400000
mmap(0x7fdbc403000, 94208, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x3000) = 0x7fdbc403000
mmap(0x7fdbc41a000, 16384, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1a000) = 0x7fdbc41a000
mmap(0x7fdbc41e000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1d000) = 0x7fdbc41e000
close(3) = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0"..., 832) = 832

```

```

pread64(3, "\6\0\0\4\0\0\0@\0\0\0\0\0\0@\0\0\0\0\0\0@\0\0\0\0\0\0"..., 784, 64) = 784
pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\02\0\0\300\4\0\0\3\0\0\0\0\0\0"..., 48, 848) = 48
pread64(3,
"\4\0\0\0\24\0\0\0\3\0\0\0GNU\0302\211\332Pq\2439\235\350\223\322\257\201\326\243\f"..., 68, 896)
= 68

newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2220400, ...}, AT_EMPTY_PATH) = 0
pread64(3, "\6\0\0\4\0\0\0@\0\0\0\0\0\0@\0\0\0\0\0\0@\0\0\0\0\0\0"..., 784, 64) = 784
mmap(NULL, 2264656, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7fdbcbc00000
mprotect(0x7fdbcbc28000, 2023424, PROT_NONE) = 0
mmap(0x7fdbcbc28000, 1658880, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7fdbcbc28000
mmap(0x7fdbcbdbd000, 360448, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1bd000) = 0x7fdbcbdbd000
mmap(0x7fdbcbe16000, 24576, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x215000) = 0x7fdbcbe16000
mmap(0x7fdbcbe1c000, 52816, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7fdbcbe1c000
close(3) = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libm.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=940560, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 942344, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7fdbcc319000
mmap(0x7fdbcc327000, 507904, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe000) = 0x7fdbcc327000
mmap(0x7fdbcc3a3000, 372736, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x8a000) = 0x7fdbcc3a3000
mmap(0x7fdbcc3fe000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe4000) = 0x7fdbcc3fe000
close(3) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x7fdbcc317000
arch_pretl(ARCH_SET_FS, 0x7fdbcc3183c0) = 0
set_tid_address(0x7fdbcc318690) = 5311
set_robust_list(0x7fdbcc3186a0, 24) = 0
rseq(0x7fdbcc318d60, 0x20, 0, 0x53053053) = 0

```

```

mprotect(0x7fdbcbel6000, 16384, PROT_READ) = 0
mprotect(0x7fdbcc3fe000, 4096, PROT_READ) = 0
mprotect(0x7fdbcc41e000, 4096, PROT_READ) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x7fdbcc315000
mprotect(0x7fdbcc21b000, 45056, PROT_READ) = 0
mprotect(0x55a420b97000, 4096, PROT_READ) = 0
mprotect(0x7fdbcc46c000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) =
0
munmap(0x7fdbcc420000, 71527) = 0
getrandom("\x50\xe5\x09\xcc\x54\xa8\x1a\x3c", 8, GRND_NONBLOCK) = 8
brk(NULL) = 0x55a4212fa000
brk(0x55a42131b000) = 0x55a42131b000
futex(0x7fdbcc22977c, FUTEX_WAKE_PRIVATE, 2147483647) = 0
newfstatat(1, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x3), ...},
AT_EMPTY_PATH) = 0
write(1, "=== BATTLESHIPS GAME SERVER ===\n", 32=== BATTLESHIPS GAME SERVER
===
) = 32
getpid() = 5311
write(1, "PID [5311]\n", 11PID [5311]
) = 11
pipe2([3, 4], 0) = 0
pipe2([5, 6], 0) = 0
pipe2([7, 8], 0) = 0
pipe2([9, 10], 0) = 0
clone(child_stack=NULL,
flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,
child_tidptr=0x7fdbcc318690) = 5312
clone(child_stack=NULL,
flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD[INFO]: Use fd 5 for input
PID 5312
, child_tidptr=0x7fdbcc318690) = 5313
close(3) = 0
close(6) = 0

```

```

close(7)                = 0
close(10)               = 0
write(1, "[INFO]: Created clients with PID"..., 48[INFO]: Created clients with PIDs 5312 and 5313
) = 48
read(5, [INFO]: Use fd 4 for input PID 5313
"_r", 2)                = 2
write(1, "[INFO]: Client 1 is ready -- OK\n", 32[INFO]: Client 1 is ready -- OK
) = 32
terminate called after throwing an instance of 'std::invalid_argument'
read(9, what(): stoul
0x55a42130c2c0, 2)      = ? ERESTARTSYS (To be restarted if SA_RESTART is set)
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_DUMPED, si_pid=5312, si_uid=1000,
si_status=SIGABRT, si_utime=0, si_stime=0} ---
read(9, "_r", 2)        = 2
write(1, "[INFO]: Client 2 is ready -- OK\n", 32[INFO]: Client 2 is ready -- OK
) = 32

```

## Вывод

В ходе выполнения данной курсовой работы мною использовались знания и навыки, полученные в курсе Операционных систем. Для клиентов создаются дочерние процессы, а общение между клиентом-клиентом и клиентом-сервером организовано через pipe`ы. Со стороны клиентов игра управляется через терминал с использованием перенаправления ввода на процесс по дескриптору, а вывод клиентов перенаправлен в файлы 0\_out 1\_out с помощью dup2.