Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

**Курсовая работа**

**«Клиент-серверная система для передачи мгновенных сообщений на memory map»**

**по курсу**

**«Операционные системы»**

Группа: М80-206Б-22

Студент: Ларченко А.О.

Преподаватель: Миронов Е.С.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: 09.02.2024

Москва, 2024

**Постановка задачи**

Клиент-серверная система для передачи мгновенных сообщений. Базовый функционал должен быть следующим:

* Клиент может присоединиться к серверу, введя логин
* Клиент может отправить сообщение другому клиенту по его логину
* Клиент в реальном времени принимает сообщения от других клиентов

**Вариант 24.**

Необходимо предусмотреть возможность создания «групповых чатов». Связь между сервером и клиентом должна быть реализована при помощи memory map

**Общий метод и алгоритм решения**

Использованные системные вызовы:

* ***getpid***() - получение ID текущего процесса
* ***kill***(int pid, signal)- отправление сигнала signal процессу с ID pid
* ***signal***(int signum, sighandler\_t handler) - устанавливает новый обработчик сигнала с номером \_signum\_ в соответствии с параметром \_sighandler\_, который может быть функцией пользователя
* ***shm\_open***(const char \*name) ***-*** создает и открывает новый (или уже существующий) объект разделяемой памяти POSIX. Объект разделяемой памяти POSIX - это обработчик, используемый несвязанными процессами для исполнения) на одну область разделяемой памяти.
* ***shm\_unlink***(const char \*name) ***-*** снимает объекты разделяемой памяти
* ***ftruncate***(int fd, off\_t length) - устанавливают длину обычного файла с файловым дескриптором \_fd\_ в \_length\_ байт.
* ***mmap***(void \*start, size\_t length, int prot , int flags, int fd, off\_t offset) - отражает файл fd в память отражает \_length\_ байтов, начиная со смещения \_offset\_ файла (или другого объекта), определенного файловым описателем \_fd\_, в память, начиная с адреса \_start\_. Настоящее местоположение отраженных данных возвращается самой функцией mmap, и никогда не бывает равным 0.
* ***munmap***(void \*start, size\_t length) - удаляет все отражения из заданной области памяти, после чего все ссылки на данную область будут вызывать ошибку "неправильное обращение к памяти" (invalid memory reference). Отражение удаляется автоматически при завершении процесса. С другой стороны, закрытие файла не приведет к снятию отражения.

Идея программы очень простая, также как и её реализация. Параллельно у нас запущены 1 сервер и произвольное количество клиентов(различных терминалов). Каждый пользователь вводит сообщение из предоставленного ему набора, после чего сообщение записывается в отображаемую память и серверу посылается сигнал. Сервер обрабатывает сообщение, если все хорошо, пересылает его дальше по назначению, если же ошибка возвращает пользователю сообщение об ошибке.

**Код программы**

**m\_map.h**

#pragma once

#include <iostream>

#include <unistd.h>

#include <sys/mman.h>

#include <sys/stat.h>

#include <fcntl.h>

#include "msg.h"

class Memory\_map{

public:

int my\_pid;

int server\_pid;

Memory\_map(string fn\_wr, string fn\_re, string fn\_p);

bool write\_msg(Message &msg);

bool read\_msg(Message &msg);

bool write\_pid(Msg\_pid &msg);

bool read\_pid(Msg\_pid &msg);

void close\_sh\_file(string fn);

~Memory\_map();

string fn\_writing;

string fn\_reading;

string fn\_pid;

};

void writing(char \*to,const char \*from, int size);

string writing\_to\_str(char \*from, int size);

**m\_map.c**

#include "m\_map.h"

void writing(char \*to,const char \*from, int size){ // \*to and \*from have equal capacity

for(int i=0; i< size;++i){

if(from[i]==EOF or from[i]=='\0' or from[i]=='\n'){

to[i]='\0';

break;

}

to[i]=from[i];

}

to[size-1]='\0';

}

string writing\_to\_str(char \*from, int size){

string tmp;

for(int i=0; i<size;++i){

if(from[i]==EOF or from[i]=='\0' or from[i]=='\n'){

tmp+='\0';

break;

}

tmp+=from[i];

}

return tmp;

}

Memory\_map::Memory\_map(string fn\_wr, string fn\_re, string fn\_p){

fn\_writing=fn\_wr;

fn\_reading=fn\_re;

fn\_pid=fn\_p;

}

Memory\_map::~Memory\_map(){

}

bool Memory\_map::read\_pid(Msg\_pid &msg){

int shm\_fd=shm\_open( fn\_pid.c\_str(), O\_CREAT | O\_RDWR, S\_IRUSR | S\_IWUSR);

if(shm\_fd==-1){

if(shm\_unlink( fn\_pid.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

if(ftruncate(shm\_fd, sizeof(Msg\_pid))==-1){

if(shm\_unlink( fn\_pid.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

Msg\_pid \*msg\_ptr=(Msg\_pid\*)mmap(NULL, sizeof(Msg\_pid), PROT\_READ | PROT\_WRITE, MAP\_SHARED, shm\_fd, 0);

if(msg\_ptr==MAP\_FAILED){

if(shm\_unlink( fn\_pid.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

msg.Pid=msg\_ptr->Pid;

munmap(msg\_ptr, sizeof(Msg\_pid));

close(shm\_fd);

return true;

}

bool Memory\_map::write\_pid(Msg\_pid &msg){

int shm\_fd=shm\_open( fn\_pid.c\_str(), O\_CREAT | O\_RDWR, S\_IRUSR | S\_IWUSR);

if(shm\_fd==-1){

if(shm\_unlink( fn\_pid.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

if(ftruncate(shm\_fd, sizeof(Msg\_pid))==-1){

if(shm\_unlink( fn\_pid.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

Msg\_pid \*msg\_ptr=(Msg\_pid\*)mmap(NULL, sizeof(Msg\_pid), PROT\_READ | PROT\_WRITE, MAP\_SHARED, shm\_fd, 0);

if(msg\_ptr==MAP\_FAILED){

if(shm\_unlink( fn\_pid.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

msg\_ptr->Pid=msg.Pid;

munmap(msg\_ptr, sizeof(Msg\_pid));

close(shm\_fd);

return true;

}

bool Memory\_map::write\_msg(Message &msg){

int shm\_fd=shm\_open( fn\_writing.c\_str(), O\_CREAT | O\_RDWR, S\_IRUSR | S\_IWUSR);

if(shm\_fd==-1){

if(shm\_unlink( fn\_writing.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

if(ftruncate(shm\_fd, sizeof(Message))==-1){

if(shm\_unlink( fn\_writing.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

Message \*msg\_ptr=(Message\*)mmap(NULL, sizeof(Message), PROT\_READ | PROT\_WRITE, MAP\_SHARED, shm\_fd, 0);

if(msg\_ptr==MAP\_FAILED){

if(shm\_unlink( fn\_writing.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

msg\_ptr->type=msg.type;

msg\_ptr->pid=msg.pid;

writing( msg\_ptr->to, msg.to, NAMECAPACITY);

writing(msg\_ptr->usr\_from, msg.usr\_from, NAMECAPACITY);

writing(msg\_ptr->data, msg.data, DATACAPACITY);

munmap(msg\_ptr, sizeof(Message));

close(shm\_fd);

return true;

}

bool Memory\_map ::read\_msg(Message &msg){

int shm\_fd=shm\_open( fn\_reading.c\_str(), O\_CREAT | O\_RDWR, S\_IRUSR | S\_IWUSR);

if(shm\_fd==-1){

if(shm\_unlink( fn\_reading.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

if(ftruncate(shm\_fd, sizeof(Message))==-1){

if(shm\_unlink( fn\_reading.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

Message \*msg\_ptr=(Message\*)mmap(NULL, sizeof(Message), PROT\_READ | PROT\_WRITE, MAP\_SHARED, shm\_fd, 0);

if(msg\_ptr==MAP\_FAILED){

if(shm\_unlink( fn\_reading.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

msg.type=msg\_ptr->type;

msg.pid=msg\_ptr->pid;

writing(msg.to, msg\_ptr->to, NAMECAPACITY);

writing(msg.usr\_from, msg\_ptr->usr\_from, NAMECAPACITY);

writing(msg.data, msg\_ptr->data, DATACAPACITY);

munmap(msg\_ptr, sizeof(Message));

close(shm\_fd);

return true;

}

void Memory\_map::close\_sh\_file(string fn){

if(shm\_unlink(fn.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

}

**client.cpp**

#include <iostream>

#include "m\_map.h"

using namespace std;

string \_MSG\_SEND = SH\_OBJ\_MSG\_GET;

string \_MSG\_GET = SH\_OBJ\_MSG\_SEND;

string \_SERVER\_PID = SH\_OBJ\_SERVER\_PID;

pthread\_mutex\_t mutex;

static int wf=0;

static bool error\_exit=false;

void wait\_sig(int sig){

wf=1;

}

bool check\_name(string name){

if(name.size()>NAMECAPACITY){

return false;

}

for(int i=0;i<name.size();++i){

if (name[i]=='\n' or name[i]==' '){

return false;

}

}

return true;

}

void\* receiving(void\* args){

Memory\_map \*binder=(Memory\_map\*) args;

// sleep(1);

while(true){

Message msg;

// cout<<"Start waiting sig...\n";

signal(SIGUSR1, wait\_sig);

// signal(SIGUSR1, wait\_msg);

while(wf!=1){

sleep(0.1);

}

// cout<<"Got sig\n";

// sleep(0.2);

// pthread\_mutex\_lock(&mutex);

binder->read\_msg(msg);

if(msg.type==message\_type::\_error){

cout<<"Error:\n";

cout<<msg.usr\_from<<":"<<msg.data;

error\_exit=true;

break;

} else if(msg.type==message\_type::\_msg\_to\_chat){

cout<<"New message:\n"<<"Chat:"<<msg.to<<":"<<msg.usr\_from<<' '<<msg.data;

} else{

cout<<"New message:\n";

cout<<msg.usr\_from<<":"<<msg.data;

}

// fflush(NULL);

// pthread\_mutex\_unlock(&mutex);

wf=0;

}

}

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

if(argc!=2){

cout<<"Uncorrect input. Input your name!\n";

exit(-1);

}

string username = argv[1];

if(!check\_name(username)){

cout<<"Uncorrect username\n";

exit(-1);

}

Memory\_map binder(\_MSG\_SEND,\_MSG\_GET, \_SERVER\_PID );

Msg\_pid server\_pid;

binder.read\_pid(server\_pid);

if(server\_pid.Pid==0){

cout<<"Server is unavailable now...\n";

exit(-1);

}

int my\_pid = getpid();

binder.my\_pid=my\_pid;

binder.server\_pid=server\_pid.Pid;

cout<<"Conecting...\n";

Message init;

writing(init.usr\_from, username.c\_str(), username.size()+1);

init.type=message\_type::\_create;

init.pid=my\_pid;

// signal(SIGUSR1, wait\_sig);

pthread\_mutex\_init(&mutex, NULL);

pthread\_t receiver;

if(pthread\_create(&receiver, NULL, receiving, &binder)!=0){

perror("Create thread error ");

}

binder.write\_msg(init);

kill(server\_pid.Pid, SIGUSR1);

// usleep(2000);

sleep(2);

pthread\_mutex\_lock(&mutex);

cout<<"Server\_pid: "<<server\_pid.Pid<<'\n';

cout<<"Welcome in our chat, "<<username<<"!\n";

cout<<"Here you can communicate with other users directly or using chats\n\n";

cout<<"--For reading messages write:\n"<<"To:'other\_username' 'you message...'\n";

cout<<"To:chat:'chat\_name' 'you message...'\n"<<"-- Or if you want to join or creata a chat write:\n";

cout<<"Join:'chat\_name'\n";

cout<<"--Write 'q' to close terminal\n";

pthread\_mutex\_unlock(&mutex);

while(true){

Message msg;

msg.pid=my\_pid;

writing(msg.usr\_from, username.c\_str(), username.size()+1);

if(error\_exit){

msg.type=message\_type::\_error;

pthread\_cancel(receiver);

pthread\_detach(receiver);

binder.write\_msg(msg);

kill(server\_pid.Pid, SIGUSR1);

cout<<"Break;";

break;

}

string request;

cout<<" > ";

getline(cin, request);

bool currect=true;

if(request=="q" or cin.eof()){

msg.type=message\_type::\_exited;

pthread\_cancel(receiver);

pthread\_detach(receiver);

binder.write\_msg(msg);

kill(server\_pid.Pid, SIGUSR1);

cout<<"Break;\n";

break;

} else if(request.substr(0, 8)=="To:chat:"){

msg.type=message\_type::\_msg\_to\_chat;

int pos=request.find(" ");

string chat\_rec=request.substr(8, pos-8);

if(pos!= string::npos){

string data\_str=request.substr(pos+1, request.size()-(pos+1));

if(data\_str.size()==0){

cout<<"Uncorrect input\n";

currect=false;

} else{

writing(msg.to, chat\_rec.c\_str(), chat\_rec.size()+1);

writing(msg.data, data\_str.c\_str(), data\_str.size()+1);

}

} else{

cout<<"Uncorrect input\n";

currect=false;

}

} else if(request.substr(0, 3)=="To:"){

msg.type=message\_type::\_msg\_to\_usr;

int pos=request.find(" ");

string usr\_rec=request.substr(3, pos-3);

if(pos!= string::npos){

string data\_str=request.substr(pos+1, request.size()-(pos+1));

if(data\_str.size()==0){

cout<<"Uncorrect input\n";

currect=false;

} else{

writing(msg.to, usr\_rec.c\_str(), usr\_rec.size()+1);

writing(msg.data, data\_str.c\_str(), data\_str.size()+1);

}

} else{

cout<<"Uncorrect input\n";

currect=false;

}

} else if(request.substr(0, 5)=="Join:"){

msg.type=message\_type::\_join\_chat;

string chat\_name=request.substr(5, request.size()-5);

if(chat\_name.size()==0){

cout<<"Uncorrect input\n";

currect=false;

} else{

writing(msg.data, chat\_name.c\_str(), chat\_name.size()+1);

}

}else if(request==""){

cout<<'\n';

currect=false;

} else{

cout<<"Uncorrect input\n";

currect=false;

}

if(currect){

binder.write\_msg(msg);

kill(server\_pid.Pid, SIGUSR1);

sleep(1);

}

}

}

**server.cpp**

#include <iostream>

#include <map>

#include <string>

#include <vector>

#include "m\_map.h"

// #include "timer.h"

using namespace std;

string \_MSG\_SEND = SH\_OBJ\_MSG\_SEND;

string \_MSG\_GET = SH\_OBJ\_MSG\_GET;

string \_SERVER\_PID =SH\_OBJ\_SERVER\_PID;

static int check=0;

void wait\_read(int sig){

check=1;

}

void exit\_signal(int sig){

cout<<"Forced exit\n";

if(shm\_unlink(\_SERVER\_PID.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

if(shm\_unlink(\_MSG\_SEND.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

if(shm\_unlink(\_MSG\_GET.c\_str())==-1){

perror("munmap trouble: ");

exit(-1);

}

exit(1);

}

struct status{

int pid;

bool active;

};

bool is\_active\_server(map<string, status> &user\_dict){

int status=false;

for(const auto& [user, user\_stat]: user\_dict){

if(user\_stat.active){

status=true;

break;

}

}

return status;

}

bool is\_in\_chat(vector<string> &array, string str){

for(int i=0; i<array.size();++i){

if(array[i]==str){

return true;

}

}

return false;

}

void notify\_all\_in\_chat(vector<string> &chat\_members, string chat\_name, map<string, status> &user\_dict){

for(int i=0; i<chat\_members.size();++i){

int cur\_pid=user\_dict[chat\_members[i]].pid;

sleep(0.2);

kill(cur\_pid, SIGUSR1);

cout<<"Notify: "<<cur\_pid<<' '<<chat\_members[i]<<'\n';

sleep(0.2);

}

}

int main(){

map<string, vector<string>> chat\_dict;

map<string, status> user\_dict;

bool q\_server=false;

int my\_pid=getpid();

// cout<<"Server pid is "<<my\_pid<<"\n";

Memory\_map binder(\_MSG\_SEND, \_MSG\_GET, \_SERVER\_PID);

Msg\_pid server\_pid;

// server\_pid.Pid=my\_pid;

binder.read\_pid(server\_pid);

if(server\_pid.Pid!=0){

cout<<"Server has already run. Exit\n";

exit(-1);

} else{

cout<<"Server pid is "<<my\_pid<<"\n";

server\_pid.Pid=my\_pid;

}

binder.write\_pid(server\_pid);

signal(SIGINT, exit\_signal);

while(true){

Message msg;

signal(SIGUSR1, wait\_read);

while(check!=1){

sleep(0.1);

}

binder.read\_msg(msg);

cout<<"New\_msg: type: "<<msg.type<<" from: <"<<msg.usr\_from<<">"<<msg.pid<<"\n";

check=0;

Message error\_msg;

error\_msg.type=message\_type::\_error;

writing(error\_msg.to, msg.usr\_from, NAMECAPACITY);

Message answ\_msg;

answ\_msg.type=message\_type::\_server\_answer;

writing(answ\_msg.to, msg.usr\_from, NAMECAPACITY);

string answ;

const char\* sender="Server\0";

string username=writing\_to\_str(msg.usr\_from, NAMECAPACITY);

cout<<"To: "<<msg.to<<" Data: "<<msg.data<<"\n\n";

switch(msg.type){

case message\_type::\_create :{

sleep(0.1);

// sender="Server\0";

if(user\_dict.count(username)>0){

if(!user\_dict[username].active){

user\_dict[username].pid=msg.pid;

user\_dict[username].active=true;

// writing(answ\_msg.to, msg.usr\_from, NAMECAPACITY);

const char \*ok\_answ="OK: Successful logined!\0";

writing(answ\_msg.data, ok\_answ, 25);

} else{

const char \*bad\_answ="Error: User have already logined!\0";

writing(error\_msg.data, bad\_answ, 35);

writing(error\_msg.usr\_from, sender, NAMECAPACITY);

binder.write\_msg(error\_msg);

kill(msg.pid, SIGUSR1);

cout<<"Send error msg\n";

break;

}

} else{

user\_dict[username].pid=msg.pid;

user\_dict[username].active=true;

const char \*ok\_answ="OK: Successful created!\0";

writing(answ\_msg.data, ok\_answ, 25);

}

writing(answ\_msg.usr\_from, sender, NAMECAPACITY);

binder.write\_msg(answ\_msg);

kill(user\_dict[username].pid, SIGUSR1);

cout<<"Sending msg:"<<"from sender: "<<sender<<" to "<<msg.pid<<' '<<username<<'\n';

cout<<"Data: "<<answ\_msg.data<<'\n';

break;

}

case message\_type::\_exited :{

user\_dict[username].active=false; //проверка на количество активных узлов, если 0 - exit

if(!is\_active\_server(user\_dict)){

q\_server=true;

}

break;

}

case message\_type::\_join\_chat :{

string chat\_name=writing\_to\_str(msg.data, NAMECAPACITY);

// const char\* data\_msg;

string data\_msg\_str;

if(chat\_dict.count(chat\_name)>0){

if(is\_in\_chat(chat\_dict[chat\_name], username)){

data\_msg\_str="Error: you have already been in this chat\0";

writing(answ\_msg.data, data\_msg\_str.c\_str(), DATACAPACITY);

writing(answ\_msg.usr\_from, sender, NAMECAPACITY);

writing(answ\_msg.to, msg.usr\_from, NAMECAPACITY);

answ\_msg.pid=msg.pid;

binder.write\_msg(answ\_msg);

kill(answ\_msg.pid, SIGUSR1);

break;

}

//if not

sleep(0.1);

data\_msg\_str="New user has joined to chat.\0";

writing(answ\_msg.data, data\_msg\_str.c\_str(), DATACAPACITY);

answ\_msg.type=message\_type::\_msg\_to\_chat;

writing(answ\_msg.usr\_from, msg.usr\_from, NAMECAPACITY);

writing(answ\_msg.to, msg.data, NAMECAPACITY);

binder.write\_msg(answ\_msg);

notify\_all\_in\_chat(chat\_dict[chat\_name], chat\_name, user\_dict);

// cout<<"=============\_has joined to chat=================\n";

// cout<<"Data: "<<data\_msg\_str<<'\n';

} else{

answ\_msg.type=message\_type::\_msg\_to\_chat;

data\_msg\_str="Chat has been created successfully.";

writing(answ\_msg.data, data\_msg\_str.c\_str(), DATACAPACITY);

writing(answ\_msg.usr\_from, msg.usr\_from, NAMECAPACITY);

writing(answ\_msg.to, msg.data, NAMECAPACITY);

answ\_msg.pid=msg.pid;

binder.write\_msg(answ\_msg);

kill(answ\_msg.pid, SIGUSR1);

// cout<<"=============\_has created the chat=================\n";

// cout<<"Data: "<<data\_msg\_str<<'\n';

}

cout<<"Sending msg:"<<"from sender: "<<answ\_msg.usr\_from<<" to "<<' '<<msg.data<<'\n';

cout<<"Data: "<<answ\_msg.data<<'\n';

chat\_dict[chat\_name].push\_back(username);

break;

}

case message\_type::\_msg\_to\_usr :{

string usr\_to=writing\_to\_str(msg.to, NAMECAPACITY);

if(user\_dict.count(usr\_to)>0){

if(user\_dict[usr\_to].active){

writing(answ\_msg.usr\_from, msg.usr\_from, NAMECAPACITY);

writing(answ\_msg.data, msg.data, DATACAPACITY);

writing(answ\_msg.to, msg.to, NAMECAPACITY);

answ\_msg.pid=user\_dict[usr\_to].pid;

} else{

const char\* data\_msg="Error: User is inactive now";

writing(answ\_msg.to, msg.usr\_from, NAMECAPACITY);

writing(answ\_msg.usr\_from, sender, NAMECAPACITY);

writing(answ\_msg.data, data\_msg, 28);

answ\_msg.pid=msg.pid;

}

} else{

const char\* data\_msg="Error: No such user";

writing(answ\_msg.to, msg.usr\_from, NAMECAPACITY);

writing(answ\_msg.usr\_from, sender, NAMECAPACITY);

writing(answ\_msg.data, data\_msg, 20);

answ\_msg.pid=msg.pid;

}

binder.write\_msg(answ\_msg);

kill(answ\_msg.pid, SIGUSR1);

break;

}

case message\_type::\_msg\_to\_chat :{

string chat\_name=writing\_to\_str(msg.to, NAMECAPACITY);

string data\_msg\_str;

if(chat\_dict.count(chat\_name)>0){

if(is\_in\_chat(chat\_dict[chat\_name], username)){

sleep(0.1);

writing(answ\_msg.data, msg.data, DATACAPACITY);

answ\_msg.type=message\_type::\_msg\_to\_chat;

writing(answ\_msg.usr\_from, msg.usr\_from, NAMECAPACITY);

writing(answ\_msg.to, msg.to, NAMECAPACITY);

binder.write\_msg(answ\_msg);

notify\_all\_in\_chat(chat\_dict[chat\_name], chat\_name, user\_dict);

} else{

data\_msg\_str="Error: You aren't in this chat\0";

writing(answ\_msg.data, data\_msg\_str.c\_str(), DATACAPACITY);

writing(answ\_msg.usr\_from, sender, NAMECAPACITY);

writing(answ\_msg.to, msg.usr\_from, NAMECAPACITY);

answ\_msg.pid=msg.pid;

binder.write\_msg(answ\_msg);

kill(answ\_msg.pid, SIGUSR1);

}

} else{

data\_msg\_str="Error: No such chat exist\0";

writing(answ\_msg.data, data\_msg\_str.c\_str(), DATACAPACITY);

writing(answ\_msg.usr\_from, sender, NAMECAPACITY);

writing(answ\_msg.to, msg.usr\_from, NAMECAPACITY);

answ\_msg.pid=msg.pid;

binder.write\_msg(answ\_msg);

kill(answ\_msg.pid, SIGUSR1);

}

break;

}

case message\_type::\_error:{

cout<<"<"<<msg.usr\_from<<">"<<" was exit with error\n";

break;

}

default:

break;

}

if(q\_server){

cout<<"Close server. There are no active users\n";

break;

}

cout<<"Current number of users: "<<user\_dict.size()<<"\n\n";

}

binder.close\_sh\_file(binder.fn\_pid);

binder.close\_sh\_file(binder.fn\_reading);

binder.close\_sh\_file(binder.fn\_writing);

}

**Протокол работы программы**

**Тестирование:**

***Терминал 1(Сервер):***

arsenii@PC-Larcha14:~/Documents/VS\_code\_prog/OSI/KP$ ./server

Server pid is 45486

New\_msg: type: 0 from: <Arsenii>45489

To: Data:

Sending msg:from sender: Server to 45489 Arsenii

Data: OK: Successful created!

Current number of users: 1

New\_msg: type: 4 from: <Arsenii>45489

Data:

Close server. There are no active users

arsenii@PC-Larcha14:~/Documents/VS\_code\_prog/OSI/KP$ make

g++ m\_map.cpp server.cpp -o server

g++ m\_map.cpp client.cpp -o client

client.cpp: In function ‘void\* receiving(void\*)’:

client.cpp:64:1: warning: no return statement in function returning non-void [-Wreturn-type]

64 | }

| ^

arsenii@PC-Larcha14:~/Documents/VS\_code\_prog/OSI/KP$ ./server

Server pid is 51339

New\_msg: type: 0 from: <Arsenii>51340

To: Data:

Sending msg:from sender: Server to 51340 Arsenii

Data: OK: Successful created!

Current number of users: 1

New\_msg: type: 0 from: <Gg>51344

To: Data:

Sending msg:from sender: Server to 51344 Gg

Data: OK: Successful created!

Current number of users: 2

New\_msg: type: 2 from: <Gg>51344

To: Arsenii Data: HI!

Current number of users: 2

New\_msg: type: 2 from: <Gg>51344

To: Arsenii Data: How are you&

Current number of users: 2

New\_msg: type: 2 from: <Arsenii>51340

To: Gg Data: Hi! Nice I've ended my kp!!!!

Current number of users: 2

New\_msg: type: 2 from: <Arsenii>51340

To: OO Data: Hi

Current number of users: 2

New\_msg: type: 1 from: <Arsenii>51340

To: OO Data: new\_chat

Sending msg:from sender: Arsenii to new\_chat

Data: Chat has been created successfully.

Current number of users: 2

New\_msg: type: 3 from: <Gg>51344

To: new\_chat Data: ggggggggggggggg

Current number of users: 2

New\_msg: type: 1 from: <Gg>51344

To: new\_chat Data: new\_chat

Notify: 51340 Arsenii

Sending msg:from sender: Gg to new\_chat

Data: New user has joined to chat.

Current number of users: 2

New\_msg: type: 2 from: <Gg>51344

To: new\_chat Data: Hi

Current number of users: 2

New\_msg: type: 3 from: <Gg>51344

To: new\_chat Data: Hi

Notify: 51340 Arsenii

Notify: 51344 Gg

Current number of users: 2

New\_msg: type: 4 from: <Arsenii>51340

To: OO Data: new\_chat

Current number of users: 2

New\_msg: type: 4 from: <Gg>51344

To: new\_chat Data: Hi

Close server. There are no active users

***Терминал 2(Клиент):***

arsenii@PC-Larcha14:~/Documents/VS\_code\_prog/OSI/KP$ ./client Arsenii

Conecting...

Server\_pid: 42308

Welcome in our chat, Arsenii!

Here you can communicate with other users directly or using chats

--For reading messages write:

To:'other\_username' 'you message...'

To:chat:'chat\_name' 'you message...'

-- Or if you want to join or creata a chat write:

Join:'chat\_name'

--Write 'q' to close terminal

> New message:

Server:OK: Successful created!

> q

Break;

arsenii@PC-Larcha14:~/Documents/VS\_code\_prog/OSI/KP$ ./client Arsenii

Server is unavailable now...

arsenii@PC-Larcha14:~/Documents/VS\_code\_prog/OSI/KP$ ./client Arsenii

Conecting...

Server\_pid: 51339

Welcome in our chat, Arsenii!

Here you can communicate with other users directly or using chats

--For reading messages write:

To:'other\_username' 'you message...'

To:chat:'chat\_name' 'you message...'

-- Or if you want to join or creata a chat write:

Join:'chat\_name'

--Write 'q' to close terminal

> New message:

Server:OK: Successful created!

> New message:

Gg:HI!

> New message:

Gg:How are you&

> To:Gg Hi! Nice I've ended my kp!!!!

> To:OO Hi

> New message:

Server:Error: No such user

> To:Gg

Uncorrect input

>

> Join:new\_chat

> New message:

Chat:new\_chat:Arsenii Chat has been created successfully.

> New message:

Chat:new\_chat:Gg New user has joined to chat.

>

> New message:

Chat:new\_chat:Gg Hi

> q

Break;

***Терминал 3(Клиент):***

arsenii@PC-Larcha14:~/Documents/VS\_code\_prog/OSI/KP$ ./client Gg

Conecting...

Server\_pid: 51339

Welcome in our chat, Gg!

Here you can communicate with other users directly or using chats

--For reading messages write:

To:'other\_username' 'you message...'

To:chat:'chat\_name' 'you message...'

-- Or if you want to join or creata a chat write:

Join:'chat\_name'

--Write 'q' to close terminal

> New message:

Server:OK: Successful created!

> To:Arsenii HI!

> To:Arsenii How are you&

> New message:

Arsenii:Hi! Nice I've ended my kp!!!!

> To:chat:new\_chat

Uncorrect input

>

> To:chat:new\_chat ggggggggggggggg

> New message:

Server:Error: You aren't in this chat

> Join:new\_chat

> To:new\_chat Hi

> New message:

Server:Error: No such user

> To:chat:new\_chat Hi

> New message:

Chat:new\_chat:Gg Hi

> q

Break;

==========================================================================================

**Strace:**

***server:*** arsenii@PC-Larcha14:~/Documents/VS\_code\_prog/OSI/KP$ strace -e trace=\!clock\_nanosleep -oserver\_log.log ./server

execve("./server", ["./server"], 0x7fff5a10a308 /\* 56 vars \*/) = 0

brk(NULL) = 0x5628ba8b3000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffdee8f3440) = -1 EINVAL (Invalid argument)

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f7cc5e03000

access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=81715, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 81715, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f7cc5def000

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\0\3\0>\0\1\0\0\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) = 0x7f7cc5a00000

mprotect(0x7f7cc5a9a000, 1576960, PROT\_NONE) = 0

mmap(0x7f7cc5a9a000, 1118208, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x9a000) = 0x7f7cc5a9a000

mmap(0x7f7cc5bab000, 454656, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1ab000) = 0x7f7cc5bab000

mmap(0x7f7cc5c1b000, 57344, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x21a000) = 0x7f7cc5c1b000

mmap(0x7f7cc5c29000, 10432, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f7cc5c29000

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\0\3\0>\0\1\0\0\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) = 0x7f7cc5dcf000

mmap(0x7f7cc5dd2000, 94208, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x3000) = 0x7f7cc5dd2000

mmap(0x7f7cc5de9000, 16384, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1a000) = 0x7f7cc5de9000

mmap(0x7f7cc5ded000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1d000) = 0x7f7cc5ded000

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\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0\0"..., 832) = 832

pread64(3, "\6\0\0\0\4\0\0\0@\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\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48

pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0 =\340\2563\265?\356\25x\261\27\313A#\350"..., 68, 896) = 68

newfstatat(3, "", {st\_mode=S\_IFREG|0755, st\_size=2216304, ...}, AT\_EMPTY\_PATH) = 0

pread64(3, "\6\0\0\0\4\0\0\0@\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, 2260560, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f7cc5600000

mmap(0x7f7cc5628000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7f7cc5628000

mmap(0x7f7cc57bd000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1bd000) = 0x7f7cc57bd000

mmap(0x7f7cc5815000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x214000) = 0x7f7cc5815000

mmap(0x7f7cc581b000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f7cc581b000

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\0\0\3\0>\0\1\0\0\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) = 0x7f7cc5ce8000

mmap(0x7f7cc5cf6000, 507904, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0xe000) = 0x7f7cc5cf6000

mmap(0x7f7cc5d72000, 372736, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x8a000) = 0x7f7cc5d72000

mmap(0x7f7cc5dcd000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0xe4000) = 0x7f7cc5dcd000

close(3) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f7cc5ce6000

mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f7cc5ce3000

arch\_prctl(ARCH\_SET\_FS, 0x7f7cc5ce3740) = 0

set\_tid\_address(0x7f7cc5ce3a10) = 52571

set\_robust\_list(0x7f7cc5ce3a20, 24) = 0

rseq(0x7f7cc5ce40e0, 0x20, 0, 0x53053053) = 0

mprotect(0x7f7cc5815000, 16384, PROT\_READ) = 0

mprotect(0x7f7cc5dcd000, 4096, PROT\_READ) = 0

mprotect(0x7f7cc5ded000, 4096, PROT\_READ) = 0

mprotect(0x7f7cc5c1b000, 45056, PROT\_READ) = 0

mprotect(0x5628ba3ee000, 4096, PROT\_READ) = 0

mprotect(0x7f7cc5e3d000, 8192, PROT\_READ) = 0

prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

munmap(0x7f7cc5def000, 81715) = 0

getrandom("\xe1\x0e\x4b\x46\x2a\xe2\x5a\xd6", 8, GRND\_NONBLOCK) = 8

brk(NULL) = 0x5628ba8b3000

brk(0x5628ba8d4000) = 0x5628ba8d4000

futex(0x7f7cc5c2977c, FUTEX\_WAKE\_PRIVATE, 2147483647) = 0

getpid() = 52571

openat(AT\_FDCWD, "/dev/shm/myserver\_pid", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 4) = 0

mmap(NULL, 4, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f7cc5e3c000

munmap(0x7f7cc5e3c000, 4) = 0

close(3) = 0

newfstatat(1, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0), ...}, AT\_EMPTY\_PATH) = 0

write(1, "Server pid is 52571\n", 20) = 20

openat(AT\_FDCWD, "/dev/shm/myserver\_pid", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 4) = 0

mmap(NULL, 4, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f7cc5e3c000

munmap(0x7f7cc5e3c000, 4) = 0

close(3) = 0

rt\_sigaction(SIGINT, {sa\_handler=0x5628ba3e3333, sa\_mask=[INT], sa\_flags=SA\_RESTORER|SA\_RESTART, sa\_restorer=0x7f7cc5642520}, {sa\_handler=SIG\_DFL, sa\_mask=[], sa\_flags=0}, 8) = 0

rt\_sigaction(SIGUSR1, {sa\_handler=0x5628ba3e331b, sa\_mask=[USR1], sa\_flags=SA\_RESTORER|SA\_RESTART, sa\_restorer=0x7f7cc5642520}, {sa\_handler=SIG\_DFL, sa\_mask=[], sa\_flags=0}, 8) = 0

--- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=52579, si\_uid=1000} ---

rt\_sigreturn({mask=[]}) = -1 EINTR (Interrupted system call)

openat(AT\_FDCWD, "/dev/shm/reader", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f7cc5e3c000

munmap(0x7f7cc5e3c000, 368) = 0

close(3) = 0

write(1, "New\_msg: type: 0 from: <Arsenii>"..., 38) = 38

write(1, "To: \7 Data: \n\n", 14) = 14

openat(AT\_FDCWD, "/dev/shm/writer", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f7cc5e3c000

munmap(0x7f7cc5e3c000, 368) = 0

close(3) = 0

kill(52579, SIGUSR1) = 0

write(1, "Sending msg:from sender: Server "..., 50) = 50

write(1, "Data: OK: Successful created!\n", 30) = 30

write(1, "Current number of users: 1\n\n", 28) = 28

rt\_sigaction(SIGUSR1, {sa\_handler=0x5628ba3e331b, sa\_mask=[USR1], sa\_flags=SA\_RESTORER|SA\_RESTART, sa\_restorer=0x7f7cc5642520}, {sa\_handler=0x5628ba3e331b, sa\_mask=[USR1], sa\_flags=SA\_RESTORER|SA\_RESTART, sa\_restorer=0x7f7cc5642520}, 8) = 0

--- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=52587, si\_uid=1000} ---

rt\_sigreturn({mask=[]}) = -1 EINTR (Interrupted system call)

openat(AT\_FDCWD, "/dev/shm/reader", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f7cc5e3c000

munmap(0x7f7cc5e3c000, 368) = 0

close(3) = 0

write(1, "New\_msg: type: 0 from: <Gg>52587"..., 33) = 33

write(1, "To: \7 Data: \n\n", 14) = 14

openat(AT\_FDCWD, "/dev/shm/writer", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f7cc5e3c000

munmap(0x7f7cc5e3c000, 368) = 0

close(3) = 0

kill(52587, SIGUSR1) = 0

write(1, "Sending msg:from sender: Server "..., 45) = 45

write(1, "Data: OK: Successful created!\n", 30) = 30

write(1, "Current number of users: 2\n\n", 28) = 28

rt\_sigaction(SIGUSR1, {sa\_handler=0x5628ba3e331b, sa\_mask=[USR1], sa\_flags=SA\_RESTORER|SA\_RESTART, sa\_restorer=0x7f7cc5642520}, {sa\_handler=0x5628ba3e331b, sa\_mask=[USR1], sa\_flags=SA\_RESTORER|SA\_RESTART, sa\_restorer=0x7f7cc5642520}, 8) = 0

--- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=52579, si\_uid=1000} ---

rt\_sigreturn({mask=[]}) = -1 EINTR (Interrupted system call)

openat(AT\_FDCWD, "/dev/shm/reader", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f7cc5e3c000

munmap(0x7f7cc5e3c000, 368) = 0

close(3) = 0

write(1, "New\_msg: type: 2 from: <Arsenii>"..., 38) = 38

write(1, "To: Gg Data: Hi!\n\n", 18) = 18

openat(AT\_FDCWD, "/dev/shm/writer", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f7cc5e3c000

munmap(0x7f7cc5e3c000, 368) = 0

close(3) = 0

kill(52587, SIGUSR1) = 0

write(1, "Current number of users: 2\n\n", 28) = 28

rt\_sigaction(SIGUSR1, {sa\_handler=0x5628ba3e331b, sa\_mask=[USR1], sa\_flags=SA\_RESTORER|SA\_RESTART, sa\_restorer=0x7f7cc5642520}, {sa\_handler=0x5628ba3e331b, sa\_mask=[USR1], sa\_flags=SA\_RESTORER|SA\_RESTART, sa\_restorer=0x7f7cc5642520}, 8) = 0

--- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=52579, si\_uid=1000} ---

rt\_sigreturn({mask=[]}) = -1 EINTR (Interrupted system call)

openat(AT\_FDCWD, "/dev/shm/reader", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f7cc5e3c000

munmap(0x7f7cc5e3c000, 368) = 0

close(3) = 0

write(1, "New\_msg: type: 1 from: <Arsenii>"..., 38) = 38

write(1, "To: Gg Data: new\_chat\n\n", 23) = 23

openat(AT\_FDCWD, "/dev/shm/writer", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f7cc5e3c000

munmap(0x7f7cc5e3c000, 368) = 0

close(3) = 0

kill(52579, SIGUSR1) = 0

write(1, "Sending msg:from sender: Arsenii"..., 46) = 46

write(1, "Data: Chat has been created succ"..., 42) = 42

write(1, "Current number of users: 2\n\n", 28) = 28

rt\_sigaction(SIGUSR1, {sa\_handler=0x5628ba3e331b, sa\_mask=[USR1], sa\_flags=SA\_RESTORER|SA\_RESTART, sa\_restorer=0x7f7cc5642520}, {sa\_handler=0x5628ba3e331b, sa\_mask=[USR1], sa\_flags=SA\_RESTORER|SA\_RESTART, sa\_restorer=0x7f7cc5642520}, 8) = 0

--- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=52579, si\_uid=1000} ---

rt\_sigreturn({mask=[]}) = -1 EINTR (Interrupted system call)

openat(AT\_FDCWD, "/dev/shm/reader", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f7cc5e3c000

munmap(0x7f7cc5e3c000, 368) = 0

close(3) = 0

write(1, "New\_msg: type: 4 from: <Arsenii>"..., 38) = 38

write(1, "To: Gg Data: new\_chat\n\n", 23) = 23

write(1, "Current number of users: 2\n\n", 28) = 28

rt\_sigaction(SIGUSR1, {sa\_handler=0x5628ba3e331b, sa\_mask=[USR1], sa\_flags=SA\_RESTORER|SA\_RESTART, sa\_restorer=0x7f7cc5642520}, {sa\_handler=0x5628ba3e331b, sa\_mask=[USR1], sa\_flags=SA\_RESTORER|SA\_RESTART, sa\_restorer=0x7f7cc5642520}, 8) = 0

--- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=52587, si\_uid=1000} ---

rt\_sigreturn({mask=[]}) = 0

openat(AT\_FDCWD, "/dev/shm/reader", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f7cc5e3c000

munmap(0x7f7cc5e3c000, 368) = 0

close(3) = 0

write(1, "New\_msg: type: 4 from: <Gg>52587"..., 33) = 33

write(1, "To: \r Data: \n\n", 14) = 14

write(1, "Close server. There are no activ"..., 40) = 40

unlink("/dev/shm/myserver\_pid") = 0

unlink("/dev/shm/reader") = 0

unlink("/dev/shm/writer") = 0

exit\_group(0) = ?

+++ exited with 0 +++

***client:*** strace -oclient\_log.log ./client Arsenii

execve("./client", ["./client", "Arsenii"], 0x7fff78f9ffd0 /\* 56 vars \*/) = 0

brk(NULL) = 0x55aaf3aad000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7fffce3d36f0) = -1 EINVAL (Invalid argument)

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f57a48d6000

access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=81715, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 81715, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f57a48c2000

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\0\3\0>\0\1\0\0\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) = 0x7f57a4600000

mprotect(0x7f57a469a000, 1576960, PROT\_NONE) = 0

mmap(0x7f57a469a000, 1118208, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x9a000) = 0x7f57a469a000

mmap(0x7f57a47ab000, 454656, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1ab000) = 0x7f57a47ab000

mmap(0x7f57a481b000, 57344, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x21a000) = 0x7f57a481b000

mmap(0x7f57a4829000, 10432, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f57a4829000

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\0\3\0>\0\1\0\0\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) = 0x7f57a48a2000

mmap(0x7f57a48a5000, 94208, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x3000) = 0x7f57a48a5000

mmap(0x7f57a48bc000, 16384, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1a000) = 0x7f57a48bc000

mmap(0x7f57a48c0000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1d000) = 0x7f57a48c0000

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\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0\0"..., 832) = 832

pread64(3, "\6\0\0\0\4\0\0\0@\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\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48

pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0 =\340\2563\265?\356\25x\261\27\313A#\350"..., 68, 896) = 68

newfstatat(3, "", {st\_mode=S\_IFREG|0755, st\_size=2216304, ...}, AT\_EMPTY\_PATH) = 0

pread64(3, "\6\0\0\0\4\0\0\0@\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, 2260560, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f57a4200000

mmap(0x7f57a4228000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7f57a4228000

mmap(0x7f57a43bd000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1bd000) = 0x7f57a43bd000

mmap(0x7f57a4415000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x214000) = 0x7f57a4415000

mmap(0x7f57a441b000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f57a441b000

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\0\0\3\0>\0\1\0\0\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) = 0x7f57a4519000

mmap(0x7f57a4527000, 507904, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0xe000) = 0x7f57a4527000

mmap(0x7f57a45a3000, 372736, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x8a000) = 0x7f57a45a3000

mmap(0x7f57a45fe000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0xe4000) = 0x7f57a45fe000

close(3) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f57a48a0000

mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f57a489d000

arch\_prctl(ARCH\_SET\_FS, 0x7f57a489d740) = 0

set\_tid\_address(0x7f57a489da10) = 52123

set\_robust\_list(0x7f57a489da20, 24) = 0

rseq(0x7f57a489e0e0, 0x20, 0, 0x53053053) = 0

mprotect(0x7f57a4415000, 16384, PROT\_READ) = 0

mprotect(0x7f57a45fe000, 4096, PROT\_READ) = 0

mprotect(0x7f57a48c0000, 4096, PROT\_READ) = 0

mprotect(0x7f57a481b000, 45056, PROT\_READ) = 0

mprotect(0x55aaf1e31000, 4096, PROT\_READ) = 0

mprotect(0x7f57a4910000, 8192, PROT\_READ) = 0

prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

munmap(0x7f57a48c2000, 81715) = 0

getrandom("\xae\xb0\x77\x17\xe7\xde\x4b\xbb", 8, GRND\_NONBLOCK) = 8

brk(NULL) = 0x55aaf3aad000

brk(0x55aaf3ace000) = 0x55aaf3ace000

futex(0x7f57a482977c, FUTEX\_WAKE\_PRIVATE, 2147483647) = 0

openat(AT\_FDCWD, "/dev/shm/myserver\_pid", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 4) = 0

mmap(NULL, 4, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f57a490f000

munmap(0x7f57a490f000, 4) = 0

close(3) = 0

getpid() = 52123

newfstatat(1, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x1), ...}, AT\_EMPTY\_PATH) = 0

write(1, "Conecting...\n", 13) = 13

rt\_sigaction(SIGRT\_1, {sa\_handler=0x7f57a4291870, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_ONSTACK|SA\_RESTART|SA\_SIGINFO, sa\_restorer=0x7f57a4242520}, NULL, 8) = 0

rt\_sigprocmask(SIG\_UNBLOCK, [RTMIN RT\_1], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f57a39ff000

mprotect(0x7f57a3a00000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f57a41ff910, parent\_tid=0x7f57a41ff910, exit\_signal=0, stack=0x7f57a39ff000, stack\_size=0x7fff00, tls=0x7f57a41ff640} => {parent\_tid=[52124]}, 88) = 52124

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

openat(AT\_FDCWD, "/dev/shm/reader", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f57a490f000

munmap(0x7f57a490f000, 368) = 0

close(3) = 0

kill(52060, SIGUSR1) = 0

clock\_nanosleep(CLOCK\_REALTIME, 0, {tv\_sec=2, tv\_nsec=0}, {tv\_sec=1, tv\_nsec=999724726}) = ? ERESTART\_RESTARTBLOCK (Interrupted by signal)

--- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=52060, si\_uid=1000} ---

rt\_sigreturn({mask=[]}) = -1 EINTR (Interrupted system call)

write(1, "Server\_pid: 52060\n", 18) = 18

write(1, "Welcome in our chat, Arsenii!\n", 30) = 30

futex(0x7f57a441ba70, FUTEX\_WAKE\_PRIVATE, 1) = 1

write(1, "Server:OK: Successful created!He"..., 97) = 97

write(1, "--For reading messages write:\n", 30) = 30

write(1, "To:'other\_username' 'you message"..., 37) = 37

write(1, "To:chat:'chat\_name' 'you message"..., 37) = 37

write(1, "-- Or if you want to join or cre"..., 50) = 50

write(1, "Join:'chat\_name'\n", 17) = 17

write(1, "--Write 'q' to close terminal\n", 30) = 30

write(1, " > ", 3) = 3

newfstatat(0, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x1), ...}, AT\_EMPTY\_PATH) = 0

read(0, "\n", 1024) = 1

write(1, "\n", 1) = 1

write(1, " > ", 3) = 3

read(0, "To:Gg Hi!\n", 1024) = 10

openat(AT\_FDCWD, "/dev/shm/reader", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f57a490f000

munmap(0x7f57a490f000, 368) = 0

close(3) = 0

kill(52060, SIGUSR1) = 0

clock\_nanosleep(CLOCK\_REALTIME, 0, {tv\_sec=1, tv\_nsec=0}, 0x7fffce3d3320) = 0

write(1, " > ", 3) = 3

read(0, 0x55aaf3abf3f0, 1024) = ? ERESTARTSYS (To be restarted if SA\_RESTART is set)

--- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=52060, si\_uid=1000} ---

rt\_sigreturn({mask=[]}) = 0

read(0, "\n", 1024) = 1

write(1, "Gg:Hello\n", 9) = 9

write(1, " > ", 3) = 3

read(0, "Join:new\_chat\n", 1024) = 14

openat(AT\_FDCWD, "/dev/shm/reader", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f57a490f000

munmap(0x7f57a490f000, 368) = 0

close(3) = 0

kill(52060, SIGUSR1) = 0

clock\_nanosleep(CLOCK\_REALTIME, 0, {tv\_sec=1, tv\_nsec=0}, {tv\_sec=0, tv\_nsec=999779195}) = ? ERESTART\_RESTARTBLOCK (Interrupted by signal)

--- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=52060, si\_uid=1000} ---

rt\_sigreturn({mask=[]}) = -1 EINTR (Interrupted system call)

futex(0x7f57a441ba70, FUTEX\_WAIT\_PRIVATE, 2, NULL) = -1 EAGAIN (Resource temporarily unavailable)

write(1, "Chat:new\_chat:Arsenii Chat has b"..., 57) = 57

futex(0x7f57a441ba70, FUTEX\_WAKE\_PRIVATE, 1) = 0

read(0, "\n", 1024) = 1

write(1, "\n", 1) = 1

write(1, " > ", 3) = 3

read(0, "q\n", 1024) = 2

rt\_sigaction(SIGRTMIN, {sa\_handler=0x7f57a4292b30, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_RESTART|SA\_SIGINFO, sa\_restorer=0x7f57a4242520}, NULL, 8) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

getpid() = 52123

tgkill(52123, 52124, SIGRTMIN) = 0

futex(0x7f57a41fffb4, FUTEX\_WAKE\_PRIVATE, 1) = 1

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

openat(AT\_FDCWD, "/dev/shm/reader", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

ftruncate(3, 368) = 0

mmap(NULL, 368, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f57a490f000

munmap(0x7f57a490f000, 368) = 0

close(3) = 0

kill(52060, SIGUSR1) = 0

write(1, "Break;\n", 7) = 7

exit\_group(0) = ?

+++ exited with 0 +++

**Вывод**

Данный курсовой проект направлен на закрепление навыков, полученных за курс Операционных систем. Конкретно мой проект был направлен на закрепление знаний в области разделяемой памяти.

В итоге я написал исправно работающий сервер для обмена сообщений, и поэтому считаю, что с поставленной задачей справился успешно.