Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

**Лабораторная работа №3 по курсу**

**«Операционные системы»**

Группа: М80-206Б-22

Студент: Ларченко А.О.

Преподаватель: Миронов Е.С.

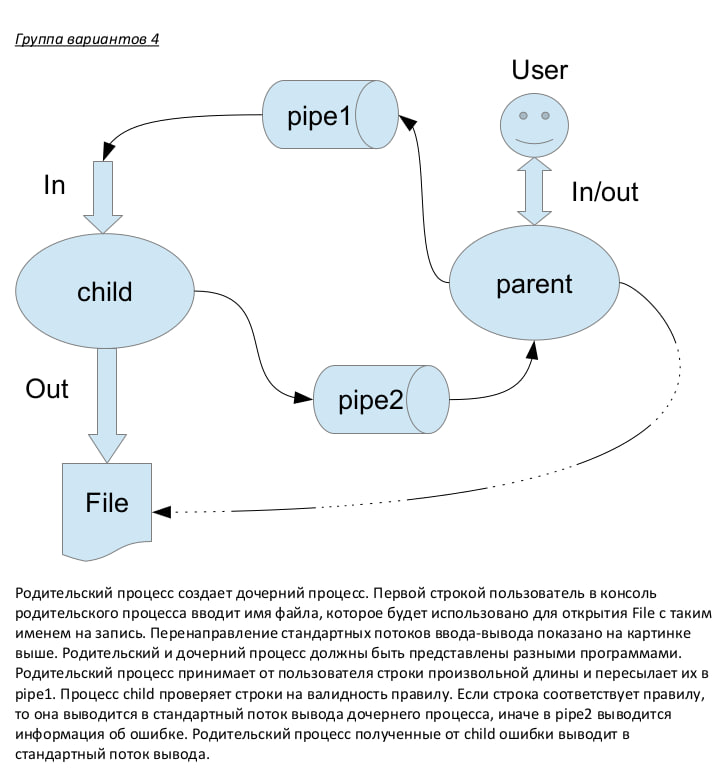
Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: 15.12.23

Москва, 2023

**Постановка задачи**

**Вариант 15.**

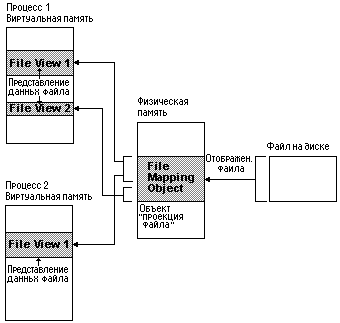
****

Правило проверки: строка должна начинаться с заглавной буквы

**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.



**Код программы**

***main.c:***

#include <stdio.h>

#include "function.h"

#include "m\_map.h"

static int check=0;

void wait\_read(int sig){

check=1;

}

int main(){

write(STDOUT\_FILENO, "Enter filename with file extension: ", 37);

char \*Filename=NULL;

if(inputing(&Filename ,STDIN\_FILENO, 0)<=0){

perror("Trying to create 0-value string: ");

exit(-1);

}

int f\_input=open(Filename, O\_WRONLY | O\_CREAT, 0777);

// FILE\* f\_input =fopen(Filename, "w");

if(f\_input==-1){

fprintf(stderr, "Can't open the file: %s", Filename);

exit(-1);

}

int parent\_pid=getpid();

char s\_ppid[sizeof(char)\*8];

sprintf(s\_ppid, "%d", parent\_pid); // int в строку чаров

// int tmp=atoi(s\_ppid);

// printf("\n%s", s\_ppid);

int pid=process\_creation();

if(pid==0){

// printf("Its child\n");

if(dup2(f\_input, STDOUT\_FILENO)==-1){

perror("Call dup2 was ended with erorr: ");

exit(-1);

}

if(execl("./child", "./child", s\_ppid, NULL)==-1){

perror("Call execl was ended with erorr: ");

exit(-1);

}

}else{ //it's parant

// printf("Its parant");

close(f\_input);

write(STDOUT\_FILENO, "Enter something you want: \n", 28);

while(true){ // \n\n - ending

int write\_state=write\_msg(pid); // -1 - erorr, 0 -OK, 1 - ending

if(write\_state==1){

break;

} else if(write\_state==-1){

perror("Something is going wrong: ");

exit(-1);

}

// sleep(1);

kill(pid, SIGUSR1);

signal(SIGUSR1, wait\_read);

while(check!=1);

int read\_err\_state=read\_error\_msg(pid);

if(read\_err\_state ==0){

} else if( read\_err\_state==1){

write(STDOUT\_FILENO, RED\_COLOR, sizeof(RED\_COLOR));

write(STDOUT\_FILENO, message\_list[0], sizeof(char)\*str\_size(message\_list[0]));

write(STDOUT\_FILENO, RESET\_COLOR, sizeof(RESET\_COLOR));

write(STDOUT\_FILENO, "\n", sizeof("\n"));

} else{

perror("Something is going wrong: ");

exit(-1);

}

check=0;

}

write(STDOUT\_FILENO, GREEN\_COLOR, sizeof(GREEN\_COLOR));

write(STDOUT\_FILENO, "\n", sizeof("\n"));

write(STDOUT\_FILENO, message\_list[5], sizeof(char)\*str\_size(message\_list[5]));

write(STDOUT\_FILENO, "\n", sizeof("\n"));

write(STDOUT\_FILENO, RESET\_COLOR, sizeof(RESET\_COLOR));

kill(pid, SIGTERM);

}

close\_sh\_file();

}

***function.h***

#ifndef function\_h

#define function\_h

#include <stdio.h>

#include <unistd.h>

#include <stdlib.h>

#include <stdbool.h>

#include <fcntl.h>

#include <sys/wait.h>

#include <signal.h>

#include <sys/types.h>

#define MIN\_LEN 10

#define MAX\_LEN 255

#define SIGTERM 15

#define RED\_COLOR "\x1b[31m"

#define GREEN\_COLOR "\x1b[32m"

#define RESET\_COLOR "\x1b[0m"

extern const char\* message\_list[6];

void pipe\_creation(int \*fd);

int process\_creation();

int str\_size(const char \*string);

bool clean\_name(char \*\*output\_name, char\* input\_name);

void check\_res(int fd\_in, int fd\_out);

int inputing(char \*\*s\_output, int fd, int endl\_status);

bool writing(char \*from, int from\_size ,char\* to, int to\_size);

int writing\_clear(char \*from, int from\_size, char \*\*to);

#endif

### function.c

#include <stdio.h>

#include "function.h"

void pipe\_creation(int \*fd){

if(pipe(fd)==-1){

perror("Call pipe was ended with error: ");

exit(-1);

}

}

int process\_creation(){

int pid =fork();

if(pid==-1){

perror("Call fork was ended with erorr: ");

exit(-1);

}

return pid;

}

int str\_size(const char \*string){

int len=0;

for(int i=0; i<MAX\_LEN; ++i){ // Fix reading '\n' bag and input string lenth counting

if(string[i]=='\n' || string[i]==EOF || string[i]=='\0'){

break;

}

len++;

}

return len;

}

bool writing(char \*from, int from\_size ,char\* to, int to\_size){

for(int i=0; i<from\_size; ++i){

to[i]=from[i];

}

to[from\_size]='\0';

return true;

}

int writing\_clear(char \*from, int from\_size, char \*\*to){

int len=0;

for(int i=0; i<from\_size;++i){

len++;

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

break;

}

}

char \*output=malloc((len+1)\*sizeof(char));

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

output[i]=from[i];

}

\*to=output;

return len;

}

int inputing(char \*\*s\_output, int fd, int endl\_status){

int new\_l=MIN\_LEN;

char \*line=(char\*)malloc(sizeof(char)\*new\_l);

int i=0;

char ch;

read(fd, &ch, sizeof(ch));

if(ch=='\n'){

line[i]='\n';

\*s\_output=line;

return -1;

}

while(ch!=EOF && ch!='\0' && ch!='\n' ){

if(i>=new\_l){

new\_l=new\_l\*2;

line=(char \*)realloc(line, sizeof(char)\*new\_l);

}

line[i]=ch;

i++;

read(fd, &ch, sizeof(ch));

}

if(endl\_status!=0){

if(i>=new\_l){

new\_l=new\_l\*2;

line=(char \*)realloc(line, sizeof(char)\*new\_l);

}

line[i]='\n';

i++;

}

\*s\_output=line;

return i;

}

void check\_res(int fd\_in, int fd\_out){

char status;

read(fd\_in, &status, sizeof(char));

if(status=='1'){

write(fd\_out, RED\_COLOR, sizeof(RED\_COLOR));

write(fd\_out, message\_list[0], str\_size(message\_list[0]));

write(fd\_out, RESET\_COLOR, sizeof(RESET\_COLOR));

write(fd\_out, "\n", sizeof("\n"));

}

}

const char\* message\_list[]={

//Errors:

"Error!\_Uncorrect input.\n",

"Call pipe was ended with error: ",

"Call fork was ended with error: ",

"Trying to create 0-value string: ",

//Normal status

"Enter filename with file extension: ",

"Program was ended successfully!\n\n",

};

### child.c

#include <stdio.h>

#include "function.h"

#include "m\_map.h"

static int wf=0;

void wait\_sig(int sig){

wf=1;

}

bool check\_first\_size(char a){

if(a>='A' && a<='Z'){

return true;

}

return false;

}

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

int ppid1=atoi(argv[1]);

// write(STDOUT\_FILENO, argv[1], sizeof(char)\*8);

while(1){

signal(SIGUSR1, wait\_sig);

while(wf!=1);

char \*output= NULL;

int read\_status=read\_msg(0, &output, CAPACITY);

// write(STDOUT\_FILENO, output, sizeof(char)\*CAPACITY);

if(read\_status==-1){

free(output);

perror("Something is going wrong: ");

exit(-1);

} else if(read\_status==0){

free(output);

break;

}

while(wf!=1);

if(check\_first\_size(output[0])==true){

write(STDOUT\_FILENO, output, sizeof(char)\*read\_status);

write\_error\_msg(0, '0');

//write(STDERR\_FILENO, "0", sizeof("0"));

} else{

write\_error\_msg(0, '1');

// write(STDERR\_FILENO, "1", sizeof("1")); //uncorrect input

}

kill(ppid1, SIGUSR1);

free(output);

wf=0;

}

return 0;

}

### m\_map.h

#pragma once

#include <stdio.h>

#include <stdlib.h>

#include <sys/mman.h>

#include <sys/stat.h>

#include <fcntl.h>

#include <unistd.h>

#include <fcntl.h>

#include <stdbool.h>

#include "function.h"

#define CAPACITY 500

#define SHARED\_OBJ1\_NAME "/mappa1"

#define SHARED\_OBJ2\_NAME "/mappa2"

typedef struct message{

// int pid;

// int len;

char buff[CAPACITY];

}message;

typedef struct error\_message{

// int pid;

char status;

}error\_message;

int write\_msg(int pid);

int read\_msg(int pid, char \*\*output, int s\_len);

int write\_error\_msg(int pid, char status);

int read\_error\_msg(int pid);

void close\_sh\_file();

### m\_map.c

#include "m\_map.h"

#include <unistd.h>

int write\_msg(int pid){

int shm\_fd=shm\_open(SHARED\_OBJ1\_NAME, O\_CREAT | O\_RDWR, S\_IRUSR | S\_IWUSR);

if(shm\_fd==-1){

if(shm\_unlink(SHARED\_OBJ1\_NAME)==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

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

if(shm\_unlink(SHARED\_OBJ1\_NAME)==-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(SHARED\_OBJ1\_NAME)==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

// msg\_ptr->pid=pid;

char \*s=NULL;

int s\_len=inputing(&s, STDIN\_FILENO, 1);

// msg\_ptr->len=s\_len;

if(s\_len==-1){

// char tmp[CAPACITY];

writing(s, 1, msg\_ptr->buff, CAPACITY);

//write(pipe\_1[1], "\n", sizeof("\n"));

free(s);

return 1;

} else if(s\_len>=CAPACITY-1){

write(STDOUT\_FILENO, "Sorry, so long message. I'm only lerning and can process

message are longer then 255 symbols. Try again\n", 105);

} else{

writing(s, s\_len, msg\_ptr->buff, CAPACITY);

}

free(s);

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

close(shm\_fd);

return 0;

}

int read\_msg(int pid, char \*\*input, int s\_len){

int shm\_fd=shm\_open(SHARED\_OBJ1\_NAME, O\_CREAT | O\_RDWR, S\_IRUSR | S\_IWUSR);

if(shm\_fd==-1){

if(shm\_unlink(SHARED\_OBJ1\_NAME)==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

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

if(shm\_unlink(SHARED\_OBJ1\_NAME)==-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(SHARED\_OBJ1\_NAME)==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_fd);

return -1;

}

if(msg\_ptr->buff[0]=='\n'){

return 0;

}

if(msg\_ptr->buff[0]=='-'){

}

// char \*output= malloc(CAPACITY\*sizeof(char));

// writing(msg\_ptr->buff, CAPACITY, output, s\_len);

char \*output=NULL;

int length=writing\_clear(msg\_ptr->buff, CAPACITY, &output);

// write(STDOUT\_FILENO, output, sizeof(char)\*CAPACITY);

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

close(shm\_fd);

\*input=output;

return length;

}

int write\_error\_msg(int pid, char status){

int shm\_err\_fd=shm\_open(SHARED\_OBJ2\_NAME, O\_CREAT | O\_RDWR, S\_IRUSR | S\_IWUSR);

if(shm\_err\_fd==-1){

if(shm\_unlink(SHARED\_OBJ2\_NAME)==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_err\_fd);

return -1;

}

if(ftruncate(shm\_err\_fd, sizeof(error\_message))==-1){

if(shm\_unlink(SHARED\_OBJ2\_NAME)==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_err\_fd);

return -1;

}

error\_message \*msg\_err\_ptr=(error\_message\*)mmap(NULL, sizeof(error\_message),

PROT\_READ | PROT\_WRITE, MAP\_SHARED, shm\_err\_fd, 0);

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

if(shm\_unlink(SHARED\_OBJ2\_NAME)==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_err\_fd);

return -1;

}

// msg\_err\_ptr->pid=pid;

msg\_err\_ptr->status=status;

munmap(msg\_err\_ptr, sizeof(error\_message));

close(shm\_err\_fd);

return 0;

}

int read\_error\_msg(int pid){

int shm\_err\_fd=shm\_open(SHARED\_OBJ2\_NAME, O\_CREAT | O\_RDWR, S\_IRUSR | S\_IWUSR);

if(shm\_err\_fd==-1){

if(shm\_unlink(SHARED\_OBJ2\_NAME)==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_err\_fd);

return -1;

}

if(ftruncate(shm\_err\_fd, sizeof(error\_message))==-1){

if(shm\_unlink(SHARED\_OBJ2\_NAME)==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_err\_fd);

return -1;

}

error\_message \*msg\_err\_ptr=(error\_message\*)mmap(NULL, sizeof(error\_message),

PROT\_READ | PROT\_WRITE, MAP\_SHARED, shm\_err\_fd, 0);

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

if(shm\_unlink(SHARED\_OBJ2\_NAME)==-1){

perror("munmap trouble: ");

exit(-1);

}

close(shm\_err\_fd);

return -1;

}

char status=msg\_err\_ptr->status;

int res;

munmap(msg\_err\_ptr, sizeof(error\_message));

close(shm\_err\_fd);

if (status=='0'){

res=0;

} else{

res=1; // error in sentence -> output error

}

return res;

}

void close\_sh\_file(){

if(shm\_unlink(SHARED\_OBJ2\_NAME)==-1){

perror("munmap trouble: ");

exit(-1);

}

if(shm\_unlink(SHARED\_OBJ1\_NAME)==-1){

perror("munmap trouble: ");

exit(-1);

}

}

**Протокол работы программы**

**Тестирование:**

arsenii@PC-Larcha14:~/Documents/VS\_code\_prog/OSI/laba\_3$ ./main

Enter filename with file extension: 1.txt

Enter something you want:

Hi! My name is Arsenii!!

oh

Error!\_Uncorrect input.

Oh

mmmm

Error!\_Uncorrect input.

Bui

bui!

Error!\_Uncorrect input.

Program was ended successfully!

arsenii@PC-Larcha14:~/Documents/VS\_code\_prog/OSI/laba\_3$ cat 1.txt

Hi! My name is Arsenii!!

Oh

Bui

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

**Strace:**

arsenii@PC-Larcha14:~/Documents/VS\_code\_prog/OSI/laba\_3$ strace -f ./main

execve("./main", ["./main"], 0x7ffd6778d858 /\* 56 vars \*/) = 0

brk(NULL) = 0x561d39c65000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffc91fb1260) = -1 EINVAL (Invalid argument)

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1,

0) = 0x7fdf5c4c6000

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=80547, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 80547, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7fdf5c4b2000

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) =

0x7fdf5c200000

mmap(0x7fdf5c228000, 1658880, PROT\_READ|PROT\_EXEC,

MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7fdf5c228000

mmap(0x7fdf5c3bd000, 360448, PROT\_READ,

MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1bd000) = 0x7fdf5c3bd000

mmap(0x7fdf5c415000, 24576, PROT\_READ|PROT\_WRITE,

MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x214000) = 0x7fdf5c415000

mmap(0x7fdf5c41b000, 52816, PROT\_READ|PROT\_WRITE,

MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fdf5c41b000

close(3) = 0

mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1,

0) = 0x7fdf5c4af000

arch\_prctl(ARCH\_SET\_FS, 0x7fdf5c4af740) = 0

set\_tid\_address(0x7fdf5c4afa10) = 30563

set\_robust\_list(0x7fdf5c4afa20, 24) = 0

rseq(0x7fdf5c4b00e0, 0x20, 0, 0x53053053) = 0

mprotect(0x7fdf5c415000, 16384, PROT\_READ) = 0

mprotect(0x561d37e7f000, 4096, PROT\_READ) = 0

mprotect(0x7fdf5c500000, 8192, PROT\_READ) = 0

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

munmap(0x7fdf5c4b2000, 80547) = 0

write(1, "Enter filename with file extensi"..., 37Enter filename with file extension: ) = 37

getrandom("\x87\x12\x50\xb6\xf6\x53\x12\xc5", 8, GRND\_NONBLOCK) = 8

brk(NULL) = 0x561d39c65000

brk(0x561d39c86000) = 0x561d39c86000

read(0, 2.txt

"2", 1) = 1

read(0, ".", 1) = 1

read(0, "t", 1) = 1

read(0, "x", 1) = 1

read(0, "t", 1) = 1

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

openat(AT\_FDCWD, "2.txt", O\_WRONLY|O\_CREAT, 0777) = 3

getpid() = 30563

clone(child\_stack=NULL,

flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLD,

child\_tidptr=0x7fdf5c4afa10) = 30564

strace: Process 30564 attached

[pid 30563] close(3) = 0

[pid 30563] write(1, "Enter something you want: \n\0", 28Enter something you want:

) = 28

[pid 30563] openat(AT\_FDCWD, "/dev/shm/mappa1",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600 <unfinished ...>

[pid 30564] set\_robust\_list(0x7fdf5c4afa20, 24 <unfinished ...>

[pid 30563] <... openat resumed>) = 3

[pid 30563] **ftruncate**(3, 500) = 0

[pid 30564] <... set\_robust\_list resumed>) = 0

[pid 30563] **mmap**(NULL, 500, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) =

0x7fdf5c4ff000

[pid 30563] read(0, <unfinished ...>

[pid 30564] dup2(3, 1) = 1

[pid 30564] execve("./child", ["./child", "30563"], 0x7ffc91fb1438 /\* 56 vars \*/) = 0

[pid 30564] brk(NULL) = 0x55a15aec8000

[pid 30564] arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7fff265e10d0) = -1 EINVAL (Invalid

argument)

[pid 30564] mmap(NULL, 8192, PROT\_READ|PROT\_WRITE,

MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f03c97d4000

[pid 30564] access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (No such file or directory)

[pid 30564] openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 4

[pid 30564] newfstatat(4, "", {st\_mode=S\_IFREG|0644, st\_size=80547, ...}, AT\_EMPTY\_PATH) = 0

[pid 30564] mmap(NULL, 80547, PROT\_READ, MAP\_PRIVATE, 4, 0) = 0x7f03c97c0000

[pid 30564] close(4) = 0

[pid 30564] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC) = 4

[pid 30564] read(4, "\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

[pid 30564] pread64(4, "\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

[pid 30564] pread64(4, "\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

[pid 30564] pread64(4, "\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

[pid 30564] newfstatat(4, "", {st\_mode=S\_IFREG|0755, st\_size=2216304, ...},

AT\_EMPTY\_PATH) = 0

[pid 30564] pread64(4, "\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

[pid 30564] mmap(NULL, 2260560, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 4, 0) =

0x7f03c9400000

[pid 30564] mmap(0x7f03c9428000, 1658880, PROT\_READ|PROT\_EXEC,

MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 4, 0x28000) = 0x7f03c9428000

[pid 30564] mmap(0x7f03c95bd000, 360448, PROT\_READ,

MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 4, 0x1bd000) = 0x7f03c95bd000

[pid 30564] mmap(0x7f03c9615000, 24576, PROT\_READ|PROT\_WRITE,

MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 4, 0x214000) = 0x7f03c9615000

[pid 30564] mmap(0x7f03c961b000, 52816, PROT\_READ|PROT\_WRITE,

MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f03c961b000

[pid 30564] close(4) = 0

[pid 30564] mmap(NULL, 12288, PROT\_READ|PROT\_WRITE,

MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f03c97bd000

[pid 30564] arch\_prctl(ARCH\_SET\_FS, 0x7f03c97bd740) = 0

[pid 30564] set\_tid\_address(0x7f03c97bda10) = 30564

[pid 30564] set\_robust\_list(0x7f03c97bda20, 24) = 0

[pid 30564] rseq(0x7f03c97be0e0, 0x20, 0, 0x53053053) = 0

[pid 30564] mprotect(0x7f03c9615000, 16384, PROT\_READ) = 0

[pid 30564] mprotect(0x55a15ad18000, 4096, PROT\_READ) = 0

[pid 30564] mprotect(0x7f03c980e000, 8192, PROT\_READ) = 0

[pid 30564] prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024,

rlim\_max=RLIM64\_INFINITY}) = 0

[pid 30564] munmap(0x7f03c97c0000, 80547) = 0

[pid 30564] rt\_sigaction(SIGUSR1, {sa\_handler=0x55a15ad15389, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_INTERRUPT|SA\_NODEFER|SA\_RESETHAND|0xffffffff00000000,

sa\_restorer=0x7f03c9442520}, {sa\_handler=SIG\_DFL, sa\_mask=[], sa\_flags=0}, 8) = 0

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, 0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if SA\_RESTART is set)

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if

SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, 0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if SA\_RESTART is set)

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, <unfinished ...>

[pid 30564] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] <... read resumed>0x7ffc91fb1277, 1) = ? ERESTARTSYS (To be restarted if SA\_RESTART is set)

[pid 30563] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 30563] read(0, Hi

"H", 1) = 1

[pid 30563] read(0, "i", 1) = 1

[pid 30563] read(0, "\n", 1) = 1

[pid 30563] munmap(0x7fdf5c4ff000, 500) = 0

[pid 30563] close(3) = 0

[pid 30563] kill(30564, SIGUSR1) = 0

[pid 30564] --- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=30563, si\_uid=1000} ---

[pid 30563] rt\_sigaction(SIGUSR1, {sa\_handler=0x561d37e7c429, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_INTERRUPT|SA\_NODEFER|SA\_RESETHAND|0xffffffff00000000,

sa\_restorer=0x7fdf5c242520}, {sa\_handler=SIG\_DFL, sa\_mask=[], sa\_flags=0}, 8) = 0

[pid 30564] rt\_sigreturn({mask=[]}) = 0

[pid 30564] openat(AT\_FDCWD, "/dev/shm/mappa1",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 4

[pid 30564] **ftruncate**(4, 500) = 0

[pid 30564] **mmap**(NULL, 500, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 4, 0) =

0x7f03c980d000

[pid 30564] getrandom("\x81\xfd\x9f\xdd\x47\x1f\xf9\x9f", 8, GRND\_NONBLOCK) = 8

[pid 30564] brk(NULL) = 0x55a15aec8000

[pid 30564] brk(0x55a15aee9000) = 0x55a15aee9000

[pid 30564] **munmap**(0x7f03c980d000, 500) = 0

[pid 30564] close(4) = 0

[pid 30564] write(1, "Hi\n", 3) = 3

[pid 30564] openat(AT\_FDCWD, "/dev/shm/mappa2",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 4

[pid 30564] **ftruncate**(4, 1) = 0

[pid 30564] **mmap**(NULL, 1, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 4, 0) =

0x7f03c980d000

[pid 30564] **munmap**(0x7f03c980d000, 1) = 0

[pid 30564] close(4) = 0

[pid 30564] kill(30563, SIGUSR1 <unfinished ...>

[pid 30563] --- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=30564, si\_uid=1000} ---

[pid 30564] <... kill resumed>) = 0

[pid 30563] rt\_sigreturn({mask=[]} <unfinished ...>

[pid 30564] rt\_sigaction(SIGUSR1, {sa\_handler=0x55a15ad15389, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_INTERRUPT|SA\_NODEFER|SA\_RESETHAND|0xffffffff00000000,

sa\_restorer=0x7f03c9442520}, <unfinished ...>

[pid 30563] <... rt\_sigreturn resumed>) = 0

[pid 30564] <... rt\_sigaction resumed>{sa\_handler=SIG\_DFL, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_NODEFER|SA\_RESETHAND, sa\_restorer=0x7f03c9442520}, 8) = 0

[pid 30563] openat(AT\_FDCWD, "/dev/shm/mappa2",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

[pid 30563] **ftruncate**(3, 1) = 0

[pid 30563] **mmap**(NULL, 1, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) =

0x7fdf5c4ff000

[pid 30563] **munmap**(0x7fdf5c4ff000, 1) = 0

[pid 30563] close(3) = 0

[pid 30563] openat(AT\_FDCWD, "/dev/shm/mappa1",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

[pid 30563] **ftruncate**(3, 500) = 0

[pid 30563] **mmap**(NULL, 500, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) =

0x7fdf5c4ff000

[pid 30563] read(0, hi

"h", 1) = 1

[pid 30563] read(0, "i", 1) = 1

[pid 30563] read(0, "\n", 1) = 1

[pid 30563] **munmap**(0x7fdf5c4ff000, 500) = 0

[pid 30563] close(3) = 0

[pid 30563] kill(30564, SIGUSR1) = 0

[pid 30564] --- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=30563, si\_uid=1000} ---

[pid 30563] rt\_sigaction(SIGUSR1, {sa\_handler=0x561d37e7c429, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_INTERRUPT|SA\_NODEFER|SA\_RESETHAND|0xffffffff00000000,

sa\_restorer=0x7fdf5c242520}, <unfinished ...>

[pid 30564] rt\_sigreturn({mask=[]} <unfinished ...>

[pid 30563] <... rt\_sigaction resumed>{sa\_handler=SIG\_DFL, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_NODEFER|SA\_RESETHAND, sa\_restorer=0x7fdf5c242520}, 8) = 0

[pid 30564] <... rt\_sigreturn resumed>) = 0

[pid 30564] openat(AT\_FDCWD, "/dev/shm/mappa1",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 4

[pid 30564] **ftruncate**(4, 500) = 0

[pid 30564] **mmap**(NULL, 500, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 4, 0) =

0x7f03c980d000

[pid 30564] **munmap**(0x7f03c980d000, 500) = 0

[pid 30564] close(4) = 0

[pid 30564] openat(AT\_FDCWD, "/dev/shm/mappa2",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 4

[pid 30564] **ftruncate**(4, 1) = 0

[pid 30564] **mmap**(NULL, 1, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 4, 0) =

0x7f03c980d000

[pid 30564] **munmap**(0x7f03c980d000, 1) = 0

[pid 30564] close(4) = 0

[pid 30564] kill(30563, SIGUSR1 <unfinished ...>

[pid 30563] --- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=30564, si\_uid=1000} ---

[pid 30564] <... kill resumed>) = 0

[pid 30563] rt\_sigreturn({mask=[]} <unfinished ...>

[pid 30564] rt\_sigaction(SIGUSR1, {sa\_handler=0x55a15ad15389, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_INTERRUPT|SA\_NODEFER|SA\_RESETHAND|0xffffffff00000000,

sa\_restorer=0x7f03c9442520}, <unfinished ...>

[pid 30563] <... rt\_sigreturn resumed>) = 0

[pid 30564] <... rt\_sigaction resumed>{sa\_handler=SIG\_DFL, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_NODEFER|SA\_RESETHAND, sa\_restorer=0x7f03c9442520}, 8) = 0

[pid 30563] openat(AT\_FDCWD, "/dev/shm/mappa2",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

[pid 30563] **ftruncate**(3, 1) = 0

[pid 30563] **mmap**(NULL, 1, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7fdf5c4ff000

[pid 30563] **munmap**(0x7fdf5c4ff000, 1) = 0

[pid 30563] close(3) = 0

[pid 30563] write(1, "\33[31m\0", 6) = 6

[pid 30563] write(1, "Error!\_Uncorrect input.", 23Error!\_Uncorrect input.) = 23

[pid 30563] write(1, "\33[0m\0", 5) = 5

[pid 30563] write(1, "\n\0", 2

) = 2

[pid 30563] openat(AT\_FDCWD, "/dev/shm/mappa1",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

[pid 30563] **ftruncate**(3, 500) = 0

[pid 30563] **mmap**(NULL, 500, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) =

0x7fdf5c4ff000

[pid 30563] read(0, hhhh

"h", 1) = 1

[pid 30563] read(0, "h", 1) = 1

[pid 30563] read(0, "h", 1) = 1

[pid 30563] read(0, "h", 1) = 1

[pid 30563] read(0, "\n", 1) = 1

[pid 30563] **munmap**(0x7fdf5c4ff000, 500) = 0

[pid 30563] close(3) = 0

[pid 30563] kill(30564, SIGUSR1) = 0

[pid 30564] --- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=30563, si\_uid=1000} ---

[pid 30563] rt\_sigaction(SIGUSR1, {sa\_handler=0x561d37e7c429, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_INTERRUPT|SA\_NODEFER|SA\_RESETHAND|0xffffffff00000000,

sa\_restorer=0x7fdf5c242520}, <unfinished ...>

[pid 30564] rt\_sigreturn({mask=[]} <unfinished ...>

[pid 30563] <... rt\_sigaction resumed>{sa\_handler=SIG\_DFL, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_NODEFER|SA\_RESETHAND, sa\_restorer=0x7fdf5c242520}, 8) = 0

[pid 30564] <... rt\_sigreturn resumed>) = 0

[pid 30564] openat(AT\_FDCWD, "/dev/shm/mappa1",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 4

[pid 30564] **ftruncate**(4, 500) = 0

[pid 30564] **mmap**(NULL, 500, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 4, 0) =

0x7f03c980d000

[pid 30564] **munmap**(0x7f03c980d000, 500) = 0

[pid 30564] close(4) = 0

[pid 30564] openat(AT\_FDCWD, "/dev/shm/mappa2",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 4

[pid 30564] **ftruncate**(4, 1) = 0

[pid 30564] **mmap**(NULL, 1, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 4, 0) =

0x7f03c980d000

[pid 30564] **munmap**(0x7f03c980d000, 1) = 0

[pid 30564] close(4) = 0

[pid 30564] kill(30563, SIGUSR1 <unfinished ...>

[pid 30563] --- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=30564, si\_uid=1000} ---

[pid 30564] <... kill resumed>) = 0

[pid 30563] rt\_sigreturn({mask=[]} <unfinished ...>

[pid 30564] rt\_sigaction(SIGUSR1, {sa\_handler=0x55a15ad15389, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_INTERRUPT|SA\_NODEFER|SA\_RESETHAND|0xffffffff00000000,

sa\_restorer=0x7f03c9442520}, <unfinished ...>

[pid 30563] <... rt\_sigreturn resumed>) = 0

[pid 30564] <... rt\_sigaction resumed>{sa\_handler=SIG\_DFL, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_NODEFER|SA\_RESETHAND, sa\_restorer=0x7f03c9442520}, 8) = 0

[pid 30563] openat(AT\_FDCWD, "/dev/shm/mappa2",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

[pid 30563] **ftruncate**(3, 1) = 0

[pid 30563] **mmap**(NULL, 1, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) =

0x7fdf5c4ff000

[pid 30563] **munmap**(0x7fdf5c4ff000, 1) = 0

[pid 30563] close(3) = 0

[pid 30563] write(1, "\33[31m\0", 6) = 6

[pid 30563] write(1, "Error!\_Uncorrect input.", 23Error!\_Uncorrect input.) = 23

[pid 30563] write(1, "\33[0m\0", 5) = 5

[pid 30563] write(1, "\n\0", 2

) = 2

[pid 30563] openat(AT\_FDCWD, "/dev/shm/mappa1",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

[pid 30563] **ftruncate**(3, 500) = 0

[pid 30563] **mmap**(NULL, 500, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) =

0x7fdf5c4ff000

[pid 30563] read(0, Ha!

"H", 1) = 1

[pid 30563] read(0, "a", 1) = 1

[pid 30563] read(0, "!", 1) = 1

[pid 30563] read(0, "\n", 1) = 1

[pid 30563] **munmap**(0x7fdf5c4ff000, 500) = 0

[pid 30563] close(3) = 0

[pid 30563] kill(30564, SIGUSR1) = 0

[pid 30564] --- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=30563, si\_uid=1000} ---

[pid 30563] rt\_sigaction(SIGUSR1, {sa\_handler=0x561d37e7c429, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_INTERRUPT|SA\_NODEFER|SA\_RESETHAND|0xffffffff00000000,

sa\_restorer=0x7fdf5c242520}, <unfinished ...>

[pid 30564] rt\_sigreturn({mask=[]} <unfinished ...>

[pid 30563] <... rt\_sigaction resumed>{sa\_handler=SIG\_DFL, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_NODEFER|SA\_RESETHAND, sa\_restorer=0x7fdf5c242520}, 8) = 0

[pid 30564] <... rt\_sigreturn resumed>) = 0

[pid 30564] openat(AT\_FDCWD, "/dev/shm/mappa1",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 4

[pid 30564] **ftruncate**(4, 500) = 0

[pid 30564] **mmap**(NULL, 500, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 4, 0) =

0x7f03c980d000

[pid 30564] **munmap**(0x7f03c980d000, 500) = 0

[pid 30564] close(4) = 0

[pid 30564] write(1, "Ha!\n", 4) = 4

[pid 30564] openat(AT\_FDCWD, "/dev/shm/mappa2",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 4

[pid 30564] **ftruncate**(4, 1) = 0

[pid 30564] **mmap**(NULL, 1, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 4, 0) =

0x7f03c980d000

[pid 30564] **munmap**(0x7f03c980d000, 1) = 0

[pid 30564] close(4) = 0

[pid 30564] kill(30563, SIGUSR1 <unfinished ...>

[pid 30563] --- SIGUSR1 {si\_signo=SIGUSR1, si\_code=SI\_USER, si\_pid=30564, si\_uid=1000} ---

[pid 30564] <... kill resumed>) = 0

[pid 30563] rt\_sigreturn({mask=[]} <unfinished ...>

[pid 30564] rt\_sigaction(SIGUSR1, {sa\_handler=0x55a15ad15389, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_INTERRUPT|SA\_NODEFER|SA\_RESETHAND|0xffffffff00000000,

sa\_restorer=0x7f03c9442520}, <unfinished ...>

[pid 30563] <... rt\_sigreturn resumed>) = 0

[pid 30564] <... rt\_sigaction resumed>{sa\_handler=SIG\_DFL, sa\_mask=[],

sa\_flags=SA\_RESTORER|SA\_NODEFER|SA\_RESETHAND, sa\_restorer=0x7f03c9442520}, 8) = 0

[pid 30563] openat(AT\_FDCWD, "/dev/shm/mappa2",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

[pid 30563] **ftruncate**(3, 1) = 0

[pid 30563] **mmap**(NULL, 1, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) =

0x7fdf5c4ff000

[pid 30563] **munmap**(0x7fdf5c4ff000, 1) = 0

[pid 30563] close(3) = 0

[pid 30563] openat(AT\_FDCWD, "/dev/shm/mappa1",

O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0600) = 3

[pid 30563] **ftruncate**(3, 500) = 0

[pid 30563] **mmap**(NULL, 500, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) =

0x7fdf5c4ff000

[pid 30563] read(0,

"\n", 1) = 1

[pid 30563] write(1, "\33[32m\0", 6) = 6

[pid 30563] write(1, "\n\0", 2

) = 2

[pid 30563] write(1, "Program was ended successfully!", 31Program was ended successfully!) = 31

[pid 30563] write(1, "\n\0", 2

) = 2

[pid 30563] write(1, "\33[0m\0", 5) = 5

[pid 30563] kill(30564, SIGTERM) = 0

[pid 30564] --- SIGTERM {si\_signo=SIGTERM, si\_code=SI\_USER, si\_pid=30563, si\_uid=1000} ---

[pid 30563] unlink("/dev/shm/mappa2") = 0

[pid 30563] unlink("/dev/shm/mappa1") = 0

[pid 30563] exit\_group(0) = ?

[pid 30564] +++ killed by SIGTERM +++

+++ exited with 0 +++

arsenii@PC-Larcha14:~/Documents/VS\_code\_prog/OSI/laba\_3$

**Вывод**

Эта лабораторная работа является фактическим дополнением к 1: то же самое задание, но вместо pipов, мы используем отображаемые файлы, еще один из способов реализации общей памяти в IPC. Для полного набора не хватает лабораторной по сокетам, я считаю.

Как обычно появились проблемы при работе с передаваемыми строчками, работа же с отображаемыми файлами трудностей не вызвала. В этой лабораторной также были использованы сигналы, потребности в которых при работе с pipами не было, т.к. у pipов - 2 файловых дескриптера, а в случае с обображаемым файлом, как и у обычного файла - 1, и чтобы дочка понимала, когда в файле есть что прочитать, мы отправляли ей сигнал с родителя сразу же после записи. Также pipы выглядят практичнее, на мой взгляд, с точки зрения “неограниченности” хранилища, в случае же с отображаемыми файлами, мы ещё и сами задаем размер. Ну и для работы с pipом не нужно использовать столько дополнительных системных вызовов, а также писать отдельные функции.

Таким образом, эту лабораторная работу можно считать напрвленной на сравнение 2 способов использование общей памяти для IPC. Как обычно, у каждого из способов есть свои преимущества и недостатки, но, так или иначе, каждый находит себе своё применение.

В итоге у меня получился исправно работающий код, считаю, что с поставленной задачей справился успешно.