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: _____

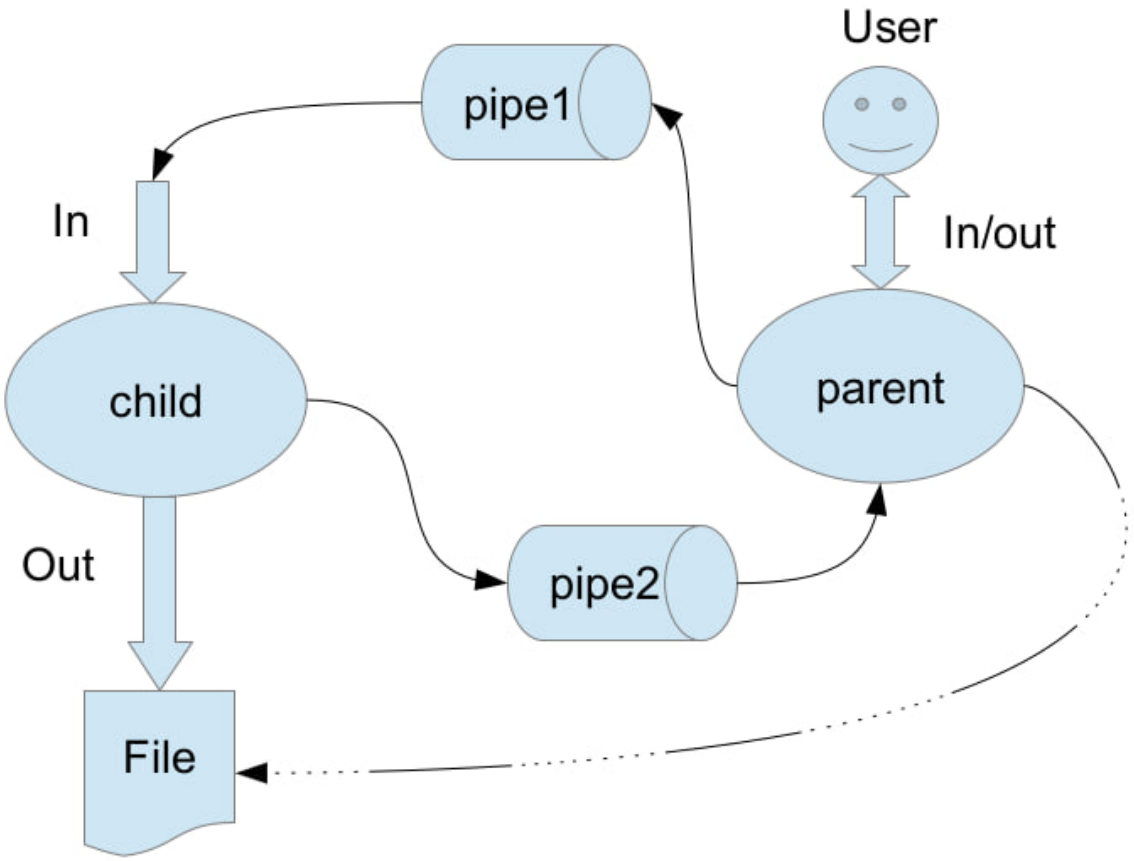
Дата: 15.12.23

Москва, 2023

Постановка задачи

Вариант 15.

Группа вариантов 4



Родительский процесс создает дочерний процесс. Первой строкой пользователь в консоль родительского процесса вводит имя файла, которое будет использовано для открытия File с таким именем на запись. Перенаправление стандартных потоков ввода-вывода показано на картинке выше. Родительский и дочерний процесс должны быть представлены разными программами. Родительский процесс принимает от пользователя строки произвольной длины и пересылает их в pipe1. Процесс child проверяет строки на валидность правилу. Если строка соответствует правилу, то она выводится в стандартный поток вывода дочернего процесса, иначе в pipe2 выводится информация об ошибке. Родительский процесс полученные от child ошибки выводит в стандартный поток вывода.

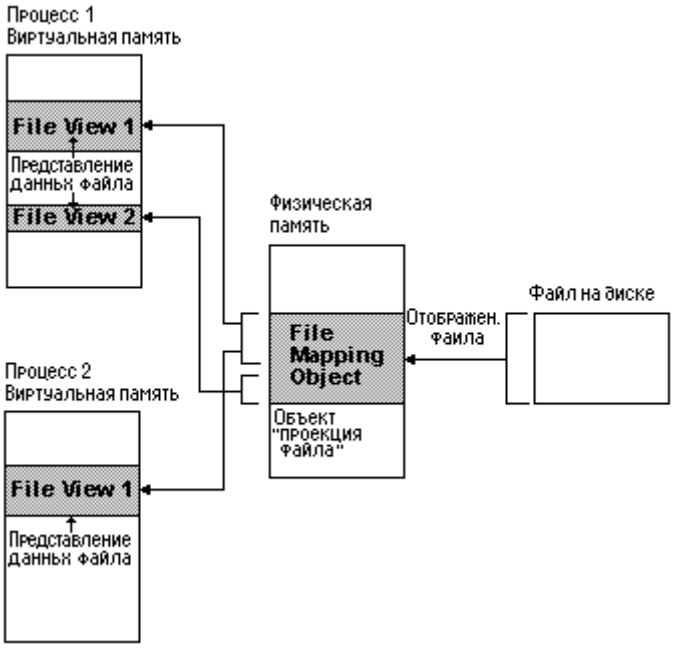
Правило проверки: строка должна начинаться с заглавной буквы

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"..., 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"..., 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.