**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ  
ФЕДЕРАЦИИ МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ  
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)**

**ЛАБОРАТОРНАЯ РАБОТА №4**

**по курсу операционные системы I семестр, 2021/22  
уч. год**

Студент: Белоусов Егор Владимирович

Группа: М8О-207Б-20

Преподаватель: Миронов Е. С.

Вариант: 3

Дата:

Оценка:

**Цель работы:**

Приобретение практических навыков в:

* Освоение принципов работы с файловыми системами
* Обеспечение обмена данных между процессами посредством технологии «File mapping»

**Введение**

Данную лабораторную работу я выполнял в среде UNIX. Мне потребовалось 2 процесса, по условию требуется передавать в две стороны, для этого созданы два shared\_memory object. Для того, чтобы предотвратить критические ситуации, когда два процесса хотят получить доступ к общей области памяти, используются

семафоры.

**Системные вызовы**

**void exit(int status) –** завершает работу программу с кодом возврата *status.*

**pid\_t fork() –** создает новый процесс, возвращает 0, для нового процесса, для родителя возвращает PID.

**void \*mmap(void \****addr***, size\_t** *length***, int** *prot***, int** *flags***,**

**int** *fd***, off\_t** *offset***) –** отображает *length* байтов, со смещением *offset*, файла с дескриптором *fd*, начиная с адреса *start.*

**int munmap(void \**start*, size\_t *length*) –** снимает отображение *length* байтов, начиная с *start*.

**sem\_t \*sem\_open(const char \**name*, int *oflag*,**

**mode\_t *mode*, unsigned int *value*) –** создает или открывает существующий семафор, с изначальным значением *value*. В случае успеха возвращает указатель на только что созданный семафор.

**int sem\_post(sem\_t \****sem***) –** увеличивает счетчик семафора *sem* на 1.

**int sem\_wait(sem\_t \**sem*) –** уменьшает счетчик семафора, пока значение больше 0. Если значение равняется 0, тогда вызов блокируется, пока не получится сделать декремент.

**int shm\_open(const char \**name*, int *oflag*, mode\_t *mode*) –** создание или открытие существующего общего объекта в памяти.

**Исходный код**

**parent.c**

#include "stdlib.h"

#include "unistd.h"

#include "stdio.h"

#include "semaphore.h"

#include "sys/wait.h"

#include "sys/mman.h"

#include "string.h"

#include "names.h"

int main(){

    char \* filename = NULL;

    size\_t length;

    if(getline(&filename,&length,stdin) == -1){

        perror("Error with getline!\n");

        exit(EXIT\_FAILURE);

    }

    filename[strlen(filename) - 1] = '\0';

    int shm\_input = shm\_open(InputFile, O\_RDWR | O\_CREAT, params);

    int shm\_error = shm\_open(ErrorFile, O\_RDWR | O\_CREAT, params);

    if(shm\_input == -1 || shm\_error == -1){

        perror("Error with opening shared file!\n");

        exit(EXIT\_FAILURE);

    }

    if(ftruncate(shm\_input, MAPPING\_SIZE) == -1){

        perror("Error with ftruncate!\n");

        exit(EXIT\_FAILURE);

    }

    if(ftruncate(shm\_error,MAPPING\_SIZE) == -1){

        perror("Error with ftruncate!\n");

        exit(EXIT\_FAILURE);

    }

    char \* addr\_input = mmap(0, MAPPING\_SIZE, PROT\_WRITE | PROT\_READ, MAP\_SHARED, shm\_input, 0);

    char \* addr\_error = mmap(0, MAPPING\_SIZE, PROT\_WRITE | PROT\_READ, MAP\_SHARED, shm\_error, 0);

    if(addr\_input == (char \*)-1 || addr\_error == (char \*)-1){

        perror("Error with mmap!\n");

        exit(EXIT\_FAILURE);

    }

    sem\_t \* sem\_input = sem\_open(InputSemaphore, O\_CREAT, params, 1);

    sem\_t \* sem\_error = sem\_open(ErrorSemaphore, O\_CREAT, params, 1);

    int input\_value, error\_value;

    if(sem\_getvalue(sem\_input, &input\_value) != 0){

        perror("Error with sem\_getvalue!\n");

        exit(EXIT\_FAILURE);

    }

    if(sem\_getvalue(sem\_error, &error\_value) != 0){

        perror("Error with sem\_getvalue!\n");

        exit(EXIT\_FAILURE);

    }

    while(input\_value < 1){

        sem\_post(sem\_input);

    }

    while(error\_value < 1){

        sem\_post(sem\_error);

    }

    int pid = fork();

    if(!pid){

        sem\_close(sem\_input);

        sem\_close(sem\_error);

        munmap(addr\_input, MAPPING\_SIZE);

        munmap(addr\_error, MAPPING\_SIZE);

        execl("child", "child", filename, NULL);

        perror("execl\n");

        exit(EXIT\_FAILURE);

        exit(0);

    }

    memset(addr\_input, '\0', sizeof(addr\_input));

    fflush(STDIN\_FILENO);

    char c;

    char buf[512];

    memset(buf, '\0', sizeof(buf));

    int ptr = 0;

    while((c = getchar()) != EOF){

        buf[ptr++] = c;

        if(c == '\n'){

            while(1){

                if(sem\_wait(sem\_input) == 0){

                    sprintf(addr\_input, "%s", buf);

                    sem\_post(sem\_input);

                    memset(buf,'\0', sizeof(buf));

                    ptr = 0;

                    break;

                }

                else{

                    exit(EXIT\_FAILURE);

                }

            }

        }

    }

    while(1){

            if(sem\_wait(sem\_input) == 0){

                addr\_input[0] = EOF;

                sem\_post(sem\_input);

                break;

            }

            else{

                exit(EXIT\_FAILURE);

            }

    }

    while(1){

        if(sem\_wait(sem\_error) == 0) {

            if(addr\_error[0] == EOF)break;

            if(addr\_error[0] == '\0'){

                sem\_post(sem\_error);

                continue;

            }

            char \* str = (char \*)(malloc(sizeof(char) \* strlen(addr\_error)));

            strcpy(str, addr\_error);

            printf("%s", str);

            sem\_post(sem\_error);

            memset(addr\_error, '\0', sizeof(addr\_error));

            free(str);

            break;

        }

        else{

            printf("Error sem\_wait!\n");

            exit(EXIT\_FAILURE);

        }

    }

    int status;

    if(wait(&status) == -1){

        perror("wait");

    }

    if(!WIFEXITED(status) || (WIFEXITED(status) && (WEXITSTATUS(status)) != 0)){

        fprintf(stderr, "Error in child process!\n");

        return 1;

    }

    munmap(addr\_error, MAPPING\_SIZE);

    munmap(addr\_input, MAPPING\_SIZE);

    sem\_close(sem\_input);

    sem\_close(sem\_error);

    return 0;

}

**child.c**

#include "stdlib.h"

#include "unistd.h"

#include "stdio.h"

#include "semaphore.h"

#include "sys/wait.h"

#include "sys/mman.h"

#include "string.h"

#include "names.h"

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

    int shm\_error = shm\_open(ErrorFile, O\_RDWR | O\_CREAT, params);

    char \* addr\_error = mmap(0, MAPPING\_SIZE, PROT\_WRITE | PROT\_READ, MAP\_SHARED, shm\_error, 0);

    sem\_t \* semerror = sem\_open(ErrorSemaphore, O\_CREAT, params, 1);

    if(shm\_error == -1){

        perror("Error with opening shared file!\n");

        exit(EXIT\_FAILURE);

    }

    if(ftruncate(shm\_error,MAPPING\_SIZE) == -1){

        perror("Error with ftruncate!\n");

        exit(EXIT\_FAILURE);

    }

    char \* filename = argv[1];

    FILE \* out = fopen(filename, "w");

    if(out == NULL){

        while(1){

            if(sem\_wait(semerror) == 0){

                perror("File not opened\n");

                sem\_post(semerror);

                exit(EXIT\_FAILURE);

            }

            else {

                perror("Error with sem\_wait\n");

                exit(EXIT\_FAILURE);

            }

        }

    }

    int shm\_input = shm\_open(InputFile, O\_RDWR | O\_CREAT, params);

    if(shm\_input == -1 ){

        perror("Error with opening shared file!\n");

        exit(EXIT\_FAILURE);

    }

    if(ftruncate(shm\_input, MAPPING\_SIZE) == -1){

        perror("Error with ftruncate!\n");

        exit(EXIT\_FAILURE);

    }

    char \* addr\_input = mmap(0, MAPPING\_SIZE, PROT\_WRITE | PROT\_READ, MAP\_SHARED, shm\_input, 0);

    sem\_t \* semptr = sem\_open(InputSemaphore, O\_CREAT, params, 1);

    while(1){

        if(sem\_wait(semptr) == 0){

            if(addr\_input[0] == EOF)break;

            if(addr\_input[0] == '\0'){

                sem\_post(semptr);

                continue;

            }

            char \* parent\_input = (char \* )malloc(strlen(addr\_input) \* sizeof(char));

            strcpy(parent\_input, addr\_input);

            int value = 0;

            double result = 0;

            int flag = 1;

            for(size\_t i = 0;parent\_input[i] != '\0';++i){

                char c = parent\_input[i];

                if(c == ' '){

                    if(flag){

                        result = value;

                        flag = 0;

                    }

                    else{

                        if(value == 0){

                            while(1){

                                if(sem\_wait(semerror) == 0){

                                    sprintf(addr\_error, "Zero division!\n");

                                    sem\_post(semerror);

                                    exit(EXIT\_FAILURE);

                                }

                            }

                        }

                        result /= (double)value;

                    }

                    value = 0;

                }

                else if(c == '\n'){

                    if(flag){

                        result = value;

                        flag = 0;

                    }

                    else{

                        if(value == 0){

                            while(1){

                                if(sem\_wait(semerror) == 0){

                                    sprintf(addr\_error, "Zero division!\n");

                                    sem\_post(semerror);

                                    exit(EXIT\_FAILURE);

                                }

                            }

                        }

                        result /= (double)value;

                    }

                    value = 0;

                    break;

                }

                else{

                    value \*= 10;

                    value += (c - '0');

                }

            }

            fprintf(out, "%f\n", result);

            memset(addr\_input, '\0', sizeof(addr\_input));

            free(parent\_input);

            sem\_post(semptr);

        }

        else {

            perror("Error with sem\_wait\n");

            exit(EXIT\_FAILURE);

        }

    }

    while(1){

        if(sem\_wait(semerror) == 0){

            addr\_error[0] = EOF;

            sem\_post(semerror);

            break;

        }

    }

    sem\_close(semptr);

    sem\_close(semerror);

    munmap(addr\_error, MAPPING\_SIZE);

    munmap(addr\_input, MAPPING\_SIZE);

    fclose(out);

    return 0;

}

names.h

#ifndef NAMES\_H

#define NAMES\_H

#include "fcntl.h"

const size\_t MAPPING\_SIZE = 4 \* 1024;

const unsigned int params = 0777;

const char \* InputFile = "input.shared";

const char \* InputSemaphore = "/input.semaphore";

const char \* ErrorFile = "error.shared";

const char \* ErrorSemaphore = "/error.semaphore";

#endif

**Пример работы**

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ gcc parent.c -o parent -lrt -lpthread*

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ gcc child.c -o child -lrt -lpthread*

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ ./parent*

*out.txt*

*1 2 3*

*3 4 5*

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ cat out.txt*

*0.166667*

*0.150000*

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ ./parent*

*file.txt*

*2 5 10*

*1 3*

*2 6*

*1 10*

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ cat file.txt*

*0.040000*

*0.333333*

*0.333333*

*0.100000*

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ ./parent*

*out.txt*

*2 5*

*3 10*

*2 0 1*

*Zero division!*

*Error in child process!*

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ cat out.txt*

*0.400000*

*0.300000*

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ ./parent*

*out.txt*

*1 0 0*

*3 4*

*5 6*

*Zero division!*

*Error in child process!*

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ cat out.txt*

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ ./parent*

*out.txt*

*1 2 0*

*3 5*

*Zero division!*

*Error in child process!*

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ cat out.txt*

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ ./parent*

*out.txt*

*1 5*

*2 0*

*3 6*

*Zero division!*

*Error in child process!*

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ cat out.txt*

*0.200000*

**Strace**

Без деления на ноль:

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ strace -f ./parent*

*execve("./parent", ["./parent"], 0x7ffc3f8cad88 /\* 29 vars \*/) = 0*

*brk(NULL) = 0x55d60ad54000*

*arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffcc393a950) = -1 EINVAL (Invalid argument)*

*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*

*fstat(3, {st\_mode=S\_IFREG|0644, st\_size=35431, ...}) = 0*

*mmap(NULL, 35431, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f6acf5e6000*

*close(3) = 0*

*openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/librt.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 7\0\0\0\0\0\0"..., 832) = 832*

*fstat(3, {st\_mode=S\_IFREG|0644, st\_size=40040, ...}) = 0*

*mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f6acf5e4000*

*mmap(NULL, 44000, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f6acf5d9000*

*mprotect(0x7f6acf5dc000, 24576, PROT\_NONE) = 0*

*mmap(0x7f6acf5dc000, 16384, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x3000) = 0x7f6acf5dc000*

*mmap(0x7f6acf5e0000, 4096, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x7000) = 0x7f6acf5e0000*

*mmap(0x7f6acf5e2000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x8000) = 0x7f6acf5e2000*

*close(3) = 0*

*openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libpthread.so.0", 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\220\201\0\0\0\0\0\0"..., 832) = 832*

*pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\345Ga\367\265T\320\374\301V)Yf]\223\337"..., 68, 824) = 68*

*fstat(3, {st\_mode=S\_IFREG|0755, st\_size=157224, ...}) = 0*

*pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\345Ga\367\265T\320\374\301V)Yf]\223\337"..., 68, 824) = 68*

*mmap(NULL, 140408, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f6acf5b6000*

*mmap(0x7f6acf5bd000, 69632, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x7000) = 0x7f6acf5bd000*

*mmap(0x7f6acf5ce000, 20480, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x18000) = 0x7f6acf5ce000*

*mmap(0x7f6acf5d3000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1c000) = 0x7f6acf5d3000*

*mmap(0x7f6acf5d5000, 13432, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f6acf5d5000*

*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\0\360q\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\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32*

*pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\t\233\222%\274\260\320\31\331\326\10\204\276X>\263"..., 68, 880) = 68*

*fstat(3, {st\_mode=S\_IFREG|0755, st\_size=2029224, ...}) = 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*

*pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32*

*pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\t\233\222%\274\260\320\31\331\326\10\204\276X>\263"..., 68, 880) = 68*

*mmap(NULL, 2036952, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f6acf3c4000*

*mprotect(0x7f6acf3e9000, 1847296, PROT\_NONE) = 0*

*mmap(0x7f6acf3e9000, 1540096, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x25000) = 0x7f6acf3e9000*

*mmap(0x7f6acf561000, 303104, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x19d000) = 0x7f6acf561000*

*mmap(0x7f6acf5ac000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1e7000) = 0x7f6acf5ac000*

*mmap(0x7f6acf5b2000, 13528, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f6acf5b2000*

*close(3) = 0*

*mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f6acf3c1000*

*arch\_prctl(ARCH\_SET\_FS, 0x7f6acf3c1740) = 0*

*mprotect(0x7f6acf5ac000, 12288, PROT\_READ) = 0*

*mprotect(0x7f6acf5d3000, 4096, PROT\_READ) = 0*

*mprotect(0x7f6acf5e2000, 4096, PROT\_READ) = 0*

*mprotect(0x55d60a5de000, 4096, PROT\_READ) = 0*

*mprotect(0x7f6acf61c000, 4096, PROT\_READ) = 0*

*munmap(0x7f6acf5e6000, 35431) = 0*

*set\_tid\_address(0x7f6acf3c1a10) = 2207*

*set\_robust\_list(0x7f6acf3c1a20, 24) = 0*

*rt\_sigaction(SIGRTMIN, {sa\_handler=0x7f6acf5bdbf0, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_SIGINFO, sa\_restorer=0x7f6acf5cb3c0}, NULL, 8) = 0*

*rt\_sigaction(SIGRT\_1, {sa\_handler=0x7f6acf5bdc90, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_RESTART|SA\_SIGINFO, sa\_restorer=0x7f6acf5cb3c0}, NULL, 8) = 0*

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

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

*brk(NULL) = 0x55d60ad54000*

*brk(0x55d60ad75000) = 0x55d60ad75000*

*fstat(0, {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x1), ...}) = 0*

*read(0, out.txt*

*"out.txt\n", 1024) = 8*

*statfs("/dev/shm/", {f\_type=TMPFS\_MAGIC, f\_bsize=4096, f\_blocks=702382, f\_bfree=702378, f\_bavail=702378, f\_files=702382, f\_ffree=702377, f\_fsid={val=[0, 0]}, f\_namelen=255, f\_frsize=4096, f\_flags=ST\_VALID|ST\_NOSUID|ST\_NODEV|ST\_NOATIME}) = 0*

*futex(0x7f6acf5d8390, FUTEX\_WAKE\_PRIVATE, 2147483647) = 0*

*openat(AT\_FDCWD, "/dev/shm/input.shared", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0777) = 3*

*openat(AT\_FDCWD, "/dev/shm/error.shared", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0777) = 4*

*ftruncate(3, 4096) = 0*

*ftruncate(4, 4096) = 0*

*mmap(NULL, 4096, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f6acf61b000*

*mmap(NULL, 4096, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 4, 0) = 0x7f6acf5ee000*

*openat(AT\_FDCWD, "/dev/shm/sem.input.semaphore", O\_RDWR|O\_NOFOLLOW) = 5*

*fstat(5, {st\_mode=S\_IFREG|0755, st\_size=32, ...}) = 0*

*mmap(NULL, 32, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 5, 0) = 0x7f6acf5ed000*

*close(5) = 0*

*openat(AT\_FDCWD, "/dev/shm/sem.error.semaphore", O\_RDWR|O\_NOFOLLOW) = 5*

*fstat(5, {st\_mode=S\_IFREG|0755, st\_size=32, ...}) = 0*

*mmap(NULL, 32, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 5, 0) = 0x7f6acf5ec000*

*close(5) = 0*

*clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLDstrace: Process 2211 attached*

*, child\_tidptr=0x7f6acf3c1a10) = 2211*

*[pid 2211] set\_robust\_list(0x7f6acf3c1a20, 24 <unfinished ...>*

*[pid 2207] read(0, <unfinished ...>*

*[pid 2211] <... set\_robust\_list resumed>) = 0*

*[pid 2211] munmap(0x7f6acf5ed000, 32) = 0*

*[pid 2211] munmap(0x7f6acf5ec000, 32) = 0*

*[pid 2211] munmap(0x7f6acf61b000, 4096) = 0*

*[pid 2211] munmap(0x7f6acf5ee000, 4096) = 0*

*[pid 2211] execve("child", ["child", "out.txt"], 0x7ffcc393aa38 /\* 29 vars \*/) = 0*

*[pid 2211] brk(NULL) = 0x555adce51000*

*[pid 2211] arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffe7e6ca870) = -1 EINVAL (Invalid argument)*

*[pid 2211] access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (No such file or directory)*

*[pid 2211] openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3*

*[pid 2211] fstat(3, {st\_mode=S\_IFREG|0644, st\_size=35431, ...}) = 0*

*[pid 2211] mmap(NULL, 35431, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f828f884000*

*[pid 2211] close(3) = 0*

*[pid 2211] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/librt.so.1", O\_RDONLY|O\_CLOEXEC) = 3*

*[pid 2211] read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0 7\0\0\0\0\0\0"..., 832) = 832*

*[pid 2211] fstat(3, {st\_mode=S\_IFREG|0644, st\_size=40040, ...}) = 0*

*[pid 2211] mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f828f882000*

*[pid 2211] mmap(NULL, 44000, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f828f877000*

*[pid 2211] mprotect(0x7f828f87a000, 24576, PROT\_NONE) = 0*

*[pid 2211] mmap(0x7f828f87a000, 16384, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x3000) = 0x7f828f87a000*

*[pid 2211] mmap(0x7f828f87e000, 4096, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x7000) = 0x7f828f87e000*

*[pid 2211] mmap(0x7f828f880000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x8000) = 0x7f828f880000*

*[pid 2211] close(3) = 0*

*[pid 2211] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libpthread.so.0", O\_RDONLY|O\_CLOEXEC) = 3*

*[pid 2211] read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\201\0\0\0\0\0\0"..., 832) = 832*

*[pid 2211] pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\345Ga\367\265T\320\374\301V)Yf]\223\337"..., 68, 824) = 68*

*[pid 2211] fstat(3, {st\_mode=S\_IFREG|0755, st\_size=157224, ...}) = 0*

*[pid 2211] pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\345Ga\367\265T\320\374\301V)Yf]\223\337"..., 68, 824) = 68*

*[pid 2211] mmap(NULL, 140408, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f828f854000*

*[pid 2211] mmap(0x7f828f85b000, 69632, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x7000) = 0x7f828f85b000*

*[pid 2211] mmap(0x7f828f86c000, 20480, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x18000) = 0x7f828f86c000*

*[pid 2211] mmap(0x7f828f871000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1c000) = 0x7f828f871000*

*[pid 2211] mmap(0x7f828f873000, 13432, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f828f873000*

*[pid 2211] close(3) = 0*

*[pid 2211] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC) = 3*

*[pid 2211] read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\360q\2\0\0\0\0\0"..., 832) = 832*

*[pid 2211] 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*

*[pid 2211] pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32*

*[pid 2211] pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\t\233\222%\274\260\320\31\331\326\10\204\276X>\263"..., 68, 880) = 68*

*[pid 2211] fstat(3, {st\_mode=S\_IFREG|0755, st\_size=2029224, ...}) = 0*

*[pid 2211] 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*

*[pid 2211] pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32*

*[pid 2211] pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\t\233\222%\274\260\320\31\331\326\10\204\276X>\263"..., 68, 880) = 68*

*[pid 2211] mmap(NULL, 2036952, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f828f662000*

*[pid 2211] mprotect(0x7f828f687000, 1847296, PROT\_NONE) = 0*

*[pid 2211] mmap(0x7f828f687000, 1540096, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x25000) = 0x7f828f687000*

*[pid 2211] mmap(0x7f828f7ff000, 303104, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x19d000) = 0x7f828f7ff000*

*[pid 2211] mmap(0x7f828f84a000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1e7000) = 0x7f828f84a000*

*[pid 2211] mmap(0x7f828f850000, 13528, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f828f850000*

*[pid 2211] close(3) = 0*

*[pid 2211] mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f828f65f000*

*[pid 2211] arch\_prctl(ARCH\_SET\_FS, 0x7f828f65f740) = 0*

*[pid 2211] mprotect(0x7f828f84a000, 12288, PROT\_READ) = 0*

*[pid 2211] mprotect(0x7f828f871000, 4096, PROT\_READ) = 0*

*[pid 2211] mprotect(0x7f828f880000, 4096, PROT\_READ) = 0*

*[pid 2211] mprotect(0x555adb08c000, 4096, PROT\_READ) = 0*

*[pid 2211] mprotect(0x7f828f8ba000, 4096, PROT\_READ) = 0*

*[pid 2211] munmap(0x7f828f884000, 35431) = 0*

*[pid 2211] set\_tid\_address(0x7f828f65fa10) = 2211*

*[pid 2211] set\_robust\_list(0x7f828f65fa20, 24) = 0*

*[pid 2211] rt\_sigaction(SIGRTMIN, {sa\_handler=0x7f828f85bbf0, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_SIGINFO, sa\_restorer=0x7f828f8693c0}, NULL, 8) = 0*

*[pid 2211] rt\_sigaction(SIGRT\_1, {sa\_handler=0x7f828f85bc90, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_RESTART|SA\_SIGINFO, sa\_restorer=0x7f828f8693c0}, NULL, 8) = 0*

*[pid 2211] rt\_sigprocmask(SIG\_UNBLOCK, [RTMIN RT\_1], NULL, 8) = 0*

*[pid 2211] prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0*

*[pid 2211] statfs("/dev/shm/", {f\_type=TMPFS\_MAGIC, f\_bsize=4096, f\_blocks=702382, f\_bfree=702378, f\_bavail=702378, f\_files=702382, f\_ffree=702377, f\_fsid={val=[0, 0]}, f\_namelen=255, f\_frsize=4096, f\_flags=ST\_VALID|ST\_NOSUID|ST\_NODEV|ST\_NOATIME}) = 0*

*[pid 2211] futex(0x7f828f876390, FUTEX\_WAKE\_PRIVATE, 2147483647) = 0*

*[pid 2211] openat(AT\_FDCWD, "/dev/shm/error.shared", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0777) = 3*

*[pid 2211] mmap(NULL, 4096, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f828f8b9000*

*[pid 2211] openat(AT\_FDCWD, "/dev/shm/sem.error.semaphore", O\_RDWR|O\_NOFOLLOW) = 4*

*[pid 2211] fstat(4, {st\_mode=S\_IFREG|0755, st\_size=32, ...}) = 0*

*[pid 2211] brk(NULL) = 0x555adce51000*

*[pid 2211] brk(0x555adce72000) = 0x555adce72000*

*[pid 2211] mmap(NULL, 32, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 4, 0) = 0x7f828f88c000*

*[pid 2211] close(4) = 0*

*[pid 2211] ftruncate(3, 4096) = 0*

*[pid 2211] openat(AT\_FDCWD, "out.txt", O\_WRONLY|O\_CREAT|O\_TRUNC, 0666) = 4*

*[pid 2211] openat(AT\_FDCWD, "/dev/shm/input.shared", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0777) = 5*

*[pid 2211] ftruncate(5, 4096) = 0*

*[pid 2211] mmap(NULL, 4096, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 5, 0) = 0x7f828f88b000*

*[pid 2211] openat(AT\_FDCWD, "/dev/shm/sem.input.semaphore", O\_RDWR|O\_NOFOLLOW) = 6*

*[pid 2211] fstat(6, {st\_mode=S\_IFREG|0755, st\_size=32, ...}) = 0*

*[pid 2211] mmap(NULL, 32, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 6, 0) = 0x7f828f88a000*

*[pid 2211] close(6) = 0*

*1 2 3*

*[pid 2207] <... read resumed>"1 2 3\n", 1024) = 6*

*[pid 2207] read(0, <unfinished ...>*

*[pid 2211] futex(0x7f828f88a000, FUTEX\_WAKE, 1) = 0*

*[pid 2211] fstat(4, {st\_mode=S\_IFREG|0777, st\_size=0, ...}) = 0*

*4 5 6*

*[pid 2207] <... read resumed>"4 5 6\n", 1024) = 6*

*[pid 2207] read(0, 2 3*

*"2 3\n", 1024) = 4*

*[pid 2207] read(0, "", 1024) = 0*

*[pid 2207] wait4(-1, <unfinished ...>*

*[pid 2211] munmap(0x7f828f88a000, 32) = 0*

*[pid 2211] munmap(0x7f828f88c000, 32) = 0*

*[pid 2211] munmap(0x7f828f8b9000, 4096) = 0*

*[pid 2211] munmap(0x7f828f88b000, 4096) = 0*

*[pid 2211] write(4, "0.166667\n0.133333\n0.666667\n", 27) = 27*

*[pid 2211] close(4) = 0*

*[pid 2211] exit\_group(0) = ?*

*[pid 2211] +++ exited with 0 +++*

*<... wait4 resumed>[{WIFEXITED(s) && WEXITSTATUS(s) == 0}], 0, NULL) = 2211*

*--- SIGCHLD {si\_signo=SIGCHLD, si\_code=CLD\_EXITED, si\_pid=2211, si\_uid=1000, si\_status=0, si\_utime=466, si\_stime=1} ---*

*munmap(0x7f6acf5ee000, 4096) = 0*

*munmap(0x7f6acf61b000, 4096) = 0*

*munmap(0x7f6acf5ed000, 32) = 0*

*munmap(0x7f6acf5ec000, 32) = 0*

*exit\_group(0) = ?*

*+++ exited with 0 +++*

В случае, если происходит деление на 0.

*egorb@LAPTOP-TIFMO0IA:/mnt/d/os/lab4$ strace -f ./parent*

*execve("./parent", ["./parent"], 0x7fff494c32a8 /\* 29 vars \*/) = 0*

*brk(NULL) = 0x561c78747000*

*arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffc8cba74e0) = -1 EINVAL (Invalid argument)*

*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*

*fstat(3, {st\_mode=S\_IFREG|0644, st\_size=35431, ...}) = 0*

*mmap(NULL, 35431, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f898d27e000*

*close(3) = 0*

*openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/librt.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 7\0\0\0\0\0\0"..., 832) = 832*

*fstat(3, {st\_mode=S\_IFREG|0644, st\_size=40040, ...}) = 0*

*mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f898d27c000*

*mmap(NULL, 44000, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f898d271000*

*mprotect(0x7f898d274000, 24576, PROT\_NONE) = 0*

*mmap(0x7f898d274000, 16384, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x3000) = 0x7f898d274000*

*mmap(0x7f898d278000, 4096, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x7000) = 0x7f898d278000*

*mmap(0x7f898d27a000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x8000) = 0x7f898d27a000*

*close(3) = 0*

*openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libpthread.so.0", 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\220\201\0\0\0\0\0\0"..., 832) = 832*

*pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\345Ga\367\265T\320\374\301V)Yf]\223\337"..., 68, 824) = 68*

*fstat(3, {st\_mode=S\_IFREG|0755, st\_size=157224, ...}) = 0*

*pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\345Ga\367\265T\320\374\301V)Yf]\223\337"..., 68, 824) = 68*

*mmap(NULL, 140408, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f898d24e000*

*mmap(0x7f898d255000, 69632, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x7000) = 0x7f898d255000*

*mmap(0x7f898d266000, 20480, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x18000) = 0x7f898d266000*

*mmap(0x7f898d26b000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1c000) = 0x7f898d26b000*

*mmap(0x7f898d26d000, 13432, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f898d26d000*

*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\0\360q\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\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32*

*pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\t\233\222%\274\260\320\31\331\326\10\204\276X>\263"..., 68, 880) = 68*

*fstat(3, {st\_mode=S\_IFREG|0755, st\_size=2029224, ...}) = 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*

*pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32*

*pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\t\233\222%\274\260\320\31\331\326\10\204\276X>\263"..., 68, 880) = 68*

*mmap(NULL, 2036952, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f898d05c000*

*mprotect(0x7f898d081000, 1847296, PROT\_NONE) = 0*

*mmap(0x7f898d081000, 1540096, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x25000) = 0x7f898d081000*

*mmap(0x7f898d1f9000, 303104, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x19d000) = 0x7f898d1f9000*

*mmap(0x7f898d244000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1e7000) = 0x7f898d244000*

*mmap(0x7f898d24a000, 13528, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f898d24a000*

*close(3) = 0*

*mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f898d059000*

*arch\_prctl(ARCH\_SET\_FS, 0x7f898d059740) = 0*

*mprotect(0x7f898d244000, 12288, PROT\_READ) = 0*

*mprotect(0x7f898d26b000, 4096, PROT\_READ) = 0*

*mprotect(0x7f898d27a000, 4096, PROT\_READ) = 0*

*mprotect(0x561c78431000, 4096, PROT\_READ) = 0*

*mprotect(0x7f898d2b4000, 4096, PROT\_READ) = 0*

*munmap(0x7f898d27e000, 35431) = 0*

*set\_tid\_address(0x7f898d059a10) = 3044*

*set\_robust\_list(0x7f898d059a20, 24) = 0*

*rt\_sigaction(SIGRTMIN, {sa\_handler=0x7f898d255bf0, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_SIGINFO, sa\_restorer=0x7f898d2633c0}, NULL, 8) = 0*

*rt\_sigaction(SIGRT\_1, {sa\_handler=0x7f898d255c90, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_RESTART|SA\_SIGINFO, sa\_restorer=0x7f898d2633c0}, NULL, 8) = 0*

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

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

*brk(NULL) = 0x561c78747000*

*brk(0x561c78768000) = 0x561c78768000*

*fstat(0, {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x1), ...}) = 0*

*read(0, out.txt*

*"out.txt\n", 1024) = 8*

*statfs("/dev/shm/", {f\_type=TMPFS\_MAGIC, f\_bsize=4096, f\_blocks=702382, f\_bfree=702378, f\_bavail=702378, f\_files=702382, f\_ffree=702377, f\_fsid={val=[0, 0]}, f\_namelen=255, f\_frsize=4096, f\_flags=ST\_VALID|ST\_NOSUID|ST\_NODEV|ST\_NOATIME}) = 0*

*futex(0x7f898d270390, FUTEX\_WAKE\_PRIVATE, 2147483647) = 0*

*openat(AT\_FDCWD, "/dev/shm/input.shared", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0777) = 3*

*openat(AT\_FDCWD, "/dev/shm/error.shared", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0777) = 4*

*ftruncate(3, 4096) = 0*

*ftruncate(4, 4096) = 0*

*mmap(NULL, 4096, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f898d2b3000*

*mmap(NULL, 4096, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 4, 0) = 0x7f898d286000*

*openat(AT\_FDCWD, "/dev/shm/sem.input.semaphore", O\_RDWR|O\_NOFOLLOW) = 5*

*fstat(5, {st\_mode=S\_IFREG|0755, st\_size=32, ...}) = 0*

*mmap(NULL, 32, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 5, 0) = 0x7f898d285000*

*close(5) = 0*

*openat(AT\_FDCWD, "/dev/shm/sem.error.semaphore", O\_RDWR|O\_NOFOLLOW) = 5*

*fstat(5, {st\_mode=S\_IFREG|0755, st\_size=32, ...}) = 0*

*mmap(NULL, 32, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 5, 0) = 0x7f898d284000*

*close(5) = 0*

*clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLDstrace: Process 3048 attached*

*, child\_tidptr=0x7f898d059a10) = 3048*

*[pid 3048] set\_robust\_list(0x7f898d059a20, 24 <unfinished ...>*

*[pid 3044] read(0, <unfinished ...>*

*[pid 3048] <... set\_robust\_list resumed>) = 0*

*[pid 3048] munmap(0x7f898d285000, 32) = 0*

*[pid 3048] munmap(0x7f898d284000, 32) = 0*

*[pid 3048] munmap(0x7f898d2b3000, 4096) = 0*

*[pid 3048] munmap(0x7f898d286000, 4096) = 0*

*[pid 3048] execve("child", ["child", "out.txt"], 0x7ffc8cba75c8 /\* 29 vars \*/) = 0*

*[pid 3048] brk(NULL) = 0x5599be591000*

*[pid 3048] arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7fff92d7a520) = -1 EINVAL (Invalid argument)*

*[pid 3048] access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (No such file or directory)*

*[pid 3048] openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3*

*[pid 3048] fstat(3, {st\_mode=S\_IFREG|0644, st\_size=35431, ...}) = 0*

*[pid 3048] mmap(NULL, 35431, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f8597fd8000*

*[pid 3048] close(3) = 0*

*[pid 3048] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/librt.so.1", O\_RDONLY|O\_CLOEXEC) = 3*

*[pid 3048] read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0 7\0\0\0\0\0\0"..., 832) = 832*

*[pid 3048] fstat(3, {st\_mode=S\_IFREG|0644, st\_size=40040, ...}) = 0*

*[pid 3048] mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f8597fd6000*

*[pid 3048] mmap(NULL, 44000, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f8597fcb000*

*[pid 3048] mprotect(0x7f8597fce000, 24576, PROT\_NONE) = 0*

*[pid 3048] mmap(0x7f8597fce000, 16384, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x3000) = 0x7f8597fce000*

*[pid 3048] mmap(0x7f8597fd2000, 4096, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x7000) = 0x7f8597fd2000*

*[pid 3048] mmap(0x7f8597fd4000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x8000) = 0x7f8597fd4000*

*[pid 3048] close(3) = 0*

*[pid 3048] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libpthread.so.0", O\_RDONLY|O\_CLOEXEC) = 3*

*[pid 3048] read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\201\0\0\0\0\0\0"..., 832) = 832*

*[pid 3048] pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\345Ga\367\265T\320\374\301V)Yf]\223\337"..., 68, 824) = 68*

*[pid 3048] fstat(3, {st\_mode=S\_IFREG|0755, st\_size=157224, ...}) = 0*

*[pid 3048] pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\345Ga\367\265T\320\374\301V)Yf]\223\337"..., 68, 824) = 68*

*[pid 3048] mmap(NULL, 140408, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f8597fa8000*

*[pid 3048] mmap(0x7f8597faf000, 69632, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x7000) = 0x7f8597faf000*

*[pid 3048] mmap(0x7f8597fc0000, 20480, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x18000) = 0x7f8597fc0000*

*[pid 3048] mmap(0x7f8597fc5000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1c000) = 0x7f8597fc5000*

*[pid 3048] mmap(0x7f8597fc7000, 13432, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f8597fc7000*

*[pid 3048] close(3) = 0*

*[pid 3048] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC) = 3*

*[pid 3048] read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\360q\2\0\0\0\0\0"..., 832) = 832*

*[pid 3048] 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*

*[pid 3048] pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32*

*[pid 3048] pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\t\233\222%\274\260\320\31\331\326\10\204\276X>\263"..., 68, 880) = 68*

*[pid 3048] fstat(3, {st\_mode=S\_IFREG|0755, st\_size=2029224, ...}) = 0*

*[pid 3048] 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*

*[pid 3048] pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32*

*[pid 3048] pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\t\233\222%\274\260\320\31\331\326\10\204\276X>\263"..., 68, 880) = 68*

*[pid 3048] mmap(NULL, 2036952, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f8597db6000*

*[pid 3048] mprotect(0x7f8597ddb000, 1847296, PROT\_NONE) = 0*

*[pid 3048] mmap(0x7f8597ddb000, 1540096, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x25000) = 0x7f8597ddb000*

*[pid 3048] mmap(0x7f8597f53000, 303104, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x19d000) = 0x7f8597f53000*

*[pid 3048] mmap(0x7f8597f9e000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1e7000) = 0x7f8597f9e000*

*[pid 3048] mmap(0x7f8597fa4000, 13528, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f8597fa4000*

*[pid 3048] close(3) = 0*

*[pid 3048] mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f8597db3000*

*[pid 3048] arch\_prctl(ARCH\_SET\_FS, 0x7f8597db3740) = 0*

*[pid 3048] mprotect(0x7f8597f9e000, 12288, PROT\_READ) = 0*

*[pid 3048] mprotect(0x7f8597fc5000, 4096, PROT\_READ) = 0*

*[pid 3048] mprotect(0x7f8597fd4000, 4096, PROT\_READ) = 0*

*[pid 3048] mprotect(0x5599bd285000, 4096, PROT\_READ) = 0*

*[pid 3048] mprotect(0x7f859800e000, 4096, PROT\_READ) = 0*

*[pid 3048] munmap(0x7f8597fd8000, 35431) = 0*

*[pid 3048] set\_tid\_address(0x7f8597db3a10) = 3048*

*[pid 3048] set\_robust\_list(0x7f8597db3a20, 24) = 0*

*[pid 3048] rt\_sigaction(SIGRTMIN, {sa\_handler=0x7f8597fafbf0, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_SIGINFO, sa\_restorer=0x7f8597fbd3c0}, NULL, 8) = 0*

*[pid 3048] rt\_sigaction(SIGRT\_1, {sa\_handler=0x7f8597fafc90, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_RESTART|SA\_SIGINFO, sa\_restorer=0x7f8597fbd3c0}, NULL, 8) = 0*

*[pid 3048] rt\_sigprocmask(SIG\_UNBLOCK, [RTMIN RT\_1], NULL, 8) = 0*

*[pid 3048] prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0*

*[pid 3048] statfs("/dev/shm/", {f\_type=TMPFS\_MAGIC, f\_bsize=4096, f\_blocks=702382, f\_bfree=702378, f\_bavail=702378, f\_files=702382, f\_ffree=702377, f\_fsid={val=[0, 0]}, f\_namelen=255, f\_frsize=4096, f\_flags=ST\_VALID|ST\_NOSUID|ST\_NODEV|ST\_NOATIME}) = 0*

*[pid 3048] futex(0x7f8597fca390, FUTEX\_WAKE\_PRIVATE, 2147483647) = 0*

*[pid 3048] openat(AT\_FDCWD, "/dev/shm/error.shared", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0777) = 3*

*[pid 3048] mmap(NULL, 4096, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x7f859800d000*

*[pid 3048] openat(AT\_FDCWD, "/dev/shm/sem.error.semaphore", O\_RDWR|O\_NOFOLLOW) = 4*

*[pid 3048] fstat(4, {st\_mode=S\_IFREG|0755, st\_size=32, ...}) = 0*

*[pid 3048] brk(NULL) = 0x5599be591000*

*[pid 3048] brk(0x5599be5b2000) = 0x5599be5b2000*

*[pid 3048] mmap(NULL, 32, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 4, 0) = 0x7f8597fe0000*

*[pid 3048] close(4) = 0*

*[pid 3048] ftruncate(3, 4096) = 0*

*[pid 3048] openat(AT\_FDCWD, "out.txt", O\_WRONLY|O\_CREAT|O\_TRUNC, 0666) = 4*

*[pid 3048] openat(AT\_FDCWD, "/dev/shm/input.shared", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0777) = 5*

*[pid 3048] ftruncate(5, 4096) = 0*

*[pid 3048] mmap(NULL, 4096, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 5, 0) = 0x7f8597fdf000*

*[pid 3048] openat(AT\_FDCWD, "/dev/shm/sem.input.semaphore", O\_RDWR|O\_NOFOLLOW) = 6*

*[pid 3048] fstat(6, {st\_mode=S\_IFREG|0755, st\_size=32, ...}) = 0*

*[pid 3048] mmap(NULL, 32, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 6, 0) = 0x7f8597fde000*

*[pid 3048] close(6) = 0*

*2 3*

*[pid 3044] <... read resumed>"2 3\n", 1024) = 4*

*[pid 3044] read(0, <unfinished ...>*

*[pid 3048] fstat(4, {st\_mode=S\_IFREG|0777, st\_size=0, ...}) = 0*

*1 0*

*[pid 3044] <... read resumed>"1 0\n", 1024) = 4*

*[pid 3044] read(0, <unfinished ...>*

*[pid 3048] write(4, "0.666667\n", 9) = 9*

*[pid 3048] exit\_group(1) = ?*

*[pid 3048] +++ exited with 1 +++*

*<... read resumed>0x561c78747320, 1024) = ? ERESTARTSYS (To be restarted if SA\_RESTART is set)*

*--- SIGCHLD {si\_signo=SIGCHLD, si\_code=CLD\_EXITED, si\_pid=3048, si\_uid=1000, si\_status=1, si\_utime=182, si\_stime=1} ---*

*read(0, "", 1024) = 0*

*fstat(1, {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x1), ...}) = 0*

*write(1, "Zero division!\n", 15Zero division!*

*) = 15*

*wait4(-1, [{WIFEXITED(s) && WEXITSTATUS(s) == 1}], 0, NULL) = 3048*

*write(2, "Error in child process!\n", 24Error in child process!*

*) = 24*

*exit\_group(1) = ?*

*+++ exited with 1 +++*

**Выводы**

В данной лабораторной работе я познакомился с двумя способами межпроцессорного взаимодействия: семафоры и разделяемой памятью. Мой способ решения задачи является альтернативным тому, который использовался во второй ЛР (**pipe**). У нового подхода есть свои преимущества:

* Сдвиг указателя в памяти для файла является менее затратным, чем обычное чтение файла, так как во втором случае требуется системный вызов.
* Нет необходимости помнить о расположении файла относительно текущей директории.

Но, есть и свои недостатки. Самый явный из них: надобность использовать семафор, чтобы избежать одновременного доступа в общую память двумя процессами.