Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

**Лабораторная работа №1 по курсу**

**«Операционные системы»**

Группа: М8О-209БВ-24

Студент: Лисов Д.С.

Преподаватель: Миронов Е.С.

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

Дата: 25.09.2025

Москва, 2025

**Постановка задачи**

**Вариант 15.**

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.

**Общий метод и алгоритм решения**

Использованные системные вызовы:

* pid\_t fork(void); – создает дочерний процесс.
* int pipe(int \*fd); – создаёт канал для межпроцессного обмена данными.
* int dup2(int oldfd, int newfd); - дублирование файлового дескриптора
* int fcntl(int fd, int op, ...args); - управление открытыми файлами. В данной работе она делает чтение из pipe2[0] неблокирующим.
* int usleep(useconds\_t usec); - приостановка выполнения на некоторое время. В данном случае использовалась для задания ожидания вывода дочернего процесса.
* pid\_t wait(int \*stat\_loc); - ожидание завершения процесса.
* int fflush(FILE \*stream); - сброс буфера потока вывода.
* int execl(const char \*path, const char \*arg, …); - заменяет текущий образ процесса на другой исполняемый файл.

Сначала программа считывает название файла, который будет использован дочерним процессом для записи. Далее создаются 2 канала (pipe1: родительский процесс → дочерний процесс, pipe2: дочерний процесс → родительский процесс), затем с помощью команды fork() создаётся дочерний процесс. Закрыв неиспользуемые каналы, происходит замена процесса на исполняемый файл ./child с помощью execl.   
 Пользователь вводит строки для проверки, которые родительский процесс считывает из stdin с помощью getline. Программа проверяет, есть ли что-либо в канале pipe2 (ошибки от дочернего процесса). Если они есть, то родительский процесс выводит их в stdout. После этого он пересылает через pipe1 строки дочернему процессу. Дочерний процесс принимает их и обрабатывает (или выводит в нужный файл или пишет об ошибке в stderr, который перенаправлен через pipe2 в parent.c). По окончании вывода закрывается канал pipe1[1] (сигнал для завершения дочернего процесса) и родитель дожидается завершения дочернего процесса через wait(NULL).

**Код программы**

**parent.c**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <sys/wait.h>

#include <fcntl.h>

#include <ctype.h>

#define BUFFER\_SIZE 1024

int main() {

char filename[BUFFER\_SIZE];

int pipe1[2], pipe2[2];

int status;

if (pipe(pipe1) == -1 || pipe(pipe2) == -1) {

perror("pipe failed");

exit(1);

}

if (fgets(filename, sizeof(filename), stdin) == NULL) {

perror("fgets failed");

exit(1);

}

filename[strcspn(filename, "\n")] = '\0';

int file\_fd = open(filename, O\_WRONLY | O\_CREAT | O\_TRUNC, 0644);

if (file\_fd == -1) {

perror("opening file ERROR");

return EXIT\_FAILURE;

}

char \*buffer = NULL;

// dup2(file\_fd, STDOUT\_FILENO);

pid\_t pid = fork();

if (pid == -1) {

perror("fork failed");

exit(1);

}

if (pid == 0) {

close(pipe1[1]);

close(pipe2[0]);

dup2(pipe1[0], STDIN\_FILENO);

dup2(pipe2[1], STDERR\_FILENO);

execl("./child", "child", NULL);

close(pipe1[0]);

close(pipe2[1]);

} else {

close(pipe1[0]);

close(pipe2[1]);

size\_t len = 0;

write(pipe1[1], filename, strlen(filename));

write(pipe1[1], "\n", strlen("\n"));

char error\_msg;

while (1) {

usleep(10000);

fcntl(pipe2[0], F\_SETFL, O\_NONBLOCK);

while (read(pipe2[0], &error\_msg, 1) > 0)

printf("%c", error\_msg);

fflush(stdout);

if (getline(&buffer, &len, stdin) == -1)

break;

write(pipe1[1], buffer, strlen(buffer));

}

close(pipe2[0]);

close(pipe1[1]);

free(buffer);

wait(NULL);

}

close(file\_fd);

return 0;

}

**child.c**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <sys/wait.h>

#include <fcntl.h>

#include <ctype.h>

#define BUFFER\_SIZE 1024

int main() {

char \*buffer;

size\_t n;

ssize\_t r;

char \*filename;

getline(&filename, &n, stdin);

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

//fprintf(f,"%s", filename);

//fflush(stdout);

int file\_fd = open(filename, O\_WRONLY | O\_CREAT | O\_TRUNC, 0644);

dup2(file\_fd, STDOUT\_FILENO);

if (file\_fd == -1)

{

perror("opening file ERROR");

return EXIT\_FAILURE;

}

while ((r = getline(&buffer, &n, stdin)) != -1) {

if (r > 0 && buffer[r - 1] == '\n') {

buffer[r - 1] = '\0';

r--;

}

if (r > 0 && isupper(buffer[0])) {

printf("%s\n", buffer);

} else if (r > 0) {

fprintf(stderr, "Error: %s is not valid\n", buffer);

} else {

fprintf( stderr, "Error: empty string\n");

}

}

fflush(stdout);

free(buffer);

close(file\_fd);

return 0;

}

**Протокол работы программы**

**Тестирование:**

$ ./parent

test.txt

A

B

G

H

hello

Error: hello is not valid

Error: empty string

B

AA

$ cat test.txt

A

B

G

H

B

AA

**Strace:**

$ strace -f ./main

execve("./main", ["./main"], 0x7ffde1b8ad38 /\* 49 vars \*/) = 0

brk(NULL) = 0x5643edd4d000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffed25bee60) = -1 EINVAL (Недопустимый аргумент)

access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (Нет такого файла или каталога)

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

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=73833, ...}) = 0

mmap(NULL, 73833, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7fb731768000

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

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fb731766000

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) = 0x7fb731574000

mprotect(0x7fb731599000, 1847296, PROT\_NONE) = 0

mmap(0x7fb731599000, 1540096, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x25000) = 0x7fb731599000

mmap(0x7fb731711000, 303104, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x19d000) = 0x7fb731711000

mmap(0x7fb73175c000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1e7000) = 0x7fb73175c000

mmap(0x7fb731762000, 13528, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fb731762000

close(3) = 0

arch\_prctl(ARCH\_SET\_FS, 0x7fb731767540) = 0

mprotect(0x7fb73175c000, 12288, PROT\_READ) = 0

mprotect(0x5643edb3c000, 4096, PROT\_READ) = 0

mprotect(0x7fb7317a8000, 4096, PROT\_READ) = 0

munmap(0x7fb731768000, 73833) = 0

brk(NULL) = 0x5643edd4d000

brk(0x5643edd6e000) = 0x5643edd6e000

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

read(0, file1.txt

"file1.txt\n", 1024) = 10

openat(AT\_FDCWD, "file1.txt", O\_WRONLY|O\_CREAT|O\_TRUNC, 0666) = 3

read(0, file2.txt

"file2.txt\n", 1024) = 10

openat(AT\_FDCWD, "file2.txt", O\_WRONLY|O\_CREAT|O\_TRUNC, 0666) = 4

**pipe([5, 6]) = 0**

**pipe([7, 8]) = 0**

**clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLD, child\_tidptr=0x7fb731767810) = 4728**

**clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLD, child\_tidptr=0x7fb731767810) = 4729**

close(3) = 0

close(4) = 0

close(5) = 0

close(7) = 0

read(0, strace: Process 4728 attached

<unfinished ...>

[pid 4728] close(4) = 0

[pid 4728] close(7) = 0

[pid 4728] close(8) = 0

[pid 4728] close(6) = 0

[pid 4728] dup2(5, 0) = 0

[pid 4728] dup2(3, 1) = 1

[pid 4728] close(3strace: Process 4729 attached

) = 0

[pid 4728] execve("child", NULL, 0x7ffed25bef48 /\* 49 vars \*/ <unfinished ...>

[pid 4729] close(3) = 0

[pid 4729] close(5) = 0

[pid 4729] close(6) = 0

[pid 4729] close(8) = 0

[pid 4729] dup2(7, 0) = 0

[pid 4729] dup2(4, 1) = 1

[pid 4729] close(4) = 0

[pid 4729] execve("child", NULL, 0x7ffed25bef48 /\* 49 vars \*/ <unfinished ...>

[pid 4728] <... execve resumed>) = 0

[pid 4728] brk(NULL <unfinished ...>

[pid 4729] <... execve resumed>) = 0

[pid 4729] brk(NULL <unfinished ...>

[pid 4728] <... brk resumed>) = 0x55b32f123000

[pid 4728] arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffc0937eb80 <unfinished ...>

[pid 4729] <... brk resumed>) = 0x55884ee36000

[pid 4729] arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffd061df910) = -1 EINVAL (Недопустимый аргумент)

[pid 4728] <... arch\_prctl resumed>) = -1 EINVAL (Недопустимый аргумент)

[pid 4729] access("/etc/ld.so.preload", R\_OK <unfinished ...>

[pid 4728] access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (Нет такого файла или каталога)

[pid 4729] <... access resumed>) = -1 ENOENT (Нет такого файла или каталога)

[pid 4728] openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC <unfinished ...>

[pid 4729] openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3

[pid 4728] <... openat resumed>) = 3

[pid 4728] fstat(3, <unfinished ...>

[pid 4729] fstat(3, <unfinished ...>

[pid 4728] <... fstat resumed>{st\_mode=S\_IFREG|0644, st\_size=73833, ...}) = 0

[pid 4729] <... fstat resumed>{st\_mode=S\_IFREG|0644, st\_size=73833, ...}) = 0

[pid 4729] mmap(NULL, 73833, PROT\_READ, MAP\_PRIVATE, 3, 0 <unfinished ...>

[pid 4728] mmap(NULL, 73833, PROT\_READ, MAP\_PRIVATE, 3, 0 <unfinished ...>

[pid 4729] <... mmap resumed>) = 0x7f9f03ba1000

[pid 4728] <... mmap resumed>) = 0x7f8c0c66c000

[pid 4728] close(3 <unfinished ...>

[pid 4729] close(3 <unfinished ...>

[pid 4728] <... close resumed>) = 0

[pid 4729] <... close resumed>) = 0

[pid 4728] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC <unfinished ...>

[pid 4729] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC) = 3

[pid 4728] <... openat resumed>) = 3

[pid 4729] read(3, <unfinished ...>

[pid 4728] read(3, <unfinished ...>

[pid 4729] <... read resumed>"\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 4728] <... read resumed>"\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 4729] pread64(3, <unfinished ...>

[pid 4728] pread64(3, <unfinished ...>

[pid 4729] <... pread64 resumed>"\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 4728] <... pread64 resumed>"\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 4729] pread64(3, <unfinished ...>

[pid 4728] pread64(3, <unfinished ...>

[pid 4729] <... pread64 resumed>"\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 4728] <... pread64 resumed>"\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 4729] pread64(3, <unfinished ...>

[pid 4728] pread64(3, <unfinished ...>

[pid 4729] <... pread64 resumed>"\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 4728] <... pread64 resumed>"\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 4729] fstat(3, {st\_mode=S\_IFREG|0755, st\_size=2029224, ...}) = 0

[pid 4729] mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f9f03b9f000

[pid 4729] 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 4729] 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 4729] 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 4729] mmap(NULL, 2036952, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f9f039ad000

[pid 4729] mprotect(0x7f9f039d2000, 1847296, PROT\_NONE) = 0

[pid 4729] mmap(0x7f9f039d2000, 1540096, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x25000) = 0x7f9f039d2000

[pid 4729] mmap(0x7f9f03b4a000, 303104, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x19d000) = 0x7f9f03b4a000

[pid 4729] mmap(0x7f9f03b95000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1e7000 <unfinished ...>

[pid 4728] fstat(3, <unfinished ...>

[pid 4729] <... mmap resumed>) = 0x7f9f03b95000

[pid 4728] <... fstat resumed>{st\_mode=S\_IFREG|0755, st\_size=2029224, ...}) = 0

[pid 4728] mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 4729] mmap(0x7f9f03b9b000, 13528, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 4728] <... mmap resumed>) = 0x7f8c0c66a000

[pid 4728] pread64(3, <unfinished ...>

[pid 4729] <... mmap resumed>) = 0x7f9f03b9b000

[pid 4729] close(3 <unfinished ...>

[pid 4728] <... pread64 resumed>"\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 4728] pread64(3, <unfinished ...>

[pid 4729] <... close resumed>) = 0

[pid 4728] <... pread64 resumed>"\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 4729] arch\_prctl(ARCH\_SET\_FS, 0x7f9f03ba0540 <unfinished ...>

[pid 4728] 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 4728] mmap(NULL, 2036952, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f8c0c478000

[pid 4728] mprotect(0x7f8c0c49d000, 1847296, PROT\_NONE) = 0

[pid 4729] <... arch\_prctl resumed>) = 0

[pid 4729] mprotect(0x7f9f03b95000, 12288, PROT\_READ <unfinished ...>

[pid 4728] mmap(0x7f8c0c49d000, 1540096, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x25000) = 0x7f8c0c49d000

[pid 4728] mmap(0x7f8c0c615000, 303104, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x19d000) = 0x7f8c0c615000

[pid 4728] mmap(0x7f8c0c660000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1e7000) = 0x7f8c0c660000

[pid 4728] mmap(0x7f8c0c666000, 13528, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f8c0c666000

[pid 4728] close(3) = 0

[pid 4728] arch\_prctl(ARCH\_SET\_FS, 0x7f8c0c66b540 <unfinished ...>

[pid 4729] <... mprotect resumed>) = 0

[pid 4728] <... arch\_prctl resumed>) = 0

[pid 4728] mprotect(0x7f8c0c660000, 12288, PROT\_READ <unfinished ...>

[pid 4729] mprotect(0x55884dfa5000, 4096, PROT\_READ) = 0

[pid 4729] mprotect(0x7f9f03be1000, 4096, PROT\_READ) = 0

[pid 4729] munmap(0x7f9f03ba1000, 73833) = 0

[pid 4728] <... mprotect resumed>) = 0

[pid 4728] mprotect(0x55b32d937000, 4096, PROT\_READ <unfinished ...>

[pid 4729] read(0, <unfinished ...>

[pid 4728] <... mprotect resumed>) = 0

[pid 4728] mprotect(0x7f8c0c6ac000, 4096, PROT\_READ) = 0

[pid 4728] munmap(0x7f8c0c66c000, 73833) = 0

[pid 4728] read(0, 123456789123456789123456789

<unfinished ...>

[pid 4726] <... read resumed>"1", 1) = 1

[pid 4726] read(0, "2", 1) = 1

[pid 4726] read(0, "3", 1) = 1

[pid 4726] read(0, "4", 1) = 1

[pid 4726] read(0, "5", 1) = 1

[pid 4726] read(0, "6", 1) = 1

[pid 4726] read(0, "7", 1) = 1

[pid 4726] read(0, "8", 1) = 1

[pid 4726] read(0, "9", 1) = 1

[pid 4726] read(0, "1", 1) = 1

[pid 4726] read(0, "2", 1) = 1

[pid 4726] read(0, "3", 1) = 1

[pid 4726] read(0, "4", 1) = 1

[pid 4726] read(0, "5", 1) = 1

[pid 4726] read(0, "6", 1) = 1

[pid 4726] read(0, "7", 1) = 1

[pid 4726] read(0, "8", 1) = 1

[pid 4726] read(0, "9", 1) = 1

[pid 4726] read(0, "1", 1) = 1

[pid 4726] read(0, "2", 1) = 1

[pid 4726] read(0, "3", 1) = 1

[pid 4726] read(0, "4", 1) = 1

[pid 4726] read(0, "5", 1) = 1

[pid 4726] read(0, "6", 1) = 1

[pid 4726] read(0, "7", 1) = 1

[pid 4726] read(0, "8", 1) = 1

[pid 4726] read(0, "9", 1) = 1

[pid 4726] read(0, "\n", 1) = 1

[pid 4726] write(8, "\33\0\0\0", 4) = 4

[pid 4726] write(8, "123456789123456789123456789", 27) = 27

[pid 4726] read(0, <unfinished ...>

[pid 4729] <... read resumed>"\33\0\0\0", 4) = 4

[pid 4729] read(0, "123456789123456789123456789", 27) = 27

[pid 4729] fstat(1, {st\_mode=S\_IFREG|0664, st\_size=0, ...}) = 0

[pid 4729] brk(NULL) = 0x55884ee36000

[pid 4729] brk(0x55884ee57000) = 0x55884ee57000

[pid 4729] write(1, "987654321987654321987654321\n", 28) = 28

[pid 4729] read(0, okay

<unfinished ...>

[pid 4726] <... read resumed>"o", 1) = 1

[pid 4726] read(0, "k", 1) = 1

[pid 4726] read(0, "a", 1) = 1

[pid 4726] read(0, "y", 1) = 1

[pid 4726] read(0, "\n", 1) = 1

[pid 4726] write(8, "\4\0\0\0", 4) = 4

[pid 4726] write(8, "okay", 4) = 4

[pid 4726] read(0, <unfinished ...>

[pid 4729] <... read resumed>"\4\0\0\0", 4) = 4

[pid 4729] read(0, "okay", 4) = 4

[pid 4729] write(1, "yako\n", 5) = 5

[pid 4729] read(0, fedor

<unfinished ...>

[pid 4726] <... read resumed>"f", 1) = 1

[pid 4726] read(0, "e", 1) = 1

[pid 4726] read(0, "d", 1) = 1

[pid 4726] read(0, "o", 1) = 1

[pid 4726] read(0, "r", 1) = 1

[pid 4726] read(0, "\n", 1) = 1

[pid 4726] write(6, "\5\0\0\0", 4) = 4

[pid 4726] write(6, "fedor", 5) = 5

[pid 4726] read(0, <unfinished ...>

[pid 4728] <... read resumed>"\5\0\0\0", 4) = 4

[pid 4728] read(0, "fedor", 5) = 5

[pid 4728] fstat(1, {st\_mode=S\_IFREG|0664, st\_size=0, ...}) = 0

[pid 4728] brk(NULL) = 0x55b32f123000

[pid 4728] brk(0x55b32f144000) = 0x55b32f144000

[pid 4728] write(1, "rodef\n", 6) = 6

[pid 4728] read(0, rodeo

<unfinished ...>

[pid 4726] <... read resumed>"r", 1) = 1

[pid 4726] read(0, "o", 1) = 1

[pid 4726] read(0, "d", 1) = 1

[pid 4726] read(0, "e", 1) = 1

[pid 4726] read(0, "o", 1) = 1

[pid 4726] read(0, "\n", 1) = 1

[pid 4726] write(6, "\5\0\0\0", 4) = 4

[pid 4726] write(6, "rodeo", 5) = 5

[pid 4726] read(0, <unfinished ...>

[pid 4728] <... read resumed>"\5\0\0\0", 4) = 4

[pid 4728] read(0, "rodeo", 5) = 5

[pid 4728] write(1, "oedor\n", 6) = 6

[pid 4728] read(0, hihihi

<unfinished ...>

[pid 4726] <... read resumed>"h", 1) = 1

[pid 4726] read(0, "i", 1) = 1

[pid 4726] read(0, "h", 1) = 1

[pid 4726] read(0, "i", 1) = 1

[pid 4726] read(0, "h", 1) = 1

[pid 4726] read(0, "i", 1) = 1

[pid 4726] read(0, "\n", 1) = 1

[pid 4726] write(8, "\6\0\0\0", 4) = 4

[pid 4726] write(8, "hihihi", 6) = 6

[pid 4726] read(0, <unfinished ...>

[pid 4729] <... read resumed>"\6\0\0\0", 4) = 4

[pid 4729] read(0, "hihihi", 6) = 6

[pid 4729] write(1, "ihihih\n", 7) = 7

[pid 4729] read(0, <unfinished ...>

[pid 4726] <... read resumed>"", 1) = 0

[pid 4726] close(6) = 0

[pid 4726] close(8) = 0

[pid 4726] exit\_group(0) = ?

[pid 4726] +++ exited with 0 +++

[pid 4728] <... read resumed>"", 4) = 0

[pid 4728] exit\_group(0) = ?

[pid 4728] +++ exited with 0 +++

<... read resumed>"", 4) = 0

exit\_group(0) = ?

+++ exited with 0 +++

**Вывод**

Лабораторная работа показалась достаточно трудной для выполнения. Сталкивался с проблемами, когда 2 процесса пытались получить доступ к одним и тем же данным. Также возникали проблемы, когда один процесс ожидал другой и блокировал выполнение другого. Научился работать с процессами, межпроцессным взаимодействием и системными вызовами.