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: М80-206Б-20

Студент: Кочев Д.О.

Преподаватель: Миронов Е.С.

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

Дата: 17.11.2023

Москва, 2023

**Постановка задачи**

**Группа вариантов 5.**

Родительский процесс создает два дочерних процесса. Первой строкой пользователь в консоль родительского процесса вводит имя файла, которое будет использовано для открытия File с таким именем на запись для child1. Аналогично для второй строки и процесса child2. Родительский и дочерний процесс должны быть представлены разными программами. Родительский процесс принимает от пользователя строки произвольной длины и пересылает их в pipe1 или в pipe2 в зависимости от правила фильтрации. Процесс child1 и child2 производят работу над строками. Процессы пишут результаты своей работы в стандартный вывод.

**Вариант 18.**

Правило фильтрации: нечетные строки отправляются в pipe1, четные в pipe2. Дочерние процессы удаляют все гласные из строк.

**Общий метод и алгоритм решения**

Использованные системные вызовы:

* pid\_t fork(void); – создает дочерний процесс.
* int pipe(int \*fd); – Функция pipe создает канал (неименованный) и помещает дескрипторы файла для чтения и записи (соответственно) в fd[0] и fd[1].
* int execl(char \*name, char \*arg0, ... /\*NULL\*/) – загружает и запускает указанную программу. Таким образом, новая программа полностью замещает текущий процесс.
* int dup2(int oldfd, int newfd) – делает newfd копией oldfd, закрывая newfd, если требуется.
* int close(int fd) - закрывает файловый дескриптор.
* ssize\_t read (int fd, void \* buffer, size\_t count) – считывает count байт из файлового дескриптора fd в buffer. Возвращает количество прочитанных байт или -1, если ошибка.
* ssize\_t write (int fd, const void \* buffer, size\_t count) – записывает count байтов из буфера buffer в файл с дескриптором fd, возвращая количество записанных байтов или -1 в случае ошибки.
* pid\_t wait(int \*status) – приостанавливает выполнение текущего процесса до тех пор, пока дочерний процесс не завершится.

После запуска программы пользователю нужно ввести в командную строку имя первого файла, затем на следующей строке имя второго файла. После этого функция open открывает файл с данным названием и очищает его. Если данного файла не было, то создаст его. Если все было введено корректно и два файла доступны для работы, то создаются два безымянных канала pipe. Далее будет создан первый дочерний процесс. В нем мы заменяем стандартный поток ввода на “чтение” из pipe (fd[READ]), а стандартный поток вывода на запись в файл с помощью функции dup2. Затем первый дочерний процесс запускает программу child.c, и программа main для этого процесса завершается. Аналогичные действия мы проделываем с вторым дочерним процессом. Далее в родительском процессе мы считываем все символы, которые вводит пользователь. Сначала мы их направляем в дескриптор первого pipe, откуда первый дочерний процесс считывает введенные данные, обрабатывает их и записывает в стандартный вывод, который подменен на вывод в файл в данном процессе. Когда пользователь введет \n, мы начинаем перенаправлять все символы в дескриптор второго pipe, откуда второй дочерний процесс так же читает данные, а потом выводит во второй файл. Программа завершает работу, когда встретит символ EOF.

**Код программы**

**laba1.c**

#include <stdio.h>

#include <unistd.h>

#include <stdlib.h>

#include <string.h>

#include <sys/types.h>

#include <sys/wait.h>

#include <fcntl.h>

#include <string.h>

#include <stdbool.h>

char\* get\_fileaname() {

int len = 0;

int capacity = 1;

char \*s = (char\*) malloc(sizeof(char));

char c = getchar();

while (c != '\n') {

s[(len)++] = c;

if (len >= capacity) {

capacity \*= 2;

s = (char\*) realloc(s, capacity \* sizeof(char));

}

if (capacity > 256) {

s = NULL;

return s;

}

c = getchar();

}

s[len] = '\0';

return s;

}

int main (){

enum {

READ = 0,

WRITE = 1

};

char \* first\_file = NULL;

first\_file = get\_fileaname();

if (first\_file == NULL) {

perror ("Large file name or no memory \n");

return -1;

}

int out = open(first\_file,O\_WRONLY| O\_CREAT | O\_TRUNC , 0666);

if (out == -1) {

perror ("There is no such file \n");

return -1;

}

char \* second\_file = NULL;

second\_file = get\_fileaname();

if (second\_file == NULL) {

perror ("Large file name or no memory\n");

return -1;

}

int out2 = open(second\_file,O\_WRONLY| O\_CREAT | O\_TRUNC , 0666);

if (out2 == -1) {

perror ("There is no such file \n");

return -1;

}

// printf("%d %d\n", out, out2);

int fd[2];

// printf("%d %d\n", fd[0], fd[1]);

if(pipe2(fd, O\_CLOEXEC) == -1) {

perror("pipe");

return -1;

}

// printf("%d %d\n", fd[0], fd[1]);

int fd2[2];

// printf("%d %d\n", fd2[0], fd2[1]);

if(pipe2(fd2, O\_CLOEXEC) == -1) {

perror("pipe");

return -1;

}

// printf("%d %d\n", fd2[0], fd[1]);

pid\_t id = fork();

if (id == 0){

close(fd[WRITE]);

dup2(fd[READ],fileno(stdin));

dup2(out,fileno(stdout));

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

perror("execl");

}

pid\_t id2 = fork();

if (id2 == 0){

close(fd2[WRITE]);

dup2(fd2[READ],fileno(stdin));

dup2(out2,fileno(stdout));

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

perror("execl");

}

if (id > 0) {

int c;

int flag = 0;

while ((c = getchar()) != EOF) {

if (flag % 2 == 0){

write(fd[WRITE], &c, sizeof(int));

} else {

write(fd2[WRITE], &c, sizeof(int));

}

if (c == '\n'){

flag ++;

}

}

close(fd2[WRITE]);

close(fd[WRITE]);

wait(NULL);

}

}

**child.c**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/wait.h>

int main (){

int a1, a2;

int c, u;

while(read(fileno(stdin), &c, sizeof(int)) != 0) {

if ((c != 'a') && (c != 'e') && (c != 'i') && (c != 'u') && (c != 'y') && (c != 'o') &&

(c != 'A') && (c != 'E') && (c != 'I') && (c != 'U') && (c != 'Y') && (c != 'O')) {

u = (char) c;

write(fileno(stdout), &c, sizeof(char));

}

}

close(fileno(stdin));

}

**Протокол работы программы**

**Тестирование:**

./parent

aaaa

bbbb

aaaaab

bbbbba

qwerty

qwertyu

ry

qwasdf

$ cat < aaaa

b

qwrt

r

qwsdf

$ cat < bbbb

bbbbb

qwrt

**Strace:**

$ strace -f ./parent

execve("./parent", ["./parent"], 0x7ffe0d0f10b8 /\* 34 vars \*/) = 0

brk(NULL) = 0x55ab32794000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7fffd7e76fa0) = -1 EINVAL (Invalid argument)

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fef25707000

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=17231, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 17231, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7fef25702000

close(3) = 0

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

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0\0"..., 832) = 832

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48

pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\244;\374\204(\337f#\315I\214\234\f\256\271\32"..., 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) = 0x7fef254da000

mmap(0x7fef25502000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7fef25502000

mmap(0x7fef25697000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1bd000) = 0x7fef25697000

mmap(0x7fef256ef000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x214000) = 0x7fef256ef000

mmap(0x7fef256f5000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fef256f5000

close(3) = 0

mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fef254d7000

arch\_prctl(ARCH\_SET\_FS, 0x7fef254d7740) = 0

set\_tid\_address(0x7fef254d7a10) = 11732

set\_robust\_list(0x7fef254d7a20, 24) = 0

rseq(0x7fef254d80e0, 0x20, 0, 0x53053053) = 0

mprotect(0x7fef256ef000, 16384, PROT\_READ) = 0

mprotect(0x55ab314c4000, 4096, PROT\_READ) = 0

mprotect(0x7fef25741000, 8192, PROT\_READ) = 0

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

munmap(0x7fef25702000, 17231) = 0

getrandom("\xa7\x27\xaa\x28\xeb\x1d\x61\x32", 8, GRND\_NONBLOCK) = 8

brk(NULL) = 0x55ab32794000

brk(0x55ab327b5000) = 0x55ab327b5000

newfstatat(0, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x4), ...}, AT\_EMPTY\_PATH) = 0

read(0, aaaa

"aaaa\n", 1024) = 5

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

read(0, bbbb

"bbbb\n", 1024) = 5

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

pipe2([5, 6], O\_CLOEXEC) = 0

pipe2([7, 8], O\_CLOEXEC) = 0

clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLDstrace: Process 11772 attached

, child\_tidptr=0x7fef254d7a10) = 11772

[pid 11772] set\_robust\_list(0x7fef254d7a20, 24 <unfinished ...>

[pid 11732] clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLD <unfinished ...>

[pid 11772] <... set\_robust\_list resumed>) = 0

strace: Process 11773 attached

[pid 11772] close(6 <unfinished ...>

[pid 11732] <... clone resumed>, child\_tidptr=0x7fef254d7a10) = 11773

[pid 11773] set\_robust\_list(0x7fef254d7a20, 24 <unfinished ...>

[pid 11772] <... close resumed>) = 0

[pid 11732] read(0, <unfinished ...>

[pid 11773] <... set\_robust\_list resumed>) = 0

[pid 11772] dup2(5, 0 <unfinished ...>

[pid 11773] close(8 <unfinished ...>

[pid 11772] <... dup2 resumed>) = 0

[pid 11773] <... close resumed>) = 0

[pid 11772] dup2(3, 1 <unfinished ...>

[pid 11773] dup2(7, 0 <unfinished ...>

[pid 11772] <... dup2 resumed>) = 1

[pid 11773] <... dup2 resumed>) = 0

[pid 11772] execve("./child", ["./child"], 0x7fffd7e77178 /\* 34 vars \*/ <unfinished ...>

[pid 11773] dup2(4, 1) = 1

[pid 11773] execve("./child", ["./child"], 0x7fffd7e77178 /\* 34 vars \*/ <unfinished ...>

[pid 11772] <... execve resumed>) = 0

[pid 11772] brk(NULL) = 0x561fc8e81000

[pid 11772] arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffed19239b0) = -1 EINVAL (Invalid argument)

[pid 11772] mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fd5bd676000

[pid 11772] access("/etc/ld.so.preload", R\_OK <unfinished ...>

[pid 11773] <... execve resumed>) = 0

[pid 11772] <... access resumed>) = -1 ENOENT (No such file or directory)

[pid 11773] brk(NULL <unfinished ...>

[pid 11772] openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC <unfinished ...>

[pid 11773] <... brk resumed>) = 0x55c13c608000

[pid 11772] <... openat resumed>) = 5

[pid 11773] arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffda0542730 <unfinished ...>

[pid 11772] newfstatat(5, "", <unfinished ...>

[pid 11773] <... arch\_prctl resumed>) = -1 EINVAL (Invalid argument)

[pid 11772] <... newfstatat resumed>{st\_mode=S\_IFREG|0644, st\_size=17231, ...}, AT\_EMPTY\_PATH) = 0

[pid 11773] mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f64a1533000

[pid 11772] mmap(NULL, 17231, PROT\_READ, MAP\_PRIVATE, 5, 0 <unfinished ...>

[pid 11773] access("/etc/ld.so.preload", R\_OK <unfinished ...>

[pid 11772] <... mmap resumed>) = 0x7fd5bd671000

[pid 11773] <... access resumed>) = -1 ENOENT (No such file or directory)

[pid 11772] close(5 <unfinished ...>

[pid 11773] openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC <unfinished ...>

[pid 11772] <... close resumed>) = 0

[pid 11773] <... openat resumed>) = 5

[pid 11772] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC <unfinished ...>

[pid 11773] newfstatat(5, "", <unfinished ...>

[pid 11772] <... openat resumed>) = 5

[pid 11772] read(5, <unfinished ...>

[pid 11773] <... newfstatat resumed>{st\_mode=S\_IFREG|0644, st\_size=17231, ...}, AT\_EMPTY\_PATH) = 0

[pid 11772] <... read resumed>"\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 11773] mmap(NULL, 17231, PROT\_READ, MAP\_PRIVATE, 5, 0) = 0x7f64a152e000

[pid 11772] pread64(5, <unfinished ...>

[pid 11773] close(5 <unfinished ...>

[pid 11772] <... 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 11773] <... close resumed>) = 0

[pid 11772] pread64(5, <unfinished ...>

[pid 11773] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC <unfinished ...>

[pid 11772] <... pread64 resumed>"\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48

[pid 11773] <... openat resumed>) = 5

[pid 11772] pread64(5, <unfinished ...>

[pid 11773] read(5, <unfinished ...>

[pid 11772] <... pread64 resumed>"\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\244;\374\204(\337f#\315I\214\234\f\256\271\32"..., 68, 896) = 68

[pid 11773] <... read resumed>"\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 11772] newfstatat(5, "", <unfinished ...>

[pid 11773] pread64(5, <unfinished ...>

[pid 11772] <... newfstatat resumed>{st\_mode=S\_IFREG|0755, st\_size=2216304, ...}, AT\_EMPTY\_PATH) = 0

[pid 11773] <... 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 11772] pread64(5, <unfinished ...>

[pid 11773] pread64(5, <unfinished ...>

[pid 11772] <... 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 11773] <... pread64 resumed>"\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48

[pid 11772] mmap(NULL, 2260560, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 5, 0 <unfinished ...>

[pid 11773] pread64(5, <unfinished ...>

[pid 11772] <... mmap resumed>) = 0x7fd5bd449000

[pid 11773] <... pread64 resumed>"\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\244;\374\204(\337f#\315I\214\234\f\256\271\32"..., 68, 896) = 68

[pid 11772] mmap(0x7fd5bd471000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 5, 0x28000 <unfinished ...>

[pid 11773] newfstatat(5, "", <unfinished ...>

[pid 11772] <... mmap resumed>) = 0x7fd5bd471000

[pid 11773] <... newfstatat resumed>{st\_mode=S\_IFREG|0755, st\_size=2216304, ...}, AT\_EMPTY\_PATH) = 0

[pid 11772] mmap(0x7fd5bd606000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 5, 0x1bd000 <unfinished ...>

[pid 11773] pread64(5, <unfinished ...>

[pid 11772] <... mmap resumed>) = 0x7fd5bd606000

[pid 11773] <... 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 11772] mmap(0x7fd5bd65e000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 5, 0x214000 <unfinished ...>

[pid 11773] mmap(NULL, 2260560, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 5, 0 <unfinished ...>

[pid 11772] <... mmap resumed>) = 0x7fd5bd65e000

[pid 11773] <... mmap resumed>) = 0x7f64a1306000

[pid 11772] mmap(0x7fd5bd664000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 11773] mmap(0x7f64a132e000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 5, 0x28000 <unfinished ...>

[pid 11772] <... mmap resumed>) = 0x7fd5bd664000

[pid 11773] <... mmap resumed>) = 0x7f64a132e000

[pid 11772] close(5 <unfinished ...>

[pid 11773] mmap(0x7f64a14c3000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 5, 0x1bd000 <unfinished ...>

[pid 11772] <... close resumed>) = 0

[pid 11773] <... mmap resumed>) = 0x7f64a14c3000

[pid 11772] mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 11773] mmap(0x7f64a151b000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 5, 0x214000 <unfinished ...>

[pid 11772] <... mmap resumed>) = 0x7fd5bd446000

[pid 11773] <... mmap resumed>) = 0x7f64a151b000

[pid 11772] arch\_prctl(ARCH\_SET\_FS, 0x7fd5bd446740 <unfinished ...>

[pid 11773] mmap(0x7f64a1521000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 11772] <... arch\_prctl resumed>) = 0

[pid 11773] <... mmap resumed>) = 0x7f64a1521000

[pid 11772] set\_tid\_address(0x7fd5bd446a10 <unfinished ...>

[pid 11773] close(5 <unfinished ...>

[pid 11772] <... set\_tid\_address resumed>) = 11772

[pid 11773] <... close resumed>) = 0

[pid 11772] set\_robust\_list(0x7fd5bd446a20, 24 <unfinished ...>

[pid 11773] mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 11772] <... set\_robust\_list resumed>) = 0

[pid 11773] <... mmap resumed>) = 0x7f64a1303000

[pid 11772] rseq(0x7fd5bd4470e0, 0x20, 0, 0x53053053 <unfinished ...>

[pid 11773] arch\_prctl(ARCH\_SET\_FS, 0x7f64a1303740 <unfinished ...>

[pid 11772] <... rseq resumed>) = 0

[pid 11773] <... arch\_prctl resumed>) = 0

[pid 11773] set\_tid\_address(0x7f64a1303a10 <unfinished ...>

[pid 11772] mprotect(0x7fd5bd65e000, 16384, PROT\_READ <unfinished ...>

[pid 11773] <... set\_tid\_address resumed>) = 11773

[pid 11772] <... mprotect resumed>) = 0

[pid 11773] set\_robust\_list(0x7f64a1303a20, 24 <unfinished ...>

[pid 11772] mprotect(0x561fc7a9e000, 4096, PROT\_READ <unfinished ...>

[pid 11773] <... set\_robust\_list resumed>) = 0

[pid 11772] <... mprotect resumed>) = 0

[pid 11773] rseq(0x7f64a13040e0, 0x20, 0, 0x53053053 <unfinished ...>

[pid 11772] mprotect(0x7fd5bd6b0000, 8192, PROT\_READ <unfinished ...>

[pid 11773] <... rseq resumed>) = 0

[pid 11772] <... mprotect resumed>) = 0

[pid 11773] mprotect(0x7f64a151b000, 16384, PROT\_READ <unfinished ...>

[pid 11772] prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

[pid 11773] <... mprotect resumed>) = 0

[pid 11773] mprotect(0x55c13c055000, 4096, PROT\_READ <unfinished ...>

[pid 11772] munmap(0x7fd5bd671000, 17231 <unfinished ...>

[pid 11773] <... mprotect resumed>) = 0

[pid 11772] <... munmap resumed>) = 0

[pid 11773] mprotect(0x7f64a156d000, 8192, PROT\_READ <unfinished ...>

[pid 11772] read(0, <unfinished ...>

[pid 11773] <... mprotect resumed>) = 0

[pid 11773] prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

[pid 11773] munmap(0x7f64a152e000, 17231) = 0

[pid 11773] read(0, qwerty

<unfinished ...>

[pid 11732] <... read resumed>"qwerty\n", 1024) = 7

[pid 11732] write(6, "q\0\0\0", 4) = 4

[pid 11772] <... read resumed>"q\0\0\0", 4) = 4

[pid 11732] write(6, "w\0\0\0", 4 <unfinished ...>

[pid 11772] write(1, "q", 1 <unfinished ...>

[pid 11732] <... write resumed>) = 4

[pid 11732] write(6, "e\0\0\0", 4) = 4

[pid 11732] write(6, "r\0\0\0", 4) = 4

[pid 11732] write(6, "t\0\0\0", 4) = 4

[pid 11732] write(6, "y\0\0\0", 4) = 4

[pid 11732] write(6, "\n\0\0\0", 4) = 4

[pid 11732] read(0, <unfinished ...>

[pid 11772] <... write resumed>) = 1

[pid 11772] read(0, "w\0\0\0", 4) = 4

[pid 11772] write(1, "w", 1) = 1

[pid 11772] read(0, "e\0\0\0", 4) = 4

[pid 11772] read(0, "r\0\0\0", 4) = 4

[pid 11772] write(1, "r", 1) = 1

[pid 11772] read(0, "t\0\0\0", 4) = 4

[pid 11772] write(1, "t", 1) = 1

[pid 11772] read(0, "y\0\0\0", 4) = 4

[pid 11772] read(0, "\n\0\0\0", 4) = 4

[pid 11772] write(1, "\n", 1) = 1

[pid 11772] read(0, uyytr

<unfinished ...>

[pid 11732] <... read resumed>"uyytr\n", 1024) = 6

[pid 11732] write(8, "u\0\0\0", 4) = 4

[pid 11773] <... read resumed>"u\0\0\0", 4) = 4

[pid 11732] write(8, "y\0\0\0", 4 <unfinished ...>

[pid 11773] read(0, <unfinished ...>

[pid 11732] <... write resumed>) = 4

[pid 11773] <... read resumed>"y\0\0\0", 4) = 4

[pid 11732] write(8, "y\0\0\0", 4 <unfinished ...>

[pid 11773] read(0, <unfinished ...>

[pid 11732] <... write resumed>) = 4

[pid 11773] <... read resumed>"y\0\0\0", 4) = 4

[pid 11732] write(8, "t\0\0\0", 4 <unfinished ...>

[pid 11773] read(0, <unfinished ...>

[pid 11732] <... write resumed>) = 4

[pid 11773] <... read resumed>"t\0\0\0", 4) = 4

[pid 11732] write(8, "r\0\0\0", 4 <unfinished ...>

[pid 11773] write(1, "t", 1 <unfinished ...>

[pid 11732] <... write resumed>) = 4

[pid 11732] write(8, "\n\0\0\0", 4) = 4

[pid 11732] read(0, <unfinished ...>

[pid 11773] <... write resumed>) = 1

[pid 11773] read(0, "r\0\0\0", 4) = 4

[pid 11773] write(1, "r", 1) = 1

[pid 11773] read(0, "\n\0\0\0", 4) = 4

[pid 11773] write(1, "\n", 1) = 1

[pid 11773] read(0, qweasd

<unfinished ...>

[pid 11732] <... read resumed>"qweasd\n", 1024) = 7

[pid 11732] write(6, "q\0\0\0", 4) = 4

[pid 11772] <... read resumed>"q\0\0\0", 4) = 4

[pid 11732] write(6, "w\0\0\0", 4 <unfinished ...>

[pid 11772] write(1, "q", 1 <unfinished ...>

[pid 11732] <... write resumed>) = 4

[pid 11732] write(6, "e\0\0\0", 4) = 4

[pid 11732] write(6, "a\0\0\0", 4) = 4

[pid 11732] write(6, "s\0\0\0", 4) = 4

[pid 11732] write(6, "d\0\0\0", 4) = 4

[pid 11732] write(6, "\n\0\0\0", 4) = 4

[pid 11732] read(0, <unfinished ...>

[pid 11772] <... write resumed>) = 1

[pid 11772] read(0, "w\0\0\0", 4) = 4

[pid 11772] write(1, "w", 1) = 1

[pid 11772] read(0, "e\0\0\0", 4) = 4

[pid 11772] read(0, "a\0\0\0", 4) = 4

[pid 11772] read(0, "s\0\0\0", 4) = 4

[pid 11772] write(1, "s", 1) = 1

[pid 11772] read(0, "d\0\0\0", 4) = 4

[pid 11772] write(1, "d", 1) = 1

[pid 11772] read(0, "\n\0\0\0", 4) = 4

[pid 11772] write(1, "\n", 1) = 1

[pid 11772] read(0, qwe

<unfinished ...>

[pid 11732] <... read resumed>"qwe\n", 1024) = 4

[pid 11732] write(8, "q\0\0\0", 4) = 4

[pid 11773] <... read resumed>"q\0\0\0", 4) = 4

[pid 11732] write(8, "w\0\0\0", 4) = 4

[pid 11773] write(1, "q", 1 <unfinished ...>

[pid 11732] write(8, "e\0\0\0", 4) = 4

[pid 11732] write(8, "\n\0\0\0", 4) = 4

[pid 11732] read(0, <unfinished ...>

[pid 11773] <... write resumed>) = 1

[pid 11773] read(0, "w\0\0\0", 4) = 4

[pid 11773] write(1, "w", 1) = 1

[pid 11773] read(0, "e\0\0\0", 4) = 4

[pid 11773] read(0, "\n\0\0\0", 4) = 4

[pid 11773] write(1, "\n", 1) = 1

[pid 11773] read(0, <unfinished ...>

[pid 11732] <... read resumed>"", 1024) = 0

[pid 11732] close(8) = 0

[pid 11732] close(6 <unfinished ...>

[pid 11773] <... read resumed>"", 4) = 0

[pid 11732] <... close resumed>) = 0

[pid 11773] close(0 <unfinished ...>

[pid 11772] <... read resumed>"", 4) = 0

[pid 11732] wait4(-1, <unfinished ...>

[pid 11773] <... close resumed>) = 0

[pid 11772] close(0 <unfinished ...>

[pid 11773] exit\_group(0 <unfinished ...>

[pid 11772] <... close resumed>) = 0

[pid 11773] <... exit\_group resumed>) = ?

[pid 11772] exit\_group(0) = ?

[pid 11773] +++ exited with 0 +++

[pid 11772] +++ exited with 0 +++

<... wait4 resumed>NULL, 0, NULL) = 11772

--- SIGCHLD {si\_signo=SIGCHLD, si\_code=CLD\_EXITED, si\_pid=11773, si\_uid=1000, si\_status=0, si\_utime=0, si\_stime=1} ---

exit\_group(0) = ?

+++ exited with 0 +++

**Вывод**

В ходе выполнения данной лабораторной работы я столкнулся с перенаправлением ввода и вывода, а также узнал о том, что одну программу можно “параллелить” на две и больше почти идентичные программы. Изначально были сложности с тем, чтобы держать все дочерние и родительский процессы в голове, и понимать, куда нужно перенаправлять их вводы и выводы. В итоге, я неплохо разобрался и осознал данную тему, что поможет мне в будущем справляться с более сложными задачами.