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-211Б-23

Студент: Савков И.И.

Преподаватель: Бахарев В.Д.

Оценка: _____

Дата: 08.01.2025

Москва, 2025

Постановка задачи

Вариант 22

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.

Правило фильтрации: с вероятностью 80% строки отправляются в pipe1, иначе в pipe2. Дочерние процессы инвертируют строки.

Общий метод и алгоритм решения

Использованные системные вызовы:

- pid_t **fork**(void); – создает дочерний процесс.
- int **pipe**(int *fd); – создает канал и помещает дескрипторы файла для чтения и записи в fd[0] и fd[1].
- pid_t getpid(void); – возвращает ID вызывающего процесса.
- int **open**(const char *__file, int __oflag, …); – используется для открытия файла для чтения, записи или и того, и другого.
- ssize_t **write**(int __fd, const void *__buf, size_t __n); – Записывает N байт из буфер(BUF) в файл (FD). Возвращает количество записанных байт или -1.
- void **exit**(int __status); – выполняет немедленное завершение программы. Все используемые программой потоки закрываются, и временные файлы удаляются, управление возвращается ОС или другой программе.
- int **close**(int __fd); – сообщает операционной системе об окончании работы с файловым дескриптором, и закрывает файл(FD).
- int **dup2(**int __fd, int __fd2); – копирует FD в FD2, закрыв FD2 если это требуется.
- int **execv**(const char *__path, char *const *__argv); – заменяет образ текущего процесса на образ нового процесса, определённого в пути path.
- ssize_t **read**(int __fd, void *__buf, size_t __nbytes); – считывает указанное количество байт из файла(FD) в буфер(BUF).
- pid_t **wait**(int *__stat_loc); – используются для ожидания изменения состояния процесса-потомка вызвавшего процесса и получения информации о потомке, чьё состояние изменилось.

Для выполнения данной лабораторной работы я изучил указанные выше системные вызовы, а также пример выполнения подобного задания.

Программа parent.c получает на вход два аргумента – пути к файлам, в которые требуется записать результат работы. После создаём два канала с помощью **pipe** для общения с двумя дочерними процессами. Далее выполняется **fork**()

Если процесс дочерний, то используем dup2() для копирования файлового дескриптора канала и с помощью **execv**() подменяем образ текущего процесса на новый(child).

Если процесс – родитель, то делаем ещё один **fork**(), далее повторяем те же действия, если мы в дочернем процессе. Если же мы родитель, то начинаем читать строки из потока ввода и по очереди передавать то первому дочернему процессу, то второму в зависимости от правила фильтрации. После окончания ввода ждём
завершения обоих дочерних процессов и программа завершается.

Программа child записывает в переназначенный канал stdout(который является открытым файлом в parent.c), после этого считывает строки из (подменён на вывод канала родительского), переворачивает и записывает в открытый файл. При окончании ввода строк файл закрывается, программа завершается.

Код программы

child.c

```c
#include <string.h>

#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>


void reverse_string(char *str) {
    int len = strlen(str);
    for (int i = 0; i < len / 2; ++i) {
        char temp = str[i];
        str[i] = str[len - i - 1];
        str[len - i - 1] = temp;
    }
}

int main(int argc, char *argv[]) {
    char *end;
    int recieved_number;
    char status;

    fread(&status, sizeof(char), 1, stdin);
    while (status != EOF) {
        fread(&recieved_number, sizeof(recieved_number), 1, stdin);
        char *row = (char *) malloc(sizeof(char) * recieved_number);
        row[0] = '\0';

        fread(row, sizeof(char), recieved_number, stdin);

        row[recieved_number] = '\0';
        reverse_string(row);

        char space = '\n';
        write(STDOUT_FILENO, row, recieved_number);
        write(STDOUT_FILENO, &space, 1);

        fread(&status, sizeof(char), 1, stdin);
        free(row);
    }

    close(STDIN_FILENO);
```

```
    return 0;
}
```

parent.c

```
#include <fcntl.h>
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <time.h>
#include <sys/wait.h>
#include <stdbool.h>

typedef enum {
    OK,
    INVALID_INPUT,
    INVALID_FILES,
    MEMORY_ERROR,
    ERROR_FORK,
    INVALID_PIPE,
    ERROR_EXECV,
} state;

char *get_row(char *symbol);

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

    //Валидация на кол-во аргументов
    if (argc != 3) {
        char *msg_error = "[PARENT] ERROR: INVALID_INPUT.\n";
        write(STDERR_FILENO, msg_error, strlen(msg_error));
        exit(INVALID_INPUT);
    }

    // Создаем на запись файл для Дочернего процесса 1
    char *input_path1 = argv[1];
    int32_t file1 = open(input_path1, O_WRONLY | O_TRUNC | 0600);
    if (file1 == -1) {
        const char msg[] = "[PARENT] ERROR: failed to open requested file\n";
        write(STDERR_FILENO, msg, sizeof(msg));
        exit(INVALID_FILES);
    }

    // Создаем на запись файл для Дочернего процесса 2
    char *input_path2 = argv[2];
    int32_t file2 = open(input_path2, O_WRONLY | O_TRUNC | 0600);
    if (file2 == -1) {
        const char msg[] = "[PARENT] ERROR: failed to open requested file\n";
        write(STDERR_FILENO, msg, sizeof(msg));
        exit(INVALID_FILES);
    }

    //Создаем каналы для передачи строк в дочерние процессы
    int pipe1[2], pipe2[2];
    if (pipe(pipe1) == -1 || pipe(pipe2) == -1) {
        const char *msg_error = "[PARENT] ERROR: INVALID_PIPE.\n";
        write(STDERR_FILENO, msg_error, strlen(msg_error));
        exit(INVALID_PIPE);
    }

    // Создаем дочерний процесс 1
    const pid_t child1 = fork();
    if (child1 == -1) {
        const char *msg_error = "[PARENT] ERROR: INVALID_FORK.\n";
```

```
        write(STDERR_FILENO, msg_error, strlen(msg_error));
        close(file1);
        close(file2);
        exit(ERROR_FORK);
    }

    // Дочерний процесс 1
    if (child1 == 0) {
        // Закрываем другой pipe для ДЧ2
        close(pipe2[1]);
        close(pipe2[0]);

        dup2(pipe1[0], STDIN_FILENO);


        //путь до child1
        const char *path1 = "./child1";

        char fd[10];
        snprintf(fd, sizeof(fd) - 1, "%d", file1);

        dup2(file1, STDOUT_FILENO);

        char *const args[] = {"child1", fd, NULL};
        int32_t status = execv(path1, args); // Запускаем child1.c

        if (status == -1) {
            const char *msg_error = "[PARENT] ERROR: ERROR_EXECV1\n";
            write(STDERR_FILENO, msg_error, strlen(msg_error));
            close(file1);
            close(file2);
            exit(ERROR_EXECV);
        }
    }

    // Создаем дочерний процесс 2
    pid_t child2 = fork();
    if (child2 == -1) {
        const char *msg_error = "[PARENT] ERROR: INVALID_FORK.\n";
        write(STDERR_FILENO, msg_error, strlen(msg_error));
        close(file1);
        close(file2);
        exit(ERROR_FORK);
    }

    // Дочерний процесс 2
    if (child2 == 0) {

        // Закрываем другой pipe для ДЧ2
        close(pipe1[1]);
        close(pipe1[0]);

        dup2(pipe2[0], STDIN_FILENO);


        //Полный путь до child2
        const char *path2 = "./child2";

        // Создаем файловый дескриптор для ДЧ2
        char fd[10];
        snprintf(fd, sizeof(fd) - 1, "%d", file2);

        dup2(file2, STDOUT_FILENO);

        char *const args[] = {"child2", fd, NULL};
        int32_t status = execv(path2, args); // Запускаем child2.c
```

```
        if (status == -1) {
            const char *msg_error = "[PARENT] ERROR: ERROR_EXECV2\n";
            write(STDERR_FILENO, msg_error, strlen(msg_error));
            close(file1);
            close(file2);
            exit(ERROR_EXECV);
        }
    }

    close(pipe1[0]);
    close(pipe2[0]);


    // Считываем из буфера ввода, пока не встретим EOF
    // In Windows, Control+Z is the typical keyboard shortcut to mean
    // "end of file", in Linux and Unix it's typically Control+D.

    //Пишем сообщение
    char *msg = "Please enter the lines you want to invert. Press 'CTRL + D' to
exit.\n";
    write(STDOUT_FILENO, msg, strlen(msg));


    srand(time(NULL));

    char symbol = '0';
    while (symbol != EOF) {
        int random_number = rand() % 100;
        char msg_pipe[512];
        int len;

        char *buf = get_row(&symbol);
        if (buf == NULL) {
            const char *msg_error = "ERROR: MEMORY_ERROR\n";
            write(STDERR_FILENO, msg_error, strlen(msg_error));
            free(buf);

            close(file1);
            close(file2);
            exit(MEMORY_ERROR);


        }
        if (symbol == EOF) {
            break;
        }

        len = strlen(buf);
        if (random_number < 80) {
            write(pipe1[1], &symbol, sizeof(char));
            write(pipe1[1], &len, sizeof(len));
            write(pipe1[1], buf, len);

            uint32_t len_msg = snprintf(msg_pipe, sizeof(msg_pipe) - 1,
                                        "[PARENT] Sent to pipe1: %s\n", buf);
            write(STDIN_FILENO, msg_pipe, len_msg);

        } else {
            write(pipe2[1], &symbol, sizeof(char));
            write(pipe2[1], &len, sizeof(len));
            write(pipe2[1], buf, len);

            uint32_t len_msg = snprintf(msg_pipe, sizeof(msg_pipe) - 1,
                                        "[PARENT] Sent to pipe2: %s\n", buf);
            write(STDIN_FILENO, msg_pipe, len_msg);
        }
        free(buf);
    }
```

```
    symbol = EOF;
    write(pipe1[1], &symbol, sizeof(char));
    write(pipe2[1], &symbol, sizeof(char));




    // Закрываем pipe-ы в родительском процессе после полной передачи строк
    close(pipe1[1]);
    close(pipe2[1]);
    close(file1);
    close(file2);

    wait(NULL);
    wait(NULL);

    return OK;
}


char *get_row(char *symbol) {
    int size = 0;
    int capacity = 2;
    char *buf = (char *) malloc(sizeof(char) * capacity);

    if (buf == NULL) {
        return NULL;
    }

    *symbol = (char) getchar();

    while (*symbol != '\n' && *symbol != EOF) {
        // Проверка на переполнение
        if (size == capacity) {
            capacity *= 2;
            char *buffer_realloc = (char *) realloc(buf, sizeof(char) * capacity);
            if (buffer_realloc == NULL) {
                free(buf);
                return NULL;
            }
            buf = buffer_realloc;
        }
        buf[size] = *symbol;
        size++;

        *symbol = (char) getchar();
    }

    buf[size] = '\0';
    return buf;
}
```

Протокол работы программы

Тестирование:

goldglaid@GoldGlaid0:/mnt/c/Users/GoldGlaid/CLionProjects/OSLab/lab1$ ./parent file1.txt file2.txt

Please enter the lines you want to invert. Press 'CTRL + D' to exit.

test

[PARENT] Sent to pipe1: test

123

[PARENT] Sent to pipe2: 123

4567824

[PARENT] Sent to pipe1: 4567824

Minecraft film soon

[PARENT] Sent to pipe1: Minecraft film soon

proverka

[PARENT] Sent to pipe1: proverka

0987654321

[PARENT] Sent to pipe1: 0987654321

goldglaid@GoldGlaid0:/mnt/c/Users/GoldGlaid/CLionProjects/OSLab/lab1$ strace -f ./parent file1.txt file2.txt

execve("./parent", ["./parent", "file1.txt", "file2.txt"], 0x7ffc7fa39f18 /* 26 vars */) = 0

brk(NULL) = 0x55cfb2657000

arch_prctl(0x3001 /* ARCH_??? */, 0x7fff4ffe9550) = -1 EINVAL (Invalid argument)

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f5f0f91f000

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=25483, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 25483, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f5f0f918000

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\0I\17\357\204\3$\f\221\2039x\324\224\323\236S"..., 68, 896) = 68

newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2220400, ...}, 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, 2264656, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f5f0f6ef000

mprotect(0x7f5f0f717000, 2023424, PROT_NONE) = 0

mmap(0x7f5f0f717000, 1658880, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7f5f0f717000

mmap(0x7f5f0f8ac000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1bd000) = 0x7f5f0f8ac000

mmap(0x7f5f0f905000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x215000) = 0x7f5f0f905000

mmap(0x7f5f0f90b000, 52816, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f5f0f90b000

close(3) = 0

mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f5f0f6ec000

arch_prctl(ARCH_SET_FS, 0x7f5f0f6ec740) = 0

set_tid_address(0x7f5f0f6eca10) = 64208

set_robust_list(0x7f5f0f6eca20, 24) = 0

rseq(0x7f5f0f6ed0e0, 0x20, 0, 0x53053053) = 0

mprotect(0x7f5f0f905000, 16384, PROT_READ) = 0

mprotect(0x55cf9eb94000, 4096, PROT_READ) = 0

mprotect(0x7f5f0f959000, 8192, PROT_READ) = 0

prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0

munmap(0x7f5f0f918000, 25483) = 0

openat(AT_FDCWD, "file1.txt", O_WRONLY|O_EXCL|O_NOCTTY|O_TRUNC) = 3

openat(AT_FDCWD, "file2.txt", O_WRONLY|O_EXCL|O_NOCTTY|O_TRUNC) = 4

pipe2([5, 6], 0) = 0

pipe2([7, 8], 0) = 0

clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLDstrace: Process 64209 attached

, child_tidptr=0x7f5f0f6eca10) = 64209

[pid 64209] set_robust_list(0x7f5f0f6eca20, 24 <unfinished ...>

[pid 64208] clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD <unfinished ...>

[pid 64209] <... set_robust_list resumed>) = 0

[pid **64209] close(8strace: Process 64210 attached**

<unfinished ...>

[pid 64208] <... clone resumed>, child_tidptr=0x7f5f0f6eca10) = 64210

[pid 64209] <... close resumed>) = 0

[pid 64208] close(5 <unfinished ...>

[pid 64210] set_robust_list(0x7f5f0f6eca20, 24 <unfinished ...>

[pid 64208] <... close resumed>) = 0

[pid 64209] close(7 <unfinished ...>

[pid 64208] close(7 <unfinished ...>

[pid 64210] <... set_robust_list resumed>) = 0

[pid 64208] <... close resumed>) = 0

[pid 64209] <... close resumed>) = 0

[pid 64208] write(1, "Please enter the lines you want "..., 69 <unfinished ...>

Please enter the lines you want to invert. Press 'CTRL + D' to exit.

[pid 64210] close(6 <unfinished ...>

[pid 64208] <... write resumed>) = 69

[pid **64209] dup2(5, 0 <unfinished ...>**

[pid 64208] getrandom(<unfinished ...>

[pid 64210] <... close resumed>) = 0

[pid 64208] <... getrandom resumed>"\xfb\xc0\xa1\x4f\x2a\xd5\x2a\x4c", 8, GRND_NONBLOCK) = 8

[pid 64209] <... dup2 resumed>) = 0

[pid 64208] brk(NULL <unfinished ...>

[pid **64210] close(5 <unfinished ...>**

[pid 64208] <... brk resumed>) = 0x55cfb2657000

[pid **64209] dup2(3, 1 <unfinished ...>**

[pid 64208] brk(0x55cfb2678000 <unfinished ...>

[pid 64210] <... close resumed>) = 0

[pid 64208] <... brk resumed>) = 0x55cfb2678000

[pid 64209] <... dup2 resumed>) = 1

[pid **64210] dup2(7, 0 <unfinished ...>**

[pid 64208] newfstatat(0, "", <unfinished ...>

[pid 64209] **execve**("./child1", ["child1", "3"], 0x7fff4ffe9738 /* 26 vars */ <unfinished ...>

[pid 64208] <... newfstatat resumed>{st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x2), ...}, AT_EMPTY_PATH) = 0

[pid 64210] <... dup2 resumed>) = 0

[pid 64208] read(0, <unfinished ...>

[pid 64210] dup2(4, 1) = 1

[pid 64210] execve("./child2", ["child2", "4"], 0x7fff4ffe9738 /* 26 vars */ <unfinished ...>

[pid 64209] <... execve resumed>) = 0

[pid 64209] brk(NULL) = 0x55948bb84000

[pid 64210] <... execve resumed>) = 0

[pid 64209] arch_prctl(0x3001 /* ARCH_??? */, 0x7ffeeebb4b40 <unfinished ...>

[pid 64210] brk(NULL <unfinished ...>

[pid 64209] <... arch_prctl resumed>) = -1 EINVAL (Invalid argument)

[pid 64210] <... brk resumed>) = 0x55da8b8d2000

[pid 64209] mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 64210] arch_prctl(0x3001 /* ARCH_??? */, 0x7ffc6291bb00 <unfinished ...>

[pid 64209] <... mmap resumed>) = 0x7fc04308d000

[pid 64210] <... arch_prctl resumed>) = -1 EINVAL (Invalid argument) [pid

64209] access("/etc/ld.so.preload", R_OK <unfinished ...>

[pid 64210] mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 64209] <... access resumed>) = -1 ENOENT (No such file or directory)

[pid 64210] <... mmap resumed>) = 0x7fa7b1923000

[pid 64209] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>

[pid 64210] access("/etc/ld.so.preload", R_OK <unfinished ...>

[pid 64209] <... openat resumed>) = 7

[pid 64210] <... access resumed>) = -1 ENOENT (No such file or directory)

[pid 64209] newfstatat(7, "", <unfinished ...>

[pid 64210] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>

[pid 64209] <... newfstatat resumed>{st_mode=S_IFREG|0644, st_size=25483, ...}, AT_EMPTY_PATH) = 0

[pid 64210] <... openat resumed>) = 5

[pid 64209] mmap(NULL, 25483, PROT_READ, MAP_PRIVATE, 7, 0 <unfinished ...>

[pid 64210] newfstatat(5, "", <unfinished ...>

[pid 64209] <... mmap resumed>) = 0x7fc043086000

[pid 64210] <... newfstatat resumed>{st_mode=S_IFREG|0644, st_size=25483, ...}, AT_EMPTY_PATH) = 0

[pid **64209] close(7 <unfinished ...>**

[pid 64210] mmap(NULL, 25483, PROT_READ, MAP_PRIVATE, 5, 0 <unfinished ...>

[pid 64209] <... close resumed>) = 0

[pid 64210] <... mmap resumed>) = 0x7fa7b191c000

[pid 64210] close(5 <unfinished ...>

[pid 64209] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC <unfinished ...>

[pid **64210] <... close resumed>) = 0**

[pid 64209] <... openat resumed>) = 7

[pid 64210] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC <unfinished ...>

[pid 64209] read(7, <unfinished ...>

[pid 64210] <... openat resumed>) = 5

```
[pid 64209] <... 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"..., 832) = 832
[pid 64210] read(5,  <unfinished ...>
[pid 64209] pread64(7,  <unfinished ...>
[pid 64210] <... 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"..., 832) = 832
[pid 64209] <... 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 64210] pread64(5,  <unfinished ...>
[pid 64209] pread64(7,  <unfinished ...>
[pid 64210] <... 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 64209] <... 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 64210] pread64(5,  <unfinished ...>
[pid 64209] pread64(7,  <unfinished ...>
[pid 64210] <... 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 64209] <... pread64
resumed>"\4\0\0\0\24\0\0\0\3\0\0\0GNU\0I\17\357\204\3$\f\221\2039x\324\224\323\236S"..., 68, 896) = 68
[pid 64210] pread64(5,  <unfinished ...>  [pid
64209] newfstatat(7, "",  <unfinished ...> [pid
64210] <... pread64
resumed>"\4\0\0\0\24\0\0\0\3\0\0\0GNU\0I\17\357\204\3$\f\221\2039x\324\224\323\236S"..., 68, 896) = 68
[pid 64209] <... newfstatat resumed>{st_mode=S_IFREG|0755, st_size=2220400, ...}, AT_EMPTY_PATH) = 0
[pid 64210] newfstatat(5, "",  <unfinished ...>
[pid 64209] pread64(7,  <unfinished ...>
[pid 64210] <... newfstatat resumed>{st_mode=S_IFREG|0755, st_size=2220400, ...}, AT_EMPTY_PATH) = 0
[pid 64209] <... 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 64210] pread64(5,  <unfinished ...>
[pid 64209] mmap(NULL, 2264656, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 7, 0 <unfinished ...>
[pid 64210] <... 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 64209] <... mmap resumed>)          = 0x7fc042e5d000
[pid 64210] mmap(NULL, 2264656, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 5, 0 <unfinished ...>
[pid 64209] mprotect(0x7fc042e85000, 2023424, PROT_NONE <unfinished ...>
[pid 64210] <... mmap resumed>)          = 0x7fa7b16f3000
[pid 64209] <... mprotect resumed>)      = 0
[pid 64210] mprotect(0x7fa7b171b000, 2023424, PROT_NONE <unfinished ...>
```

[pid 64209] mmap(0x7fc042e85000, 1658880, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 7, 0x28000 <unfinished ...>

[pid 64210] <... mprotect resumed>) = 0

[pid 64209] <... mmap resumed>) = 0x7fc042e85000

[pid 64210] mmap(0x7fa7b171b000, 1658880, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 5, 0x28000 <unfinished ...>

[pid 64209] mmap(0x7fc04301a000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 7, 0x1bd000 <unfinished ...>

[pid 64210] <... mmap resumed>) = 0x7fa7b171b000

[pid 64209] <... mmap resumed>) = 0x7fc04301a000

[pid 64210] mmap(0x7fa7b18b0000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 5, 0x1bd000 <unfinished ...>

[pid 64209] mmap(0x7fc043073000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 7, 0x215000 <unfinished ...>

[pid 64210] <... mmap resumed>) = 0x7fa7b18b0000

[pid 64209] <... mmap resumed>) = 0x7fc043073000

[pid 64210] mmap(0x7fa7b1909000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 5, 0x215000 <unfinished ...>

[pid 64209] mmap(0x7fc043079000, 52816, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 64210] <... mmap resumed>) = 0x7fa7b1909000

[pid 64209] <... mmap resumed>) = 0x7fc043079000

[pid 64210] mmap(0x7fa7b190f000, 52816, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid **64209] close(7 <unfinished ...>**

[pid 64210] <... mmap resumed>) = 0x7fa7b190f000

[pid 64209] <... close resumed>) = 0

[pid 64210] close(5) = 0

[pid 64209] mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 64210] mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 64209] <... mmap resumed>) = 0x7fc042e5a000

[pid 64210] <... mmap resumed>) = 0x7fa7b16f0000

[pid 64209] arch_prctl(ARCH_SET_FS, 0x7fc042e5a740 <unfinished ...>

[pid 64210] arch_prctl(ARCH_SET_FS, 0x7fa7b16f0740 <unfinished ...>

[pid 64209] <... arch_prctl resumed>) = 0

[pid 64210] <... arch_prctl resumed>) = 0

[pid 64209] set_tid_address(0x7fc042e5aa10 <unfinished ...>

[pid 64210] set_tid_address(0x7fa7b16f0a10 <unfinished ...>

[pid 64209] <... set_tid_address resumed>) = 64209

[pid 64210] <... set_tid_address resumed>) = 64210

[pid 64209] set_robust_list(0x7fc042e5aa20, 24 <unfinished ...>

[pid 64210] set_robust_list(0x7fa7b16f0a20, 24 <unfinished ...>

[pid 64209] <... set_robust_list resumed>) = 0

[pid 64210] <... set_robust_list resumed>) = 0

[pid 64209] rseq(0x7fc042e5b0e0, 0x20, 0, 0x53053053 <unfinished ...>

[pid 64210] rseq(0x7fa7b16f10e0, 0x20, 0, 0x53053053 <unfinished ...>

[pid 64209] <... rseq resumed>)   = 0

[pid 64210] <... rseq resumed>)   = 0

[pid 64210] mprotect(0x7fa7b1909000, 16384, PROT_READ <unfinished ...>

[pid 64209] mprotect(0x7fc043073000, 16384, PROT_READ <unfinished ...>

[pid 64210] <... mprotect resumed>)   = 0

[pid 64209] <... mprotect resumed>)   = 0

[pid 64210] mprotect(0x55da6b23b000, 4096, PROT_READ <unfinished ...>

[pid 64209] mprotect(0x559454063000, 4096, PROT_READ <unfinished ...>

[pid 64210] <... mprotect resumed>)   = 0

[pid 64209] <... mprotect resumed>)   = 0

[pid 64210] mprotect(0x7fa7b195d000, 8192, PROT_READ <unfinished ...>

[pid 64209] mprotect(0x7fc0430c7000, 8192, PROT_READ <unfinished ...>

[pid 64210] <... mprotect resumed>)   = 0

[pid 64209] <... mprotect resumed>)   = 0

[pid 64210] prlimit64(0, RLIMIT_STACK, NULL, <unfinished ...>

[pid 64209] prlimit64(0, RLIMIT_STACK, NULL, <unfinished ...>

[pid 64210] <... prlimit64 resumed>{rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0

[pid 64209] <... prlimit64 resumed>{rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0

[pid 64210] munmap(0x7fa7b191c000, 25483) = 0

[pid 64209] munmap(0x7fc043086000, 25483 <unfinished ...>

[pid 64210] newfstatat(0, "", <unfinished ...>

[pid 64209] <... munmap resumed>)   = 0

```
[pid 64210] <... newfstatat resumed>{st_mode=S_IFIFO|0600, st_size=0, ...}, AT_EMPTY_PATH) = 0
[pid 64209] newfstatat(0, "",  <unfinished ...>
[pid 64210] getrandom( <unfinished ...>
[pid 64209] <... newfstatat resumed>{st_mode=S_IFIFO|0600, st_size=0, ...}, AT_EMPTY_PATH) = 0
[pid 64210] <... getrandom resumed>"\x2f\x7c\x46\x0e\xef\xba\x1b\x75", 8, GRND_NONBLOCK) = 8
[pid 64209] getrandom( <unfinished ...>
[pid 64210] brk(NULL <unfinished ...>
[pid 64209] <... getrandom resumed>"\x59\x0c\x48\x50\x52\xed\xd2\xa0", 8, GRND_NONBLOCK) = 8
[pid 64210] <... brk resumed>)          = 0x55da8b8d2000
[pid 64209] brk(NULL <unfinished ...>
[pid 64210] brk(0x55da8b8f3000 <unfinished ...>
[pid  64209] <... brk resumed>)      = 0x55948bb84000
[pid  64210] <... brk resumed>)      = 0x55da8b8f3000
[pid 64209] brk(0x55948bba5000 <unfinished ...>
[pid 64210] read(0,  <unfinished ...>
[pid  64209] <... brk resumed>)      = 0x55948bba5000
[pid 64209] read(0, test
 <unfinished ...>
[pid 64208] <... read resumed>"test\n", 1024) = 5
[pid 64208] write(6, "\n", 1)           = 1
[pid 64209] <... read resumed>"\n", 4096) = 1 [pid
64208] write(6, "\4\0\0\0", 4 <unfinished ...> [pid
64209] read(0,  <unfinished ...>
[pid 64208] <... write resumed>)          = 4
[pid 64209] <... read resumed>"\4\0\0\0", 4096) = 4
[pid 64208] write(6, "test", 4 <unfinished ...>
[pid 64209] read(0,  <unfinished ...>
[pid 64208] <... write resumed>)          = 4
[pid 64209] <... read resumed>"test", 4096) = 4
[pid 64208] write(0, "[PARENT] Sent to pipe1: test\n", 29 <unfinished ...> [PARENT]
Sent to pipe1: test
[pid 64209] write(1, "tset", 4 <unfinished ...>
[pid 64208] <... write resumed>)          = 29
```

```
[pid 64208] read(0,  <unfinished ...>  [pid
64209] <... write resumed>)               = 4
[pid 64209] write(1, "\n", 1)             = 1
[pid 64209] read(0, Goaaaal
 <unfinished ...>
[pid 64208] <... read resumed>"Goaaaal\n", 1024) = 8
[pid 64208] write(6, "\n", 1)             = 1
[pid 64209] <... read resumed>"\n", 4096) = 1
[pid 64208] write(6, "\7\0\0\0", 4)       = 4
[pid 64209] read(0,  <unfinished ...>
[pid 64208] write(6, "Goaaaal", 7 <unfinished ...>
[pid 64209] <... read resumed>"\7\0\0\0", 4096) = 4
[pid 64208] <... write resumed>)          = 7
[pid 64209] read(0,  <unfinished ...>
[pid 64208] write(0, "[PARENT] Sent to pipe1: Goaaaal\n", 32 <unfinished ...>
[pid 64209] <... read resumed>"Goaaaal", 4096) = 7
[PARENT] Sent to pipe1: Goaaaal
[pid 64208] <... write resumed>)          = 32
[pid 64209] write(1, "laaaaoG", 7 <unfinished ...>
[pid 64208] read(0,  <unfinished ...>
[pid 64209] <... write resumed>)          = 7
[pid 64209] write(1, "\n", 1)             = 1
[pid 64209] read(0, HOOOOOOOOOL
 <unfinished ...>
[pid 64208] <... read resumed>"HOOOOOOOOOL\n", 1024) = 12
[pid 64208] write(6, "\n", 1)             = 1
[pid 64209] <... read resumed>"\n", 4096) = 1 [pid
64208] write(6, "\v\0\0\0", 4 <unfinished ...> [pid
64209] read(0,  <unfinished ...>
[pid 64208] <... write resumed>)          = 4
[pid 64209] <... read resumed>"\v\0\0\0", 4096) = 4
[pid 64208] write(6, "HOOOOOOOOOL", 11 <unfinished ...>
[pid 64209] read(0,  <unfinished ...>
```

[pid 64208] <... write resumed>)   = 11

[pid 64209] <... read resumed>"HOOOOOOOOOL", 4096) = 11

[pid 64208] write(0, "[PARENT] Sent to pipe1: HOOOOOOO"..., 36 <unfinished ...> [PARENT]

Sent to pipe1: HOOOOOOOOOL

[pid 64209] write(1, "LOOOOOOOOOH", 11 <unfinished ...>

[pid 64208] <... write resumed>)   = 36

[pid 64208] read(0, <unfinished ...>

[pid 64209] <... write resumed>)   = 11

[pid 64209] write(1, "\n", 1)   = 1

[pid 64209] read(0, What are you searching here?

 <unfinished ...>

[pid 64208] <... read resumed>"What are you searching here?\n", 1024) = 29

[pid 64208] write(6, "\n", 1)   = 1

[pid 64209] <... read resumed>"\n", 4096) = 1   [pid

64208] write(6, "\34\0\0\0", 4 <unfinished ...> [pid

64209] read(0, <unfinished ...>

[pid 64208] <... write resumed>)   = 4

[pid 64209] <... read resumed>"\34\0\0\0", 4096) = 4

[pid 64208] write(6, "What are you searching here?", 28 <unfinished ...>

[pid 64209] read(0, <unfinished ...>

[pid 64208] <... write resumed>)   = 28

[pid 64209] <... read resumed>"What are you searching here?", 4096) = 28

[pid 64208] write(0, "[PARENT] Sent to pipe1: What are"..., 53 <unfinished ...>

[PARENT] Sent to pipe1: What are you searching here?

[pid 64209] write(1, "?ereh gnihcraes uoy era tahW", 28 <unfinished ...>

[pid 64208] <... write resumed>)   = 53

[pid 64208] read(0, <unfinished ...>

[pid 64209] <... write resumed>)   = 28

[pid 64209] write(1, "\n", 1)   = 1

[pid 64209] read(0, <unfinished ...>

[pid 64208] <... read resumed>"", 1024) = 0

[pid 64208] write(6, "\377", 1)   = 1

[pid 64209] <... read resumed>"\377", 4096) = 1

[pid 64208] write(8, "\377", 1 <unfinished ...>

[pid **64209] close(0 <unfinished ...>**

[pid 64208] <... write resumed>) = 1

[pid 64210] <... read resumed>"\377", 4096) = 1

[pid 64209] <... close resumed>) = 0

[pid **64208] close(6 <unfinished ...>**

[pid **64210] close(0 <unfinished ...>**

[pid 64208] <... close resumed>) = 0

[pid 64210] <... close resumed>) = 0

[pid **64208] close(8 <unfinished ...>**

[pid 64209] exit_group(0 <unfinished ...>

[pid 64208] <... close resumed>) = 0

[pid 64210] exit_group(0 <unfinished ...>

[pid **64208] close(3 <unfinished ...>**

[pid 64209] <... exit_group resumed>) = ?

[pid 64208] <... close resumed>)= 0 [pid

64210] <... exit_group resumed>) = ? [pid

64208] close(4) = 0

[pid **64208] wait4(-1, <unfinished ...>**

[pid 64209] +++ exited with 0 +++

[pid 64208] <... wait4 resumed>NULL, 0, NULL) = 64209

[pid 64208] --- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=64209, si_uid=1000, si_status=0, si_utime=0, si_stime=1} ---

[pid **64208] wait4(-1, <unfinished ...>**

[pid 64210] +++ exited with 0 +++

<... wait4 resumed>NULL, 0, NULL) = 64210

--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=64210, si_uid=1000, si_status=0, si_utime=0, si_stime=0} ---

exit_group(0) = ?

+++ exited with 0 +++

Вывод

В ходе написания данной лабораторной работы я научился работать с системными вызовами в СИ. Научился создавать программы, состоящие из нескольких процессов, и передавать данные между процессами по каналам. Во время отладки программы я

познакомился с утилитой strace, она оказалась достаточно удобной для получения информации о работе многопоточных программ.