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-216Б-24

Студент: Ходаков Павел

Преподаватель: Бахарев В.Д.

Оценка: _____

Дата: XX.XX.24

Москва, 2025

Постановка задачи

Вариант 7.

Родительский процесс создает дочерний процесс. Первой строчкой пользователь в консоль родительского процесса вводит имя файла, которое будет использовано для открытия файла с таким именем на чтение. Стандартный поток ввода дочернего процесса переопределяется открытым файлом. Дочерний процесс читает команды из стандартного потока ввода.

Стандартный поток вывода дочернего процесса перенаправляется в pipe1. Родительский процесс читает из pipe1 и прочитанное выводит в свой стандартный поток вывода. Родительский и дочерний процесс должны быть представлены разными программами.

В файле записаны команды вида: «число число число<endline>». Дочерний процесс считает их сумму и выводит результат в стандартный поток вывода. Числа имеют тип float. Количество чисел может быть произвольным.

Общий метод и алгоритм решения

Использованные системные вызовы:

- int open(const char *pathname, int flags, mode_t mode) – возвращает файловый дескриптор
- ssize_t read(int fd, void *buf, size_t count) – возвращает число реально прочитанных байт
- ssize_t write(int fd, const void *buf, size_t count) – возвращает число реально записанных байт
- int close(int fd) – закрывает объект по файловому дескриптору
- int dup2(int oldfd, int newfd) – дублирует файловый дескриптор, потом newfd указывает туда же, куда и oldfd
- int pipe(pipefd[2]) – создаёт неименованный канал
- pid_t fork(void); – создает дочерний процесс.
- int execl(const char *path, const char *arg, (char *) NULL – заменяет код текущего процесса новой программой
- pid_t waitpid(pid, &status, options) – ожидает завершения процесса с данным pid
- void exit(status) – завершает процесс с очисткой
- void _exit(status) – завершает процесс без очистки

В рамках лабораторной работы я изучил теоретический материал (материалы лекций и примеры кода, материалы из интернета по системным вызовам).

Родительский процесс:

1. Посимвольно считывает из стандартного ввода имя файла до конца строки или конца ввода.
2. Открывает указанный файл на чтение.
3. Создаёт неименованный канал (pipe) для обмена данными с дочерним процессом.
4. Порождает дочерний процесс с помощью fork.
5. После fork:
   закрывает ненужный конец канала;
   читает данные, поступающие из канала от дочернего процесса;
   выводит их в собственный стандартный вывод.
6. Ожидает завершения дочернего процесса (waitpid) и корректно завершает работу.

Дочерний процесс:

1. Перенаправляет стандартный ввод на открытый файл, а стандартный вывод — на конец для записи в канал.

2. Закрывает лишние файловые дескрипторы.

3. Вызывает другую программу через execl.

4. Эта программа:

посимвольно считывает строку из стандартного ввода (теперь это файл);

парсит строку как неизвестное количество чисел с плавающей точкой, разделённых пробелами;

вычисляет их сумму;

записывает результат в стандартный вывод (который перенаправлен в канал).

Код программы

parent.cpp

```
#include <cstdlib>

#include <fcntl.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/wait.h>

#include <string.h>

#include <stdio.h>


#include <string>


int main() {

    const size_t BUF_SIZE = 4096;

    std::string filename;

    char buf[BUF_SIZE];


    while (true) {

        ssize_t n = read(STDIN_FILENO, buf, 1);

        if (n < 0) {

            if (errno == EINTR) {

                continue;

            }
```

```
        perror("read");
        exit(EXIT_FAILURE);
    }
    if (n == 0) {
        break; // got EOF
    }

    if (buf[0] == '\n') {
        break;
    }
    filename.push_back(buf[0]);
    if (filename.size() >= BUF_SIZE) {
        const char *msg = "Filename is too long.\n";
        write(STDERR_FILENO, msg, strlen(msg));
        exit(EXIT_FAILURE);
    }
}

if (filename.empty()) {
    const char *msg = "Please provide filename.\n";
    write(STDERR_FILENO, msg, strlen(msg));
    exit(EXIT_SUCCESS);
}

int fd_in = open(filename.c_str(), O_RDONLY);
if (fd_in < 0) {
    perror("open");
    exit(EXIT_FAILURE);
}

int pipefd[2];
if (pipe(pipefd) != 0) {
    perror("pipe");
```

```
    close(fd_in);
    exit(EXIT_FAILURE);
}

const pid_t pid = fork();
switch (pid) {
case -1: {
    perror("fork");
    close(fd_in);
    close(pipefd[0]);
    close(pipefd[1]);
    exit(EXIT_FAILURE);
} break;

case 0: {
    // child
    if (dup2(fd_in, STDIN_FILENO) < 0) {
        _exit(EXIT_FAILURE);
    }
    if (dup2(pipefd[1], STDOUT_FILENO) < 0) {
        _exit(EXIT_FAILURE);
    }
    if (dup2(pipefd[1], STDERR_FILENO) < 0) {
        _exit(EXIT_FAILURE);
    }

    close(fd_in);
    close(pipefd[0]);
    close(pipefd[1]);

    execl("./child", "./child", (char*)NULL);
    perror("execl");
    _exit(EXIT_FAILURE);
```

```
} break;


default: {
    // parent
    close(fd_in);
    close(pipefd[1]);


    const size_t RBUF = 4096;
    char buf[RBUF];
    bool had_error = false;
    while (true) {
        ssize_t r = read(pipefd[0], buf, RBUF);
        if (r < 0) {
            if (errno == EINTR) {
                continue;
            }
            perror("read");
            had_error = true;
            break;
        }
        if (r == 0) {
            break; // got EOF
        }


        ssize_t woff = 0;
        while (woff < r) {
            ssize_t w = write(STDOUT_FILENO, buf + woff, r - woff);
            if (w < 0) {
                if (errno == EINTR) {
                    continue;
                }
                perror("write");
                had_error = true;
```

```
                    break;
                }
                woff += w;
            }
            if (had_error) {
                break;
            }
        }

        close(pipefd[0]);

        int status = 0;
        if (waitpid(pid, &status, 0) < 0) {
            perror("waitpid");
            exit(EXIT_FAILURE);
        }

        if (had_error) {
            exit(EXIT_FAILURE);
        }
        exit(EXIT_SUCCESS);
    } break;
    }
}
```

child.cpp

```
#include <unistd.h>

#include <string>
#include <vector>
#include <cstring>
#include <cstdlib>

int main() {
```

```
std::string line;
char ch;
while (true) {
    ssize_t n = read(STDIN_FILENO, &ch, 1);
    if (n < 0) {
        if (errno == EINTR) {
            continue;
        }
        perror("read");
        exit(EXIT_FAILURE);
    }
    if (n == 0) {
        // EOF
        if (line.empty()) {
            break;
        }
    } else {
        if (ch != '\n') {
            line.push_back(ch);
            continue;
        }
    }

    std::vector<float> nums;
    std::string cur;
    for (size_t i = 0; i < line.size(); ++i) {
        char c = line[i];
        if (c == ' ') {
            if (!cur.empty()) {
                nums.push_back(std::atof(cur.c_str()));
                cur.clear();
            }
        } else {
```

```
        cur.push_back(c);
    }
}
if (!cur.empty()) {
    nums.push_back(std::atof(cur.c_str()));
    cur.clear();
}

if (!nums.empty()) {
    float sum = 0;
    for (size_t i = 0; i < nums.size(); ++i) {
        sum += nums[i];
    }
    std::string out = std::to_string(sum) + "\n";
    size_t woff = 0;
    while (woff < out.size()) {
        ssize_t w = write(STDOUT_FILENO, out.c_str() + woff,
                          out.size() - woff);
        if (w < 0) {
            if (errno == EINTR) {
                continue;
            }
            perror("write");
            return EXIT_FAILURE;
        }
        woff += w;
    }
}

line.clear();
if (n == 0) {
    break;
}
```

```
    }


    exit(EXIT_SUCCESS);

}
```

Протокол работы программы

Вывод strace:

```
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, "98765432198765432198765432 1\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

Вывод 3-4 предложения. Также опишите проблемы, с которыми столкнулись при выполнении лабораторной работы (если они были), пожелания и т.д.