

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-214БВ-25

Студент: Татулян А. Г.

Преподаватель: Бахарев В.Д.

Оценка: \_\_\_\_\_

Дата: 01.10.25

Москва, 2025

## Постановка задачи

### Вариант 21.

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 пересыпает 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: нечетные строки отправляются в pipe1, четные 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 неименованный односторонний канал
- **int dup2(int oldfd, int newfd);** – перенаправляет файловый дескриптор.
- **int execv(const char \*path, char \*const argv[]);** – запускает новую программу
- **ssize\_t read(int fd, void \*buf, size\_t count);** – чтение данных (используется родителем для построчного чтения с консоли и дочерним процессом для чтения данных из канала)
- **ssize\_t write(int fd, const void \*buf, size\_t count);** – запись данных (используется родителем для записи фильтрованных строк в каналы, а также дочерним процессом для вывода обработанных строк в файл и на консоль)
- **int close(int fd);** – закрывает файловый дескриптор
- **pid\_t waitpid(pid\_t pid, int \*status, int options);** – ожидает завершения дочернего процесса

Алгоритм решения:

1. Создается родительский процесс и запрашиваются имена двух выходных файлов. Создаются два неименованных канала (pipe1 pipe2)

и дважды вызывается fork(), чтобы создать два дочерних процесса.

Каждый процесс закрывает ту часть канала, которую он не будет использовать.

2. Родитель читает строки, которые вводятся с консоли, разделяя их по индексу, ввод пользователя разделяется на два параллельных потока.

- Нечетные строки: Родитель записывает ее в pipe1[1]
- Четные строки: Родитель записывает ее в pipe2[1]

3. Каждый процесс server читает данные со своего STDIN. На каждой строке вызывается функция, которая разворачивает строку. Результат записывается на выходной файл и на стандартный вывод.

4. Когда родитель завершает ввод, он вызывает close() для обоих концов записи (pipe1[1] и pipe2[1]), функция read() в дочернем процессе возвращает 0. Дочерние процессы выходят из цикла чтения и завершаются. Родитель вызывает waitpid() для каждого дочернего процесса. Он блокируется, пока оба ребенка не завершатся.

## Код программы

### client.c

```
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <fcntl.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>

#define MAX_LINE_LEN 4096

ssize_t read_line(int fd, char *buffer, size_t max_len) {
```

```
ssize_t total_read = 0;
char c;
while (total_read < max_len - 1) {
    ssize_t bytes_read = read(fd, &c, 1);
    if (bytes_read <= 0) {
        return bytes_read; // EOF или ошибка
    }
    if (c == '\n') {
        break;
    }
    buffer[total_read++] = c;
}
buffer[total_read] = '\0';
return total_read;
}
```

```
int main() {
    char filename1[256], filename2[256];
    printf("Введите имя файла для child1: ");
    fflush(stdout);
    if (fgets(filename1, sizeof(filename1), stdin) == NULL) {
        perror("Ошибка чтения имени файла 1");
        exit(EXIT_FAILURE);
    }
    filename1[strcspn(filename1, "\n")] = '\0';
```

```
printf("Введите имя файла для child2: ");
fflush(stdout);

if (fgets(filename2, sizeof(filename2), stdin) == NULL) {
    perror("Ошибка чтения имени файла 2");
    exit(EXIT_FAILURE);
}

filename2[strcspn(filename2, "\n")] = '\0';
```

```
// Создаем два канала (pipe)
int pipe1[2], pipe2[2];
if (pipe(pipe1) == -1 || pipe(pipe2) == -1) {
    perror("Ошибка создания pipe");
    exit(EXIT_FAILURE);
}
```

```
// child1
pid_t pid1 = fork();
if (pid1 == -1) {
    perror("Ошибка fork для child1");
    exit(EXIT_FAILURE);
}
```

```
if (pid1 == 0) {
    // Закрываем неиспользуемые концы pipe
    close(pipe1[1]); // Закрываем конец записи pipe1
    close(pipe2[0]); // Закрываем оба конца pipe2
    close(pipe2[1]);
    // Перенаправляем стандартный ввод на чтение из pipe1
    dup2(pipe1[0], STDIN_FILENO);
```

```
close(pipe1[0]);
execl("./server", "server", filename1, NULL);

}

perror("Ошибка запуска server для child1");
exit(EXIT_FAILURE);
}
```

```
// child2

pid_t pid2 = fork();
if (pid2 == -1) {
perror("Ошибка fork для child2");
exit(EXIT_FAILURE);
}

if (pid2 == 0) {
close(pipe2[1]); // Закрываем конец записи pipe2
close(pipe1[0]); // Закрываем оба конца pipe1
close(pipe1[1]);
// Перенаправляем стандартный ввод на чтение из pipe2
dup2(pipe2[0], STDIN_FILENO);
close(pipe2[0]);

execl("./server", "server", filename2, NULL);
```

```
perror("Ошибка запуска server для child2");
exit(EXIT_FAILURE);
}
```

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

printf("\nВведите строки (Только латинские буквы. Ctrl+D для завершения):\n");
fflush(stdout);

char line[MAX_LINE_LEN];
int line_number = 0;
ssize_t line_length;

// Читаем строки от пользователя и отправляем в соответствующие pipe
while ((line_length = read_line(STDIN_FILENO, line, sizeof(line))) > 0) {
    line_number++;
    int target_pipe;
    const char *child_name;
    if (line_number % 2 == 1) { // Нечетная строка
        target_pipe = pipe1[1];
        child_name = "child1";
    } else { // Четная строка
        target_pipe = pipe2[1];
        child_name = "child2";
    }
    // Отправляем строку в pipe
    write(target_pipe, line, line_length);
    write(target_pipe, "\n", 1);
    printf("[Родитель] Стока %d отправлена в %s: %s\n",
           line_number, child_name, line);
```

```
fflush(stdout);

}

// Закрываем концы pipe для записи
close(pipe1[1]);
close(pipe2[1]);
waitpid(pid1, NULL, 0);
waitpid(pid2, NULL, 0);

printf("Родительский процесс завершен.\n");
return 0;
}
```

### **server.c**

```
#include <unistd.h>
#include <fcntl.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>

#define MAX_LEN 4096

void reverse_string(char *str) {
if (str == NULL || *str == '\0') {
return;
}
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;
}
```

```
str[i] = str[len - 1 - i];
str[len - 1 - i] = temp;
}

}

int main(int argc, char *argv[]) {
if (argc != 2) {
fprintf(stderr, "Использование: %s <имя_файла>\n", argv[0]);
exit(EXIT_FAILURE);
}

const char *filename = argv[1];
int fd = open(filename, O_WRONLY | O_CREAT | O_TRUNC, 0644);
if (fd == -1) {
perror("Ошибка открытия файла");
exit(EXIT_FAILURE);
}

// Получаем PID для идентификации процесса
pid_t my_pid = getpid();
printf("[Server PID=%d] Начал работу. Файл: %s\n", my_pid, filename);
fflush(stdout);

char buffer[MAX_LEN];
ssize_t bytes_read;

// Читаем строки из стандартного ввода (который перенаправлен из pipe)
while ((bytes_read = read(STDIN_FILENO, buffer, sizeof(buffer) - 1)) > 0) {
// Завершаем строку нулевым символом
buffer[bytes_read] = '\0';
if (buffer[bytes_read - 1] == '\n') {
buffer[bytes_read - 1] = '\0';
bytes_read--;
}
}
```

```
if (bytes_read == 0) {
    continue;
}

char original[MAX_LEN];
strncpy(original, buffer, sizeof(original) - 1);
original[sizeof(original) - 1] = '\0';
reverse_string(buffer);

char output[MAX_LEN * 2];
int output_len = snprintf(output, sizeof(output),
    "[Server PID=%d] Исходная: '%s' -> Развернутая: '%s'\n",
    my_pid, original, buffer);

if (write(fd, output, output_len) == -1) {
    perror("Ошибка записи в файл");
}

if (write(STDOUT_FILENO, output, output_len) == -1) {
    perror("Ошибка вывода на экран");
}

if (bytes_read == -1) {
    perror("Ошибка чтения из pipe");
}

close(fd);
printf("[Server PID=%d] Завершил работу.\n", my_pid);
fflush(stdout);
return 0;
}
```

## Протокол работы программы

### Тесты

```
vscode → /workspaces/MAI_OS_Labs/lab1/src (main) $ gcc -o server server.c
vscode → /workspaces/MAI_OS_Labs/lab1/src (main) $ gcc -o client client.c
vscode → /workspaces/MAI_OS_Labs/lab1/src (main) $ ./client
Введите имя файла для child1: f1.txt
Введите имя файла для child2: f2.txt

Вводите строки (Только латинские буквы. Ctrl+D для завершения):
[Server PID=12643] Начал работу. Файл: f1.txt
[Server PID=12644] Начал работу. Файл: f2.txt
hello
[Родитель] Стока 1 отправлена в child1: hello
[Server PID=12643] Исходная: 'hello' -> Развернутая: 'olleh'
GoodBye
[Родитель] Стока 2 отправлена в child2: GoodBye
[Server PID=12644] Исходная: 'GoodBye' -> Развернутая: 'eyBdooG'
i dont know77
[Родитель] Стока 3 отправлена в child1: i dont know77
[Server PID=12643] Исходная: 'i dont know77' -> Развернутая: '77wonk tnod i'
[Server PID=12644] Завершил работу.
[Server PID=12643] Завершил работу.
Родительский процесс завершен.
```

```
lab1 > src > ≡ f1.txt
1 [Server PID=12643] Исходная: 'hello' -> Развернутая: 'olleh'
2 [Server PID=12643] Исходная: 'i dont know77' -> Развернутая: '77wonk tnod i'
3
```

```
lab1 > src > ≡ f2.txt
1 [Server PID=12644] Исходная: 'GoodBye' -> Развернутая: 'eyBdooG'
2
```

Strace

rseq(0xfffffa9cc0600, 0x20, 0, 0xd428bc00) = 0  
mprotect(0xfffffa9c6d000, 12288, PROT\_READ) = 0  
mprotect(0xaaab17bf000, 4096, PROT\_READ) = 0  
mprotect(0xfffffa9cc5000, 8192, PROT\_READ) = 0  
prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024,  
rlim\_max=RLIM64\_INFINITY}) = 0  
munmap(0xfffffa9cb9000, 21571) = 0  
fstat(1, {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0), ...}) = 0  
getrandom("6d7c0c6b2c", 8, GRND\_NONBLOCK) = 8  
brk(NULL) = 0xaaaaea5a4000  
brk(0xaaaaea5c5000) = 0xaaaaea5c5000  
write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265  
\320\270\320\274\321\217 \321\204\320\260\320\271\320\273\320\260"..., 48Ведите  
имя файла для child1: ) = 48  
fstat(0, {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0), ...}) = 0  
read(0, f1.txt  
"f1.txt", 1024) = 7  
write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265  
\320\270\320\274\321\217 \321\204\320\260\320\271\320\273\320\260"..., 48Ведите  
имя файла для child2: ) = 48  
read(0, f2.txt  
"f2.txt", 1024) = 7  
**pipe2**([3, 4], 0) = 0  
**pipe2**([5, 6], 0) = 0  
**clone**(child\_stack=NULL,  
flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLDstrace:  
Process 14142 attached  
, child\_tidptr=0xfffffa9cbffb0) = 14142  
[pid 14142] set\_robust\_list(0xfffffa9cbfffc0, 24 <unfinished ...>

[pid 13875] **clone**(child\_stack=NULL,  
flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLD <unfinished  
...>

[pid 14142] <... set\_robust\_list resumed>) = 0

[pid 14142] **close**(4<strace: Process 14143 attached  
) = 0

[pid 13875] <... clone resumed>, child\_tidptr=0xfffffa9cbffb0) = 14143

[pid 14143] set\_robust\_list(0xfffffa9cbfffc0, 24 <unfinished ...>

[pid 14142] **close**(5 <unfinished ...>

[pid 13875] **close**(3 <unfinished ...>

[pid 14143] <... set\_robust\_list resumed>) = 0

[pid 14142] <... close resumed>) = 0

[pid 13875] <... close resumed>) = 0

[pid 14143] **close**(6 <unfinished ...>

[pid 14142] **close**(6 <unfinished ...>

[pid 13875] **close**(5 <unfinished ...>

[pid 14143] <... close resumed>) = 0

[pid 14142] <... close resumed>) = 0

[pid 13875] <... close resumed>) = 0

[pid 14143] **close**(3 <unfinished ...>

[pid 14142] **dup3**(3, 0, 0 <unfinished ...>

[pid 14143] <... close resumed>) = 0

[pid 14142] <... dup3 resumed>) = 0

[pid 13875] write(1, "", 1 <unfinished ...>

[pid 14143] **close**(4 <unfinished ...>

[pid 14142] **close**(3  
<unfinished ...>

[pid 13875] <... write resumed>) = 1

[pid 14143] <... close resumed>) = 0

[pid 14142] <... close resumed>) = 0

[pid 14142] **execve**("./server", ["server", "f1.txt"], 0xfffff3f252d8 /\* 39 vars \*/  
<unfinished ...>

[pid 13875] write(1, "\320\222\320\262\320\276\320\264\320\270\321\202\320\265\321\201\321\202\321\200\320\276\320\272\320\270 (\320\242\320"..., 110 <unfinished ...>

[pid 14143] **dup3**(5, 0, 0) Вводите строки (Только латинские буквы. Ctrl+D для завершения):

<unfinished ...>

[pid 13875] <... write resumed>) = 110

[pid 14143] <... dup3 resumed>) = 0

[pid 14143] **close**(5 <unfinished ...>

[pid 13875] read(0, <unfinished ...>

[pid 14143] <... close resumed>) = 0

[pid 14143] **execve**("./server", ["server", "f2.txt"], 0xfffff3f252d8 /\* 39 vars \*/) = 0

[pid 14142] <... execve resumed>) = 0

[pid 14143] brk(NULL <unfinished ...>

[pid 14142] brk(NULL <unfinished ...>

[pid 14143] <... brk resumed>) = 0xaaab160c8000

[pid 14142] <... brk resumed>) = 0xaaaaf494f000

[pid 14143] mmap(NULL, 8192, PROT\_READ|PROT\_WRITE,  
MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 14142] mmap(NULL, 8192, PROT\_READ|PROT\_WRITE,  
MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 14143] <... mmap resumed>) = 0xfffffb76c4000

[pid 14142] <... mmap resumed>) = 0xfffffb1877000

[pid 14143] faccessat(AT\_FDCWD, "/etc/ld.so.preload", R\_OK) = -1 ENOENT (No such file or directory)

[pid 14143] openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) =

[pid 14143] mmap(NULL, 21571, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0xfffffb76be000

[pid 14143] close(3) = 0

[pid 14142] faccessat(AT\_FDCWD, "/etc/ld.so.preload", R\_OK) <unfinished ...>

[pid 14143] openat(AT\_FDCWD, "/lib/aarch64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC <unfinished ...>

[pid 14142] <... faccessat resumed>) = -1 ENOENT (No such file or directory)

[pid 14143] <... openat resumed>) = 3

[pid 14143] read(3, <unfinished ...>

[pid 14142] openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC  
<unfinished ...>

[pid 14143]<... read

resumed>“\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\0\0\3\0\267\0\1\0\0\0\360\206\2\0\0\0\0\0”..., 832) = 832

[pid 14142]<... openat resumed>) = 3

[pid 14142] fstat(3, <unfinished ...>

```
[pid 14143] fstat(3, {st_mode=S_IFREG|0755, st_size=1722920, ...})=0
```

[pid 14142] <... fstat resumed>{st\_mode=S\_IFREG|0644, st\_size=21571, ...}) = 0

```
[pid 14143] mmap(NULL, 1892240, PROT_NONE,  
MAP_PRIVATE|MAP_ANONYMOUS|MAP_DENYWRITE, -1, 0 <unfinished ...>
```

[pid 14142] mmap(NULL, 21571, PROT\_READ, MAP\_PRIVATE, 3, 0  
<unfinished ...>

[pid 14143] <... mmap resumed>) = 0xfffffb74be000

[pid 14142] <... mmap resumed>) = 0xfffffb1871000

[pid 14143] mmap(0xffffb74c0000, 1826704, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0 <unfinished ...>

[pid 14142] close(3) = 0

[pid 14143] <... mmap resumed>) = 0xfffffb74c0000

[pid 14143] munmap(0xfffffb74be000, 8192 <unfinished ...>

[pid 14142] openat(AT\_FDCWD, "/lib/aarch64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC <unfinished ...>



[pid 14143] <... set\_tid\_address resumed>) = 14143

[pid 14142] munmap(0xfffffb183e000, 3984 <unfinished ...>

[pid 14143] set\_robust\_list(0xfffffb76c4fc0, 24 <unfinished ...>

[pid 14142] <... munmap resumed>) = 0

[pid 14143] <... set\_robust\_list resumed>) = 0

[pid 14142] mprotect(0xfffffb1819000, 81920, PROT\_NONE <unfinished ...>

[pid 14143] rseq(0xfffffb76c5600, 0x20, 0, 0xd428bc00 <unfinished ...>

[pid 14142] <... mprotect resumed>) = 0

[pid 14143] <... rseq resumed>) = 0

[pid 14142] mmap(0xfffffb182d000, 20480, PROT\_READ|PROT\_WRITE,  
MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x19d000) = 0xfffffb182d000

[pid 14143] mprotect(0xfffffb766d000, 12288, PROT\_READ <unfinished ...>

[pid 14142] mmap(0xfffffb1832000, 49040, PROT\_READ|PROT\_WRITE,  
MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 14143] <... mprotect resumed>) = 0

[pid 14143] mprotect(0xaaadce0f000, 4096, PROT\_READ <unfinished ...>

[pid 14142] <... mmap resumed>) = 0xfffffb1832000

[pid 14143] <... mprotect resumed>) = 0

[pid 14142] close(3 <unfinished ...>

[pid 14143] mprotect(0xfffffb76ca000, 8192, PROT\_READ <unfinished ...>

[pid 14142] <... close resumed>) = 0

[pid 14143] <... mprotect resumed>) = 0

[pid 14143] prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024,  
rlim\_max=RLIM64\_INFINITY}) = 0

[pid 14143] munmap(0xfffffb76be000, 21571 <unfinished ...>

[pid 14142] set\_tid\_address(0xfffffb1877fb0) = 14142

[pid 14143] <... munmap resumed>) = 0

[pid 14142] set\_robust\_list(0xfffffb1877fc0, 24 <unfinished ...>

[pid 14143] **openat**(AT\_FDCWD, “f2.txt”, O\_WRONLY|O\_CREAT|O\_TRUNC, 0644 <unfinished ...>

[pid 14142] <... set\_robust\_list resumed>) = 0

[pid 14142] rseq(0xfffffb1878600, 0x20, 0, 0xd428bc00) = 0

[pid 14143] <... openat resumed>) = 3

[pid 14142] mprotect(0xfffffb182d000, 12288, PROT\_READ <unfinished ...>

[pid 14143] **getpid**(<unfinished ...>

[pid 14142] <... mprotect resumed>) = 0

[pid 14143] <... getpid resumed>) = 14143

[pid 14142] mprotect(0xaaad7f1f000, 4096, PROT\_READ <unfinished ...>

[pid 14143] fstat(1, <unfinished ...>

[pid 14142] <... mprotect resumed>) = 0

[pid 14143] <... fstat resumed>{st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0), ...}) = 0

[pid 14142] mprotect(0xfffffb187d000, 8192, PROT\_READ <unfinished ...>

[pid 14143] getrandom(<unfinished ...>

[pid 14142] <... mprotect resumed>) = 0

[pid 14143] <... getrandom resumed>“4a4f”, 8, GRND\_NONBLOCK) = 8

[pid 14142] prlimit64(0, RLIMIT\_STACK, NULL, <unfinished ...>

[pid 14143] brk(NULL <unfinished ...>

[pid 14142] <... prlimit64 resumed>{rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

[pid 14143] <... brk resumed>) = 0xaab160c8000

[pid 14142] munmap(0xfffffb1871000, 21571) = 0

[pid 14143] brk(0xaab160e9000 <unfinished ...>

[pid 14142] **openat**(AT\_FDCWD, “f1.txt”, O\_WRONLY|O\_CREAT|O\_TRUNC, 0644 <unfinished ...>

[pid 14143] <... brk resumed>) = 0xaab160e9000

[pid 14143] **write**(1, “[Server PID=14143]\320\235\320\260\321\207\320\260\320\273 \321\200”..., 61[Server PID=14143] Начал работу. Файл: f2.txt

) = 61

[pid 14142] <... openat resumed>) = 3

[pid 14143] **read**(0, <unfinished ...>

[pid 14142] **getpid**() = 14142

[pid 14142] fstat(1, {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0), ...}) = 0

[pid 14142] getrandom(“1a3a7c”, 8, GRND\_NONBLOCK) = 8

[pid 14142] brk(NULL) = 0xaaaaf494f000

[pid 14142] brk(0xaaaaf4970000) = 0xaaaaf4970000

[pid 14142] **write**(1, “[Server PID=14142]\320\235\320\260\321\207\320\260\320\273 \321\200”..., 61[Server PID=14142] Начал

работу. Файл: f1.txt

) = 61

[pid 14142] **read**(0, hello

<unfinished ...>

[pid 13875] <... read resumed>“h”, 1) = 1

[pid 13875] **read**(0, “e”, 1) = 1

[pid 13875] **read**(0, “l”, 1) = 1

[pid 13875] **read**(0, “l”, 1) = 1

[pid 13875] **read**(0, “o”, 1) = 1

[pid 13875] **read**(0, “”, 1) = 1

[pid 13875] **write**(4, “hello”, 5) = 5

[pid 14142] <... read resumed>“hello”, 4095) = 5

[pid 13875] **write**(4, “”, 1 <unfinished ...>

[pid 14142] **write**(3, “[Server PID=14142]\320\230\321\201\321\205\320\276\320\264\320\275\320”..., 80 <unfinished ...>

\320\230\321\201\321\205\320\276\320\264\320\275\320”..., 80 <unfinished ...>

[pid 13875] <... write resumed>) = 1

[pid 13875] **write**(1,  
“[\320\240\320\276\320\264\320\270\321\202\320\265\320\273\321\214]  
\320\241\321\202\321\200\320\276\320\272\320\260”..., 72 <unfinished ...>

[pid 14142] <... write resumed>) = 80

[Родитель] Стока 1 отправлена в child1: hello

[pid 13875] <... write resumed>) = 72

[pid 14142] **write**(1, “[Server PID=14142]  
\320\230\321\201\321\205\320\276\320\264\320\275\320”..., 80[Server PID=14142]  
Исходная: ‘hello’ -> Развернутая: ‘olleh’

<unfinished ...>

[pid 13875] **read**(0, <unfinished ...>

[pid 14142] <... write resumed>) = 80

[pid 14142] **read**(0, “”, 4095) = 1

[pid 14142] **read**(0, hi

<unfinished ...>

[pid 13875] <... read resumed>“h”, 1) = 1

[pid 13875] **read**(0, “i”, 1) = 1

[pid 13875] **read**(0, “”, 1) = 1

[pid 13875] **write**(6, “hi”, 2) = 2

[pid 14143] <... read resumed>“hi”, 4095) = 2

[pid 13875] **write**(6, “”, 1 <unfinished ...>

[pid 14143] **write**(3, “[Server PID=14143]  
\320\230\321\201\321\205\320\276\320\264\320\275\320”..., 74 <unfinished ...>

[pid 13875] <... write resumed>) = 1

[pid 13875] **write**(1,  
“[\320\240\320\276\320\264\320\270\321\202\320\265\320\273\321\214]  
\320\241\321\202\321\200\320\276\320\272\320\260”..., 69[Родитель] Стока 2  
отправлена в child2: hi

) = 69

[pid 13875] **read**(0, <unfinished ...>

[pid 14143] <... write resumed>) = 74

[pid 14143] **write**(1, “[Server PID=14143]  
\\320\\230\\321\\201\\321\\205\\320\\276\\320\\264\\320\\275\\320”..., 74[Server PID=14143]  
Исходная: ‘hi’ -> Развернутая: ‘ih’

) = 74

[pid 14143] **read**(0, “”, 4095) = 1

[pid 14143] **read**(0, bye

<unfinished ...>

[pid 13875] <... read resumed>“b”, 1) = 1

[pid 13875] **read**(0, “y”, 1) = 1

[pid 13875] **read**(0, “e”, 1) = 1

[pid 13875] **read**(0, “”, 1) = 1

[pid 13875] **write**(4, “bye”, 3) = 3

[pid 14142] <... read resumed>“bye”, 4095) = 3

[pid 13875] **write**(4, “”, 1 <unfinished ...>

[pid 14142] **write**(3, “[Server PID=14142]

\\320\\230\\321\\201\\321\\205\\320\\276\\320\\264\\320\\275\\320”..., 76 <unfinished ...>

[pid 13875] <... write resumed>) = 1

[pid 13875] **write**(1,

“[\\320\\240\\320\\276\\320\\264\\320\\270\\321\\202\\320\\265\\320\\273\\321\\214]

\\320\\241\\321\\202\\321\\200\\320\\276\\320\\272\\320\\260”..., 70[Родитель] Стока 3  
отправлена в child1: bye

) = 70

[pid 13875] **read**(0, <unfinished ...>

[pid 14142] <... write resumed>) = 76

[pid 14142] **write**(1, “[Server PID=14142]

\\320\\230\\321\\201\\321\\205\\320\\276\\320\\264\\320\\275\\320”..., 76[Server PID=14142]

Исходная: ‘bye’ -> Развернутая: ‘eyb’

) = 76

[pid 14142] **read**(0, “”, 4095) = 1

[pid 14142] **read**(0, bye-bye

<unfinished ...>

[pid 13875] <... read resumed>“b”, 1) = 1  
[pid 13875] **read**(0, “y”, 1) = 1  
[pid 13875] **read**(0, “e”, 1) = 1  
[pid 13875] **read**(0, “-”, 1) = 1  
[pid 13875] **read**(0, “b”, 1) = 1  
[pid 13875] **read**(0, “y”, 1) = 1  
[pid 13875] **read**(0, “e”, 1) = 1  
[pid 13875] **read**(0, “”, 1) = 1  
[pid 13875] **write**(6, “bye-bye”, 7) = 7  
[pid 14143] <... read resumed>“bye-bye”, 4095) = 7  
[pid 13875] **write**(6, “”, 1 <unfinished ...>  
[pid 14143] **write**(3, “[Server PID=14143]  
\320\230\321\201\321\205\320\276\320\264\320\275\320”..., 84 <unfinished ...>  
[pid 13875] <... write resumed>) = 1  
[pid 13875] **write**(1,  
“[\320\240\320\276\320\264\320\270\321\202\320\265\320\273\321\214]  
\320\241\321\202\321\200\320\276\320\272\320\260”..., 74 [Родитель] Стока 4  
отправлена в child2: bye-bye  
) = 74  
[pid 13875] **read**(0, <unfinished ...>  
[pid 14143] <... write resumed>) = 84  
[pid 14143] **write**(1, “[Server PID=14143]  
\320\230\321\201\321\205\320\276\320\264\320\275\320”..., 84 [Server PID=14143]  
Исходная: ‘bye-bye’ -> Развернутая: ‘eyb-eyb’  
) = 84  
[pid 14143] **read**(0, “”, 4095) = 1  
[pid 14143] **read**(0, no  
<unfinished ...>  
[pid 13875] <... read resumed>“n”, 1) = 1  
[pid 13875] **read**(0, “o”, 1) = 1

[pid 13875] **read**(0, "", 1) = 1  
[pid 13875] **write**(4, "no", 2) = 2  
[pid 14142] <... read resumed> "no", 4095) = 2  
[pid 13875] **write**(4, "", 1 <unfinished ...>  
[pid 14142] **write**(3, "[Server PID=14142]  
\320\230\321\201\321\205\320\276\320\264\320\275\320"..., 74 <unfinished ...>  
[pid 13875] <... write resumed>) = 1  
[pid 13875] **write**(1,  
"\320\240\320\276\320\264\320\270\321\202\320\265\320\273\321\214]  
\320\241\321\202\321\200\320\276\320\272\320\260"..., 69 [Родитель] Стока 5  
отправлена в child1: no  
) = 69  
[pid 13875] **read**(0, <unfinished ...>  
[pid 14142] <... write resumed>) = 74  
[pid 14142] **write**(1, "[Server PID=14142]  
\320\230\321\201\321\205\320\276\320\264\320\275\320"..., 74 [Server PID=14142]  
Исходная: 'no' -> Развернутая: 'on'  
) = 74  
[pid 14142] **read**(0, "", 4095) = 1  
[pid 14142] **read**(0, <unfinished ...>  
[pid 13875] <... read resumed> "", 1) = 0  
[pid 13875] **close**(4) = 0  
[pid 14142] <... read resumed> "", 4095) = 0  
[pid 13875] **close**(6 <unfinished ...>  
[pid 14142] **close**(3 <unfinished ...>  
[pid 13875] <... close resumed>) = 0  
[pid 14143] <... read resumed> "", 4095) = 0  
[pid 14142] <... close resumed>) = 0  
[pid 13875] **wait4**(14142, <unfinished ...>  
[pid 14143] **close**(3 <unfinished ...>

[pid 14142] **write**(1, “[Server PID=14142]\320\227\320\260\320\262\320\265\321\200\321\210\320”..., 50[Server PID=14142]  
Завершил работу.

<unfinished ...>

[pid 14143] <... close resumed>) = 0

[pid 14142] <... write resumed>) = 50

[pid 14143] **write**(1, “[Server PID=14143]\320\227\320\260\320\262\320\265\321\200\321\210\320”..., 50[Server PID=14143]  
Завершил работу.

) = 50

[pid 14142] **exit\_group**(0 <unfinished ...>

[pid 14143] **exit\_group**(0 <unfinished ...>

[pid 14142] <... exit\_group resumed>) = ?

[pid 14143] <... exit\_group resumed>) = ?

[pid 14143] +++ exited with 0 +++

[pid 14142] +++ exited with 0 +++

<... wait4 resumed>NULL, 0, NULL) = 14142

– SIGCHLD {si\_signo=SIGCHLD, si\_code=CLD\_EXITED, si\_pid=14143,  
si\_uid=1000, si\_status=0, si\_utime=0, si\_stime=0} –

**wait4**(14143, NULL, 0, NULL) = 14143

**write**(1,

“\320\240\320\276\320\264\320\270\321\202\320\265\320\273\321\214\321\201\320\272  
\320\270\320\271 \320\277\321\200\320\276\321”..., 58Родительский процесс  
завершен.

) = 58

**exit\_group**(0) = ?

+++ exited with 0 +++

## **Вывод**

В ходе выполнения лабораторной работы была разработана программа на языке С, демонстрирующая работу с процессами и их взаимодействие в среде Linux. Для решения поставленной задачи создаются несколько дочерних процессов, обмен данными между которыми осуществляется через каналы (pipe). В программе предусмотрена обработка возможных системных ошибок.