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: Лоскутова А. Д.

Преподаватель: Бахарев В.Д.

Оценка: _____

Дата: 01.10.25

Москва, 2025

Постановка задачи

Вариант 17.

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: строки длины больше 10 символов отправляются в pipe2, иначе в pipe1. 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); – создает неименованный однонаправленный 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 длине, ввод пользователя разделяется на два параллельных потока.

    - Короткая строка (≤10): Родитель записывает ее в pipe1[1]
    - Длинная строка (>10): Родитель записывает ее в pipe2[1]

3. Каждый процесс server читает данные со своего STDIN. На каждой строке вызывается функция, которая удаляет все гласные. 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 <errno.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 && read(fd, &c, 1) == 1) {
        if (c == '\n') {
            break;
        }
        buffer[total_read++] = c;
    }
    if (total_read == 0 && c != '\n'){
         return -1;
    }
    buffer[total_read] = '\0';
    return total_read;
}

int main() {
    char filename1[256], filename2[256];
```

```
write(STDOUT_FILENO, "Enter the file name for child1: ", 32);//получения названия 1txt
ssize_t r1 = read(STDIN_FILENO, filename1, sizeof(filename1) - 1);
if (r1 <= 0){
     exit(EXIT_FAILURE);
}
if (filename1[r1 - 1] == '\n') {
    filename1[r1 - 1] = '\0';
}
write(STDOUT_FILENO, "Enter the file name for child2: ", 32);//получения названия 2txt
ssize_t r2 = read(STDIN_FILENO, filename2, sizeof(filename2) - 1);
if (r2 <= 0) {
    exit(EXIT_FAILURE);
}
if (filename2[r2 - 1] == '\n'){
    filename2[r2 - 1] = '\0';
}

int pipe1[2], pipe2[2];//создали каналы
if (pipe(pipe1) == -1 || pipe(pipe2) == -1) {
    perror("pipe creation failed");
    exit(EXIT_FAILURE);
}
pid_t pid1 = fork();//процесс для первого
if (pid1 == -1) {
    perror("fork child1");
    exit(EXIT_FAILURE);
}
if (pid1 == 0) {
    close(pipe1[1]);
    dup2(pipe1[0], STDIN_FILENO);
    close(pipe1[0]);
```

```
    close(pipe2[0]);
    close(pipe2[1]);

    char *args[] = {"./server", filename1, NULL};
    execv("./server", args);
    perror("execv child1 failed");
    exit(EXIT_FAILURE);
}
pid_t pid2 = fork();//процесс для второго
if (pid2 == -1) {
     perror("fork child2");
     exit(EXIT_FAILURE);
}
if (pid2 == 0) {
    close(pipe2[1]);
    dup2(pipe2[0], STDIN_FILENO);
    close(pipe2[0]);
    close(pipe1[0]);
    close(pipe1[1]);
    char *args[] = {"./server", filename2, NULL};
    execv("./server", args);
    perror("execv child2 failed");
    exit(EXIT_FAILURE);
}
close(pipe1[0]);
close(pipe2[0]);
int wfd1 = pipe1[1];
int wfd2 = pipe2[1];
char line[MAX_LINE_LEN];
ssize_t len;
write(STDOUT_FILENO, "Enter the lines:\n", 17);
while ((len = read_line(STDIN_FILENO, line, sizeof(line))) != -1) {
```

```
        if (len == 0){
            break;
        }
        int target_fd;
        if (len > 10) { //фильтрация
            target_fd = wfd2;
        } else {
            target_fd = wfd1;
        }
        if (write(target_fd, line, len) == -1 || write(target_fd, "\n", 1) == -1) {
            perror("write to pipe failed");
            break;
        }
    }
    close(wfd1);
    close(wfd2);
    waitpid(pid1, NULL, 0);
    waitpid(pid2, NULL, 0);
    write(STDOUT_FILENO, "Parent: all children finished.\n", 31);
    return 0;
}
```

server.c

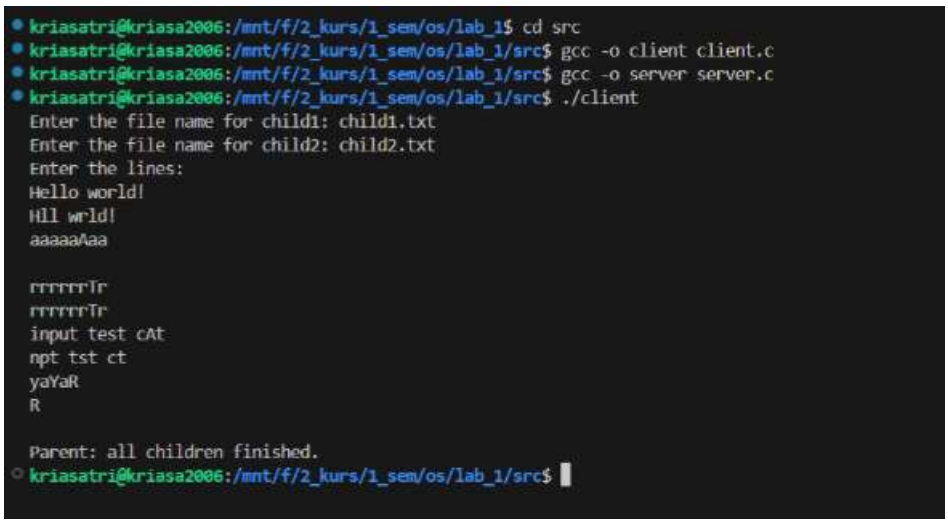
```
#include <unistd.h>
#include <fcntl.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <stdio.h>
#define MAX_LEN 4096
void remove_vowels(char *s)
{
    char *src = s, *dst = s;
```

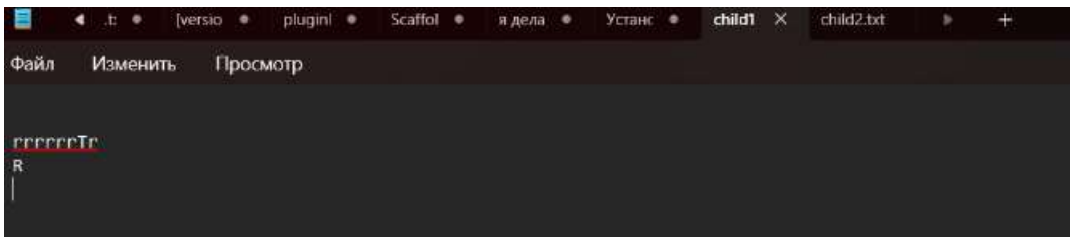
```
    while (*src) {
        char c = *src;
        if (!(c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u' || c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U'
|| c == 'y' || c == 'Y')) {
            *dst++ = c;
        }
        src++;
    }
    *dst = '\0';
}
int main(int argc, char **argv)
{
    if (argc != 2) {
        const char msg[] = "usage: server filename\n";
        write(STDERR_FILENO, msg, sizeof(msg) - 1);
        exit(EXIT_FAILURE);
    }
    int fd = open(argv[1], O_WRONLY | O_CREAT | O_TRUNC, 0644);
    if (fd == -1) {
        perror("open file failed");
        exit(EXIT_FAILURE);
    }
    char buf[MAX_LEN];
    ssize_t n;
    while ((n = read(STDIN_FILENO, buf, sizeof(buf) - 1)) > 0) {

        if (buf[n - 1] == '\n') {
            buf[n - 1] = '\0';
        } else {
            buf[n] = '\0';
        }
        remove_vowels(buf);
        size_t len = strlen(buf);
```

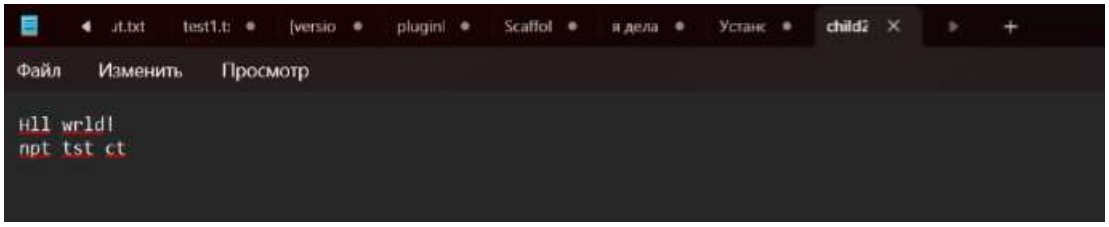
```
        if (write(fd, buf, len) == -1 || write(fd, "\n", 1) == -1) {
            perror("write file failed");
        }

        if (write(STDOUT_FILENO, buf, len) == -1 || write(STDOUT_FILENO, "\n", 1) == -1) {
            perror("write stdout failed");
        }
    }
    if (n == -1) {
        perror("read stdin failed");
    }
    close(fd);
    return 0;
}
```

Протокол работы программы

Тесты







Strace

kriasatri@kriasa2006:/mnt/f/2_kurs/1_sem/os/lab_1/src$ strace -f ./client

execve("./client", ["./client"], 0x7ffc432d7b68 /* 36 vars */) = 0

brk(NULL) = 0x5c6ec34d7000

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7ad4aca1b000

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

fstat(3, {st_mode=S_IFREG|0644, st_size=19711, ...}) = 0

mmap(NULL, 19711, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7ad4aca16000

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\220\243\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

fstat(3, {st_mode=S_IFREG|0755, st_size=2125328, ...}) = 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, 2170256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7ad4ac800000

mmap(0x7ad4ac828000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7ad4ac828000

mmap(0x7ad4ac9b0000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b0000) = 0x7ad4ac9b0000

mmap(0x7ad4ac9ff000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1fe000) = 0x7ad4ac9ff000

mmap(0x7ad4aca05000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7ad4aca05000

close(3) = 0

mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7ad4aca13000

arch_prctl(ARCH_SET_FS, 0x7ad4aca13740) = 0

set_tid_address(0x7ad4aca13a10) = 52632

set_robust_list(0x7ad4aca13a20, 24)   = 0

rseq(0x7ad4aca14060, 0x20, 0, 0x53053053) = 0

mprotect(0x7ad4ac9ff000, 16384, PROT_READ) = 0

mprotect(0x5c6e87717000, 4096, PROT_READ) = 0

mprotect(0x7ad4aca53000, 8192, PROT_READ) = 0

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

munmap(0x7ad4aca16000, 19711)   = 0

write(1, "Enter the file name for child1: ", 32Enter the file name for child1:) = 32

read(0, child1.txt

"child1.txt\n", 255)   = 11

write(1, "Enter the file name for child2: ", 32Enter the file name for child2:) = 32

read(0, child2.txt

"child2.txt\n", 255)   = 11

pipe2([3, 4], 0)   = 0

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

clone(child_stack=NULL,
flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLDstrace: Process 52726 attached

, child_tidptr=0x7ad4aca13a10) = 52726

[pid 52726] set_robust_list(0x7ad4aca13a20, 24) = 0

[pid 52632] clone(child_stack=NULL,
flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD <unfinished ...>

[pid 52726] close(4)   = 0

[pid 52726] dup2(3, 0)   = 0

[pid 52726] close(3strace: Process 52727 attached

)   = 0

[pid 52632] <... clone resumed>, child_tidptr=0x7ad4aca13a10) = 52727

[pid 52727] set_robust_list(0x7ad4aca13a20, 24 <unfinished ...>

[pid 52632] close(3 <unfinished ...>

[pid 52726] close(5 <unfinished ...>

[pid 52632] <... close resumed>)   = 0

[pid 52727] <... set_robust_list resumed>) = 0

[pid 52632] close(5 <unfinished ...>

[pid 52726] <... close resumed>)   = 0

[pid 52632] <... close resumed>) = 0

[pid 52727] close(6 <unfinished ...>

[pid 52632] write(1, "Enter the lines:\n", 17 <unfinished ...>

Enter the lines:

[pid 52726] close(6 <unfinished ...>

[pid 52632] <... write resumed>) = 17

[pid 52727] <... close resumed>) = 0

[pid 52632] read(0, <unfinished ...>

[pid 52726] <... close resumed>) = 0

[pid 52727] dup2(5, 0 <unfinished ...>

[pid 52726] execve("./server", ["./server", "child1.txt"], 0x7ffe57399c58 /* 36 vars */ <unfinished ...>

[pid 52727] <... dup2 resumed>) = 0

[pid 52727] close(5) = 0

[pid 52727] close(3) = 0

[pid 52727] close(4) = 0

[pid 52727] execve("./server", ["./server", "child2.txt"], 0x7ffe57399c58 /* 36 vars */) = 0

[pid 52726] <... execve resumed>) = 0

[pid 52727] brk(NULL <unfinished ...>

[pid 52726] brk(NULL <unfinished ...>

[pid 52727] <... brk resumed>) = 0x624926023000

[pid 52726] <... brk resumed>) = 0x589376eb4000

[pid 52727] mmap(NULL, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 52726] mmap(NULL, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 52727] <... mmap resumed>) = 0x72b281063000

[pid 52727] access("/etc/ld.so.preload", R_OK <unfinished ...>

[pid 52726] <... mmap resumed>) = 0x7b2ce50bf000

[pid 52727] <... access resumed>) = -1 ENOENT (No such file or directory)

[pid 52726] access("/etc/ld.so.preload", R_OK <unfinished ...>

[pid 52727] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>

[pid 52726] <... access resumed>) = -1 ENOENT (No such file or directory)

[pid 52727] <... openat resumed>) = 3

[pid 52727] fstat(3, <unfinished ...>

[pid 52726] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>

[pid 52727] <... fstat resumed>{st_mode=S_IFREG|0644, st_size=19711, ...}) = 0

[pid 52726] <... openat resumed>)   = 3

[pid 52727] mmap(NULL, 19711, PROT_READ, MAP_PRIVATE, 3, 0 <unfinished ...>

[pid 52726] fstat(3, <unfinished ...>

[pid 52727] <... mmap resumed>)   = 0x72b28105e000

[pid 52726] <... fstat resumed>{st_mode=S_IFREG|0644, st_size=19711, ...}) = 0

[pid 52727] close(3 <unfinished ...>

[pid 52726] mmap(NULL, 19711, PROT_READ, MAP_PRIVATE, 3, 0 <unfinished ...>

[pid 52727] <... close resumed>)   = 0

[pid 52726] <... mmap resumed>)   = 0x7b2ce50ba000

[pid 52727] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC
<unfinished ...>

[pid 52726] close(3 <unfinished ...>

[pid 52727] <... openat resumed>)   = 3

[pid 52726] <... close resumed>)   = 0

[pid 52727] read(3, <unfinished ...>

[pid 52726] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC
<unfinished ...>

[pid 52727] <... read resumed>"\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\243\2\0\0\0\0\0"...,
832) = 832

[pid 52726] <... openat resumed>)   = 3

[pid 52727] pread64(3, <unfinished ...>

[pid 52726] read(3, <unfinished ...>

[pid 52727] <... 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 52726] <... read resumed>"\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\243\2\0\0\0\0\0"...,
832) = 832

[pid 52727] fstat(3, <unfinished ...>

[pid 52726] pread64(3, <unfinished ...>

[pid 52727] <... fstat resumed>{st_mode=S_IFREG|0755, st_size=2125328, ...}) = 0

[pid 52726] <... 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 52727] pread64(3, <unfinished ...>

[pid 52726] fstat(3, <unfinished ...>

[pid 52727] <... 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 52726] <... fstat resumed>{st_mode=S_IFREG|0755, st_size=2125328, ...}) = 0

[pid 52727] mmap(NULL, 2170256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0 <unfinished ...>

[pid 52726] pread64(3, <unfinished ...>

[pid 52727] <... mmap resumed>)   = 0x72b280e00000

[pid 52726] <... 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 52727] mmap(0x72b280e28000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000 <unfinished ...>

[pid 52726] mmap(NULL, 2170256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0 <unfinished ...>

[pid 52727] <... mmap resumed>)   = 0x72b280e28000

[pid 52726] <... mmap resumed>)   = 0x7b2ce4e00000

[pid 52727] mmap(0x72b280fb0000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b0000 <unfinished ...>

[pid 52726] mmap(0x7b2ce4e28000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000 <unfinished ...>

[pid 52727] <... mmap resumed>)   = 0x72b280fb0000

[pid 52726] <... mmap resumed>)   = 0x7b2ce4e28000

[pid 52727] mmap(0x72b280fff000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1fe000 <unfinished ...>

[pid 52726] mmap(0x7b2ce4fb0000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b0000 <unfinished ...>

[pid 52727] <... mmap resumed>)   = 0x72b280fff000

[pid 52726] <... mmap resumed>)   = 0x7b2ce4fb0000

[pid 52727] mmap(0x72b281005000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 52726] mmap(0x7b2ce4fff000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1fe000 <unfinished ...>

[pid 52727] <... mmap resumed>)   = 0x72b281005000

[pid 52726] <... mmap resumed>)   = 0x7b2ce4fff000

[pid 52727] close(3 <unfinished ...>

[pid 52726] mmap(0x7b2ce5005000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 52727] <... close resumed>)   = 0

[pid 52726] <... mmap resumed>)   = 0x7b2ce5005000

[pid 52727] mmap(NULL, 12288, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 52726] close(3 <unfinished ...>

[pid 52727] <... mmap resumed>)   = 0x72b28105b000

[pid 52726] <... close resumed>)   = 0

[pid 52727] arch_prctl(ARCH_SET_FS, 0x72b28105b740) = 0

[pid 52726] mmap(NULL, 12288, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 52727] set_tid_address(0x72b28105ba10 <unfinished ...>

[pid 52726] <... mmap resumed>)   = 0x7b2ce50b7000

[pid 52727] <... set_tid_address resumed>) = 52727

[pid 52726] arch_prctl(ARCH_SET_FS, 0x7b2ce50b7740 <unfinished ...>

[pid 52727] set_robust_list(0x72b28105ba20, 24 <unfinished ...>

[pid 52726] <... arch_prctl resumed>)   = 0

[pid 52727] <... set_robust_list resumed>) = 0

[pid 52726] set_tid_address(0x7b2ce50b7a10 <unfinished ...>

[pid 52727] rseq(0x72b28105c060, 0x20, 0, 0x53053053 <unfinished ...>

[pid 52726] <... set_tid_address resumed>) = 52726

[pid 52727] <... rseq resumed>)   = 0

[pid 52726] set_robust_list(0x7b2ce50b7a20, 24) = 0

[pid 52727] mprotect(0x72b280fff000, 16384, PROT_READ <unfinished ...>

[pid 52726] rseq(0x7b2ce50b8060, 0x20, 0, 0x53053053 <unfinished ...>

[pid 52727] <... mprotect resumed>)   = 0

[pid 52726] <... rseq resumed>)   = 0

[pid 52727] mprotect(0x6248f4ed0000, 4096, PROT_READ) = 0

[pid 52726] mprotect(0x7b2ce4fff000, 16384, PROT_READ <unfinished ...>

[pid 52727] mprotect(0x72b28109b000, 8192, PROT_READ <unfinished ...>

[pid 52726] <... mprotect resumed>)   = 0

[pid 52727] <... mprotect resumed>)   = 0

[pid 52726] mprotect(0x589366132000, 4096, PROT_READ <unfinished ...>

[pid 52727] prlimit64(0, RLIMIT_STACK, NULL, <unfinished ...>

[pid 52726] <... mprotect resumed>)   = 0

[pid 52727] <... prlimit64 resumed>{rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0

[pid 52726] mprotect(0x7b2ce50f7000, 8192, PROT_READ <unfinished ...>

[pid 52727] munmap(0x72b28105e000, 19711 <unfinished ...>

[pid 52726] <... mprotect resumed>)   = 0

[pid 52727] <... munmap resumed>)   = 0

[pid 52726] prlimit64(0, RLIMIT_STACK, NULL, <unfinished ...>

[pid 52727] openat(AT_FDCWD, "child2.txt", O_WRONLY|O_CREAT|O_TRUNC, 0644 <unfinished ...>

[pid 52726] <... prlimit64 resumed>{rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0

[pid 52726] munmap(0x7b2ce50ba000, 19711) = 0

[pid 52726] openat(AT_FDCWD, "child1.txt", O_WRONLY|O_CREAT|O_TRUNC, 0644 <unfinished ...>

[pid 52727] <... openat resumed>)   = 3

[pid 52727] read(0, <unfinished ...>

[pid 52726] <... openat resumed>)   = 3

[pid 52726] read(0, Hello world1

<unfinished ...>

[pid 52632] <... read resumed>"H", 1)   = 1

[pid 52632] read(0, "e", 1)   = 1

[pid 52632] read(0, "l", 1)   = 1

[pid 52632] read(0, "l", 1)   = 1

[pid 52632] read(0, "o", 1)   = 1

[pid 52632] read(0, " ", 1)   = 1

[pid 52632] read(0, "w", 1)   = 1

[pid 52632] read(0, "o", 1)   = 1

[pid 52632] read(0, "r", 1)   = 1

[pid 52632] read(0, "l", 1)   = 1

[pid 52632] read(0, "d", 1)   = 1

[pid 52632] read(0, "1", 1)   = 1

[pid 52632] read(0, "\n", 1)   = 1

[pid 52632] write(6, "Hello world1", 12) = 12

[pid 52727] <... read resumed>"Hello world1", 4095) = 12

[pid 52632] write(6, "\n", 1 <unfinished ...>

[pid 52727] write(3, "Hll wrld1", 9 <unfinished ...>

[pid 52632] <... write resumed>) = 1

[pid 52632] read(0, <unfinished ...>

[pid 52727] <... write resumed>) = 9

[pid 52727] write(3, "\n", 1) = 1

[pid 52727] write(1, "Hll wrld1", 9Hll wrld1) = 9

[pid 52727] write(1, "\n", 1

) = 1

[pid 52727] read(0, "\n", 4095) = 1

[pid 52727] write(3, "", 0) = 0

[pid 52727] write(3, "\n", 1) = 1

[pid 52727] write(1, "", 0) = 0

[pid 52727] write(1, "\n", 1

) = 1

[pid 52727] read(0, aaaAaAAA!

<unfinished ...>

[pid 52632] <... read resumed>"a", 1) = 1

[pid 52632] read(0, "a", 1) = 1

[pid 52632] read(0, "a", 1) = 1

[pid 52632] read(0, "A", 1) = 1

[pid 52632] read(0, "a", 1) = 1

[pid 52632] read(0, "A", 1) = 1

[pid 52632] read(0, "A", 1) = 1

[pid 52632] read(0, "A", 1) = 1

[pid 52632] read(0, "!", 1) = 1

[pid 52632] read(0, "\n", 1) = 1

[pid 52632] write(4, "aaaAaAAA!", 9) = 9

[pid 52726] <... read resumed>"aaaAaAAA!", 4095) = 9

[pid 52632] write(4, "\n", 1 <unfinished ...>

[pid 52726] write(3, "!", 1 <unfinished ...>

[pid 52632] <... write resumed>) = 1

[pid 52632] read(0, <unfinished ...>

[pid 52726] <... write resumed>) = 1

[pid 52726] write(3, "\n", 1) = 1

[pid 52726] write(1, "!", 1!) = 1

[pid 52726] write(1, "\n", 1

) = 1

[pid 52726] read(0, "\n", 4095) = 1

[pid 52726] write(3, "", 0) = 0

[pid 52726] write(3, "\n", 1) = 1

[pid 52726] write(1, "", 0) = 0

[pid 52726] write(1, "\n", 1

) = 1

[**pid 52726] read(0, input test cat**

<unfinished ...>

[pid 52632] <... read resumed>"i", 1) = 1

[pid 52632] read(0, "n", 1) = 1

[pid 52632] read(0, "p", 1) = 1

[pid 52632] read(0, "u", 1) = 1

[pid 52632] read(0, "t", 1) = 1

[pid 52632] read(0, " ", 1) = 1

[pid 52632] read(0, "t", 1) = 1

[pid 52632] read(0, "e", 1) = 1

[pid 52632] read(0, "s", 1) = 1

[pid 52632] read(0, "t", 1) = 1

[pid 52632] read(0, " ", 1) = 1

[pid 52632] read(0, "c", 1) = 1

[pid 52632] read(0, "a", 1) = 1

[pid 52632] read(0, "t", 1) = 1

[pid 52632] read(0, "\n", 1) = 1

[pid 52632] write(6, "input test cat", 14) = 14

[pid 52727] <... read resumed>"input test cat", 4095) = 14

[pid 52632] write(6, "\n", 1 <unfinished ...>

[pid 52727] write(3, "npt tst ct", 10 <unfinished ...>

[pid 52632] <... write resumed>) = 1

[pid 52632] read(0, <unfinished ...>

[pid 52727] <... write resumed>) = 10

[pid 52727] write(3, "\n", 1) = 1

[pid 52727] write(1, "npt tst ct", 10npt tst ct) = 10

[pid 52727] write(1, "\n", 1

) = 1

[pid 52727] read(0, "\n", 4095) = 1

[pid 52727] write(3, "", 0) = 0

[pid 52727] write(3, "\n", 1) = 1

[pid 52727] write(1, "", 0) = 0

[pid 52727] write(1, "\n", 1

) = 1

[pid 52727] read(0,

<unfinished ...>

[pid 52632] <... read resumed>"\n", 1) = 1

[pid 52632] close(4) = 0

[pid 52726] <... read resumed>"", 4095) = 0

[pid 52632] close(6 <unfinished ...>

[pid 52726] close(3 <unfinished ...>

[pid 52632] <... close resumed>) = 0

[pid 52727] <... read resumed>"", 4095) = 0

[pid 52632] wait4(52726, <unfinished ...>

[pid 52727] close(3 <unfinished ...>

[pid 52726] <... close resumed>) = 0

[pid 52727] <... close resumed>) = 0

[pid 52726] exit_group(0) = ?

[pid 52727] exit_group(0) = ?

[pid 52726] +++ exited with 0 +++

[**pid 52632] <... wait4 resumed>NULL, 0, NULL) = 52726**

[pid 52632] --- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=52726, si_uid=1000, si_status=0, si_utime=0, si_stime=0} ---

[pid 52727] +++ exited with 0 +++

--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=52727, si_uid=1000, si_status=0, si_utime=0, si_stime=0} ---

wait4(52727, NULL, 0, NULL) = 52727

write(1, "Parent: all children finished.\n", 31Parent: all children finished.

) = 31

exit_group(0) = ?

+++ exited with 0 +++

Вывод

В ходе выполнения лабораторной работы была разработана программа на языке C, 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.