

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-215Б-23

Студент: Кармишен Е.С

Преподаватель: Миронов Е.С. (ПМИ)

Оценка: _____

Дата: 14.02.24

Москва, 2024

Постановка задачи

Вариант 12.

Родительский процесс создает два дочерних процесса. Перенаправление стандартных потоков ввода-вывода показано на картинке выше. Child1 и Child2 можно «соединить» между собой дополнительным каналом. Родительский и дочерний процесс должны быть представлены разными программами.

Родительский процесс принимает от пользователя строки произвольной длины и пересылает их в pipe1. Процесс child1 и child2 производят работу над строками. Child2 пересылает результат своей работы родительскому процессу. Родительский процесс полученный результат выводит в стандартный поток вывода.

12 вариант) Child1 переводит строки в верхний регистр. Child2 убирает все задвоенные пробелы.

Общий метод и алгоритм решения

Использованные системные вызовы:

- **fork()** – создает дочерний процесс (копия родительского).
- **pipe()** – создает канал (pipe) для обмена данными между процессами.
- **exit()** – завершает выполнение процесса.
- **execl()** – заменяет текущий процесс новым (загрузка программы).
- **close()** – закрывает файловый дескриптор (например, конец канала).
- **dup2()** – перенаправляет дескриптор (например, pipe на stdin/stdout).
- **perror()** – выводит сообщение об ошибке (на основе errno).
- **wait()** – ожидает завершения дочернего процесса.

Алгоритм работы программы

Родительский процесс

1. Родительский процесс (Parent.c)

1. Создание каналов:

- Родительский процесс создает два канала (pipe1 и pipe2) для обмена данными между процессами.

2. Создание первого дочернего процесса (child1):

- Родительский процесс создает первый дочерний процесс с помощью fork().
- В дочернем процессе:
 - Закрываются ненужные концы каналов: конец записи pipe1 и конец чтения pipe2.
 - Стандартный ввод (stdin) перенаправляется на чтение из pipe1, а стандартный вывод (stdout) — на запись в pipe2.
 - Выполняется программа child1, которая будет читать данные из pipe1, обрабатывать их и записывать результат в pipe2.

3. Создание второго дочернего процесса (child2):

- Родительский процесс создает второй дочерний процесс с помощью fork().
- В дочернем процессе:
 - Закрываются ненужные концы каналов: оба конца pipe1 и конец записи pipe2.
 - Стандартный ввод (stdin) перенаправляется на чтение из pipe2.
 - Выполняется программа child2, которая будет читать данные из pipe2, обрабатывать их и записывать результат в стандартный вывод.

4. Родительский процесс продолжает работу:

- Родительский процесс закрывает ненужные концы каналов: конец чтения `pipe1` и оба конца `pipe2`.
- Родительский процесс читает строки от пользователя и записывает их в `pipe1`.

5. Завершение работы:

- После завершения ввода данных родительский процесс закрывает конец записи `pipe1`.
- Родительский процесс ожидает завершения обоих дочерних процессов с помощью `wait()`.
- После завершения всех процессов выводится сообщение о завершении работы.

2. Первый дочерний процесс ('child1')

1. Читает строку из стандартного ввода (перенаправленный `pipe1`).
2. Преобразует строку в верхний регистр.
3. Отправляет измененную строку в `pipe2` (перенаправленный стандартный вывод).
4. Завершает работу после обработки всех строк.

3. Второй дочерний процесс ('child2')

1. Читает строку из стандартного ввода (перенаправленный `pipe2`).
2. Удаляет лишние пробелы в строке.
3. Выводит итоговую строку в стандартный вывод, чтобы родительский процесс мог ее прочитать.
4. Завершает работу после обработки всех строк.

Код программы

parent.c

```
#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <string.h>

#include <sys/wait.h>

#define MAX_LINE 1000

int main() {

    int pipe1[2], pipe2[2];

    pid_t child1, child2;

    // Создаем два канала

    if (pipe(pipe1) == -1 || pipe(pipe2) == -1) {

        perror("Ошибка создания канала");

        exit(1);
    }
}
```

```
1

// Создаем первый дочерний процесс

if ((child1 = fork()) == -1) {

perror("Ошибка создания первого дочернего процесса");

exit(1);

}

if (child1 == 0) {

// Код для первого дочернего процесса (child1)

close(pipe1[1]); // Закрываем конец записи pipe1

close(pipe2[0]); // Закрываем конец чтения pipe2

dup2(pipe1[0], STDIN_FILENO); // Перенаправляем чтение с pipe1 на stdin

dup2(pipe2[1], STDOUT_FILENO); // Перенаправляем вывод на pipe2

close(pipe1[0]);

close(pipe2[1]);

exec1("./child1", "child1", NULL);

perror("Ошибка exec1 для child1");

exit(1);

}

// Создаем второй дочерний процесс

if ((child2 = fork()) == -1) {

perror("Ошибка создания второго дочернего процесса");

exit(1);

}

if (child2 == 0) {

// Код для второго дочернего процесса (child2)

close(pipe1[0]);
```

```
close(pipe1[1]); // Не используем pipe1

close(pipe2[1]); // Закрываем конец записи pipe2

dup2(pipe2[0], STDIN_FILENO); // Перенаправляем pipe2 на stdin

close(pipe2[0]);

execl("./child2", "child2", NULL);

perror("Ошибка execl для child2");

exit(1);
}

// Родительский процесс

close(pipe1[0]); // Закрываем конец чтения pipe1

close(pipe2[0]); // Закрываем конец чтения pipe2

close(pipe2[1]); // Закрываем конец записи pipe2 для родителя

char line[MAX_LINE];

printf("Введите строки (Ctrl+D для завершения):\n");

// Чтение строк от пользователя и пересылка их в child1

while (fread(line, sizeof(char), MAX_LINE, stdin) != 0) {

write(pipe1[1], line, strlen(line)); //записывает данные из line в канал pipe1

}

close(pipe1[1]); // Закрываем конец записи pipe1 после ввода

// Ожидание завершения дочерних процессов

wait(NULL); // Ждем завершения child1

wait(NULL); // Ждем завершения child2

printf("\nВсе процессы завершены.\n");
```

```
return 0;
```

```
}
```

child1.c

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <ctype.h>
```

```
#include <unistd.h>
```

```
#include <string.h>
```

```
#define MAX_LINE 1000
```

```
int main() {
```

```
char line[MAX_LINE];
```

```
// Чтение строк из stdin (перенаправленный pipe1)
```

```
while (read(STDIN_FILENO, line, MAX_LINE) != 0) {
```

```
// Преобразование строки в верхний регистр
```

```
for (int i = 0; line[i]; i++) {
```

```
line[i] = toupper(line[i]);
```

```
}
```

```
// Отправка строки в pipe2 (stdout был перенаправлен)
```

```
write(STDOUT_FILENO, line, strlen(line)); // записываем в стандартный поток вывода  
строку line длины strlen(line)
```

```
}
```

```
return 0;
```

```
}
```

child2.c

```
#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <unistd.h>

#include <string.h>

#define MAX_LINE 1000

void remove_extra_spaces(char *str) {

    int i = 0, j = 0;

    int space_found = 0;

    while (str[i]) {

        if (str[i] == ' ') {

            if (!space_found) {

                str[j++] = str[i];

                space_found = 1;

            }

            } else {

                str[j++] = str[i];

                space_found = 0;

            }

            i++;

        }

        str[j] = '\0';

    }

    int main() {

        char line[MAX_LINE];
```

```
// Чтение строк из stdin (перенаправленный pipe2)
while (read(STDIN_FILENO, line, MAX_LINE) != 0) {
    // Удаление лишних пробелов
    remove_extra_spaces(line);
    // Вывод результата в stdout (будет прочитан родительским процессом)
    write(STDOUT_FILENO, line, strlen(line));
}

return 0;
}
```

Протокол работы программы

Тест:

apple@MacBook-Pro-apple lab1 % ./parent

Введите строки (Ctrl+D для завершения):

Sun of the sleepless! Melancholy star!

Whose tearful beam glows tremulously far,

That show'st the darkness thou canst not dispel,

How like art thou to joy remember'd well!

SUN OF THE SLEEPLESS! MELANCHOLY STAR!

WHOSE TEARFUL BEAM GLOWS TREMULOUSLY FAR,

THAT SHOW'ST THE DARKNESS THOU CANST NOT DISPEL,

HOW LIKE ART THOU TO JOY REMEMBER'D WELL!

Все процессы завершены.

Strace:

strace -f ./parent

execve("./parent", ["/parent"], 0x7fffc9da9a68 /* 21 vars */) = 0

brk(NULL) = 0x794000

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f950c739000

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=25258, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 25258, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f950c732000

close(3) = 0

openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\211\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\20t\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

newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=1922136, ...}, 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, 1970000, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f950c551000

mmap(0x7f950c577000, 1396736, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x26000) = 0x7f950c577000

mmap(0x7f950c6cc000, 339968, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x17b000) = 0x7f950c6cc000

mmap(0x7f950c71f000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1ce000) = 0x7f950c71f000


```

mmap(0x7f950c725000, 53072, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) =
0x7f950c725000
close(3) = 0
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f950c54e000
arch_prctl(ARCH_SET_FS, 0x7f950c54e740) = 0
set_tid_address(0x7f950c54ea10) = 1763
set_robust_list(0x7f950c54ea20, 24) = 0
rseq(0x7f950c54f060, 0x20, 0, 0x53053053) = 0
mprotect(0x7f950c71f000, 16384, PROT_READ) = 0
mprotect(0x403000, 4096, PROT_READ) = 0
mprotect(0x7f950c76b000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
munmap(0x7f950c732000, 25258) = 0
pipe2([3, 4], 0) = 0
pipe2([5, 6], 0) = 0
clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLDstrace: Process 1764 attached
, child_tidptr=0x7f950c54ea10) = 1764
[pid 1764] set_robust_list(0x7f950c54ea20, 24 <unfinished ...>
[pid 1763] clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD <unfinished ...>
[pid 1764] <... set_robust_list resumed>) = 0
[pid 1764] close(4strace: Process 1765 attached
) = 0
[pid 1763] <... clone resumed>, child_tidptr=0x7f950c54ea10) = 1765
[pid 1765] set_robust_list(0x7f950c54ea20, 24 <unfinished ...>
[pid 1764] close(5 <unfinished ...>
[pid 1765] <... set_robust_list resumed>) = 0
[pid 1763] close(3 <unfinished ...>
[pid 1764] <... close resumed>) = 0
[pid 1763] <... close resumed>) = 0
[pid 1765] close(3 <unfinished ...>
[pid 1763] close(5 <unfinished ...>
[pid 1764] dup2(3, 0 <unfinished ...>
[pid 1763] <... close resumed>) = 0
[pid 1765] <... close resumed>) = 0
[pid 1763] close(6 <unfinished ...>
[pid 1764] <... dup2 resumed>) = 0
[pid 1763] <... close resumed>) = 0
[pid 1765] close(4 <unfinished ...>
[pid 1764] dup2(6, 1 <unfinished ...>
[pid 1763] newfstatat(1, "", <unfinished ...>
[pid 1765] <... close resumed>) = 0
[pid 1763] <... newfstatat resumed>{st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x1), ...}, AT_EMPTY_PATH) = 0
[pid 1764] <... dup2 resumed>) = 1
[pid 1765] close(6 <unfinished ...>
[pid 1763] getrandom(<unfinished ...>
[pid 1764] close(3 <unfinished ...>
[pid 1763] <... getrandom resumed>"xf7\xb1\xdb\x02\xc9\xbb\x0d\xc3", 8, GRND_NONBLOCK) = 8
[pid 1765] <... close resumed>) = 0
[pid 1763] brk(NULL <unfinished ...>
[pid 1764] <... close resumed>) = 0
[pid 1763] <... brk resumed>) = 0x794000
[pid 1765] dup2(5, 0 <unfinished ...>
[pid 1763] brk(0x7b5000 <unfinished ...>
[pid 1764] close(6 <unfinished ...>
[pid 1763] <... brk resumed>) = 0x7b5000
[pid 1765] <... dup2 resumed>) = 0
[pid 1763] write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265
\321\201\321\202\321\200\320\276\320\272\320\270 (Ctrl"..., 66 <unfinished ...>
[pid 1764] <... close resumed>) = 0
Введите строки (Ctrl+D для завершения):
[pid 1765] close(5 <unfinished ...>
[pid 1763] <... write resumed>) = 66

```

[pid 1764] execve("./child1", ["child1"], 0x7ffe64516438 /* 21 vars */ <unfinished ...>
[pid 1763] newfstatat(0, "", <unfinished ...>
[pid 1765] <... close resumed>) = 0
[pid 1763] <... newfstatat resumed>{st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x1), ...}, AT_EMPTY_PATH) = 0
[pid 1765] execve("./child2", ["child2"], 0x7ffe64516438 /* 21 vars */ <unfinished ...>
[pid 1763] read(0, <unfinished ...>
[pid 1765] <... execve resumed>) = 0
[pid 1764] <... execve resumed>) = 0
[pid 1765] brk(NULL <unfinished ...>
[pid 1764] brk(NULL <unfinished ...>
[pid 1765] <... brk resumed>) = 0x1bdc000
[pid 1764] <... brk resumed>) = 0x14c8000
[pid 1765] mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>
[pid 1764] mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>
[pid 1765] <... mmap resumed>) = 0x7f515d657000
[pid 1764] <... mmap resumed>) = 0x7fbc566bb000
[pid 1765] access("/etc/ld.so.preload", R_OK <unfinished ...>
[pid 1764] access("/etc/ld.so.preload", R_OK <unfinished ...>
[pid 1765] <... access resumed>) = -1 ENOENT (No such file or directory)
[pid 1764] <... access resumed>) = -1 ENOENT (No such file or directory)
[pid 1765] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>
[pid 1764] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>
[pid 1765] <... openat resumed>) = 3
[pid 1764] <... openat resumed>) = 3
[pid 1765] newfstatat(3, "", <unfinished ...>
[pid 1764] newfstatat(3, "", <unfinished ...>
[pid 1765] <... newfstatat resumed>{st_mode=S_IFREG|0644, st_size=25258, ...}, AT_EMPTY_PATH) = 0
[pid 1764] <... newfstatat resumed>{st_mode=S_IFREG|0644, st_size=25258, ...}, AT_EMPTY_PATH) = 0
[pid 1765] mmap(NULL, 25258, PROT_READ, MAP_PRIVATE, 3, 0 <unfinished ...>
[pid 1764] mmap(NULL, 25258, PROT_READ, MAP_PRIVATE, 3, 0 <unfinished ...>
[pid 1765] <... mmap resumed>) = 0x7f515d650000
[pid 1764] <... mmap resumed>) = 0x7fbc566b4000
[pid 1765] close(3 <unfinished ...>
[pid 1764] close(3 <unfinished ...>
[pid 1765] <... close resumed>) = 0
[pid 1764] <... close resumed>) = 0
[pid 1765] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC <unfinished ...>
[pid 1764] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC <unfinished ...>
[pid 1765] <... openat resumed>) = 3
[pid 1764] <... openat resumed>) = 3
[pid 1765] read(3, <unfinished ...>
[pid 1764] read(3, <unfinished ...>
[pid 1765] <... read resumed>"\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\20t\2\0\0\0\0"..., 832) = 832
[pid 1764] <... read resumed>"\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\20t\2\0\0\0\0"..., 832) = 832
[pid 1765] pread64(3, <unfinished ...>
[pid 1764] pread64(3, <unfinished ...>
[pid 1765] <... 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"..., 784, 64) = 784
[pid 1765] newfstatat(3, "", <unfinished ...>
[pid 1764] <... 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"..., 784, 64) = 784
[pid 1765] <... newfstatat resumed>{st_mode=S_IFREG|0755, st_size=1922136, ...}, AT_EMPTY_PATH) = 0
[pid 1764] newfstatat(3, "", <unfinished ...>
[pid 1765] pread64(3, <unfinished ...>
[pid 1764] <... newfstatat resumed>{st_mode=S_IFREG|0755, st_size=1922136, ...}, AT_EMPTY_PATH) = 0
[pid 1765] <... 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"..., 784, 64) = 784
[pid 1764] pread64(3, <unfinished ...>
[pid 1765] mmap(NULL, 1970000, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0 <unfinished ...>
[pid 1764] <... 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"..., 784, 64) = 784
[pid 1765] <... mmap resumed>) = 0x7f515d46f000
[pid 1764] mmap(NULL, 1970000, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0 <unfinished ...>
[pid 1765] mmap(0x7f515d495000, 1396736, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x26000 <unfinished ...>
[pid 1764] <... mmap resumed>) = 0x7fbc564d3000

```

[pid 1765] <... mmap resumed>      = 0x7f515d495000
[pid 1764] mmap(0x7fbc564f9000, 1396736, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x26000 <unfinished ...>
[pid 1765] mmap(0x7f515d5ea000, 339968, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x17b000 <unfinished ...>
[pid 1764] <... mmap resumed>      = 0x7fbc564f9000
[pid 1765] <... mmap resumed>      = 0x7f515d5ea000
[pid 1764] mmap(0x7fbc5664e000, 339968, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x17b000 <unfinished ...>
[pid 1765] mmap(0x7f515d63d000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1ce000 <unfinished ...>
[pid 1764] <... mmap resumed>      = 0x7fbc5664e000
[pid 1765] <... mmap resumed>      = 0x7f515d63d000
[pid 1764] mmap(0x7fbc566a1000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1ce000 <unfinished ...>
[pid 1765] mmap(0x7f515d643000, 53072, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>
[pid 1764] <... mmap resumed>      = 0x7fbc566a1000
[pid 1765] <... mmap resumed>      = 0x7f515d643000
[pid 1764] mmap(0x7fbc566a7000, 53072, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>
[pid 1765] close(3 <unfinished ...>
[pid 1764] <... mmap resumed>      = 0x7fbc566a7000
[pid 1765] <... close resumed>      = 0
[pid 1764] close(3 <unfinished ...>
[pid 1765] mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>
[pid 1764] <... close resumed>      = 0
[pid 1765] <... mmap resumed>      = 0x7f515d46c000
[pid 1764] mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>
[pid 1765] arch_prctl(ARCH_SET_FS, 0x7f515d46c740) = 0
[pid 1764] <... mmap resumed>      = 0x7fbc564d0000
[pid 1765] set_tid_address(0x7f515d46ca10 <unfinished ...>
[pid 1764] arch_prctl(ARCH_SET_FS, 0x7fbc564d0740 <unfinished ...>
[pid 1765] <... set_tid_address resumed> = 1765
[pid 1764] <... arch_prctl resumed> = 0
[pid 1765] set_robust_list(0x7f515d46ca20, 24 <unfinished ...>
[pid 1764] set_tid_address(0x7fbc564d0a10 <unfinished ...>
[pid 1765] <... set_robust_list resumed> = 0
[pid 1764] <... set_tid_address resumed> = 1764
[pid 1765] rseq(0x7f515d46d060, 0x20, 0, 0x53053053 <unfinished ...>
[pid 1764] set_robust_list(0x7fbc564d0a20, 24 <unfinished ...>
[pid 1765] <... rseq resumed>      = 0
[pid 1764] <... set_robust_list resumed> = 0
[pid 1764] rseq(0x7fbc564d1060, 0x20, 0, 0x53053053 <unfinished ...>
[pid 1765] mprotect(0x7f515d63d000, 16384, PROT_READ <unfinished ...>
[pid 1764] <... rseq resumed>      = 0
[pid 1765] <... mprotect resumed> = 0
[pid 1765] mprotect(0x403000, 4096, PROT_READ <unfinished ...>
[pid 1764] mprotect(0x7fbc566a1000, 16384, PROT_READ <unfinished ...>
[pid 1765] <... mprotect resumed> = 0
[pid 1764] <... mprotect resumed> = 0
[pid 1765] mprotect(0x7f515d689000, 8192, PROT_READ <unfinished ...>
[pid 1764] mprotect(0x403000, 4096, PROT_READ <unfinished ...>
[pid 1765] <... mprotect resumed> = 0
[pid 1764] <... mprotect resumed> = 0
[pid 1765] prlimit64(0, RLIMIT_STACK, NULL, <unfinished ...>
[pid 1764] mprotect(0x7fbc566ed000, 8192, PROT_READ <unfinished ...>
[pid 1765] <... prlimit64 resumed>{rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY} = 0
[pid 1764] <... mprotect resumed> = 0
[pid 1765] munmap(0x7f515d650000, 25258) = 0
[pid 1764] prlimit64(0, RLIMIT_STACK, NULL, <unfinished ...>
[pid 1765] newfstatat(0, "", <unfinished ...>

```

```

[pid 1764] <... prlimit64 resumed>{rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
[pid 1765] <... newfstatat resumed>{st_mode=S_IFIFO|0600, st_size=0, ...}, AT_EMPTY_PATH) = 0
[pid 1765] getrandom( <unfinished ...>
[pid 1764] munmap(0x7fbc566b4000, 25258 <unfinished ...>
[pid 1765] <... getrandom resumed>"x5c\xbf\xaf\x8a\xed\x83\x71\xcb", 8, GRND_NONBLOCK) = 8
[pid 1764] <... munmap resumed>) = 0
[pid 1765] brk(NULL) = 0x1bdc000
[pid 1764] newfstatat(0, "", <unfinished ...>
[pid 1765] brk(0x1bfd000 <unfinished ...>
[pid 1764] <... newfstatat resumed>{st_mode=S_IFIFO|0600, st_size=0, ...}, AT_EMPTY_PATH) = 0
[pid 1765] <... brk resumed>) = 0x1bfd000
[pid 1764] getrandom( <unfinished ...>
[pid 1765] read(0, <unfinished ...>
[pid 1764] <... getrandom resumed>"x93\x37\xcc\x65\x88\xf1\xc2\x41", 8, GRND_NONBLOCK) = 8
[pid 1764] brk(NULL) = 0x14c8000
[pid 1764] brk(0x14e9000) = 0x14e9000
[pid 1764] read(0, Sun of the sleepless! Melancholy star!
Whose tearful beam glows tremulously far,
That show'st the darkness thou canst not dispel,
How like art thou to joy remember'd well! <unfinished ...>
[pid 1763] <... read resumed>"Sun of the sleepless! Melanc"..., 1024) = 44
[pid 1763] write(4, "Sun of the sleepless! Melanc"..., 44) = 44
[pid 1764] <... read resumed>"Sun of the sleepless! Melanc"..., 4096) = 44
[pid 1763] read(0, <unfinished ...>
[pid 1764] newfstatat(1, "", <unfinished ...>
[pid 1763] <... read resumed>"Whose tearful beam glows tre"..., 1024) = 47
[pid 1764] <... newfstatat resumed>{st_mode=S_IFIFO|0600, st_size=0, ...}, AT_EMPTY_PATH) = 0
[pid 1763] write(4, "Whose tearful beam glows tre"..., 47 <unfinished ...>
[pid 1764] read(0, <unfinished ...>
[pid 1763] <... write resumed>) = 47
[pid 1764] <... read resumed>"Whose tearful beam glows tre"..., 4096) = 47
[pid 1763] read(0, <unfinished ...>
[pid 1764] read(0, <unfinished ...>
[pid 1763] <... read resumed>"That show\342\200\231st the darkness "..., 1024) = 58
[pid 1763] write(4, "That show\342\200\231st the darkness "..., 58) = 58
[pid 1764] <... read resumed>"That show\342\200\231st the darkness "..., 4096) = 58
[pid 1763] read(0, <unfinished ...>
[pid 1764] read(0,
<unfinished ...>
[pid 1763] <... read resumed>"How like art thou to joy r"..., 1024) = 51
[pid 1763] write(4, "How like art thou to joy r"..., 51) = 51
[pid 1764] <... read resumed>"How like art thou to joy r"..., 4096) = 51
[pid 1763] read(0, <unfinished ...>
[pid 1764] read(0, <unfinished ...>
[pid 1763] <... read resumed>"", 1024) = 0
[pid 1763] close(4) = 0
[pid 1764] <... read resumed>"", 4096) = 0
[pid 1763] wait4(-1, <unfinished ...>
[pid 1764] write(1, "SUN OF THE SLEEPLESS! MELANC"..., 200) = 200
[pid 1765] <... read resumed>"SUN OF THE SLEEPLESS! MELANC"..., 4096) = 200
[pid 1764] exit_group(0) = ?
[pid 1765] newfstatat(1, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x1), ...}, AT_EMPTY_PATH) = 0
[pid 1765] write(1, "SUN OF THE SLEEPLESS! MELANCHOLY"..., 39 <unfinished ...>
[pid 1764] +++ exited with 0 +++
SUN OF THE SLEEPLESS! MELANCHOLY STAR!
[pid 1763] <... wait4 resumed>NULL, 0, NULL) = 1764
[pid 1765] <... write resumed>) = 39
[pid 1763] --- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=1764, si_uid=0, si_status=0, si_utime=0, si_stime=0}
---
[pid 1765] write(1, "WHOSE TEARFUL BEAM GLOWS TREMULO"..., 42WHOSE TEARFUL BEAM GLOWS TREMULOUSLY
FAR,
<unfinished ...>

```

```

[pid 1763] wait4(-1, <unfinished ...>
[pid 1765] <... write resumed>      = 42
[pid 1765] write(1, "THAT SHOW\342\200\231ST THE DARKNESS THOU"..., 51THAT SHOW'ST THE DARKNESS THOU
CANST NOT DISPEL,
) = 51
[pid 1765] write(1, "HOW LIKE ART THOU TO JOY REMEMBE"..., 44HOW LIKE ART THOU TO JOY REMEMBER'D WELL!
) = 44
[pid 1765] read(0, "", 4096)        = 0
[pid 1765] exit_group(0)            = ?
[pid 1765] +++ exited with 0 +++
<... wait4 resumed>NULL, 0, NULL)    = 1765
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=1765, si_uid=0, si_status=0, si_utime=0, si_stime=0} ---
write(1, "\320\222\321\201\320\265 \320\277\321\200\320\276\321\206\320\265\321\201\321\201\321\213
\320\267\320\260\320\262\320\265"..., 44Все процессы завершены.
) = 44
exit_group(0)                      = ?
+++ exited with 0 +++

```

Вывод

Эта лабораторная работа позволила мне лучше понять механизмы межпроцессного взаимодействия в операционных системах, в частности, работу с каналами (pipes) и перенаправлением ввода-вывода. Я научился создавать дочерние процессы с помощью `fork()` и управлять их взаимодействием через каналы, что является важным аспектом разработки многозадачных приложений. Работа также помогла закрепить навыки работы с системными вызовами, такими как `pipe()`, `dup2()`, и `execl()`, что будет полезно для дальнейшего изучения операционных систем. В целом, лабораторная работа была интересной и познавательной, хотя потребовала внимательности при отладке взаимодействия процессов.