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: Тараскаев Д.М.

Преподаватель: Миронов Е.С.

Оценка: _____

Дата: 01.11.24

Москва, 2024

Постановка задачи

Вариант 16.

Правило проверки: строка должна оканчиваться на “.” или “;”.

Общий метод и алгоритм решения

Использованные системные вызовы:

- pid_t pid = fork(void); – создает дочерний процесс.
- int pipe1[2], pipe2[2]; - создание каналов.
- close(pipe); - закрытие стороны канала.
- dup2(pipe, STDXX_FILENO); - перенаправление канала.
- execpl(); - запуск сторонней исполняемой программы.
- waitpid(); - ожидание завершения дочернего процесса.

Родительский процесс выполняет следующие действия:

1. Создает два канала для двусторонней связи между процессами.
2. Создает дочерний процесс с помощью fork().
3. В дочернем процессе перенаправляет стандартный ввод и вывод на каналы и запускает исполняемый файл дочернего процесса с помощью execlp().

Дочерний процесс выполняет следующие действия:

1. Читает имя файла из стандартного ввода и открывает его для записи.
2. Читает сообщение из стандартного ввода.
3. Проверяет, оканчивается ли сообщение на точку или точку с запятой.
4. Закрывает файл и завершает работу.

Код программы

parent.c

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

int main() {
    int pipe1[2];
    int pipe2[2];
    pid_t pid;
    if (pipe(pipe1) == -1 || pipe(pipe2) == -1) {
        perror("pipe failed");
        return 1;
    }
    pid = fork();
    if (pid == -1) {
```

```
        perror("fork failed");
        return 1;
    }
    // pipe[0] - чтение, pipe[1] - запись
    if (pid == 0) {
        close(pipe1[1]);
        close(pipe2[0]);

        dup2(pipe1[0], STDIN_FILENO); // Перенаправляем стандартный ввод на pipe1[0]
        dup2(pipe2[1], STDOUT_FILENO); // Перенаправляем стандартный вывод на pipe2[1]

        close(pipe1[0]);
        close(pipe2[1]);

        execlp("./child", "child", NULL);
        perror("execlp failed");
        exit(1);
    }
    else {
        char msg[100];
        char read_msg[100];
        close(pipe1[0]);
        close(pipe2[1]);
        printf("write a doc name\n");
        fgets(msg, sizeof(msg), stdin);
        write(pipe1[1], msg, strlen(msg) + 1);
        while (1) {
            printf("Enter a message: ");
            fgets(msg, sizeof(msg), stdin);
            if (strncmp(msg, "exit", 4) == 0) {
                break;
            }
            write(pipe1[1], msg, strlen(msg) + 1);
            read(pipe2[0], read_msg, sizeof(read_msg));
            printf("Parent read: %s\n", read_msg);
        }
        close(pipe1[1]);
        close(pipe2[0]);
        waitpid(pid, NULL, 0);
    }
    return 0;
}
```

child.c

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

```
int main() {
    char read_msg[100];
    char response_msg[100];
    // Читаем имя файла из stdin
    read(STDIN_FILENO, read_msg, sizeof(read_msg));
    FILE *fp = fopen(read_msg, "w");
    if (!fp) {
        perror("file error");
        return -1;
    }
    while (1) {
        int bytes_read = read(STDIN_FILENO, read_msg, sizeof(read_msg));
        if (bytes_read <= 0) {
            break;
        }
        int len = strlen(read_msg);
        if (len > 0 && (read_msg[len - 2] == ';' || read_msg[len - 2] == '.')) {
            fputs(read_msg, fp);
            strcpy(response_msg, "Успешно");
        } else {
            strcpy(response_msg, "Предложение не оканчивается на . или ;");
        }
        // Пишем данные в stdout
        write(STDOUT_FILENO, response_msg, strlen(response_msg) + 1);
    }
    fclose(fp);
    return 0;
}
```

Протокол работы программы

Здесь нужно показать тесты программы (текст или скриншоты), а затем показать полный вывод утилиты strace (или какой-либо другой утилиты на Windows, если вы выполняете лабы на этой операционной системе).

В strace нужно обязательно выделить, где происходят системные вызовы, которые вы использовали в лабораторной работе (например, где в первой лабораторной работе был вызван fork и другие вызовы). Полный список вызовов, которые нужно будет выделить в выводе strace, будет указан при выдаче лабы в нашем канале.

Тестирование:

```
$ ./parent
write a doc name
hellow
Enter a message: hellow;
Parent read: Успешно
Enter a message: qwe
Parent read: Предложение не оканчивается на . или ;
Enter a message: asfdd;
Parent read: Успешно
Enter a message: xddddd;
```

```
Parent read: Успешно
Enter a message: exit
```

Strace:

```
$ strace -f ./main
execve("./parent", ["./parent"], 0x7ffd5c20faa8 /* 50 vars */) = 0
brk(NULL)                               = 0x5cec77aeb000
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=121423, ...}) = 0
mmap(NULL, 121423, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fb904dc3000
close(3)                                = 0
openat(AT_FDCWD, "/usr/lib/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\340_\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
fstat(3, {st_mode=S_IFREG|0755, st_size=2014520, ...}) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7fb904dc1000
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, 2034616, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fb904bd0000
mmap(0x7fb904bf4000, 1511424, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x24000) = 0x7fb904bf4000
mmap(0x7fb904d65000, 319488, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x195000) = 0x7fb904d65000
mmap(0x7fb904db3000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x1e3000) = 0x7fb904db3000
mmap(0x7fb904db9000, 31672, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,
-1, 0) = 0x7fb904db9000
close(3)                                = 0
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7fb904bcd000
arch_prctl(ARCH_SET_FS, 0x7fb904bcd740) = 0
set_tid_address(0x7fb904bcda10)         = 36016
set_robust_list(0x7fb904bcda20, 24)     = 0
rseq(0x7fb904bce060, 0x20, 0, 0x53053053) = 0
mprotect(0x7fb904db3000, 16384, PROT_READ) = 0
mprotect(0x5cec42537000, 4096, PROT_READ) = 0
mprotect(0x7fb904e1b000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
munmap(0x7fb904dc3000, 121423)          = 0
pipe2([3, 4], 0)                        = 0
pipe2([5, 6], 0)                        = 0
clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,
child_tidptr=0x7fb904bcda10) = 36017
```

```
strace: Process 36017 attached
[pid 36016] close(3)                    = 0
[pid 36016] close(6 <unfinished ...>
[pid 36017] set_robust_list(0x7fb904bcda20, 24 <unfinished ...>
[pid 36016] <... close resumed>)        = 0
[pid 36016] fstat(1,  <unfinished ...>
[pid 36017] <... set_robust_list resumed>) = 0
[pid 36016] <... fstat resumed>{st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}) =
0
[pid 36016] getrandom("\x30\x10\xd3\x1b\xe3\x53\x96\x65", 8, GRND_NONBLOCK) = 8
[pid 36016] brk(NULL)                   = 0x5cec77aeb000
[pid 36017] close(4 <unfinished ...>
[pid 36016] brk(0x5cec77b0c000)         = 0x5cec77b0c000
[pid 36017] <... close resumed>)        = 0
[pid 36016] write(1, "write a doc name\n", 17 <unfinished ...>
write a doc name
[pid 36017] close(5 <unfinished ...>
[pid 36016] <... write resumed>)        = 17
[pid 36017] <... close resumed>)        = 0
[pid 36016] fstat(0, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}) = 0
[pid 36017] dup2(3, 0 <unfinished ...>
[pid 36016] read(0,  <unfinished ...>
[pid 36017] <... dup2 resumed>)         = 0
[pid 36017] dup2(6, 1)                  = 1
[pid 36017] close(3)                    = 0
[pid 36017] close(6)                    = 0
[pid 36017] execve("./child", ["child"], 0x7ffc5812e4b8 /* 50 vars */) = 0
[pid 36017] brk(NULL)                   = 0x569d17ded000
[pid 36017] access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)
[pid 36017] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
[pid 36017] fstat(3, {st_mode=S_IFREG|0644, st_size=121423, ...}) = 0
[pid 36017] mmap(NULL, 121423, PROT_READ, MAP_PRIVATE, 3, 0) = 0x74a86bff5000
[pid 36017] close(3)                    = 0
[pid 36017] openat(AT_FDCWD, "/usr/lib/libc.so.6", O_RDONLY|O_CLOEXEC) = 3

[pid 36017] read(3,
"\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\340_\2\0\0\0\0\0"..., 832) = 832

[pid 36017] 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 36017] fstat(3, {st_mode=S_IFREG|0755, st_size=2014520, ...}) = 0

[pid 36017] mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x74a86bff3000

[pid 36017] 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 36017] mmap(NULL, 2034616, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x74a86be02000

[pid 36017] mmap(0x74a86be26000, 1511424, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x24000) = 0x74a86be26000
```

```
     [pid 36017] mmap(0x74a86bf97000, 319488, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x195000) = 0x74a86bf97000

     [pid 36017] mmap(0x74a86bfe5000, 24576, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e3000) = 0x74a86bfe5000

     [pid 36017] mmap(0x74a86bfeb000, 31672, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x74a86bfeb000
     [pid 36017] close(3)                    = 0

     [pid 36017] mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x74a86bdff000
     [pid 36017] arch_prctl(ARCH_SET_FS, 0x74a86bdff740) = 0
     [pid 36017] set_tid_address(0x74a86bdffa10) = 36017
     [pid 36017] set_robust_list(0x74a86bdffa20, 24) = 0
     [pid 36017] rseq(0x74a86be00060, 0x20, 0, 0x53053053) = 0
     [pid 36017] mprotect(0x74a86bfe5000, 16384, PROT_READ) = 0
     [pid 36017] mprotect(0x569cef063000, 4096, PROT_READ) = 0
     [pid 36017] mprotect(0x74a86c04d000, 8192, PROT_READ) = 0

     [pid 36017] prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024,
rlim_max=RLIM64_INFINITY}) = 0
     [pid 36017] munmap(0x74a86bff5000, 121423) = 0
     [pid 36017] read(0, hellow
      <unfinished ...>
     [pid 36016] <... read resumed>"hellow\n", 1024) = 7
     [pid 36016] write(4, "hellow\n\0", 8)   = 8
     [pid 36017] <... read resumed>"hellow\n\0", 100) = 8
     [pid 36016] write(1, "Enter a message: ", 17 <unfinished ...>
     [pid 36017] getrandom(Enter a message:  <unfinished ...>
     [pid 36016] <... write resumed>)        = 17

     [pid 36017] <... getrandom resumed>"\xaa\x27\x82\x44\x3f\x93\xbb\x5e", 8,
GRND_NONBLOCK) = 8
     [pid 36016] read(0,  <unfinished ...>
     [pid 36017] brk(NULL)                   = 0x569d17ded000
     [pid 36017] brk(0x569d17e0e000)         = 0x569d17e0e000
     [pid 36017] openat(AT_FDCWD, "hellow\n", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 3
     [pid 36017] read(0, hellow;
      <unfinished ...>
     [pid 36016] <... read resumed>"hellow;\n", 1024) = 8
     [pid 36016] write(4, "hellow;\n\0", 9)  = 9
     [pid 36017] <... read resumed>"hellow;\n\0", 100) = 9
     [pid 36016] read(5,  <unfinished ...>
     [pid 36017] fstat(3, {st_mode=S_IFREG|0644, st_size=0, ...}) = 0
     [pid 36017] write(1, "\320\243\321\201\320\277\320\265\321\210\320\275\320\276\0", 15)
= 15

     [pid 36016] <... read
resumed>"\320\243\321\201\320\277\320\265\321\210\320\275\320\276\0", 100) = 15
     [pid 36017] read(0,  <unfinished ...>

     [pid 36016] write(1, "Parent read:
\320\243\321\201\320\277\320\265\321\210\320\275\320\276\n", 28Parent read: Успешно

     ) = 28
```

```
[pid 36016] write(1, "Enter a message: ", 17Enter a message: ) = 17
[pid 36016] read(0, bye
"bye\n", 1024)      = 4
[pid 36016] write(4, "bye\n\0", 5)      = 5
[pid 36017] <... read resumed>"bye\n\0", 100) = 5
[pid 36016] read(5,  <unfinished ...>
[pid 36017] write(1,
"\320\237\321\200\320\265\320\264\320\273\320\276\320\266\320\265\320\275\320\270\320\265
\320\275\320\265 \320\276\320\272"..., 69 <unfinished ...>
[pid 36016] <... read
resumed>"\320\237\321\200\320\265\320\264\320\273\320\276\320\266\320\265\320\275\320\270\32
0\265 \320\275\320\265 \320\276\320\272"..., 100) = 69
[pid 36017] <... write resumed>)        = 69
[pid 36016] write(1, "Parent read:
\320\237\321\200\320\265\320\264\320\273\320\276\320\266\320\265\320\275\320"..., 82
<unfinished ...>
[pid 36017] read(0, Parent read: Предложение не оканчивается на . или ;
 <unfinished ...>
[pid 36016] <... write resumed>)        = 82
[pid 36016] write(1, "Enter a message: ", 17Enter a message: ) = 17
[pid 36016] read(0, asd;
"asd;\n", 1024)     = 5
[pid 36016] write(4, "asd;\n\0", 6)     = 6
[pid 36017] <... read resumed>"asd;\n\0", 100) = 6
[pid 36016] read(5,  <unfinished ...>
[pid 36017] write(1, "\320\243\321\201\320\277\320\265\321\210\320\275\320\276\0", 15)
= 15
[pid 36016] <... read
resumed>"\320\243\321\201\320\277\320\265\321\210\320\275\320\276\0", 100) = 15
[pid 36017] read(0,  <unfinished ...>
[pid 36016] write(1, "Parent read:
\320\243\321\201\320\277\320\265\321\210\320\275\320\276\n", 28Parent read: Успешно
) = 28
[pid 36016] write(1, "Enter a message: ", 17Enter a message: ) = 17
[pid 36016] read(0, exit
"exit\n", 1024)     = 5
[pid 36016] close(4)                    = 0
[pid 36017] <... read resumed>"", 100)  = 0
[pid 36016] close(5)                    = 0
[pid 36017] write(3, "hellow;\nasd;\n", 13 <unfinished ...>
[pid 36016] wait4(36017,  <unfinished ...>
[pid 36017] <... write resumed>)        = 13
[pid 36017] close(3)                    = 0
[pid 36017] exit_group(0)               = ?
[pid 36017] +++ exited with 0 +++
<... wait4 resumed>NULL, 0, NULL)       = 36017
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=36017, si_uid=1000,
si_status=0, si_utime=0, si_stime=0} ---
```

```
exit_group(0)                           = ?
+++ exited with 0 +++
```

Вывод

Программа успешно демонстрирует использование каналов для межпроцессного взаимодействия, позволяя родительскому и дочернему процессам обмениваться данными. Родительский процесс отправляет сообщения дочернему процессу, который проверяет их и записывает в файл, если они соответствуют заданным условиям.