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: М80-206Б-22

Студент: Филатов А. К.

Преподаватель: Миронов Е.С.

Оценка: _____

Дата: 08.11.23

Москва, 2023

Постановка задачи

Группа вариантов 2.

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.

Вариант 6.

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

Общий метод и алгоритм решения

Использованные системные вызовы:

- pid_t fork(void); – создает дочерний процесс, возвращает PID дочернего процесса, а процессу потомку возвращается 0, а в случае ошибки -1.
- int pipe(int *fd); – создает неименованный канал, у которого первое поле отвечает за чтение, а второе - за запись.
- int execl(const char *__path, char *const *__argv, ...); - предоставляет новой программе список аргументов в виде массива указателей на строки, заканчивающиеся (char *)0.
- int dup2(int oldfd, int newfd); - создает копию файлового дескриптора oldfd *(1 поле)*, используя для нового дескриптора newfd *(2 поле)* файловый дескриптор (они становятся взаимозаменяемыми).
- _exit(int status); – выходит из процесса с заданным статусом.
- pid_t wait(int *status); – приостаналивает выполнение текущего процесса до тех пор, пока дочерний процесс не завершится.
- int read(int fd, void *buffer, int nbyte); – читает nbyte байтов из файлового дескриптора fd в буффер buffer.

Сначала пользователь в качестве аргумента командной строки пишет имя файла, которое будет использоваться для открытия файла с таким же именем на чтение. Если строка введена корректно, и файл с таким именем существует, то создается дочерний процесс, и происходит переопределение стандартного ввода для дочернего процесса: стандартным вводом теперь является открытый файл, имя которого пользователь указал, и стандартный вывод дочернего процесса переопределяется каналом pipe. Родительский процесс считывает из pipe результат работы дочернего процесса и выводит его на стандартный ввод. В противном случае дочерний процесс вернет значение -1, на экран будет выведено сообщение «Something went wrong» и работа завершится.

Код программы

parent.c

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

void check_error(bool expression, char* message) {
    if (expression) {
        write(STDOUT_FILENO, message, strlen(message) * sizeof(char));
        write(STDOUT_FILENO, "\n", 1);
        exit(-1);
    }
}

int main (int argc, char* argv[]) {
    pid_t pid;
    int pipe_1[2];
    if (pipe(pipe_1) == -1) {
        perror("pipe");
        _exit(EXIT_FAILURE);
    }
    if (argc != 2) {
        write(1, "Error: no filename\n", 20);
        exit(EXIT_FAILURE);
    }
    int fd = open(argv[1], O_RDONLY);
    check_error(fd == -1, "Can't open file");
    pid = fork();
    if (pid == -1) {
        perror("fork");
        return -1;
```

```
    }
    else if (pid == 0) {
        close (pipe_1[0]);
        check_error(dup2(fd, STDIN_FILENO) < 0, "Error dup");
        dup2(pipe_1[1], STDOUT_FILENO);
        execl("./child", "/.child", NULL);
        perror("execl");
        return 1;
    }
    else {
        check_error((pid == -1), "Process error");
        close(pipe_1[1]);
        wait(0);
        int result;
        char answer[50];
        while ((read(pipe_1[0], &result, sizeof(int))) > 0) {
            if (result == -1) {
                write(STDOUT_FILENO, "Something went wrong\n", 22);
                exit(EXIT_FAILURE);
            }
            sprintf(answer, "%d\n", result);
            check_error(write(STDOUT_FILENO, answer, strlen(answer)) == -1,
"Write error\n");
            check_error(write(STDOUT_FILENO, "\n", 1) == -1, "Write error\n");
        }
    }
    return 0;
}
```

child.c

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/types.h>
#include <fcntl.h>
#include <string.h>
#include <stdbool.h>
#include <math.h>
#define buf_size 100

int main() {
    int c;
    bool not_end = true;
    int nmbr = 0;
    int result = 0;
    int count = 0;
    int numbers[100];

    do {
        if (not_end) {
            if (c <= '9' && c >= '0') {
                nmbr = nmbr * 10 + c - '0';
            }
            if (c == ' ' || c == '\n' || c == EOF) {
                numbers[count] = nmbr;
                nmbr = 0;
                count++;
                if (c == '\n' || c == EOF) {
                    for (int i = 0; i < count; i++) {
                        result = result + numbers[i];
                    }
                    not_end = false;
                    count = 0;
                }
            }
        }
        if (c == '\n' || c == EOF) {
            write(STDOUT_FILENO, &result, sizeof(result));
            result = 0;
            not_end = true;
        }
    } while((read(STDIN_FILENO, &c, sizeof(char))) > 0);
    return 0;
}
```

Протокол работы программы

Тестирование:

```
cblphblu@DESKTOP-3PDEL6G:/mnt/c/Users/user/Desktop/МАИ/2
курс/ОСИ/LR1FILATOV$ ./parent file.txt
9


16


15


17


19


0


17


55
```

Strace

```
cblphblu@DESKTOP-3PDEL6G:/mnt/c/Users/user/Desktop/МАИ/2 курс/ОСИ/LR1FILATOV$ strace -f ./parent file.txt
execve("./parent", ["./parent", "file.txt"], 0x7fff52abbb60 /* 19 vars */) = 0
brk(NULL)                               = 0x5615d631f000
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffe739a3550) = -1 EINVAL (Недопустимый аргумент)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f577a189000
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
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=34303, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 34303, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f577a180000
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\0P\237\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"..., 784, 64) = 784
pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0"..., 48, 848) = 48
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0i8\235HZ\227\223\333\350s\360\352,\223\340."..., 68, 896) = 68
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2216304, ...}, 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"..., 784, 64) = 784
mmap(NULL, 2260560, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f5779f58000
mmap(0x7f5779f80000, 1658880, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7f5779f80000
mmap(0x7f577a115000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1bd000) = 0x7f577a115000
mmap(0x7f577a16d000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x214000) = 0x7f577a16d000
mmap(0x7f577a173000, 52816, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f577a173000
close(3)                                = 0
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f5779f55000
arch_prctl(ARCH_SET_FS, 0x7f5779f55740) = 0
set_tid_address(0x7f5779f55a10)         = 177
set_robust_list(0x7f5779f55a20, 24)     = 0
rseq(0x7f5779f560e0, 0x20, 0, 0x53053053) = 0
mprotect(0x7f577a16d000, 16384, PROT_READ) = 0
mprotect(0x5615d4506000, 4096, PROT_READ) = 0
mprotect(0x7f577a1c3000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
munmap(0x7f577a180000, 34303)           = 0
pipe2([3, 4], 0)                        = 0
openat(AT_FDCWD, "file.txt", O_RDONLY)  = 5
clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLDstrace: Process 178 attached
, child_tidptr=0x7f5779f55a10) = 178
[pid   178] set_robust_list(0x7f5779f55a20, 24 <unfinished ...>
[pid   177] close(4 <unfinished ...>
```

```
[pid   178] <... set_robust_list resumed>) = 0
[pid   177] <... close resumed>)        = 0
[pid   178] close(3 <unfinished ...>
[pid   177] wait4(-1,  <unfinished ...>
[pid   178] <... close resumed>)        = 0
[pid   178] dup2(5, 0)                  = 0
[pid   178] dup2(4, 1)                  = 1
[pid   178] execve("./child", ["/.child"], 0x7ffe739a3730 /* 19 vars */) = 0
[pid   178] brk(NULL)                   = 0x563668ca2000
[pid   178] arch_prctl(0x3001 /* ARCH_??? */, 0x7ffe42a75460) = -1 EINVAL (Недопустимый
аргумент)
[pid   178] mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -
1, 0) = 0x7f4baa88d000
[pid   178] access("/etc/ld.so.preload", R_OK) = -1 ENOENT (Нет такого файла или каталога)
[pid   178] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
[pid   178] newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=34303, ...}, AT_EMPTY_PATH) = 0
[pid   178] mmap(NULL, 34303, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f4baa884000
[pid   178] close(3)                    = 0
[pid   178] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
[pid   178] read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0\0"..., 832) = 832
[pid   178] 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\0\0"..., 784, 64) =
784
[pid   178] pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48
[pid   178] pread64(3,
"\4\0\0\0\24\0\0\0\3\0\0\0GNU\0i8\235HZ\227\223\333\350s\360\352,\223\340."..., 68, 896) = 68
[pid   178] newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2216304, ...}, AT_EMPTY_PATH) = 0
[pid   178] 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\0\0"..., 784, 64) =
784
[pid   178] mmap(NULL, 2260560, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7f4baa65c000
[pid   178] mmap(0x7f4baa684000, 1658880, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7f4baa684000
[pid   178] mmap(0x7f4baa819000, 360448, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1bd000) = 0x7f4baa819000
[pid   178] mmap(0x7f4baa871000, 24576, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x214000) = 0x7f4baa871000
[pid   178] mmap(0x7f4baa877000, 52816, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f4baa877000
[pid   178] close(3)                    = 0
[pid   178] mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS,
-1, 0) = 0x7f4baa659000
[pid   178] arch_prctl(ARCH_SET_FS, 0x7f4baa659740) = 0
[pid   178] set_tid_address(0x7f4baa659a10) = 178
[pid   178] set_robust_list(0x7f4baa659a20, 24) = 0
[pid   178] rseq(0x7f4baa65a0e0, 0x20, 0, 0x53053053) = 0
[pid   178] mprotect(0x7f4baa871000, 16384, PROT_READ) = 0
[pid   178] mprotect(0x563667c3d000, 4096, PROT_READ) = 0
[pid   178] mprotect(0x7f4baa8c7000, 8192, PROT_READ) = 0
[pid   178] prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) =
0
[pid   178] munmap(0x7f4baa884000, 34303) = 0
[pid   178] read(0, "9", 1)             = 1
```

```
[pid   178] read(0, "\r", 1)            = 1
[pid   178] read(0, "\n", 1)            = 1
[pid   178] write(1, "\t\0\0\0", 4)     = 4
[pid   178] read(0, "1", 1)             = 1
[pid   178] read(0, "0", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "2", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "4", 1)             = 1
[pid   178] read(0, "\r", 1)            = 1
[pid   178] read(0, "\n", 1)            = 1
[pid   178] write(1, "\20\0\0\0", 4)    = 4
[pid   178] read(0, "0", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "3", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "1", 1)             = 1
[pid   178] read(0, "2", 1)             = 1
[pid   178] read(0, "\r", 1)            = 1
[pid   178] read(0, "\n", 1)            = 1
[pid   178] write(1, "\17\0\0\0", 4)    = 4
[pid   178] read(0, "1", 1)             = 1
[pid   178] read(0, "3", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "1", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "3", 1)             = 1
[pid   178] read(0, "\r", 1)            = 1
[pid   178] read(0, "\n", 1)            = 1
[pid   178] write(1, "\21\0\0\0", 4)    = 4
[pid   178] read(0, "1", 1)             = 1
[pid   178] read(0, "2", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "7", 1)             = 1
[pid   178] read(0, "\r", 1)            = 1
[pid   178] read(0, "\n", 1)            = 1
[pid   178] write(1, "\23\0\0\0", 4)    = 4
[pid   178] read(0, "0", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "0", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "0", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "0", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "\r", 1)            = 1
[pid   178] read(0, "\n", 1)            = 1
[pid   178] write(1, "\0\0\0\0", 4)     = 4
[pid   178] read(0, "1", 1)             = 1
[pid   178] read(0, "7", 1)             = 1
[pid   178] read(0, "\r", 1)            = 1
[pid   178] read(0, "\n", 1)            = 1
```

```
[pid   178] write(1, "\21\0\0\0", 4)    = 4
[pid   178] read(0, "1", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "2", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "3", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "4", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "5", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "6", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "7", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "8", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "9", 1)             = 1
[pid   178] read(0, " ", 1)             = 1
[pid   178] read(0, "1", 1)             = 1
[pid   178] read(0, "0", 1)             = 1
[pid   178] read(0, "\r", 1)            = 1
[pid   178] read(0, "\n", 1)            = 1
[pid   178] write(1, "7\0\0\0", 4)      = 4
[pid   178] read(0, "", 1)              = 0
[pid   178] exit_group(0)               = ?
[pid   178] +++ exited with 0 +++
<... wait4 resumed>NULL, 0, NULL)       = 178
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=178, si_uid=1000, si_status=0,
si_utime=0, si_stime=2} ---
read(3, "\t\0\0\0", 4)                  = 4
write(1, "9\n", 29
)                       = 2
write(1, "\n", 1
)                       = 1
read(3, "\20\0\0\0", 4)                 = 4
write(1, "16\n", 316
)                       = 3
write(1, "\n", 1
)                       = 1
read(3, "\17\0\0\0", 4)                 = 4
write(1, "15\n", 315
)                       = 3
write(1, "\n", 1
)                       = 1
read(3, "\21\0\0\0", 4)                 = 4
write(1, "17\n", 317
)                       = 3
write(1, "\n", 1
)                       = 1
read(3, "\23\0\0\0", 4)                 = 4
write(1, "19\n", 319
```

```
)                       = 3
write(1, "\n", 1
)                       = 1
read(3, "\0\0\0\0", 4)                  = 4
write(1, "0\n", 20
)                       = 2
write(1, "\n", 1
)                       = 1
read(3, "\21\0\0\0", 4)                  = 4
write(1, "17\n", 317
)                       = 3
write(1, "\n", 1
)                       = 1
read(3, "7\0\0\0", 4)                   = 4
write(1, "55\n", 355
)                       = 3
write(1, "\n", 1
)                       = 1
read(3, "", 4)                          = 0
exit_group(0)                           = ?
+++ exited with 0 +++
```

Вывод

Благодаря выполнению данной работы я изучил принцип работы с каналами для межпроцессорного взаимодействия. Я немного пощупал процесс перенаправления ввода и вывода процесса, узнал, что такое файловый дескриптор и понял, что важно вовремя закрывать их. Во время выполнения лабораторной были 2 основные трудности: сначала мне было трудно написать рабочий парсер, но потом, по совету одногруппника, я все-таки смог это сделать. После этого у меня возникли вопросы по поводу работы функции write, так как я сталкиваюсь с ней впервые. Единственное, мне очень повезло с вариантом задания, сложить числа в строке я могу. В целом, я подчеркнул много нового для себя, что поможет мне в написании будущих более сложных кодов.