Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №1 по курсу**

**«Операционные системы»**

**УПРАВЛЕНИЕ ПРОЦЕССАМИ В ОС И ОБМЕН ДАННЫМИ МЕЖДУ НИМИ**

Студент: Корнев Максим Сергеевич

Группа: М8О–212Б–22

Вариант: 16

Преподаватель: Соколов Андрей Алексеевич

Оценка: \_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_

Москва, 2023.

**Постановка задачи**

## Цель работы

Целью является приобретение практических навыков в:

* Управление процессами в ОС
* Обеспечение обмена данными между процессами посредством труб

## Задание

Составить и отладить программу на языке Си, осуществляющую работу с процессами и взаимодействие между ними в одной из двух операционных систем. В результате работы программа (основной процесс) должен создать для решения задачи один или несколько дочерних процессов. Взаимодействие между процессами осуществляется через системные сигналы/события и/или каналы (pipe).

**Вариант 16)** Родительский процесс создает дочерний процесс. Первой строкой пользователь в консоль родительского процесса вводит имя файла, которое будет использовано для открытия File с таким именем на запись. Перенаправление стандартных потоков ввода-вывода показано на картинке выше. Родительский и дочерний процесс должны быть представлены разными программами. Родительский процесс принимает от пользователя строки произвольной длины и пересылает их в pipe1. Процесс child проверяет строки на валидность правилу. Если строка соответствует правилу, то она выводится в стандартный поток вывода дочернего процесса, иначе в pipe2 выводится информация об ошибке. Родительский процесс полученные от child ошибки выводит в стандартный поток вывода.

Правило проверки: строка должна оканчиваться на «.» или «;»

**Общие сведения о программе**

Программа компилируется из файла main.c. Также используется заголовочные файлы: stdio.h, stdlib.h, unistd.h, sys/wait.h, sys/fcntl.h. Программы собраны с помощью Cmake. В программе используются следующие системные вызовы:

1. read — считывает с консоли
2. write — записывает в консоль
3. exit — завершает программу
4. close — закрывает файл
5. dup2 — переназначение файлового дескриптора.
6. execvp — запускает исполняемый файл изнутри другой программы.
7. open — открывает файл.
8. pipe — создаёт одностороннюю трубу для передачи данных между процессами.
9. fork — создаёт дочерний процесс, который повторяет программу родительского процесса.

**Общий метод и алгоритм решения**.

Для реализации поставленной задачи необходимо:

1. Изучить принципы работы функции **fork()**, **pipe()**.
2. Создать **child.c** и **main.c**.
3. Реализовать в **main**:

3.1. Создание двух процессов и общение между программами **child** и **main** с помощью двух односторонних труб.

3.2. Прием ответа из **child** и вывод ответа.

1. Реализовать в **child**:

4.1. Считывание строки с помощью системного вызова read.

4.2. Функцию, которая будет решать задачу и проверять на что оканчивается строка.

4.3. Вывод значения с помощью системного вызова write.

**Основные файлы программы**

**main.c:**

#include <unistd.h>

#include <sys/wait.h>

#include <string.h>

#include <stdio.h>

#include <stdlib.h>

#include <fcntl.h>

int main() {

char FileName[256];

write(STDOUT\_FILENO, "Введите имя файла для записи: ", 55);

read(STDIN\_FILENO, FileName, 256);

FileName[strlen(FileName) - 1] = '\0';

int file = open(FileName, O\_WRONLY, 0777); //0777 uses for permission

if(!file) {

write(STDERR\_FILENO, "Opening error file", 19);

exit(-1);

}

int pipe1[2], pipe2[2];

pipe(pipe1);

pipe(pipe2);

pid\_t child\_pid = fork();

if (child\_pid == 0) {

// Дочерний процесс

close(pipe1[1]); //na zapis

close(pipe2[0]); //na chtenie

dup2(pipe1[0], STDIN\_FILENO);

dup2(pipe2[1], STDOUT\_FILENO);

execlp("./child", "./child", NULL);

\_exit(1);

} else {

// Родительский процесс

close(pipe1[0]);

close(pipe2[1]);

char \_String[256];

while (1) {

write(STDOUT\_FILENO, "Введите строку: ", 30);

int bytesRead = read(STDIN\_FILENO, \_String, sizeof(\_String));

write(pipe1[1], \_String, bytesRead);

int bytesReadChild = read(pipe2[0], \_String, sizeof(\_String));

write(STDOUT\_FILENO, "Строка удовлетворяет условию: ", 57);

write(STDOUT\_FILENO, \_String, bytesReadChild);

}

close(pipe1[1]);

close(pipe2[0]);

wait(NULL);

close(file);

}

return 0;

}

**child.c**

#include <unistd.h>

int main() {

char \_string[256];

int bytesRead;

while (1) {

bytesRead = read(STDIN\_FILENO, \_string, sizeof(\_string));

if (\_string[bytesRead - 2] == ';' || \_string[bytesRead - 2] == '.') {

// строка заканчивается на ";" или "."

write(STDOUT\_FILENO, \_string, bytesRead);

} else {

// строка не соответствует условию

write(STDERR\_FILENO, "Ошибка: строка не заканчивается на \";\" или \".\"\n", 80);

exit(-1);

}

}

return 0;

}

**CmakeLists.txt**

cmake\_minimum\_required(VERSION 3.10)

project(lab1\_os)

set(CMAKE\_C\_STANDARD 11)

add\_executable(child child.c)

set(REQUIRED\_FILES child)

add\_executable(main main.c)

**Пример работы**

radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: out.txt

Введите строку: asdaf;

Строка удовлетворяет условию: asdaf;

radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: out.txt

Введите строку: фвфыфа;

Строка удовлетворяет условию: фвфыфа;

radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: out.txt

Введите строку: sadaf.

Строка удовлетворяет условию: sadaf.

radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: out.txt

Введите строку: agsalsaf;.

Строка удовлетворяет условию: agsalsaf;.

radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: out.txt

Введите строку: hello world.

Строка удовлетворяет условию: hello world.

radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: out.txt

Введите строку: no ;

Строка удовлетворяет условию: no ; radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: out.txt

Введите строку: no

Ошибка: строка не заканчивается на ";" или "."

radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: as

Введите строку: asd

Ошибка: строка не заканчивается на ";" или "."

radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: task.txt

Введите строку: hello.txt;

Строка удовлетворяет условию: hello.txt;

radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: task.txt

Введите строку: aasdo;

Строка удовлетворяет условию: aasdo;

radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: task.txt

Введите строку: asfas.

Строка удовлетворяет условию: asfas.

radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: task.txt

Введите строку: kolkol.

Строка удовлетворяет условию: kolkol.

radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: task.txt

Введите строку: lll.

Строка удовлетворяет условию: lll.

radioactive@DESKTOP-RNP2IGB:/mnt/d/labs/os\_lab\_1/src$ ./main

Введите имя файла для записи: task.txt

Введите строку: ;;/

Ошибка: строка не заканчивается на ";" или "."

**Вывод**

Я изучил основные системные вызовы для работы с трубами и процессами. Научился запускать несколько процессов и обмениваться между ними с помощью pipe(). Также, для решения проблемы с отправлением информации в child программу, пришлось научиться пользоваться dup2.