Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №2 по курсу**

**«Операционные системы»**

Студент: Матвеев Данил

Группа: М8О-207Б-21

Вариант: 22

Преподаватель: Черемисинов Максим

Оценка: \_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_

Москва, 2022

**Содержание**

1. Репозиторий
2. Постановка задачи
3. Общие сведения о программе
4. Общий метод и алгоритм решения
5. Исходный код
6. Демонстрация работы программы
7. Выводы

**Репозиторий**

https://github.com/MrDenli/OsLabs

**Постановка задачи**

**Цель работы**

Приобретение практических навыков в:

 Управление процессами в ОС

 Обеспечение обмена данных между процессами посредством каналов

**Задание**

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.

8 вариант) В файле записаны команды вида:«число число число<endline>». Дочерний процесс   
производит деление первого числа команда, на последующие числа в команде, а результат   
выводит в стандартный поток вывода. Если происходит деление на 0, то тогда дочерний и   
родительский процесс завершают свою работу. Проверка деления на 0 должна осуществляться на   
стороне дочернего процесса. Числа имеют тип int. Количество чисел может быть произвольным.

**Общие сведения о программе**

Программа компилируется из файла main.cpp с помощью cmake. Дочерний процесс представлен в exec\_child.cpp

Системные вызовы:

* fork() - создание дочернего процесса
* int execpl(const char \*filename, char \*const argv[], char \*const ) - замена образа памяти процесса
* int pipe(int pipefd[2]) - создание неименованного канала для передачи данных между процессами

**Общий метод и алгоритм решения**

Родительский процес считывает имя файла, мы передаем в дочерний процесс с помощь pipe имя считанного файла, там мы открываем этот файл, считываем нужную нам информацию и проводим вычесления, в конце передаем вычисления в родительский процесс с помощью pipe и там выводим ответ.

**Исходный код**

**main.cpp**

#include <unistd.h>

#include <iostream>

#include <fcntl.h>

#include <fstream>

using namespace std;

int main(){

string file, name;

cin >> name;

int rez;

char \*const child\_args[] = { "./child", NULL };

int pipefd[2];

int pip = pipe(pipefd);

if (pip == -1) {

cout << "error with oppening the pipe";

return 0;

}

int id = fork();

if (id == -1) {

cout << "error with fork";

return 0;

} else if (id == 0) {

execlp(child\_args[0], to\_string(pipefd[0]).c\_str(), to\_string(pipefd[1]).c\_str(), name.c\_str(), NULL);

} else {

while (read(pipefd[0], &rez, sizeof(int))) {

if (rez == -1) {

break;

}

cout << rez << endl;

}

}

close(pipefd[0]);

close(pipefd[1]);

return 0;

}

**child.cpp**

**#include <unistd.h>**

**#include <sstream>**

**#include <iostream>**

**#include <fstream>**

**#include <string>**

**using namespace std;**

**int get\_number(string &s) {**

**if (s.size() == 0) {**

**return -1;**

**}**

**if (s[s.size() - 1] != ' ') {**

**s += ' ';**

**}**

**int ans = 0;**

**int i = 0;**

**string temp;**

**while (s[i] != ' ') {**

**temp += s[i];**

**i++;**

**}**

**s.erase(s.begin(), s.begin() + i + 1);**

**ans = stoi(temp);**

**return ans;**

**}**

**int main(int argc, char \*argv[]){**

**int pipefd[2];**

**string file\_name = argv[2];**

**pipefd[0] = atoi(argv[0]);**

**pipefd[1] = atoi(argv[1]);**

**//cout << file\_name;**

**string s;**

**ifstream fin(file\_name);**

**//fin >> s;**

**while(!fin.eof()) {**

**getline(fin, s);**

**// << s << endl;**

**//cout << s << endl;**

**int a = get\_number(s);**

**bool error = false;**

**while(true) {**

**int del = get\_number(s);**

**if (del == -1) {**

**break;**

**}**

**if (del == 0) {**

**error = true;**

**break;**

**}**

**a /= del;**

**}**

**//cout << a << endl;**

**if (!error) {**

**write(pipefd[1], &a, sizeof(int));**

**}**

**}**

**fin.close();**

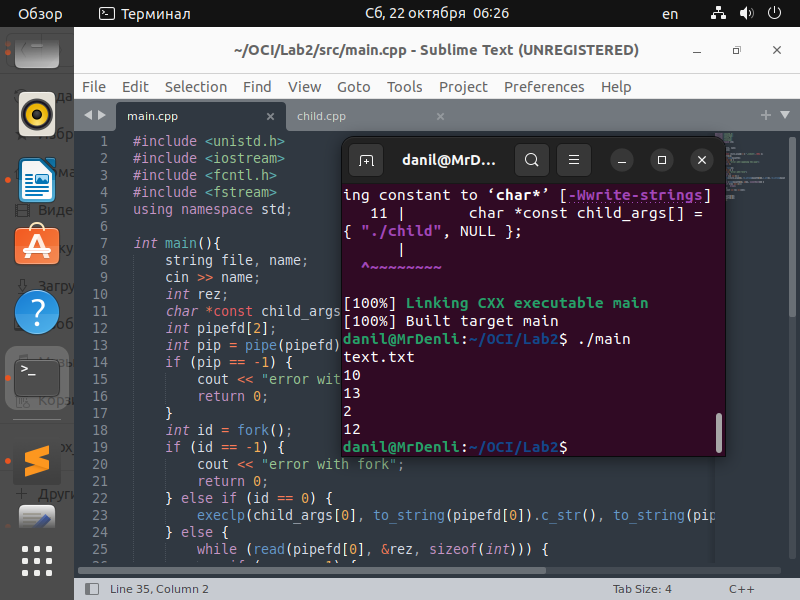
**close(pipefd[0]);**

**close(pipefd[1]);**

**return 0;**

**}**

**Демонстрация работы программы**



**Выводы**

Мне понравилась данная лабораторная работа, я научился работать с процессами в языке с++, а именно разделять процессы, взаимодействовать между эти процессы. Я считаю эта лабораторная работа очень полезна для меня, потому что полученные навыки с большой вероятностью помогут мне в будущем.