Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №1 по курсу**

**«Операционные системы»**

Студент: Велиев Рауф Рамиз оглы

Группа: М8О-209Б-23

Вариант: 4

Преподаватель: Миронов Евгений Сергеевич

Оценка: \_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_

Москва, 2024

**Содержание**

* Репозиторий
* Постановка задачи
* Исходный код
* Демонстрация работы программы
* Выводы

**Репозиторий**

<https://github.com/velievrauf/OS/tree/main/lab_1>

**Постановка задачи**

**Цель работы**

Приобретение практических навыков управления процессами в ОС и обеспечения обмена данными между процессами посредством каналов.

**Задание**

Составить и отладить программу, осуществляющую работу с процессами и взаимодействие между ними в одной из двух ОС.

Пользователь вводит команды вида: «число число число». Далее эти числа передаются от родительского процесса в дочерний. Дочерний процесс производит деление parent child(args: fileName) pipe1 pipe2 In/out User File In Out первого числа, на последующие, а результат выводит в файл. 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 float. Количество чисел может быть произвольным.

**Исходный код**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/wait.h>

int main() {

int pipe1[2], pipe2[2];

pid\_t pid;

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

perror("pipe");

exit(EXIT\_FAILURE);

}

pid = fork();

if (pid < 0) {

perror("fork");

exit(EXIT\_FAILURE);

}

if (pid == 0) {

close(pipe1[1]);

close(pipe2[0]);

float numbers[100];

int count = read(pipe1[0], numbers, sizeof(numbers));

close(pipe1[0]);

for (int i = 1; i < count / sizeof(float); i++) {

if (numbers[i] == 0) {

printf("Ошибка: деление на ноль\n");

exit(EXIT\_FAILURE);

}

}

float result = numbers[0];

for (int i = 1; i < count / sizeof(float); i++) {

result /= numbers[i];

}

int file = open("result.txt", O\_WRONLY | O\_CREAT, 0644);

if (file < 0) {

perror("open");

exit(EXIT\_FAILURE);

}

dprintf(file, "Результат: %.2f\n", result);

close(file);

exit(EXIT\_SUCCESS);

} else {

close(pipe1[0]);

close(pipe2[1]);

float numbers[100];

int count = 0;

printf("Введите числа (разделенные пробелами), завершите <endline>: ");

while (scanf("%f", &numbers[count]) != EOF) {

count++;

}

write(pipe1[1], numbers, sizeof(numbers[0]) \* count);

close(pipe1[1]);

wait(NULL);

}

return 0;

}

**Демонстрация работы программы**

Введите числа (разделенные пробелами), завершите <endline>: 100 2 10 5 -2

Результат: -0.50

Введите числа (разделенные пробелами), завершите <endline>: -15 2 0

Результат: Ошибка: деление на ноль

**Выводы**

В результате выполнения лабораторной работы я освоил процесс разработки программ, в которых несколько процессов обмениваются данными через каналы. Я научился передавать данные между процессами, обрабатывать исключительные ситуации (например, деление на ноль) и записывать результаты вычислений в файл. Работа продемонстрировала, как эффективно использовать каналы для взаимодействия процессов и организовывать корректное управление ресурсами.