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»

Выполнил: Д. М. Мишин
Группа: М8О-207БВ-24
Преподаватель: Е. С. Миронов

Москва, 2025

Условие

Цель работы:
Приобретение практических навыков в:

- Управлении процессами в ОС
- Обеспечение обмена данных между процессами посредством каналов

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 решение задачи один или несколько дочерних процессов. Взаимодействие между процессами осуществляется через системные сигналы/-события и/или каналы (pipe). Необходимо обрабатывать системные ошибки, которые могут возникнуть в результате работы.

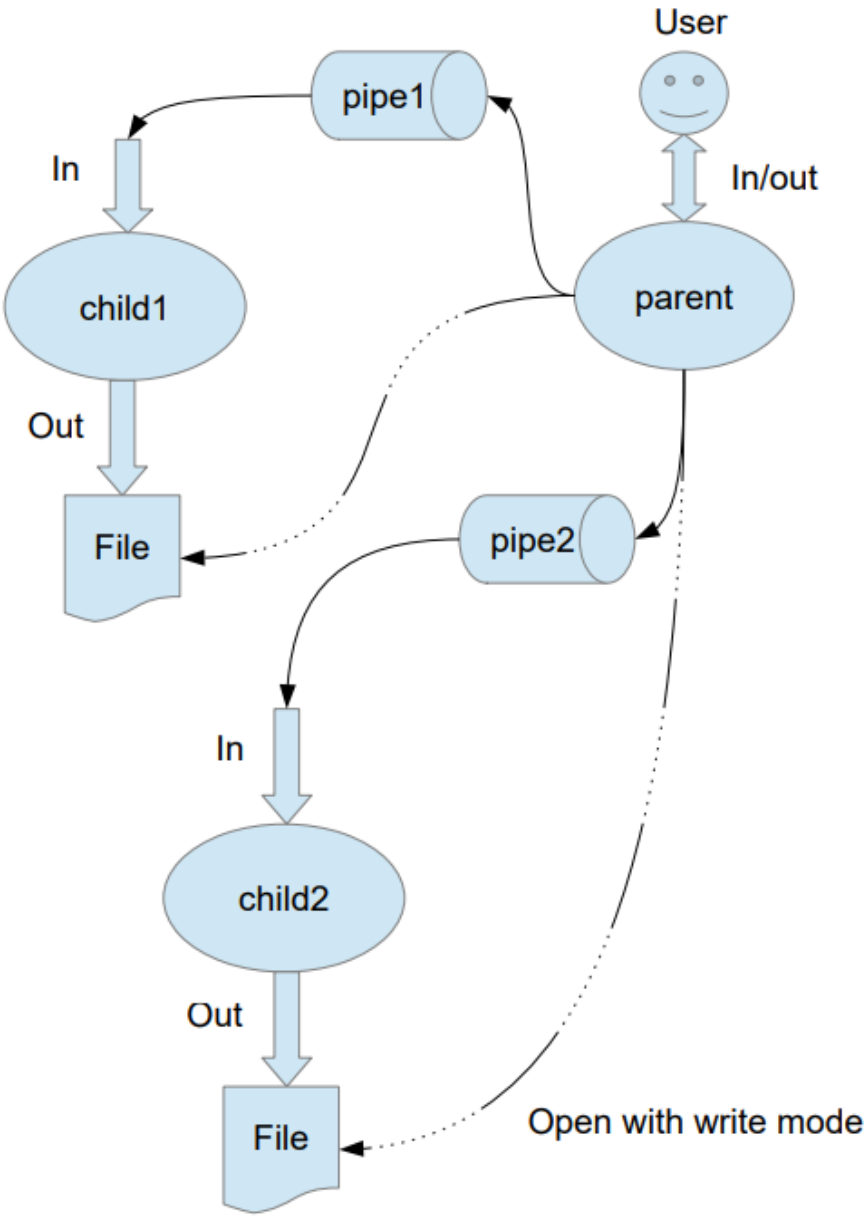

Рис. 1: Схема работы процессов.

Вариант: 22
Родительский процесс создает два дочерних процесса. Первой строкой пользователь в консоль родительского процесса вводит имя файла, которое будет использовано для открытия File с таким именем на запись для child1. Аналогично для второй строки и процесса child2. Родительский и дочерний процесс должны быть представлены разными программами. Родительский процесс принимает от пользователя строки произвольной длины и пересылает их в pipe1 или в pipe2 в зависимости от правила фильтрации. Процесс child1 и child2 производят работу над строками. Процессы пишут результаты своей работы в стандартный вывод. Правило фильтрации: с вероятностью 80% строки отправляются в pipe1, иначе в pipe2. Дочерние процессы инвертируют строки.

Метод решения
Алгоритм решения задачи:

1. Пользователь в консоль родительского процесса вводит имена файлов, которые будет использовано для открытия (создания) файлов с таким именем для child1 и child2 соответственно.

2. Создается объект класса Parent, так называемый "родитель".

3. У родителя вызывается метод "CreateChildProcesses() родительский процесс создает два канала (pipe) для передачи данных между родительским и дочерними процессами. Затем делает первый fork для создания дочернего процеса. Первый дочерний процесс закрывает соответсвующий канал для записи, "перенаправляет"второй "конец"канала для чтения на стандартный ввод, закрывает второй "конец"канала. Дочерний процесс пытается запустить бинарный файл для дочернего процесса, если не получилось — завершается. Аналогично для второго дочернего процесса.

4. Родительский процесс закрывает "концы"каналов для чтения и отправляет имена файлов для их открытия (создания) дочерними процессами.

5. Каждый из дочерних процессов пытается открыть (создать) указанный файл в спец. директории, если не получилось — процесс завершается.

6. Родительский процесс проверяет "жизнидеятельность"процессов, если любой из них уже завершен, то программа прекращает работу со статусом ошибки, затем считывает пользовательский ввод и отправляет дочернему процессу через pipe по правилу фильтрации.

7. Дочерний процесс инвертирует строку и записывает её в соответствующий файл.

8. По команде "exit"или "quit"родитель останавливает дочерние процессы и программа завершается.

Ссылки:
`https://pubs.opengroup.org/onlinepubs/009696799/functions/write.html`
`https://pubs.opengroup.org/onlinepubs/009696799/functions/execl.html`
`https://pubs.opengroup.org/onlinepubs/009696799/functions/getppid.html`
`https://pubs.opengroup.org/onlinepubs/009696799/functions/getpid.html`

https://pubs.opengroup.org/onlinepubs/009696799/functions/waitpid.html
https://pubs.opengroup.org/onlinepubs/009696799/functions/sleep.html
https://pubs.opengroup.org/onlinepubs/009696799/functions/exit.html
https://pubs.opengroup.org/onlinepubs/009696799/functions/kill.html
https://pubs.opengroup.org/onlinepubs/009696799/functions/dup2.html
https://pubs.opengroup.org/onlinepubs/009696799/functions/pipe.html
https://pubs.opengroup.org/onlinepubs/009696799/functions/fork.html
https://pubs.opengroup.org/onlinepubs/009696799/functions/close.html

Описание программы

Разделение по файлам, описание основных типов данных и функций. Обязательно написать используемые системные вызовы.

Результаты

Описать полученное решение, ключевые особенности. В ЛР с потоками привести график замеров.

Выводы

Исходная программа

Сюда вставляем ваш код (весь)

```
#include "parent.h"

int main() {
    std::string filename1;
    std::string filename2;
    std::cout << "Parent[" << os::GetPid() << "]: Enter filename for child1" << std::endl
        ;
    std::getline(std::cin, filename1);
    std::cout << "Parent[" << os::GetPid() << "]: Enter filename for child2" << std::endl
        ;
    std::getline(std::cin, filename2);
    parent::Parent p;
    try {
        p.CreateChildProcesses(filename1, filename2);
        p.Work();
        p.EndChildren();
    } catch (const exceptions::CreatePipeException&) {
        std::cout << "Parent[" << os::GetPid() << "]: Failed to create pipe. Ending
            programm." << std::endl;
        return 1;
    } catch (const exceptions::ChildProcessEndException&) {
        std::cout << "Parent[" << os::GetPid() << "]: Child process is ended with error.
            Ending programm." << std::endl;
        return 1;
    }
    return 0;
}
```

Листинг 1: *Сюда описание*

А таким образом можно вставлять "сырой"код в виде текста

```
#include <iostream>

int main() {
    std::cout << "Hello, world!" << std::endl;
    return 0;
}
```

Также сюда необходимо будет вставить strace (утилита, которая будет выводить системные вызовы вашей программы)