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: Хомяков Иван Андреевич

Группа: М8О-207Б-21

Вариант: Foo Bar

Преподаватель: Миронов Евгений Сергеевич

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/EbumbaE/OS\_LAB/lab2

**Постановка задачи**

**Цель работы**

Приобретение практических навыков в:

* Управление 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.

Блок вариантов 5: 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.

Вариант 19: Правило фильтрации: с вероятностью 80% строки отправляются в pipe1, иначе в pipe2. Дочерние процессы удаляют все гласные из строк.

**Общие сведения о программе**

Программа компилируется из файла main.c. Также используется заголовочный файл: main.h, который описывает функцию и подключает include.h. Также программа собирает child1 и child2, которые используют include.h и workWithFile.h (подключает нужные для работы с файлами библиотеки и описывает несколько функций). workWithFile.c реализует функции, описываемые в workWithFile.h.

В программе используются следующие системные вызовы:

1. CreatePipe – создает канал для общения между программами, принимает два
2. SetHandleInformation – проверяет правильность установленных атрибутов, в данной ситуации, что канал ввода не наследуется (канал вывода нас не интересует)
3. ZeroMemory – заполняет блок памяти нолями
4. CreateProcess – создает процесс, принимает кучу параметров, важные из них (szCmdline – командная строка, запускающая child.exe; TRUE – наследуем pipe; siStartInfo – сами pipe; siProcInfo – устанавливаем информацию о процессе)
5. WriteFile – пишем в pipe
6. ReadFile – читаем из pipe
7. GetStdHandle – получаем pipe от главного процесса
8. CloseHandle – закрываем pipe

**Общий метод и алгоритм решения**

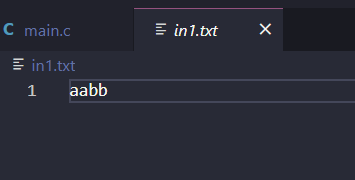
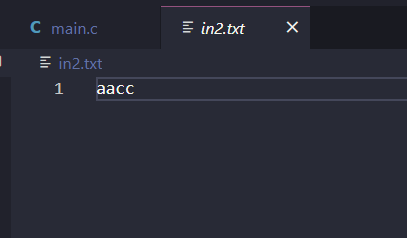
Родительский процесс создает два дочерних процесса child1 и child2. Первой строкой пользователь в консоль родительского процесса вводит имя файла, которое будет использовано child1. Аналогично для второй строки и процесса child2. Родительский и дочерниt процессы представлены разными программами. Родительский с вероятностью 80% выбирает child1, иначе child2, он передает дочернему процессу имя файла. Дочерний процесс открывает два файла, один читает, в другой записывает только согласные буквы. После выполнения, дочерний процесс закрывает каналы для связи.

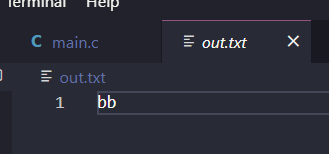
**Исходный код**

В репозитории.

**Демонстрация работы программы**





**Выводы**

Научился создавать дочерние процессы и каналы для их общения.