МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

ФЕДЕРАЦИИ

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»

Студент: Григорьев Т. А.

Группа: М8О-208Б-23

Преподаватель: Живалев Е. А.

Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2024

**Тема:** Управление процессами и межпроцессное взаимодействие в ОС

**Цель работы:** Приобретение практических навыков в:

* Управлении процессами в операционной системе.
* Организации обмена данными между процессами посредством каналов (pipe).

**Вариант:** 18. Правило фильтрации: нечетные строки отправляются в pipe1, четные в pipe2. Дочерние процессы удаляют все гласные из строк

**Задачи:**

1. Разработать программу на языке Си, реализующую управление процессами и их взаимодействие через каналы.
2. Реализовать фильтрацию строк, вводимых пользователем, для передачи их в соответствующие процессы через каналы.
3. Обеспечить обработку ошибок, возникающих при выполнении программы.
4. Выполнить удаление гласных в строках в дочерних процессах и записать результат в файл.

**Описание решения:** Программное решение состоит из трех основных модулей:

* **Parent (родительский процесс):** создает два дочерних процесса и организует межпроцессное взаимодействие через каналы.
* **Child1 и Child2 (дочерние процессы):** принимают строки из соответствующих каналов, удаляют гласные и записывают результат в указанные пользователем файлы.

Программа функционирует следующим образом:

1. Родительский процесс создает два канала и выполняет два вызова fork для создания дочерних процессов.
2. После создания каналов пользователь вводит два имени файлов, которые передаются в качестве аргументов дочерним процессам.
3. Родительский процесс принимает строки произвольной длины от пользователя. Фильтрация строк осуществляется на основании их индекса: нечетные строки отправляются в первый канал (pipe1), четные — во второй (pipe2).
4. Дочерние процессы (Child1 и Child2):
   * Читают строки из соответствующего канала.
   * Выполняют удаление гласных с использованием вспомогательной функции RemoveVowels.
   * Записывают полученные строки в указанные файлы.
5. Обработка ошибок предусмотрена на всех этапах работы программы (например, ошибки открытия файлов, создания процессов, записи и чтения данных).

**Исходный код:** Программное обеспечение состоит из следующих файлов:

1. **main.c**: Инициализация родительского процесса.
2. **parent.c**: Основная логика родительского процесса (создание каналов, управление дочерними процессами, фильтрация строк).
3. **child1.c и child2.c**: Логика дочерних процессов (обработка строк и запись в файл).
4. **utils.c**: Вспомогательная функция для инвертирования строк.

**Репозиторий:** <https://github.com/timofeez/os-labs/tree/main/01>

**Пример кода:**

// Пример функции RemoveVowels из utils.c

void RemoveVowels(char\* str) {

char\* src = str;

char\* dst = str;

while (\*src) {

char lower = tolower((unsigned char)\*src);

if (lower != 'a' && lower != 'e' && lower != 'i' && lower != 'o' && lower != 'u') {

\*dst++ = \*src;

}

src++;

}

\*dst = '\0';

}

**Пример работы:**

./lab1

first

second

abc

abcd

q

// first

bc

// second

bcd

**Вывод:** В рамках выполнения лабораторной работы все цели и задачи были успешно достигнуты. Разработанная программа корректно создает два дочерних процесса и организует обмен данными между ними и родительским процессом с использованием каналов. Функционал фильтрации строк, удаления гласных и записи результатов в файлы реализован и работает без ошибок. В процессе работы приобретены практические навыки управления процессами, работы с межпроцессным взаимодействием и обработки ошибок в операционной системе. Программа протестирована в среде Linux и продемонстрировала надежность и стабильность работы.