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: Ширшов Д. К.

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:** 19. Правило фильтрации: с вероятностью 80% строки отправляются в pipe1, иначе 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 из строк в дочерних процессах и записать результаты в соответствующие файлы.

**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 рандомайзера чисел: строки которым выпали числа кратные 80 идут в первый канал (pipe1), которым выпали числа кратные 20 — во второй (pipe2).
4. Дочерние процессы (Child1 и Child2):
   * Читают строки из соответствующего канала.
   * Выполняют удаление гласных используя вспомогательную функцию RemoveVowels.
   * Записывают инвертированные 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 удаления гласных из строк.

**Репозиторий:** https://github.com/TwixXleb/labs\_os/tree/main/LW1

**Пример кода:**

void remove\_vowels(char\* str) {

if (str == NULL) {

return;

}

int i = 0, j = 0;

while (str[i] != '\0') {

char lower\_char = tolower(str[i]);

if (lower\_char != 'a' && lower\_char != 'e' && lower\_char != 'i' &&

lower\_char != 'o' && lower\_char != 'u' && lower\_char != 'y') {

str[j++] = str[i];

}

i++;

}

str[j] = '\0';

}

**Пример работы:**

./lab1

output1.txt

output2.txt

abba

aaaaa

qqqqq

gagaga

// first

Bb

ggg

// second

qqqqq

**Вывод:** В ходе выполнения лабораторной работы были выполнены все поставленные задачи. Программа успешно создает два дочерних процесса и организует обмен данными между ними и родительским процессом посредством каналов. Фильтрация строк, удаление из них гласных и запись в файлы выполняются корректно. Были приобретены практические навыки в работе с процессами, каналами и обработке ошибок в операционных системах. Программа протестирована на операционной системе Linux и показала стабильную работу.