ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ (НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

**ОТЧЕТ**

**О ВЫПОЛНЕНИИ 3 ЛАБОРАТОРНОЙ РАБОТЫ**

**ПО ДИСЦИПЛИНЕ «ОПЕРАЦИОННЫЕ СИСТЕМЫ»**

Выполнил(а) студент группы М8О-208Б-23

Федорова Екатерина Васильевна

Проверили и приняли:

Живалев Е.А. Катаев Ю. И.

Москва, 2024

**Тема:** «Взаимодействие процессов через отображаемые файлы (memory-mapped files)»

**Цели работы:**

* Приобретение практических навыков в использовании технологий работы с файловыми системами.
* Освоение принципов взаимодействия между процессами через отображаемые файлы (memory-mapped files).
* Обеспечение обмена данными между процессами с использованием системных вызовов Linux.
* Обработка системных ошибок, возникающих в процессе работы.

**Задание:** реализовать программу на языке C, которая осуществляет взаимодействие между процессами через отображаемые файлы. Основной процесс создает два дочерних процесса. Родительский процесс принимает от пользователя строки произвольной длины и пересылает их в один из двух каналов (с вероятностью 80% в первый канал, 20% во второй). Дочерние процессы обрабатывают строки, инвертируют их и записывают результаты в файлы, имена которых задаются пользователем. Взаимодействие между процессами осуществляется через отображаемые файлы (memory-mapped files).

**Вариант 22**: Правило фильтрации: с вероятностью 80% строки отправляются в первый канал, иначе во второй. Дочерние процессы инвертируют строки.

**Описание решения:**

1. **Родительский процесс**:

- Создает два объекта разделяемой памяти с помощью `shm\_open` и отображает их в адресное пространство с помощью `mmap`.

- Принимает от пользователя два имени файлов, которые будут использоваться дочерними процессами для записи результатов.

- Создает два дочерних процесса с помощью `fork()`.

- Читает строки из стандартного ввода и пересылает их в один из двух объектов разделяемой памяти в зависимости от правила фильтрации (80% в первый, 20% во второй).

- Сигнализирует дочерним процессам о завершении работы, отправляя специальный сигнал (пустую строку).

2. **Дочерние процессы**:

- Получают доступ к соответствующему объекту разделяемой памяти.

- Читают строки из разделяемой памяти, инвертируют их и записывают результаты в файл, имя которого передается в качестве аргумента.

- После обработки строки сигнализируют родительскому процессу о готовности к приему следующей строки.

3. **Синхронизация**:

- Для синхронизации используется флаг `ready` в структуре данных, хранящейся в разделяемой памяти. Родительский процесс ждет, пока дочерний процесс обработает текущую строку, прежде чем отправить следующую.

**Заключение:** в ходе работы была реализована программа, обеспечивающая взаимодействие между процессами через отображаемые файлы (memory-mapped files). Программа успешно выполняет поставленные задачи: распределяет строки между каналами, обрабатывает их в дочерних процессах и записывает результаты в файлы. Были приобретены практические навыки работы с файловыми системами, системными вызовами Linux и синхронизацией процессов. Программа продемонстрировала стабильную работу и корректное выполнение всех требований задания.

**Вывод:** работа позволила освоить технологии взаимодействия процессов через отображаемые файлы. Программа успешно реализует обмен данными между процессами, обработку строк и запись результатов в файлы.