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: Попов А. Д.
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: 7. Родительский процесс создает дочерний процесс, передает ему имя файла для чтения, перенаправляет его поток вывода в pipe1, а затем читает данные из pipe1 и выводит их в стандартный поток вывода. Родительский и дочерний процессы реализованы как отдельные программы.

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 вычисление суммы чисел в строках и записать полученный результат в канал.

Описание решения: Программное решение состоит из трех основных модулей:

1. **Основной файл,** который инициализирует программу и запрашивает у пользователя имя файла для обработки.
2. **Файл с описанием родительского процесса.** Содержит функции для создания дочерних процессов, чтения данных из каналов и выполнения основных операций.
3. **Файл с описанием дочернего процесса.** Реализует логику обработки данных в дочернем процессе, включая чтение строк из файла, вычисление суммы чисел в строках и запись результата в канал.

Программа функционирует следующим образом:

1. Родительский процесс запрашивает у пользователя имя файла, который будет обработан.
2. Родительский процесс создает канал и дочерний процесс с помощью функции fork().

3. Дочерний процесс выполняет программу divergent, которая читает данные из указанного файла, вычисляет сумму чисел в каждой строке и записывает результат в канал.
4. Родительский процесс читает данные из канала и выводит их на экран.
5. Обработка ошибок предусмотрена на всех этапах работы программы, включая ошибки открытия файлов, создания процессов, записи и чтения данных.

Исходный код: Программное обеспечение состоит из следующих файлов:

1. **main.c**: Запрос входного файла у пользователя и инициализация родительского процесса.
2. **lab_1_utils.c**: Основная логика родительского процесса.
3. **divergent.c**: Логика дочернего процесса.
4. **lab1.h**: Заголовочный файл.

Репозиторий: https://github.com/aldpopov/OS_labs/tree/master/LW1

Пример кода:

```
// Пример функции ProcessData из divergent.c
void ProcessData(int writeFd) {
    float num, sum = 0;
    char buffer[1024];
    char *token;

    while (fgets(buffer, sizeof(buffer), stdin) != NULL) {
        char *line = strtok(buffer, "\n");
        while (line != NULL) {
            sum = 0;
            token = strtok(line, " ");
            while (token != NULL) {
                num = atof(token);
                sum += num;
                token = strtok(NULL, " ");
            }
            char resultStr[1024];
            snprintf(resultStr, sizeof(resultStr), "Sum: %.2f\n", sum);
            ssize_t     bytesWritten     =     write(writeFd,     resultStr,
strlen(resultStr));
            if (bytesWritten == -1) {
                fprintf(stderr, "Write error: %s\n", strerror(errno));
```

```
                exit(1);
            } else if (bytesWritten < strlen(resultStr)) {
                fprintf(stderr, "Warning: written only %zd bytes from %zu\n",
bytesWritten, strlen(resultStr));
            }
            line = strtok(NULL, "\n");
        }
    }
    close(writeFd);
}
```

Пример работы:

```
make run

Enter file's name:

test.txt

Sum: 10.00

Sum: 15.00

Sum: 20.00
```

Вывод: В ходе выполнения лабораторной работы были выполнены все поставленные задачи. Программа успешно создает дочерний процесс и организует обмен данными между родительским и дочерним процессами посредством каналов. Обработка данных, вычисление суммы чисел в строках и вывод результата 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.