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»

Группа: М8О-210Б-23

Студент: Стаценко В.А.

Преподаватель: Бахарев В.Д.

Оценка: _____

Дата: 7.11.24

Москва, 2024

Постановка задачи

Вариант 1.

Пользователь вводит команды вида: «число число число». Далее эти числа передаются от родительского процесса в дочерний. Дочерний процесс считает их сумму и выводит её в файл. Числа имеют тип int. Количество чисел может быть произвольным.

Общий метод и алгоритм решения

Использованные системные вызовы:

- pid_t fork(void); – создает дочерний процесс.
- int pipe(int pipefd[2]) - создает наименованный канал для передачи данных между процессами
- void exit(int status) - завершает выполнение процесса и возвращение статуса
- int dup2(int oldfd, int newfd) - переназначает файловый дескриптор
- int open(const char* pathname, int flags, mode_t mode) - открывает\создает файл
- int close(int fd) - закртывает файл
- int execv(const char *path, char* const argv[]) - заменяет текущий исполняемый файл на path
- int write(int fd, void* buffer, int count) - записывает данные в файл, связанный с файловым дескриптором
- int read(int fd, void* buffer, int count) - читает данные из файла, связанного с файловым дескриптором
- pid_t wait(int status) - ожидает завершение дочернего процесса

В данной лабораторной работе я написала программу, состоящую из двух процессов: родительского и дочернего, которые взаимодействуют друг с другом с помощью канала (pipe). Родительский процесс запрашивает ввод чисел и передает их дочернему процессу для обработки. Дочерний процесс читает данные из канала, вычисляет сумму введенных чисел каждой новой строки, пока не встретит end, и записывает результаты в указанный файл. Программа включает в себя обработку ошибок, таких как отсутствие аргументов командной строки и сбои при создании процессов и открытии файлов.

Код программы

parent.c:

```c
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>

int main(int argc, char *argv[]) {
    int pipe1[2];
    pid_t pid;
    char buffer[100];

    if (argc < 2) {
        char error_msg[100];
        snprintf(error_msg, sizeof(error_msg), "Необходимо указать имя файла в качестве аргумента.\n");
        write(STDERR_FILENO, error_msg, strlen(error_msg));
        exit(EXIT_FAILURE);
    }

    if (pipe(pipe1) == -1) {
        perror("pipe");
        exit(EXIT_FAILURE);
    }

    pid = fork();

    if (pid == -1) {
        perror("fork");
        exit(EXIT_FAILURE);

    } else if (pid == 0) {
        close(pipe1[1]);
        dup2(pipe1[0], STDIN_FILENO);
        close(pipe1[0]);
        char *args[] = {"./child", argv[1], NULL};
        execv(args[0], args);
        perror("execv");
        exit(EXIT_FAILURE);
```

```c
    } else {
        close(pipe1[0]);
        while (1) {
            write(STDOUT_FILENO, "Введите числа (или end для завершения): ", strlen("Введите числа (или end для завершения): "));
            read(STDIN_FILENO, buffer, sizeof(buffer));
            buffer[strcspn(buffer, "\n")] = 0;

            if (strcmp(buffer, "end") == 0) {
                break;
            }

            write(pipe1[1], buffer, strlen(buffer));
            write(pipe1[1], "\n", 1);
        }

        close(pipe1[1]);
        wait(NULL);
    }

    return 0;
}
```

child.c:

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>

int main(int argc, char *argv[]) {
    char buffer[4096];
    char *token;
    int sum;

    if (argc < 2) {
        char error_msg[128];
        snprintf(error_msg, sizeof(error_msg), "Необходимо указать имя файла в качестве аргумента.\n");
        write(STDERR_FILENO, error_msg, strlen(error_msg));
        exit(EXIT_FAILURE);
    }

    char *filename = argv[1];

    while (fgets(buffer, sizeof(buffer), stdin) != NULL) {
        buffer[strcspn(buffer, "\n")] = 0;
        sum = 0;
        token = strtok(buffer, " ");
        while (token != NULL) {
            sum += atoi(token);
            token = strtok(NULL, " ");
        }

        int fd = open(filename, O_WRONLY | O_CREAT | O_APPEND, 0644);
        if (fd == -1) {
            perror("open");
            exit(1);
        }
        char sum_str[20];
        snprintf(sum_str, sizeof(sum_str), "%d\n", sum);
        write(fd, sum_str, strlen(sum_str));
        close(fd);
    }
    return 0;
}
```