Лабораторная работа №1

по курсу "Операционные системы"

Выполнил: Юнусов Рустам М8О-310Б-22. 5 Вариант

Преподаватель: Миронов Евгений Сергеевич

Задание:

Пользователь вводит команды вида: «число». Далее это число передается от родительского процесса в дочерний. Дочерний процесс производит проверку на простоту. Если число составное, то в это число записывается в файл. Если число отрицательное или простое, то тогда дочерний и родительский процессы завершаются.

Листинг программы

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

int create_process();

const char END_LINE = '\n';

uint8_t is_prime_or_negative(int number) {
    if(number < 0) {
        return 1;
    }
    for(unsigned long long i = 2; i*i <= number; ++i){
        if( number % i == 0){
            return 0;
        }
    }
    return 1;
}

int write_to_file(FILE* file, int number){
    char result[32];
    sprintf(result, "%d\n", number);
    int bytes = fwrite(result, strlen(result), 1, file);
    if(1 != bytes){
        return -1;
```

```
    }
    return 0;
}

int main() {
    int pipe_fd[2];
    int err = pipe(pipe_fd);
    if (err == -1)
    {
        perror("pipe");
        return -1;
    }

    pid_t pid = create_process();
    if (0 == pid) {
        close(pipe_fd[1]);
        FILE* output = fopen("composite_numbers.txt", "w");
        if(output == NULL){
            perror("file error");
            return -1;
        }
        int n = 0;
        for(;;) {
            read(pipe_fd[0], &n, sizeof(n));
            int ok = is_prime_or_negative(n);
            if(ok == 0) {
                err = write_to_file(output, n);
                if( err < 0){
                    perror("error file write");
                    return -1;
                }
            }else{
                break;
            }
        }
        fclose(output);
        close(pipe_fd[0]);
        exit(EXIT_SUCCESS);
    } else {
        close(pipe_fd[0]);
        int n = 0;
        for(;;){
            printf("Enter number: ");
            scanf("%d", &n);
            write(pipe_fd[1], &n, sizeof(n));
            int ok = is_prime_or_negative(n);
            if(ok) {
                break;
            }
        }
        wait(NULL);
        exit(EXIT_SUCCESS);
    }
```

```
    return 0;
}

int create_process() {
    pid_t pid = fork();
    if (-1 == pid)
    {
        perror("fork");
        exit(-1);
    }
    return pid;
}
```

Вывод

Программа организует обмен данными между родительским и дочерним процессами через pipe. Родительский процесс отправляет числа, дочерний проверяет их: составные записывает в файл, а при вводе простого или отрицательного числа оба процесса завершаются. Результат — файл с составными числами, пока не введено число для завершения. Это была полезная лабораторная работа, на которой мне удалось научить общаться процессы между собой.