Operating Systems. Homework 4

Operating Systems
Александр Васюков | БПИ235

Задание

Разработать программу использующую для работы с текстовыми файлами только системные вызовы. Программа на языке C (или C++ в стиле C) должна прочитать, используя буфер, размер которого превышает читаемые файлы и записать прочитанный файл в файл с другим именем. Имена файлов для чтения и записи задавать с использованием аргументов командной строки.

Вместо большого буфера использовать для работы с файлами буфер ограниченного размера, требующий циклического использования как для чтения, так и для записи.

Читать и переписывать не только текстовые, но и исполняемые файлы (текстовые и бинарные, содержащие признак исполнимости), включая скрипты, которые сохраняют режим доступа исходных файлов, обеспечивающий их запуск. При этом обычные текстовые файлы запускаться не должны. Для них режим доступа должен оставаться прежним.

Решение на 10 баллов

Код и отчёт на гитхабе: https://github.com/vasyukov1/HSE-FCS-SE-2-year/tree/main/Operating%20Systems/Homeworks/04

Программа работает следующим образом:

1. Проверяется, что в аргументах даны названия файлов.
2. Открывается файл, который будем читать.
3. Получаем информацию о файле, чтобы знать его режим доступа.
4. Открываем файл для записи.
5. В цикле читает в буффер данные из первого файла и записываем не более, чем `size` символов во второй.
6. Закрываем файлы.

```c
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <unistd.h>

const int size = 2;
```

```c
int main(int argc, char* argv[]) {
    if (argc != 3) {
        fprintf(stderr, "Need 2 arguments.\n");
        exit(-1);
    }

    const char *read_file_name = argv[1];
    const char *write_file_name = argv[2];

    char buffer[size];
    ssize_t read_bytes;

    int file;
        if ((file = open(read_file_name, O_RDONLY)) < 0) {
        printf("Can't open file for reading.\n");
        exit(-1);
    }

    struct stat st;
    if (fstat(file, &st) == -1) {
        printf("Can't get file info.\n");
        close(file);
        exit(-1);
    }

    int new_file;
    if ((new_file = open(write_file_name, O_WRONLY | O_CREAT, st.st_mode)) <
0) {
        printf("Can't open file for writing.\n");
        close(file);
        exit(-1);
    }

    do {
        read_bytes = read(file, buffer, size);
        if (read_bytes == -1) {
            printf("Can't write this file.\n");
            close(file);
            close(new_file);
            exit(-1);
        }
        write(new_file, buffer, read_bytes);
    } while (read_bytes == size);

    if (close(new_file) < 0) {
        printf("Can't close file for writing.\n");
    }
    if (close(file) < 0) {
        printf("Can't close file for reading.\n");
    }
    return 0;
}
```

Результат

Запустим код 3 раза.

```
alexvasyukov@Alexanders-MacBook-Air 04 % gcc hw4.c
alexvasyukov@Alexanders-MacBook-Air 04 % ./a.out hw4.c test1
alexvasyukov@Alexanders-MacBook-Air 04 % ./a.out a.out test2
alexvasyukov@Alexanders-MacBook-Air 04 % ./test2 test1 test3
alexvasyukov@Alexanders-MacBook-Air 04 %
```

1. В качестве аргумента передадим текст данной программы, запишем данные в файл `test1`. Файл правильно записался:

test1

```c
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <unistd.h>

const int size = 2;

int main(int argc, char* argv[]) {
    if (argc != 3) {
        fprintf(stderr, "Need 2 arguments.\n");
        exit(-1);
    }

    const char *read_file_name = argv[1];
    const char *write_file_name = argv[2];

    char buffer[size];
    ssize_t read_bytes;

    int file;
    if ((file = open(read_file_name, O_RDONLY)) < 0) {
        printf("Can't open file for reading.\n");
        exit(-1);
    }

    struct stat st;
    if (fstat(file, &st) == -1) {
        printf("Can't get file info.\n");
        close(file);
        exit(-1);
    }
```

2. В качестве аргумента передадим исполняемый файл данного скрипта и выведем запишем данные в файл `test2`. Выглядит немного страшно, но работает.

3. И чтобы проверить, что в прошлом тесте всё хорошо сработало, запустим файл `test2` с аргументами `test1` и `test2`. Всё отлично))

test3

```c
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <unistd.h>

const int size = 2;

int main(int argc, char* argv[]) {
    if (argc != 3) {
        fprintf(stderr, "Need 2 arguments.\n");
        exit(-1);
    }

    const char *read_file_name = argv[1];
    const char *write_file_name = argv[2];

    char buffer[size];
    ssize_t read_bytes;

    int file;
    if ((file = open(read_file_name, O_RDONLY)) < 0) {
        printf("Can't open file for reading.\n");
        exit(-1);
    }

    struct stat st;
    if (fstat(file, &st) == -1) {
        printf("Can't get file info.\n");
        close(file);
        exit(-1);
    }
```