**Липецкий государственный технический университет**

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №1

по дисциплине

«Операционные системы»

«Разработка программ обработки файлов в ОС Windows»

Вариант 2

Студент

Группа АИ-20-1 \_\_\_\_\_\_\_\_\_\_ Глубоков Г.В.

Подпись, дата

Руководитель

к.т.н.,доц. Батищев Р. В.

Подпись, дата

Липецк 2022г.

# Задание кафедры:

Написать программу вставки n строк исходного файла в заданную позицию результирующего файла и последующей установки/получения атрибутов результирующего файла. Исходные данные указываются пользователем в аргументах командной строки: количество строк - в качестве второго аргумента, номера строк - в качестве третьего - [(n+3)-1]-го аргументов, далее следуют имена исходного и результирующего файлов, следующий аргумент - позиция, в которую следует вставлять строки (исходные данные выбираются из табл. 1 прил.).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № вар. | Количество строк, n | Номера строк | Позиция в результирующем файле | Установка атрибутов |
| 2 | 3 | 4,7,10 | Текущая позиция | «архивный», «скрытый» |

# Теоретические сведения:

Чтение из файла: **read()**

Системный вызов **read()** позволяет считывать данные из открытого файла, на который ссылается дескриптор fd.

#include ssize\_t **read**(int fd, void \*buffer, size\_t count);

Возвращает количество считанных байтов, 0 при EOF или –1 при ошибке Аргумент count определяет максимальное количество считываемых байтов (тип данных size\_t — беззнаковый целочисленный). Аргумент buffer предоставляет адрес буфера памяти, в который должны быть помещены входные данные. Этот буфер должен иметь длину в байтах не менее той, что задана в аргументе count.

Запись в файл: **write()**

Системный вызов **write()** записывает данные в открытый файл.

#include ssize\_t **write**(int fd, const void \*buffer, size\_t count);

Возвращает количество записанных байтов или –1 при ошибке 116 Глава 4 • Файловый ввод-вывод: универсальная модель ввода-вывода Аргументы для write() аналогичны тем, что использовались для read(): buffer представляет собой адрес записываемых данных, count является количеством записываемых из буфера данных, а fd содержит дескриптор файла, который ссылается на тот файл, куда будут записываться данные.

Изменение файлового смещения: **lseek()**

off\_t **lseek**(int fd, off\_t offset, int whence);

Аргумент offset определяет значение смещения в байтах. (Тип данных off\_t — целочисленный тип со знаком, определенный в SUSv3.) Аргумент whence указывает на отправную точку, от которой отсчитывается смещение, и может иметь следующие значения:

SEEK\_SET — файловое смещение устанавливается в байтах на расстоянии offset от начала файла;

SEEK\_CUR — смещение устанавливается в байтах на расстоянии offset относительно текущего файлового смещения;

SEEK\_END — файловое смещение устанавливается на размер файла плюс offset. Иными словами, offset рассчитывается относительно следующего байта после последнего байта файла.

# Ход работы:

## Код программы на C++:

#include <sys/stat.h>

#include <fcntl.h>

#include <unistd.h>

#include <stdlib.h>

#include <iostream>

using namespace std;

int main(int argc, char \*argv[]){

int fd\_f = open(argv[1], O\_RDONLY);

size\_t count=10;

int n=0;

string arg[3] = {argv[3], argv[4],argv[5]};

off\_t size\_n = lseek(fd\_f, 0, SEEK\_END);

lseek(fd\_f, 0, SEEK\_SET);

char buff[size\_n+1];

ssize\_t numRead = read(fd\_f, buff, size\_n);

char newbuff[size\_n];

int newbuff\_size=0;

for (int i=0, currentline=1; i<size\_n; i++) {

if(currentline==stoi(arg[0])||currentline==stoi(arg[1])||currentline==stoi(arg[2])){

newbuff[newbuff\_size]=buff[i];

newbuff\_size++;

}

if(buff[i]=='\n'){

currentline++;

}

}

int fd\_l = open(argv[2], O\_RDWR);

off\_t curr = lseek(fd\_l, 0, SEEK\_CUR);

ssize\_t numWrite = write(fd\_l, newbuff, newbuff\_size);

close(fd\_f);

chmod(argv[2], S\_IRUSR | S\_IWUSR);

close(fd\_l);

## Пример выполнения:

На рисунке 1 показаны два файла: исходный и результирующий.

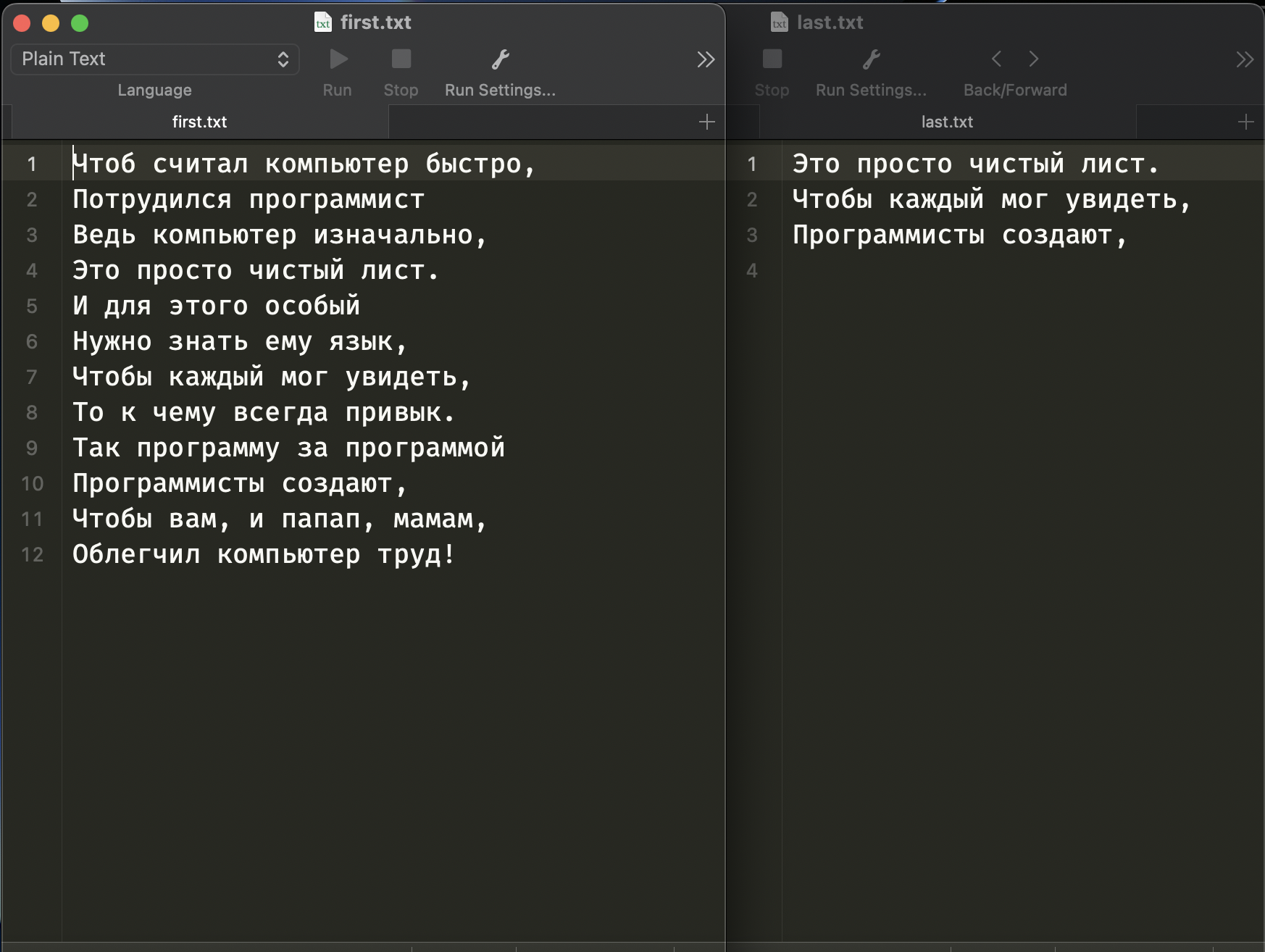


Рисунок 1 – Копирование необходимых строк в результирующий файл

# Заключение:

В ходе выполнения лабораторной работы были получены основные навыки работы с функциями Linux API, которые обеспечивают обработку файлов.