ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ассистент |  |  |  | М. А. Мурашова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| Обработка данных в файлах |
| по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № |  |  |  |  |  |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2022

**1.Цель работы:** Целью работы является изучение файловых операций ввода-вывода

**2.Задачи работы:**

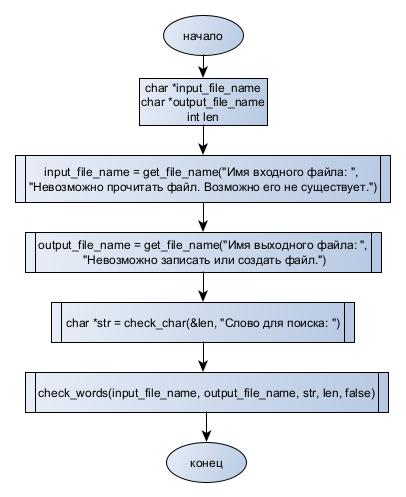
Написать программу, которая считывает текст из файла и записывает его в другой файл, добавляя после каждого предложения, сколько раз встретилось в нѐм введѐнное с клавиатуры слово. **3.Описание функций:**  
**1)Имя: main для fstream и cstdio**  
**Назначение:** ввод пользователем имя файла и слова для поиска, вывод на экран количества повторений слова в предложении после каждого предложения

**Входные данные:** нет  
**Выходные данные:** нет  
**Побочный эффект:** отсутствует.

**Тестовые данные:**   
**Ввод: 3D**

**Вывод:** **What is 3D printing?(1)**  
**Прототип:** int main()  
**Псевдокод:**ввод имени файла

Подсчет количества найденных слов

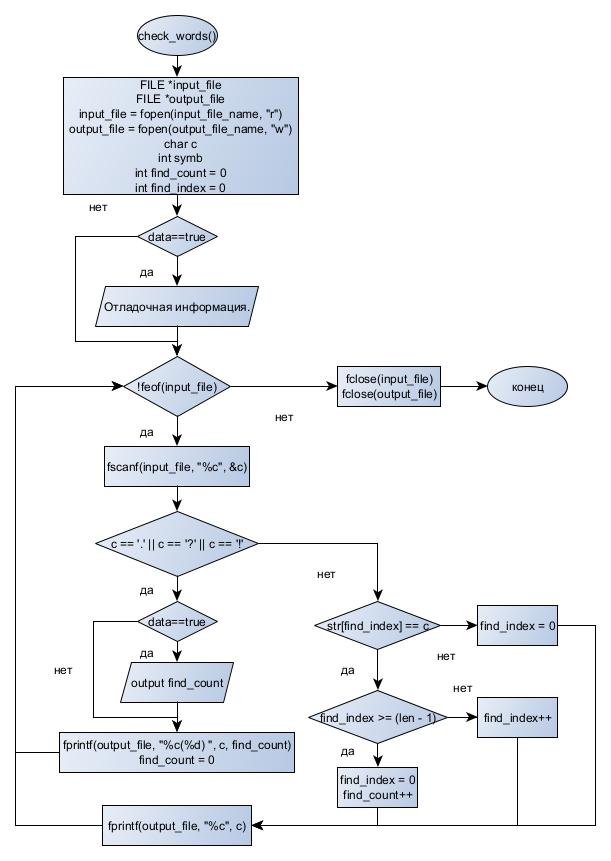
Запись в файл  
**Блок-схема:**

**2)Имя: check\_words() для fstream**  
**Назначение:** функция для подсчета слов в предожениях

**Входные данные:** \*input\_file\_name, \*output\_file\_name, \*str, len, data  
**Выходные данные:** нет  
**Побочный эффект:** отсутствует.   
**Прототип:** void check\_words()  
**Псевдокод:**если встречаем конец предложения, то записываем в него количество слов и обнуляем переменные для подсчёта

в остальных случаях ищем слова

если слово полностью совпадает, то обновляем счётсчик

сохранение и закрытие файла  
**Блок-схема:**  
****

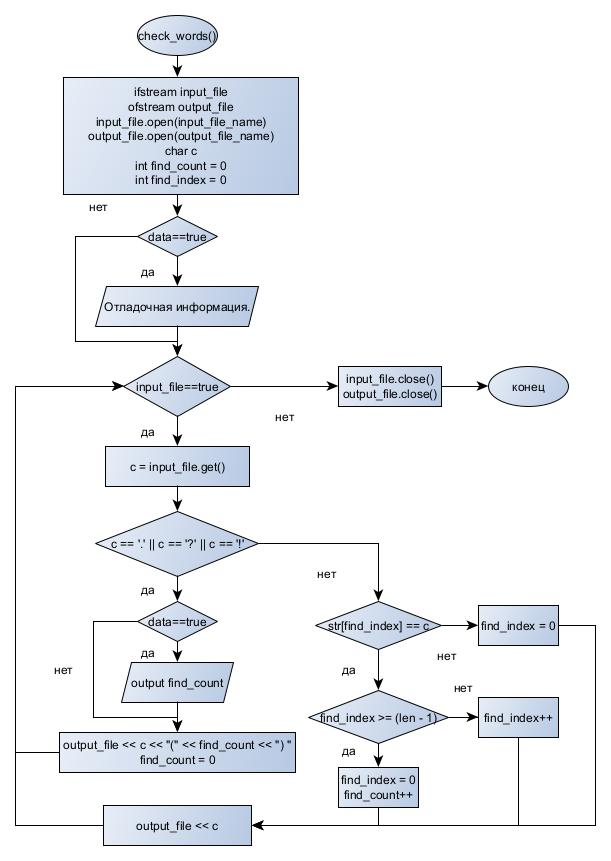
**3)Имя: check\_words() для cstdio**  
**Назначение:** функция для подсчета слов в предожениях

**Входные данные:** \*input\_file\_name, \*output\_file\_name, \*str, len, data  
**Выходные данные:** нет  
**Побочный эффект:** отсутствует.   
**Прототип:** void check\_words()  
**Псевдокод:**если встречаем конец предложения, то записываем в него количество слов и обнуляем переменные для подсчёта

в остальных случаях ищем слова

если слово полностью совпадает, то обновляем счётсчик

сохранение и закрытие файла  
**Блок-схема:**



**4)Имя: get\_file() для cstdio**  
**Назначение:** функция для подсчета слов в предожениях

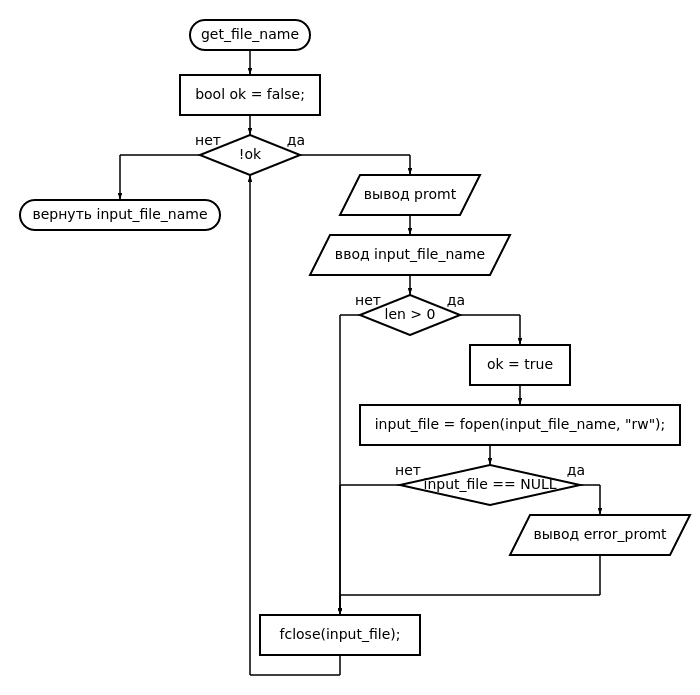
**Входные данные:** \*input\_file\_name, \*output\_file\_name, \*str, len, data  
**Выходные данные:** нет  
**Побочный эффект:** отсутствует.   
**Прототип:** char\* get\_file()  
**Псевдокод:**Ввод названия файла

Открываем файл

Если файла не существует, либо его нельзя записать, то повторяем всё заново

Закрываем файл

Возвращаем название файла  
**Блок-схема:**

****

**4)Имя: get\_file() для fstream**  
**Назначение:** функция для подсчета слов в предожениях

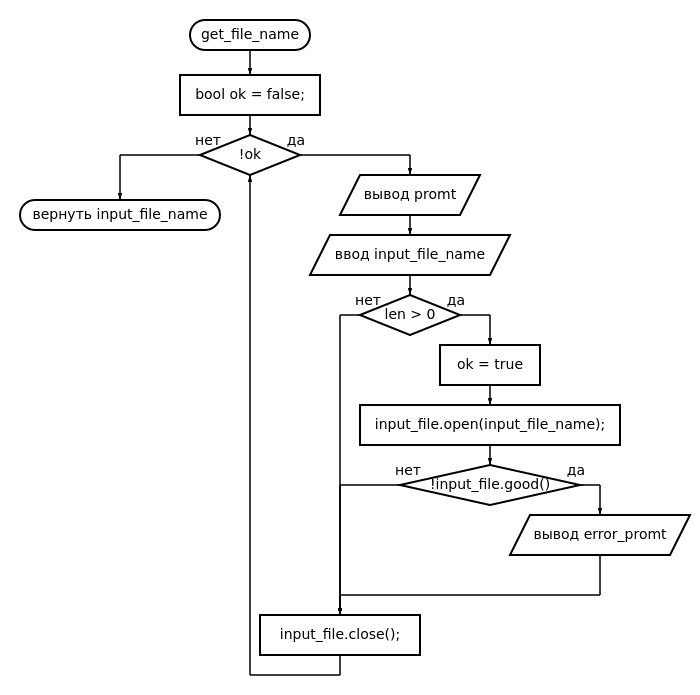
**Входные данные:** \*input\_file\_name, \*output\_file\_name, \*str, len, data  
**Выходные данные:** нет  
**Побочный эффект:** отсутствует.   
**Прототип:** char\* get\_file()  
**Псевдокод:**Ввод названия файла

Открываем файл

Если файла не существует, либо его нельзя записать, то повторяем всё заново

Закрываем файл

Возвращаем название файла  
**Блок-схема:**



**5.ЛИСТИНГ КОДА:  
1)Для fstream**

#include <iostream>

using namespace std;

#include "libs/lib.h"

#include <cmath>

#include <time.h>

#include <fstream>

#include <cctype>

#include "libs/simple\_char.h"

#include "fstream\_files.h"

char \*get\_file\_name(const char \*promt = "", const char \*error\_promt = "", bool new\_file = false) {

ifstream input\_file;

int len;

char \*input\_file\_name;

bool ok = false;

while (!ok) {

cout << promt;

input\_file\_name = get\_string(&len);

if (len > 0) {

ok = true;

input\_file.open(input\_file\_name);

if (!input\_file.good()) {

if (new\_file) {

ofstream ost(input\_file\_name);

} else {

cout << error\_promt << endl;

ok = false;

}

}

input\_file.close();

}

}

return input\_file\_name;

}

// функция для подсчёта слов

void check\_words(char \*input\_file\_name, char \*output\_file\_name, char \*str, int len, bool data = false) {

ifstream input\_file;

ofstream output\_file;

// открываем файлы

input\_file.open(input\_file\_name);

output\_file.open(output\_file\_name);

char c;

if (data) {

draw\_line(20);

cout << "Отладочная информация." << endl;

draw\_line(20);

}

int find\_count = 0;

int find\_index = 0;

// цикл по всему файлу

while (input\_file) {

// берём из файла 1 символ

c = input\_file.get();

// если встречаем конец предложения, то записываем в него количество слов и обнуляем переменные для подсчёта

if (c == '.' || c == '?' || c == '!') {

if (data) cout << c << "(" << find\_count << ") ";

output\_file << c << "(" << find\_count << ") ";

find\_count = 0;

} else {

// в остальных случаях ищем слова

if (str[find\_index] == c) {

// если слово полностью совпадает, то обновляем счётсчик

if (find\_index >= (len - 1)) {

find\_index = 0;

find\_count++;

} else {

find\_index++;

}

} else {

find\_index = 0;

}

//

output\_file << c;

}

}

input\_file.close();

output\_file.close();

}

int main() {

system("chcp 65001");

//clear\_scr();

char \*input\_file\_name;

char \*output\_file\_name;

// вводим имена файлов (с проверкой)

input\_file\_name = get\_file\_name("Имя входного файла: ", "Невозможно прочитать файл. Возможно его не существует.");

output\_file\_name = get\_file\_name("Имя выходного файла: ", "Невозможно записать или создать файл.", true);

draw\_line(20);

int len;

// вводим слово для поиска

char \*str = check\_char(&len, "Слово для поиска: ");

// подсчитываем найденые слова

check\_words(input\_file\_name, output\_file\_name, str, len, false);

draw\_line(20);

cout << "В файл " << output\_file\_name << " был успешно записан результат." << endl;

return 0;

}

**2)Для cstdio**

char \*get\_file\_name(const char \*promt = "", const char \*error\_promt = "") {

FILE \*input\_file;

int len;

char \*input\_file\_name;

bool ok = false;

while (!ok) {

cout << promt;

input\_file\_name = get\_string(&len);

if (len > 0) {

ok = true;

input\_file = fopen(input\_file\_name, "rw"); // если выдаёт ошибку поставь "r" вместо "rw"

if (input\_file == NULL) {

cout << error\_promt << endl;

ok = false;

}

fclose(input\_file);

}

}

return input\_file\_name;

}

void check\_words(char \*input\_file\_name, char \*output\_file\_name, char \*str, int len, bool data = false) {

FILE \*input\_file;

FILE \*output\_file;

input\_file = fopen(input\_file\_name, "r");

output\_file = fopen(output\_file\_name, "w");

char c;

int symb;

if (data) {

draw\_line(20);

cout << "Отладочная информация." << endl;

draw\_line(20);

}

int find\_count = 0;

int find\_index = 0;

while (!feof(input\_file)) {

fscanf(input\_file, "%c", &c);

if (c == '.' || c == '?' || c == '!') {

if (data) cout << c << "(" << find\_count << ") ";

fprintf(output\_file, "%c(%d) ", c, find\_count);

find\_count = 0;

} else {

if (str[find\_index] == c) {

if (find\_index >= (len - 1)) {

find\_index = 0;

find\_count++;

} else {

find\_index++;

}

} else {

find\_index = 0;

}

fprintf(output\_file, "%c", c);

}

}

fclose(input\_file);

fclose(output\_file);

}

int main() {

// смена кодировки

system("chcp 65001");

// очистка терминала

//clear\_scr();

char \*input\_file\_name;

char \*output\_file\_name;

input\_file\_name = get\_file\_name("Имя входного файла: ", "Невозможно прочитать файл. Возможно его не существует.");

output\_file\_name = get\_file\_name("Имя выходного файла: ", "Невозможно записать или создать файл.");

draw\_line(20);

int len;

char \*str = check\_char(&len, "Слово для поиска: ");

check\_words(input\_file\_name, output\_file\_name, str, len, false);

draw\_line(20);

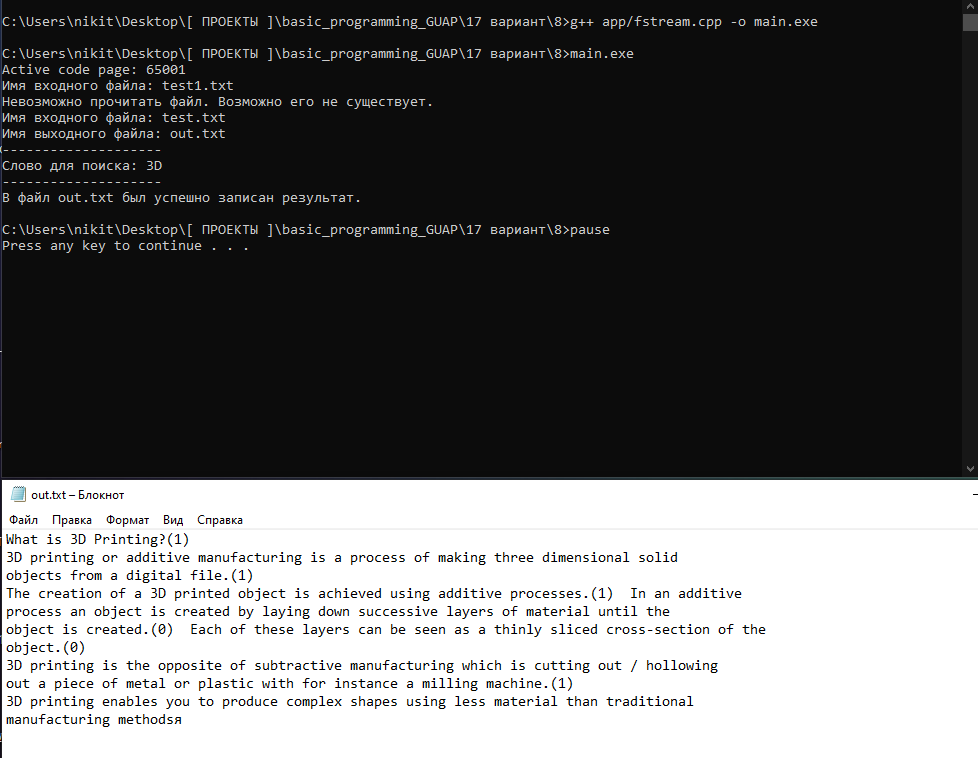
cout << "В файл " << output\_file\_name << " был успешно записан результат." << endl;

return 0;

}

**6.Пример выполнения программы:**

**main**



**7.Анализ результатов и выводы:**

В ходе этой лабораторной работы мы изучили структуру данных одномерных массивов и перезаписывать их.

Из плюсов можно отметить, что программа работает корректно.

Существенных минусов не обнаружено.