Title

Содержание

[Введение 3](#_Toc514617415)

[Краткое описания метода решения задачи 4](#_Toc514617416)

[Описание алгоритма работы программы 5](#_Toc514617417)

[Блок-схема программы 7](#_Toc514617418)

[Текст программы и результаты её работы 9](#_Toc514617419)

[Заключение 14](#_Toc514617420)

[Список литературы 15](#_Toc514617421)

# Введение

Задачей данной работы является разработка программы, решающей следующую задачу.

Необходимо прочитать текст из файла, вывести его на экран. В тексте найти все слова, у которых первый и последний символы совпадают. Далее выводится общее количество найденных слов, а по нажатию на пробел слова показываются. Имя файла запрашивается у пользователя.

# Краткое описания метода решения задачи

Программа выполняется в среде MS Visual Studio 2015. Ввиду того, что в этой версии для С/С++ не поддерживается графическая оболочка, программа разработана как консольное приложение. Поэтому результаты работы программы вместо подсвечивания слова на экране будет каждый раз выводить слово в новой строке с указанием номера строки и позиции в строке. Номера строк начинаются с 1, номера позиций – с 0.

Главная программа запрашивает имя файла у пользователя и передаёт его в качестве параметра основной обрабатывающей функции. Если функция возвращает значение false, выдаётся сообщение об ошибке, если true – производится вывод слов, которые нашла функция.

Работа со строками реализована с использованием типа string из стандартной библиотеки, работа с файлами – с использованием потоков ввода-вывода оттуда же.

# Описание алгоритма работы программы

Входными данными для программы является тестовый файл со словами, разделёнными знаком пробела, причём только одним. Строка не должна начинаться с пробела и заканчиваться им.

Основной алгоритм программы реализован в функции do\_work(string FName, v\_file &mas), где тип v\_file описан следующим образом

typedef vector<string> v\_file; typedef vector<string> v\_file;

Функция получает в качестве параметров строку с именем файла и массив строк для выдачи результатов. Первый параметр входной, второй выходной.

Далее осуществляется попытка открытия файла с именем, переданным в переменной FName для чтения. Если файл не открыт, происходит возврат в основную программу с результатом false. В случае удачи происходит выполнение основного алгоритма.

В цикле считывается строка из входного файла. Если строка прочитана, то она поступает на обработку.

Цикл по нахождению в строке символа-разделителя слов. Вначале для поиска разделителя была использована функция find, которая возвращает номер позиции, с которой найдено вхождение строки-параметра в исходную строку. Но потом было решено использовать функцию find\_first\_of, которая находит не вхождение одной строки в другую, а вхождение символов из строки-параметра. Таким образом в будущем можно в качестве строки параметра задать строку из разделителей слов, которые могут быть не только пробелом, а и табуляцией, или символом точка с запятой или какими-то другими.

string delimeter = " ;,\t"; // Разделитель слов

Цикл по нахождению пробела в строке. В случае нахождения разделителя берётся часть строки от места предыдущего поиска до текущего – это и есть слово, стоящее между двумя разделителями. Если оно удовлетворяет условию, то происходит запоминание этого слова, его строки и позиции в строке.

После прохождения цикла по строке у нас остаётся часть сроки от последнего разделителя до конца. Её мы тоже проверяем на совпадение начального и конечного символов, после чего увеличиваем счётчик строк и переходим к внешнему циклу чтения строки из файла.

После завершения работы внешнего цикла по чтению из файла мы имеем массив строк mas, в котором отфиксированы все строки, для которых условие выполняется, с указанием номера строки и позиции в строке. Происходит возврат в главную функцию с результатом true.

Для проверки условия была написана небольшая функция, которая вначале проверяла длину слова. Мы считаем, что слово длиной в 1 символ удовлетворяет условию. Однако после некоторого анализа стало понятно, что в слове длиной 1 первый символ совпадает по номеру с последним, поэтому эта проверка оказалась лишней.

// Равенство начального и конечного элементов

bool is\_eq(string strtemp)

{

//if (strtemp.length() == 1)

// return true;

return(strtemp[0] == strtemp[strtemp.length() - 1]);

}

# Блок-схема программы

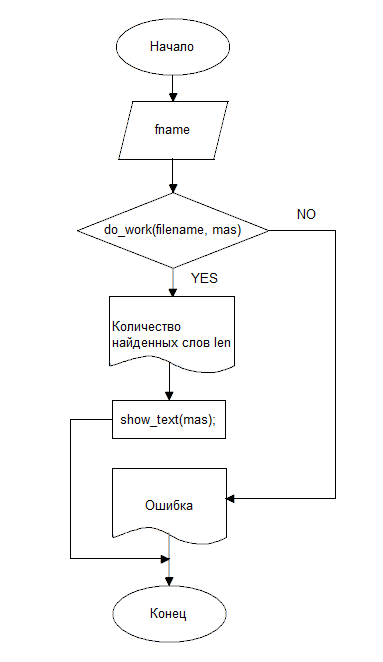


Рис. 1 Блок-схема основной программы

На рис. 1 приведена блок-схема основной программы. В ней используются три функции, из которых две состоят всего из нескольких строк, а вот блок-схема основной функции обработки текста, в которой реализован непосредственно алгоритм, приведена на рисунке 2.

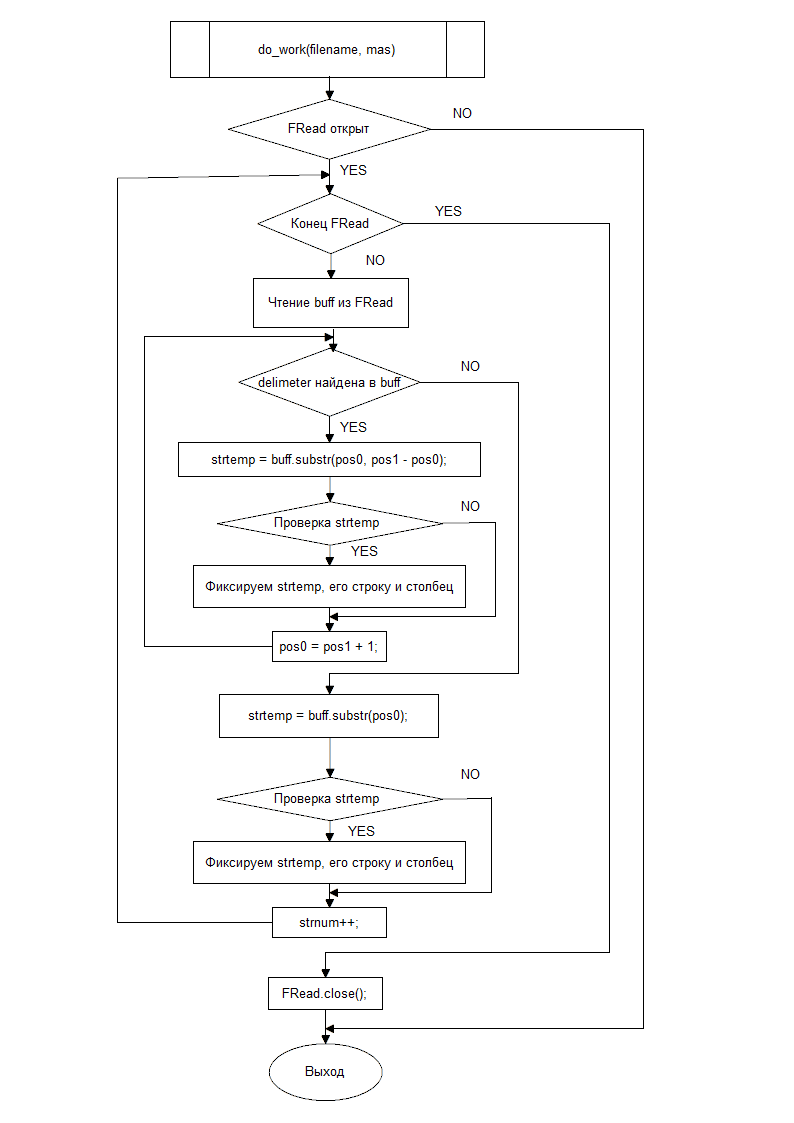


Рис. 2 Блок-схема основной функции

# Текст программы и результаты её работы

Текст всей программы находится в одном файле.

// ConsoleApplication82.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include <vector>

#include <fstream>

#include <iostream>

#include <string>

#include <conio.h>

using namespace std;

typedef vector<string> v\_file; // Вектор строк

// Равенство начального и конечного элементов

bool is\_eq(string strtemp)

{

//if (strtemp.length() == 1)

// return true;

return(strtemp[0] == strtemp[strtemp.length() - 1]);

}

// Основная процедура обработки

bool do\_work(string FName, v\_file &mas)

{

ifstream FRead(FName); // Файл для чтения

bool ret = true;

string buff; // Буфер для чтения строки из файла

try

{

if (!FRead.is\_open()) // если файл не открыт

{

return false;

}

else

{

int strnum = 1; // Номер текущей строки

string delimeter = " "; // Разделитель слов

//Цикл - пока из файла прочитана строка

while (getline(FRead, buff)) // Читаем строку из файла в буфер

{

cout << strnum << ": " << buff << std::endl; // Вывод строки - номер и содержание

string strtemp; // Вспомогательная строка

std::string::size\_type pos0 = 0; // Позиция в строке

std::string::size\_type pos1; // Позиция в строке

// Цикл пока в строке найдено следующее вхождение разделителя

while ((pos1 = buff.find\_first\_of(delimeter, pos0)) != std::string::npos)

{

strtemp = buff.substr(pos0, pos1 - pos0); // Часть исходной строки от предыдущей до текущей позиции

if (is\_eq(strtemp))

{

// Фиксируем найденное слово

mas.push\_back("Слово " + strtemp + " строка " + to\_string(strnum) + " позиция "+ to\_string((int)pos0));

}

pos0 = pos1 + 1; // Смещаем указатель на начало необработанной части строки

}

strtemp = buff.substr(pos0); // Добавляем то, что осталось за последним вхождением разделителем

if (is\_eq(strtemp))

{

// Фиксируем найденное слово

mas.push\_back("Слово " + strtemp + " строка " + to\_string(strnum) + " позиция " + to\_string((int)pos0));

}

strnum++; // Счётчик обработанных строк

}

FRead.close(); // закрываем файл

}

}

catch (...)

{

ret = false;

}

return ret;

}

void show\_text(v\_file mas)

{

int len = mas.size();

for (int i = 0; i < len; ++i)

{

cout << mas[i] << std::endl; // Вывод информации о слове

\_getch(); // Ждём нажатия клавиши

}

}

int main()

{

v\_file mas; // Вектор строк для сохранения описания о найденных словах

string filename;

setlocale(LC\_ALL, "rus");

cout << "Введите имя файла ";

cin >> filename;

if (do\_work(filename, mas))

{

int len = mas.size(); // Строк в массиве - количество найденных слов

cout << std::endl << "Результаты поиска в файле " << filename << std::endl;

cout << "Всего найдено " << len << " слов" <<std::endl;

show\_text(mas);

}

else

{

cout << "Ошибки в работе ";

}

cout << "Конец просмотра ";

\_getch();

return 0;

}

В главной программе для показа результата работы используется функция void show\_text(v\_file mas), которая выдаёт на экран одну строку массива mas и ожидает нажатие на клавишу.

В ходе проверки работы программы был взят текстовый файл следующего содержания:

12345 554 56 4 56565 43434 dfgd dfgh ytr

wert tre erte 123 4564 54 34 5555

zserd rfr ere rerere 121

eeer tyyui

fgf 12341 fg56f erer

В результате выполнения программы были получены следующие результаты. Вначале программа запросила имя входного файла и получив его выполнила обработку, выдала на экран первое слово, после чего ожидала нажатие клавиши (рис. 1).

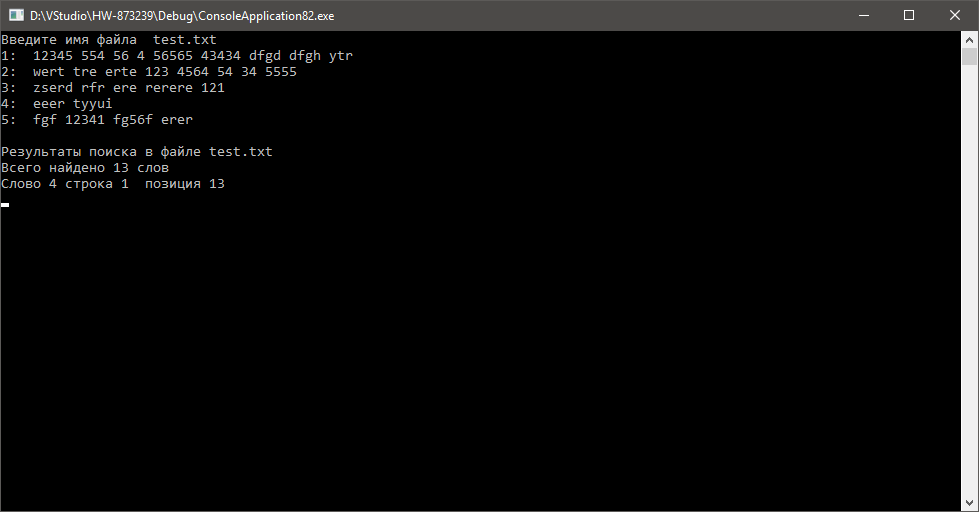


Рис. 3. Результат работы программы, кадр 1.

После нескольких нажатий мы получили следующий экран (рис. 4).

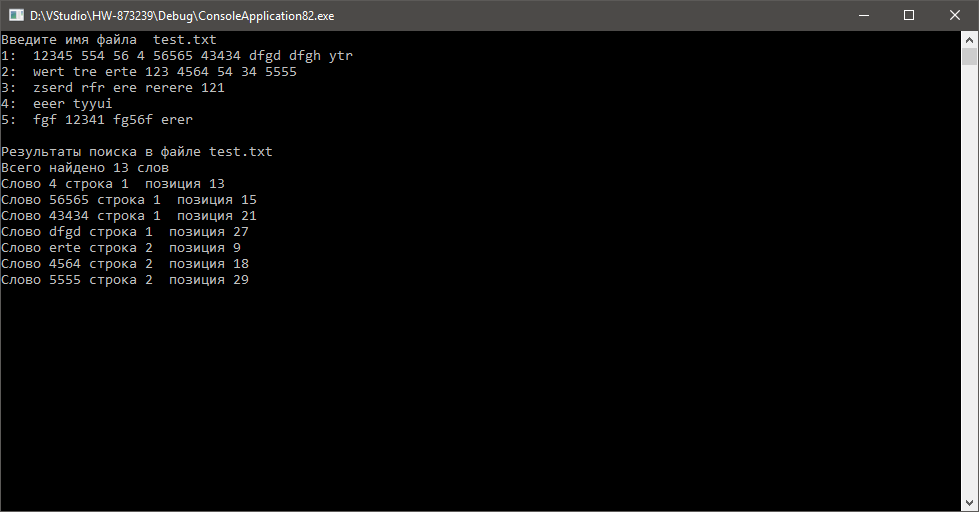


Рис. 4. Результат работы программы, кадр 2.

После показа всех найденных слов программа выдала сообщение о завершении работы и ожидала нажатия клавиши (рис. 5).

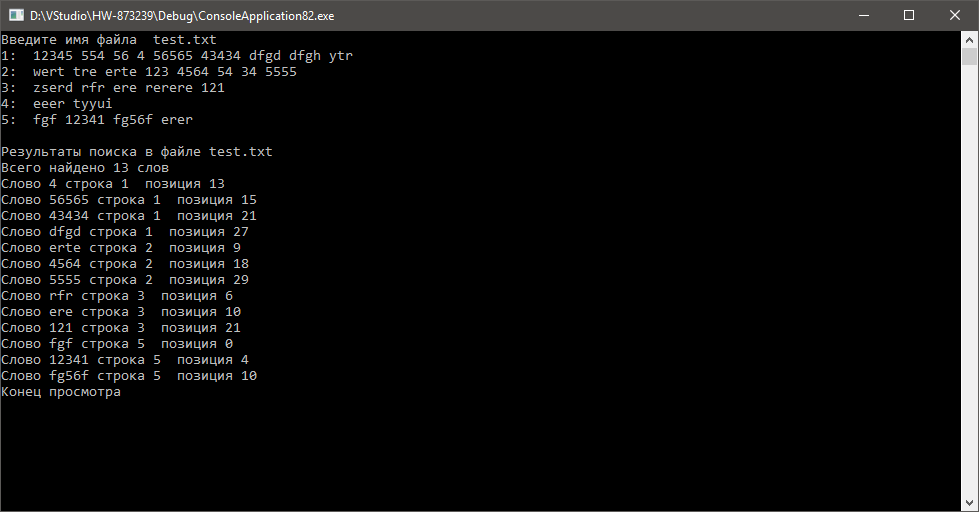


Рис. 5. Результат работы программы, кадр 3.

# Заключение

В результате выполнения данной работы была написана программа, реализующая решение поставленной задачи. В ходе разработки программы были изучены разделы работы со строками и с потоками ввода-вывода из стандартной библиотеки языка С++.

# Список литературы

1. Вирт Н. Алгоритмы + структуры данных = программы. – М. : Мир, 1976, 736с.
2. К. Джамса. Учимся программировать на языке C++: Пер. с англ. - Москва: Мир, 1997. 320с.
3. Роберт, С. Сикорд Безопасное программирование на C и C++ / Роберт С. Сикорд. - Москва: РГГУ, 2014. - 496 c.
4. [www.cplusplus.com/](http://www.cplusplus.com/)
5. <https://metanit.com/cpp/>