Пермский Национальный Исследовательский Политехнический Университет

Лабораторная работа №9

"Строковый ввод-вывод"

21 вариант

Выполнил:

Студент группы ИВТ-20-2б

Галинов Олег Юрьевич

Проверила:

Доцент кафедры ИТАС

Полякова О.А.

Пермь 2020

**Постановка задачи**

1) Скопировать из файла F1 в файл F2 все строки, в которых есть слова, совпадающие с первым словом.

2) Определить количество согласных букв в последней строке файла F2.

**Анализ задачи**

1. *Используемые типы данных.*

В программе используются: bool, string, int, vector<string>.

Тип bool проверяет наличие слова, совпадающего с первым.

bool wordlikefirst(vector<string>Listofwords)

{

for (int i = 1; i < Listofwords.size(); i++)

{

if (Listofwords[0] == Listofwords[i])

return true;

}

return false;

}

Тип vector<string> хранит слова. Он упрощает обработку слов в программе.

vector<string> StringToWords(string line);

Данные типа string используются для хранения и ввода строк в программу, хранение слов, из которых состоит строка.

string currentLine;

Данные типа int используются как счетчики (подсчет кол-ва согласных в последней строке).

int soglasn;

1. *Действия над используемыми данными.*

С данными типа vector<string> производятся следующие действия: добавление элементов в вектор, сравнение первого слова с другими в строке, разбив строки на слова.

vector<string> res;

res.push\_back(line.erase(0, wordBeginning));

bool wordlikefirst(vector<string>Listofwords)

{

for (int i = 1; i < Listofwords.size(); i++)

{

if (Listofwords[0] == Listofwords[i])

return true;

}

return false;

}

С данными типа string производятся следующие действия: удаление символов из строки (удаление пробелов и разделение строки на слова), подсчет кол-ва согласных букв в последней строке.

for (int i = 0; i < NewWords.size(); i++)

{

switch (NewWords[i])

{

case 'b' : case 'c': case 'd': case 'f' : case 'g' : case 'h': case 'j':

case 'k': case 'l': case 'm': case 'n': case 'p': case 'q' : case 'r':

case 's': case 't': case 'v': case 'w': case 'x': case 'z': soglasn++;

}

С данными типа int производятся следующие действия: математические операции (инкремент в счетчиках и в циклах с параметром).

soglasn++;

for (int i = 1; i < Listofwords.size(); i++)

1. *Вид данных.*

Данные вводятся и выводятся через текстовые файлы (F1.txt и F2.txt соответственно).

1. *Структура.*

Структуры не используются.

1. *Ввод и вывод.*

Ввод всех данных осуществляется через файл F1.txt. Скопированные по условию строки выводятся в файл F2.txt. Последняя строка и кол-во согласных букв в ней выводятся в консоль.

1. *Действия для решения задачи.*

Для удаления пробелов в строке используются: цикл с предусловием while, оператор сравнения if.

string DeleteSpaces(string line)

{

while (line.find(" ") != -1)

line.replace(line.find(" "), 2, " ");

if (line[0] == ' ') line.erase(0, 1);

if (line[line.length() - 1] == ' ') line.erase(line.length() - 1);

return line;

}

Для разделения строки на слова используются: цикл с параметром for для перебора символов в строке, условный оператор if для поиска пробелов в строке.

vector<string> StringToWords(string line)

{

vector<string> res;

line = DeleteSpaces(line);

int wordBeginning = 0;

for (int i = 0; i < line.length(); i++)

{

if (line[i] == ' ')

{

string lineCopy = line;

res.push\_back(lineCopy.erase(i).erase(0, wordBeginning));

wordBeginning = i + 1;

i++;

}

}

res.push\_back(line.erase(0, wordBeginning));

return res;

}

Для поиска одинаковых слов в строке используются: цикл с параметром for для перебора всех слов, условный оператор if для сравнения этих слов между собой.

bool wordlikefirst(vector<string>Listofwords)

{

for (int i = 1; i < Listofwords.size(); i++)

{

if (Listofwords[0] == Listofwords[i])

return true;

}

return false;

}

Для подсчета согласных букв в последней строке используется цикл for и оператор switch для сравнения букв в строке с заданными (согласными).

for (int i = 0; i < NewWords.size(); i++)

{

switch (NewWords[i])

{

case 'b' : case 'c': case 'd': case 'f' : case 'g' : case 'h': case 'j':

case 'k': case 'l': case 'm': case 'n': case 'p': case 'q' : case 'r':

case 's': case 't': case 'v': case 'w': case 'x': case 'z': soglasn++;

}

}

Для проверки доступности файла на чтение используется условный оператор if.

if (in.is\_open())

Для чтение данных из файла используется цикл с предусловием while.

while (!in.eof())

{

}

**Код программы**

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

using namespace std;

bool wordlikefirst(vector<string>Listofwords)

{

for (int i = 1; i < Listofwords.size(); i++)

{

if (Listofwords[0] == Listofwords[i])

return true;

}

return false;

}

string DeleteSpaces(string line)

{

while (line.find(" ") != -1)

line.replace(line.find(" "), 2, " ");

if (line[0] == ' ') line.erase(0, 1);

if (line[line.length() - 1] == ' ') line.erase(line.length() - 1);

return line;

}

vector<string> StringToWords(string line)

{

vector<string> res;

line = DeleteSpaces(line);

int wordBeginning = 0;

for (int i = 0; i < line.length(); i++)

{

if (line[i] == ' ')

{

string lineCopy = line;

res.push\_back(lineCopy.erase(i).erase(0, wordBeginning));

wordBeginning = i + 1;

i++;

}

}

res.push\_back(line.erase(0, wordBeginning));

return res;

}

int main()

{

ifstream in("F1.txt");

string NewWords;

if (in.is\_open())

{

ofstream out("F2.txt");

while (!in.eof())

{

string currentline;

getline(in, currentline);

vector<string>Words = StringToWords(currentline);

if (wordlikefirst(Words))

{

out << currentline << endl;

NewWords = currentline;

}

}

cout << NewWords << endl;

int soglasn = 0;

for (int i = 0; i < NewWords.size(); i++)

{

switch (NewWords[i])

{

case 'b' : case 'c': case 'd': case 'f' : case 'g' : case 'h': case 'j':

case 'k': case 'l': case 'm': case 'n': case 'p': case 'q' : case 'r':

case 's': case 't': case 'v': case 'w': case 'x': case 'z': soglasn++;

}

}

cout << soglasn << endl;

out.close();

}

else

{

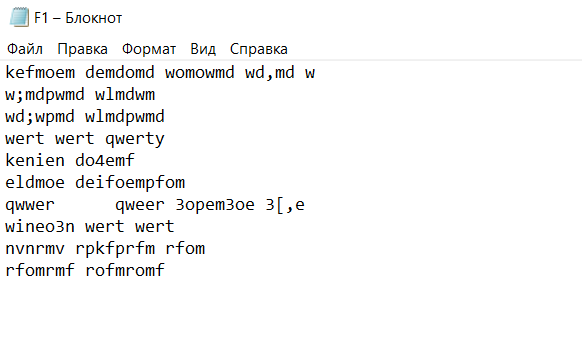
cout << "Error!" << endl;

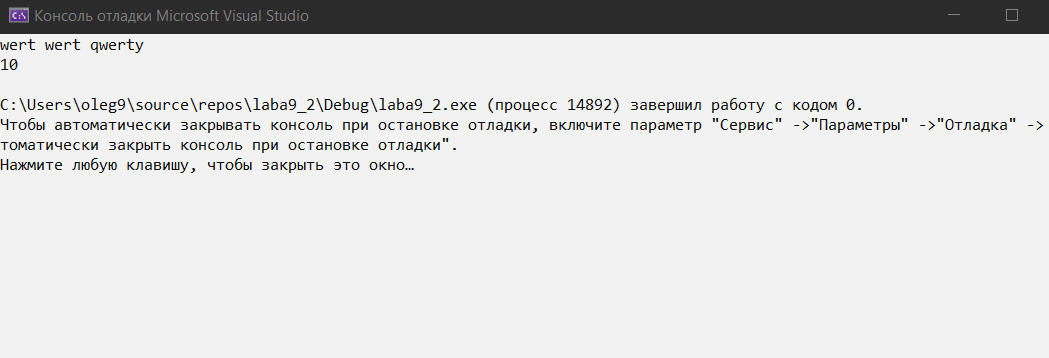
}

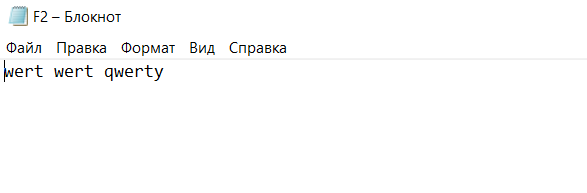
in.close();

}

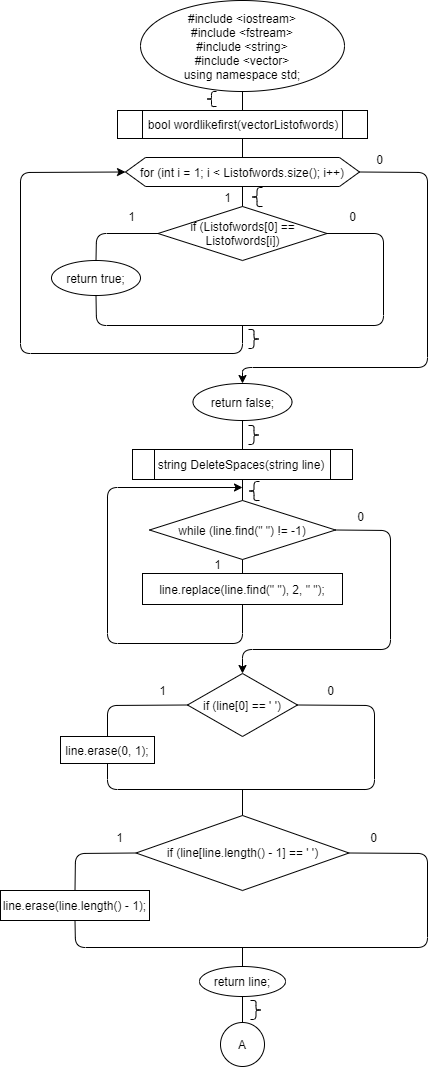
**Скриншоты**

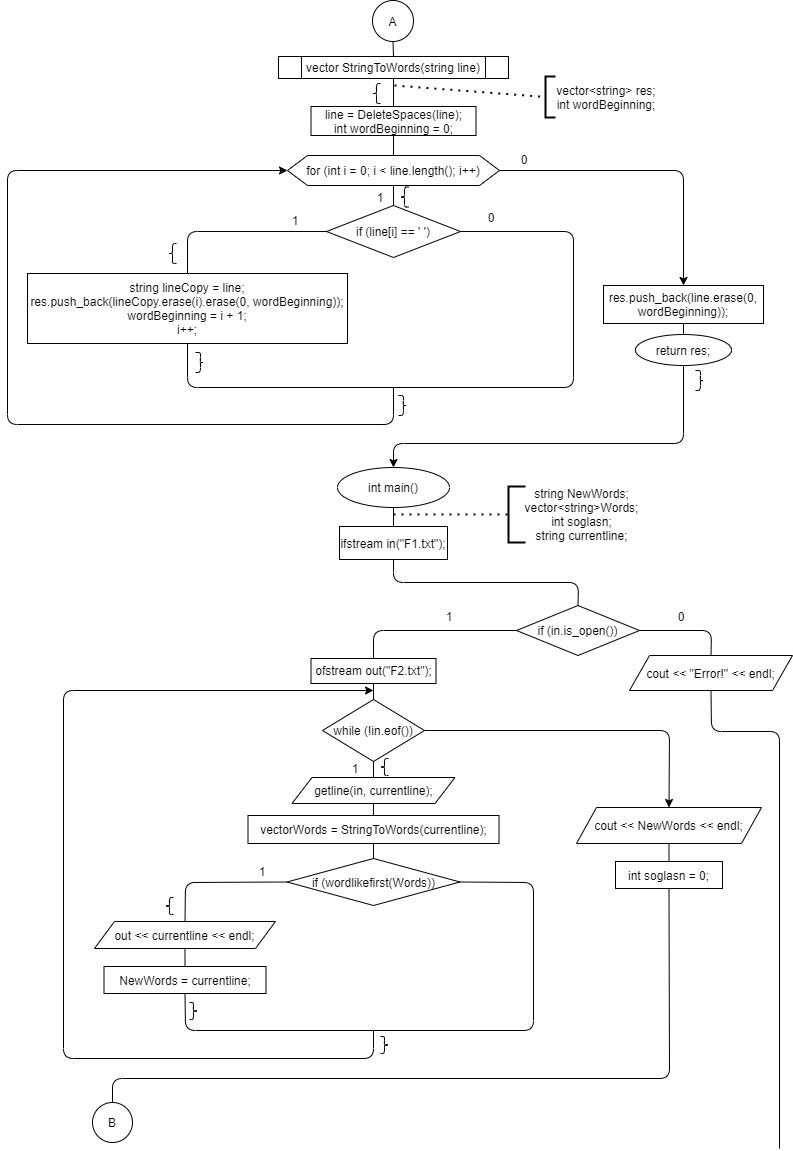
****

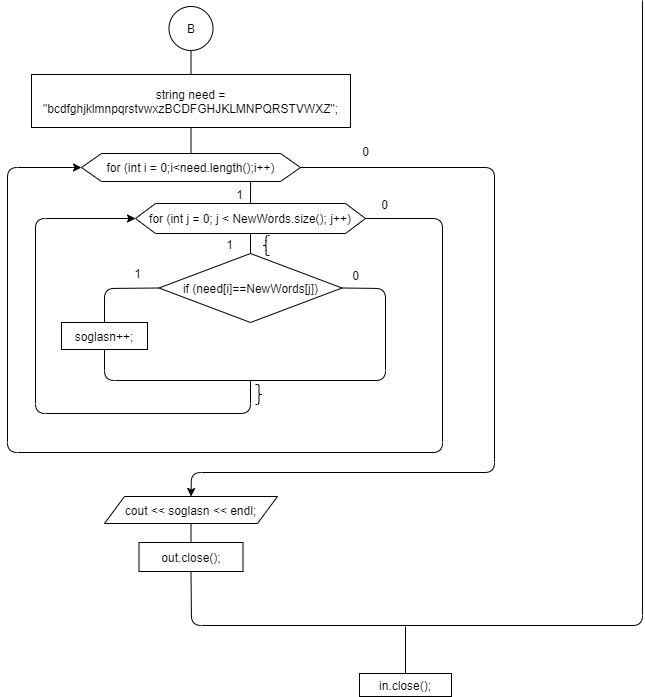
****

****

**Блок-схема**

****

****

****