Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе №9**

Дисциплина: «Информатика»

Тема: Структурное программирование. Файловые потоки.

Вариант 23

Выполнил:

Студент группы РИС-20-1б

Кузнецов Михаил Сергеевич

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Цель работы**

Работа с текстовыми файлами, ввод-вывод текстовой информации и ее хранение на внешних носителях.

**Постановка задачи**

1) Скопировать из файла F1 в файл F2 все строки, в которых содержится только одно слово.

2) Определить номер слова, в котором больше всего согласных букв.

**Анализ задачи**

1. Определить какие действия необходимо выполнить для решения задачи.
   1. Проверить, существует ли файл ввода и в случае его отсутствия сообщить об ошибке.

if (ifile.is\_open())

* 1. Разработать функцию, которая проверяет является ли буква согласной.

bool is\_char\_in\_string(char a, string s)

{

int ind = 0;

bool find = false;

while (ind < s.length() && !find)

{

if (s[ind] == a)

{

find = true;

}

ind++;

}

return find;

}

* 1. Разработать функцию, которая считывает количество слов в строке и номер элемента в котором больше всего согласных букв.

void string\_processing(string\_ex s[], int size)

{

int num\_word = 0;

int max\_cons = 0;

int cons\_in\_word = 0;

int pos\_max\_cons = 0;

bool word\_started = false;

string cons = "БбВвГгДдЖжЗзЙйКкЛлМмНнПпРрСсТтФфХхЦцЧчШшЩщ";

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

{

for (int j = 0; j < s[i].line.length(); j++)

{

if (s[i].line[j] != ' ')

{

if (!word\_started)

{

word\_started = true;

num\_word++;

cons\_in\_word = 0;

}

if (is\_char\_in\_string(s[i].line[j], cons))

{

cons\_in\_word++;

}

}

if (s[i].line[j] == ' ' || j + 1 == s[i].line.length())

{

word\_started = false;

if (cons\_in\_word > max\_cons)

{

max\_cons = cons\_in\_word;

pos\_max\_cons = num\_word;

}

}

}

s[i].num\_word = num\_word;

s[i].pos\_max\_cons = pos\_max\_cons;

num\_word = 0;

max\_cons = 0;

cons\_in\_word = 0;

pos\_max\_cons = 0;

word\_started = false;

}

}

1. Для решения задачи использовались следующие типы данных.

2.1 Файлы ввода/вывода представлены классами ifstream/ofstream.

ifstream ifile("inLab9.txt");

ofstream outfile("OutLab9.txt");

2.2 Каждая строка из файла хранится в массиве и представлена в виде структуры string\_ex в которой хранится сама строка, количество слов строке и позиция слова с наибольшим количеством согласных букв.

struct string\_ex

{

string line;

int num\_word = 0;

int pos\_max\_cons = 0;

};

string\_ex\* data = new string\_ex[arr\_size];

1. Считывание данных из файла осуществляется функцией read\_file, которая получает на вход ссылку на файл и ссылку на массив, в котором будут храниться строки.

void read\_file(ifstream &file, string\_ex arr[])

{

int ind = 0;

while (!file.eof())

{

getline(file, arr[ind].line);

ind++;

}

}

1. Запись данных файл осуществляется функцией write\_in\_file, которая на вход получает ссылку на массив со строками, ссылку на файл для записи и размер массива.

void write\_in\_file(string\_ex arr[], ofstream& file, int size)

{

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

{

if (arr[i].num\_word == 1)

{

file << arr[i].line << endl;

}

}

}

**Код**

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

struct string\_ex

{

string line;

int num\_word = 0;

int pos\_max\_cons = 0;

};

void read\_file(ifstream &file, string\_ex arr[])

{

int ind = 0;

while (!file.eof())

{

getline(file, arr[ind].line);

ind++;

}

}

int get\_file\_rows(ifstream& file)

{

string currentLine;

int count = 0;

while (!file.eof())

{

getline(file, currentLine);

count++;

}

file.seekg(0);

return count;

}

void print\_arr(string\_ex arr[], int size)

{

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

{

cout << "Row №" << i + 1 << endl;

cout << "Row: " << arr[i].line << endl;

cout << "Numbers of word: " << arr[i].num\_word << endl;

cout << "The word with the maximum number of consonants under the number: " << arr[i].pos\_max\_cons << endl << endl;

}

}

bool is\_char\_in\_string(char a, string s)

{

int ind = 0;

bool find = false;

while (ind < s.length() && !find)

{

if (s[ind] == a)

{

find = true;

}

ind++;

}

return find;

}

void string\_processing(string\_ex s[], int size)

{

int num\_word = 0;

int max\_cons = 0;

int cons\_in\_word = 0;

int pos\_max\_cons = 0;

bool word\_started = false;

string cons = "БбВвГгДдЖжЗзЙйКкЛлМмНнПпРрСсТтФфХхЦцЧчШшЩщ";

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

{

for (int j = 0; j < s[i].line.length(); j++)

{

if (s[i].line[j] != ' ')

{

if (!word\_started)

{

word\_started = true;

num\_word++;

cons\_in\_word = 0;

}

if (is\_char\_in\_string(s[i].line[j], cons))

{

cons\_in\_word++;

}

}

if (s[i].line[j] == ' ' || j + 1 == s[i].line.length())

{

word\_started = false;

if (cons\_in\_word > max\_cons)

{

max\_cons = cons\_in\_word;

pos\_max\_cons = num\_word;

}

}

}

s[i].num\_word = num\_word;

s[i].pos\_max\_cons = pos\_max\_cons;

num\_word = 0;

max\_cons = 0;

cons\_in\_word = 0;

pos\_max\_cons = 0;

word\_started = false;

}

}

void write\_in\_file(string\_ex arr[], ofstream& file, int size)

{

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

{

if (arr[i].num\_word == 1)

{

file << arr[i].line << endl;

}

}

}

int main()

{

system("chcp 1251>null");

ifstream ifile("inLab9.txt");

remove("OutLab9.txt");

ofstream outfile("OutLab9.txt");

if (ifile.is\_open())

{

int arr\_size = get\_file\_rows(ifile);

string\_ex\* data = new string\_ex[arr\_size];

read\_file(ifile, data);

string\_processing(data, arr\_size);

print\_arr(data, arr\_size);

write\_in\_file(data, outfile, arr\_size);

}

else

{

cout << "File inLab9.txt not found";

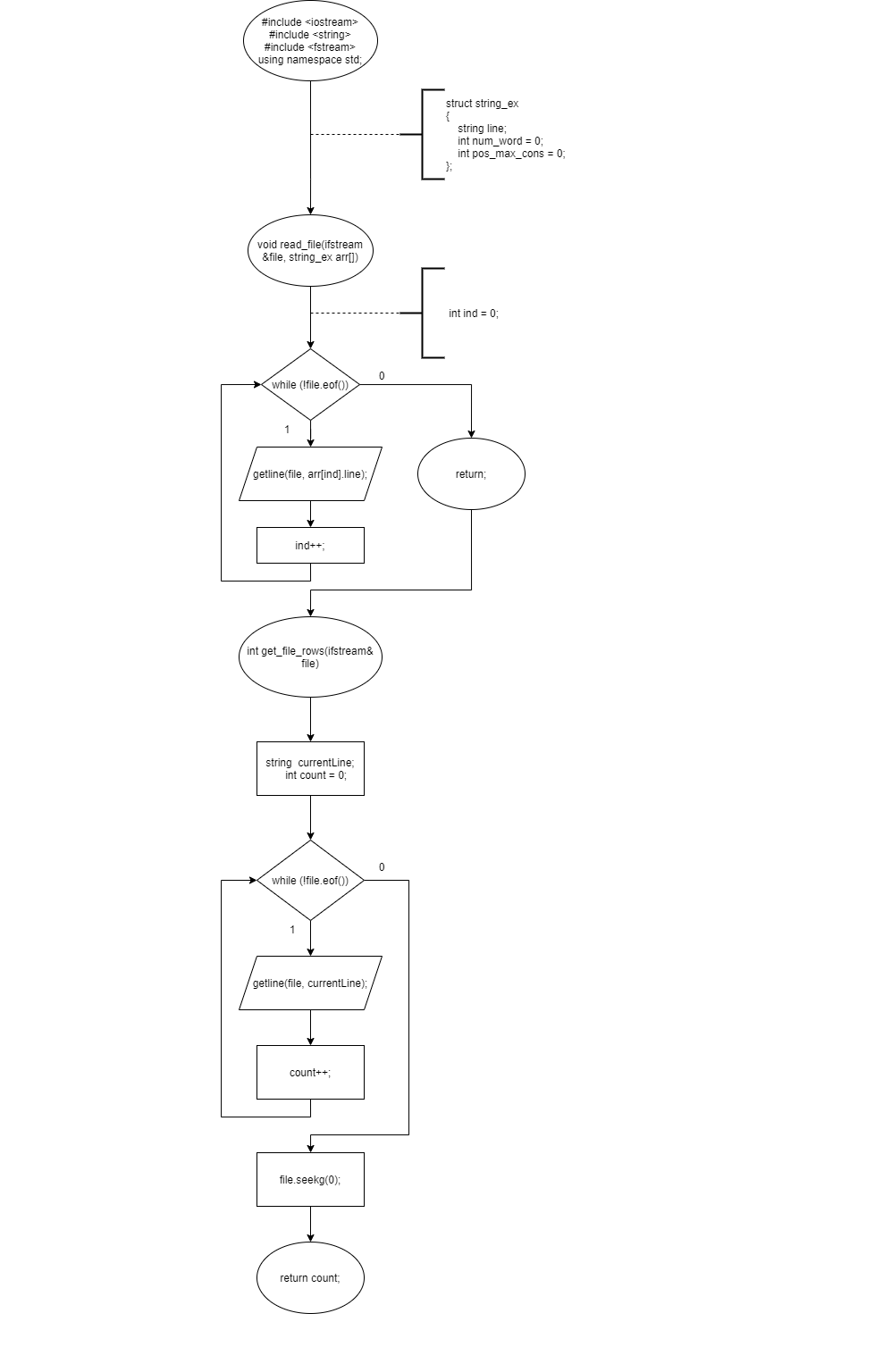
}

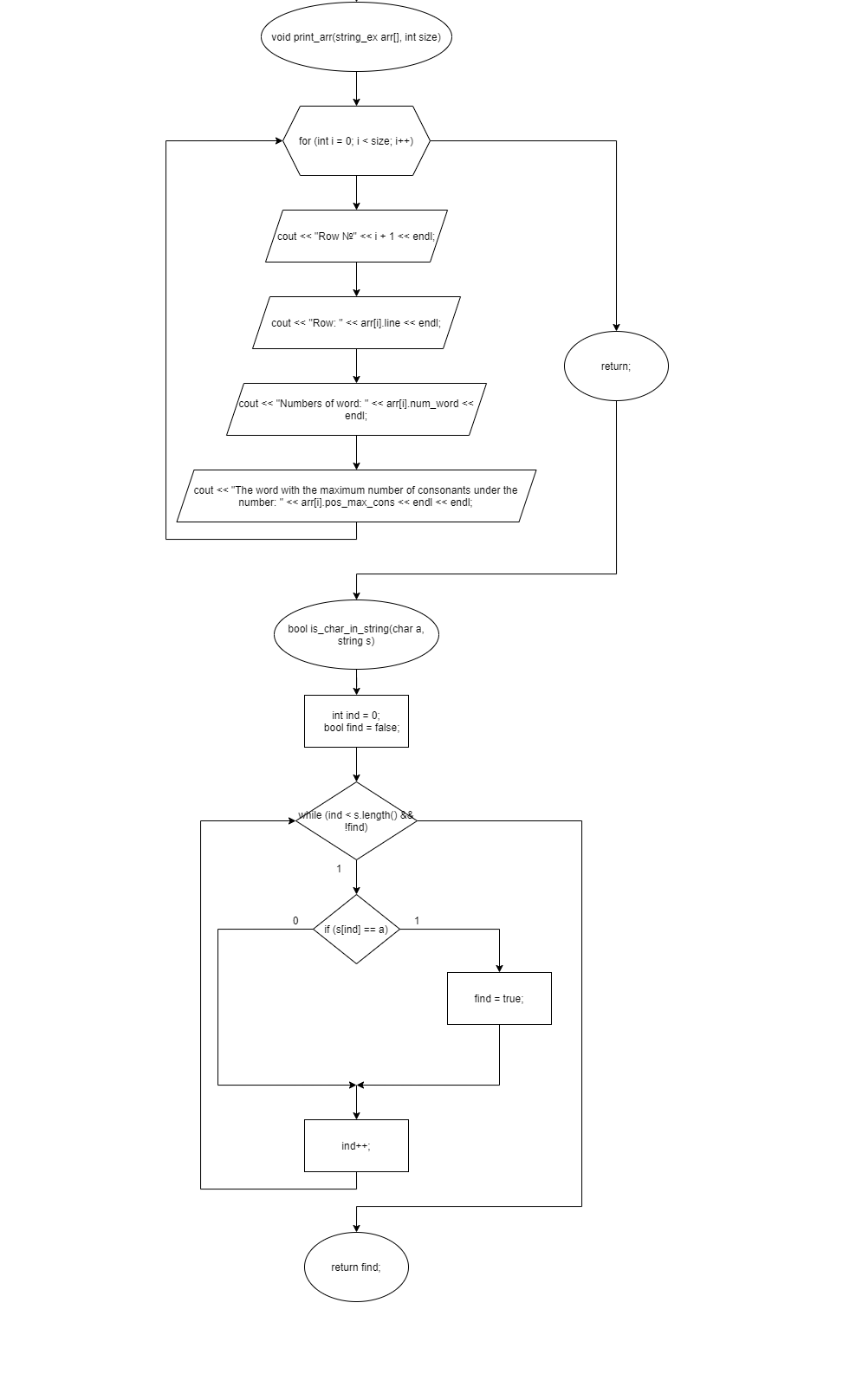
ifile.close();

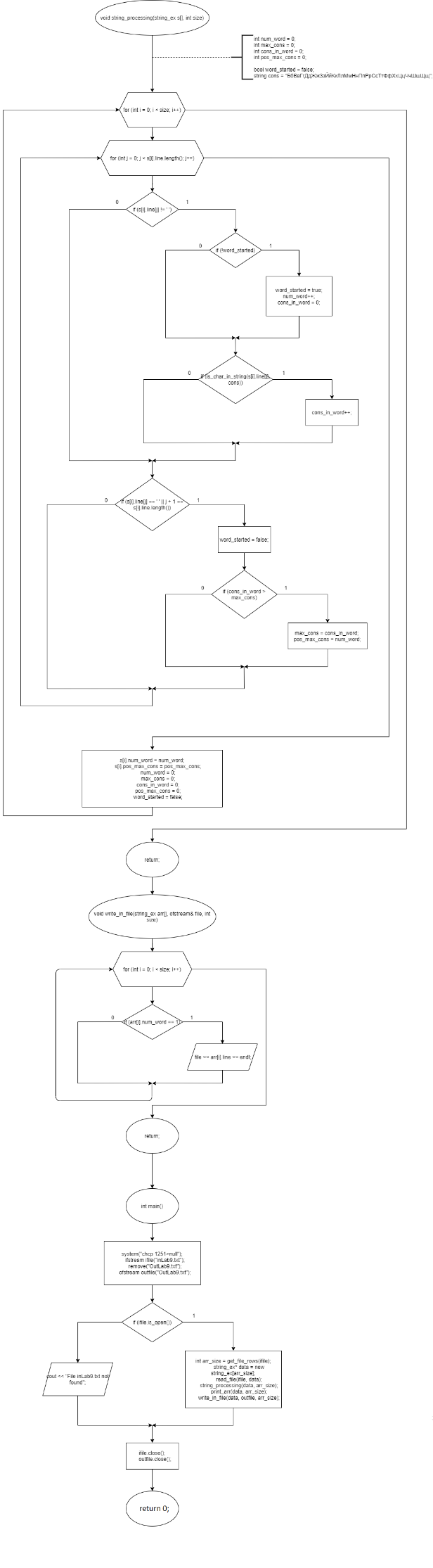
outfile.close();

}

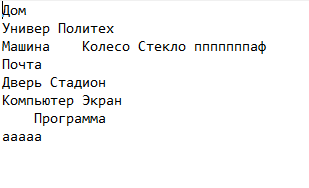
**Блок-схема**

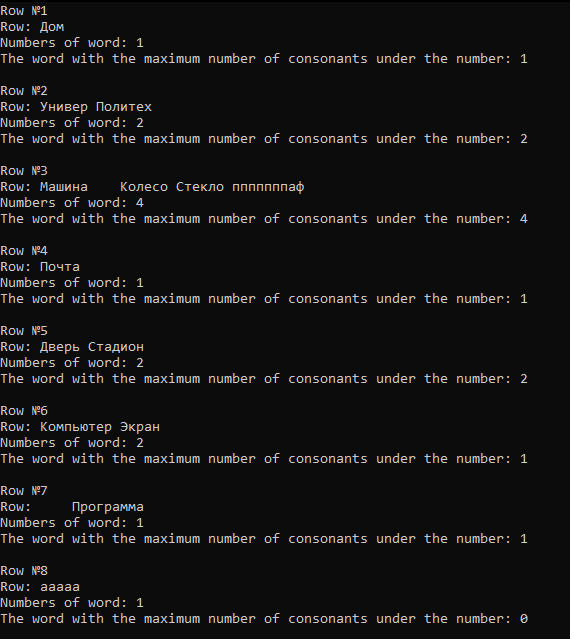
****

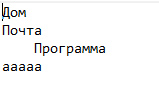
****

****

**Скриншоты тестов**

****

****

****