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 №15**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: Методы внутренней сортировки массивов: быстрая сортировка, сортировка подсчётом

Вариант 17

Выполнил:

Студент группы Рис-20-1б

Томилов Владислав Алексеевич

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

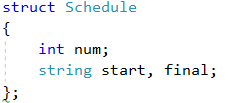
**Постановка задачи**

Дано железнодорожное расписание. Упорядочить его по номеру маршрута следования поездов.

**Анализ задачи (Метод Шелла)**

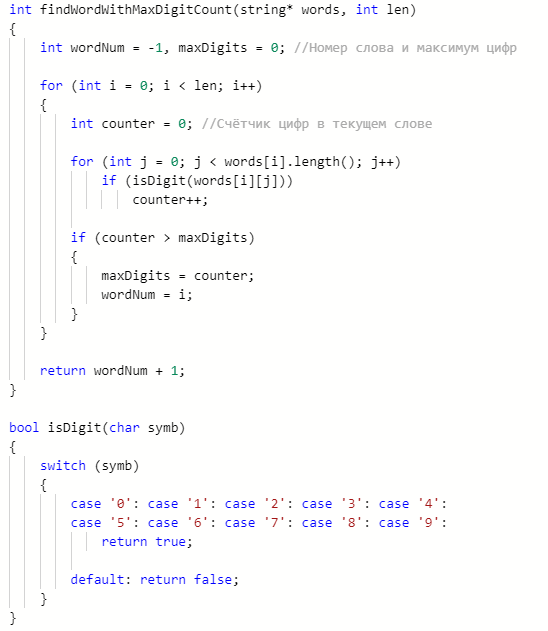
**1.** Для решения задачи необходимо:

**1.1.** Создать структуру расписания Schedule:

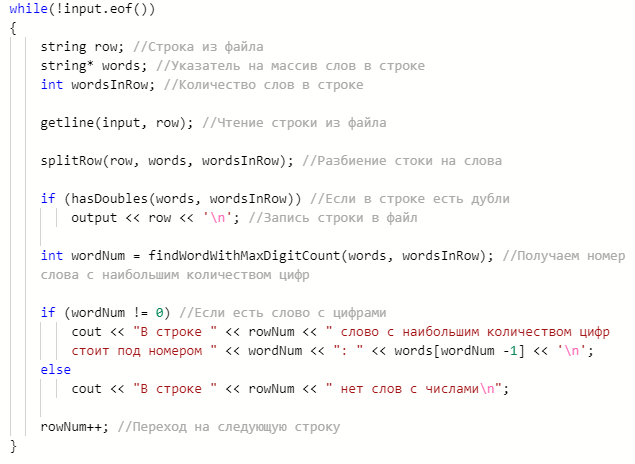


**1.2.** Разработать функцию ShellSort (Schedule\* schedule, int size),

**1.3.**Разработать функцию findWordWithMaxDigitCount, чтобы определить номер слова с наибольшим количеством цифр и фунциюisDigit проверяющую, является ли символ цифрой.



**1.4.** Реализовать применение этих функций для каждой строки файлаF1.txt.



**2.**В ходе работы были использованы следующие типы данных:

**2.1.** Для получения данных из файла используется класс ifstream:

****

**2.2.** Для вывода данных в файл используется класс ofstream:



**2.3.** Для функции splitRow используются следующие аргументы:

1. Тип string: Входная строка, которая будет разделена на слова.

2. Ссылка на указатель на string: Массив из string, полученный в результате работы функции.

3. Ссылка на int: Количество слов в строке.

Сама функция имеет тип void, так как в ходе ее работы нет необходимости возвращать значение для этой функции.



**2.4.** Для функцииwordsCounterиспользуются следующие аргументы:

1. Тип string: Входная строка, в которой необходимо найти количество слов.

Сама функция имеет тип int для возвращения количества слов в строке.



**2.5.** Для функции splitRowOnWords используются следующие аргументы:

1. Указатель на string: Массив со словами.

2. Тип string: Входная строка, которая будет разбита на слова

3. Типint: Количество слов в строке.

Сама функция имеет тип void, так как в ходе ее работы нет необходимости возвращать значение для этой функции.



**2.6.** Для функции findWordWithMaxDigitCount используются следующие аргументы:

1.Указатель на string:Массив слов.

2. Тип int: Количество слов в массиве.

Сама функция имеет тип int для возвращения позиции слова с наибольшим количеством цифр в составе.



**2.7.** Для функции isDigit используются следующие аргументы:

1. Тип char: Входной символ.

Сама функция имеет тип bool для передачи информации о том, является ли символ цифрой.



**2.8.** Для функции hasDoublesиспользуются следующие аргументы:

1. Указатель на string: Массив слов.

2. Тип int: Количество слов в массиве.



**3.**Для решения задачи данные были представлены в следующем виде:

**3.1.** Для обработки строк из файла данныебыли записанны в виде массива из слов (массив из string).

**4.**Для операций ввода и вывода использовались следующие операторы и функции:

**4.1.**Ввод данных из файла F1.txt реализован с помощью функции getline.



**4.2.** Вывод данных на консоль реализован с помощью оператора cout.



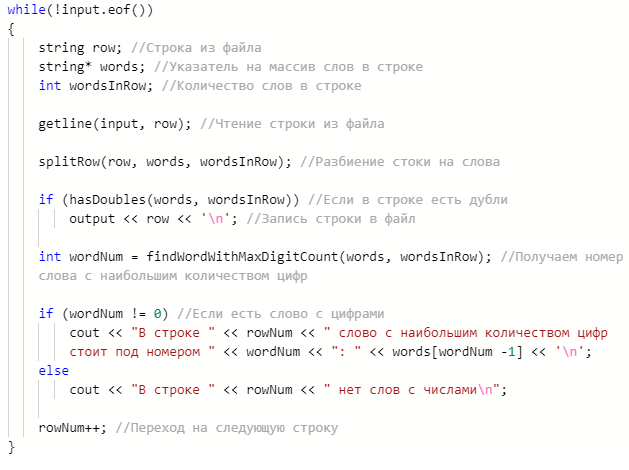


**4.3.**Вывод данных в файл реализован с помощью оператора вывода в файл.

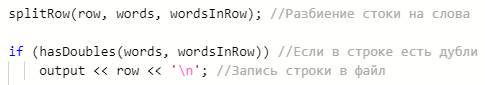


**5.**Поставленные задачи будут решены следующими действиями:

**5.1.** Для работы с информацией из файла построчно, все операции со строкой проводятся внутри цикла while, после каждого полного прохождения тела цикла производится считывание следующей строки.Повторение цикла продолжается, пока не достигнут конец файла.



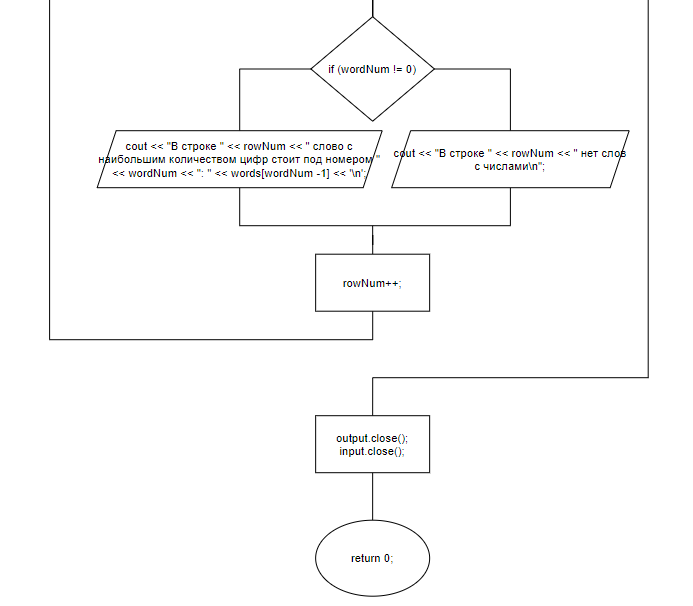
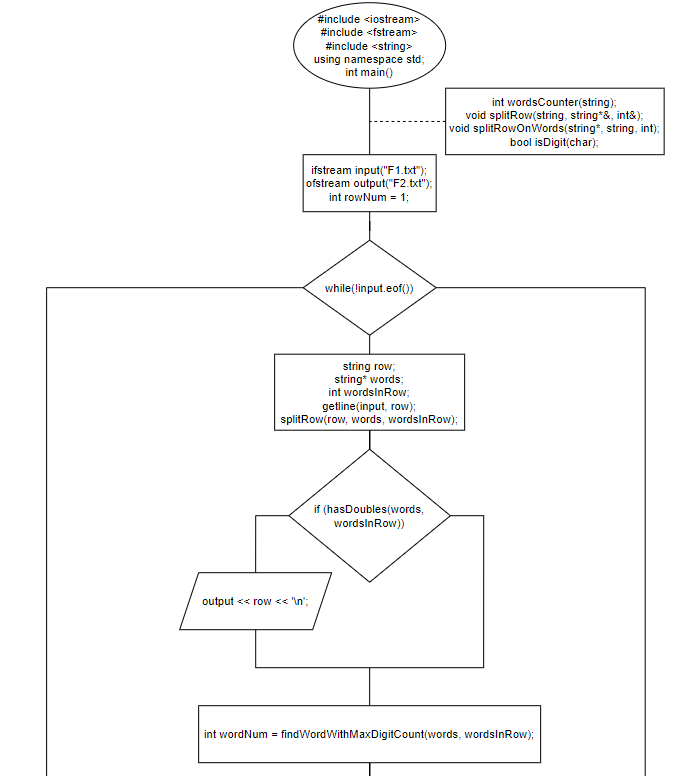
**5.2.**Поиск и копирование строк из файла F1 в F2реализованы с помощью функций splitRowи hasDoubles.

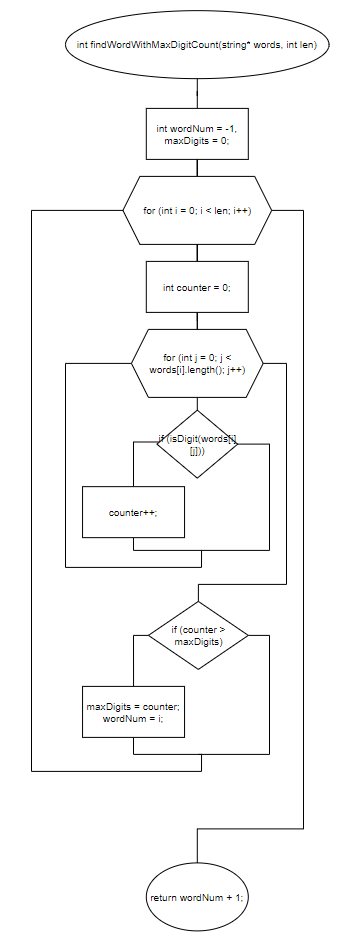
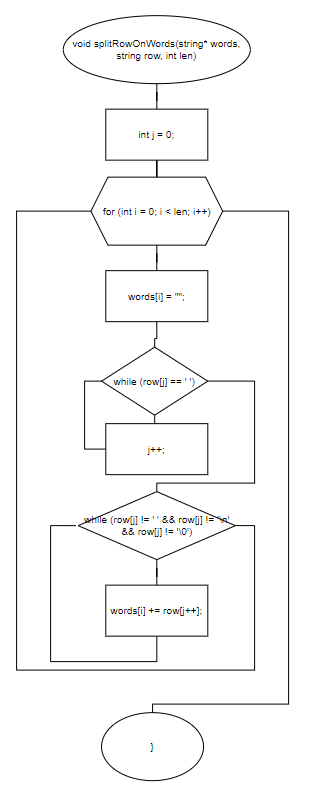
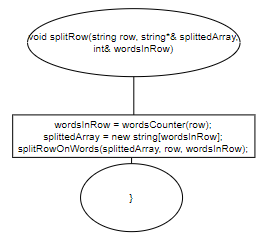
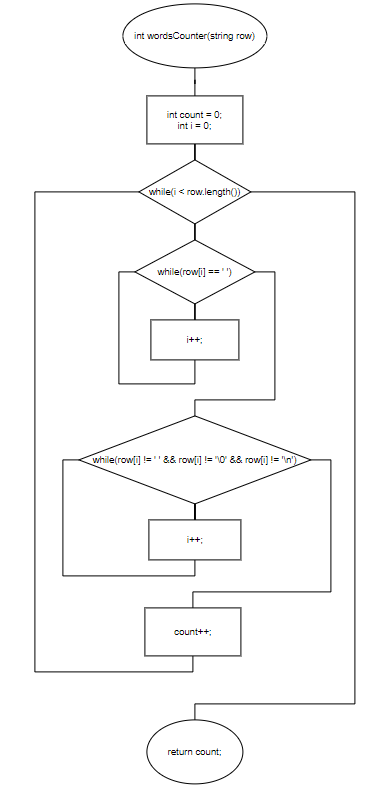
****

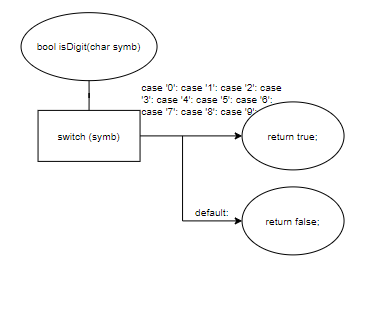
**5.3.** Определение позиции слова с наибольшим количеством цифр в составе реализовано с использованием функции findWordWithMaxDigitCount

****

**Блок-схемапрограммы**







**Решение**

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

intwordsCounter(string); //Подсчётслов в строке

voidsplitRow(string, string\*&, int&); //Разбиение строки по словам, нахождение их количества

voidsplitRowOnWords(string\*, string, int); //Разбиение строки по словам, запись в массив

boolhasDoubles(string\*, int); //Есть ли в строке дубли

intfindWordWithMaxDigitCount(string\*, int); //Поиск слова с максимальным количеством цифр

boolisDigit(char); //Проверка является ли символ цифрой

voidsolution()

{

ifstreaminput("F1.txt"); //Открываем файл с исходными данными

if (input.is\_open()) //если файл был открыт

{

ofstreamoutput("F2.txt"); //открываем файл для записи

introwNum = 1; //Номер обрабатываемой строки

while(!input.eof())

{

stringrow; //Строка из файла

string\* words; //Массив слов в строке

intwordsInRow; //Количество слов в строке

getline(input, row); //Чтение строки из файла

splitRow(row, words, wordsInRow); //Разбиение стоки на слова

if (hasDoubles(words, wordsInRow)) //Если в строкеестьдубли

output<<row<< '\n'; //Запись строки в файл

intwordNum = findWordWithMaxDigitCount(words, wordsInRow); //Получаем номер слова с наибольшим количеством цифр

if (wordNum != 0) //Если есть слово с цифрами

cout<< "В строке " <<rowNum<< " слово с наибольшим количеством цифр стоит под номером " <<wordNum<< ": " <<words[wordNum -1] << '\n';

else

cout<< "В строке " <<rowNum<< " нет слов с числами\n";

rowNum++; //Переход на следующую строку

}

output.close(); //Закрываем файл для вывода

}

else //Если файл не был открыт - сообщение об ошибке

cout<< "Не удалось открыть файл F1.txt\n";

input.close(); //Закрываем файл с исходными данными

}

intwordsCounter(string row)

{

int count = 0; //количествослов

inti = 0;

while(i<row.length())

{

while(row[i] == ' ') i++; //Проходимпробелы

while(row[i] != ' ' && row[i] != '\0' && row[i] != '\n') i++; //Проходим буквы

count++; //Прошли все буквы => +1 слово

}

return count;

}

voidsplitRow(string row, string\*&splittedArray, int&wordsInRow)

{

wordsInRow = wordsCounter(row); //Подсчётслов в строке

splittedArray = new string[wordsInRow]; //Создаёммассивподслова

splitRowOnWords(splittedArray, row, wordsInRow); //Записываемслова в массив

}

voidsplitRowOnWords(string\* words, string row, intlen)

{

int j = 0;

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

{

words[i] = "";

while (row[j] == ' ') j++;

while (row[j] != ' ' && row[j] != '\n' && row[j] != '\0') words[i] += row[j++];

}

}

boolhasDoubles(string\* words, intlen)

{

for (inti = 0; i< len-1; i++)

for (int j = i+1; j <len; j++)

if (words[i] == words[j])

returntrue; //Поиск прекращается при нахождении повтора

return false;

}

intfindWordWithMaxDigitCount(string\* words, intlen)

{

intwordNum = -1, maxDigits = 0; //Номер слова и максимум цифр

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

{

intcounter = 0; //Счётчик цифр в текущем слове

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

if (isDigit(words[i][j]))

counter++;

if (counter >maxDigits)

{

maxDigits = counter;

wordNum = i;

}

}

returnwordNum + 1;

}

boolisDigit(char symb)

{

switch (symb)

{

case '0': case '1': case '2': case '3': case '4':

case '5': case '6': case '7': case '8': case '9':

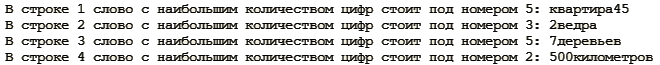
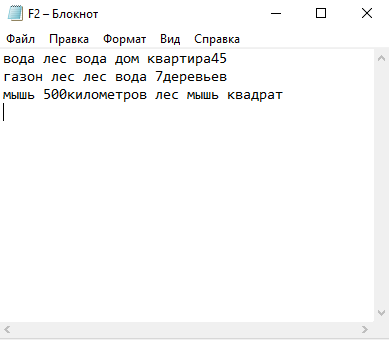
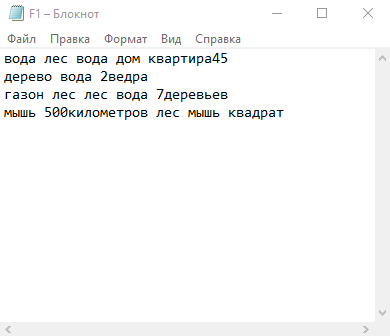
return true;

default: return false;

}

}

**Скриншоты результатов работы программы**

****