

- **Цель работы**

Целью курсового проектирования является закрепление теоретических и практических знаний, полученных на лекционных, лабораторных и практических занятиях по курсу, а также получение навыков разработки, отладки и тестирования программ на алгоритмических языках программирования.

- **Задание**

3	<p>Предметная область – «Расписание рейсов самолетов».</p> <p>Данные о рейсе хранятся в структуре с именем AEROFLOT, содержащей следующие поля:</p> <ul style="list-style-type: none">• название пункта назначения рейса;• номер рейса;• тип самолёта. <p>Задание на поиск: найти рейсы, вылетающие в пункт назначения, название которого совпало с названием, введённым с клавиатуры.</p>
---	--

- **Описание созданных функций**

Для реализации задания были использованы следующие функции:

Имя: input_string.

Назначение: ввод строки, пробелы заменяются на нижние подчёркивания;

Входные данные:

Выходные данные:

- Отформатированная введённая строка.

Побочный эффект: отсутствует.

Прототип: string input_string()

Алгоритм:

- псевдокод

Считать введённое слово и, если, есть пробелы, то заменить на _

Имя: input_int.

Назначение: ввод числа с проверкой на числовое значение

Входные данные:

Выходные данные:

- принадлежит/не принадлежит поступивший символ числовому значению

Побочный эффект: отсутствует.

Прототип: input_int();

Алгоритм:

- псевдокод

Взять поступивший символ и сравнить его с заданным диапазоном чисел, если прошёл проверку, то вернуть число, если нет, выдать ошибку и повторить ввод

Имя: compareByDestination

Назначение: сравнение двух пунктов прибытия.

Входные данные:

Aeroflot &a1 – объект структуры Aeroflot

Aeroflot &a2 – объект структуры Aeroflot

Выходные данные:

- какая из строк является меньшей

Побочный эффект: отсутствует.

Прототип: bool compareByDestination(const Aeroflot &a1, const Aeroflot &a2)

Алгоритм:

- псевдокод

Взять из структуры Aeroflot a1 поле destination и сравнить его с таким же полем структуры Aeroflot a2.

Имя: Aeroflot

Назначение: структура, хранящая в себе несколько разных значений.

Входные данные:

Выходные данные:

Побочный эффект: отсутствует.

Прототип: Aeroflot;

Алгоритм:

- псевдокод

выделить память под string;

выделить память под int;

выделить память под string;

Имя: menu

Назначение: функция для вывода пунктов меню.

Входные данные:

Выходные данные:

- вывод пунктов меню в консоль

Побочный эффект: отсутствует.

Прототип: menu();

Алгоритм:

- псевдокод

вывести в консоль сообщение

Имя: load

Назначение: загрузить из файла записи.

Входные данные:

Выходные данные:

Побочный эффект: отсутствует.

Прототип: void Database::load();

Алгоритм:

- псевдокод

Открыть файл, если файл не был найден, вывести ошибку.

Считать количество записей в файле.

Создать объект типа Aeroflot.

Прочитать первую запись и записать её в объект типа Aeroflot.

Имя: print

Назначение: печать в консоль.

Входные данные:

Выходные данные:

Побочный эффект: отсутствует.

Прототип: void Database::print();

Алгоритм:

- псевдокод

В цикле выводятся записи в консоль.

Имя: append

Назначение: добавление рейсов.

Входные данные:

Выходные данные:

Побочный эффект: отсутствует.

Прототип: void Database::append();

Алгоритм:

- псевдокод

Создается объект типа Aeroflot

Вывод на консоль вводимого пункта рейса.

Ввод пункта рейса.

Имя: find

Назначение: поиск рейсов по пункту назначения.

Входные данные:

string destination – искомый пункт прибытия

Выходные данные:

Побочный эффект: отсутствует.

Прототип: void Database::find(string destination);

Алгоритм:

- псевдокод

Сравнить искомый пункт прибытия и, если пункт найден, то напечатать его.

Имя: remove

Назначение: удаление рейса по номеру рейса.

Входные данные:

int id– номер рейса

Выходные данные:

Побочный эффект: отсутствует.

Прототип: void Database::remove(int id);

Алгоритм:

- псевдокод

Создать переменную, указывающую на начало вектора, состоящего из структур.

В цикле пройти до конца вектора и вычислить итератор, если итератор равен номеру удаляемого рейса, то удалить его итератор.

Имя: save

Назначение: сохранение в файл рейсов.

Входные данные:

Выходные данные:

Побочный эффект: отсутствует.

Прототип: void Database::save();

Алгоритм:

- псевдокод

Открыть файл для записи, если он не был открыт, то вывести ошибку.

Считать размер вектора и записать его в файл.

В цикле вывести каждую структуру в файл.

Имя: sort_by_destination

Назначение: сортировка по пункту прибытия.

Входные данные:

Выходные данные:

Побочный эффект: отсутствует.

Прототип: void Database::sort_by_destination();

Алгоритм:

- псевдокод

Сортировать вектор исходя из сравнения сделанного в функции сравнения.

Имя: is_number

Назначение: проверка является ли число введённым значением.

Входные данные:

Выходные данные:

Побочный эффект: отсутствует.

Прототип: bool is_number(string s);

Алгоритм:

- псевдокод

Сравнить каждый символ введённого значения на принадлежность к цифре.

Если хотя бы одно значение не совпадает, то вывести false, если всё хорошо, то true.

Имя: flight_id_exists

Назначение: совпадает ли введённое число с номером уже существующего рейса.

Входные данные:

Выходные данные:

Побочный эффект: отсутствует.

Прототип: bool Database::flight_id_exists(int id);

Алгоритм:

- псевдокод

Если введённое значение совпадает с уже существующим то, true, иначе else.

Имя: edit

Назначение: редактирование пунктов рейса по номеру рейса.

Входные данные:

Выходные данные:

Побочный эффект: отсутствует.

Прототип: void Database::edit(int id);

Алгоритм:

- псевдокод

Проверить совпадает ли введённый адрес рейса с существующим, если совпал, то вводим новые данные для рейса, если нет, то выдаём ошибку что такого рейса нет.

4. Листинг программы

```
1. #include <iostream>
2. #include <iomanip> // Манипуляторы ввода-вывода (табличная печать)
3. #include <fstream>
4. #include <vector>
5. #include <string>
6. #include <algorithm>
7. #include <Windows.h>
8.
9. using namespace std;
10.
11. // Ввод строки, пробелы заменяются на подчеркивания
12. string input_string()
13. {
14.     string s;
15.     getline(cin, s);
16.
17.     // Заменяем все пробелы на подчеркивание
18.     for (size_t i = 0; i < s.size(); i++)
19.     {
20.         if (s[i] == ' ' || s[i] == '\t')
21.             s[i] = '_';
22.     }
23.     return s;
24. }
25.
26. bool is_number(string s)
27. {
28.     for (size_t i = 0; i < s.size(); i++)
29.     {
30.         if (s[i] < '0' || s[i] > '9')
31.         {
32.             return false;
33.         }
34.     }
35.     return true;
36. }
37.
38. int is_number1(string s)
39. {
40.     int n;
41.     for (size_t i = 0; i < s.size(); i++)
42.     {
43.         if (s[i] < '0' || s[i] > '9')
44.         {
45.             size_t sz;
46.             n = stoi(s, &sz);
47.         }
48.     }
49.     return n;
50. }
51.
52. // ввод числа с проверкой, если не число, то повторный ввод
53. int input_int()
54. {
55.
56.     while (1)
57.     {
58.         string s;
59.         getline(cin, s);
60.         if (!s.empty())
61.         {
62.             if (is_number(s))
```

```

63.         {
64.             size_t sz;
65.             int value = stoi(s, &sz);
66.             return value;
67.         }
68.     }
69.
70.     cout << "Ошибка: ожидается число!" << endl;
71. }
72. }
73. //-----
74.
75. struct Aeroflot
76. {
77.     string destination;
78.     int flight_id;
79.     string plane_type;
80.
81.     void print() const {
82.         // setw - задает ширину поля для вывода
83.         cout << left;
84.         cout << setw(20) << destination;
85.         cout << setw(15) << flight_id;
86.         cout << setw(20) << plane_type << endl;
87.     }
88. };
89.
90. class Database
91. {
92.     vector<Aeroflot> flight;
93. public:
94.     void load();
95.     void print();
96.     void append();
97.     void find(string destination);
98.     void save();
99.     void sort_by_destination();
100.    void remove(int id);
101.    void edit(int id);
102.
103.    bool flight_id_exists(int id);
104. };
105.
106. void menu() // ф-ция для вывода пунктов меню в консоль
107. {
108.     cout << "0. Выйти из программы" << endl;
109.     cout << "1. Загрузить из файла" << endl;
110.     cout << "2. Печать рейсов" << endl;
111.     cout << "3. Добавить рейс" << endl;
112.     cout << "4. Найти рейс по пункту назначения" << endl;
113.     cout << "5. Сохранить в файл" << endl;
114.     cout << "6. Сортировать по пункту назначения" << endl;
115.     cout << "7. Удалить рейс по номеру" << endl;
116.     cout << "8. Изменить рейс по номеру" << endl;
117.     cout << endl;
118. }
119.
120. void Database::load()
121. {
122.     ifstream fin("input.txt");
123.     if (!fin)
124.     {
125.         cout << "Ошибка: не могу открыть файл!" << endl;
126.         return;
127.     }

```



```

128.
129.         flight.clear();
130.         int n;
131.         fin >> n;
132.
133.         if (!fin.good()) {
134.             cout << "Ошибка формата файла или пустой файл!" << endl;
135.             return;
136.         }
137.
138.         if (n == 0) {
139.             cout << "В файле нет записей" << endl;
140.         }
141.
142.         for (int i = 0; i < n; i++)
143.         {
144.             Aeroflot f;
145.             fin >> f.destination;
146.
147.             string str_id;
148.             fin >> str_id;
149.             if (is_number(str_id))
150.             {
151.                 size_t sz;
152.                 f.flight_id = stoi(str_id, &sz);
153.
154.                 if (flight_id_exists(f.flight_id)) {
155.                     fin >> f.plane_type;
156.                     continue;
157.                 }
158.             }
159.             else {
160.                 fin >> f.plane_type;
161.                 continue;
162.             }
163.
164.             fin >> f.plane_type;
165.             flight.push_back(f);
166.         }
167.
168.         fin.close();
169.     }
170.
171.     void Database::print()
172.     {
173.         cout << left;
174.         cout << setw(20) << "Пункт назначения";
175.         cout << setw(15) << "Номер рейса";
176.         cout << setw(20) << "Тип самолета" << endl;
177.
178.         for (size_t i = 0; i < flight.size(); i++)
179.         {
180.             flight[i].print();
181.         }
182.
183.         cout << endl;
184.     }
185.
186.     bool Database::flight_id_exists(int id)
187.     {
188.         // Проверим, что рейса с таким номером еще нет
189.         for (int i = 0; i < flight.size(); i++) {
190.             if (flight[i].flight_id == id) {
191.                 return true; // рейс найден
192.             }

```

```

193.     }
194.     return false;
195. }
196.
197. void Database::append()
198. {
199.     Aeroflot f;
200.     cout << "Номер рейса: ";
201.     f.flight_id = input_int();
202.
203.     if (flight_id_exists(f.flight_id)) {
204.         cout << "Ошибка: рейс с таким номером уже есть!" << endl;
205.         return;
206.     }
207.
208.     cout << "Пункт назначения: ";
209.     f.destination = input_string();
210.     cout << "Тип самолета: ";
211.     f.plane_type = input_string();
212.
213.     flight.push_back(f);
214. }
215.
216. void Database::find(string destination)
217. {
218.     for (const auto& f : flight)
219.     {
220.         if (f.destination == destination)
221.         {
222.             f.print();
223.         }
224.     }
225. }
226.
227. void Database::remove(int id)
228. {
229.     auto it = flight.begin();
230.     for (; it != flight.end(); ++it)
231.     {
232.         if (it->flight_id == id)
233.             break;
234.     }
235.
236.     if (it != flight.end())
237.         flight.erase(it);
238.     else
239.         cout << "Удаляемый рейс не найден!" << endl;
240. }
241.
242. void Database::edit(int id)
243. {
244.     auto it = flight.begin();
245.     for (; it != flight.end(); ++it)
246.     {
247.         if (it->flight_id == id)
248.             break;
249.     }
250.
251.     if (it != flight.end())
252.     {
253.         Aeroflot f;
254.         cout << "Пункт назначения: ";
255.         f.destination = input_string();
256.
257.         do {

```

```

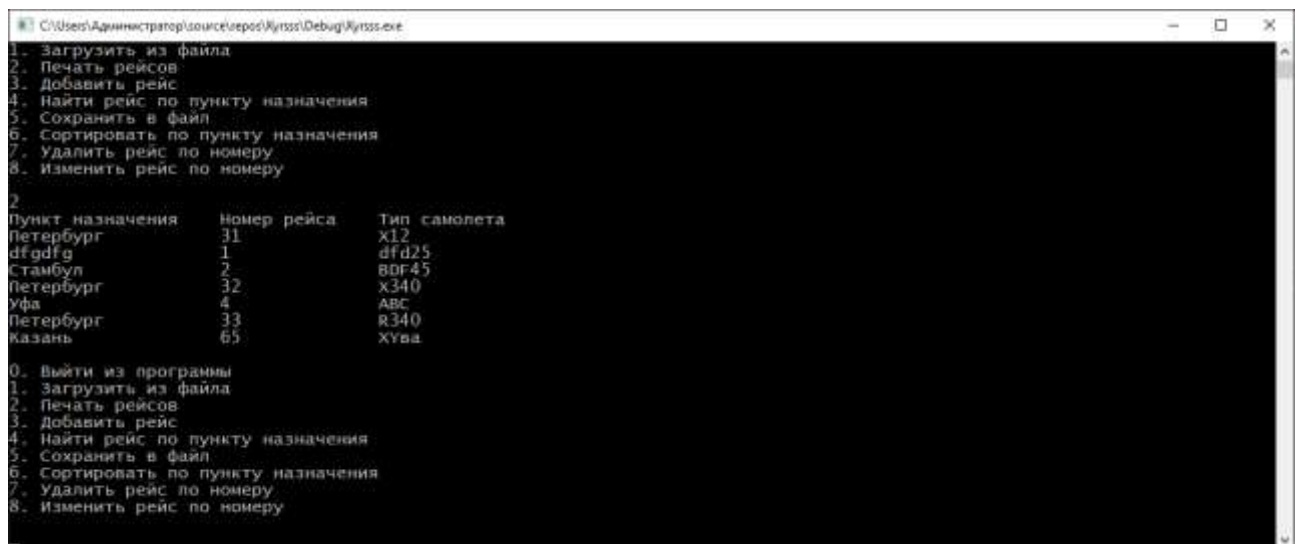
258.             cout << "Номер рейса: ";
259.             f.flight_id = input_int();
260.             if (flight_id_exists(f.flight_id) && f.flight_id != it-
>flight_id) {
261.                 cout << "Ошибка: рейс с таким номером уже есть!"
<< endl;
262.             }
263.             else
264.                 break;
265.         } while (1);
266.
267.         cout << "Тип самолета: ";
268.         f.plane_type = input_string();
269.         *it = f;
270.     }
271.     else
272.         cout << "Редактируемый рейс не найден!" << endl;
273. }
274.
275. void Database::save()
276. {
277.     ofstream fout("input.txt");
278.
279.     int n = flight.size();
280.     fout << n << endl;
281.
282.     for (const auto& f : flight) {
283.         fout << f.destination << "\t" << f.flight_id << "\t" <<
f.plane_type << endl;
284.     }
285.
286.     fout.close();
287. }
288.
289. bool compareByDestination(const Aeroflot &a1, const Aeroflot &a2)
290. {
291.     return a1.destination < a2.destination;
292. }
293.
294. void Database::sort_by_destination()
295. {
296.     sort(flight.begin(), flight.end(), compareByDestination);
297. }
298.
299. int main()
300. {
301.     setlocale(LC_ALL, "Russian");
302.     SetConsoleCP(1251); // Ввод с консоли в кодировке 1251
303.     SetConsoleOutputCP(1251);
304.
305.     Database db;
306.     string n;
307.     int num;
308.
309.     int choise = 0;
310.
311.     do
312.     {
313.         menu();
314.         cin >> choise;
315.         cin.get();
316.         switch (choise)
317.         {
318.             case 1:
319.                 db.load();

```

```
320.         break;
321.     case 2:
322.         db.print();
323.         break;
324.     case 3:
325.         db.append();
326.         break;
327.     case 4:
328.         cout << "Введите пункт назначения искомого рейса: ";
329.         //cin >> n;
330.         n = input_string();
331.         db.find(n);
332.         break;
333.     case 5:
334.         db.save();
335.         break;
336.     case 6:
337.         db.sort_by_destination();
338.         break;
339.     case 7:
340.         cout << "Введите номер удаляемого рейса: ";
341.         num = input_int();
342.         db.remove(num);
343.         break;
344.     case 8:
345.         cout << "Введите номер редактируемого рейса: ";
346.         int num = input_int();
347.         db.edit(num);
348.         break;
349.     }
350. } while (choise != 0);
351.
352. return 0;
353. }
```

5. Пример выполнения программы

Ниже приведён пример выполнения программы

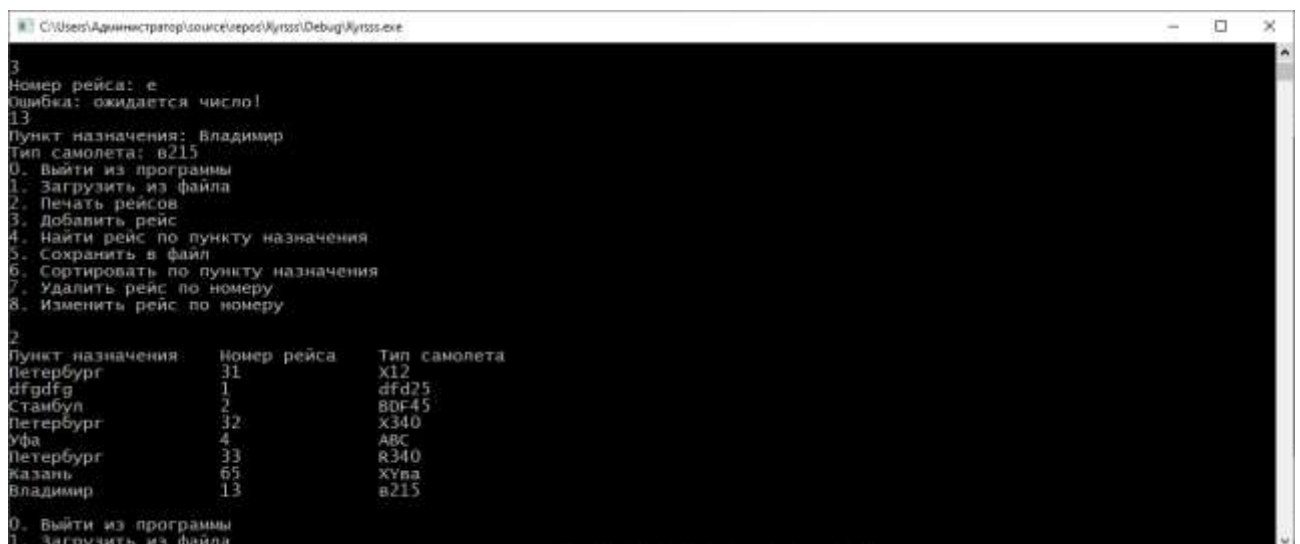


```
C:\Users\Администратор\source\repos\X\Xr\Debug\Xr.exe
1. Загрузить из файла
2. Печать рейсов
3. Добавить рейс
4. Найти рейс по пункту назначения
5. Сохранить в файл
6. Сортировать по пункту назначения
7. Удалить рейс по номеру
8. Изменить рейс по номеру

2
Пункт назначения    Номер рейса    Тип самолета
Петербург           31             X12
dfgdfg              1              dfd25
Стамбул             2              BDF45
Петербург           32             X340
Уфа                 4              ABC
Петербург           33             R340
Казань              65             XYva

0. Выйти из программы
1. Загрузить из файла
2. Печать рейсов
3. Добавить рейс
4. Найти рейс по пункту назначения
5. Сохранить в файл
6. Сортировать по пункту назначения
7. Удалить рейс по номеру
8. Изменить рейс по номеру
```

Рис 1. Пример выполнения программы



```
C:\Users\Администратор\source\repos\X\Xr\Debug\Xr.exe
3
Номер рейса: e
Ошибка: ожидается число!
13
Пункт назначения: Владимир
Тип самолета: v215
0. Выйти из программы
1. Загрузить из файла
2. Печать рейсов
3. Добавить рейс
4. Найти рейс по пункту назначения
5. Сохранить в файл
6. Сортировать по пункту назначения
7. Удалить рейс по номеру
8. Изменить рейс по номеру

2
Пункт назначения    Номер рейса    Тип самолета
Петербург           31             X12
dfgdfg              1              dfd25
Стамбул             2              BDF45
Петербург           32             X340
Уфа                 4              ABC
Петербург           33             R340
Казань              65             XYva
Владимир            13             v215

0. Выйти из программы
1. Загрузить из файла
```

Рис 2. Пример выполнения программы

```
C:\Users\Администратор\source\repos\flyrussi\Debug\flyrussi.exe
Петербург      33      я340
казань         65      ХУва
Владимир       13      в215

0. Выйти из программы
1. Загрузить из файла
2. Печать рейсов
3. Добавить рейс
4. Найти рейс по пункту назначения
5. Сохранить в файл
6. Сортировать по пункту назначения
7. Удалить рейс по номеру
8. Изменить рейс по номеру

4
Введите пункт назначения искомого рейса: Петербург
Петербург      31      х12
Петербург      32      х340
Петербург      33      я340

0. Выйти из программы
1. Загрузить из файла
2. Печать рейсов
3. Добавить рейс
4. Найти рейс по пункту назначения
5. Сохранить в файл
6. Сортировать по пункту назначения
7. Удалить рейс по номеру
8. Изменить рейс по номеру
```

Рис 3. Пример выполнения программы

```
C:\Users\Администратор\source\repos\flyrussi\Debug\flyrussi.exe

2. Печать рейсов
3. Добавить рейс
4. Найти рейс по пункту назначения
5. Сохранить в файл
6. Сортировать по пункту назначения
7. Удалить рейс по номеру
8. Изменить рейс по номеру

2
Пункт назначения    Номер рейса    Тип самолета
dfgdfg              1              dfd25
Владимир            13             в215
казань              65             ХУва
Петербург           31             х12
Петербург           32             х340
Петербург           33             я340
Стамбул             2              В0645
уфа                 4              АВС

0. Выйти из программы
1. Загрузить из файла
2. Печать рейсов
3. Добавить рейс
4. Найти рейс по пункту назначения
5. Сохранить в файл
6. Сортировать по пункту назначения
7. Удалить рейс по номеру
8. Изменить рейс по номеру
```

Рис 4. Пример выполнения программы

```
C:\Users\Администратор\source\repos\flyrussi\Debug\flyrussi.exe

1. Загрузить из файла
2. Печать рейсов
3. Добавить рейс
4. Найти рейс по пункту назначения
5. Сохранить в файл
6. Сортировать по пункту назначения
7. Удалить рейс по номеру
8. Изменить рейс по номеру

2
Пункт назначения    Номер рейса    Тип самолета
dfgdfg              1              dfd25
казань              65             ХУва
Петербург           31             х12
Петербург           32             х340
Петербург           33             я340
Стамбул             2              В0645
уфа                 4              АВС

0. Выйти из программы
1. Загрузить из файла
2. Печать рейсов
3. Добавить рейс
4. Найти рейс по пункту назначения
5. Сохранить в файл
6. Сортировать по пункту назначения
7. Удалить рейс по номеру
8. Изменить рейс по номеру
```

Рис 5. Пример выполнения программы

```
C:\Users\Администратор\source\repos\X\irssn\Debug\X\irssn.exe
8
Введите номер редактируемого рейса: 65
Пункт назначения: Великий Новгород
Номер рейса: 333
Тип самолета: в6558
0. Выйти из программы
1. Загрузить из файла
2. Печать рейсов
3. Добавить рейс
4. Найти рейс по пункту назначения
5. Сохранить в файл
6. Сортировать по пункту назначения
7. Удалить рейс по номеру
8. Изменить рейс по номеру
2
Пункт назначения    Номер рейса    Тип самолета
dfgdfg             1             dfd25
Великий_Новгород    333           в6558
Петербург           31            X12
Петербург           32            X340
Петербург           33            R340
Стамбул             2             BDF45
Уфа                 4             ABC
0. Выйти из программы
1. Загрузить из файла
2. Печать рейсов
3. Добавить рейс
4. Найти рейс по пункту назначения
```

Рис 5. Пример выполнения программы

```
input.txt - Блокнот
Файл  Правка  Формат  Вид  Справка
7
dfgdfg 1      dfd25
Великий_Новгород    333    в6558
Петербург           31     X12
Петербург           32     X340
Петербург           33     R340
Стамбул 2        BDF45
Уфа      4        ABC
Строка 7, столбец 16    100%    Windows (CRLF)    ANSI
```

Рис 6. Пример выполнения программы