# МИНОБРНАУКИ РОССИИ

# САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

# ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

# «ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

# Кафедра САПР

# Отчет

# по лабораторной работе №3

# по дисциплине «Алгоритмы и структуры данных»

# Вариант: 1

|  |  |  |
| --- | --- | --- |
| Студентка гр. 9301 |  | Служевская А.С. |
| Преподаватель |  | Тутуева А.В. |

Санкт-Петербург

2021

**Постановка задачи.**

Дан список возможных авиарейсов в текстовом файле в формате:

Город отправления 1;Город прибытия 1;цена прямого перелета 1;цена обратного перелета 1

Город отправления 2;Город прибытия 2;цена перелета 2;цена обратного перелета 1

Город отправления N;Город прибытия N;цена перелета N;цена обратного перелета N

В случае, если нет прямого или обратного рейса, его цена будет указана как N/A (not available).

*Задание:* найти наиболее эффективный по стоимости перелет из города ***i*** в город ***j***.

Алгоритм и структура данных: алгоритм Дейкстры и списки смежности.

**Описание реализуемого класса и методов**

Хранение смежности и цен производится в списках. Для их хранения вместе используется массив списков. Номер города и его название хранятся в структуре для сохранения связи между ними. Структуры хранятся в массиве.

**Оценка временной сложности алгоритмов:**

Таблица 1. Временные затраты

|  |  |
| --- | --- |
| readList() | O(n) |
| Dijkstra() | O(n2) |
| writeList() | O(n) |

**Реализованные Unit-тесты:**

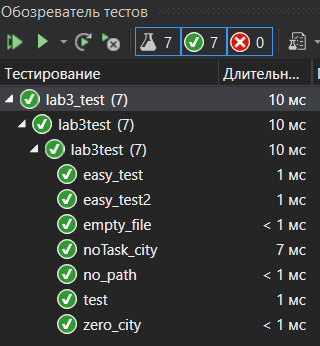


Рисунок 1 — Тесты

1. easy\_test — построение прямого пути между городами
2. easy\_test\_2 — проверка нахождения наиболее дешевого пути между городами
3. empty\_file — проверка ошибки при пустом файлу
4. noTask\_city — проверка ошибки при отсутствия городов прибытия или назначения
5. no\_path — проверка ошибки на изолированной вершине
6. test — проверка большего количества городов
7. zero\_city — проверка ошибки при отсутствии данных, но не пустом файле

**Пример работы**

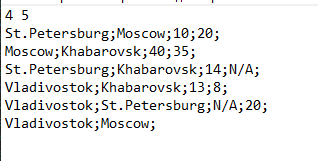


Рисунок 2 — Входные данные

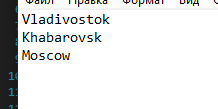


Рисунок 3 — Результат работы

**Листинг**

#include <iostream>

#include <fstream>

#include "List.h"

#include "Dijkstra.h"

using namespace std;

int main()

{

string input\_file\_name = "input.txt";

string output\_file\_name = "output.txt";

Algorithm Alg;

int size;

Alg.readList(input\_file\_name);

Alg.Dijkstra(size);

Alg.writeList(output\_file\_name);

}