**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №2**

**по дисциплине «Построение и анализ алгоритмов»**

**Тема: «Алгоритмы на графах»**

Студент гр. 6381 \_\_\_\_\_\_\_\_\_\_\_\_\_ Федянин Н.И.

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_ Филатов А.Ю.

Санкт-Петербург

2018

**Цель работы**

Изучение жадного алгоритма(greedy) для поиска пути в графе и алгоритма А\*(a\_star) для поиска кратчайшего пути в графе.

**Описание жадного алгоритма**

Работа алгоритма основана на принятии локально оптимальных решений при выборе вершины для перехода. Все вершины и ребра записываются в матрицу смежности (вектор векторов float). Затем, пока текущая вершина не равна конечной вершине, в каждой строчке находится минимальное ребро и соседняя вершина, в которую ведет это ребро записывается в вектор выходных данных. Таким образом, на каждом этапе работы алгоритма будет выбрана вершина, смежная с последней обработанной и имеющая наименьшую стоимость перехода, что обеспечивает жадность.

**Описание алгоритма А\***

Работа алгоритма основана на поиске по первому наилучшему совпадению, т.е. расширению наиболее перспективных узлов (в данном случае имеющих минимальную стоимость полного пути), выбираемых в соответствии с эвристической функцией (стоимость перехода (вес ребра) + значение функции эвристической оценки расстояния от рассматриваемой вершины до конечной (в данном алгоритме используется разница между кодами ASCII вершин)). Начиная с начальной вершины, из которой необходимо найти путь в конечную, происходит вставка всех смежных вершин в очередь с приоритетами. В качестве приоритета используется значение эвристической функции. Таким образом, за счёт сортировки приоритетов от меньшего к большому, на каждом этапе работы алгоритма будет выбрана вершина, имеющая наименьшую цену полного пути, то есть являющуюся наиболее перспективной, что и обеспечивает поиск оптимального решения.

**Описание функций и структур данных.**

**lab2\_1.cpp:**

Структуры данных:

vector<vector <float>> matrix - матрица смежности;

char f\_vertex - начальная вершина;

char l\_vertex - конечная вершина;

float weight – вес;

vector <char> result - массив выходных данных;

Функции:

void PrintMatrix(vector<vector<float>>& matrix) – печать матрицы;

int GreedyAlgorithm(vector<vector<float>>& matrix, vector <char>& result, char& f\_vertex, char& l\_vertex) – жадный алгоритм;

float MaxElement(vector<vector<float>>& matrix) – нахождение максимального элемента;

**lab2\_2.cpp:**

Структуры данных и методы:

struct Info { - информация о имени вершины в которую есть путь из данной вершины и её вес;

Info(char name, float weight): name(name), weight(weight) {};

char name;

float weight;

};

class Vertex { - вершина с прилегающим к ней списком смежных вершин и их путь;

public:

Vertex(char name) : name(name){};

char get\_name() const {

return name;

}

list<Info> list\_edge - список всевозможных путей, в другие вершины + вес;

private:

char name;

};

class Info\_prior { - класс приоритетов для приоритетной очереди;

public:

Info\_prior(Vertex vert, float prior): vert(vert), prior(prior){};

Vertex vert;

float prior;

};

class Adjacency\_list { - список смежности;

public:

Adjacency\_list() { }

Vertex& at(char a) – взятие по индексу;

float heuristic(Info temp, Vertex last) - эвристической функция;

void search\_result (Vertex first, Vertex last) – алгоритм а\*;

void print\_result(Vertex first, Vertex last, map<char, char>& prevV\_nextV) – вывод выходных данных;

priority\_queue<Info\_prior> queue - приоритетная очередь

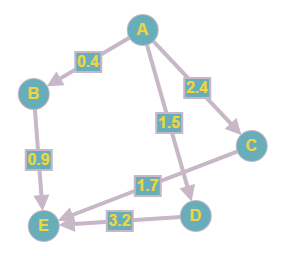
vector<Vertex> list\_vertex – вектор вершин;

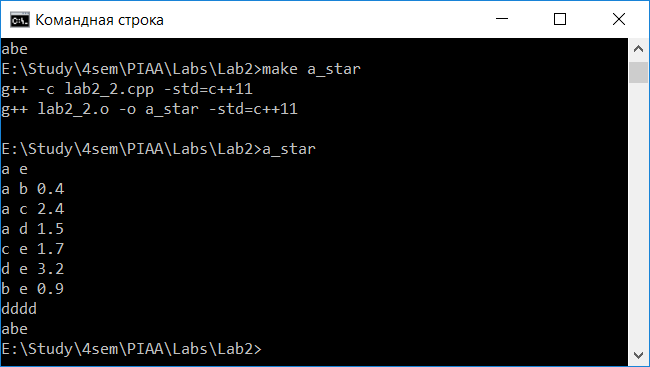
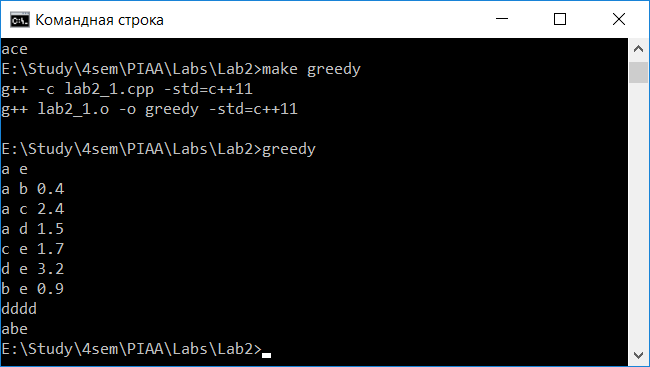
map <char, char> prevV\_nextV – контейнер, в котором хранятся пары из вершин: из которой ушли и в которую пришли;

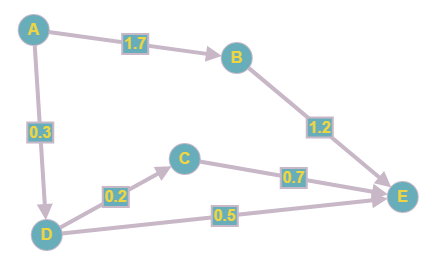
map <char, float> vertex\_to\_weight – контейнер, в котором хранятся пары из вершины и стоимости пути до нее из начальной вершины;

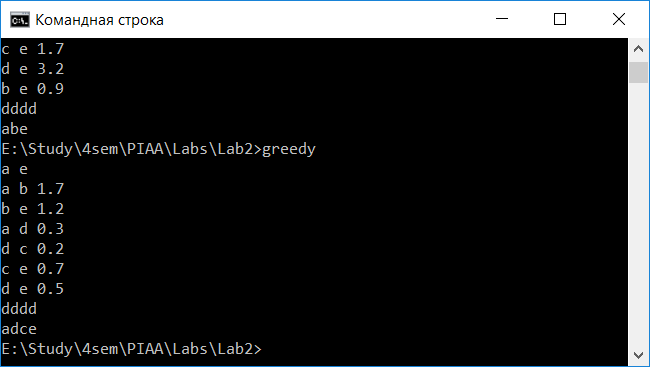
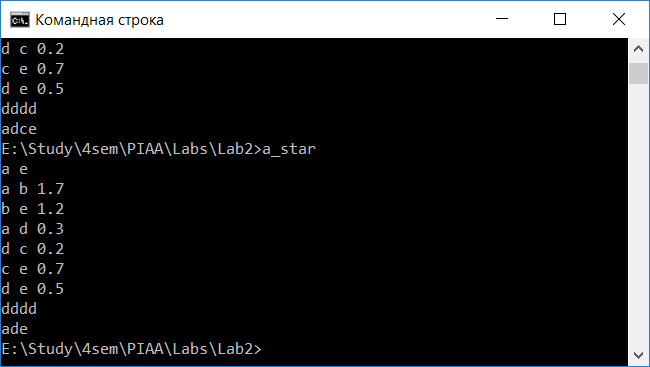
};

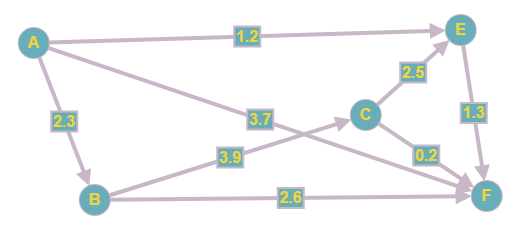
**Тестирование**



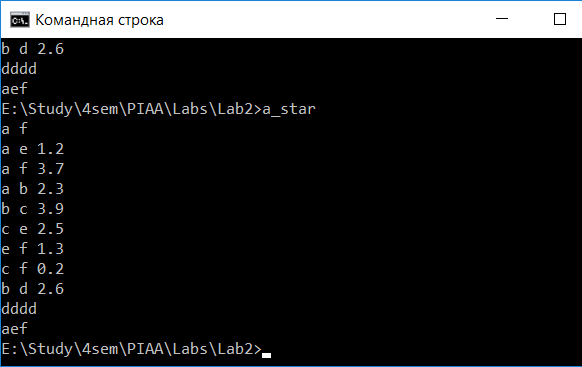
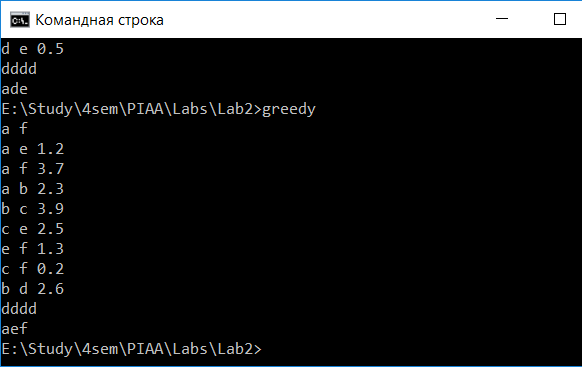
Жадный алгоритм А\* алгоритм 



Жадный алгоритм А\* алгоритм 



Жадный алгоритм А\* алгоритм



**Вывод**

В процессе выполнения лабораторной работы был реализован жадный алгоритм для поиска пути в графе и алгоритм А\* для поиска кратчайшего пути в графе.