**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 |  | Герасимова Д.В. |
| Преподаватель |  | Филатов А.Ю. |

Санкт-Петербург

2018

**Цель работы**

Изучение жадного алгоритма для поиска пути в графе и алгоритма А\* для поиска кратчайшего пути в графе.

**Описание жадного алгоритма**

Работа алгоритма основана на принятии локально оптимальных решений при выборе вершины для перехода. Начиная с начальной вершины, из которой необходимо найти путь в конечную, происходит вставка в очередь с приоритетами всех смежных вершин в порядке возрастания стоимости перехода (в данном случае эта стоимость определяется весом ребра, соединяющего две вершины). В качестве приоритета используется порядковый номер текущей вершины, определяемый на этапе выполнения. Таким образом, за счёт сортировки приоритетов от большего к меньшему, 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 методов и полей классов**

class Node – вершина графа

Поля класса:

* char name – имя вершины
* bool visited – была ли посещена вершина
* list<pair<Node\*, double>> neighbours – список смежных вершин и весов ребер до них

Методы класса:

* explicit Node(char name) : name(name), visited(false) – конструктор
* char getName() const – имя вершины
* bool isVisited() const – была ли вершина посещена
* void visit() – отметить вершину как посещенную

class Graph - граф

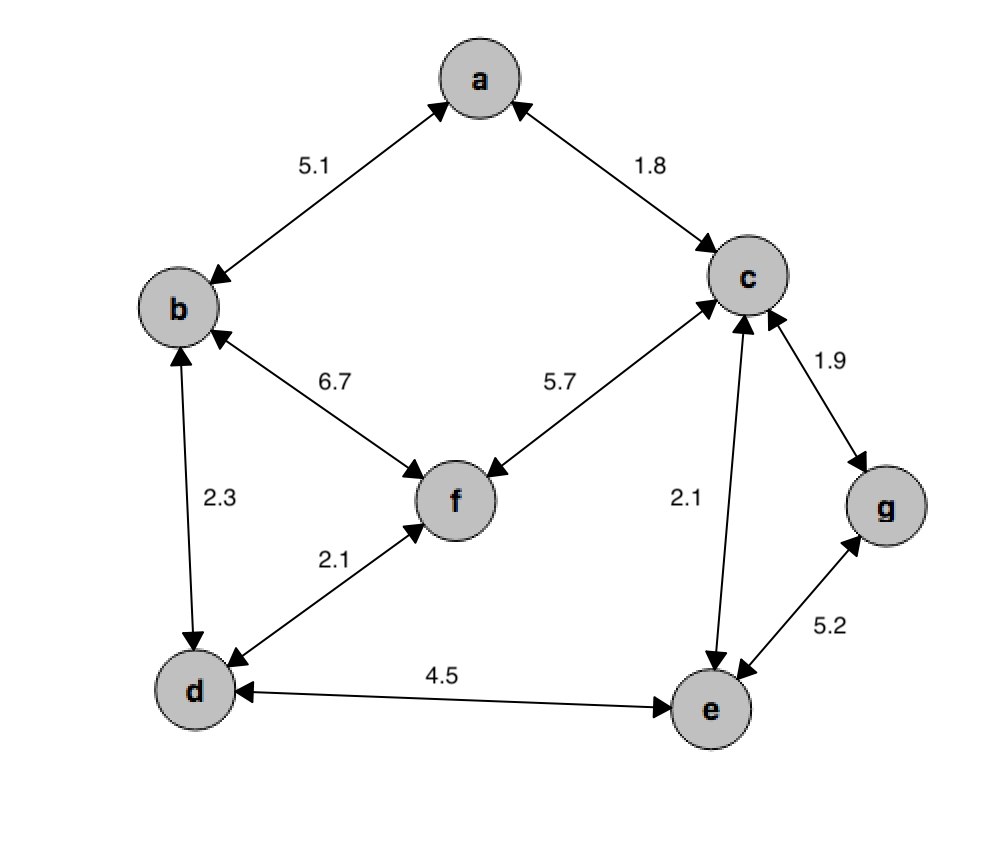
Поля класса:

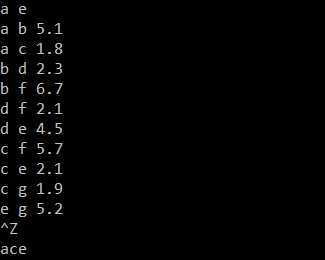
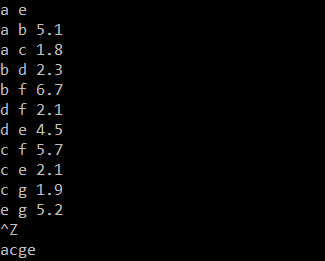
* list<Node\*> nodes – список вершин графа
* map<char, char> came\_from – ассоциативный массив переходов из вершину в вершину
* map<char, double> cost\_to\_node – ассоциативный массив стоимости перехода в вершину
* priority\_queue<pair<Node\*, double>, vector<pair<Node\*, double>>, Comparator> next\_nodes – очередь вершин с приоритетами, которые необходимо посетить
* Node\* point\_end – вершина, в которую необходимо найти путь

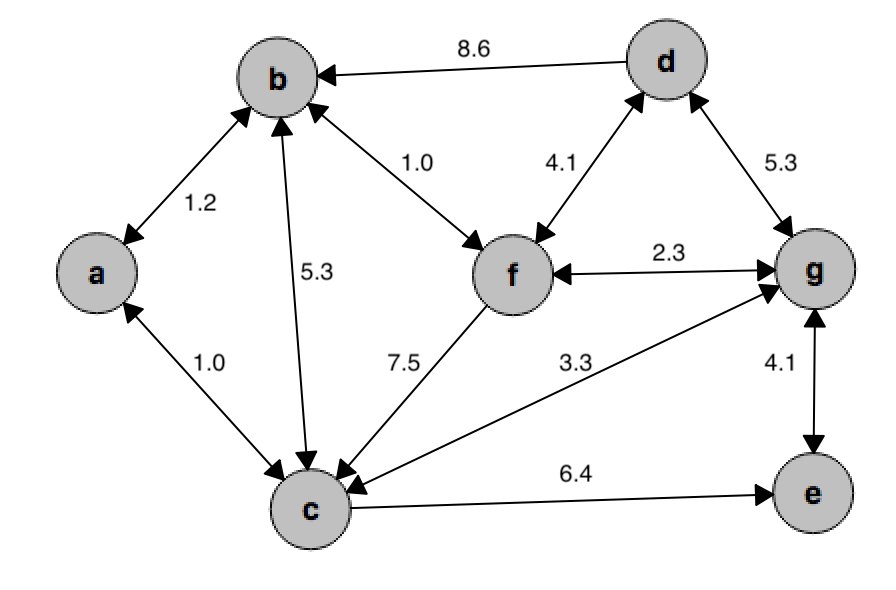
Методы класса:

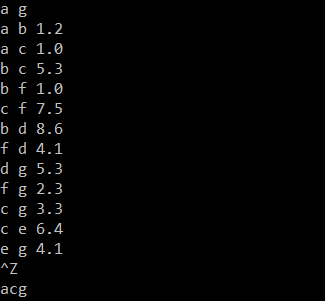
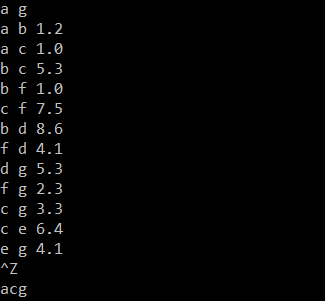
* Node\* getNode(char name) – возврат указателя на объект вершины по ее имени name
* void visitNode(Node\* node) – посещение вершины node
* void findPath(Node\* point\_from, Node\* point\_to) – поиск пути в графе из point\_from в point\_to
* double move\_cost(Node\* current, Node\* next) – расчет стоимости перехода из текущей вершины current в вершину next
* int heuristic(Node\* next, Node\* end) – эвристическая функция оценки расстояния от вершины next до конечной вершины end

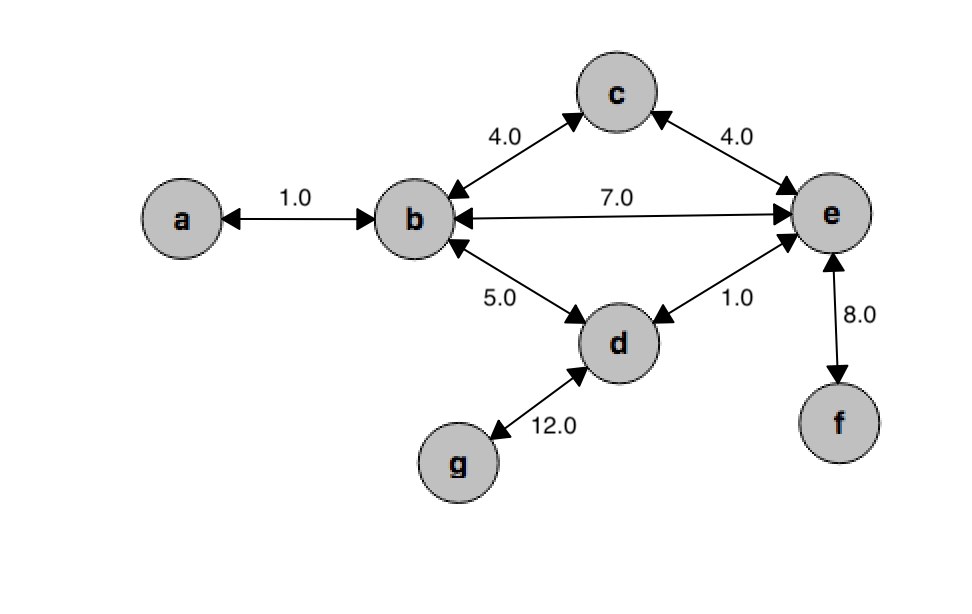
**Тестирование**

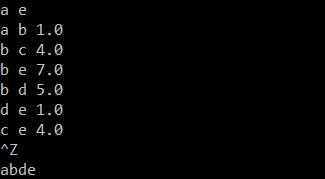
****

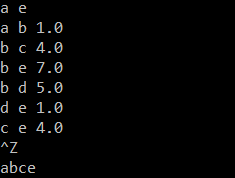
****Жадный алгоритм А\* алгоритм****

****

****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.