**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 |  | Токун М.С. |
| Преподаватель |  | Филатов А. Ю. |

Санкт-Петербург

2018

**Цель работы.**

Изучение жадного алгоритма и алгоритма AStar для поиска кратчайших путей в графе

**Описание алгоритмов.**

*Жадный алгоритм* - это алгоритм, который на каждом шагу делает локально наилучший выбор в надежде, что итоговое решение будет оптимальным. Поэтому изначально в функцию нахождения путей(*findWay*) передается список инцидентности для заданного графа, в котором для каждой вершины все доступные из нее вершины отсортированы в порядке возрастания весов соответствующих ребер. Функция каждый раз берет первую вершину в списке инцидентности для переданной в функцию вершины (путь в эту вершину будет являться минимально возможным) и переходит в нее. Алгоритм выполняется до тех пор, пока не будет найдена требуемая вершина или пока не будут проверены все вершины.

*Алгоритм AStar* - аналог жадного алгоритма, однако на каждом шаге выбирает вершину с максимальным приоритетом (приоритет складывается из текущего пути до этой вершины и результата эвристической функции, которая возвращает оцененное расстояние до цели). Для работы алгоритма создана очередь с приоритетом, в которую в ходе работы алгоритма добавляются вершины и их приоритет. Алгоритм начинает свою работу с того, что заносит в очередь стартовую вершину и ее приоритет 0. На каждом последующем шаге алгоритм рассматривает все вершины, в которые можно прийти из вершины с наибольшим приоритетом, рассчитывает их приоритет и добавляет в очередь, после чего данная вершина удаляется. Цикл повторяется до тех пор, пока не будет найдена конечная вершина, либо очередь не окажется пустой.

**Описание функций.**

1. Жадный алгоритм

**bool getTop(vector<char> tops, char top) –** проверяет, занесена ли вершина в вектор.

**int getIndex(vector<char> tops, char get\_start)** – принимает на вход вершину и находит ее место в векторе.

**void printStack(stack <char> stack)** – принимает на вход стек, и выводит его содержимое.

**void findWay(vector<char> tops, vector<vector<Path> > paths, int startIndex, char end, stack <char> stack) -** функция принимает на вход вектор вершин и вектор ребер, стартовую и конечную вершины, позицию стартовой вершины в веторе вершин, и стек. Функция берет первый элемент в списке инцидентности (vector<vector<Path> > paths), проверяет не является ли он конечным. Если является, то функция выводит путь на экран и завершается. Если элемент не является конечным, то рекурсивно вызывается функция поиска пути уже из этой вершины.

1. Алгоритм AStar

**bool getTop(vector<char> tops, char top) –** проверяет, занесена ли вершина в вектор.

**int getIndex(vector<char> tops, char get\_start)** – принимает на вход вершину и находит ее место в векторе.

**void printWay(vector<char> from, vector<char> to, char start, char end)** – печатает найденный путь.

**void findWay(vector<char> tops, vector<vector<Path> > paths, char start, char end)** *-* функция начинает свою работы с того, что заносит в очередь с приоритетом стартовую вершину и ее приоритет 0. Затем до тех пор, пока очередь не пуста, функция берет первый элемент очереди, находит приоритеты всех доступных вершин и заносит их в очередь. Функция продолжает свою работы пока очередь не пуста, либо не найден путь из начальной в конечную вершину.

**Тестирование.**

|  |  |  |
| --- | --- | --- |
| Данные на вход | Greedy | AStar |
| a e a b 3.0 b c 1.0 c d 1.0 a d 5.0  d e 1.0 |  |  |

**Вывод.**

На основе жадного алгоритма и алгоритма AStar были разработаны программы для поиска путей в графе.