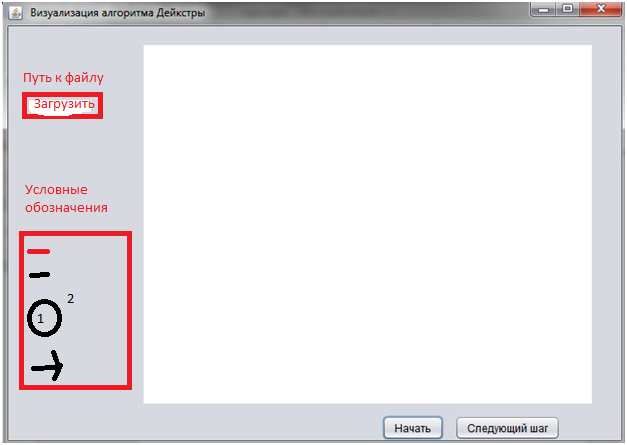
**Спецификация программы:**

В работе будет реализована визуализация алгоритма Дейкстры, с ее помощью пользователь сможет детально увидеть работу поиска кратчайших путей до всех точек, проследить за ходом выбора текущей точки и разобраться в логике алоритма. Визуализация подразумевает наличие графического интерфейса, в котором содержится текущее состояние(изображение) направленного взвешенного графа.

**Окно для визуализации:**



**UML-диаграмма:**





**Структура проекта:**

Класс GraphManager предназначен для создания графа, путем чтения из файла информации, также в нем будут храниться данные о графе.

Класс DeikstraWorker предназначен для расчета минимальных путей до каждой вершины. Данный класс будет реализован с использованием паттерна проектирования Singleton(lazyinitialization).С помощью класса GraphManager текущий класс получает доступ к информации о заданном графе. А с помощью функции startDeikstra рассчитает минимальные стоимости до остальных вершин.

**Информация о графе в классе содержит:**

ArrayList<ArrayList<Pair<Integer, Integer>>> table; таблица стоимостей ребер из каждой вершины

ArrayList<Character> isVisit; список посещенности вершин

ArrayList<Integer> distance; список стоимостей до вершин

**Входные данные:**

Формат чтения информации из файла:

Input.txt

4 //количество вершин

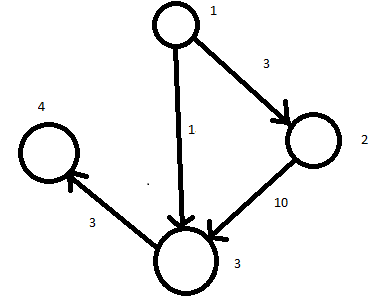
1 2 3 // из вершины 1 в вершину 2, направленное ребро со стоимостью 3

1 3 1

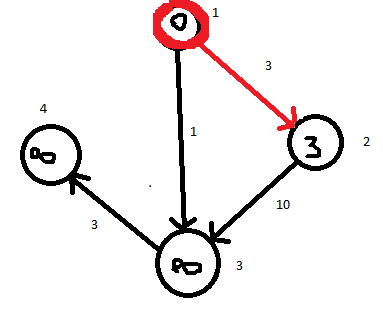
2 3 10

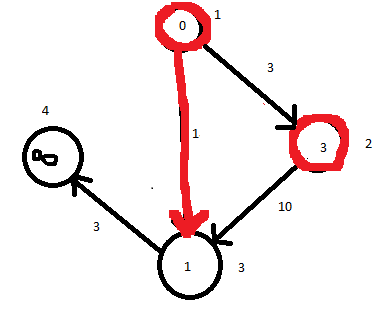
3 4 3

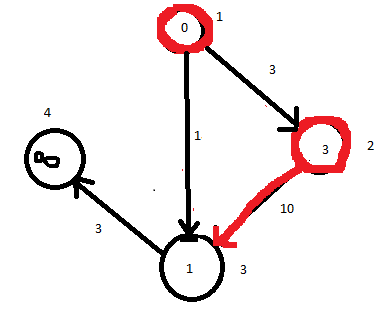
В результате чтения данных из файла получится такой граф.

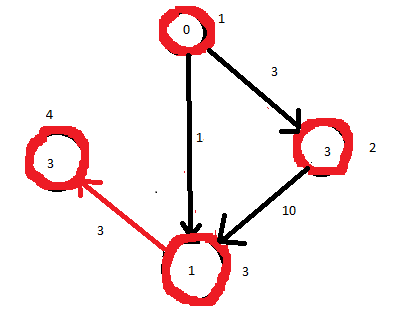


При нажатии клавиши Следующий шаг осуществляется переход к следующему состоянию графа, обновляются дополнительные пометки вершины. Вершины, для которых рассмотрены все входящие ребра окрашиваются в красный цвет. После завершения последнего шага, пометки вершин будут обозначать минимальную стоимость из начальной вершины до данной вершины.









**Выходные данные:**

Затем будет выдано сообщение, что работа алгоритма закончена.

Пример:

Алгоритм завершил работу!

Стоимость из вершины 1 в 2 = 3

Стоимость из вершины 1 в 3 = 1

Стоимость из вершины 1 в 4 = 4