Software Requirements Specification

1 Content of the product

The compressed directory contains the following files:

- .java files for all classes and tests
- The documentation of all files as .html
- Two UML class diagrams (one with variables and functions, one with classnames only) as .pdf files
- Documentation of the tests as .pdf file
- Background pictures for the GUI as .jpg files
- Some .txt files which can be selected by the user (some are also needed for tests)

2 Definitions, acronyms and abbreviations

Since one functional requirement demands a proper graph layout via one force directed layout algorithm, we will now define an aesthetic layout for graphs. Such a layout has three main properties. First, an even vertex distribution is requested. As second property, the graph should be drawn as symmetrically (in regard to the representation of the graph) as possible. The last requested property is that all edges should be nearly of equal length. If a graph layout does fulfill more or rather better than an other layout, so is the first one more aesthetically pleasing than the last one.
Above all cliques are separated in an aesthetic layout (if present).

3 References

For the force directed algorithm we used the source "Force-Directed Drawing Algorithms" by Stephen G. Kobourov. More precisely, we adopted the core idea of the algorithms of Eades from p.385 and Fruchterman and Reingold p.387 figure 12.2.

4 Product perspective

The product is supposed to be open source. It is Java based and developed under the usage of the Java development kit jdk 1.8.0_171 and the Java runtime environment jre 1.8.0_171, both use the current Java 8 release. For the tests we chose JUnit 5 as the testing framework, which requires Java 8 as runtime environment. The software is developed for graph visualization and manipulation. Furthermore, the Dijkstra algorithm can be executed on the graph to find a shortest path between two given vertices (if such a path exists). The process of finding a path shall be displayed step-wisely. There shall also be the option to skip right to the last step of the algorithm after the fist step is done.

The software contains following functionalities:

- Add vertices manually (not available after loading a graph)
- Remove a chosen vertex manually
- Add an edge between two chosen vertices. The weight of the edge can be chosen as a non-negative integer
- Remove an existing edge between two vertices
- Show/hide edgeweights
- Move a vertex (and also its incident edges)
- Find a shortest path between two vertices via Dijkstra algorithm
- Load a graph from a .txt file and adjust the layout via one force directed layout algorithm to get a more aesthetic layout
- Change the background picture
- Reset the draw panel (delete the drawn graph)

5 User characteristics

This program is made for users with interests in graphs.

6 General constraints

- The edge-weights are supposed to be non-negative, since the Dijkstra algorithm does not operate as intended on graphs that contain circles with negative weights. Therefore is neither the input of letters and symbols nor the input of negative integers possible at the setting of edge-weights.
- The amount of vertices in a .txt file, which can be drawn nicely, is limited by the screensize. Since every vertex is drawn as a circle with a radius of 10 pixels they start to overlap if too many vertices are drawn. Isolated vertices (those without any edges) start to pile up on the boundary of the force directed layout placement (which is the area 100 pixels from the maximal width and height of the used screen), since they only get repelled.
- The amount of manually drawable vertices is also only restricted by the screensize. When adding vertices manually, there is an invisible circle with a radius of 25 pixel, which inhibits the drawing of any further vertices to prevent overlapping. Hence it is possible that at some point there is no further vertex drawable, since the whole screen (except the draw-inhibtion-zone around every vertex) is already filled with vertices.
- The software only supports directed graphs. The software does not support multigraphs (in which multiple edges between two vertices can occur).

7 User requirements

Functional requirements
The software shall fulfill the following functional conditions:

- When left-clicking on the DrawPanel (this area is marked by the background picture), a new Vertex shall be displayed.
- The user shall be able to move vertices in the DrawPanel. This can be done with two clicks in the DrawPanel: After the first click, the selected vertex is colored blue. The second click specifies where the user wants the blue vertex to be. After the second click the blue vertex shall be moved to the specified position.
- There shall be the option to remove vertices by clicking on the vertex the user wants to be removed. Afterwards the selected vertex shall be marked and a frame shall open which asks the user if he is sure he wants to remove that vertex.
- There shall be the option to add an edge to the graph via clicking twice on the drawPanel: with the first click the start vertex is selected and with the second click the end vertex is selected. Ater the second click a frame shall open which asks the user to input a weight for the edge. It shall only be possible to input a non-negative integer for the weight.
- It shall be possible to choose whether the edge-weights are displayed or not.(Via clicking on the menu "Edges" there shall be the option"show weights".)
- The user shall be able to remove edges by clicking first on the start vertex of the edge and clicking on the end vertex afterwards.
- It shall be possible to read in a graph as a .txt file. The format shall be as follows: The first line has to give the number of vertices

Nonfunctional requirements

8 System requirements

Functional requirements
The software shall fulfill the following functional conditions:

- When left-clicking on the DrawPanel (this area is marked by the background picture), a new Vertex shall be displayed. The added vertices shall have indices that start with 0. (Starting with vertex 1) every vertex added shall have the index of the last added vertex +1.
- If the user wants to remove a vertex when there are no vertices displayed, an error message shall be shown.
- If the user wants to add an edge when there are no vertices displayed, an error message shall be shown.
- It shall not be possible for the user to input letters or symbols as edge-weights. Negative edge-weights shall be forbidden as well.
- If the user wants to remove an edge when there are no vertices displayed, an error message shall be shown.
- When the user wants to move a vertex, the vertex shall be the one with the closest distance to the point the user selected.
- When the user wants to add an edge, the start and end vertex shall be the ones with the closest distance to the points the user selected.
- When the user wants to remove an edge, the start and end vertex shall be the ones with the closest distance to the points the user selected.
- The program shall be terminated by clicking on the red "x" in the upper right corner.

Nonfunctional requirements