ID1020 – Lab 5
Graph

1 Organisation

This lab is a programming lab that is to be *presented orally* in the lab session (redovisning).

All the questions must be answered in Canvas. For each programming task, you need to upload your java files into its designated answer box in Canvas.

2 Goals

This lab has the following goals:

- Reason about the connectivity of directed graphs
- Work with algorithms to find shortest paths in directed weighted graphs

3 Tasks

3.1 Connected Graph (C) – 50P

Preparation: There is a maven dependency of our graphs project in Canvas, add it into your project. Moreover, copy the Paths class from Canvas to your project to load the graph you are supposed to work on for this and the next task.

If you should encounter any errors, try to move your code into a directory hierarchy that doesn't contain any spaces or special characters. Java's URL and File classes don't get along well.

The data structures are similar (though not identical) to the ones presented in the lecture for the EdgeWeightedGraph. Additionally, vertices are labelled with a name in Vertex.label.

Task: Implement an algorithm to answer the following question:
Is g fully connected? If not, how many subgraphs can you find?
Be prepared to argue why your algorithm was a good choice as compared to alternatives.

Tip: While the implementation of g is technically a Digraph, you might notice that in fact all edges occur in both directions (with the same weight). Thus calculating strong connectivity on the digraph is equivalent to calculating connectivity on the undirected interpretation of the graph.

3.2 Shortest Path (C) – 50P

Use the same setup as in section 3.1 and implement another algorithm to answer the following question:
What is the shortest path between the vertices labelled with “Renyn” and “Parses”

a) ignoring edge-weights

b) taking edge-weights into account

If your algorithm from section 3.1 could solve this problem, you have to implement another one anyway for full points (otherwise -25P).