Lab 9

Adjacency List Implementation of a Graph and DFS

Download the archive `Lab9.zip` and extract the `SimpleGraph` application. The application reads an edge-list from a file and constructs a Graph. The Graph is stored as `AdjacencyMapGraph`. The application simply prints all the vertices followed by all the edges. The archives also contain two graph files which you can use as input for the application:

- `graph.txt`
- `graph2.txt`

Presentation by your TA

Your TA will explain the recursive implementation of Depth First Search.
See Graph Traversals slides (for 2015 slides, this can be found in page 37 and examples from page 38-40).

New Graph

Define a new graph in a textfile and print it with the application.

Study the Goodrich et al. Implementation

Study the `AdjacencyMapGraph` implementation of the `Graph` ADT by Goodrich et al. Keep in mind the structure discussed in class. Notice that the adjacency lists are implemented in the inner class `MyVertex` and the references from the edges in the inner class `MyEdge`. The `Graph` ADT does not define any traversal method. You are supposed to implement your own Depth-First Search inside the class SimpleGraph, in order to print its vertices in a depth-first search manne (for the next part). These classes are all part of the net.datastructures package which is contained in the zip archive for this lab.

Printing with Depth-First Search (simple Depth First Search, more complex version to be seen in lecture)

The program `SimpleGraph` has a routine `void printDFS( String vert )` that calls `DFS(graph,v)` which currently does nothing. Please implement the method `DFS(graph,v)` with a recursive implementation of Depth-first Search (page 37 of lecture notes in graph traversals). The routine should print the vertices as they are visited by the traversal: once when they are first visited and a second time when a vertex is finished. In order to do that, auxiliary methods `startVisit(v)` and `finishVisit(v)` are provided.