Data Structures - Lab Worksheet

Graphs

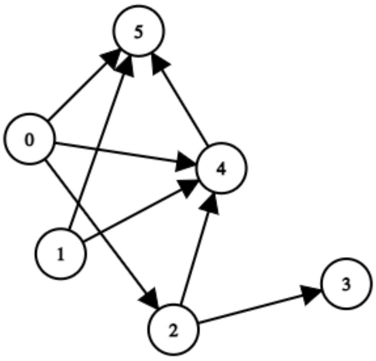
Start by downloading the provided coding canvas for graphs.

To use the visualization tools in `graph_output_tools.py`, you need to install three extra python libraries. First install Graphviz, then type the following command lines in your terminal:

```
$ pip install matplotlib
$ pip install networkx
$ pip install pygraphviz
```

You can browse the following example output of a full test run to see what it should look like once you've implemented all of the methods.

Question 1 - Adjacency Matrix



Give the adjacency matrix and the adjacency lists representation of the graph in the figure above.

Q: Which do you think is better suited for representing dense graphs? What about sparse ones?

Question 2 - Graph exploration

Implement the breadth-first search and the depth-first search functions in module `graph_algorithms`.

Implement the breadth-first search iteratively (method `bfs`) and the depth-first search recursively (function `recursiveDFS` initiates the first recursion that then repeats in `recursive_dfs`).

Question 3 - Connectivity Graph

Implement the function `computeConnectivity` in module `graph_algorithms`. It takes in an existing graph G as argument, and outputs another graph G' which is the transitive closure of G.

Q: Why do the mesh and the strongly-connected graph have identical transitive closures?

Question 4 - Minimal Spanning Tree

Implement the function `computeMinimumSpanningTree` in module `graph_algorithms`. It takes in an existing graph G and a vertex identifier V as arguments, and outputs another graph G' which is a minimal spanning tree of G with V as the root of the tree.

Q: Can two different minimal spanning trees be associated with the same graph?