Week 03 Tutorial

Advanced Graph Traversal

Aims

This exercise aims to get you to:

- Implement the topological sorting for directed graphs.
- Generate connected components by tracking unvisited vertices.

Exercise 1: Topological Sorting

1. Load the graph in Figure 1 via the class 'DirectedGraph' in tutorial_3.py.
2. Implement the function **topoSorting(G)** in tutorial_3.py which inputs the graph in Figure 1 and outputs one of the correct topological sorting sequences of vertices.
3. As the output result is not unique, here we list all the correct sequences for your reference:

- $b\ a\ d\ e\ c\ h\ g\ i\ f$
- $b\ a\ d\ e\ c\ h\ g\ f\ i$
- $b\ a\ e\ d\ c\ h\ g\ i\ f$
- $b\ a\ e\ d\ c\ h\ g\ f\ i$
- $b\ d\ a\ e\ c\ h\ g\ i\ f$
- $b\ d\ a\ e\ c\ h\ g\ f\ i$

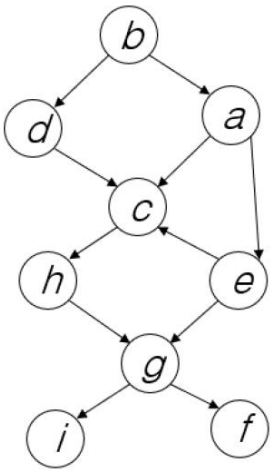

Figure 1

Exercise 2: Tracking Unvisited Vertices

1. Load the graph in Figure 2 via the class ‘UndirectedGraph’ in tutorial_3.py.

2. Implement the function **trackConnectComponents(G)** in tutorial_3.py which inputs the graph in Figure 2 and print the connected components. Example of conn_comp (which is not unique):

a	b	c	d	e	f	g	h	i
a	a	c	c	e	e	e	c	e

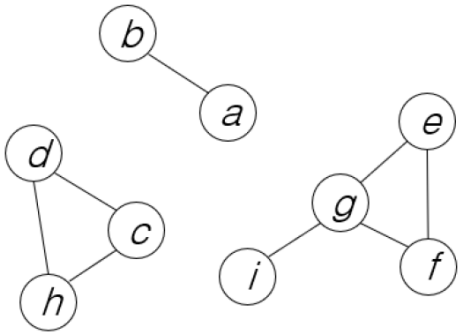

Figure 2