MINIA UNIVERSITY
FACULTY SCIENCE
Department of Computer Science
Data Structures Using Python

Exercises #10
Graph ADT

1. Give a Python implementation of the remove_vertex(v) method for our adjacency map implementation of Graph ADT, making sure your implementation works for both directed and undirected graphs.

2. Give a Python implementation of the remove_edge(e) method for our adjacency map implementation of Graph ADT, making sure your implementation works for both directed and undirected graphs.

3. Let *G* be an undirected graph whose vertices are the integers 1 through 8, and let the adjacent vertices of each vertex be given by the table below:

Vertex	Adjacent vertices
1	(2, 3, 4)
2	(1, 3, 4)
3	(1, 2, 4)
4	(1, 2, 3, 6)
5	(6, 7, 8)
6	(4, 5, 7)
7	(5, 6, 8)
8	(5, 7)

Assume that, in a traversal of *G*, the adjacent vertices of a given vertex are returned in the same order as they are listed in the table above.

a. Build *G*.

b. Give the sequence of vertices of *G* visited using a DFS traversal starting at vertex 1.

c. Give the sequence of vertices visited using a BFS traversal starting at vertex 1.

4. Bob loves foreign languages and wants to plan his course schedule for the following years. He is interested in the following nine language courses: LA15, LA16, LA22, LA31, LA32, LA126, LA127, LA141, and LA169.

The course prerequisites are:

LA15:	(none)
LA16:	LA15
LA22:	(none)
LA31:	LA15
LA32:	LA16, LA31
LA126:	LA22, LA32
LA127:	LA16
LA141:	LA22, LA16
LA169:	LA32

In what order can Bob take these courses, respecting the prerequisites?

5. Provide an implementation of the BFS algorithm that uses a FIFO queue, rather than a level-by-level formulation, to manage vertices that have been visited until the time when their neighbors are considered.

6. The solution to reporting a path from *u* to *v* using **construct_path()** method could be made more efficient if the DFS process ended as soon as *v* is discovered. Describe how to modify DFS code to implement this optimization.