ETEC 3401 **Lab #1: Graph class Part I**

Objectives

- Develop and implement a Graph class in Python
- Extend the capabilities of your Graph class with Numpy
- Use implementation to process data and solve graph problems

Submit a single program called *graph_YourInitials.py*. Complete the following tasks. You can use the Vector class and the Matrix class in *vector.py* and *matrix.py*.

Tasks

Create a Graph class

1. The constructor should allow the user to input a graph using either the adjacency list concept or an adjacency matrix.
2. The two properties **self.order** and **self.size** must be accessible to the user.
3. The two properties **self.adj_list** and **self.adj_matrix** must be accessible to the user.

Required methods

4. The user should be able to get and set weights for each vertex.
5. The user should be able to get and set weights for each edge.
6. **add_vertex** : The user should be able to add vertices.
7. **del_vertex** : The user should be able to remove vertices. Removal of a vertex removes all incident edges.
8. **add_edge** : The user should be able to add edges. An optional parameter should allow loops or multiple edges between two vertices, but the default should be to not allow loops or multiple edges.
9. **del_edge** : The user should be able to remove edges.
10. **is_directed** : return True if the instance of the Graph class is directed and False otherwise.
11. **is_connected** : return True if the instance of the Graph class is connected and False otherwise. If the graph is directed, then True means *strongly connected*.
12. **is_unilaterally_connected** : return True if the instance of the Graph class is directed and for every pair of vertices *v* and *w* there is a path from *v* to *w* or from *w* to *v*, but not necessarily both. Return False otherwise.
13. **is_weakly_connected** : return True if the instance of the Graph class is directed and the underlying undirected graph is connected. Return False otherwise.
14. **is_tree** : return True if the instance of the Graph class is a tree and False otherwise.
15. **components** : return the number of components in the instance of the Graph class.
16. **girth** : return the length of a shortest cycle in the instance of the Graph class.
17. **circumference** : return the length of a longest cycle in the instance of the Graph class.

Required function

18. Write a function to convert a Matrix to a 2D Numpy array.