Homework 4 Test Cases: Pt 2

In this document, the test cases for handling `csv` files are discussed. In order to carry out these test cases, you will need to implement constructor function(s) that can takes in files in csv format and represent it internally via adjacency lists.

At this stage, your graph class should have the functions as stated earlier to handle test cases for building undirected and directed graphs (Part 1).

Part 2 Test Cases for reading in undirected and directed graphs in `csv` format:

In these group of test cases, the experiments start with a given graph data stored in `csv` format. Since we have to handle both directed graphs and undirected graphs, we will denote the empty directed graph by D_e and the empty undirected graph by U_e.

Test Case 3 (For undirected graph)

Given a undirected graph U (with at least one vertex) stored in csv format, implement a constructor function which takes in the csv file of the graph and its type (either `directed` or `undirected`) as input, store the graph in adjacency list form.

Carry out the following steps in the order specified:

1. Use the constructor function to read in the graph from the a given `csv` file. We will state in advance if it represents a undirectd graph or not.
2. Print the dot file of the graph.
3. If the vertices are $\{1, \ldots, n\}$, insert n edges (randomly generated) to the graph U.
4. Show the graph (output the dot file of the graph) after all the edges have been added.

Test Case 4 (For directed graph)

Carry out the following steps in the order specified:

1. Use the constructor function to read in the graph from the a given `csv` file. We will state in advance if it represents a undirectd graph or not.
2. Print the dot file of the graph.
3. If the vertices are $\{1, \ldots, n\}$, insert n edges (randomly generated) to the graph U.
4. Show the graph (output the dot file of the graph) after all the edges have been added.
5. If the vertices are $\{1, \ldots, n\}$, remove n edges (randomly generated) to the graph U. Note that the edge may or may not be in the graph. Show the graph (output the dot file of the graph) after all the edges have been added.