Programming Assignment 3:

For this assignment you are to write a class and program for Graphs. Recall a graph is a data structure that consists of the following two components:

1. A finite set of vertices also called as nodes.
2. A finite set of ordered pair of the form (u, v) called as edge. The pair is ordered because (u, v) is not the same as (v, u) in case of a directed graph(di-graph). The pair of the form (u, v) indicates that there is an edge from vertex u to vertex v. Sometimes an edge has a third component, known as either a weight or a cost.

The most commonly used representations of a graph are:

1. Adjacency Matrix
2. Adjacency List

There are other representations also like, Incidence Matrix and Incidence List. The choice of the graph representation is up to you.

Required parts

For a C: An implemented graph class with the following methods:

1. **Cycle finding**, take a directed graph and determine if graph is acyclic. If there are cycles, determine what vertices are involved.
2. **Topological sort**, create an ordering ordering of vertices in a directed acyclic graph, such that if there is a path from v_i to v_j, then v_j appears after v_i in the ordering.
3. **Single-Source Shortest-Path unweighted graph**, given as input an unweighted graph, $G = (V, E)$, and a distinguished vertex, s, find the shortest path from s to every other vertex in G.
4. **Prim's algorithm**, given a weighted undirected graph, $G = (V, E)$, and an arbitrary vertex, s, find the minimum spanning tree.

For a B: Implemented the following methods:

1. **Allow the input file to be defined by the DOT language,** the abstract grammar for defining *Graphviz* nodes, edges, graphs, subgraphs, and clusters.
2. **Dijkstra's Algorithm**, given as input a weighted graph, $G = (V, E)$, and a distinguished vertex, s, find the shortest weighted path from s to every other vertex in G.
3. **Ford–Fulkerson algorithm,** given a weighted directed graph $G = (V, E)$, and two distinguished vertices, s and t, also called the *source* and *sink*. Find maximum amount of flow that the network would allow to flow from source to sink.
4. **Kruskal's algorithm**, given a weighted undirected graph, $G = (V, E)$ find a minimum spanning forest.

For an A: Implement the following

1. **Euler Circuits**, given an undirected graph $G = (V, E)$, implement Fleury's Algorithm to find the Euler path or circuit.
2. **Full Graphviz incorporation,** at a minimum allow the output graphs for the topological sort, Prim's, Ford-Fulkerson's, Kruskal's and Fleury's algorithm to be saved as DOT language files that can be inputs to Graphviz to create a graphic file of the graphs. A better solution is to use the C API to create the graphics output directly in your program.

Submission due 12/7

When you turn your solution in tell me how far your solution is implemented. Also include any test-files you have used. Be very clear in your documentation to describe how to run your program and how to use your test-files.
Notice, I have written these instructions in incremental steps. So if you can't figure out step B, don't go on to step A, as step A expects that everything up to and including to step B is working. As usual zip your solution and submit it to the dropbox.