Assignment-4

Implement directed graph ADT using adjacency list data structure.

Input file format

---

n m //#verices #edges

1 2 7 // Edge = (v1 v2 edge_weight)

2 4 8

1 3 5

……

Q1. Perform DFS traversal on the graph and classify all the edges. After DFS show the annotated graph where each vertex is labeled with the DFS start and end time and each edge is labeled with its edge type (tree/forward/back/cross). **[10 points]**

Q2. Find all strongly connected components using Tarjan's algorithm. **[10 points]**

Q3. Given a directed graph G = (V, E), create another graph G'= (V, E'), where E' is a subset of E, such that (a) G' has the same strongly connected components as G, (b) G' has the same component graph as G, and (c) E' is as small as possible. **[30 points]**

Q4. A directed graph G = (V, E) is semiconnected if, for all pairs of vertices u, v in V, we have either a path u ---> v or a path v ---> u. Design an efficient algorithm to determine whether or not G is semiconnected. Implement your algorithm and analyze its running time. **[30 points]**

Q5. Implement Dijkstra's shortest path algorithm (in O(Elog V) using a priority queue) to find a shortest-cost path between a source vertex s to any target vertex t. **[20 points]**

Note: You need to show the outputs visually by using graphviz.

Deadline: 31st Oct 2021.