National University of Computer & Emerging Sciences, Lahore
Department of Computer Science (Spring 2025)

CS2009 – Design and analysis of Alogrithms (BCS-6A)

Assignment #4

Course Instructor: Usama Hassan **Due Date: May 8, 2025**

Course TA: Talha Azhar **Total Marks:100**

Instructions:

1. Submit with your Roll Number (format: 22L-1234_A4.pdf) and hard form in class.
2. Do not submit multiple copies or empty files.
3. You can **do this assignment in pairs, or max group of 3** but 4 or more is not allowed.
4. It should be clear that your assignment will not get any credit if:
   - The assignment is submitted after the due date.
   - Assignment is plagiarized from any source. (Online/other students/etc.)

Do not panic and remember to take breaks from your device screen while attempting this assignment!

Assignment Overview: efficient algorithms

This assignment covers essential graph algorithms including DFS, BFS, Dijkstra’s Algorithm, Minimum Spanning Trees, and Topological Sorting. It is designed to strengthen problem-solving skills by applying traversal techniques, finding shortest paths, and optimizing graph structures.

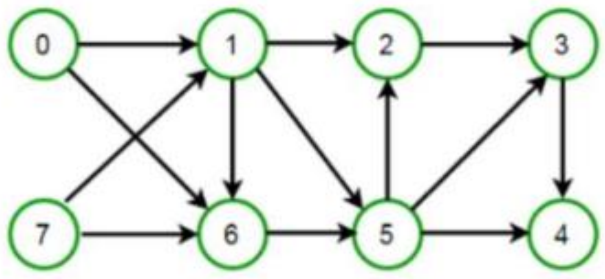
General Guidelines:

- Attempt the questions in sequence starting from Q#1 and clearly mention Q# and part#.
- Start each question from a new page.
- Attempt any 10 questions from the below.
- For Question 1, 2, attempt any one part from each question.
- For Question 6, attempt any 2 parts.
- Please complete one task questions before moving to next.
- Your Algorithms must be generic.
- Your assignment should be neat in such a way that it is readable.

Task : Graph Algorithms — [100 Marks]

Question 1:

(a) Given a directed graph in the form of adjacency list, find the total number of routes to reach the destination from a given source with exactly "m" edges. For example, consider the following graph:

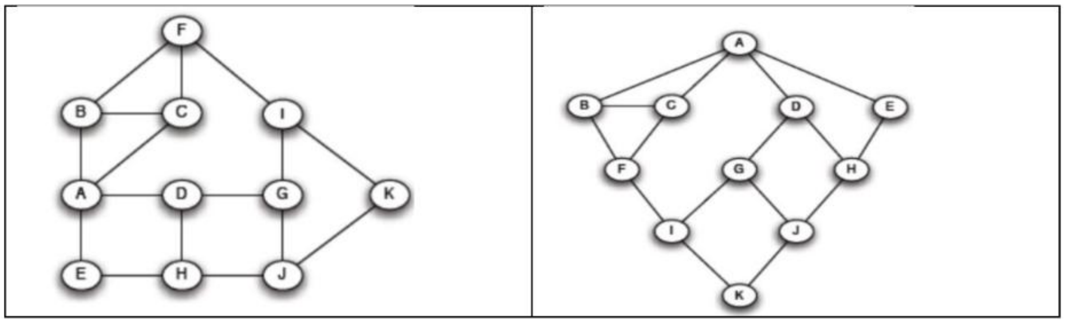


Let source = 0, destination = 3, number of edges 'm' = 4. The graph has 3 routes from source 0 to destination 3 with exactly 4 edges. The solution should return the total number of routes i.e., 3 in this case. Design an efficient algorithm for solving this problem and analyze its time complexity. Explain the main idea in plain English in maximum three to four lines.

Input Parameters: G, src, m, dest

(b) BFS can be used to solve the problem of single source shortest path for unweighted graph G(V, E). Also, BFS only finds one shortest path between "s" and "d", where "s" is the source vertex and "d" is any other vertex of the graph. But the shortest path may not be unique. Suppose we want to count the number of distinct shortest paths between "s" and for each "d" ε G.V. For example, in the graph given below if "A" is the source vertex then the number of distinct shortest paths for each pair are:

From A to B: 1	From A to C: 1	From A to D: 1	From A to E: 1	From A to F: 2
From A to G: 1	From A to H: 2	From A to I: 3	From A to J: 3	From A to K: 6



Design and explain a linear time algorithm to count the number of distinct shortest paths between “s” and all the other vertices in the graph. The graph is available in the form of an adjacency list.
Hint: you can use the tree structure of BFS to count the distinct shortest paths.

Question 2:

(a) Given a weighted undirected graph G(V, E) and its Minimum Spanning Tree T(V, E’). Both G(V,E)and T(V,E’) are represented in the form of adjacency list, where E’ represents the edges present in the MST. Assume that an edge (a, b) ∈ E’ has been deleted, devise an algorithm to efficiently update the MST after the deletion of (a, b). Assume that all edge weights are distinct, and the graph remains connected after the deletion of edge (a, b). Explain the main idea of your algorithm along with an explanation of its time complexity in plain English as well.

(b) Given a weighted undirected graph G(V, E) and its Minimum Spanning Tree T(V, E’). Both G(V,E) and T(V,E’) are represented in the form of adjacency list, where E’ represents the edges present in the MST. Assume that an edge (a, b) ∈ E has been relaxed. There is a strong possibility that MST needs to be updated because of this edge relaxation. Devise an algorithm to efficiently update the MST after the relaxation of (a, b). Your algorithm should have a time complexity better than running a full MST algorithm from scratch. Assume that all edge weights are distinct. Explain the main idea of your algorithm along with an explanation of its time complexity in plain English as well.

Question 3:

Suppose you are given a weighted directed graph G(V, E) with possibly negative edge weights.Suppose you also know that every path between all pairs of nodes in this graph has at most eight edges. We are interested to compute single source shortest paths in this graph given a source vertex “s”. Is it possible to Device an O(V+E) time algorithm for this problem. If “yes” then design and explain the algorithm.

Question 4:

The government of Pakistan is planning to build a new town near Lahore to provide houses to homeless people. Initially, it was planned that all the roads will be two way. However, the newly elected government changed the plan and mark all the roads as one way roads. The directions of the roads are marked in a way that that all places are reachable from the Town Hall. Due to this change, not all the places in the town are reachable from each other. In other words there are some pair of places that has no directed path between them. This means some new roads must be constructed to provide path between all those pair of places that are not connected in this one way road network. The government wants to spend minimum amount of money on this project. You are asked to analyze the current road network and add minimum possible new roads such that all pair of places has a path between them. Formulate this problem as a graph problem and give a linear time algorithm that can compute the new edges (roads) given the existing road network.

Question 5:

Can you use the DFS algorithm to compute the number of distinct paths between two given vertices ‘s’ and ‘t’? Two paths between ‘s’ and ‘t’ are considered distinct if they differ by 1 or more edges. If you think the answer is yes, then provide the pseudo-code for a DFS based algorithm that computes this number in $O(|V|+|E|)$. If you think the answer is no, draw a graph with at-least eight vertices in total, with two of its vertices labeled “s” and “t”, in which DFS will fail to compute the number of distinct paths between “s” and “t”.

Question 6:

(a) Give a counterexample to the conjecture that if a directed graph G contains a path from u to v, then any depth-first search must result in $v.d \leq u.f$.

(b) Give an example of a directed graph G = (V,E), a source vertex s € V, and a set of tree edges $E\pi \subseteq E$ such that for each vertex v € V, the unique simple path in the graph (V, Eπ) from s to v is a shortest path in G, yet the set of edges E, cannot be produced by running BFS on G, no matter how the vertices are ordered in each adjacency list.

(c) Give a counterexample to the conjecture that if a directed graph G contains a path from u to v, and if $u.d < v.d$ in a depth-first search of G, then v is a descendant of u in the depth-first forest produced.

Question 7:

Imagine you come across poor Abdul Bari, a student at Fast University, completely overwhelmed by the never-ending loop of assignments and deadlines. He's practically stuck in a negative cycle of stress and exhaustion, desperately seeking a way out.
You decide to play the role of the hero and introduce Abdul Bari to the Floyd-Warshall algorithm, explaining how it's like a magical map-maker for graphs. But then, with a twinkle in your eye, you hit him with the big question: "Hey Abdul Bari, now that you've got this superpower in your hands, how do you think we could use the Floyd-Warshall algorithm to uncover those sneaky negative-weight cycles hiding in Fast University's workload?"

Question 8:

So, check it out, the Floyd-Warshall algorithm is all like, "Hey, I need O(n^3) space because I'm fancy, computing dij^k for i, j, k = 1, 2, 3, ….. , n." But hold on, here comes this other procedure, it's like the laid-back version of Floyd-Warshall. It's just like, "Nah, we don't need all those superscripts, man." And boom, it does the job with just O(n^2) space.

Show that the following procedure, which simply drops all the superscripts, is correct, and thus only O(n2) space is required. FLOYD-WARSHALL(W)

1. n = W.rows
2. D = W
3. for k = 1 to n
4. for i = 1 to n
5. for j = 1 to n
6. dij = min (dij, dik + dkj)
7. Return D

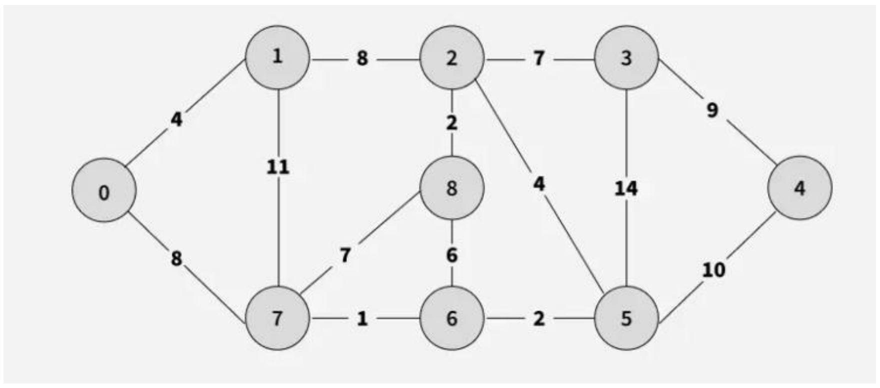
Question 9:
In a weighted, directed graph G = (V, E) with a source vertex s, and after initializing G using INITIALIZE-SINGLE-SOURCE(G, s), demonstrate that if a series of relaxation steps assigns a non-NIL value to s:PI, then G contains a negative-weight cycle.

Question 10:
Articulation Point: A vertex of a graph whose removal increases the number of connected components or converts the directed graph into a disconnected graph.
Bridge: An edge of a graph whose removal increases the number of connected components or converts the directed graph into a disconnected graph.

Approach to identify Bridges and Articulation point using DFS:
Delete a vertex or an edge of a graph and then apply DFS and keep the count of connected components. If the number of components in the graph are greater than 1 then it means that the deleted vertex/edge is an articulation point/bridge respectively.

Write an algorithm for using above approach.

Question 11:



Make minimum spanning tree of the above graph using any algorithm.

BONUS: Attempting all questions will land you bonus points!!!

Good Luck and Happy Coding :)