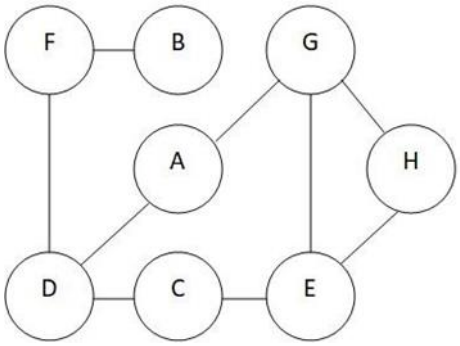
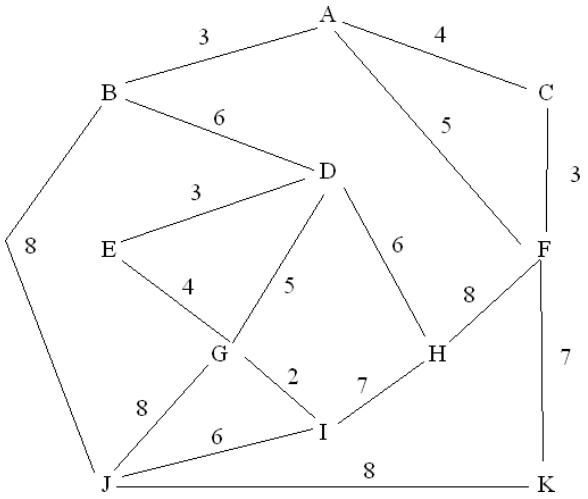
1. Run a **breadth first search** through the following graph starting with node A, and record the order that each node is visited. If a node has more than one neighbor, the next node to visit will be the one that comes first lexicographically.



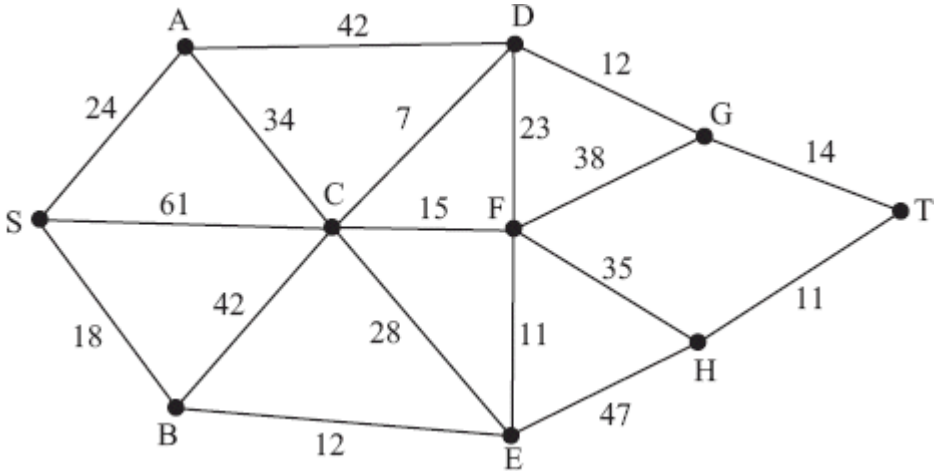
2. Answer True or False. Justify your answer.
   I. If $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$, then we have $(f(n))^2 = \Theta((g(n))^2)$
   II. $2^n + n^2 = O(2^n)$
   III. $2^n + n^2 = O(3^n)$
   IV. The topological sort of an arbitrary directed graph $G(V;E)$ can be computed in linear time.
   V. Kruskal's algorithm for minimum weight spanning trees is an example of a divide and conquer programming algorithm.
   VI. The shortest path between two vertices is unique if all edge weights are distinct.
   VII. An arbitrary graph with $G(V;E)$, with $|E| = |V|$ edges is a tree.
   VIII. A directed graph is strongly connected if and only if a DFS started from any vertex will visit every vertex in the graph without needing to be restarted.

3. 

Consider the graph in Figure 1. Unless otherwise indicated, always visit adjacent nodes in alphabetical order.
   I. Provide the **DFS tree** starting at node A.
   II. Provide the **BFS tree** starting at node A.
   III. Use **Kruskal's algorithm** to derive the MST. (Hint: Sort the edges and then work through them.)
   IV. Use **Prim's algorithm** to derive the MST starting at node A. (Hint: Sort the edges and then work through them.)

4. Can Dijkstra's algorithm be used to find the shortest path in a directed acyclic graph (DAG)? Why or why not?

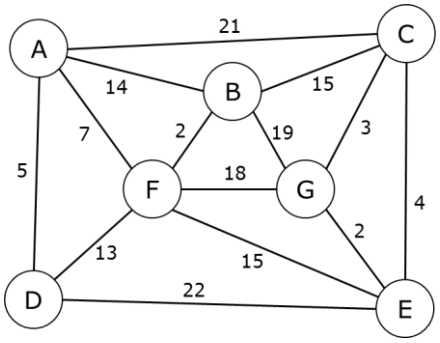
5.



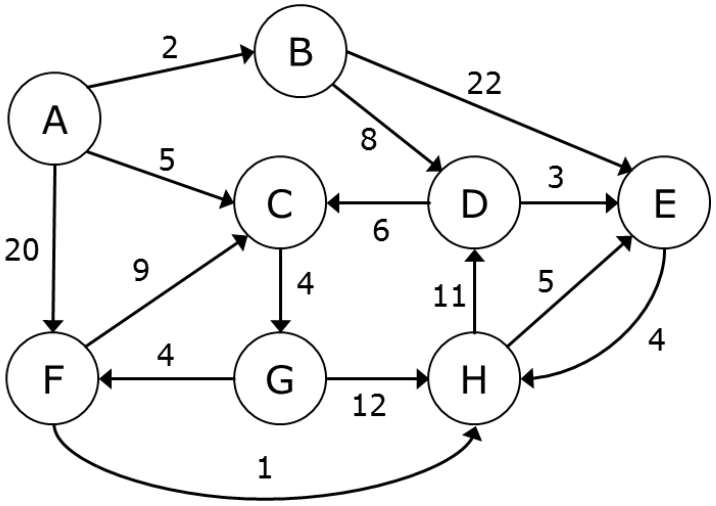
The diagram above shows a network of cycle tracks within a national park. The number on each arc represents the time taken, in minutes, to cycle along the corresponding track. Use **Dijkstra's algorithm** to find the shortest route from S to T. State your quickest route and the time it takes. State your shortest route and its length.

6. Consider the following graphs to construct minimum spanning tree by using both **Prim's** & **Kruskal's** algorithm:

a.



b.



7. Answer the following question on NP hard and NP complete:

I. Define NP hard and NP complete. Discuss about NP, NP-hard and NP-complete problems.

II. Is P=NP?

III. If $P = NP$ then Shortest-Path is NP-complete. Do you agree with this statement? Justify your answer.

IV. True/False (with Justification):

a) The problem of determining whether there exists a cycle in an undirected graph is in P.
b) The problem of determining whether there exists a cycle in an undirected graph is in NP.
c) If a problem A is NP-Complete, there exists a non-deterministic polynomial time algorithm to solve A.
d) Let X be a problem that belongs to the class NP. If X can be solved deterministically in polynomial time, then P = NP.

8. Can the master method be applied to the recurrence $T(n) = 4T(n/2) + n^2 \lg n$? Why or why not? Give an asymptotic upper bound for this recurrence.

9. Solve the following recurrences by using the master method:

I. $T(n) = 2T(n/4) + 7$.
II. $T(n) = 7T(n/2) + n^2$
III. $T(n) = 3T(n/9) + \text{root}(n)$
IV. $T(n) = T(9n/10) + n$
V. $T(n) = 2T(n/4) + n \log n$.

10. Answer the following question on greedy algorithm and dynamic programming:

a) How does the divide-and-conquer approach differ from other problem-solving approaches, such as dynamic programming or greedy algorithms?
b) Do you think greedy algorithm always provide a optimal solution? Explain with example.
c) How do you analyze the efficiency or time complexity of a greedy algorithm?
d) What is dynamic programming? What characteristics should dynamic programming have?
e) Determine an LCS of $\langle 1, 0, 0, 1, 0, 1, 0, 1 \rangle$ and $\langle 0, 1, 0, 1, 1, 0, 1, 1, 0 \rangle$.
f) How will you calculateFibonacci sequence using dynamic programming? Write pseudo code or code to find out Fibonacci sequence using dynamic programming with O(n)-time.
g) Differentiate top-down dynamic programming (memoization), and bottom-up dynamic programming.
h) What are sub problems and how do we re-use them?