UNIT V – GRAPH ADT – QUESTION BANK

Part A – 2 Marks Questions

1. Define Graph ADT.
2. What are the different ways to represent a graph?
3. Define adjacency matrix and adjacency list.
4. What is a Directed Acyclic Graph (DAG)?
5. Define topological sorting.
6. What is the difference between DFS and BFS?
7. Mention the applications of graph traversal.
8. What is a minimum spanning tree?
9. Define dynamic programming.
10. State Dijkstra's algorithm in brief.
11. Define greedy algorithm with an example.
12. What is the time complexity of Prim's algorithm?
13. Define NP and NP-complete problems.
14. What is the significance of the P vs NP problem?
15. Mention two limitations of algorithmic power.

Part B – 13 Marks Questions

1. Explain the adjacency matrix and adjacency list representations of a graph with examples.
2. Write and explain DFS and BFS traversal algorithms with suitable examples.
3. What is a DAG? Explain its properties and applications.
4. Describe the algorithm for topological sorting with a suitable example.
5. Explain Dijkstra's algorithm for finding the shortest path. Illustrate with a graph.
6. Describe the Bellman-Ford algorithm and compare it with Dijkstra's algorithm.
7. Explain Kruskal's and Prim's algorithms for finding MST. Compare their performance.
8. Discuss the characteristics and applications of greedy algorithms.
9. Explain the difference between greedy and dynamic programming approaches.
10. Describe the classes P, NP, and NP-complete. Give examples.
11. Analyze why certain problems are considered intractable. Give examples.
12. Solve a shortest path problem using dynamic programming approach.
13. Illustrate a minimum spanning tree construction using Prim's algorithm step-by-step.

Part C – 15 Marks Case Study-Based Questions

1. **Case Study – Traffic Navigation System**:
   A smart city uses sensors and graphs to monitor traffic and suggest fastest routes. Model this using graph representations and implement shortest path algorithm. Compare Dijkstra and Bellman-Ford results.

2. **Case Study – Course Prerequisite Scheduling**:
   A university course system must schedule subjects based on prerequisites. Represent this as a DAG and perform topological sorting to find valid subject orders.
3. **Case Study – Network Cabling Optimization**:
   An IT park needs to lay cables to connect all buildings with minimum cost. Formulate the problem using MST and solve using Kruskal's algorithm.
4. **Case Study – Logistics Optimization Using Graphs**:
   A delivery company needs to find the most efficient delivery routes. Use dynamic programming for shortest paths and greedy algorithms for cost efficiency.
5. **Case Study – NP-Complete Problem Detection in Project Scheduling**:
   A complex project plan has dependencies making scheduling difficult. Explain how this relates to NP-completeness and what approximate or heuristic methods can be used.
6. **Case Study – Resource Allocation in Distributed Systems**:
   Resources must be assigned to multiple dependent jobs with constraints. Model using graphs and analyze using topological sort and shortest path techniques.