**Beginner Level:**

1. Implement a graph using adjacency matrix
2. Implement a graph using adjacency list
3. Add and remove vertices in a graph
4. Add and remove edges in a graph
5. Check if a path exists between two vertices (using DFS)
6. Check if a path exists between two vertices (using BFS)
7. Count the number of vertices and edges in a graph
8. Find the degree of a vertex
9. Print all the neighbors of a vertex
10. Check if a graph is connected
11. Find the number of connected components
12. Perform DFS traversal of a graph
13. Perform BFS traversal of a graph
14. Detect a cycle in an undirected graph (using DFS)
15. Detect a cycle in an undirected graph (using BFS)
16. Convert adjacency matrix to adjacency list
17. Convert adjacency list to adjacency matrix
18. Check if two graphs are isomorphic
19. Find all nodes reachable from a given node (using BFS)
20. Find all nodes reachable from a given node (using DFS)

**Medium Level:**

1. Topological sorting (Kahn's algorithm)
2. Topological sorting (DFS based)
3. Find strongly connected components (Kosaraju's algorithm)
4. Find strongly connected components (Tarjan's algorithm)
5. Find the shortest path in an unweighted graph (BFS)
6. Find the shortest path in a weighted graph (Dijkstra's algorithm)
7. Find the shortest path in a weighted graph (Bellman-Ford algorithm)
8. Find the shortest path between all pairs of vertices (Floyd-Warshall algorithm)
9. Minimum spanning tree (Kruskal's algorithm)
10. Minimum spanning tree (Prim's algorithm)
11. Check if a graph is bipartite (using BFS)
12. Check if a graph is bipartite (using DFS)
13. Find articulation points in a graph
14. Find bridges in a graph
15. Find the longest path in a directed acyclic graph
16. Find the maximum flow in a flow network (Ford-Fulkerson algorithm)
17. Find Eulerian path in a graph
18. Find Eulerian circuit in a graph
19. Find Hamiltonian path in a graph
20. Find Hamiltonian cycle in a graph

**Advanced Level:**

1. Find the shortest path in a graph with negative weights (Johnson's algorithm)
2. Planarity testing of a graph
3. Graph coloring problem (Greedy algorithm)
4. Graph coloring problem (Backtracking)
5. Implement the Hungarian algorithm for the assignment problem
6. Implement the Edmonds-Karp algorithm for maximum flow
7. Implement the Hopcroft-Karp algorithm for maximum matching in bipartite graphs
8. Implement the Bron-Kerbosch algorithm for finding all maximal cliques
9. Implement the Push-Relabel algorithm for maximum flow
10. Implement the Fleury's algorithm for finding Eulerian path/circuit