**1. Introduction to Graph Theory**

- Definition of graphs (nodes and edges)

- Types of graphs: Directed vs. undirected, weighted vs. unweighted

- Applications of graphs in real-world scenarios (mazes, navigation, networks)

**2. Basic Graph Traversal Algorithms**

- Breadth-First Search (BFS)

- How BFS works (FIFO queue)

- Use cases: Shortest path in unweighted graphs

- Depth-First Search (DFS)

- How DFS works (LIFO stack/recursion)

- Use cases: Exploring all paths, detecting cycles

**3. Shortest Path Algorithms**

- Dijkstra’s Algorithm

- How it works: Priority queue, edge relaxation

- Use cases: Finding shortest paths in weighted graphs

- A\* Algorithm

- Introduction to heuristics (e.g., Manhattan distance)

- How it improves upon Dijkstra’s by reducing search space

**4. Advanced Pathfinding Topics**

- Greedy Best-First Search (an alternative heuristic-based search)

- Bellman-Ford Algorithm (for graphs with negative edge weights)

- Floyd-Warshall Algorithm (for all-pairs shortest paths)

**5. Optimization Techniques**

- Heuristics in A\* for efficient search

- Time and space complexity analysis for BFS, DFS, Dijkstra’s, and A\*

**6. Applications of Pathfinding Algorithms**

- Game development (AI movement, dragon maze)

- Robotics and navigation (robot path planning)

- Networking (data routing)

- AI search problems (prison break, puzzles)

**7. Comparisons and Trade-offs**

- BFS vs DFS (advantages and disadvantages in different scenarios)

- Dijkstra’s vs A\* (efficiency in weighted graphs)

- Time complexity comparisons: BFS (O(V + E)), DFS (O(V + E)), Dijkstra’s (O(V²) or O(E + V log V))

**8. Real-world Example Problems**

- Dragon maze simulation (2D/3D grid with obstacles)

- Prison break simulation (maze escape with guards as obstacles)

**9. Visualization Techniques**

- Using animations to demonstrate algorithm progress (how paths are found)

- Graphical tools like Graphviz, Pygame, or online graph simulation tools

These topics cover everything from basic graph theory to complex algorithms, making your presentation well-rounded and informative.