Amazon SDE I Interview: Trees & Graphs Cheat Sheet

Tree Basics

Trees are non-linear, hierarchical data structures with nodes and edges.

A binary tree has at most two children per node. A binary search tree (BST) maintains order: left < root < right.

Traversals:

- Inorder (Left, Root, Right)
- Preorder (Root, Left, Right)
- Postorder (Left, Right, Root)
- Level-order (BFS using queue)

Important Tree Problems

- Maximum Depth of Binary Tree
- Invert Binary Tree
- Diameter of Binary Tree
- Lowest Common Ancestor
- Validate BST
- Serialize and Deserialize Binary Tree

Tree Techniques

- Use recursion for depth-based traversal
- Use queue for level-order traversal
- Use stack for iterative inorder/preorder
- Maintain parent or depth info when needed

Graph Basics

Graphs are a set of nodes (vertices) connected by edges.

Can be directed/undirected, weighted/unweighted.

Representations:

- Adjacency List: {node: [neighbors]}

Amazon SDE I Interview: Trees & Graphs Cheat Sheet

- Adjacency Matrix: 2D array
- Edge List: [(u, v)]

Graph Traversals

- Depth-First Search (DFS): Stack or recursion
- Breadth-First Search (BFS): Queue
- Detecting cycles in DFS (use visited & recursion stack)
- Topological sort (for DAGs)

Graph Techniques

- Use visited set to avoid re-visits
- Detect cycles with extra structures (e.g., parent or recursion stack)
- BFS for shortest path in unweighted graphs
- Union-Find for connected components / cycle detection

Top LeetCode Problems

Trees:

1. Maximum Depth of Binary Tree
2. Invert Binary Tree
3. Binary Tree Level Order Traversal
4. Path Sum
5. Lowest Common Ancestor of a BST

Graphs:

1. Number of Islands
2. Clone Graph
3. Course Schedule
4. Pacific Atlantic Water Flow
5. Detect Cycle in Directed/Undirected Graph