Competitive Programming Training

Breadth-First Search

Graph Traversal Methods

Breadth-First Search (BFS)

visits the child vertices before visiting the sibling vertices

uses queues as data structure

Depth-First Search (DFS)

visits the neighbor vertices before visiting the child vertices

uses stacks as data structure

Problem 1: Given a graph, check if there is a path between any pair of nodes

- ➔ Pick any node and traverse the graph from said node
- ➔ After the traversal, check if every node has been visited at least once.

Problem 2: Given an undirected graph, check the shortest distance between a given pair of nodes

- ➔ Pick the first node from the pair and start traversal using BFS
- ➔ During BFS, record the current length of the path visited
  - ◆ If there is an edge between u and v, then
    - `dist[v] = dist[u] + 1`
- ➔ When the second node is encountered, the distance stored is the shortest distance
- ➔ Note that this only works on **undirected graphs**

Problem 3: Given an undirected graph, check if the graph is a tree

- ➔ In a tree, the number of nodes is **exactly 1 more** than the number of edges
- ➔ In addition, the given graph has to be connected (Problem 1)
- ➔ If **both conditions are true**, then the given graph is a tree