**BFS and DFS**

<https://youtu.be/pcKY4hjDrxk?si=kqwW8ayrJNWQkazc>

BFS 🡪 Breadth First Search

DFS 🡪Depth First Search

Bothe of them is **graph traversal**

Two terms need to remember:

1. Visiting a vertex
2. Exploration of vertex (if I am on a particular vertex then visiting all of its adjacent vertex)

|  |
| --- |
| BFS: 1, 2, 4, 5, 7, 3, 6  Approach: We will visit a vertex then will go to the next vertex for exploration  DFS: 1, 2, 3,6, 7, 4, 5  Approach: We go forwards exploration to the end as the end reaches then we return immediate then continue |
| since it is a binary tree, we will perform level order  BFS = level order of a binary tree  BFS: 1, 2, 3, 4, 5, 6, 7  DFS = preorder  DFS: 1, 2, 4, 5, 3, 6, 7 |
| **BFS:**   1. Take a Queue 2. Initial step 3. takes any vertex and add it to queue 4. Repeating steps (take out the vertex from queue and start exploring it to its adjacent)   **BFS spanning tree:**   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 1 | 4 | 2 | 3 | 5 | 8 | 7 | 10 | 9 | 6 |     **What we learn:**  we can take any vertex  While exploring I can take any order I like  **Rule:**  when I visit a vertex, I must visit all of is adjacent vertices then go for next new vertex for exploration from the Queue only  Queue and exploration should be completely done  **Few more valid BFS:**   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 1 | 2 | 4 | 8 | 5 | 7 | 3 | 6 | 10 | 9 |   1)   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 5 | 2 | 8 | 7 | 6 | 3 | 1 | 9 | 10 | 4 |   2)  Can perform numerous numbers of valid BFS |
| **DFS:**     1. Take a Stack 2. I can start any vertex I like 3. Repeating step (as the new vertex is visited start exploring it ) 4. Rule once you visited one vertex suspend its exploration and start exploring new vertex one adjacent and then leave another adjacent vertex we will see them afterwards  |  | | --- | |  | |  | | ~~7~~ | | 8 | | 2 | | 4 | | 1 |  |  | | --- | |  | |  | |  | | ~~8~~ | | 2 | | 4 | | 1 |  |  | | --- | |  | |  | |  | |  | | ~~2~~ | | 4 | | 1 |  |  | | --- | |  | |  | |  | |  | |  | |  | | ~~1~~ |  |  | | --- | |  | |  | |  | |  | |  | |  | |  |   Dfs: Pre order: 1, 4, 3, 10, 9, 2, 8, 7, 5, 6   |  | | --- | |  | |  | |  | |  | |  | | ~~4~~ | | 1 |  |  | | --- | |  | |  | |  | |  | | ~~3~~ | | 4 | | 1 |  |  | | --- | |  | | ~~5~~ | | 7 | | 8 | | 2 | | 4 | | 1 |   **DFS spanning tree:**    ⇐ Back edges  Few more valid DFS:   1. 1, 2, 8, 7, 5, 6, 3, 9, 10, 3, 4 2. 3, 4, 1, 2, 5, 6, 7, 8, 10, 9 |
| Time complexity for both of this methos is O(n) n = number of vertices |

**Practice problem of DFS and BFS from different section**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 1. Flood Fill |  |  |  |
| 1. Number of Islands |  |  |  |
| 1. Find the level of given node in an Undirected Graph |  |  |  |
| 1. Connected Components in a Graph |  |  |  |
| 1. Monk and the Islands |  |  |  |
| 1. Detect cycle in an undirected graph |  |  |  |
| 1. Detect Cycle in a Directed Graph |  |  |  |

**Problems related to Graph Traversal (BFS+DFS)**

1. https://leetcode.com/problems/number-of-islands
2. https://www.geeksforgeeks.org/find-the-level-of-given-node-in-an-undirected-graph
3. https://www.hackerearth.com/problem/algorithm/connected-components-in-a-graph
4. https://www.hackerearth.com/practice/algorithms/graphs/breadth-first-search/practice-problems/algorithm/monk-and-the-islands
5. https://www.geeksforgeeks.org/detect-cycle-undirected-graph
6. https://www.geeksforgeeks.org/detect-cycle-in-a-graph