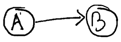
BFS DFS ABAIAJ section

26 February 2025 19:39

Graph Data Structure is a collection of **nodes**. Nodes are connected by **edges**. Edges represent connection between nodes.

Directed graph:



You can go from node A to B, but not B to A. Arrow will be present.

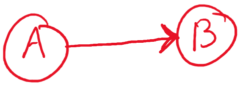
Undirected graph:

You can go from B to A and also from B to A. Arrow is absent.

BFS is a graph traversal algorithm that explores all the neighbours of a node before moving on to their neighbours.

DFS is a graph traversal algorithm that explores as far as possible along each branch before backtracking.

Directed



A to B,

Undirected

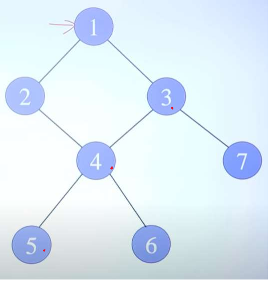
A to B ~~& B to A~~

Graphs Traversal

To traverse a Graph means to start in one vertex, and go along the edges to visit other vertices until all vertices, or as many as possible, have been visited.

2 techniques: BFS (Breadth first search), DFS (depth first search)

Queue →

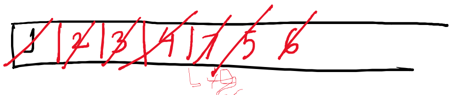


BFS traversal from node 4.

BFS Algorithm

1. Push the starting node into the queue and mark it as visited.
2. While the queue is not empty, repeat:
   - **Remove** an element (node) from the front of the queue.
   - **Process the node** (if required). Print it
   - **Push all its unvisited neighboring nodes** into the queue and mark them as visited.

Queue



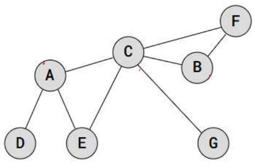
Print → 1 2 3 4 7.5 6.

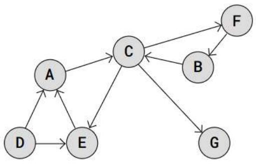
2 3 5 6

V V

Iterative DFS Algorithm (Using a Stack)

1. **Push start element in stack and print it.**
2. **Repeat till stack is not empty:**
   a. **See the top element in stack.**
   b. **If all its neighbours have been visited, remove the top item from stack.**
   c. **Else push one of its unvisited neighbours and continue the process.**



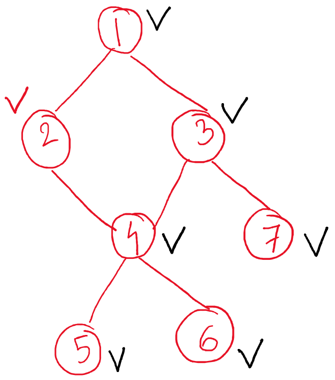


1,4 → U,V .

DFS Traversal.

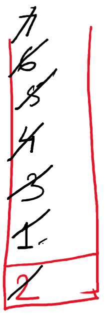
Stack → FILO / LIFO .

4,7



Iterative DFS Algorithm (Using a Stack)

1. **Push start element in stack and print it. Mark it visited.**
2. **Repeat till stack is not empty:**
   a. **See the top element in stack.**
   b. **If all its neighbours have been visited, remove the top item from stack.**
   c. **Else push one of its unvisited neighbours, mark it visited, print it and continue the process.**



top= ~~2~~ ~~1~~ ~~3~~ ~~4~~ ~~5~~ ~~4~~.~~6~~ ~~4~~.~~3~~.~~1~~

~~3~~ ~~1~~ 2

Print: 2 1 3 4 5 6 7

5,6