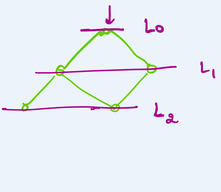
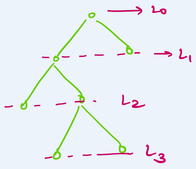
Wednesday, 6 October 2021 12:04 PM

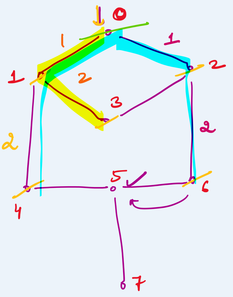
Breadth First traversal

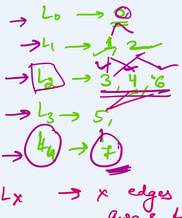
(level by level)





1. Appoint any node as root (L_0)
2. Nodes adjacent to root are considered as L_1.
3. adj to adj → L_2





L_x → x edges away from root.

✓ 0 | 1 | 2 | 3 | 4 | 6 | 5 | 7

0, 1, 2, 3, 4, 6, 5, 7

pseudo code BFS (G, s) {

vis → [ ]
Q → queue
q.enqueue(s);
mark s as visited vis[s] = t;
while (Q is not empty) {
v = q.dequeue();
// processing all neighbours of v
if w is not visited
q.enqueue(w);
mark w as visited

}

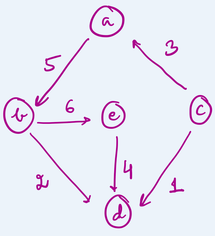
∴ BFS can be used to find the shortest path b/w two nodes in a undirected & unweighted graph

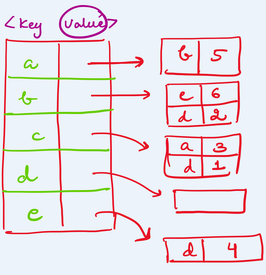
Graphs (Implementation)

- directed / undirected
- weighted

Hashmap < String, Vertices > (String: node value; Vertices: neighbours.)

Hashmap < String, Integer >

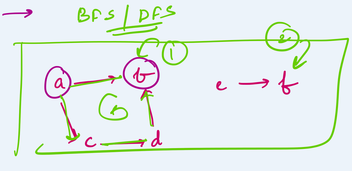




a, b → 2

↳ undirected edge

1. Given a graph, check whether it is connected or not.



t	t	t	t	t	t
a	b	c	d	e	f