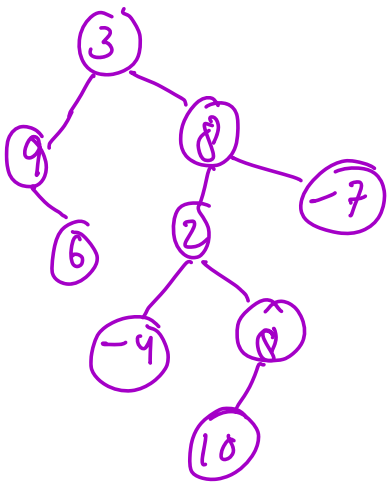
Level order traversal



3, 9, 8, 6, 2, -7, -4, 8, 10

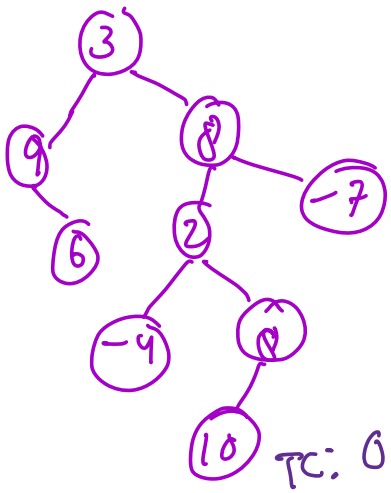
3 9 8 6 2 -7 -4 8 10

3 9 8 6 2 -7
-4 8 10

```
void level (root)
{
    queue <Node> q;
    q.push(root)
    while (q.size() >0)
    {
      Node f = q.front();
        q.pop();
        print(f.data);
          if (f.left != null)
            q.push (f.left)
          if (f.right != null)
            q.push (f.right);
}
```

Shivank · agrawal_1
@ rcaleb.com

DFS
BFS



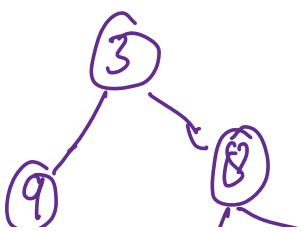
TC: O(N)
SC: O(N)

```
void level (root)
{
    queue < Node > q;
    q.push(root)
    while ( q.size() >0)
    {
     Node f = q.front();
        q.pop();
        print( f.data);
      → if (f.left != null)
          q.push (f.left)
        if ( f.right != null)
          q.push (f.right);
}
```

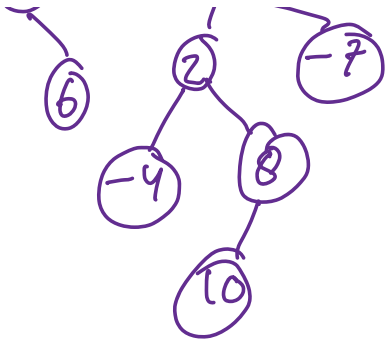
f 

3, 9

---

0 ⇒ 

3 in
9 8 in



TC : O(N)
SC : O(N)

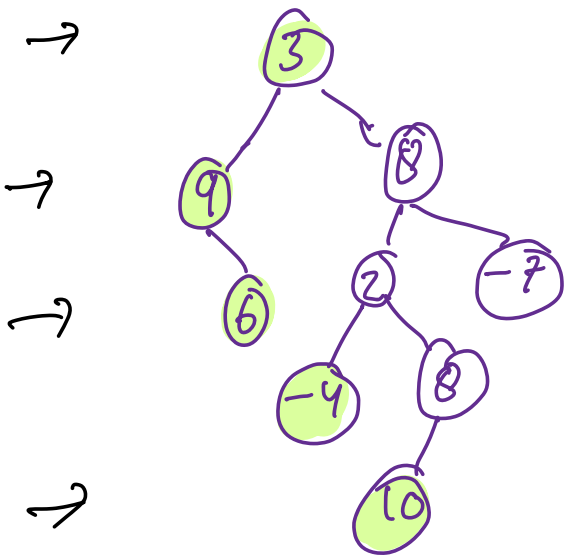
3 \n
9 8 \n

| 3 | Null | 9 | 8 | Null | 6 | 2 | -7 | Null |

```
void level (root)
{
    queue < Node > q;
    q.push (root);
    q.push (Null);
    while ( q.size() > 0)
    {
      Node f = q.front();
      q.pop();
       if (f == Null)
       {
          print ('\n');
          if ( q.size () > 0) q.push(Null
          continue;
       }
      print (f.data)
      if ( f.left != null)
          q.push (f.left)
```

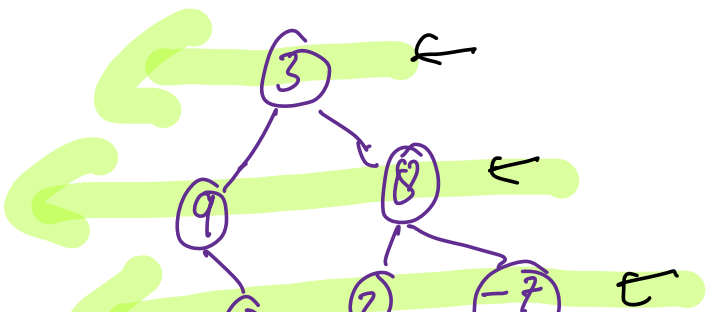
```
if (f.right != null)
    q.push(f.right);
}
```

Q⇒ Left view

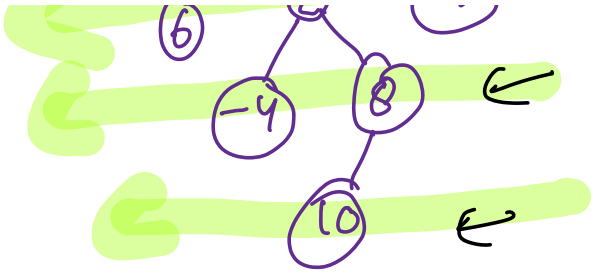


① We need to point first ele of each level.

② Imidite value after NULL except root node



3
8 9
-7 2 6
0 -4

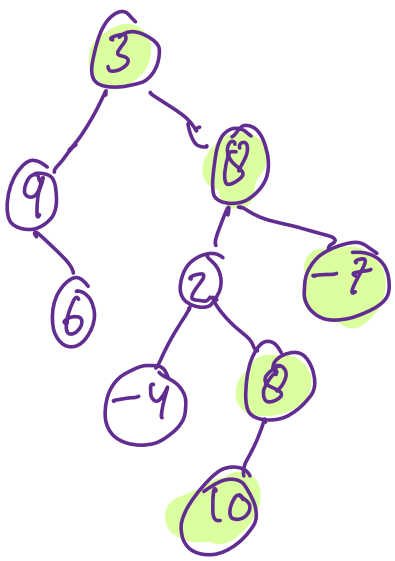


10

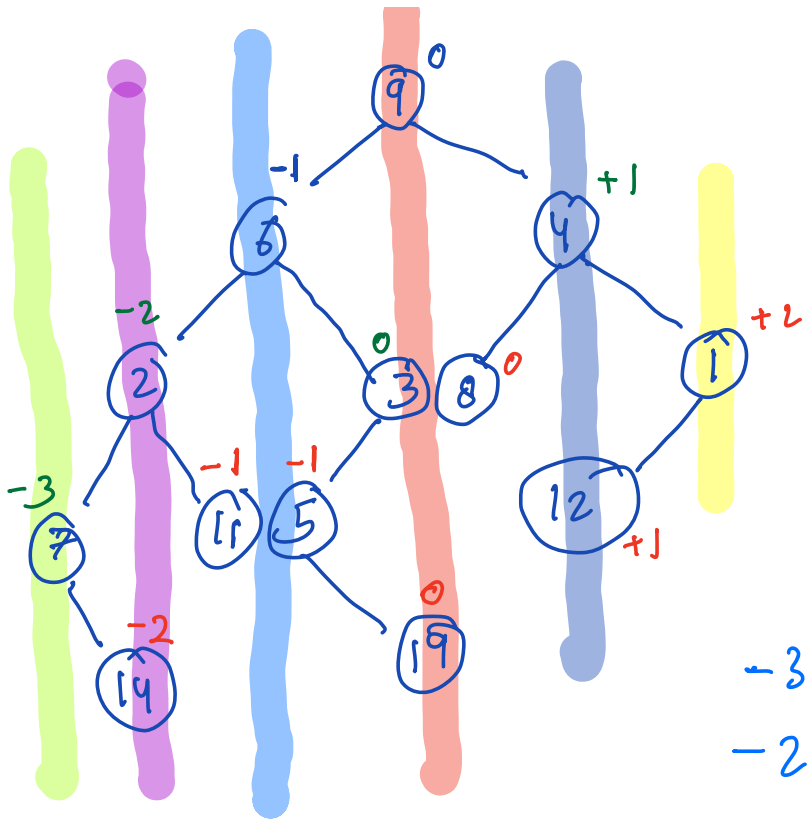
3 \n
8 9 \n

3	NULL	8	9	NULL	-7	2	6	NULL

Right view



Vertical level order Traversal



7
2, 14
6, 11, 5
9, 3, 8, 19
4, 12
1

-3 : 7
-2 : 2, 14

(9, 0)	(6, -1)	(4, 1)	(2, -2)	(3, 0)	(8, 0)	(1, 2)

(7, -3)	(11, -1)	(5, -1)

TC: O(N)
SC: O(N)

0 : 9, 3
-1, : 6
1 : 4
-2 : 2

queue < pair < Node, int > > q
map < int, List < Node >> mp.
q. insert ({ root, 0 })
minLevel = +∞

Class Pair
{

```
maxLevel = -0
while( q.size() >0)
{
pair<Node, int> p = q.front();
  q.pop();
 Node t = p.first;
 int level = p.second;
minLevel = min(minLevel, level)
maxLevel = max(maxLevel, level)
 mp[level].add(t);
   if (t.left != Null)
          q.insert({t.left, level-1})
   if (t.right != Null)
          q.insert({t.right, level+1})
}

for (i = minLevel , i <= maxLevel; i++)
{
          // To Do  Hw

}
```

Node fr,
Node sec

Break: 9:26

Top view

Bottom view

} H.W