Worksheet:
Binary Tree

Recap

- A tree is a data structure that has a hierarchical structure where there are parents and children, unlike normal graphs
- A binary tree is a tree in which each node has at most two children, left and right
- There are different ways of traversing a binary tree in a Depth First Search manner:
  a. Preorder: Node, Left, Right
  b. Inorder: Left, Node, Right
  c. Postorder: Left, Right, Node
- You can traverse trees recursively and iteratively
- You can also traverse a tree with BFS (Breadth First Search), useful for “level order traversal”
-

Exercise 1

Learn the basics traversals

You should be able to do these in your sleep. If you have difficulties writing traversal, then you should practice these until you get them fundamentally so that your coding speed is fast once you’re getting interviewed. When you’re interviewing, you shouldn’t get bogged down by implementation of algorithms, you should use your brain power to solve the “brain teaser” aspect of the problem.

Binary Tree Preorder Traversal

Example:

```
Input: [1,null,2,3]
   1
    \
     2
    /
   3

Output: [1,2,3]
```

Follow up: Do it Iteratively

Binary Tree Inorder Traversal

Example:

```
Input: [1,null,2,3]
   1
    \
     2
    /
   3

Output: [1,3,2]
```

Follow up: Do it Iteratively

Binary Tree Inorder Traversal

Example:

```
Input: [1,null,2,3]
   1
    \
     2
    /
   3

Output: [3,2,1]
```

Follow up: Do it Iteratively

Binary Tree Level Order Traversal

For example:

Given binary tree `[1,2,3,null,null,4,5],`

```
    1
   / \
  2   3
    /  \
   4    5
```

return its level order traversal as:

```
[
  [1],
  [2,3],
  [4,5]
]
```

Exercise 2

Solve Common Binary Tree Questions

Maximum Depth of Binary Tree

Given a binary tree, find its maximum depth.

The maximum depth is the number of nodes along the longest path from the root node down to the farthest leaf node.

Note: A leaf is a node with no children.

Example:

Given binary tree `[1,2,3,null,null,4,5]`,

```
    1
   / \
  2   3
    /  \
   4    5
```

return its depth = 3.

Symmetric Tree

Given a binary tree, check whether it is a mirror of itself (ie, symmetric around its center).

For example, this binary tree `[1,2,2,3,4,4,3]` is symmetric:

```
    1
   / \
  2   2
 / \ / \
3  4 4  3
```

But the following `[1,2,2,null,3,null,3]` is not:

```
    1
   / \
  2   2
   \   \
   3    3
```

Follow up:

Solve it recursively and iteratively