Data Structures

Binary Search Homework 3

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

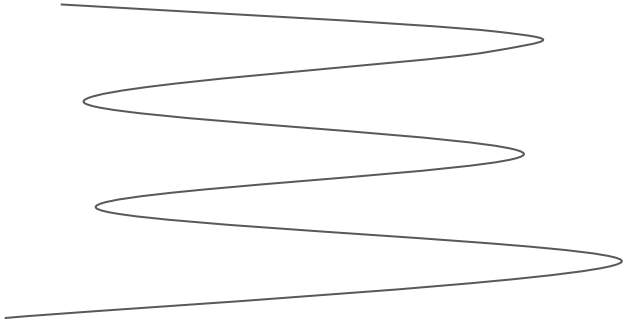
Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

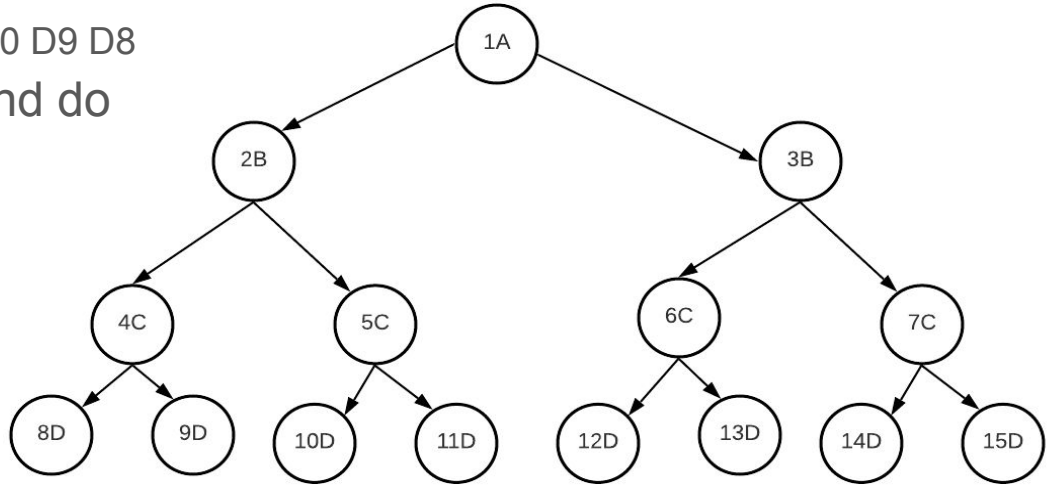
Problem #1: Recursive Level Order Traversal

- **def** level_order_traversal_recursive(self)
- We would like to print the level-order traversal in a recursive way, instead of the iterative approach using a queue
  - Don’t worry if much worse in complexity
- Compute the time complexity

Problem #2: LeetCode 103 - Binary Tree Zigzag Level Order Traversal

- Return a spiral way to the tree
  - **Even** levels are **reversed**
- Output
  - Level 0: A1
  - Level 1: B3 B2
  - Level 2: C4 C5 C6 C7
  - Level 3: D15 D14 D13 D12 D11 D10 D9 D8
- We can trivially, get the levels and do Reversing.
  Don’t use reversing operation



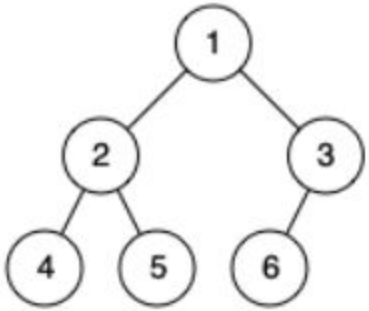


Problem #3: LeetCode 958 - Check Completeness of a Binary Tree

Given the `root` of a binary tree, determine if it is a *complete binary tree*.

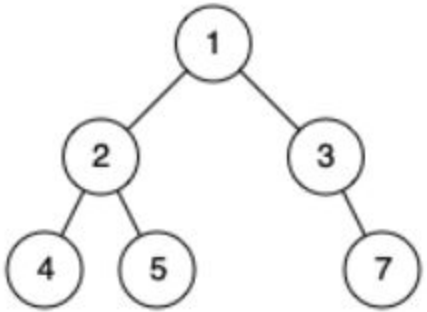
In a **complete binary tree**, every level, except possibly the last, is completely filled, and all nodes in the last level are as far left as possible. It can have between `1` and 2^h nodes inclusive at the last level `h`.

Example 1:



Input: root = [1,2,3,4,5,6]
Output: true
Explanation: Every level before the last is full (ie. levels with node-values {1} and {2, 3}), and all nodes in the last level ({4, 5, 6}) are as far left as possible.

Example 2:




```
Input: root = [1,2,3,4,5,null,7]
Output: false
Explanation: The node with value 7 isn't as far left as possible.
```

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."