Trees and their Basic Properties!

In this chapter, we are going to study the basics of the tree data structure!

We'll cover the following

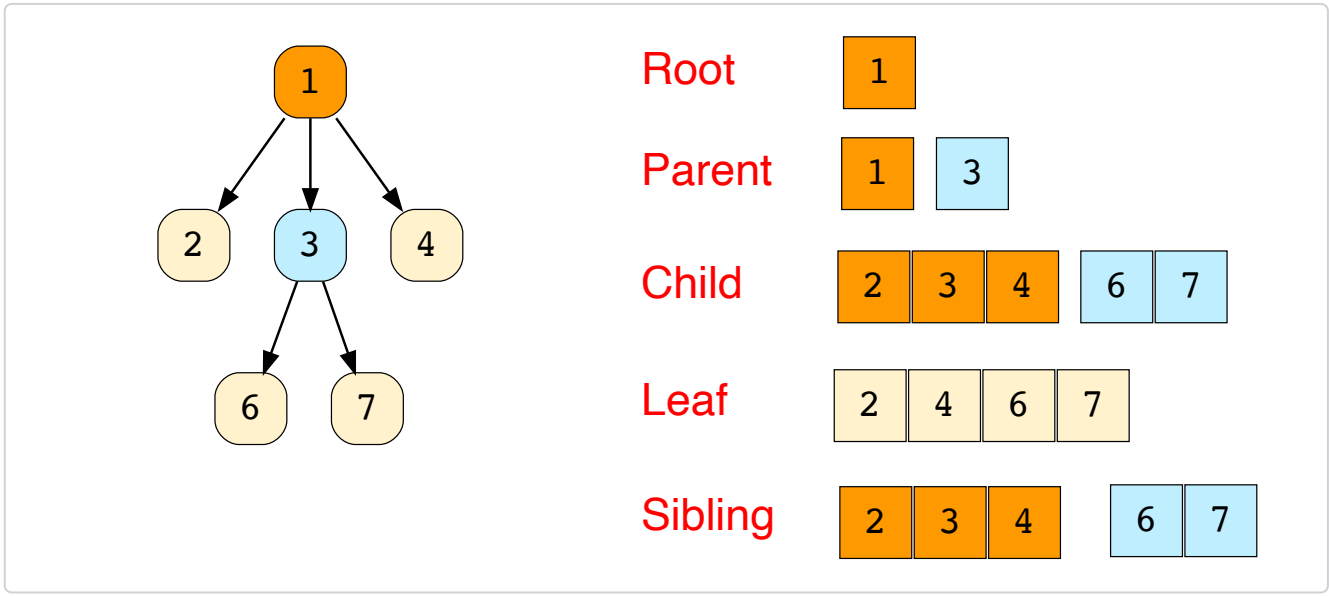
Introduction#

Trees consist of vertices (nodes) and edges that connect them. Unlike the linear data structures that we have studied so far, trees are hierarchical. They are similar to Graphs, except that a **cycle** cannot exist in a Tree - they are **acyclic**. In other words, there is always exactly one path between any two nodes.

- **Root Node**: A node with no parent nodes. Generally, trees don't *have* to have a root. However, **rooted trees** have one distinguished node and are largely what we will use in this course.
- **Child Node**: A Node which is linked to an upper node (*Parent Node*)
- **Parent Nodes**: A Node that has links to one or more *Child Nodes*
- **Sibling Node**: Nodes that share same *Parent Node*
- **Leaf Node**: A node that doesn't have any *Child Node*

- **Ancestor Nodes**: the nodes on the path from a node *d* to [illegible]. Ancestor nodes include node *d*'s parents, grandparents, and so on

The figure below shows all the terminologies described above:



Explanation#

In the figure above, **1** is the *Root* as well as *parent node* to child nodes **2,3,** and **4**. Node **3** is a parent node to child nodes **6** and **7**. And as nodes **2,3, and 4** share the same parent node **1**, so they are siblings to each other. Similarly, **6** and **7** are also *sibling nodes* as their parent is same, that is **3**.

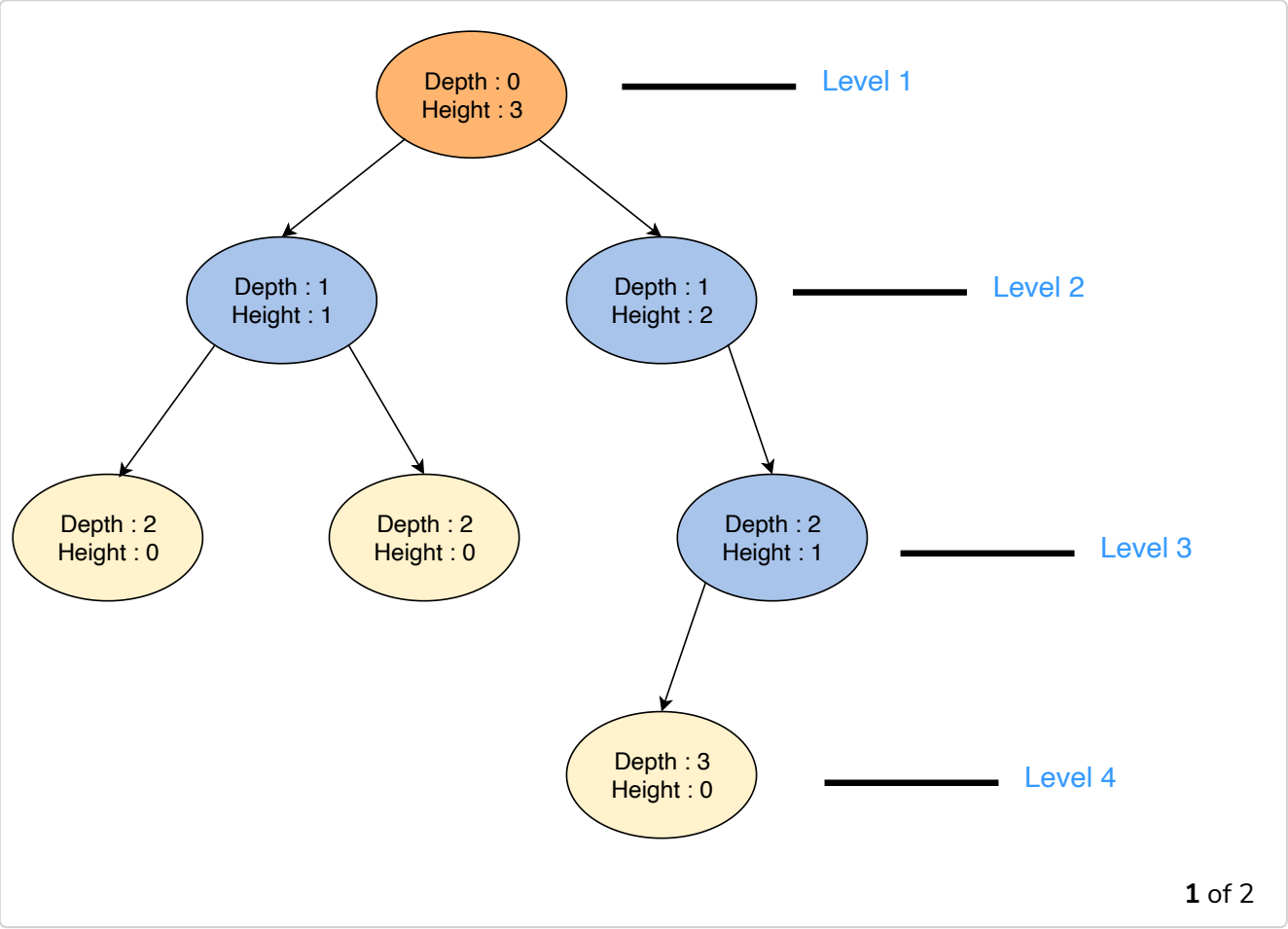
Other Terminology and Formulas#

Some other common terminologies used in trees are:

- **Sub-tree**: For a particular non-leaf node, a collection of nodes, essentially the tree, starting from its child node. The tree formed by a node and its descendants.

- **Degree of a node**: Total number of children of a node

- **Length of a path**: The number of edges in a path
- **Depth of a node *n***: The length of the path from a node *n* to the root node. The depth of the root node is 0.
- **Level of a node *n***: (Depth of a Node)+1
- **Height of a node *n***: The length of the path from *n* to its deepest descendant. So the height of the tree itself is the height of the root node and the height of leaf nodes is always 0.
- **Height of a Tree**: Height of its root node



There are a total of six sub-trees in the figure above where each node is labeled with a number. Each node has two sub-trees, namely the *left* and *right* subtrees. Leaf nodes have no sub-trees.

Some Tree Types#

Many different types of trees exist which are optimized for particular use-cases. Each tree type offers its own particular structure and hence space-time complexity for different operations. Some commonly used trees include,

- *Binary Trees*
- *Binary Search Trees*
- *AVL Trees*
- *Red-Black Trees*
- *2-3 Trees*

We'll study each of these types in detail in upcoming chapters!

The N-ary Tree#

In graph theory, an N-ary tree is a rooted tree in which each node has no more than N children. It is also sometimes known as a k-way tree, a k-ary tree, or an M-ary tree. A binary tree is a special case where k=2, so they can have a maximum of **2** child nodes and a minimum of **0** child nodes. Binary trees are used extensively in a plethora of important algorithms!

Quick quiz on N-ary trees!

Q Can you guess what the value of N is in the figure below?

A) 5

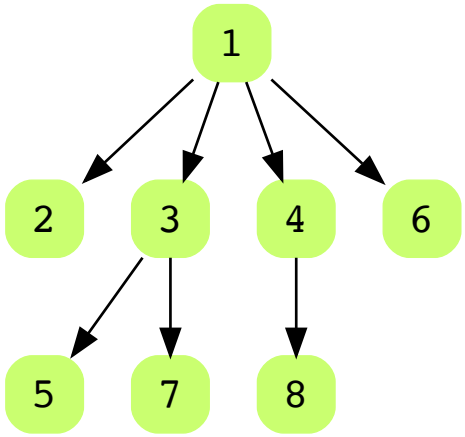
B) 1

C) 4

D) 3

Submit Answer

Reset Quiz



N-ary tree for quiz

In the next lesson, we'll be looking at a couple of important tree properties - tree height balancedness and tree depth balancedness.

Back

Graph Quiz: Test your understanding ...

Next

What makes a tree 'balanced'?

Completed

Report an Issue