# **Binary Search Tree Properties**

## **Path**

* A **path** from node *n*1 to *nk* is defined as
  + A sequence of nodes *n*1,*n*2,...,*nk*
  + such that
  + *ni* is the parent of *ni*+1 for 1 ≤ *i* < *k*
* Notice that in a tree there is exactly **one path** **from the root to each node**.
* If there is a path from *n*1 to *n*2, then
  + *n*1 is an **ancestor** of *n*2
  + and
  + *n*2 is a **descendant** of *n*1.
* If *n*1 != *n*2
  + *n*1 is a **proper ancestor** of *n*2
  + and
  + *n*2 is a **proper descendant** of *n*1.

## **Length**

* The **length** of this path is the **number of edges on the path**.
* From the recursive definition, we find that a tree is a collection of
  + ***N* nodes**, one of which is the root

and

* + ***N* − 1 edges**
* That there are *N* − 1 edges follows from the fact that each edge connects some node to its parent, and every node except the root has one parent.
* Therefore, if there is a path from *n*1 to nk, then the path length will be equal to ***k* − 1**
* There is a path of length **zero** from every **node to itself**.

## **Depth**

* For any node ***ni***, the **depth** of ***ni***is the length of the ***unique path*** from the **root to *ni***.
* The depth of a tree is equal to the depth of the deepest leaf
  + This is always equal to the height of the tree
* The **root is at depth 0**.

## **Height**

* The **height** of ***ni***is the ***length*** of the ***longest path*** from ***ni*** to the ***furthest leaf node***.
* This is found by counting the number of edges between ***ni*** and its ***furthest leaf node***.
* The **height of a tree** is equal to the **height of the root**.
  + Height of ***T*** (tree) is equal to height of ***r*** (root)
* The height of a tree would be
  + the height of its root node  
    or equivalently,
  + the depth of its deepest leaf
* All **leaves** are at height 0.

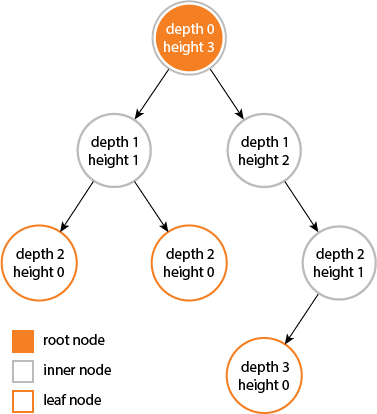
## **Width**

* The diameter (or width) of a tree is the number of ***nodes*** on the **longest path between any two leaf nodes**.
* Note that this path **does not have to pass through the root**.

## **Example**

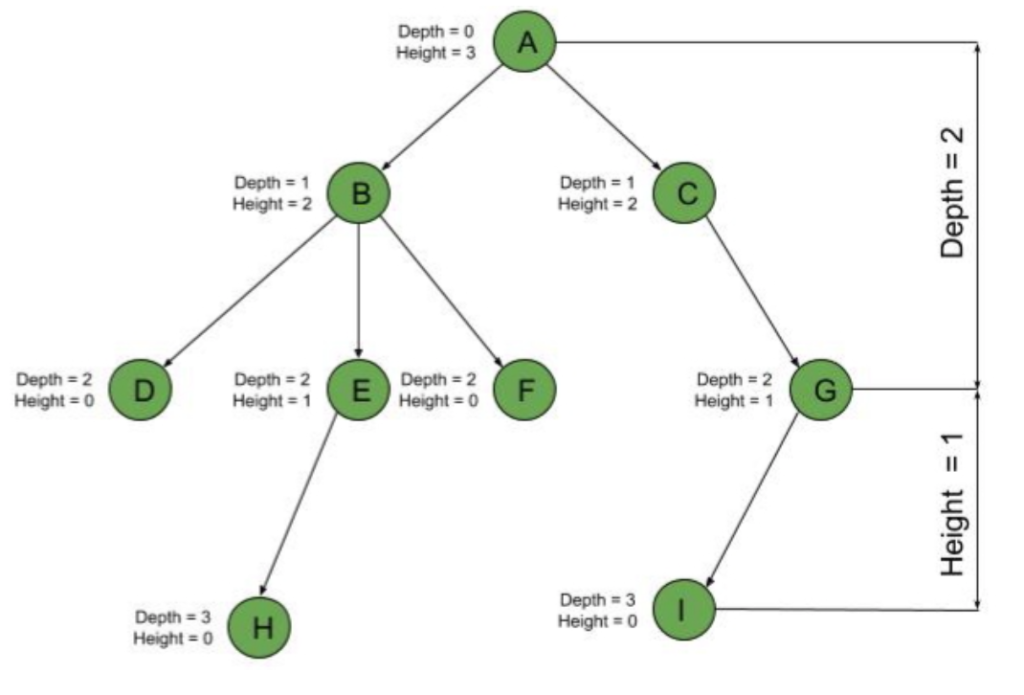
Height: 3

Width: 6 nodes



## **Depth vs. Height**

* The depth of a node is synonymous to the **level** of a node (depending on who you ask).
* Two nodes may have the same depth (or be at the same level) but can vary in height.
* This is because they are the same distance from the root but may have varying depths to leaves.
* This is made clearer in the following figure.



* Notice how nodes D and G are at the same depth, which is 2, but have different heights.
* D has a height of 0, because it has no children, and G has a height of 1, because it has one child, I.

## **Binary Tree Height**

For binary trees, it is often convenient to use an equivalent recursive definition of height:

**T is a binary tree if either**

* If *T* is empty, its height is 0.
* If *T* is a nonempty binary tree, then because *T* is of the form *r*

*r*

page455image1660818112page455image1660819840

*TL*  *TR*

the height of *T* is 1 greater (the +1 is the root) than the height of its root’s taller subtree:

***height* (*T*) = 1 + *max*{*height* (*T*L), *height* (*T*R)}**

* Remember,
  + The depth of a tree is equal to the depth of the deepest leaf
  + The height of a tree is equal to the height of the root.
* Therefore, the depth and height of the tree are always equal.
  + the height of the tree is equal to the depth of the tree
  + the maximum depth is equal to the maximum height.