Data Structure & Algorithms

Balanced Binary Tree

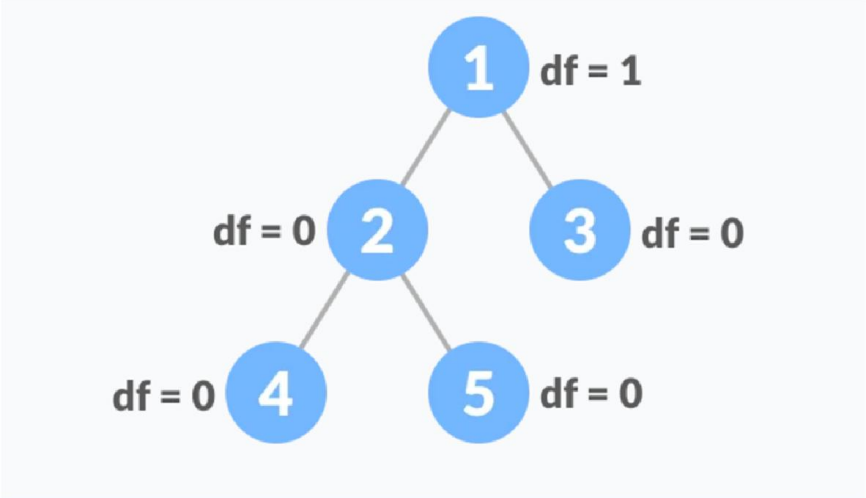
Definition

- A **balanced binary tree** is a binary tree in which the difference between the height of the left subtree and right subtree of each node is not more than 1.

- A balanced binary tree is also known as a *height balanced tree*.

- In a balanced binary tree, the tree height is ***O(log n)***, where n is the number of nodes.

Example

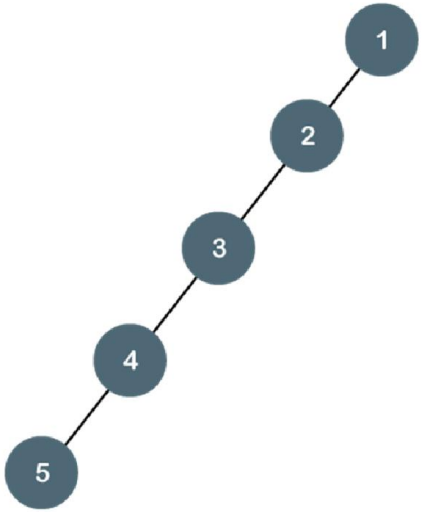


Balanced Binary Tree with depth at each level

df = |height of left child - height of right child|

Example

An example of a *imbalanced* binary tree is a **degenerate tree** where every parent node has only one child node:



Why is it important that a binary tree be balanced?

- We have seen that the efficiency of many important operations on trees is related to the Height of the tree - for example searching, inserting, and deleting in a BST are all *O(height)*.
- In general, the relation between Height (*h*) and the number of nodes (*n*) in a tree can vary from *h = n* (degenerate tree) to *h = log(n)* (balanced binary tree).
- **For efficiency's sake, we would like to guarantee that *h* equals to *O(log n)*.**

Self-balancing binary search tree

- **Self-Balancing Binary Search Trees** are *height-balanced* binary search trees that automatically keeps height as small as possible when insertion and deletion operations are performed on tree. The height is typically maintained in the order of *log n* so that all operations take *O(log n)* time on average.

- A **red-black** tree is a kind of self-balancing binary search tree.