How to determine if a binary tree is height-balanced?

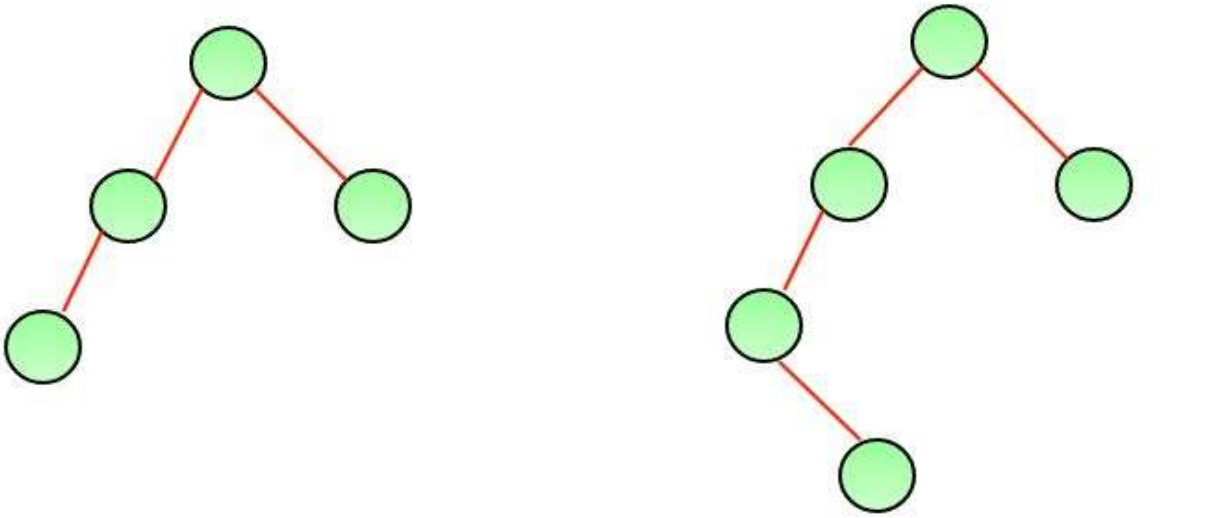
A tree where no leaf is much farther away from the root than any other leaf. Different balancing schemes allow different definitions of “much farther” and different amounts of work to keep them balanced.

Consider a height-balancing scheme where following conditions should be checked to determine if a binary tree is balanced.

An empty tree is height-balanced. A non-empty binary tree T is balanced if:

1) Left subtree of T is balanced
2) Right subtree of T is balanced
3) The difference between heights of left subtree and right subtree is not more than 1.

The diagram below shows two trees, one of them is height-balanced and other is not. The second tree is not height-balanced because height of left subtree is 2 more than height of right subtree.



A height balanced tree

Not a height balanced tree

To check if a tree is height-balanced, get the height of left and right subtrees. Return true if difference between heights is not more than 1 and left and right subtrees are balanced, otherwise return false.

How to determine if a binary tree is height-balanced?

Additional Tree Functions

Method	Description	Have we covered it?
`boolean` **`isBalanced`**`()`	returns true if binary tree with root as root is height-balanced	
`void` **`mirror`**`()`	Swaps the left and right subtree to generate a mirror of the original tree. Flips the order.	

Continue to implement the methods of a binary tree given above and last class. These methods will be very useful in the near future.