Trees

Balanced Binary Search Tree

Intermediate - java

Given a sorted array of numbers `a`, write a function `balancedBst(a)` to create a balanced binary search tree. A balanced Binary Search Tree has no more than one level of depth difference between the right and left sides of the tree.

Each value in the array `a` should correspond to a node value. The return value of `balancedBst()` will be the root node of the balanced tree. An empty array passed to `balancedBst()` should return `null`.

For example, given an array `a = {1,2,3,4,5,6,7,8}`, you want to create a balanced tree that may resemble the following:

```
       5
     /   \
    3     7
   / \   / \
  2   4 6   8
 /
1
```

The above figure represents a balanced tree because there is at most 1 level of difference between the depths of each side of the tree.

For this challenge you are given the class `TreeNode` with the members:

- `value`: the node value
- `left`: the left child node; defaults to `null`
- `right`: the right child node; defaults to `null`

The `print()` function is also implemented in the class `TreeNode`, so at any time you can print the root node to see a basic representation of the tree.

This challenge and variations of it were reported to have been asked at interviews with Google. If you've covered the material in Pass the Technical Interview with Java or an equivalent, you should be able to solve this challenge. If you have trouble, try refreshing your knowledge there first.