CSE 332: Data Structures and Parallelism

Exercises (AVL Trees)

Directions: *Submit your solutions using* `https://grinch.cs.washington.edu/cse332/verify-avl`*. Note that this problem is a bit different than the other on-line exercises, in that you do not need to "submit" your answer. You are allowed an infinite number of attempts, and as soon as you pass all 14 test cases, a score of 20 will be added to Canvas. This one is a bit tricky, but it is also a common interview question!*

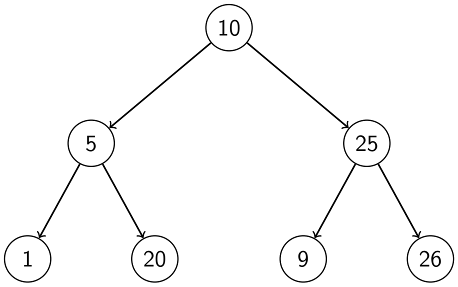
EX07. Always Verify Time! (20 points)

Write code for an $\Theta(n)$ worst-case algorithm that verifies that a tree is actually an AVL tree. You may assume the nodes of the tree look like:

```
class AVLNode {
   int key;
   V value;
   int height;
   AVLNode left;
   AVLNode right;
}
```

You must verify the **BST Property**, the **AVL Balance Condition**, and the correctness of the height information (that is, the value stored in the height field may not be correct). In each of the cases where it fails a property, return `false`. Otherwise, return `true`. A null node is considered a proper AVL tree of height -1. You are welcome to (encouraged!) to write helper functions.

As you are writing your code, we recommend that you keep in mind the following tree. Is THIS tree a proper AVL Tree? Look carefully....


10
5
25
1
20
9
26