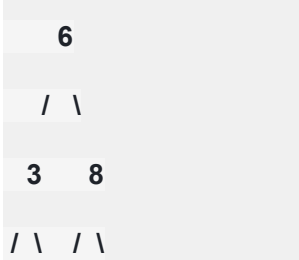
Given a Binary Search Tree of size N, find the Median of its Node values.

Example 1: Input:



Output: 6 Explanation: Inorder of Given BST will be: 1, 3, 4, 6, 7, 8, 9. So, here median will 6.

Example 2: Input:

Output: 5 Explanation:Inorder of Given BST will be: 1, 3, 4, 6, 7, 8. So, here median will (4 + 6)/2 = 10/2 = 5.

Your Task:

You don't need to read input or print anything. Your task is to complete the function findMedian() which takes the root of the Binary Search Tree as input and returns the Median of Node values in the given BST. Median of the BST is:

- **If number of nodes are even: then median = (N/2 th node + (N/2)+1 th node)/2**
- **If number of nodes are odd : then median = (N+1)/2th node.**

Expected Time Complexity: O(N).

Expected Auxiliary Space: O(Height of the Tree).

Constraints: 1<=N<=10000

Solution:

```
class Tree {

   public static void inorder(Node root, ArrayList<Integer> arr) {

      if (root == null) return;

      inorder(root.left, arr);

      arr.add(root.data);

      inorder(root.right, arr);

   }

   public static float findMedian(Node root) {

      ArrayList<Integer> arr = new ArrayList<>();

      inorder(root, arr);

      int n = arr.size();

      if (n % 2 == 0) {

            return (arr.get(n / 2 - 1) + arr.get(n / 2)) / 2.0f;

      } else {

               return arr.get(n / 2);

      }

   }

}
```