# PROBLEM

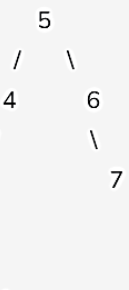
## Minimum element in BST

**Basic** **Accuracy: 70.95%** **Submissions: 147K+** **Points: 1**

Given the root of a **Binary Search Tree**. The task is to find the minimum valued element in this given BST.

**Example 1:**

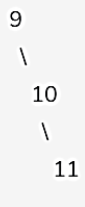
**Input:**



**Output:** 1

**Example 2:**

**Input:**



**Output:** 9

**Your Task:**

The task is to complete the function **minValue()** which takes root as the argument and returns the minimum element of BST. If the tree is empty, there is no minimum element, so return **-1** in that case.

**Expected Time Complexity:** O(Height of the BST)

**Expected Auxiliary Space:** O(1).

**Constraints:**

$0 <= n <= 10^4$

# CODE

```
#User function Template for python3


"""
class Node:
    def _init_(self, val):
        self.right = None
        self.data = val
        self.left = None
"""
```

```
class Solution:
   #Function to find the minimum element in
the given BST.
   def minValue(self, root):
      ##Your code here
      if root.left == None:
         return root.data
      return self.minValue(root.left)
```

## EXPLANATION

**Class Definition:**

**class Solution:**

Here, a class named Solution is defined, which will contain the method to find the minimum element in a BST.

**Method Definition:**

**def minValue(self, root):**

This method named minValue takes two parameters:

**self:** This parameter refers to the instance of the class itself. It's required in Python methods.

**root:** This parameter represents the root node of the BST.

**Code Explanation:**

**if root.left == None:**

**return root.data**

This condition checks if the left child of the current node root is None. If it is, then root itself contains the minimum value in the BST, so we return root.data.

**return self.minValue(root.left)**

If the left child of the current node is not None, it means the minimum value lies somewhere in the left subtree. So, we recursively call the minValue function on the left child of the current node until we reach a node whose left child is None. This effectively traverses down the left branch of the BST until we find the minimum value.

**Visualization:**

Think of a binary search tree. The minimum value is always the leftmost node in the tree. This code recursively traverses the left children of each node until it reaches a node without a left child, indicating the minimum value in the BST.