Solution: Find minimum value in Binary Search Tree

This review provides a detailed analysis of the different ways to solve the Find minimum value in Binary Search Tree challenge

We'll cover the following

Solution #1: Iterative `findMin()`

main.py

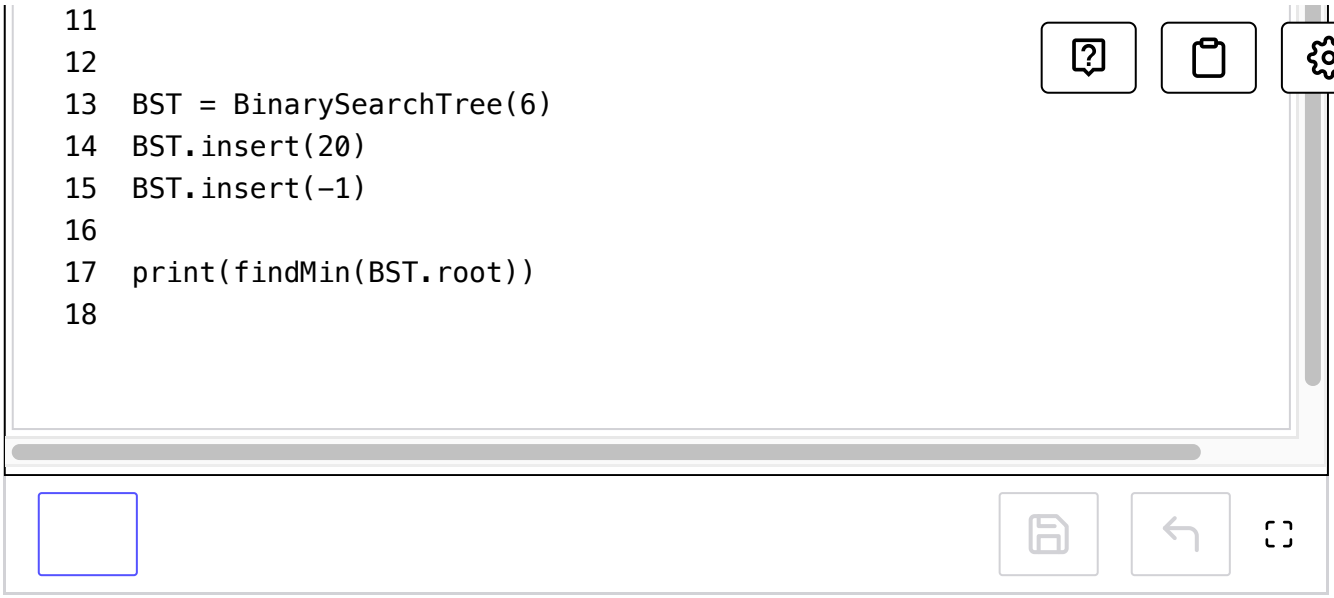
BinarySearchTree.py

Node.py

```
from Node import Node
from BinarySearchTree import BinarySearchTree


def findMin(root):
    if root is None:  # check for None
        return None
    while root.leftChild:  # Traverse until the last child
        root = root.leftChild
    return root.val  # return the last child
```

```
11
12
13  BST = BinarySearchTree(6)
14  BST.insert(20)
15  BST.insert(-1)
16
17  print(findMin(BST.root))
18
```



This solution first checks if the given root is `None` and returns `None` if it is. Then, it moves on to the left sub-tree and keeps going to each node's left child until the left-most child is found.

Time Complexity#

The time complexity of this solution is in $O(h)$. In the worst case, the BST will be left skewed and the height will be n and so the time complexity will be $O(n)$.

Solution #2 : Recursive `findMin()`#

main.py

BinarySearchTree.py

Node.py

```
from Node import Node
from BinarySearchTree import BinarySearchTree


def findMin(root):
    if root is None:  # check if root exists
```

```
        return None
    elif root.leftChild is None:  # check if left child exists
        return root.val  # return if not left child
    else:
        return findMin(root.leftChild)  # recurse onto the left child


BST = BinarySearchTree(6)
BST.insert(20)
BST.insert(-1)

print(findMin(BST.root))
```

In this solution, we check if the root is `None`, if it is, `None` is returned. Otherwise, we check to see if the left child of the current node is `None`, if it is, then this root is the left most node and so we return the value there. If a left node exists, we call the `findMin()` function on it.

Time Complexity#

The time complexity of this solution is the same as the time complexity of the solution above, namely $O(h)$ and in the worst case $O(n)$.


Re
Is