DCSCI/CSC 2720: Data Structures
Assignment 3

Instructor: Shiraj Pokharel

Due : @ 11:00 PM ET , 11/08.
Late Submission (with a 25% penalty) deadline : 11:00 PM ET, 11/09

Answer the below questions.
You must submit your responses as a SINGLE Jupyter Notebook, where each program is put in separate Jupyter Notebook cells within that SINGLE Jupyter Notebook. Do **NOT** submit Colab links.
Failure to comply with this simple requirement will result in a score of Zero. Please, be careful not to be assigned a Zero score this way.

Few Rules to be followed, else will receive a score of ZERO

(1) Your submissions will work exactly as required.

(2) Your files shall not be incomplete or worse corrupted such that the file does not compile at all. Make sure you submit a file that compiles.

(3) Your submission will show an output. Should you recive a Zero for no output shown do not bother to email me with "but the logic is perfect" !

Note that your program's output must **exactly** match the specs(design , style) given here for each problem to pass the instructor's test cases .
Design refers to how well your code is written (i.e. is it clear, efficient, and elegant), while *Style* refers to the readability of your code (commented, correct indentation, good variable names).

PROBLEM STATEMENT :
In this assignment we will explore a specific way to delete the root node of the **Binary Search Tree (BST)** while maintaining the **Binary Search Tree (BST)** property after deletion.

Your implementation will be as stated below:

[1] Delete the root node value of the BST and replace the root value with the appropriate value of the existing BST .

[2] Perform the BST status check by doing an **In-Order Traversal** of the BST such that even after deletion the BST is maintained.

```
# Class to represent Tree node
class Node:
   # A function to create a new node
   def __init__(self, key):
      self.data = key
      self.left = None
      self.right = None
```

The root element of the Binary Search Tree is given to you. Below is an illustrated sample of Binary Search Tree nodes for your reference, which in-fact is the same example we discussed in the lecture.

```
root = Node(4)
root.left = Node(2)
root.right = Node(6)
root.left.left = Node(1)
root.left.right = Node(3)
root.right.left = Node(5)
root.right.right = Node(7)
```

Your code will delete the root node value provided and replace the root value with appropriate value such that the BST property is maintaned which would be known by performing the In-Order Traversal of the new BST. Obviously, the In-order traversal of the resulting BST would need to be ascending order

When you follow the deletion process and the BST traversal specified - the complexity of the solution will be as below.

Time Complexity: $O(n)$
Space Complexity: $O(n)$

Submissions that don't meet the linear Time and Space complexities will only receive 50% credit.

Very Very Important :

(1) Your code should be well commented which explains all the steps you are performing to solve the problem. A submission without code comments will immediately be deducted 15 points !

(2) As a comment in your code, please write your test-cases on how you would test your solution assumptions and hence your code.
A submission without test cases (as comments) will immediately be deducted 15 points ! Please Remember : Although, written as comments - You will address your test cases in the form of code and not prose :)