Online Test COMP2100/6442 Semester 2

Question 2:
This question is composed of two items:

- A Red Black Tree Question (item 2.a) and
- A Branch Complete Question (item 2.b).

Please upload the following files to Wattle after you complete the tasks: `RBTree.java` and `BSTBranchCompleteTest.java`. In each file **write your UID and name where indicated** in the code.

2.a) [4 marks] Implement a method to count the number of **non-duplicate** broken nodes that fail to fulfill the properties 1 or 2 of a Red-Black Tree.

1. The root and leaf (NULL) nodes are black
2. If a node is red, then its parent is black

If a leaf node is broken (counts as 2).

By non-duplicate we mean: do not count the same broken node twice.

Hint: You should consider the following three cases:

Case 1:
If the root node is red, the root node is a broken node.

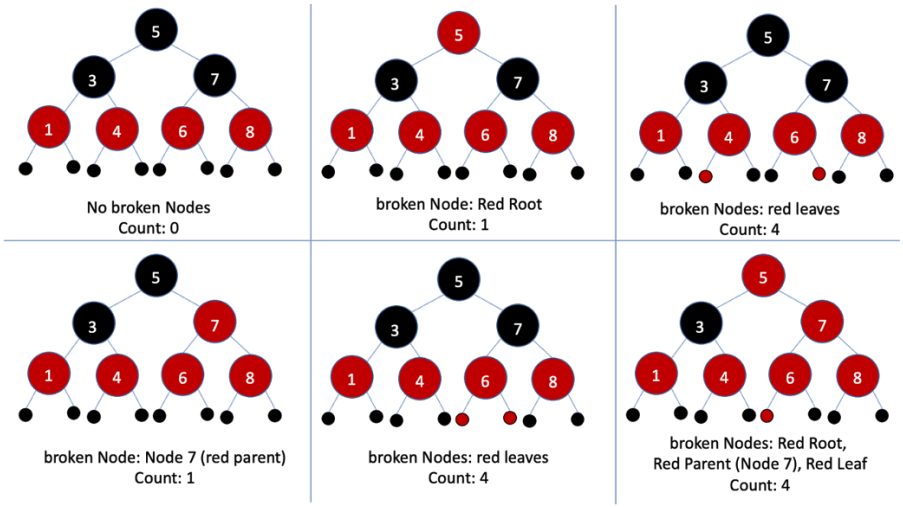
Case 2:
If a leaf node is red, the leaf node is a broken node.

Case 3:
If a red node has a red parent, the red parent node is a broken node. Note that if a leaf node is red and its parent is also red, then its parent is not a broken node (the leaf is the broken node (see case 2)).

You can define additional methods if you need. The method signature (the method that will be tested) is:

public int countBrokenNodes()

Example:



2.b) [2 marks] Implement the minimum number of JUnit test cases for **calcRBTree ()** that is branch complete. Read the code of **calcRBTree()** in the `BranchComplete.java` file and implement the **minimum number of test cases** in the `BSTBranchCompleteTest.java` file.

Note that each execution of an assertion counts as a single test case, therefore loops that execute the same assertion multiple times count as multiple tests.