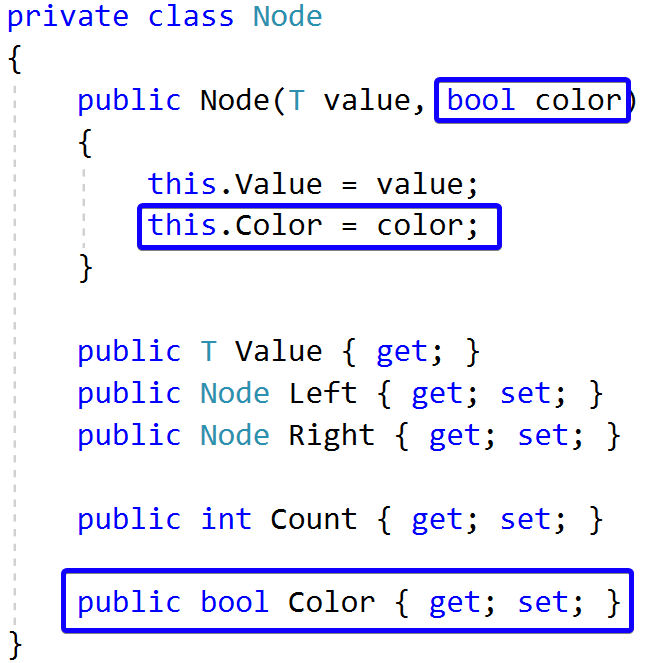
# Exercises: Implement Binary Search Tree

This document defines the **in-class exercises** assignments for the ["Data Structures" course @ Software University](https://softuni.bg/opencourses/data-structures).

This lab aims to implement the insertion functionality of a **red-black** **binary search tree**. You are given the skeleton of a fully implemented **BST**.

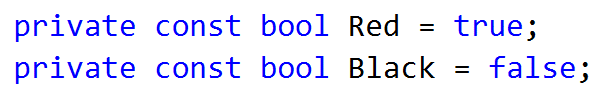
## Change the Node Data Structure

First, you will need to add a **color bit** to our node class:

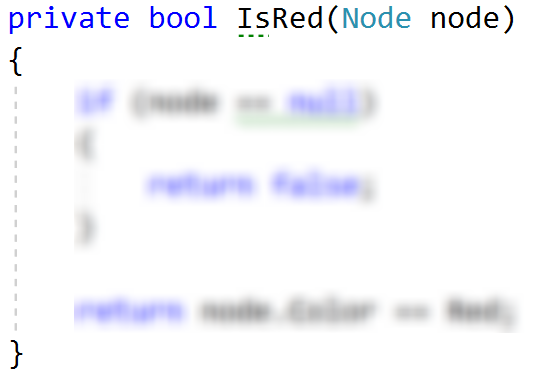


## Check Red Node

Add the following constants to your RedBlackTree class:

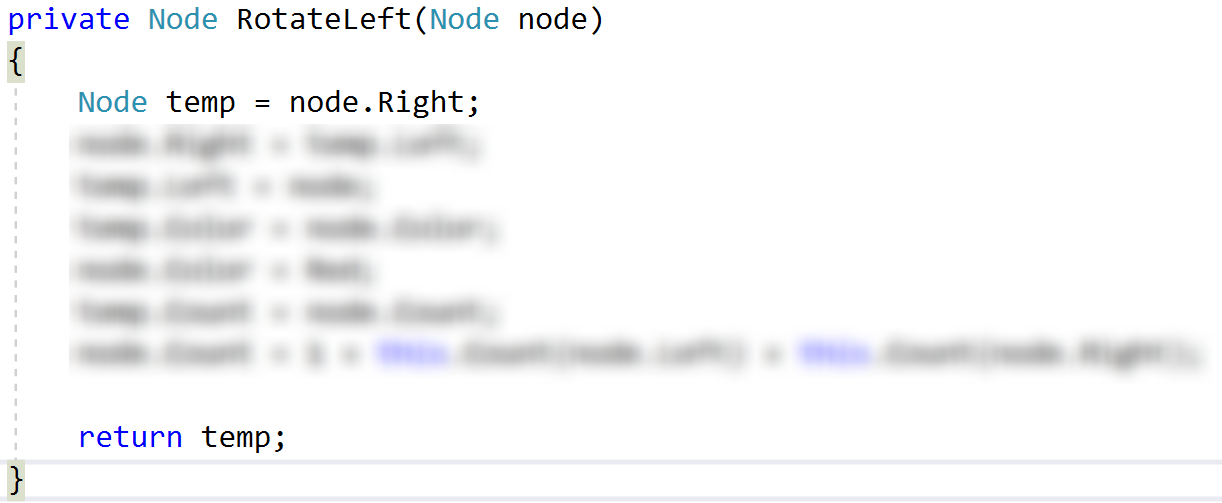


Now create a helper method that will check if a node is red:



## Left Rotation

Create a method that will accomplish the left rotation for a given node.



## Right Rotation

Create a method that will perform right rotation on a given node. The code is similar to the left rotation.

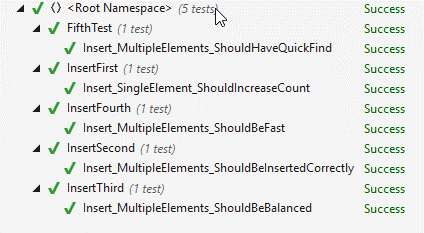
## Flip Colours

Implement a method that will make a node "**black**" and its children "**red**".

## Insert

Modify the existing insert() method. It should create new **red** node for every insert, **balance** the tree and **recolour** the nodes if needed.

## Run Unit Tests



That's it, you're ready to start implementing delete. :)