SOEN 331 - Assignment 2

Sam Assaf - 6150748
Jessica Falco - 6597882

Liskov’s Substitution Principle

Liskov's substitution principle states that all instances of a base class can be replaced by an instance of one of its subclasses.

In addition, the principle states that when dealing with inheritance, the following restrictions on contracts in the subclass apply:

- the preconditions cannot be strengthened,
- the postconditions cannot be weakened,
- the invariants of the base class must be preserved.

Indeed, in our assignment, no precondition is strengthened and no postcondition is weakened. In this assignment, subclasses do not override any methods and simply inherit all preconditions and postconditions from the **`BinTree`** superclass.

Invariants are indeed preserved. The invariant of the superclass is inherited and is enforced in all subclasses.

For the most part, Liskov’s principle holds in our assignment. There is, however, one violation of the principle:

As it holds, an instance of **`BinTree`** cannot be safely replaced with an instance of **`FullBinaryTree`** or its subclass **`PerfectBinaryTree`**. That is because the invariant of **`FullBinaryTree`** will most definitely be violated if only one of its child nodes has been set by using the inherited **`BinTree.setLeft(BinTree iBinTree)`** or **`BinTree.setRight(BinTree iBinTree)`** methods. However, an instance of type **`BinTree`** can be expected to safely call these two methods. Instances of **`FullBinaryTree`** and **`PerfectBinaryTree`** can only safely set nodes using the inherited **`BinTree.setLeftRight(BinTree a, BinTree b)`** method.