Assignment 2

Answer all questions – maximum 100 marks. You must score at least 50 to pass the assignment.

1. (15 marks) Design an algorithm for the following operations for a binary tree BT, and show the worst-case running times for each implementation:

   `preorderNext(x)`: return the node visited after node `x` in a pre-order traversal of BT.

   `postorderNext(x)`: return the node visited after node `x` in a post-order traversal of BT.

   `inorderNext(x)`: return the node visited after node `x` in an in-order traversal of BT.

2. (25 marks) Design a recursive linear-time algorithm that tests whether a binary tree satisfies the search tree order property at every node.

3. (20 marks) Exercise 8.2. Illustrate what happens when the sequence 1, 5, 2, 4, 3 is added to an empty `ScapegoatTree`, and show where the credits described in the proof of Lemma 8.3 go, and how they are used during this sequence of additions.

4. (20 marks) Implement a commonly used hash table in a program that handles collision using linear probing. Using (K mod 13) as the hash function, store the following elements in the table: {1, 5, 21, 26, 39, 14, 15, 16, 17, 18, 19, 20, 111, 145, 146}.

5. (20 marks) Exercise 6.7. Create a subclass of `BinaryTree` whose nodes have fields for storing preorder, post-order, and in-order numbers. Write methods `preOrderNumber()`, `inOrderNumber()`, and `postOrderNumbers()` that assign these numbers correctly. These methods should each run in $O(n)$ time.