

60-254 Data Structures and Algorithms – Winter 2017

Lab Assignment 5

Deadline: **Friday March 27/29, 2017** (15 minutes prior to the end of the lab)

Regulations: This assignment must be done individually. Any similarity found between your code/answers and another student's, or a carbon-copy of an answer found in the web will be considered cheating. You must implement your own binary search tree and its operations, and not use the built in ones from the programming language you are using.

Objective: The aim of this assignment is to obtain hands on experience in implementing, testing and understanding the binary search tree ADT and its operations.

Tasks:

1. Consider a binary search tree (BST). Assume that the keys are integers and no values (objects) other than the keys are stored in each node. By following the explanations and examples given in class, design an algorithm for operation `remove(k)`.
2. Using your favorite programming language, implement the BST ADT in a linked structure, along with the following operations: `find`, `insert` and `remove`. The `find` method should call `TreeSearch`, which you have to implement too.
3. Implement the `InOrder` traversal method for showing the keys of the BST in increasing order.
4. Insert 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 in the tree (in that order). Show all keys in the tree (for this and the following, use the `InOrder` traversal method you implemented in #3).
5. Perform `find(1)` and `find(15)` 100,000 times and record the CPU time taken by each of these.
6. Remove key 5. Show all keys in inorder traversal. Remove 15. Show all the keys in inorder traversal. Remove 1. Show all the keys in inorder traversal. Insert key 2. Show all keys in inorder traversal.
7. Create a new empty binary search tree. Insert 8, 4, 12, 2, 6, 10, 14, 1, 3, 5, 7, 9, 11, 13, 15, in that order. Show all keys of the tree in inorder traversal.
8. Perform `find(1)` and `find(15)` 100,000 times and record the CPU time taken by each of these.
9. Compare the CPU times of #5 and #8, and explain how they are related to the best and worst-case running times of the `find` operation.
10. Remove key 8 and show the keys of the resulting tree (again, in inorder traversal).

Submission:

1. Your assignment must be submitted during the lab session in the section you are registered. Any submission *on or 15 minutes* prior to the end of the lab session will **not** be accepted and a **zero** mark will be given. Late assignments will be given a **zero** mark. Submissions by email will **not** be accepted.
2. Provide the `remove` algorithm **in pseudocode** (e.g., written on paper or Word). Code in a programming language like Java, C/C++ or any other language will **not** be accepted.
3. Run all BST operations as indicated in items #4 to #10.
4. Explain how your BST and its operations work. This will be asked when the lab assignment is being submitted – marks will be deducted if not clear how it works.