C S 272/463 Introduction to data structures
Fall 2019
Lab 12: Binary Search Tree

1 Learning objectives

Objective 1 (binary search tree), Objective 2 (recursive thinking), Objective 3 (searching), Objective 5, Objective 6, Objective 7 in course syllabus.

2 Requirements

2.1 Tasks

- Implement the following methods for binary search tree (**BST.java**). You are given the definitions of facilitating classes and methods in `https://www.cs.nmsu.edu/~hcao/teaching/cs272/lab/code/BST.java`.

2.2 Detailed instructions for program design and implementation

1. (20 points) Insert a new element `e` into the binary search tree. NO duplicate values are allowed in the tree. When `e` exists in the tree, return false; Otherwise, insert `e` to the tree and return true.

   ```
   public boolean insert (int e)
   ```

2. (20 points) Remove a specified element from the binary search tree. When `e` exists in the tree and one instance is successfully removed, return true; Otherwise, return false.

   ```
   public boolean remove (int e)
   ```

3. (15 points) Design a recursive function to search whether an element exists in a binary search tree. If `e` exists, return the node that contains this element; Otherwise, return null.

   ```
   public BSTNode searchRecursion(int e)
   ```

   Please analyze its running time and get its complexity in Big-O. Analysis takes **5 points**.
4. (20 points) Design a non-recursive function to search whether an element exists in a binary search tree. If `e` exists, return the node that contains this element; Otherwise, return null.

   ```
   public BSTNode searchNonRecursion(int e)
   ```

5. (20 points) Design a recursive function to add up all the elements in this binary search tree. Return the summation of all the elements. (Hint: you can use any type of traversal.)

   ```
   public int sum()
   ```

6. (5 points) Design test cases to test your program **thoroughly**. Please put your test cases in the main function in **BST.java**.
   If your test cases cannot cover important conditions, points may be deducted.
   In the given BST.java, one set of test cases are given in `test1()` function.
   You are encouraged to let your code pass all the test cases. The results for running `test1()` is at `https://www.cs.nmsu.edu/~hcao/teaching/cs272/lab/code/BST_test1_output.txt`.

3 Note

- **Specifications** for all your classes and methods:
  Please properly explain (1) the functionality of the methods, (2) the parameters, (3) the return values, (4) the pre-conditions if there is any;
  Please use inline comments, meaningful variable names, indentation, formatting, and whitespace throughout your program to improve its readability.
- You can (but are not required to) design and implement other facilitating methods (E.g., other get and set methods, toString method) to finish the implementation of the required methods.

4 Submission

Submit through canvas a zipped file containing your java file(s) (not `.class` files).

5 Grading criteria

(1) The score allocation is already put in the questions.

(2) Please make sure that you test your code **thoroughly** by considering all possible test cases. Your code may be testd using more test cases.

(3) 5 points will be deducted if submitted files (including files types, file names, etc.) do not follow the instructions.

(4) At least 20 points will be deducted if your code cannot be run on CS servers.