C S 272/463 Introduction to Data Structures

Lab 10: Recursive Reasoning; Binary Search Tree

I. Learning objectives

Objectives 4, 6, 7, 8, 9.

II. Requirements

Q1. Please answer the following questions.

Please put your solutions to a word file. Let us define `gcd(a,b)` to be the greatest common divisor for two non-negative integers.

1a. (10 points) You are given the following code to calculate `gcd(a,b)`.

Please draw the recursive trace for a function call `gcd(8,6)`.

```
int gcd(int a, int b) {
      /* Pre: a>b i b>=0*/
      /* Post: gcd(a, b) = GCD(a, b) */
      if(b==0) return a;
      else return gcd(b, a%b);
}
```

1b. (20 points) Please reason the correctness of the above recursive function to calculate the value of `gcd(a,b)`.

Q2. Implement the following methods for binary search tree (BST.java). You are given the definitions of facilitating classes and methods in BST.java.

2a. (10 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)
```

2b. (10 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)
```

2c. (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.

2d. (10 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)
```

2e. (20 points) Given a BST and a positive number k, write a static recursive method to find the kth largest node in the BST.

2f. (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 provided in BST_test1_output.txt.

III. 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.

IV. Submission

Submit through canvas a zipped file containing your (1) java file(s) (not .class files).

V. Grading Criteria

1. The score allocation is beside the questions.
2. Please make sure that you test your code thoroughly by considering all possible test cases. Your code may be tested using more test cases.