[**Closed book and notes.**] Show all of your work clearly in the space provided. Be sure to **read each problem carefully**. Note that the exam is double sided.

1. (10 points) Explain concisely and precisely why a balanced binary search tree has an asymptotic time complexity of $O(\log(n))$ for its contains() method.

2. (8 points) What is the worst possible $O()$ time for contains() if the binary search tree is not balanced? Justify your answer.

3. (8 points) For a binary tree (not binary search tree), what is the $O()$ time for contains(). Justify your answer.

4. (20 points) Implement the height method for a binary search tree. Assume that an empty tree has a root==`null`. If you chose to implement this with recursion, include both the non-recursive and recursive methods.

5. (16 points) Using Java code, pseudocode, or an English description, explain how a stack could be used to solve the following problem:

> Given a .java file, verify that the file has a balanced number of openning and closing curly braces and parentheses. Consider the following examples:

```
{{()}}                  GOOD
{(})                    BAD
{ { { ( } } } )         BAD
{ () () {} { () } } GOOD
```

6. (9 points) What is the worst-case asymptotic time complexity of for the offer() and remove() methods for a Queue that is implemented with a LinkedList? Justify your answer.

7. (9 points) What is the worst-case asymptotic time complexity of for the offer() and remove() methods for a Queue that is implemented with a ArrayList? Justify your answer.

8. (20 points) Suppose we have a $2 \times n$ checkerboard (two rows and n columns). Write a recurrsive function **domino()** that, given n, will calculate the number of possible ways in which we can cover the board with 1×2 dominos. Just to get you started:

Number(1) = 1 

Number(2) = 2 

Number(3) = 3 

Additional work area for any problem. Clearly identify which problem is associated with the work on this page.