**Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

For each of the below questions, write a short sentence or two to express (in your own words) your answer. Keep the answers short, but use complete, correct, English sentences.

If it helps to clarify the questions, feel free to mentally prefix all the questions with the phrase "According to the video…"

1. After you’ve watched all the videos, please answer this question:  
   Of all the videos that you watched, if you could pick one video to be re-recorded by the instructor outside of class which would you choose? Why?  
   (Keep in mind the recording outside of class will omit any pauses from the instructor answering student questions, have less hemming and hawing, etc, and generally be more concise)

|  |
| --- |
| < Write your answer here > |

**VIDEO: BST: Basics, Find, Add Overviews**

1. Intuitively, how is a linked list set up?

|  |
| --- |
| < Write your answer here > |

1. What's the running time (and space) of finding something in a linked list (in Big Oh notation) **using a loop**?

|  |
| --- |
| < Etc. > |

1. What's the running time (and space) of finding something in a linked list (in Big Oh notation) **using a recursive approach**?

|  |
| --- |
|  |

1. For pretty much anything you want to do to an entire list (printing, finding a value, etc) what will the running be (in Big Oh notation)?

|  |
| --- |
|  |

1. How is a Binary Search Tree (aka BST) set up? At each node, what will all the nodes to the left of have in common? What will the nodes to the right all have in common?

|  |
| --- |
|  |

1. In a linked list we had a **front** reference. What will the BST call the reference to the first node?

|  |
| --- |
|  |

1. Explain why the value 15 was added where it was to the tree that contained {20, 10, 50}

|  |
| --- |
|  |

1. Is there any restriction on the things that you can put into a BST?

|  |
| --- |
|  |

1. Which nodes were examined while searching for the value 12 in the tree that contained (20, 10, 50, 15)? List them in the order that they were examined.

|  |
| --- |
|  |

1. How did you know that 12 was NOT present in the tree?

|  |
| --- |
|  |

1. Describe, in your own words, how to find a value in a binary search tree.  
   (This is demonstrated in the video multiple times, but isn't really summarized)

|  |
| --- |
|  |

1. In the C# source code, what stops us from looking to the left of the node containing the value 12?  
   (This is described verbally before it's defined in on-screen C# code)

|  |
| --- |
|  |

1. List out the C# source code that defines a BST node here:

|  |
| --- |
|  |

1. When searching the BST, how many nodes will be ignored each time the algorithm takes a step down the tree? (Assume that the BST is a 'best case' BST)

|  |
| --- |
|  |

1. What will the tree look like in the **best case** for a BST?

|  |
| --- |
|  |

1. What will the tree look like in the **worst case** for a BST?

|  |
| --- |
|  |

1. What is the running time (in Big Oh notation) of the **best case** of the BST?

|  |
| --- |
|  |

1. What is the running time (in Big Oh notation) of the **worst case** of the BST?

|  |
| --- |
|  |

1. Are BSTs ever used in production code? If not, what sort of trees are used instead (and why are these other trees used)?

|  |
| --- |
|  |

**VIDEO: BST Basic Class Definitions**

1. What will the BST (Binary Search Tree) object manage?

|  |
| --- |
|  |

1. What is the name of the one reference that a BST will have?  
   (This is for a BST, NOT a BSTNode)

|  |
| --- |
|  |

1. What are the three fields that are present within a Binary Search Tree Node (a BSTNode) object?

|  |
| --- |
|  |

**VIDEO: BigOh Applied To Space & Time**

1. Summarize how the Print() method is implemented in 2-3 concise yet clear sentences

|  |
| --- |
| < Etc. > |

1. What is the running time (in Big Oh notation) of the Print() method?

|  |
| --- |
|  |

1. List some possible functions that might describe how much space an algorithm might use:

|  |
| --- |
|  |

1. When figuring out how much space the Print() method requires, do we count the space occupied by the nodes in the list?

|  |
| --- |
|  |

1. How much space does the Print() method require (using the Big Oh notation)? Make sure that you're comfortable both with how you'd write this, and how you'd talk about this in English.

|  |
| --- |
|  |

1. How can we know that the private version of PrintR() will never be given a null reference as a parameter?

|  |
| --- |
|  |

1. What is the running time (in Big Oh notation) of the Print**R**() method?

|  |
| --- |
|  |

1. The number of times that the private PrintR is going to call itself is directly proportional to what?

|  |
| --- |
|  |

1. How much space does the PrintR() method require (using the Big Oh notation)?

|  |
| --- |
|  |

**VIDEO: Side-Topic: Comparing Two Implementations of Pow**

1. When you're considering solving a given problem either using a loop or using recursion, it might be tempting to always use a loop in order to avoid what sort of overhead from the recursion?

|  |
| --- |
|  |

1. In Pow1, if the **exponent** parameter is N, how many times will the loop be run?  
   (If you can, also list the running time using in Big Oh notation)

|  |
| --- |
|  |

1. In Pow2, how many times will the function recursively call itself (assume that the **exponent** parameter has the value N)?  
   (If you can, also list the running time using in Big Oh notation)

|  |
| --- |
|  |

**VIDEO: BST: Implementing Add**

1. When a new BST object is created, what is the value of it's **top** field?

|  |
| --- |
|  |

1. If top is null, then how many nodes are in the tree?

|  |
| --- |
|  |

1. When a new BSTNode object is created, what is the value of it's **left** and **right** fields?

|  |
| --- |
|  |

1. While walking down the tree, what comparison are you going to do at each node in the tree?

|  |
| --- |
|  |

1. Under what circumstance will you add the new node to the left of the current node?

|  |
| --- |
|  |

1. Under what circumstance will you **move the cur reference** to the left of the current node?

|  |
| --- |
|  |

1. The code that was written in the video, what is the third possible case that the function needs to handle?

|  |
| --- |
|  |

1. For the normal/basic strategy for dealing with duplicates, what should be done?

|  |
| --- |
|  |

1. If you did want to add duplicate values to the tree, what off-the-cuff, not-at-all-tested solution does the teacher speculate might work?  
   (And what other solution does the teacher demonstrate does NOT work?)

|  |
| --- |
|  |

**VIDEO: BST: Implementing Find, Using A Loop**

1. When looking for a value in a BST, what's the first thing that should be checked for?

|  |
| --- |
|  |

1. At the start of the loop, what node does **cur** refer to?

|  |
| --- |
|  |

1. Within the loop, what's the first thing we check for?

|  |
| --- |
|  |

1. If the function leaves the loop, what does the function do (and why)?

|  |
| --- |
|  |

1. Why do we NOT need to check for cur.left (or cur.right) being null inside the loop?

|  |
| --- |
|  |

**VIDEO: BST: Implementing Add, Recursively**

1. Why did the teacher decide to first assign the "find a value in a BST iteratively" exercise before this one? What general strategy is sometimes helpful in writing recursive logic?

|  |
| --- |
|  |

1. What's the first thing that the public FindR method should do?

|  |
| --- |
|  |

1. In the BSTNode.FindR method, what are the three basic possibilities that might occur?

|  |
| --- |
|  |

(There's a fair amount time spent responding to a question, which implements the recursive FindR on the BST class, instead of the BSTNode class)

1. Within the FindRINternal, if the value we're looking for is less than the current node's key, then what are the two possible actions we might take (and under what circumstances do we do each one)?

|  |
| --- |
|  |

(There's an in-depth demonstration of how FindRINternal works, then we go back to FindR\_Node)

1. In FindR\_Node, where does the "this" field/parameter come from?

|  |
| --- |
|  |

1. How is the C# source code of FindR\_Node different from that of FindRINternal?

|  |
| --- |
|  |