CS 1332 Exam 1 - SECTION A

Spring Semester: February 8, 2016

Name (print clearly including your first and last name): ____________________

Signature: ____________________

GT account username (gtg, gth, msmith3, etc): ____________________

CS1332 Section (A1,A2,A3,A4,A5,A6,B1,B2,B3,B4,B5,B6,GR2): ____________________

- Signing and/or taking this exam signifies you are aware of and in accordance with the **Academic Honor Code of Georgia Tech** and the **Georgia Tech Code of Conduct**.
- Notes, books, calculators, phones, laptops, smart watches, headphones, etc. are not allowed.
- Extra paper is not allowed. If you have exhausted all space on this test, talk with your instructor.
- Pens/pencils and erasers are allowed. Do not share.
- All code must be in Java.
- Efficiency matters. For example, if you code something that uses $O(n)$ time or worse when there is an obvious way to do it in $O(1)$ time, your solution may lose credit. If your code traverses the data 5 times when once would be sufficient, then this also is considered poor efficiency even though both are $O(n)$.
- Style standards such as (but not limited to) use of good variable names and proper indentation is always required. (Don't fret too much if your paper gets messy, use arrows or whatever it takes to make your answer clear when necessary.)
- Comments are not required unless a question explicitly asks for them.

Question	Points Possible	Points Received	Points Lost	Graded by TA Name (print)
Page 2.	20			
Page 3.	5			
Page 4.	25			
Page 5.	25			
Page 6.	25			
TOTAL	100			

[5] 1. Given a doubly-linked list implementation having a head reference and tail reference, give and explain the Big O of deleting the last node. You must explain your answer.

[5] 2. You have a queue backed by an array of size 3. You perform the following operations:

```
queue.enqueue("happy");
queue.enqueue("camper");
queue.enqueue("fun");
queue.dequeue();
```

Draw the array (including all the indices) showing where the remaining two strings will be located after these operations finish. (Draw null for any slot without data that is part of the queue.) Assume that the queue is implemented in the most efficient way possible.

[5] 3. You invented a new data structure implemented as a generic class called BucketOfStuff. A BucketOfStuff stores data of type E. You want users to be able to traverse a BucketOfStuff object easily by using a for-each-style for loop in their code. With this limited information, give the class header for the generic class BucketOfStuff showing the interface it must implement. Include import statement(s) if necessary.

[5] 4. You would like to use a `java.util.Queue` of Integer objects. Write the declaration and instantiation for this queue, using `java.util.LinkedList` as the backing structure for the queue. Include import statement(s) if necessary.

Name:____

[5] 5. Below, you are given the `Node` class for a doubly-linked list. Demonstrate your ability to do proper constructor chaining by crossing out and rewriting only the necessary parts of this code. (No need to rewrite all of it.)

```
public class Node<T> {
   private T data;
   private Node<T> next;
   private Node<T> previous;
   public Node(T data) {
      this.data = data;
      next = null;
      previous = null;
   }
   public Node(T data, Node<T> next, Node<T> previous) {
      this.data = data;
      this.next = next;
      this.previous = previous;
   }
}
```

[25] 6. Code for `SinglyLinkedList` and private inner class `Node` are provided below. Do not alter the code that is given. `thresholdCount` returns the number of items strictly greater than the data passed in. If there are no items larger than `data`, then `thresholdCount` is 0. Complete the instance method.

Special case:

If the method parameter is null, throw an exception (the type of which is your choice) whose message is "the data cannot be null". You may assume data in the linked list will never be null. Private instance members of a private inner class are accessible by the outer class so access Node fields directly as you do not have getters nor setters.

```
public class SinglyLinkedList<T extends Comparable<? super T>> {
    private Node<T> head;
    private int size;  // the number of nodes in the list
    // Unnecessary methods are not included.  Do not assume nor write any extra methods.

    public int thresholdCount(T data) {






    private class Node<T> {
        private T data;
        private Node<T> next;
    }
}
```

[25] 7. Using the classes from the previous problem, write an instance method called `toArray` that returns an array of the data found in the linked list. If the list is empty, then return a 0 length array object. Note that `SinglyLinkedList` has an instance variable `size`.

```
public Comparable[] toArray() {
```

[25] 8. You are required to use recursion for this problem. Code for `BST` and private inner class `BSTNode` are provided. Do not alter the code that is given. `recursiveSum` returns the sum of all data in the tree.

Recall that instance members of a private inner class are accessible by the outer class so access Node fields directly as you do not have getters nor setters.

The method must be recursive or call a recursive helper method. You may write private helper methods if you need them.

If the tree is empty, the sum should be 0. Complete the instance method using recursion.

```
public class BST {
    private BSTNode root;
    // Unnecessary methods are not included.  Do not assume nor write any extra methods.

    public int recursiveSum() {









    private class BSTNode {
        private int data;
        private Node left;
        private Node right;
    }
}
```

Name:____

This page is intentionally left blank. Use the space if you need it.

This page is intentionally left blank. Use the space if you need it.