 Your Name:_____

I commit to uphold the ideals of honor and integrity by refusing to betray the trust bestowed upon me as a member of the Georgia Tech community.

CS 1316 Exam 2
Summer 2009

Section/Problem	Points Earned	Points Possible
1. Terms & Concepts		62
2. Fill in the Blank		5
3. Trees		9
4. Code Comprehension		10
5. Turtle Graphics		8
6. Write Code: Copy A Queue		10
7. Write Code: AddLast		10
8. Write Code: Simple Plugin		15
Total Points:		129

 Your Name:_____

1. Terms & Concepts (62 points)

For each of the terms below, write 1 or 2 sentences defining the term and proving you understand what your definition means. You may include an example if you think it will help your explanation. Be concise and precise.

1. abstract (super) class

2. abstract data type

3. anonymous inner class

4. binary search tree

5. circular linked list

6. class

7. dynamic data structure

8. final static variable

9. graph (directed, undirected, acyclic)

10. in-order traversal (of a tree)

11. inheritance

12. interface

13. layout manager

Your Name:_____

14.leaf node

15.LIFO

16.linked list

17.object

18.private (keyword)

19.queue

20.recursion

21.refactor(ing) code

22.reference

23.scene graph

24.spanning tree

25.static field (variable)

26.static method

27.superclass

 Your Name:_____

28.this (keyword)

29.traverse

30.user interface events

31.void (keyword)

2. Fill in the Blank (5 points)

In Java, logical **and** is written using the _____ symbol, and logical **or** is written using the _____ symbol.

Assume that the Student class is a subclass of the Person class, and the Person class is a subclass of the Human class. A variable that is defined to be of type Person can refer to (hold) an object of type _____ or type _____ but a variable defined to be of type Student can only refer to an object of type _____.

3. Trees (9 points)

a. If the in-order traversal of the binary tree T is: A D B G C F E, draw the tree:

b. If the pre-order traversal of the binary tree T is: A D B G C F E, draw the tree:

c. If the post-order traversal of the binary Tree is A D B G C F E, draw the tree:

 Your Name:_____

4. Code Comprehension (10 points):

Consider the method **`mystery`** below that manipulates linked lists of integers. What does this code do? (hint: draw a linked list of integers, then apply the code to the example list. IntNode is a linked list node similar to AgentNode or LLNode, which contains integers.)

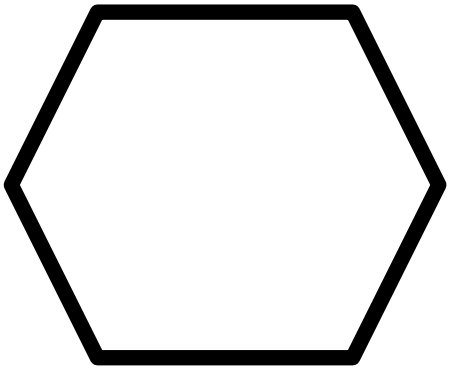
If q references the list, what is returned by `mystery(q, null)`?

```
public IntNode mystery(IntNode aList, IntNode upTillNow ) {
   if (aList == null) {
       return upTillNow;
   } else {
       IntNode temp = aList.getNext();
       aList.setNext( upTillNow);
       return( mystery( temp, aList) );
   }
}
```

5. Turtle Graphics (8 points)

The following code creates a turtle, add the code to draw a hexagon like as the one shown to the right. (You may draw your hexagon in any orientation, and with any length sides you want. You may start drawing the hexagon as soon as the turtle is created, you do not need to do any movement or rotation of the turtle to "get it to the hexagon" before you start drawing.)

```
Turtle t = new Turtle(new World());
```



 Your Name:_____

6. Copy a queue (10 points):

Write code that accepts a queue as a parameter, creates a new empty queue, fills the new empty queue with the same contents as the original queue, and returns the new empty queue. Note: Your function must leave the original queue in the same state that it was in when you received it! You may assume the Queue supports the standard enqueue(item), dequeue(), peek(), size(), and isEmpty() methods.
We have given you the shell of the method, fill in the missing statements so that it works.

```
public static Queue<String> copyQ( Queue<String>  original) {
  // Create a new queue:
   Queue<String> newQ = new Queue<String>();
   int numElements = original.size();
   int doneSoFar = 0;
   while( doneSoFar < numElements) {
     // Add the missing statements here:









    } // end while
    return(newQ);
} // end
```

 Your Name:_____

7. Write Code – Add Last (10 points)

Write a public method addLast(String data, StringNode first) that will create a new StringNode (containing the String "data") and add it to the end of the linked list (the "first" variable contains a reference to the first node in the linked list). You may assume that the first node will always exist (e.g. first will never be null). The class definition for StringNode is below:

```
public class StringNode {

    private myString;
    private StringNode next;

    public StringNode( String d) {
            myString = d;
            next = null;
    }

    public StringNode getNext() { return(next); }
    public void setNext( StringNode n) { next = n; }
    public void setData( String d) { myString = d; }
    public String getData() { return( myString );   }

 } // end class StringNode
```

 Your Name:_____

8. Create an Object: SimplePlugin (15 points)

Below is an interface for writing a plugin for an imaginary program. Your task is to write a class called SimplePlugin (the plugins' name is also "SimplePlugin") that implements PluginDesign but also contains two additional methods: one called "store" and one called "match":

- The "store" method accepts a generic object (of type "T") as it's only parameter and sets its value to a private generic class variable "myData" defined in the SimplePlugin class.
- The "match" method accepts a generic object (of type "T") as it's only parameter and compares it to the

private class variable "myData". It returns a true (Boolean) if they match (contain the same data, even if they are not the same object) and false (Boolean) otherwise.

You may assume that the generic object of type "T" has an equals() method. Be sure to consider the case when your SimplePlugin does not have an object stored.

```
public interface PluginDesign {

  //Return the name of your plugin

  public String getName();




  //Clears all class variables in the plugin

  public void resetVariables();

}
```