CS 1316 - Exam 3 - Spring 2010

Name: _____

Grading TA: _____ **Section:** _____

INTEGRITY: By taking this exam, you pledge that this is your work and you have neither given nor received inappropriate help during the taking of this exam in compliance with the Academic Honor Code of Georgia Tech. Do NOT sign nor take this exam if you do not agree with the honor code.

DEVICES: If your cell phone, pager, PDA, beeper, iPod, or similar item goes off during the exam, you will lose 10 points on this exam. Turn all such devices off and put them away now. You cannot have them on your desk.

ACADEMIC MISCONDUCT will not be tolerated. You are to uphold the honor and integrity bestowed upon you by the Georgia Institute of Technology. Penalties for misconduct will be a zero on this exam, an F grade in the course, and/or other disciplinary action.

- Keep your eyes on your own paper.
- Do your best to prevent anyone else from seeing your work.
- Do NOT communicate with anyone other than a proctor for ANY reason in ANY language in ANY manner.
- Do NOT share ANYTHING during the exam. (This includes no sharing of pencils, paper, erasers).
- Follow directions given by the proctor(s).
- Stop all writing when told to stop. Failure to stop writing on this exam when told to do so is academic misconduct.
- Do not use notes, books, calculators, etc during the exam.

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.

Signature: _____

Problem	Points Earned	Points Possible	Grader Initials
1. Vocabulary		12	
2. Tree Travesals		6	
3. GUI Code Reading		6	
4. Code Reading		10	
5. Try/Catch		4	
6. Small Questions		4	
7. Linked List		10	
TOTAL:		52	

Exam Percentage: _____ / 52 = _____ %

1. Vocabulary (12 points)

For each of the following words, write a 1-2 sentence definition of the word as used in this class. Your definition should be concise and to the point, while demonstrating that you know what the term means.

a) binary search tree – A Binary tree that is sorted so that items within it can be found quickly. Elements that are greater than or equal to a node are to the right of the node, elements that are smaller are to the left.

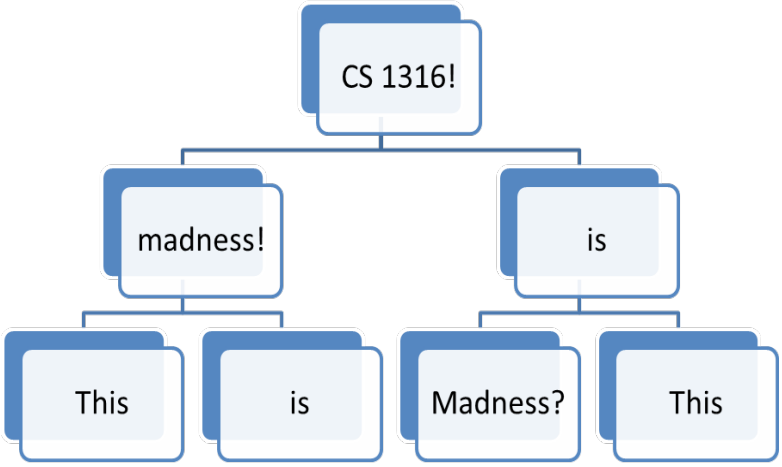
b) recursive traversal (of a list) – A Recursive traversal is when your code visits each node of the linked list by using the same method which repeatedly calls itself.

c) simulation – Simulations are representations of the world (models) that are executed or made to behave like things in the world.

d) user interface events – Event that is dispatched when the user does something to a GUI, like clicking a button, or moving the mouse.

2. Tree Traversals (6 points)

Examine the following tree. Write the correct strings stored in the nodes for each of the following traversal methods.



Pre-Order: CS1316!, madness!, This, is, is, Madness?, This

In-Order: This, madness!, is, CS1316!, Madness?, is, This

Post-Order: This, is, madness!, Madness?, This, is, CS1316!

3. GUI Code Reading (6 points)
You are given the following lines of code:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class MysteryGUI extends JFrame implements ActionListener{
  JButton jb;
  JLabel jl;

  public MysteryGUI() {
    this.getContentPane().setLayout(new BorderLayout());

    JPanel panel1 = new JPanel();
    jb = new JButton("Click me!");
    panel1.add(jb);
    this.getContentPane().add(panel1, BorderLayout.NORTH);
    jb.addActionListener(this);


    JPanel panel2 = new JPanel();
    jl = new JLabel("Hello");
    panel2.add(jl);
    this.getContentPane().add(panel2, BorderLayout.SOUTH);

    pack();
    setVisible(true);
  }

  public void actionPerformed (ActionEvent e) {

    if (e.getSource() == jb) {
      jl.setText ("Good bye");
    }
  }
}
```

1. Draw the GUI the code above produces initially (4 points):

Grading:
- 1 point: JFrame
- 1 point: "Click me!" JButton
- 1 point: "Hello" JLabel
- 1 point: correct order of button / label.

2. What happens when we click the JButton jb (2 points)?

Clicking the button changes the JLabel's text from "Hello" to "Good bye".

4. Code Reading(10 points)

Examine the code below. For each commented position (1-6) determine whether the access operation is legal or not. If it is illegal, write "ERROR" and briefly explain why it is illegal. If it is legal, write "LEGAL". In determining your answers, ONLY consider the position in question; that is, for each part, assume that all of the other code works correctly, even if you believe it would not in reality.

```
public class Landlord{
   private String name;
   public Building building;
   private static double income = 0.0;

   public Landlord(String name, Building building){
      this.name = name;
      this.building = building;
   } //end constructor

   public int headCount(){
      int count = building.numTenants; // Illegal, numTenants has private access in Building
      return count;
   } //end headCount method

   private double getIncome(double rate){
      return building.income(rate); // Illegal, income method has private access in Building
   } //end getIncome method

   public static void main(String[] args){
      Building b = new Building("Woodruff", null);
      Landlord bob = new Landlord("Bob", b);
      double income = getIncome(500.0); // Illegal, non-static method cannot be referenced from a static context; AND/OR you would
need to call the method on an instance of the Landlord class
      if(income < 50000){
         Building.demolish(); // Legal
      }
      System.out.println("Income is " + bob.getIncome(500.0));
   } //end main method
}//end Landlord class

public class Building{
   private String address;
   private static int numTenants = 0;

   public Building(String a, Landlord landlord){
      address = a;
      landlord.building = this; // Legal
   } //end constructor

   public static void demolish(){
      address = "Building no longer exists"; // Illegal, non-static variable (address) cannot be referenced from a static context
   } //end demolish method

   private double income(double rate){
      return numTenants * rate;
   } // end income method
}//end Building class
```

Grading:

1 point for each correct legal answer
2 points for correct "illegal" answers, one for label, one for explanation

5. Try/Catch (4 points)

Examine the following code:

```
try {

      Picture p = null;
      p.getPixels();
      System.out.println("Open the pod bay doors, HAL.");

    }
catch (Exception NullPointerException) {

      System.out.println("I'm sorry, Dave.");
      System.out.println("I'm afraid I can't do that.");
}
```

a. Circle the line that would throw an exception.

p.getPixels(); throws a NullPointerException (2pts)

b. Write the text that the code would output to the interactions pane:

>> I'm sorry, Dave.
>> I'm afraid I can't do that.
(2pts for correction output, no points if they have "Open the pod bay doors, HAL" written.)

6. Small Questions (4 pts)

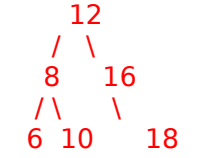
FIFO is an acronym for: (1 pt)

a) First In Final Out
b) Final In First Out
c) First In First Out
d) Final In Final Out

Suppose there is a binary tree T with 14 nodes. What is the minimum possible height of T? (1 pt)

a) 0 b) 3
c) 4 d) 5
e) 1 f) 2

Draw a binary search tree where the in-order traversal of nodes produces: 6, 8, 10, 12, 16, 18 (2 pts)



Note that other (correct) binary search trees are possible (for example, the 16 and 18 can be swaped)!

7. Linked List (10 points)

Write a static method named goTo that takes in an integer location and a pointer to an LLNode. It should return the (String) data in the appropriate node. Assume that all node contents/data are strings and that the LLNode class has a method called getData() that returns the String within a node. Also assume that LLNodes have a getNext() method that returns a reference to the next node in the list (or null if you are at the end of the list). You should start counting the elements in your list at 0. You **must check for invalid inputs**--negative inputs or inputs beyond the length of the list. If the inputs are not valid return null ***and*** print an appropriate message to the interactions pane.

```
public static String goTo (int location, LLNode head) {

 if(head==null) {
  System.out.println("Invalid input.");
  return null;
 }

 LLNode current = head;
 int position = 0;

 while(current.getNext()!=null && position<location) {
  current = current.getNext();
  position++;
 }

 if(position==location) {
  return current.getData();
 }

 System.out.println("Invalid input.");
 return null;
}
```

Grading:

- 2 points: correct method header
- 2 points: using println and returning null on invalid location input (don't worry about null LLNode!)
- 2 points: traversing the LinkedList
- 2 points: stopping the traversal before the end (avoiding null pointer exception)
- 2 points: returning the String value from the correct node