# CS 1316 Spring 2011 Midterm Exam #2

# Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Problem | Points | Lost | Gained | Running Total |
| 1a. |  15 |  |  |  |
| 1b. |  15 |  |  |  |
| 1c. |  35 |  |  |  |
| 2a. | 15 |  |  |  |
| 2b. | 20 |  |  |  |
| Total | 100 |  |  |  |

#1. Imagine that you have a collection of nodes that have links like the diagream below. Let’s say that **node1** is pointing to an instance of class **SongNodeBranch** and that all other nodes are instances of class **SongNodeDoubly**:

You should assume that the accessors and manipulators **getPrevious, getNext, setPrevious, setNext,** and **getFirstChild and setFirstChild** all exist. You may not use any other methods (e.g., **remove, add, last, insertAfter**).

1. Imagine that you have a new node referenced by variable **newNode** (containing, say, “riff5”). What code would you use to insert **newNode** between the nodes containing “riff3” and “riff5”? (Don’t worry about any casting here. We’re not so worried about syntax details.) You ***only*** have the variable reference to **node1** to start from, but you can assume that you know the structure above.

1. Now imagine that you have a new node referenced by variable **newNode2** (containing, say, “pattern1”). Write the program code to insert **newNode2** as the child of the branch **node1**, *before* the node containing “riff3.” Make sure all **next** and **previous** links are correct.

1. Write a new method for the class **SongNodeBranch** called **copyChildren**. **copyChildren** can assume that all nodes linked to **firstChild** are instances of **SongNodeDoubly**. You can assume that instances of **SongNodeDoubly** know how to create a **copy** of themselves. The method **copyChildren** should create a copy of each the **SongNodeDoubly** nodes attached to the receiver object (e.g., **node1.copyChildren()**), with all **next** and **previous** links set up correctly.

For reference:

Recall that repeatNextInserting() looks like this :

 public void repeatNextInserting(SongNode nextOne, int count){

 SongNode current = this; // Start from here

 SongNode copy; // Where we keep the current copy

 for (int i=1; i <= count; i++)

 {

 copy = nextOne.copyNode(); // Make a copy

 current.insertAfter(copy); // INSERT after current

 current = copy; // Now append to copy

 }

 }

And HBranch’s drawWith() looks like this:

 /\*\*

 \* Ask all our children to draw,

 \* then tell the next element to draw

 \* @param turtle Turtle to draw with

 \*/

 public void drawWith(Turtle turtle) {

 // start with the first child

 DrawableNode current = this.getFirstChild();

 // Have my children draw

 while (current != null) {

 current.drawWith(turtle);

 turtle.moveTo(turtle.getXPos()+gap,turtle.getYPos());

 current = current.getNext();

 }

 // Have my next draw

 if (this.getNext() != null) {

 current = this.getNext();

 current.drawWith(turtle);

 }

 }

}

#2. Imagine that you are working with three classes that have these relationships and methods:

Class **Primate**

Knows how to **stand(), step(), eat(),** and **sleep().**

Class **Mammal** extends **Primate**

Knows how to **brushFur(), carryChildren()**

Class **Beaver** extends **Mammal**

Knows how to **swim(), slapTail(), catchFish()**

subclass

subclass

A. Complete the following method in class **Beaver**—show us the Java code, and tell us in a comment (like below) the class where each method is defined. (Each call to **step()** takes one step.)

/\*\* When a Beaver wakes up, it stands, takes three steps to

its hole, then swims, catches a fish, and then eats. \*\*/

public void wakeInMorning() {

 this.stand(); // From class Mammal

// Complete the rest…

}

B. In the middle of some other method, there comes this section of code describing the mating dance of the Beaver. The challenge is that the variable **male** is of class **Primate**. Fill in the missing code.

public void matingDance() {

Primate male = new Beaver();

/\*\* The male Beaver takes three steps then 5 times in a row,

takes a step, brushes his fur, slaps his tail,

then takes another step. \*\*/

for (int i = 0; i < 3; i++){

 male.step();}

//That was the three steps – you do the rest.

//PLEASE NOTE: Do think about casting!