Recitation Guide for Monday June 4, 2007

1. **Exam 1**
   a. Comments? Complaints? Too easy? Too hard? Too short? Too long?
2. **Quiz 1**
   a. Hand back quizzes and go over solutions (if they want you to)
      i. Average 90% (including 0s)
      ii. Quiz solutions available on TA coweb
      iii. Remind them that they should address any grading questions with you first (probably after class). If you do decide to award points back, be sure to mark that on his/her quiz and sign it. If you do not decide to award points back, the student can still appeal to the head TA then the professor.
3. **Sounds**
   a. MySoundCollage.java – Good example of how to use declare Sounds, use append, and play the appended Sounds.
   b. blockingPlay() versus play() (Both are found in SimpleSound.java)
      i. blockingPlay() – will stop everything else that is going on and only play the Sound
      ii. play() – will play the Sound, probably over something that is already happening
   c. The Array data structure (Source: manipulating-sounds.ppt pp.26)
      i. Pros
         1. Easy to understand
         2. Generally very efficient (in the age-old time versus space argument sense)
         3. Very fast to access a specific (nth) element
         4. Static – always the same length from moment of declaration
      ii. Cons
         1. Hard to insert and delete in the middle
         2. Static – always the same length from moment of declaration
   d. The evil Sound methods
      i. Explain the logic behind these methods from Sound:
         1. insertAfter(Sound, int) – manipulating-sounds.ppt pp.11-19
         2. delete(int, int) – manipulating-sounds.ppt pp.20-25

4. **Creating Music**
   a. SongNode – a LinkedList of nodes containing SongPhrases
      i. LinkedLists – a dynamic data structure composed of nodes that contain data and a pointer to the next node.
         1. Pros
            a. Easier to insert and delete than an Array
            b. Dynamic – can grow to any length
         2. Cons

a. More complex to traverse
b. Slower to access a specific element

b. SongPhrase – the data contained within a SongNode
    i. Static methods – class methods; can be used without declaring an instance of the class.
        1. They have already seen static methods from FileChooser and Math
c. Declaring a linked list of SongNodes
    i.
```
import jm.JMC;
SongNode node1 = new SongNode();
node1.setPhrase(SongPhrase.riff1());
SongNode node2 = new SongNode();
node2.setPhrase(SongPhrase.riff2());
SongNode node3 = new SongNode();
node3.setPhrase(SongPhrase.riff1());
node1.setNext(node2);
node2.setNext(node3);
node1.showFromMeOn(JMC.SAX);
```
    ii. Explain the above code and make sure they understand what is happening.
    iii. Try adding more nodes and taking other nodes out. Explain what happens to the linked list and contrast this to what happens in an array.
d. Weave, insertNext, repeatNext, repeatNextInserting, insertAfter
    i. Explain weave and use the weave powerpoint to help you http://coweb.cc.gatech.edu/cs1316/uploads/336/dissectingWeave_DawnFinney.ppt
    ii. repeatNext – inserts copies of the desired node after the node it is called on. Does not preserve the rest list.
    iii. repeatNextInserting– inserts copies of the desired node after the node it is called on. Does preserve the rest list.
    iv. insertAfter – just inserts a node after the node it is called on and preserves the rest of list.
e. How do we play the music and get rid of the notes display?
    i. JMusic API – http://jmusic.ci.qut.edu.au/jmDocumentation/index.html
    ii. `Play.midi(score,false)` – will play a score in the background (`false` keeps it from quitting Java after playing).
    iii. `Play.waitCycle(score)` will block anything else from happening until the score is done playing – essentially, letting you block like `blockingPlay`.
    iv. The modified SongNode I wrote already includes two methods that just plays without the note display called:
```
public void playFromMeOn(int instrument)
public void playFromMeOn(String songName, double tempo, int
timeSignatureTop, int timeSignatureBottom, int instrument)
```