Practice Problems 3/21/2016

Part 1: Lists

HINT: Get out some pen and paper!

Given a class representing a singly linked list with a firstNode field, a numberOfEntries field, and a getNodeAt(int givenPosition) method, implement the following methods. The Node class **only** has a setNextNode method. Refer to the lecture notes for examples of other method implementations:

1.
```
/**
* Remove the middle element of the list. If the middle is between
* two nodes, remove the node to the right.
* @return the node removed
* @throws EmptyListException if the list is empty.
*/
public Node<T> removeMiddle() {

}
```

2.
```
/**
* Add the newNode object to the middle of the list. If the middle
* is not between two nodes, add the node to the right.
* @param newNode is the node to add
*/
public void addToMiddle(Node<T> newNode) {

}
```

3.
```
/**
* Reverse the order of the list.
*/
public void reverseList() {

}
```

4.
```
/**
* Swap the positions of node1 and node2
* @Precondition numberOfEntries == 2 || bounded by other nodes
* @param pos1 is the position of the first node
* @param pos2 is the position of the second node
*/
public void swapNodes(int pos1, int pos2) {

}
```

Part 2: Debugging

Ensure that you know how to do the following debugging actions (helpful for lab this week):

1. Set a break point.

2. Debug a test class to get to the breakpoint.

3. View the variables and their values at that break point.

4. Step to the next line after the break point.

5. Step into a method call.