Theory:
+3 Q1 algorithm
+2 Q1 running time
+1 Q2 algorithm
- your algorithm will get stuck in an infinite loop if they are not the same circular linked list. The definition of a circular linked list is that it's a circle so there will never be an itr1.next == null. Also, if you do find that it1.next == tHead, you can just return true because if they are the same node then it is guaranteed that the rest of the nodes will be the same since each node can only point to one next node.
+2 Q2 running time
+5 Q3 algorithm
+3 Q3 using a stack
+2 Q3 running time
- very nice. slight bug in your code since you never got the string postfix from anywhere, it would have been better to write your code as a method that takes in a string argument perhaps, but overall it works.
+1 Q4 path A B H P Q R J K (O I N M G L C)
- followed incorrect pattern, you should be pushing in order West, South, East, North as it said in the algorithm, which means that when you are popping off the next node to explore you would see H and all of its following pushes before you get to C.
+1.5 Q5a running time
+1 Q5a explanation
+2.5 Q5b disadvantage
- true, but what else are you losing about expected Big-O of linked list operations if you change the implementation to get O(1) for the get(n) operation?

Programming:
Compilation (3.5 points):
+3.5 for compiling
Functionality (35 points):
+4 Stack push method
+3 Stack pop method
+1 for EmptyStackException
+4 Stack size method
+4 Stack isEmpty
+10 for palindrome checker general case working
+5 for not caring about punctuation, spacing, or capitalization
+5 for palindrome checker correctly calling methods of MyStack
+4 for tester class
Design (10.5 points):
+1 for Stack size in constant time
+1 for Stack isEmpty in constant time
+1 for Stack push in constant time
+1 for Stack pop in constant time
+2 for throwing appropriate exceptions
+2 for method design of palindrome code
+1.5 for general design
- in your Stack class you were really supposed to implement it using a Linked List, not extending the Linked List class. You should have just had an instance variable that was a linked list and you would have called methods on it from within your stack methods.
Style (7 points):
+2 javadocs for all methods
+2 javadocs for all classes
- I think you edited your code after you wrote some of the javadocs so the comments were a bit confusing. For example in your javadoc for the MyStack class you mention something about using Strings. Make sure your comments are consistent if you update your code!
+1 good comments otherwise
+2 good variable names

Testing (14 points):
+1.5 testing Stack push
+1.5 testing Stack pop
+1.5 testing Stack size
+1.5 testing Stack isEmpty
+4 testing palindrome
- nice job testing, I like the bats palindrome example, hadn't heard that one before haha! Although it may be easier to run your code coverage with all of your tests in one test file, it is probably better style to write separate test classes, one for each class, to keep things more organized and allow you to be sure that you are testing all cases that you want to test.
+4 for 90% code coverage report for all classes

Nice job! Total 93/100