CIT 594 – Homework 2

Due – Feb 17, 2016 at 12pm

Part 1 – Theory (30 points)

Please do the following problems:

1. Given two linked lists L1 and L2 (each of size n) sorted in an ascending order, describe an algorithm to create a new linked list L that contains all the nodes from L1 and L2 ensuring that L is still sorted. What is the running time of your algorithm? (5 points)
2. Suppose you are given two circular linked lists, S and T. Describe an algorithm for determining if S and T are really the same list of nodes, but with different starting points (head node). What is the running time of your algorithm? (5 points)
3. Postfix notation is an unambiguous way of writing an arithmetic expression without parentheses. It is defined so that if "(exp1) op (exp2)" is a normal fully parenthesized expression whose operation is op, then the postfix version of this is "pexp1 pexp2 op", where pexp1 is the postfix version of exp1 and pexp2 is the postfix version of exp2. The postfix version of a single number or variable is just that number or variable. So, for example, the postfix version of "((5 + 2) * (8 – 3))/4" is "5 2 + 8 3 – * 4 /". Describe an algorithm for evaluating an expression in postfix notation. What is the running time of your algorithm? (10 points)
4. Exercise R-15.21 from Big Java (5 points)
5. Exercise R-16.16 from Big Java (5 points)

Part 2 – Programming (70 points)

Palindromes: A palindrome is a sequence of characters that can be read the same way forwards or backwards. E.g., "kayak", "racecar", "Doc, note: I dissent. A fast never prevents a fatness. I diet on cod." are all palindromes. For more details on Palindromes, see http://en.wikipedia.org/wiki/Palindrome.

From the Wikipedia page, "According to Guiness World Records, the Finnish word saippuakivikauppias (soapstone vendor), a 19 letter word, is claimed to be the world's longest palindromic word in everyday use. A meaningful derivative from it is saippuakalasalakauppias (soapfish bootlegger). An even longer effort is saippuakuppinippukauppias (soap dish wholesale vendor). Almost equally long is the Estonian word kuulilennuteetunneliluuk (the hatch a bullet flies out of when exiting a tunnel)."

For this assignment, you will detect if the word/phrase input by the user is a palindrome or not using Stacks.

The first task will be to implement a Stack using Linked Lists. You can only use the linked list example shown in class and posted to canvas – you will need the files `MyGenericNode.java` and `MyGenericLinkedList.java`. You can modify these as needed.

Create a class called `MyStack` that supports the following methods – `push`, `pop`, `size`, `isEmpty`. (You also might want to display the contents of your stack for debugging purposes – even though it's not a valid stack operation.)

Next, create a class called `PalindromeChecker`. The class should use the `MyStack` class to check whether the input is a palindrome or not. Your code does need to account correctly for whitespace, upper- and lower- case letters, and all punctuation. E.g., if the user enters "A man, A plan, A canal, Panama", your program should say it's a palindrome.

Finally, create a tester class that has the `main` method, which asks the user for input and says whether the input is a palindrome or not using the `PalindromeChecker` class.

Unit Testing and Code Coverage

For this homework, you will perform unit testing using JUnit and EclEmma.

Please follow these steps for testing your homework:

1. Install EclEmma for Eclipse (http://eclemma.org/installation.html).
2. Set it up to track coverage for your homework.
3. The goal is to perform unit testing for the following classes:
    a. `MyGenericNode.java`
    b. `MyGenericLinkedList.java`
    c. `MyStack.java`
    d. `PalindromeChecker.java`
4. You can ignore the tester class for unit testing and code coverage.
5. Ensure that you have at least 90% code coverage for your entire project as tracked by EclEmma.
6. Generate the final report of coverage and submit this file – Please save this file and submit along with the deliverables. (Right click on in the Coverage Tab in Eclipse -> Export Session -> Java -> Coverage Report -> Pick the most recent session; HTML Format; Destination -> Finish).

Grading Criteria

5% for compilation – If your code compiles, you get full credit. If not, you get a 0.
50% for functionality – Does the code work as required? Does it crash while running? Are there bugs? …
15% for design – Is your code well designed? Does it handle errors well? …
10% for style – Do you have good comments in the code? Are your variables named appropriately? ...
20% for unit testing and code coverage – Are your tests well written? Do you have the required coverage? …

Programming – General Comments

Here are some guidelines with respect to programming style.

Please use Javadoc-style comments.

For things like naming conventions, please see Appendix I (Page A-79) of the Horstmann book. You can also install the Checkstyle plugin (http://eclipse-cs.sourceforge.net/) in Eclipse, which will automatically warn you about style violations.

Submission Instructions

We recommend submitting the theory part electronically also. However, you can turn in a physical copy at the start of class, if you prefer. Please **do not** print out the Java source.

In addition to the theory writeup, you should also submit a text file titled readme.txt. That is, write in plain English, instructions for using your software, explanations for how and why you chose to design your code the way you did. The readme.txt file is also an opportunity for you to get partial credit when certain requirements of the assignment are not met. Think of the readme as a combination of instructions for the user and a chance for you to get partial credit.

Please create a folder called YOUR_PENNKEY. Places all your files inside this – theory writeup, the Java files, the readme.txt file. Zip up this folder. It will thus be called YOUR_PENNKEY.zip. So, e.g., my homework submission would be swapneel.zip. Please submit this zip file via canvas.