CT2109 – OOP: Data Structures & Algorithms

Assignment 2

Linked List Implementation of a Queue

Task 1 (20 marks):

In lecture 1 we examined the array implementations of Stacks and Queues. For this assignment, you will write a **linked-list version of a Queue class**, which implements the same Queue interface that we previously implemented using an array. Use the Singly Linked List ADT from lecture 2 as the basis for your implementation.

Note that Queue.java (lecture 1) as well as Node.java & SLinkedList.java (lecture 2) must be used without any modifications from the versions provided on Blackboard, except for a new method gotoTail() in SLinkedList.java, which you will find useful for your implementation.

Test your implementation fully and submit the test code and results (e.g. meaningful screenshots) as well as the code for your Linked List Queue. As part of your testing, I recommend that you enqueue() the same items onto both a Linked List Queue and the Array Queue, and verify that you get the same results when you retrieve items from both.

Some hints:

- Your main goal is to write a new class, LLQueue.java, similar in operation to ArrayQueue.java, but that uses the singly linked list as its internal storage instead of the array.
- Examine every method in ArrayQueue.java and see how it needs to be modified. For example:
    - The Constructor will create an empty linked list, not an array
    - You don't need to store its capacity and there is no fixed limit to the linked list storage
    - You will have to decide how the enqueue() and dequeue() methods should operate relative to the head and tail of the underlying linked list. Is it easier to insert or delete objects at one end or the other?

Please note:

- This is an individual submission.
- Indicate clearly with comments where your modifications to the original code are (e.g. gotoTail() in SLinkedList.java).
- Additional comments at the beginning of each non-trivial method must clearly explain how your method or the underlying algorithm works.
  These are complemented by useful comments throughout your code.
- Note that you are likely to score higher, the better your code is commented.

- Plagiarism and cheating will result in zero points. In particular, if we think that any two final submissions are too similar, we reserve the right to give each submission zero points.
- Please return your submission via Blackboard by Friday, February 10th 2017.
- Your final submission should consist of:
    - each of the .java files you developed (not zipped), e.g. don't submit Queue.java and Node.java.
    - for each .java file you developed, a PDF file containing the code of the .java file.
    - meaningful screenshots of your test results.