COMP-2540 Data Structures and Algorithms – Winter 2024
Lab Assignment 2

Deadline: The assignment must be submitted in the lab section students are registered in, on **February 5 or 7, 2024** (15 minutes prior to the end of the lab)

Regulations: This assignment must be done individually. Any similarity between your code/answers and another student's, or a carbon-copy of an answer found on the Web will be considered plagiarism. None of the built-in classes/methods/functions of Java/C/C++/Python or any other language can be used for the ADTs. You must implement **your own** ADTs and their operations.

Objective: The aim of this assignment is to obtain hands on experience in implementing ADTs, and understanding the theoretical concepts seen in class on algorithm design and analysis. At the end of this assignment, students will know how to implement Stacks, Queues and Linked Lists, how to analyze the complexity of their operations, and work on relevant applications.

Tasks:

1. [3 marks] Using your favorite programming language, implement a singly linked list with the following operations: addFirst, removeFirst, addLast, removeLast, getFirst, getLast, size.
2. [3 marks] Implement a stack on the singly linked list with the operations of Lab Assignment 1. Hint: If using Java (or OOP), use the same Stack class you implemented, and change the array to an object of the singly linked list class. The functionality of push and pop is now based on the methods of the linked list class.
3. Practical application 1: Balanced brackets.
   (a) [2 marks] Implement the balanced-bracket checker algorithm of Lab Assignment 1 using the singly linked list implementation of the stack.
   (b) [2 marks] Explain how your algorithm checks balanced parenthesis in O(n), where the input is a string of length n.
4. [2 marks] Implement a queue that is store on the singly linked list of Question #1, with the following operations: enqueue, dequeue, front, size, isEmpty. Note: both operations, enqueue and dequeue, must run in O(1).
5. Practical application 2: The merge problem. Given two sorted lists of integers, A and B, the aim is to merge A and B, producing a sorted list S.
   (a) [4 marks] Write an algorithm that stores A and B in two queues (one in each queue), merges A and B, and outputs a sorted list S (also stored in a queue) in O(n) worst-case time. Note: your algorithm must use queues for A, B and S.
   (b) [2 marks] Implement the algorithm in your favorite programming language.
   (c) [2 marks] Explain how your algorithm runs in O(n), where n is the size of S.
   (d) [1 bonus mark] Run your algorithm on randomly generated sorted lists (queues) A and B of different sizes: 100, 200, 400, …, 51200 and record the CPU time taken by the algorithm. Create a table (or a plot) and explain how your algorithm runs in O(n) by inspecting your results.
6. [1 bonus mark] Write and implement a recursive algorithm that shows all the elements of the singly linked list, starting from the head. Explain the worst-case running time of your algorithm.

Total: 20 marks (22 with bonus marks)

Submission:

1. Your assignment must be submitted during the lab session in the section you are registered in. Any submission *on or 15 minutes* prior to the end of the lab session will **not** be accepted and a **zero** mark will be given. Late assignments will be given a **zero** mark. Submissions by email or videoconferencing will **not** be accepted.

2. Provide the source code for all your programs.
3. Run the balanced-bracket checker during the lab and show they work for various sample inputs, including (more may be included during the evaluation in the lab; a and d are balanced):
    a. (9*[3*{[(3+3)/5]*7}])
    b. {3*(2+[3-[4/[6/9]]]})
    c. ((3*(9-(4*(6-5))))
    d. {2-{3*{6/[[[(((9-0)))]]]}}/7}
4. Run the Merge algorithm with the following inputs:
    (i) A = 1,3,5,7,9 B = 2,3,6,8,10
    (ii) A = 1,2,3,4,5 B = 6,7,8,9,10
    (iii) A = 2,4,8,16,32 B = 1,3,5,7,9
    (iv) A = 10,11,12,13 B = 1,2,3,4
5. Run the recursive algorithm that shows the elements of the resulting queue of the Merge algorithm in the examples of #4.
6. Explain how your singly linked list, stack and queue work. These will be asked when the lab assignment is submitted. Marks will be deducted if not clear how your algorithms and programs work.