CSC 228-01 Data Structures and Algorithms, Spring 2018
Instructor: Dr. Natarajan Meghanathan

Project 3: Implementation of the Stack ADT using Singly Linked List and the Time Complexity Analysis of the Push and Pop Operations

Due: Feb. 21 by 1 PM (submission through Canvas)

In this project, you will implement the Stack ADT as a singly linked list. As discussed in class, the push, pop and peek operations for a Stack can be accomplished using the insertAtIndex(0, data), deleteElement(0) and read(0) functions for a singly linked list respectively, where the '0' in the function calls corresponds to the insertIndex, deleteIndex and readIndex. In such an implementation, the asymptotic time complexity is O(1) for all the three operations.

You are given the code for a singly linked list with the Node and List classes and a main function that pushes a sequence of integers to a Stack and then pops them out. You are required to modify the code for the List class to now reflect the code for a Stack class (including the name change for the classes). Only those functions that are needed for a Stack class should be retained (and renamed as per the Stack ADT) and others should be removed. Also, the implementation for the push, pop and peek operations in the Stack class should not have any loops; but, conditional statements (like 'if') could be included.

As you can see the code for the main function, the timers are setup for each trial and the times for the push and pop operations are measured (in microseconds) and then averaged out. You do not have to write any code in the main function.

You run the complete code for three different values of the Stack Size: 10000, 100000, 1000000. The maximum value for any element is always 50000 and the number of trials is always 50.

Report Submission (through Canvas): Submit everything together as **one PDF file**:
(a - 60 pts) Your code for the Stack class (that is adapted from the code given to you for the List class) implementing the push, pop and peek operations in constant time, i.e., O(1) time.
(b - 10 pts) Tabulate the average run time (in microseconds) for the push and pop operations for the different values of the Stack Size. Also, add columns that include the ratio of the Stack Size to the average run times for each operation (push and pop).
(c - 6 pts) Include screenshots of the outputs obtained for the average run time for each of the three Stack Size values.
(d - 12 pts) Explain why you observe a significant difference in the average run time for the push and pop operations? The push operation is expected to take a relatively longer time compared to the pop operation. Why is that?
(e - 12 pts) Even though theoretically, the push and pop operations are expected to take constant time (i.e., O(1) time), you are more likely to observe the average run times of the push and pop operations to increase with Stack size. Why do you think we can still claim that these operations theoretically take constant time? (Hint: Look at the values for the ratios of the Stack Size and the average run time for the push and pop operations that you computed and tabulated for (b)).