**CSC 228-01 Data Structures and Algorithms, Fall 2019**
**Instructor: Dr. Natarajan Meghanathan**

# <u>Project 4</u>: Implementation of the Queue ADT using a Stack Object and Performance Comparison of the EnqueueCostly and the DequeueCostly Approaches

**Due by: Oct. 17th, 11.59 PM**

In this project, you will implement the Queue ADT using a Stack object as the underlying data structure. The Stack is in turn implemented using a Doubly Linked List.

You are provided the code for the Node class of a Doubly Linked List, the Stack class and the Queue class as well as a main function.

The Queue class has two Stack objects (stack1 and stack2). We will use stack1 to store the contents of the queue and stack2 is primarily going to be an auxiliary object. Now, depending on where (in stack1) we store the front of the queue, we could have two versions of the queue. We will call these queues as the EnqueueCostlyQueue and DequeueCostlyQueue, primarily for the following reasons:

> In an EnqueueCostlyQueue, the dequeue operation is cheaper. Hence, the front of the queue (to be the data that is returned by a dequeue operation) should correspond to the top of stack1. To enqueue in such a queue, we have to go through the following sequence of actions:
>
> > First pop the contents of stack1 and push them to stack2 as they are popped.
> > Push the new data to be enqueued to stack1
> > Finally, pop the contents of stack2 and push them to stack1 as they are popped.
>
> In a DequeueCostlyQueue, the enqueue operation is cheaper. Hence, the top of stack1 corresponds to the data that has been enqueued recently, and the bottom of stack1 would correspond to the data that was enqueued first. Thus, the front of the queue (to be the data that is returned by a dequeue operation) should correspond to the bottom of stack1. To dequeue from such a queue, we have to go through the following sequence of actions:
>
> > First pop the contents of stack1 and push them to stack2 as they are popped.
> > Pop stack2 once and store the popped value in a temporary variable.
> > Pop the rest of stack2 and push the popped values to stack1 as they are popped.
> > Return the value of the temporary variable as the dequeued value.

The Queue class given to you has five functions:

- (i) isEmpty( ) function that is common to both the queue versions
- (ii) the enqueueCheaper(int) function that would be used for the DequeueCostlyQueue
- (iii) the dequeueCheaper( ) function that would be used for the EnqueueCostlyQueue
- (iv) the enqueueCostly(int) function that would be used for the EnqueueCostlyQueue
- (v) the dequeueCostly( ) function that would be used for the DequeueCostlyQueue.

Functions (i), (ii) and (iii) are already implemented.

**Primary Task:** Your primary task in this project is to implement the (iv) enqueueCostly(int) function for an EnqueueCostlyQueue and the (v) dequeueCostly( ) function for a DequeueCostlyQueue.

**Performance Assessment:** After implementing the functions (iv) and (v), you will test run your code (the main function already has all the timers and variables set for this purpose) for the following range of values for the queue size: 1000, ..., 10000 (in increments of 1000); the data values for all the trials range from 1...5000 and the number of trials for each value of queue size is 5. Measure the average enqueue times and average dequeue times for both the EnqueueCostlyQueue and the DequeueCostlyQueue.

**Submission:**

(1 - 70 pts) Submit your entire code as a separate .cpp file

Submit a single PDF file that has your responses for the following:
(2 - 15 pts) Illustrate the Enqueue operation in an EnqueueCostlyQueue and the Dequeue operation in a DequeueCostlyQueue with an example queue (stack1) of 5 elements. Show all the steps for each case.

(3 - 15 pts) Tabulate the results for the average enqueue time for an EnqueueCostlyQueue and the average dequeue time for a DequeueCostlyQueue for the queue size values ranging from 1000 to 10000. Are the time values approximately the same or appreciably different from each other? Interpret the results.