**Department of Computing**

**CS250: Data Structures and Algorithms**

**Class: BEE-6AB**

# Lab 4: Stack & Queues

**Date: 8th October, 2015**

**Time: 10am-1pm & 2pm-5pm**

# Instructor: Mr. Faisal Shafiyat

**Lab 4: Stack & Queues**

**Introduction**

In this lab, you will be introduced to what is necessary to implement an abstract stack and queues.

**Objectives**

Objective of this lab is to get familiar with implementation of stacks, queues and a revision of linkedlist.

**Tools/Software Requirement**

Visual Studio C++

**Description**

Follow the instructions given in the lab tasks and complete the lab.

**Lab Tasks**

**Task 1: Linked List**

Commit Assignment 01 on bitbucket (in a new repository named ployadd). Revise the code to use linked lists from standard template library (stl::forward\_list) instead of your custom made linked list class. Run the test cases you wrote earlier to verify that the new code for adding polynomials is working correctly. Push the revised code to bitbucket repository. Upload a screen-shot of bitbucket repository commit history as well as the revised code.

**Task 2: An abstract stack**

The goal of this task is to gain familiarity with an abstract stack; complete Lab Stack.cpp.

1. Begin this lab by completing a simple class that represents a "stack".

2. Implement the function definitions of the Stack class.

(a) Implement the Push() method. A stack class will typically contain a Node reference for the top of the stack that is set to null at construction time to indicate an empty stack.

(b) Implement the Pop() method. The method should remove the item at the top of the stack.

(c) Implement the Peek() method. The method should return the value at the top of the stack.

3. Execute the completed code, when elements are pushed in the order 100, 200, 300, 400, 500, and then popped out.

**Task 3: An abstract queue**

The goal of this task is to gain familiarity with an abstract queue.

1. Begin this lab by completing a simple class that represents a "queue".

2. Implement the function definitions of the Queue class.

(a) Implement the Enqueue() method. A queue class will typically contain a Node reference for the back of the queue that is set to null at construction time to indicate an empty queue.

(b) Implement the Dequeue() method. The method should remove the item at the front of the queue.

(c) Implement the Front() method. The method should return the value at the front of the queue.

(d) Implement the Back() method. The method should return the value at the back of the queue.

3. Execute the completed code by en-queuing and de-queuing 100, 200, 300, 400, 500.

**Deliverables**

Students are required to upload the lab on LMS before deadline.

**Note:** Use proper indentation and comments. Lack of comments and indentation will result in deduction of marks. You will submit your workingcodes in **word document** (do **NOT** take screenshot of code, just copy your code and paste it). The name of word document should follow this format. i.e. **YOUR\_NAME\_Lab#**