

**COMP 2230 – Data Structures and Algorithm Analysis**

Assignment #3: Linked Stacks

## Due Date: Section 01 - Sept. 26th 2024 Section 02 Sept 27th 2024

**Chapter 13**

**Problem 1**:

Implementation of the Size(), isEmpty(), and toString() methods into the LinkedStack class.

Input: an element to add onto the stack.

Output: elements removed from the stack.

**Analysis of the problem:**

Subtask and Analysis:

* Initialize stack(O(1))
  + ‘count’ variable set to 0
  + ‘top’ variable set to null
* Push elements to the top of the stack(O(1))
  + Creates a ‘temp’ variable of type LinearNode and passes the value into the node
  + Sets the ‘temp’ variable reference to the ‘top’
  + Sets ‘top’ to ‘temp’
  + ‘Count’ is incremented
* Pop elements from the top of the stack (O(1))
  + Validate if stack is empty and throws emptyCollectionException if true
  + Sets ‘result’ variable of type T to the value of the top node
  + ‘top’ is set to the next node
  + ‘Count’ is decremented
* Peek element at the top of the stack(O(1))
  + Validates if stack is empty and throws emptyCollectionException if true
  + Returns the element of the top node
* Check is the stack is empty(O(1))
  + Returns true if the ‘count’ is 0
* Get the size of the stack(O(1))
  + Returns ‘count’
* Format string to display stack(O(1))
  + Returns the stack in a string

Prototypes:

* + **LinkedStack()**
    - Pre condition: none
    - Post condition: new LinkedStack created with ‘top’ variable set to null and ‘count’ set to 0.
  + **push(T element)**
    - Pre condition: non null element to be pushed
    - Post condition: element added to top node of stack and count variable incremented
  + T **Pop()**
    - Pre condition: non empty stack
    - Post condition: top node is removed and count variable decremented, removed element returned
  + **T Peek()**
    - Pre condition: non empty stack
    - Post condition: top element returned without being removed
  + **Boolean isEmpty()**
    - Pre condition: none
    - Post condition: returns true if count is 0 and false otherwise.
  + **Int size()**
    - Pre condition: none
    - Post condition: returns the ‘count’ variable

Test Cases:

* Create an empty arraystack with initial capacity
* Push elements to capacity and then over capacity to test expandCapacity()(also tests push method)
* Test pop() method that removes the top of the stack
* Test peek() method that views the top of the stack without removing it
* Test isEmpty() method that checks if the stack is empty and returns true if it is and false if not
* Test size() method that returns the number of elements without modifying the stack
* Tests peek() while stack is empty to show it throws exception
* Test pop() while stack is empty to show it throws exception

**Problem 2**: Complete the implementation of a Linked Drop Out Stack Class.

**Subclass from the LinkedStack**

Remember to verify the functionality of your programs, **verify all methods**.

**Assignment Submission:**

Submit a print-out of the program source code and a sample of the output, for each problem. Note you must follow the marking guidelines as identified in the LabMark document.