COMP20230: Data Structures & Algorithms
Assignment 2 Briefing

Dr Madhusanka Liyanage

Office: B1.14
School of Computer Science
University College Dublin

madhusanka@ucd.ie

Learning Outcomes

On successful completion of this project the learner will be able to:

- Understand the structure, nature and use of fundamental data structures including, arrays, linked lists, stacks, and queues
- Implement the linked lists data structures in Python
- Implement the stack data structures in Python
- Implement the queue data structures in Python
- Design programs using these constructs to solve large problems.

Assignment 2

You need to implement following tasks:

Part I:

- 1) Write a class called myLinkedList that implements the following singly linked list interface:
  - add_first: adding an element at the front
  - add_last: adding an element at the end
  - remove_first: removing a node at the front
  - list_traversal: every node in the list has been seen
- 2) Define in your own words the terms: stack ADT and queue ADT. List the key operations and support operations commonly associated with these ADTs. For each ADT, give two real world data examples and explain them briefly.

Continued..

Assignment 2

Part II:

- 1) Adopt the ADT concepts of Part I (task 2) to provide a complete implementation of a stack ADT
- 2) What values are returned during the following series of stack operations, if executed upon an initially empty stack S? push(5), push(3), pop(), push(2), push(8), pop(), pop(), push(9), push(1), pop(), push(7), push(6), pop(), pop(), push(4), pop(), and pop()
- 3) Suppose an initially empty stack S has executed a total of 35 push operations, 15 top operations, and 10 pop operations, 3 of which raised Empty errors that were caught and ignored. What is the current size of S?

Continued..

Assignment 2

Part III:

- 1) Adopt the ADT concepts of Part I (task 2) to provide a complete implementation of a queue ADT
- 2) What values are returned during the following sequence of queue operations, if executed on an initially empty queue Q? enqueue(5), enqueue(3), dequeue(), enqueue(2), enqueue(8), dequeue(), dequeue(), enqueue(9), enqueue(1), dequeue(), enqueue(7), enqueue(6), dequeue(), dequeue(), enqueue(4), dequeue(), dequeue()
- 3) Suppose an initially empty queue Q has executed a total of 50 enqueue operations, 15 top operations, and 15 dequeue operations, 5 of which raised Empty errors that were caught and ignored. What is the current size of Q?

Your report should:

- Have proper structure with introduction, main body covering the various areas you address, and conclusions
- Have satisfactory technical coverage, balancing breadth of coverage with depth
- Have soundness of argument
- Have good clarity of expression and level of readability
- Clearly reference sources of information, and avoid plagiarism

Things to consider

- Give examples to test the operation of myLinkedList such as adding new elements to the list and removing elements from the list. Print the content of the list after each operation
- Implement some other support operations (ex. *is_Empty*(), *size*) if needed
- Use arrays OR linked lists to implement the stack/queue ADT
- Code should be efficient and fast
- Class names, method names, variable names should be meaningful
- Remove unnecessary or unused variables

Assessment: Things to consider

Speed: Test running time as input size grows. Compare running-time vs time complexity
Efficiency (operations/capacity): Can you estimate big-O (or big-theta), capacity requirements
Clarity: - are class names, method names, variable names meaningful; is a consistent style used throughout;
- code refactored (remove unnecessary or unused variables, loops etc.)
Correctness: - does it do what it is meant to do? - is it scaleable?
- is it hardcoded
Maintainability: - Are appropriate comments included in code?
- Are unit tests provided to test the key functionality

Assessment

Submission on Friday 16th of April 2021

- This assignment is worth 40% of the overall mark for COMP20230
- It should be uploaded to Brightspace by 2330pm on Friday 16th of April 2021
- Submission: A report (PDF) contains codes and analysis of your solution.**In addition, the codes should be submitted in Jupiter notebooks**
- Evaluation Criteria: Logic 50%, Code 25% and Creativity of report 25%