Lab 1 – Fundamental Data Structures

Release: 25 Aug 2023 (Fri, Week 2)
Due: Online submission in eLearn by Sunday, 3 Sep 2023, 11pm

You can use the similar setup as what you have installed for the cs102/is442 Programming Fundamentals 2/ Object Oriented Programming course (e.g. Java JDK, Visual Studio Code).

Visual Studio Code
https://code.visualstudio.com/download

Java Development Kit
(You don't need to reinstall if you have an earlier version from cs102/is442, e.g. from Java 11 onwards)
https://www.oracle.com/java/technologies/downloads/

Setting Java Path for Windows notebooks
https://mkyong.com/java/how-to-set-java_home-on-windows-10/

Instructions:

1. This is an individual lab. You will be awarded class participation points for the labs component based on the completion of submission, not on the efficiency of solution (you are encouraged to explore the most efficient approach for your own learning purposes)

2. You can discuss the general approach to solve the questions but you should not be sharing your codes with anyone.

3. Unless specifically stated in the question, you are allowed to create additional methods in the classes.

4. Please ensure that all your codes can be compiled.

5. Multiple submissions are allowed but only the most recent submission will be kept in the system as the final submission.

6. Please zip up your submission as named the zip file as <StudentName>.zip.

7. The deadline is final, late submissions will not be accepted.

Q1 - Arrays

Given a non-empty Integer array and the values of the elements in the array are greater than 0. Complete the following requirements in **NumbersArray.java**.

- Write a method named **findMax** that returns the maximum value in the array
- Write a method named **findDuplicates** that returns an array containing the duplicate values in the array
- Write a method named **findUnique** that returns an array containing the unique values in the array

```
Input : [1, 2, 3, 4, 1, 5]
Max number : 5
Duplicate numbers : [1]
Unique numbers : [2, 3, 4, 5]

Input : [1, 2, 3, 4, 1, 2, 5, 3]
Max number : 5
Duplicate numbers : [1, 2, 3]
Unique numbers : [4, 5]
```

Q2 – SinglyLinkedList

An implementation of a Single Linked List has been given in SinglyLinkedList.java. Study the codes and complete the following requirements in SinglyLinkedList.java to extend the functionality of the linked list. Please ensure the head and tail nodes are set accordingly if they are impacted by the methods. Compile and run SinglyLinkedListTest.java to test your implementation.

- Write a method named **toString** to print out the elements starting from the head element
- Write a method named **removeLast** that removes the last element
- Write a method named **reverse** that reverse the sequence of all the elements

```
Add First - 1
Linked List : 1
First Element : 1
Last Element : 1

Remove Last
Linked List :
First Element : null
Last Element : null

Add First - 1
Linked List : 1
First Element : 1
Last Element : 1

Add First - 2
Linked List : 21
First Element : 2
Last Element : 1

Add Last - 3
Linked List : 213
First Element : 2
Last Element : 3

Add Last - 4
Linked List : 2134
First Element : 2
Last Element : 4

Remove Last
Linked List : 213
First Element : 2
Last Element : 3

Reverse Linked List
Linked List : 312
First Element : 3
Last Element : 2
```

Q3 – Grouping Nodes within a Double Linked List

An implementation of a Double Linked List has been given in DoublyLinkedList.java. Study the codes and complete the following requirements in DoublyLinkedList.java to extend the functionality of the linked list. Please ensure the header and trailer nodes are set accordingly if they are impacted by the method. Compile and run DoublyLinkedListTest.java to test your implementation.

- Write a method named **group** that will group all the nodes with a null values together while preserving the order of the non-null values. Refer to the examples on the order of the expected output.

Before Group :
Linked List : 1 null 2
First Element : 1
Last Element : 2

After Group :
Linked List : null 1 2
First Element : null
Last Element : 2

Before Group :
Linked List : 4 null 1 null 3
First Element : 4
Last Element : 3

After Group :
Linked List : null null 4 1 3
First Element : null
Last Element : 3