LAB 3: Doubly LinkedList

[CO3]

Instructions for students:

- You may use Java / Python to complete the tasks.
- If you are using **JAVA**, then follow the **Java Template**.
- If you are using **PYTHON**, then follow the **Python template**.

NOTE:

- **YOU CANNOT USE ANY OTHER DATA STRUCTURE OTHER THAN THE LINKED LIST YOU'RE CREATING.**
- **IF ONLY THE QUESTION ALLOWS YOU TO CREATE OTHER DATA STRUCTURES, THEN YOU CAN.**
- **YOUR CODE SHOULD WORK FOR ANY VALID INPUTS.**

The Lab Tasks should be completed during the lab class
YOU HAVE TO SUBMIT ONLY THE ASSIGNMENT TASK

Total Lab 3 Assignment Tasks: 3
Total Marks: 15

Lab Task

In this part of the lab, we will play with Dummy Headed Doubly Circular Linked List. If you want to read about this type of linked list then check **this file.**

> **Note: Even though we're learning Dummy Headed Doubly Circular Linked List, the methods you'll be writing simulate a different data structure. Can you guess what that is? Hint: It's something you face in front of the elevator everyday.**

In the first part of this lab, you have to implement a waiting room management system in an emergency ward of a hospital. Your program will serve a patient on a **first-come, first-served basis.** The **Simulation of WRM** can be found **here**.

Solve the above problem using a **Dummy Headed Doubly Circular Linked List.**

1. You need to have a **Patient** class so that you can create an instance of it (patient) by assigning id(integer), name (String), age (integer), and bloodGroup (String).
2. Write a **WRM** (waiting room management) class that will contain the following methods.
    a. **registerPatient(id, name, age, bloodgroup):** This method will register a patient into your system. The method will create a Patient type object with the information received as a parameter. It means this method will add a patient-type object to your linked list.
    b. **servePatient():** This method calls a patient to provide hospital service to him/her. In this method, you need to ensure that to serve the patient who was registered first. (serving means removing the patient from line)
    c. **cancelAll():** This method cancels all appointments of the patients so that the doctor can go to lunch.
    d. **canDoctorGoHome():** This method returns true if no one is waiting; otherwise, it returns false.
    e. **showAllPatient():** This method prints all IDs of the waiting patients in sequential order. It means the patient who got registered first will come first, and so on.
    f. **reverseTheLine():** This method reverses the patient line. It means the patient who got registered last will come first, and so on.

Important Hints:

1. In your program, there won't be any class called **"Node".** Instead, the **Patient class** will work as the Node class.
2. On the other hand, the **WRM** class will act as the Dummy Headed Doubly Circular Linked List, which will contain all the other functions/methods.

Assignment Tasks [Need to Submit]

If you need to use extra helper functions/methods feel free to create & use them. If you use them then make sure to submit those extra functions/methods as well. Here's the Java Template & Python Template

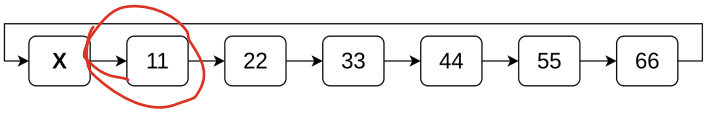
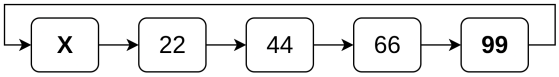
continued from the previous assignment...

5. Sum Odd Append

Write a method/function called **sumOddAppend().** The function will take only one parameter, referencing a **dummy-headed singly circular linked list**. The function/method removes all the nodes containing odd values while summing them up. Finally, a node containing the summation is inserted at the end.

NOTE:

1. YOU **CAN** CREATE **ONLY 1 EXTRA NEW NODE** using the given Node Class.

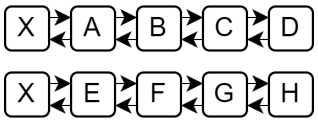
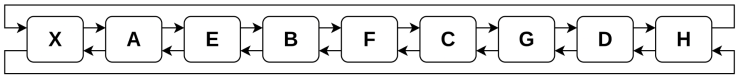
Given Dummy Headed Singly Circular Linked List	Expected Modified Linked List
	

6. Pair Join

Write a method/function called **pairJoin()**. The function takes two parameters, referencing **two dummy heads** of two different Dummy Headed Doubly Linked Lists. The method/function will modify the connections of the two given linked lists and combine them in pairs as shown in the example below. The method/function will not return anything because the first dummy head will naturally become the new head.

NOTE:

1. YOU **CANNOT** CREATE ANY NEW NODES.
2. YOU CAN ASSUME THE LENGTH OF THE GIVEN TWO LISTS WILL ALWAYS BE EQUAL.

Given Dummy Headed Doubly Linked List	Expected Modified Linked List
	

7. Range Move

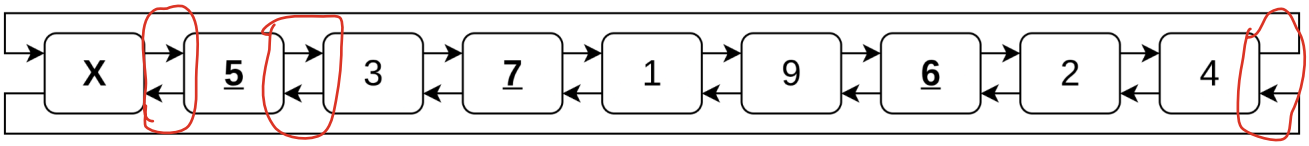
Write a method/function called **rangeMove()**. The function takes three parameters, referencing **one dummy head and two integers**. The method/function will move all the nodes that fall within the given range of integers to the back of the linked list. The method/function will not return anything because the dummy head won't change.

NOTE:

3. YOU **CANNOT** CREATE ANY NEW LINKED LIST FOR THIS TASK
4. YOU **CANNOT** CREATE ANY EXTRA NODE.

Given Dummy Headed Doubly Circular Linked List

Given Range: [start,end] = [5,7]



Expected Modified Linked List