**LAB 2: Node & 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 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 2 Assignment Tasks: 4**

**Total Marks: 20**

# **Intro Lab Task**

# **Basics of Node**

| **This is an intro task & there isn’t any driver code for this**   * For java, create a separate folder for this task and follow the instructions. * For Python, create a separate colab/ipynb/py file and follow the instructions. |
| --- |

* Design a Node class.
  + Declare two instance variables, one called **elem** (Object data type for java) and another one called **next** (Node datatype).
  + Write a constructor that has only one parameter (Object data type for java) and assigns the parameter value to the **elem** instance variable.
* Design a Tester class (for java) / Design another code cell (for python)

| * + Create 5 different objects of Node class. Assign values as shown in the illustration.   + Variable names should be: **n1, n2, n3, n4, n5**.   + Connect the 5 nodes as shown in the illustration.   **Now, execute the lines given below and try to understand the output. You may need pen & paper.**  **If there are errors, try to figure out why that error occurred and how to fix it.**  **Note: Java & Python output won’t always be the same.** | | |  | | |
| --- | --- | --- | --- | --- | --- |
| **JAVA** | | | **PYTHON** | | |
| System.out.println( n1.next ); | | | print( n1.next ) | | |
| System.out.println( n3.next.elem ); | | | print( n3.next.elem ) | | |
| Node x = n4.next;  System.out.println( n1.elem + x.elem); | | | x = n3.next  print( x.elem + n4.elem) | | |
| x.next = n3;  System.out.println(n2.next.next+n5.next); | | | x.next = n2  print(n.next.next+n.next) | | |
| x.next.next = null;  n1.next.elem = 321; | | | x.next.next = None  n2.next.elem = 7.98 | | |
| n4.next = 532;  System.out.println(n4.next.elem); | | | n4.next = 532  print(n4.next.elem) | | |

| **Now we can move on to actual problem solving using nodes/linkedlist.**  **Please use the** [**Java Template**](https://drive.google.com/file/d/1hpTsXA4GuQIjnpOZP5n7PSjoOcMGkxxQ/view?usp=sharing) **or** [**Python Template**](https://colab.research.google.com/drive/13CnYeaml3NjNJGxcN312AhyTVJ66DKFH?usp=sharing) |
| --- |

# **Main Lab Tasks [No Need to Submit]**

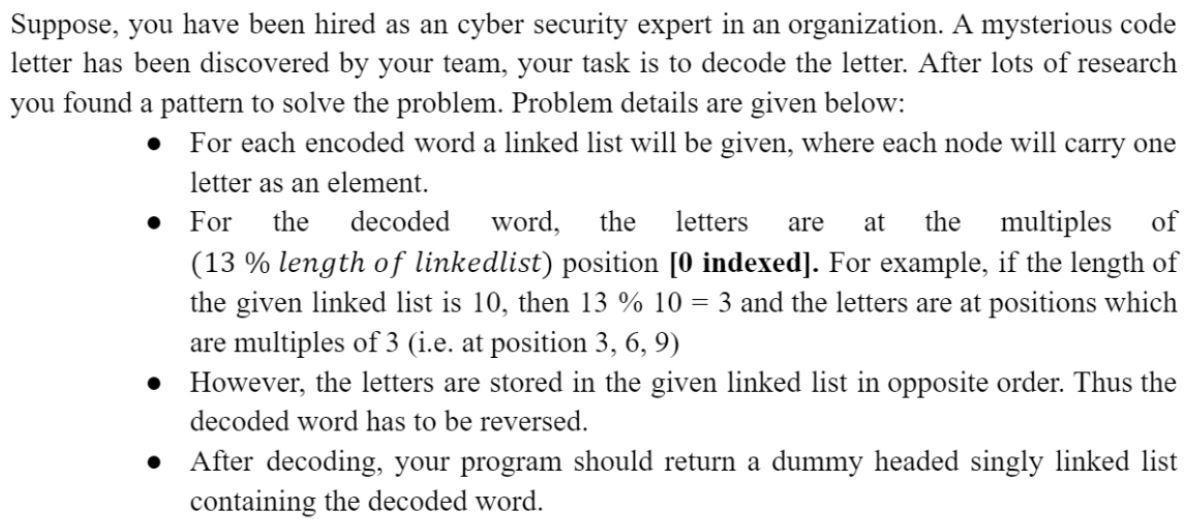
1. **Assemble Conga Line**

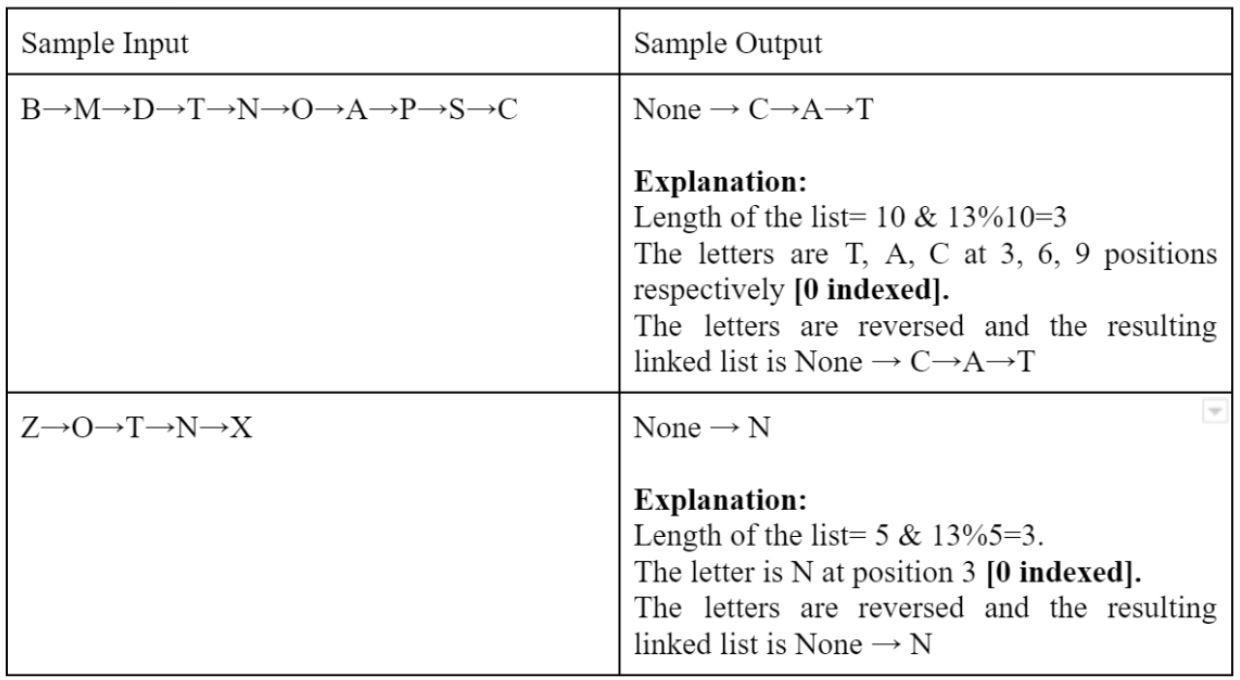
Have you ever heard the term [conga line](https://en.wikipedia.org/wiki/Conga_line)? Basically, it’s a carnival dance where the dancers form a long line. Everyone holds the waist of the person in front of them and their waists are held in turn by the person to their rear, excepting only those in the front and the back. It kind of looks like [this](https://www.pixtastock.com/illustration/11648015)-



By now, you can quite understand the suitable data structure to represent a conga line. Now you are the choreographer of the Conga Dance in a Summer Festival. You wish to arrange the conga line **ascending** age wise and tell the participants to stand in a line likewise. Now as technical you are, can you write a method that will take the conga line and return True if everyone stands according to your instruction. Otherwise returns False.

| **Sample Input** | **Sample Returned Result** |
| --- | --- |
| **10 → 15 → 34 → 41 → 56 → 72** | **True** |
| **10 → 15 → 44 → 41 → 56 → 72** | **False** |

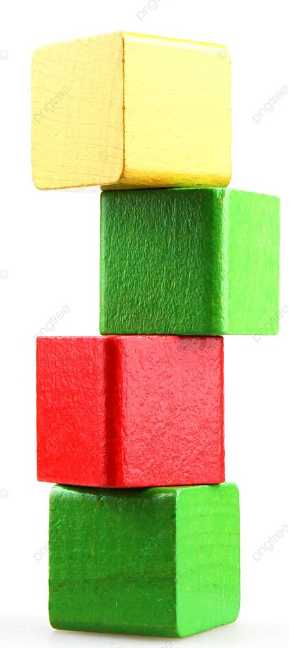
1. **Word Decoder**

****

# **Assignment Tasks [Need to Submit]**

### **Building Blocks**

Your twin and you are under an experiment where the amount of thinking similarities you two have is being observed. As per the experiment, you are given a certain number of building blocks of different colors and are told to make a building using those blocks in two different rooms.



After the buildings are finished, the observers check whether the two buildings are the same based on the block colors. Now, you are the tech guy of that team and you are instructed to write a program that will output “Similar” or “Not Similar” given the two buildings. For fun, you decided to represent those buildings as a linked list!

**NB:** Red means a red block

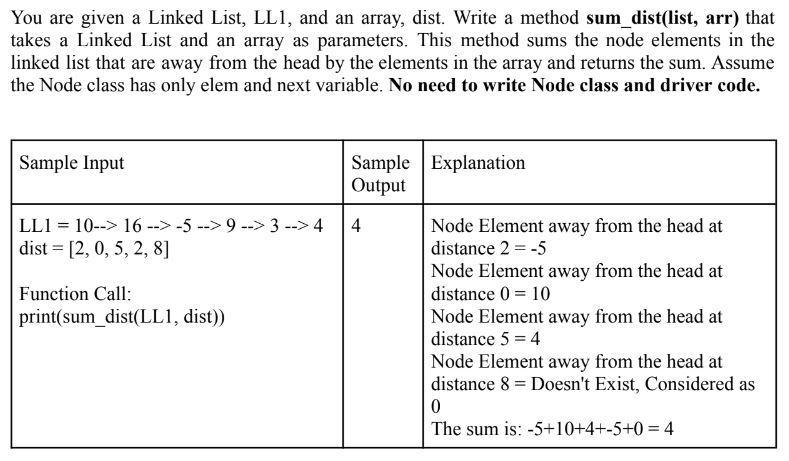
Blue means a blue block

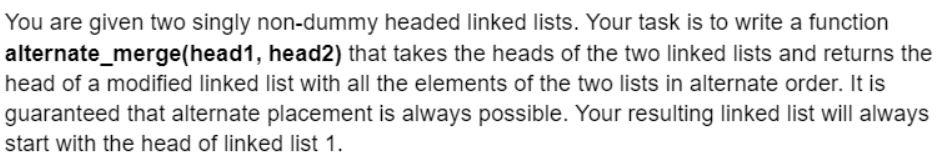
Yellow means a yellow block

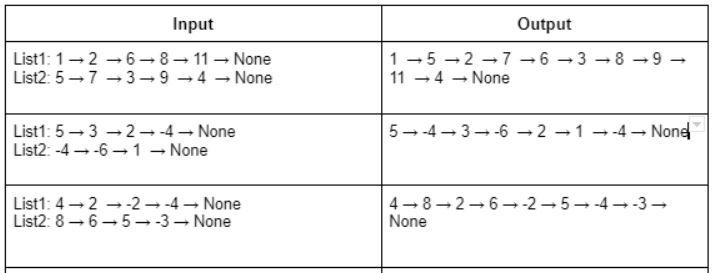
Green means a green block.

| **Sample Input** | **Sample Output** |
| --- | --- |
| **building\_1 =**  **Red→ Green→ Yellow→ Red→ Blue→ Green**  **building\_2 =**  **Red→ Green→ Yellow→ Red→ Blue→ Green** | **Similar** |
| **building\_1 =**  **Red→ Green→ Yellow→ Red→ Yellow→ Green**  **building\_2 =**  **Red→ Green→ Yellow→ Red→ Blue→ Green** | **Not Similar** |
| **building\_1 =**  **Red→ Green→ Yellow→ Red→ Blue→ Green**  **building\_2 =**  **Red→ Green→ Yellow→ Red→ Blue→ Green→ Blue** | **Not Similar** |

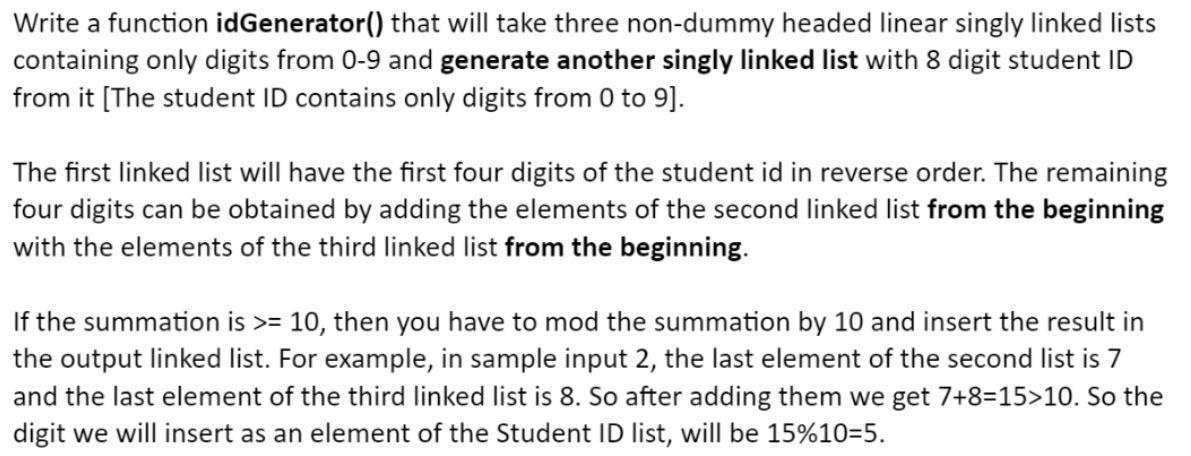
1. **Sum of Nodes**

****

1. **Alternate Merge**



**NOTE: The space complexity of the solution must be O(1). Which means, You’re NOT ALLOWED to create any new linked list for this task.**

1. **ID Generator**

### 

# 

### 

| **You Will Have To Submit Lab 2 and Lab 3 as a single file after next week. Basically, There will be only 1 combined submission for Lab 2 & Lab 3.** |
| --- |

### 