**CSc 2720 - Data Structures: Lab 5**

**How to Submit:**

Please submit your answers in icollege once you have completed. Failure to submit will result in a *zero* for this lab.

**Refresher:**

We have seen in class that deletion and insertion operations in a linked list necessitate a `curr` and `prev` pointers. To delete a node which `curr` references: `prev.next = curr.next;`

**Problem 1:**

Write a function `deleteAtIndex` that takes a linked list `head` and an integer `index` as an argument, looks for the node in the linked list at `index` and delete the node from the linked list (disconnect it from the list).

**Assumptions:**

- Users always gives a valid index.
- User will *never delete the head/first node* in the list. (i.e: the user will never enter index=0).
- The linked list will have at least two elements.

**Example:** Given linked list -- [1,2,3], which looks like the following:

```
1 ------ 2 ------ 3
```

**Input:** `index = 1`

**Output:** List After Deletion: 1 3
public class Node {
    int item;
    Node next;
    // Node Constructor
    Node(int d) {
        item = d;
        next = null;
    }
}

public class Tester {
    public static void main(String[] args){
        Node head = new Node(1);
        Node second = new Node(2);
        Node third = new Node(3);
        head.next = second;
        second.next = third;

        /* The current linked list is as follows:
         * head second third
         * +-------+-------+-------+
         * | 1    | 2      | 3      |
         * +-------+-------+-------+
        */
        System.out.println("List Before Deletion");
        printLinkedList(head); // Should be 1 2 3

        // User wants to delete at index 2
        deleteAtIndex(2, head);
        System.out.println("List After Deletion at index 2");
        printLinkedList(head); // Should be 1 2

        // User wants to delete at index 1
        deleteAtIndex(1, head);
        System.out.println("List After Deletion at index 1");
        printLinkedList(head); // Should be 2
    }
    // To pass the linked list to a function, you only need to pass the head
    public static void deleteAtIndex(int value, Node head){
        Node prev, curr;
        int counter = 0; // increment count as you traverse the list
        // INSERT CODE HERE
    }
    // Node traversal and printing
    public static void printLinkedList(Node head){
        for(Node cur = head; cur!=null; cur=cur.next){
            System.out.print(cur.item+" ");
        }
        System.out.println();
    }
}