Objective(s) :

- To understand the basic implementation of a singly linked list
- Be able to manipulate list structure.

task 1: Implement **MyLinkedListTricky extends MyLinkedList** with the following methods

public void q1_rotate_counter_clockwise(int k) -> Rotate the linked list counter-clockwise by k nodes where k is a positive integer not larger than the list's size.

public void q2_reverse() -> Reverse the list's element.

public void q3_remove_dup() -> Remove duplicates (node which its value the list already has a node with the value.) from the list (if exists). Your solution will keep the value closet to the head.

public void q4_add_one() -> Given a number represented in a linked list such that each digit corresponds to a node in a linked list. Add 1 to it. For example, 9999 is represented as (9->9->9->9) and adding 1 to it should result in (1->0->0->0->0)

public boolean q5_isPalindrome() -> Given a singly linked list of integers, the method returns true if the list is palindrome, else false.

```java
static void q1() {
    int [] d = {10,20,30,40,50};
    MyLinkedListTricky mList = new MyLinkedListTricky();
    mList.add(d);
    System.out.println("before -> " + mList);
    mList.q1_rotate_clockwise(1);
    System.out.println("(k = " + 1 + ")  -> " + mList);
    mList.q1_rotate_clockwise(3);
    System.out.println("(k = " + 3 + ")  -> " +mList);
    mList.q1_rotate_clockwise(7);
    System.out.println("(k = " + 7 + ")  -> " +mList);
}
```

```
before -> head->(10)->(20)->(30)->(40)->(50)->null
(k = 1)  -> head->(20)->(30)->(40)->(50)->(10)->null
(k = 3)  -> head->(50)->(10)->(20)->(30)->(40)->null
(k = 7)  -> head->(50)->(10)->(20)->(30)->(40)->null
```

```java
static void q2() {
    int [] d = {1,2,3,4,5,6,7,8};
    MyLinkedListTricky mList = new MyLinkedListTricky();
    mList.add(d);
    System.out.println("before -> " + mList);
    mList.q2_reverse();
    System.out.println("after  -> " + mList);
}
```

```
-q2--------
before -> head->(1)->(2)->(3)->(4)->(5)->(6)->(7)->(8)->null
after  -> head->(8)->(7)->(6)->(5)->(4)->(3)->(2)->(1)->null
```

```java
static void q3() {
    int [] d = {13, 11, 4, 15, 4};
    MyLinkedListTricky mList = new MyLinkedListTricky();
    mList.add(d);
    System.out.println("before -> " + mList);
    mList.q3_remove_dup();
    System.out.println("after  -> " + mList);
    int [] e = {13, 11, 15, 4};
    mList = new MyLinkedListTricky();
    mList.insert(e);
    System.out.println("before -> " + mList);
    mList.q3_remove_dup();
    System.out.println("after  -> " + mList);
}
```

```
-q3--------
before -> head->(13)->(11)->(4)->(15)->(4)->null
after  -> head->(13)->(11)->(4)->(15)->null
before -> head->(13)->(11)->(15)->(4)->null
after  -> head->(13)->(11)->(15)->(4)->null
```

```java
static void q4() {
    int [] d = {1, 9, 9, 9};
    MyLinkedListTricky mList = new MyLinkedListTricky();
    //mList.add(d);
    System.out.println("before -> " + mList);
    mList.q4_add_one();
    System.out.println("after  -> " + mList);
}
```

```java
static void q5() {
    int [] d = {21, 33, 33, 21};
    boolean isPalind;
    MyLinkedListTricky mList = new MyLinkedListTricky();
    mList.add(d);
    isPalind = mList.q5_isPalindrome();
    System.out.println(mList + " isPalindrome() = " + isPalind);
    int [] e = {21, 33, 44, 33, 21};
    mList = new MyLinkedListTricky();
    mList.add(e);
    isPalind = mList.q5_isPalindrome();
    System.out.println(mList + " isPalindrome() = " + isPalind);
    int [] f = {1, 9, 9, 9};
    mList = new MyLinkedListTricky();
    mList.add(f);
    isPalind = mList.q5_isPalindrome();
    System.out.println(mList + " isPalindrome() = " + isPalind);
}
```

```
-q4--------
before -> head->(1)->(9)->(9)->(9)->null
after  -> head->(2)->(0)->(0)->(0)->null
-q5--------
head->(21)->(33)->(33)->(21)->null isPalindrome() = true
head->(21)->(33)->(44)->(33)->(21)->null isPalindrome() = true
head->(1)->(9)->(9)->(9)->null isPalindrome() = false
```

submission: MyLinkedListTricky_XXYYYY.java where XX is the first 2 digit and YYYY is the last 4 digit of your student id.

Due date: TBA