**Singly Linked List**

In this assignment, you will create your own implementation of a singly linked list in Java. You will need 3 files: Node.java, SinglyLinkedList.java, and ListTester.java. As long as these files are in the same directory, you will not need to import them explicitily. We recommend that you create a ListAssignment folder and place your 3 files there.

Node.java

public class Node {

public int value;

public Node next;

public Node(int value) {

// your code here

}

}

SinglyLinkedList.java

public class SinglyLinkedList {

public Node head;

public SinglyLinkedList() {

// your code here

}

// SLL methods go here. As a starter, we will show you how to add a node to the list.

public void add(int value) {

Node newNode = new Node(value);

if(head == null) {

head = newNode;

} else {

Node runner = head;

while(runner.next != null) {

runner = runner.next;

}

runner.next = newNode;

}

}

}

ListTester.java

public class ListTester {

public static void main(String[] args) {

SinglyLinkedList sll = new SinglyLinkedList();

sll.add(3);

sll.add(4);

sll.add(10);

sll.add(5);

sll.add(15);

sll.add(2);

sll.remove();

sll.remove();

sll.printValues();

}

}

Objectives:

* Practice creating classes and objects.
* Learn and implement a singly linked list in Java.

Tasks:

* Create a Node class like the above.
* Fill in the constructor method that sets the value to a given number and next to null of your Node objects.
* Create a SinglyLinkedList class like the above.
* Create a constructor method that sets the head to null of your SinglyLinkedList objects
* Create a remove() method that will remove a node from the end of your list
* Create printValues() method that will print all the values of each node in the list in order
* Create a ListTester class like the above.

Optional Challenges:

* Implement a find(int) method that will return the first node with the value in the parameter
* Implement a removeAt(int) method that will remove the node after n nodes, where n is the parameter. For example, if n is 0, remove the first node. If n is 1, remove the second node. Similar to Arrays.