Back to linked lists

• Snarf the code to today’s recitation
• Look at the code
• What is similar between a Node in a LinkedList and a Node in a tree?
  (compare this code to Wednesday’s)

Announcements

• New assignment (on linked lists) after fall break
Today

- Revisit linked lists
- Code with linked lists
  - very helpful for the next assignment

```java
IntTreeNode root = null;

public class IntTreeNode {
    public int myValue;
    public IntTreeNode myLeft; // holds smaller tree nodes
    public IntTreeNode myRight; // holds larger tree nodes

    public IntTreeNode(int val) { value = val; }
}
```
private Node myHead;

private class Node {
    String myValue;
    Node myNext;

    Node(String value, Node next) {
        myValue = value;
        myNext = next;
    }
}

public void addAtBeginning(String valueToAdd) {
    myHead = new Node(valueToAdd, myHead);
}

StringLinkedList s = new StringLinkedList();
s.addAtBeginning("Hi");
public void addAtEnd(String valueToAdd)
{
    //some code goes here
}

s.addAtEnd("World");

---

public void removeLongestString()
{
    // your code goes here
    // when you implement this function, be sure to think about
    // a. what if the list is empty
    // b. what if the longest element is the first element
    // c. what if the list has only 1 element
}

---
• doubleList()
  • Takes a list and doubles each element \([a,b,c] \rightarrow [a,a,b,b,c,c]\)

• moveToEnd(int k)
  • move k elements to the end of the list
  • if \(k = 2\), \([a,b,c,d] \rightarrow [c,d,a,b]\)

• reverse()
  • \([a,b,c] \rightarrow [c,b,a]\)

• Make sure your code passes the unit tests!

• submit to recitation_6 folder