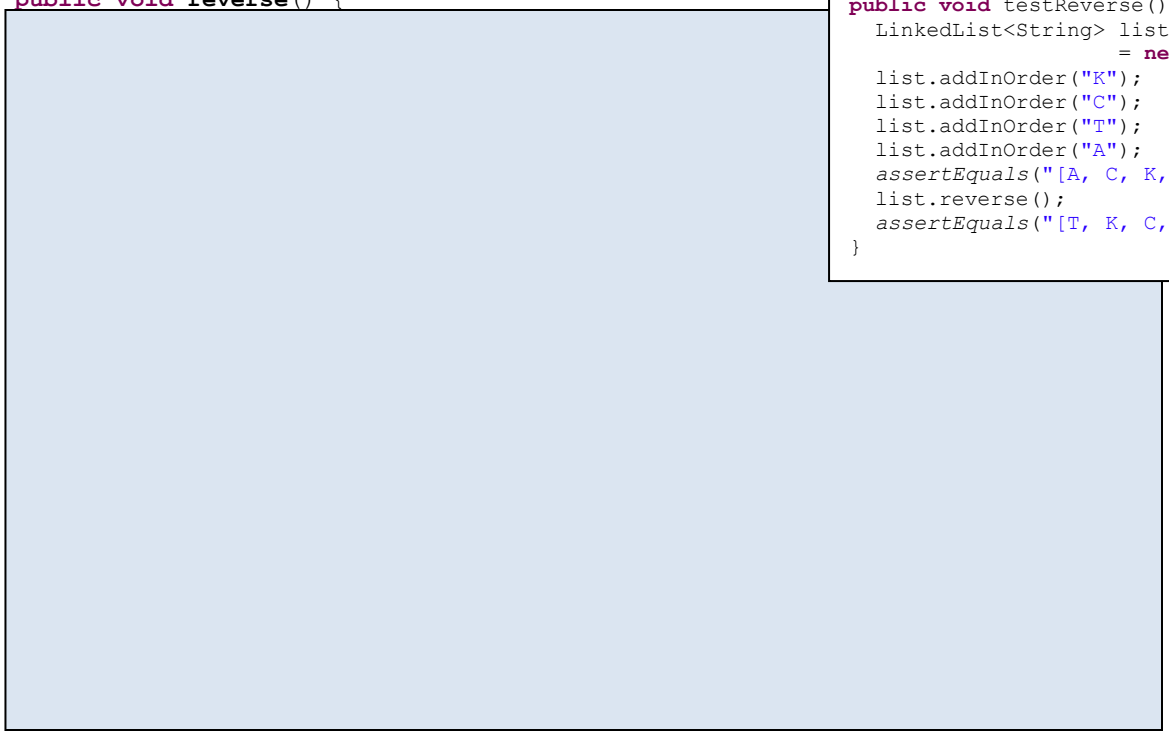
C Sc 227 Assignment: LinkedList reverse

To class LinkedList<E>, add a) method **addInOrder(E)** so all elements are add in their natural ordering. Also add b) method **reverse()** so all elements are stored in reverse order. The code in the box must compile and the assertions must pass. You cannot pretend there are other LinkedList methods. If you need another method, write it.

```
public class LinkedList<E> {

  private class Node {
    private E data;
    private Node next;

    public Node(E objectReference) {
      data = objectReference;
    }

    public Node(E objectReference, Node nextReference) {
      data = objectReference;
      next = nextReference;
    }
  }

  private Node first;
  private int n;

  public LinkedList() {
    first = null;
    n = 0;
  }

  public int size() {
    return n;
  }

  public E get(int index) {
    Node ref = first;
    for (int i = 0; i < index; i++)
      ref = ref.next;
    return ref.data;
  }

  public void addInOrder(E element) {  /* Rick does this is a code Demo */ }

  public void reverse() {
```



```
@Test
public void testReverse() {
  LinkedList<String> list
                    = new LinkedList<String>();
  list.addInOrder("K");
  list.addInOrder("C");
  list.addInOrder("T");
  list.addInOrder("A");
  assertEquals("[A, C, K, T]", list.toString());
  list.reverse();
  assertEquals("[T, K, C, A]", list.toString());
}
```