# Assignment 6

| | |
|---|---|
| reverse() | 25 pts. |
| reverseToString() and recReverseToString() | 25 pts. |
| Copy Constructor Method | 25 pts. |
| smallestFirst() | 25 pts. |

TOTAL: 100 pts.

***General Requirements***

- *All files you are turning in must be signed. Please add your name to the source code in comments. Unsigned work automatically loses **5 pts**.*
- *Add comments to the source code you are writing:*
  - *Describe the purpose of every variable*
  - *Explain the algorithm you are using for solution*
  - *Add proper comments for all methods. Include @param, @return, and @throws tags*
- *Uncommented work loses up to **25% of the grade.***
- *Archive the entire project using ZIP or RAR utility. Turn it in into the digital drop box.*

## reverse() Method for List class

Start with classes List.java and ListDemo.java that can be found next to this assignment. Add a new public method `void reverse()` to the class. The method must reverse the linked list data structure. You can implement either iterative or recursive solution. Test your method in ListDemo.java. Make sure it works for empty list as well as with a list with just one element.
**Requirement:** Complexity of your solution must be O(N).

## reverseToString() and recReverseToString()

Continue working with the same List.java class. Add two more methods to the class:

- public String reverseToString() method creates a string that holds all the integers in the list in reverse order.
  **Requirements:**
  - DO NOT reverse list, just traverse it and create a string that holds the elements in reverse order.
  - Your solution must be O(N).
  - This solution must be iterative.

- public String recReverseToString() is a RECURSIVE method with the same functionality - creates a string that holds all the integers in the list in reverse order.
  **Requirements:**
  - DO NOT reverse list, just traverse it and create a string that holds the elements in reverse order.
  - Your solution must be O(N).
  - This solution must be recursive.

## Copy Constructor Method

Continue working with the same List.java class. Add a copy constructor to create a deep copy of the list.

## smallestFirst() Method

Continue working with the same List.java class. Add `void smallestFirst()` method that moves the node with the smallest integer in the list to become the first node.