Lab 04- 30 points

For today's lab you will be designing 2 classes, one of which will use the other. You will also be creating two testers using the "Class (Unit) Testing" approach outlined in lecture and the textbook for each of your 2 classes.

Part 1 (15 points)

Design a class ***Message*** that models an e-mail message. A message has a recipient, a sender, and a message body representing the text of the message. Support the following methods:

1. A single constructor (no more) that takes in the sender and recipient.
2. A method `append` that appends a provided (parameter) line of text to the message body
3. Override the `toString` method so that it returns a string representation of the message as one long string like this: `"From: Harry Morgan\nTo: Rudolf Reindeer\n …(Message Body HERE)… \n".`
    - For this method, you must create and utilize two **non-changing/same across all object instances constant** strings for the substrings "From: " and "\nTo: ". Don't forget the extra \n at the end of the "To" line and the message, however that does not need to be done as a constant.

(8 points)

Write a tester program as discussed in class, called MessageTester, that uses this Message class to make a message and print it. Ensure you test all methods and complete all 4 testing steps for EACH method.

(4 Points)

(3 points for file names, comments, style, et cetera)

Part 2 (15 points)

Design a class, called ***Mailbox***, that stores (a collection of) e-mail messages, using the ***Message*** class of Part 1. Create a single appropriate constructor that makes a blank Mailbox. In terms of the data for the mailbox, look to the "Collecting Values" pattern presented in lecture. You must implement the following three methods that manipulate the Mailboxes data appropriately.

1. `public void addMessage(Message m)`
2. `public Message getMessage(int i)`
3. `public void removeMessage(int i)`

In addition, each individual mailbox (object instance) must have a **constant** string that represents its unique individual signature. This signature must be appended to the end of each message **after it is added to the mailbox.**

(8 points)

Write a tester program MailboxTester that tests all methods and various cases as we laid out in the 4 steps in the lecture and textbook. In this tester, you will have to create Messages to properly test the methods. Ensure you test all methods and complete all 4 testing steps for EACH method.
(4 Points)

(3 points for file names, comments, style, et cetera)

Submission

Turn in your 4 .java files to Canvas by the end of Tuesday (11:59pm):

1. Message.java
2. MessageTester.java
3. Mailbox.java
4. MailboxTester.java

If you have multiple submissions, you can submit a .zip file as the file names must correspond to the class names or you will lose points.

Grading will be based on conforming to the standards we reviewed in class as well as following the requirements of this lab.