Small Assignment #4
Due: Wednesday, 2/19/2025 by 11:59 PM

Submission. Submit this as a single PDF on Gradescope. Assign pages to questions.

All of these questions are based on the code provided, which includes a Book class and a LibraryModel class for managing a list of Books. In an application, this would be the Model, which would work together with a View (and possibly a Controller) to provide the functionality listed below.

The expected functionality:
- search for books by title
- search for books by author
- search for books by rating
- add a book
- set a book to read (indicating that the book has been read)
- rate a book on a scale of 1 to 5 (note that ratings for books are not required)
- get a list of all the books in the library
- get a list of all the books that have been read
- get a list of all the books that have not been read
- suggest a random unread book to read
- remove a book from the library
- retrieve the number of books in the library
- retrieve the number of unread books in the library
- retrieve the number of read books in the library

Question 1.
There is an antipattern called SPECULATIVE GENERALITY that means that you add functionality to your code because *you might need it later.* Since simple code and simple public interfaces are generally easier to use and maintain, this is something we should avoid.
Take a look at the two classes provided and determine if any of the methods should be removed. The functionality we currently need is listed above, so you should be able to determine what is necessary and what is not. List the methods in each class that should be removed. You should also actually remove them from the code.

Question 2.
We discussed guidelines for the use of visibility modifiers when designing classes that are well-encapsulated and have a clear public interface that is as narrow as possible. Take a look at the two classes and fix any issues with visibility modifiers and describe your changes here. Also give a brief explanation for why each change was a good idea.

Question 3.
We've also discussed the DUPLICATED CODE antipattern. Do you see any instances of this antipattern? If so, describe them, fix, them, and explain how you fixed them.

Question 4.
Another antipattern we've seen is TEMPORARY FIELD. Do you see any instances of this antipattern? If so, describe them, fix, them, and explain how you fixed them.

Question 5.
A third antipattern we've seen is PRIMITIVE OBSESSION. Do you see any instances of this antipattern? If so, describe them, fix, them, and explain how you fixed them.

Question 6.
We've been talking about the problem of escaping references, which can potentially lead to issues with encapsulation. Are there any escaping references that pose a problem for encapsulation? If so, identify them, explain why they are issues for encapsulation, fix them, and explain how you fixed them.

Question 7.
Copy and paste your final `Book.java` class into the document and submit it here.

Question 8.
Copy and paste your final `LibraryModel.java` class into the document and submit it here.

Question 9.
If you created any other classes as part of the process of fixing this code, copy and paste them and submit them here.