COS126 **Programming Exam 2** **Spring '19**

Instructions. You will have 50 minutes to create and submit two programs. Download the project zip file, which includes all the required files, from the **Exams** page. Do not open it until instructed.

Resources. You may use your book, your notes, your code from programming assignments and precepts, the code on the COS 126 course website, the booksite, and you may read Piazza. No form of communication is permitted (e.g., talking, texting, etc.) during the exam, except with course staff.

Grading. Your program will be graded on correctness, design, efficiency, and comments. You will receive partial credit for a program that correctly implements some of the required functionality.

Submissions. Submit your work using the link on the **Exams** page.

Discussing this exam. Due to travel for extracurriculars and sports, some of your peers will take this exam next week. Do not discuss exam contents with anyone who has not taken the exam.

This paper. In addition to submitting your code electronically, you *must* return this paper. Fill in the information below, then transcribe and sign the Honor Code pledge. You may do so now.

NAME: ____________________

NETID: ____________________

PRECEPT: ____________________

EXAM ROOM: ____________________

"I pledge my honor that I will not violate the Honor Code during this examination."

____

____

SIGNATURE: ____________________

Create a `Book` object that stores a book's title, author, and publication year. Then, create a client of `Book` called `BookShelf` whose only instance variable is of type `Queue<Book>`. Implement the following methods:

```
public class Book

// creates a new book
// throws a runtime exception if year is negative
public Book(String title, String author, int year)

// returns "[title], [author], [year]"
public String toString()

// returns book title, author, year (individually)
public String getTitle()
public String getAuthor()
public int getYear()

// returns true if this book is older than "that"
public boolean isOlderThan(Book that)

// 1. instantiates exactly two Book objects,
// 2. prints the output of toString() on both,
// 3. calls isOlderThan() and prints result
public static void main(String[] args)
```

```
public class BookShelf

// creates an empty bookshelf
public BookShelf()

// adds "book" to bookshelf
public void add(Book book)

// appends each Book's toString(),
// one per line (example below)
public String toString()

// returns true if bookshelf contains
// a book with the specified title
public boolean contains(String title)

// extra credit! (more info below)
public void removeDupes()

// tests BookShelf (more info below)
public static void main(String[] args)
```

For `BookShelf`, your `main()` method must read in book data from `StdIn`, remove duplicates (if any exist), and print the output of `toString()`. The input consists of multiples of three lines, with a title, author, and year each on their own line. Your output should condense this, such that each book's data fits on one line. For example:

```
% more books.txt
To Kill A Mockingbird
Harper Lee
1960
The Great Gatsby
F. Scott Fitzgerald
1925
The Catcher in the Rye
J.D. Salinger
1951
```

```
% java-introcs BookShelf < books.txt
To Kill A Mockingbird, Harper Lee, 1960
The Great Gatsby, F. Scott Fitzgerald, 1925
The Catcher in the Rye, J.D. Salinger, 1951
```

Reminders. Here are a few helpful reminders:

- To test whether `String a` has the same value as `String b`, use `a.equals(b)`, not `a == b`.
- To read an entire line of input from `StdIn`, use `StdIn.readLine()`.
- You can use `Integer.parseInt()` on the result of `StdIn.readLine()`.
- To add a newline to a `String`, append `"\n"`.

Extra credit. The `removeDupes()` method is challenging. It will only be worth a few points. Only attempt this method when you've finished all the other methods. A book is considered a duplicate if its title, author, and year exactly match another book's title, author, and year. `removeDupes()` must remove all duplicates for full credit.