CMSC203, Assignment 3

Spring 2017

**Concepts Utilized in this Project**

* UML diagrams
* Menu driven program
* Java fundamentals, including decision structures, loops
* Constructors, Overloaded constructors
* toString method
* Formatting output with DecimalFormat
* Random Class
* Java objects & classes

**Overview**

Write an application that simulates adding new books to the Amazon database.

The Application repeatedly asks the information of a new book and once the information is gathered, it will be displayed to the user. Note that this application does not save the information of the book.

Information of the book includes its title, author, the number of ratings, the sum of all the rating (the rating given to a book is 1 or 2 or 3 or 4), its price and whether the book has a hardcover or not.

When adding a new book, the user can choose to provide only the book title and its author. The rest of the information will be set to default values by the program: Price to random number between 1 and 10, the number of ratings and the sum of ratings to 0 and no hardcover for the book.

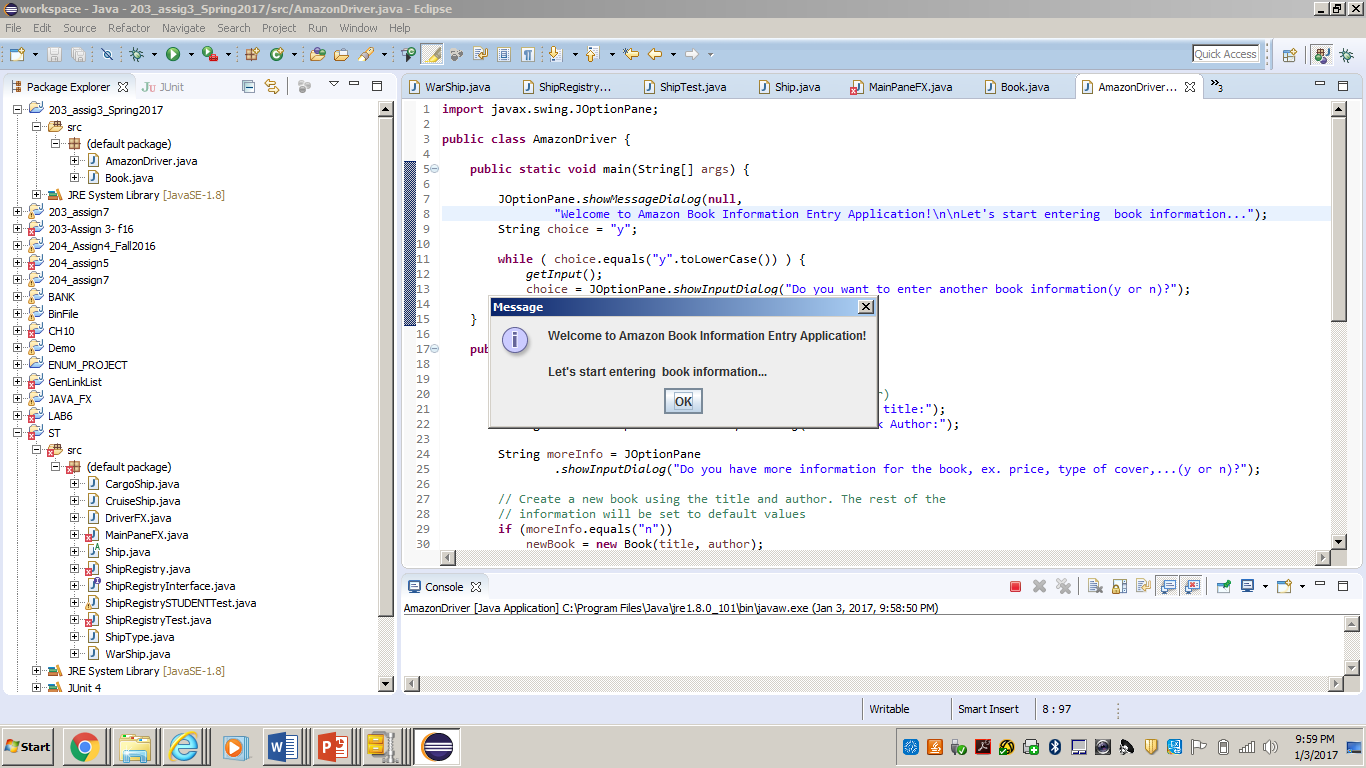
The user can also choose to enter all the book’s information.

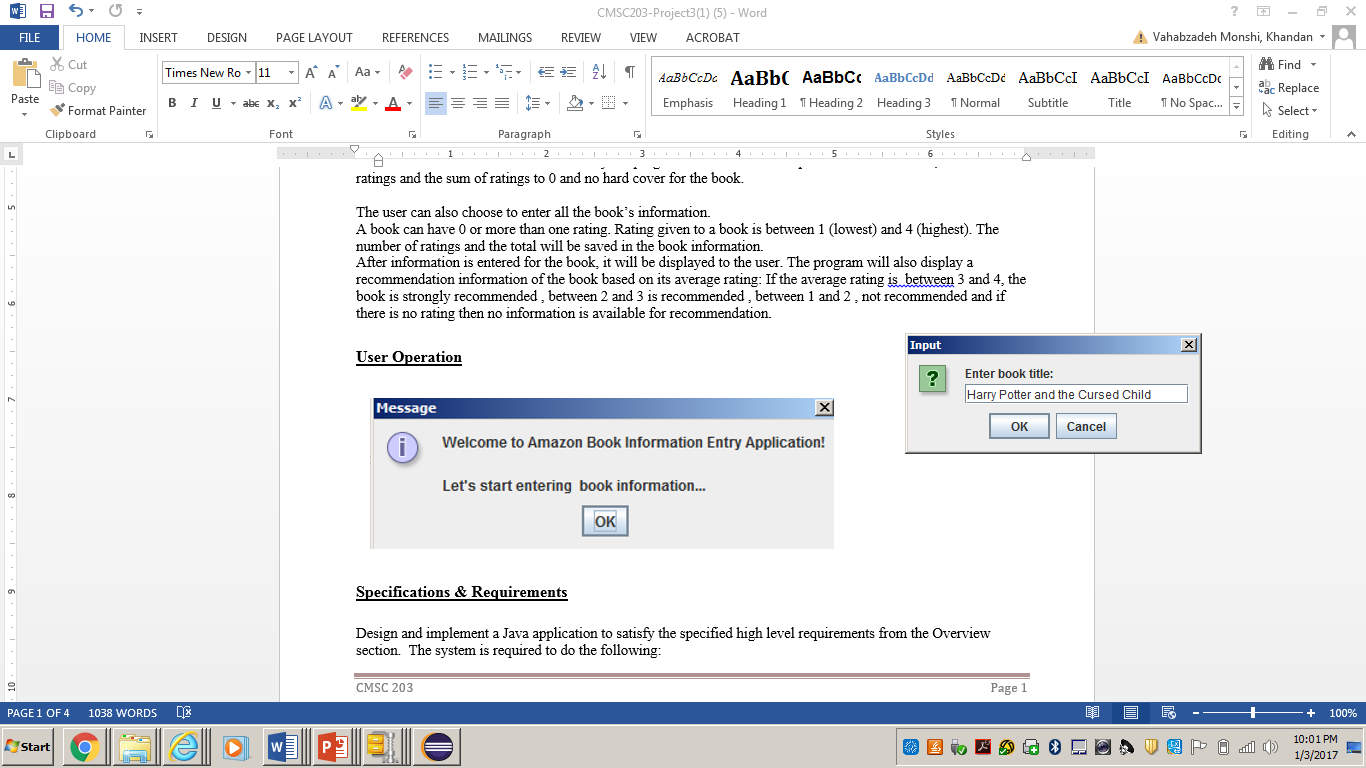
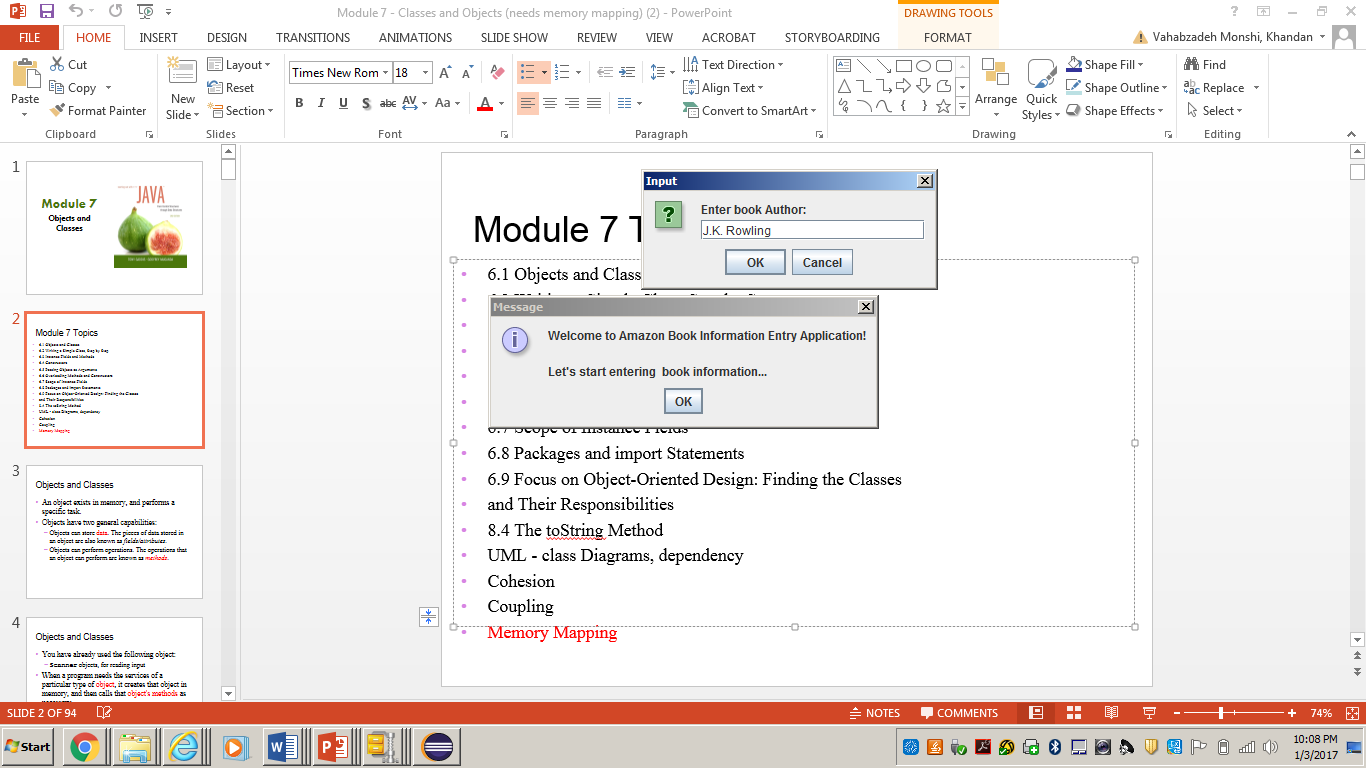
A book can have 0 or more than one rating. Rating given to a book is a number between 1 (lowest) and 4 (highest). The number of ratings and the total will be saved in the book’s information.

After information is entered for the book, it will be displayed to the user. The program will also display a recommendation information for the book based on its average rating: If the average rating is between 3 and 4, the book is strongly recommended, between 2 and 3 is recommended, between 1 and 2, not recommended and if there is no rating then no information is available for recommendation.

## User Operation

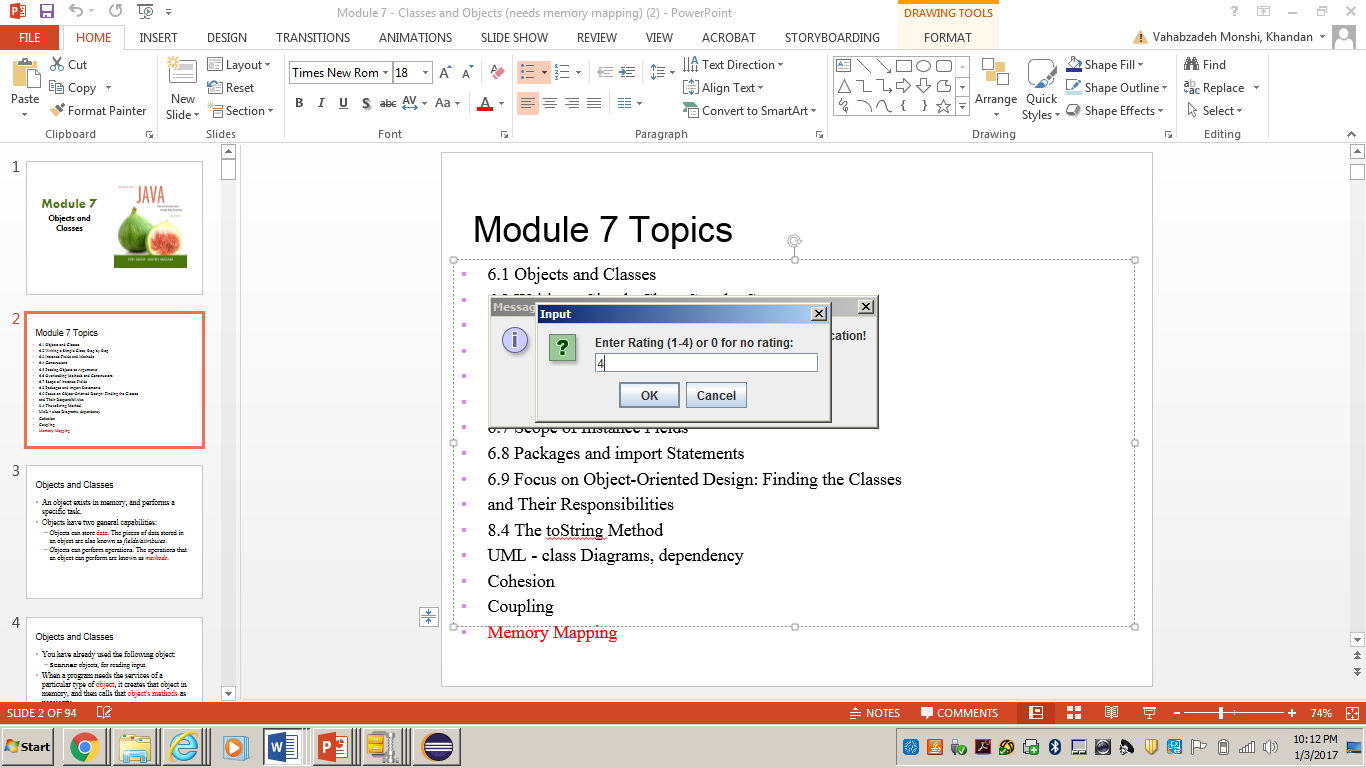
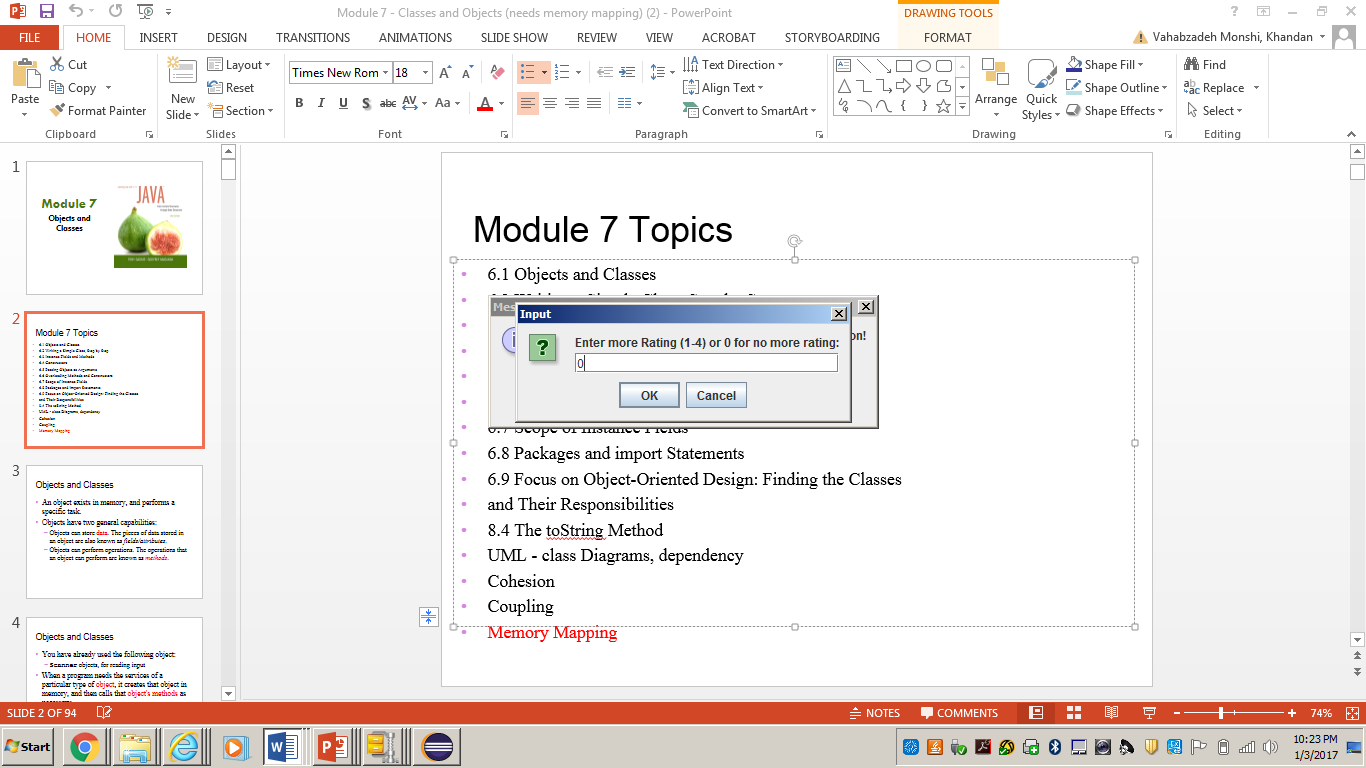
Following is a Sample of program run:

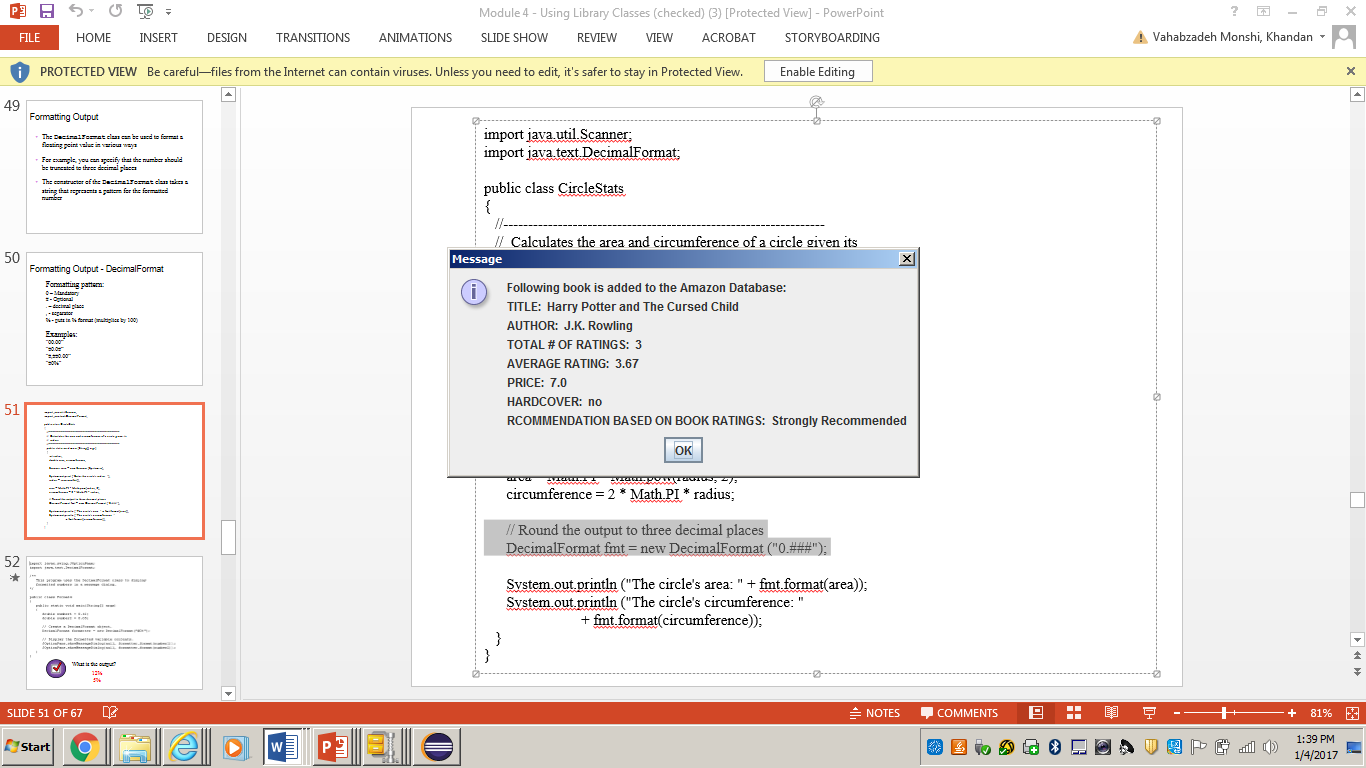


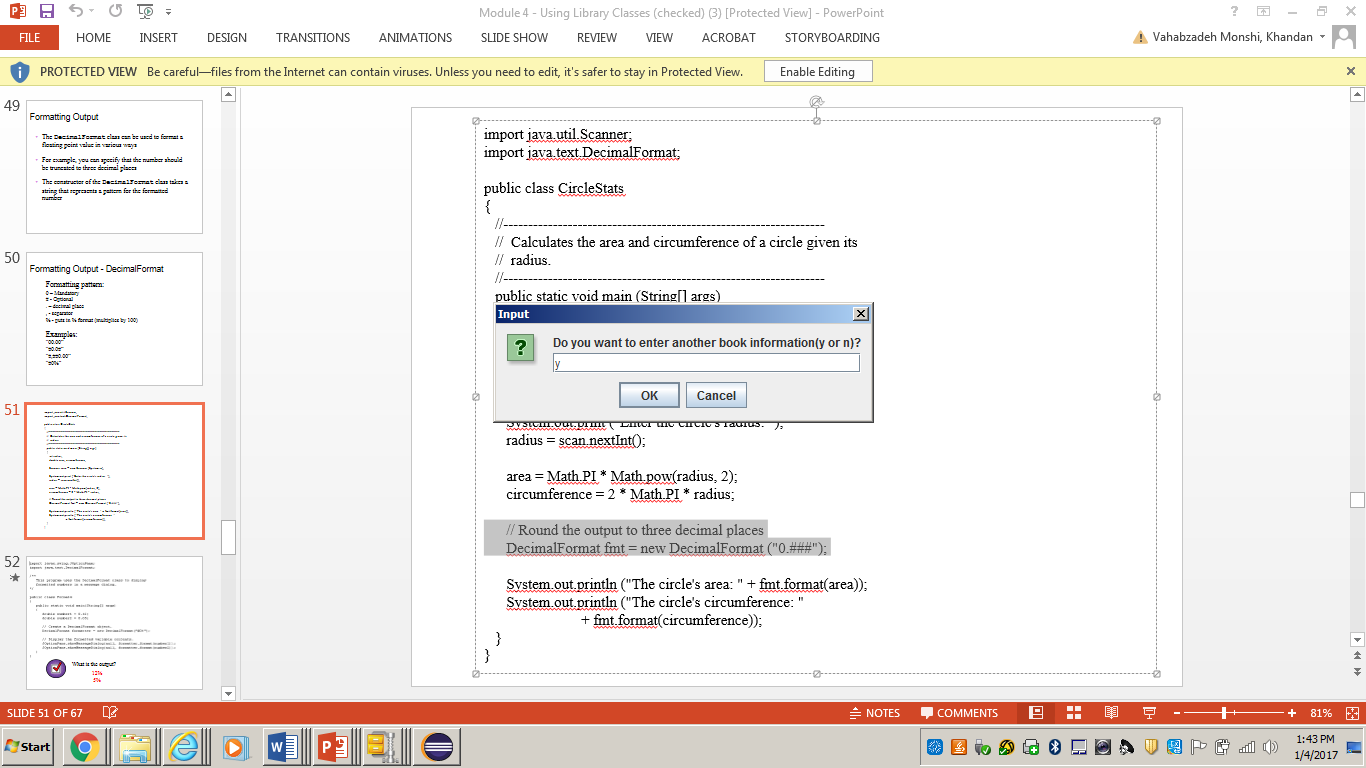


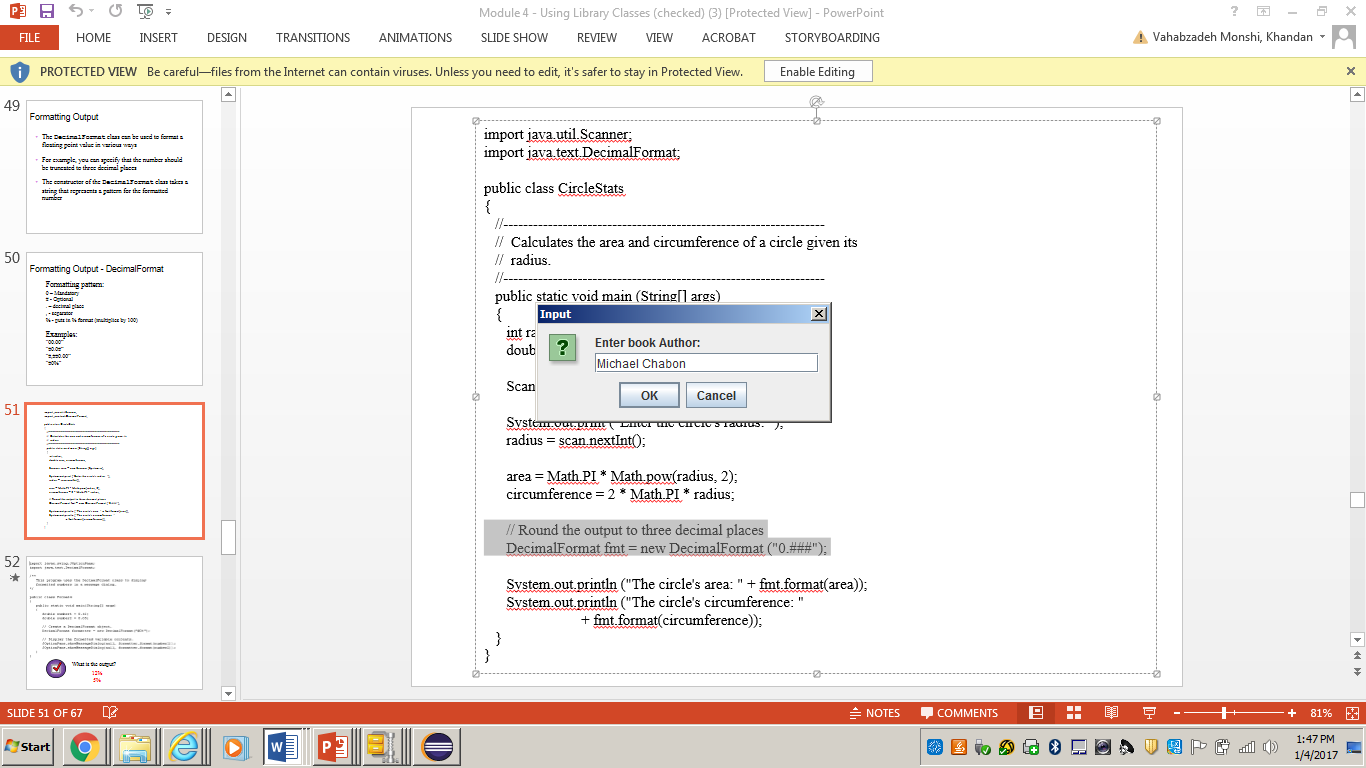
## 

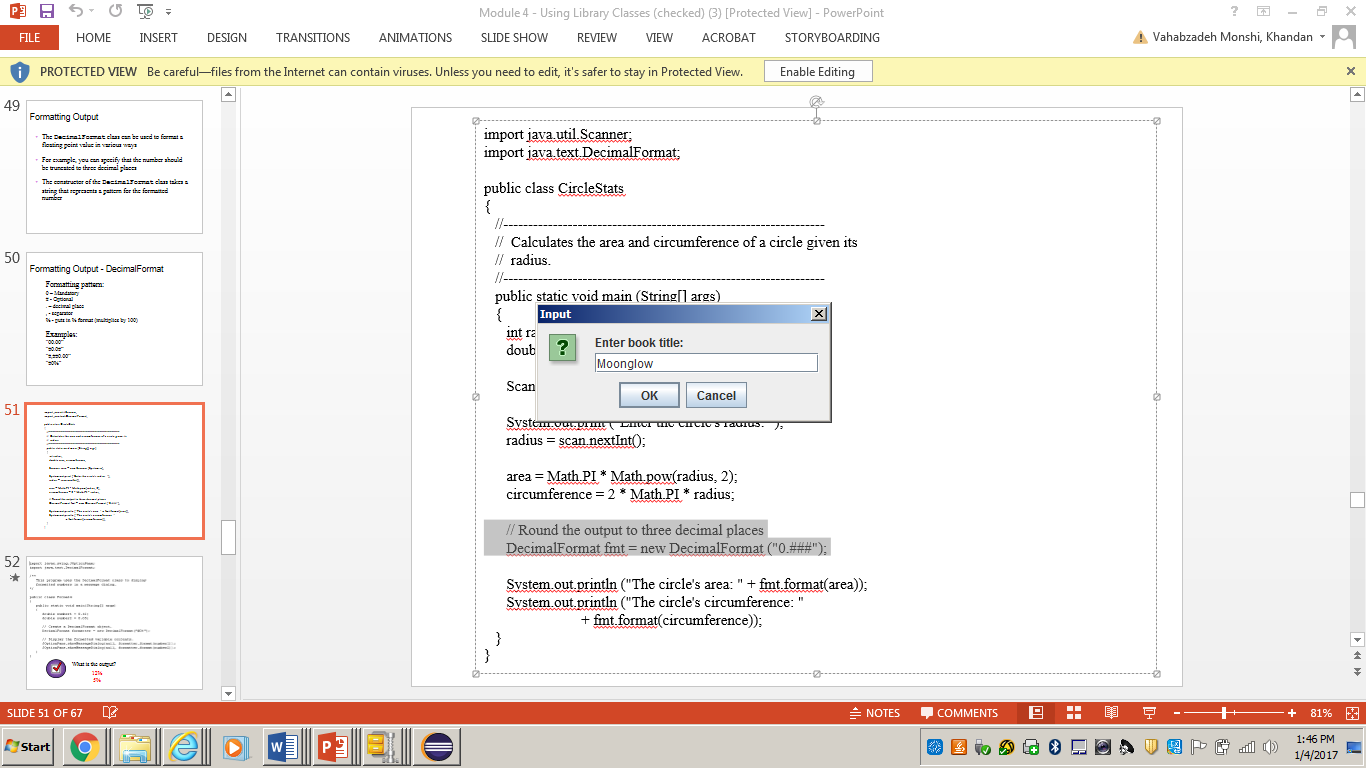
## 

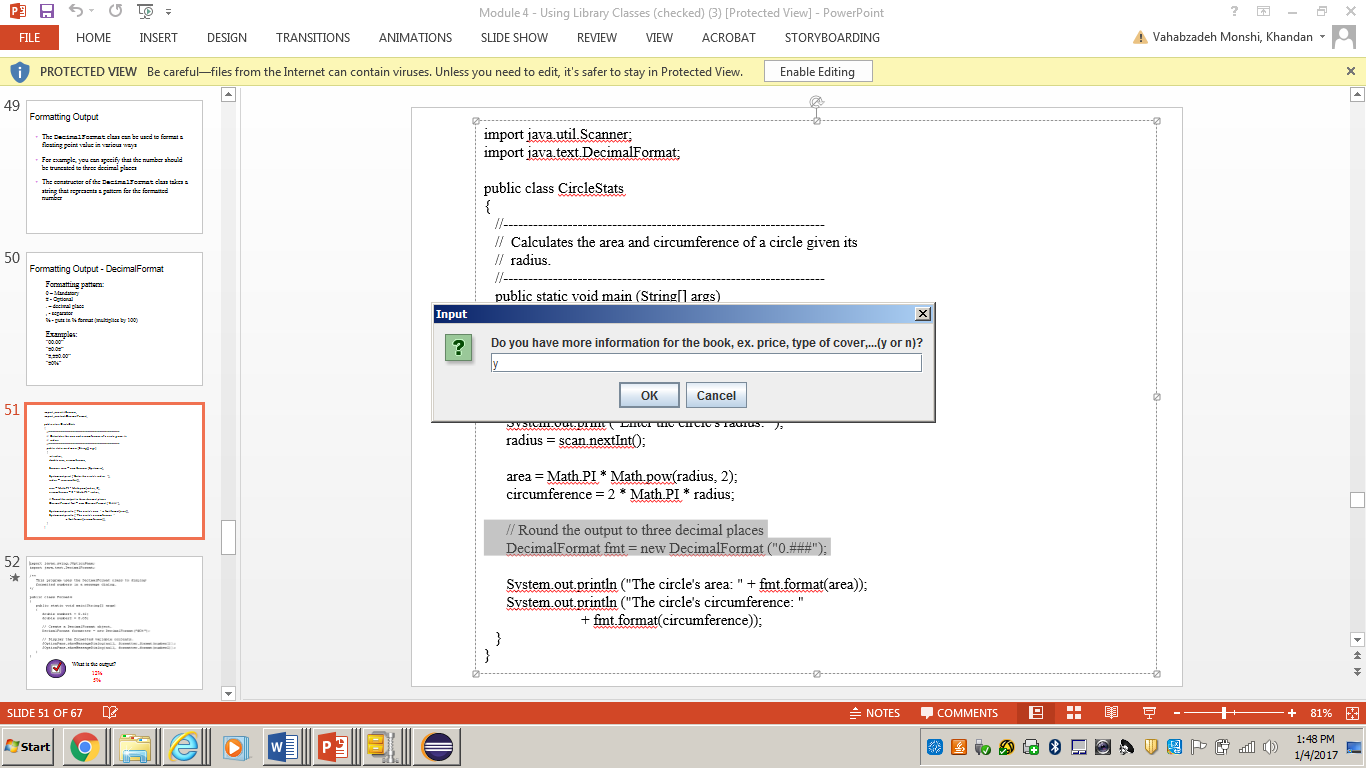


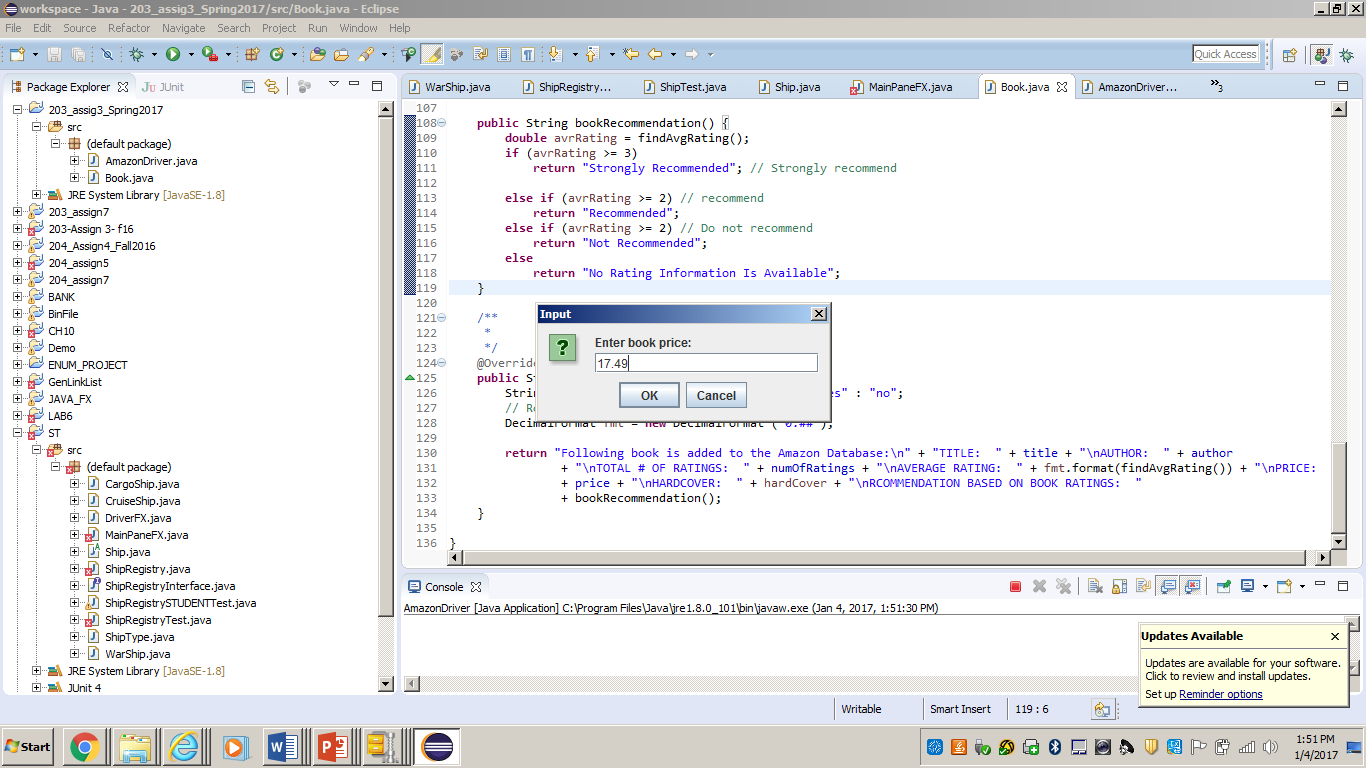


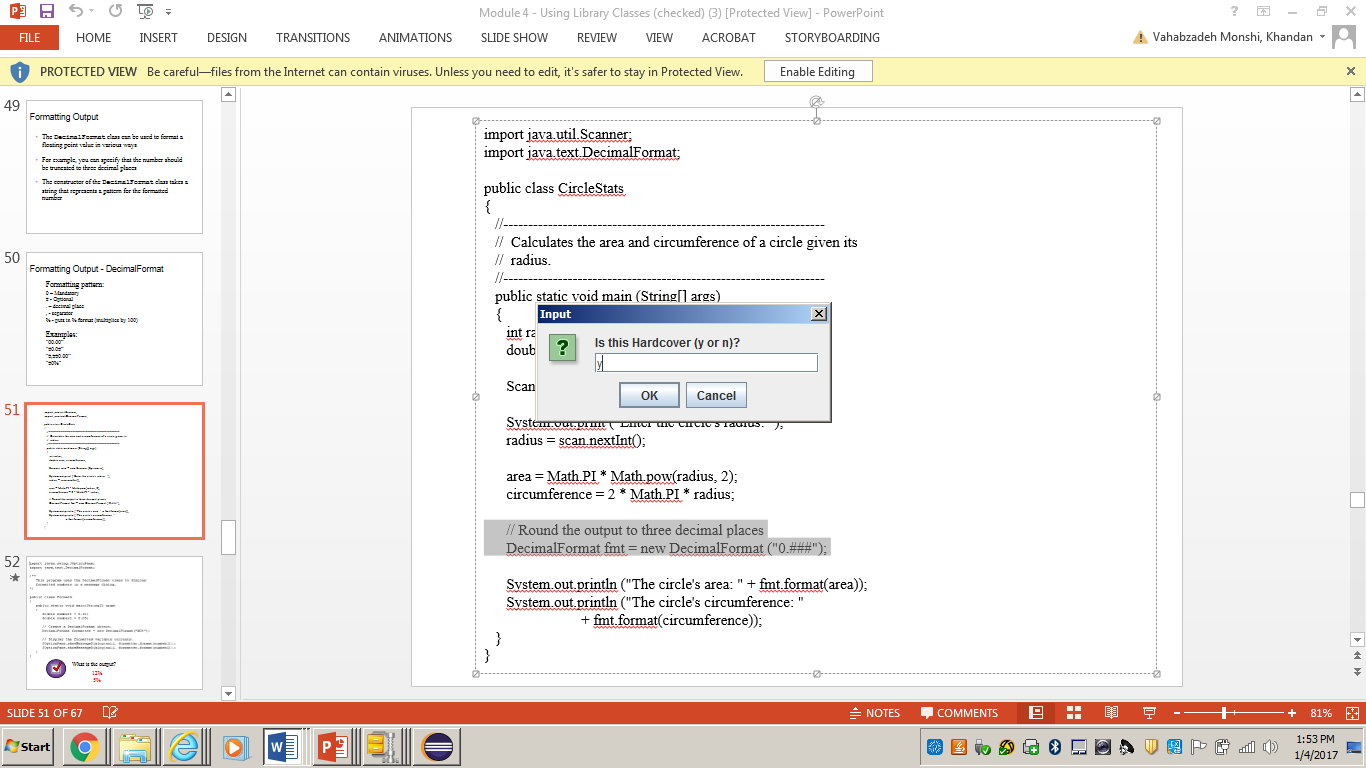
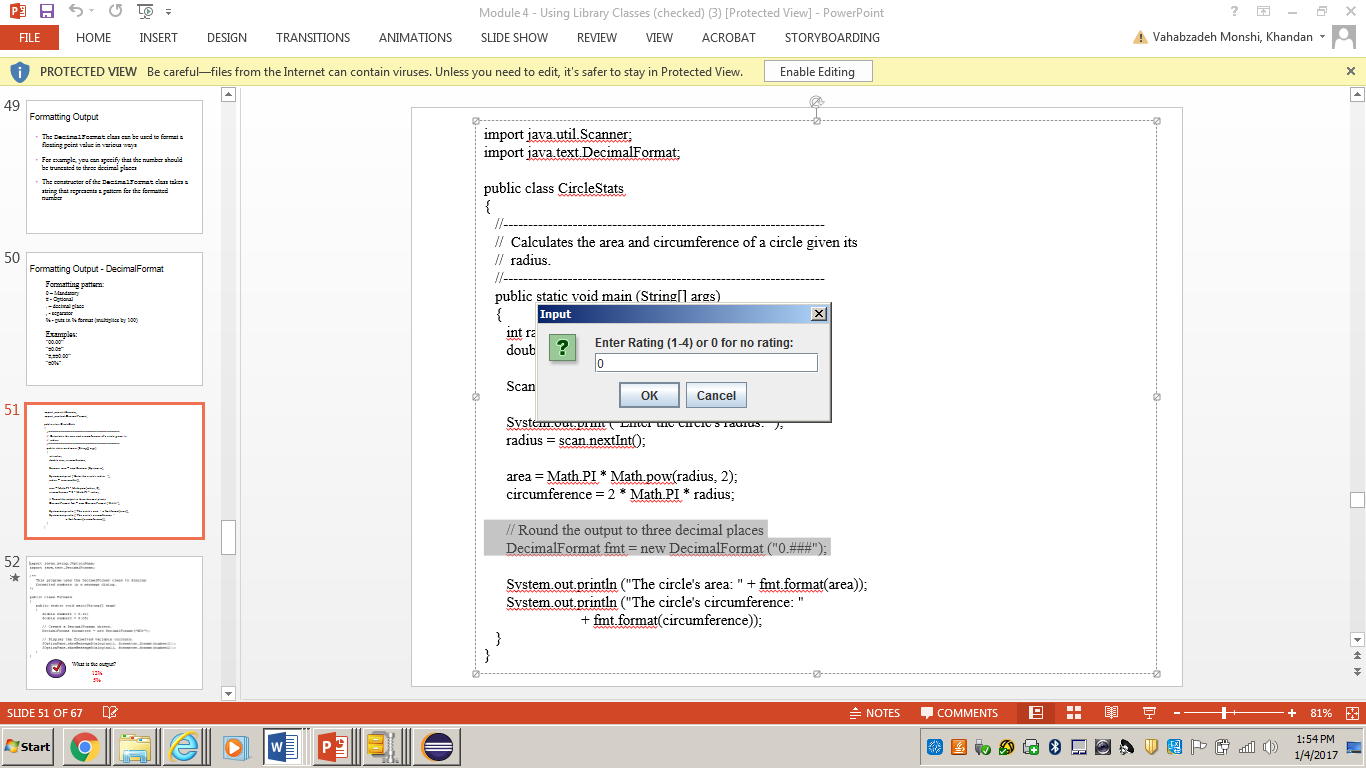


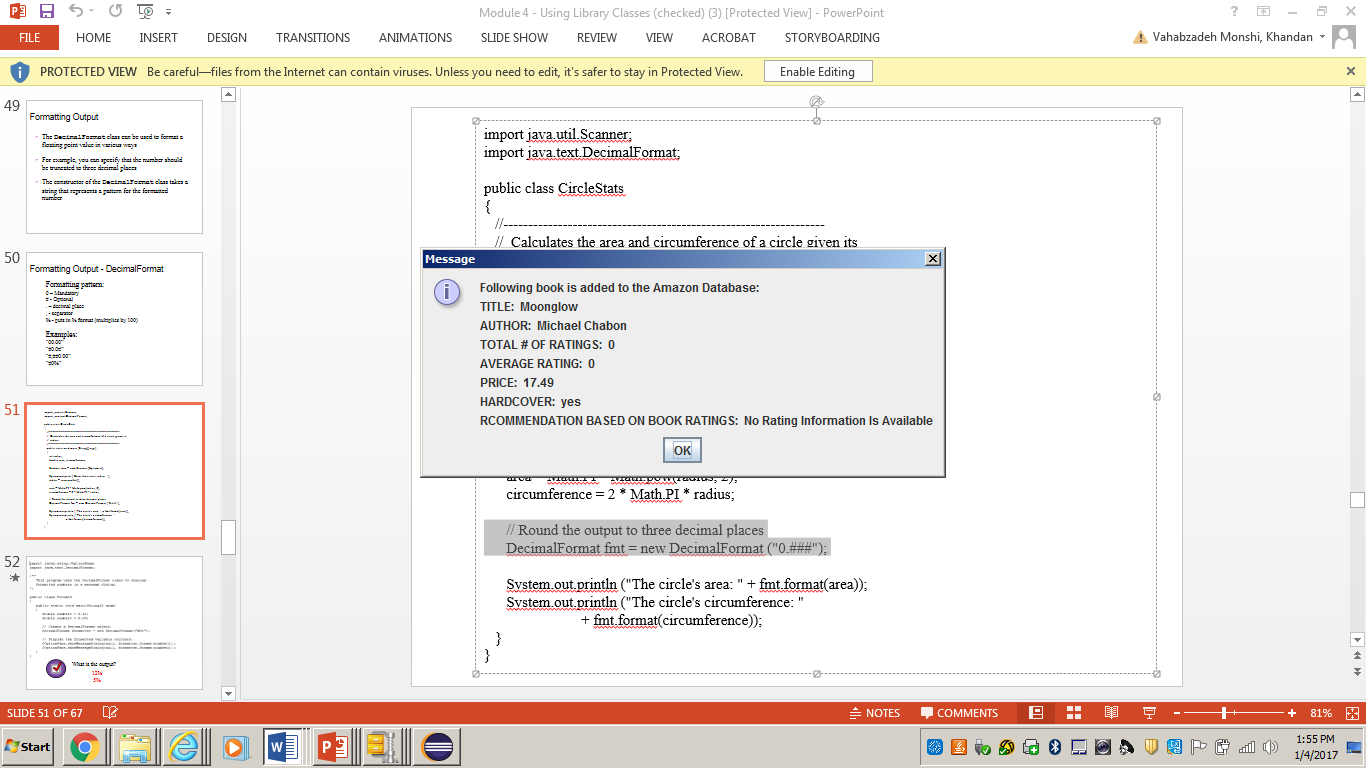












## Specifications & Requirements

Design and implement a Java application to satisfy the specified high level requirements from the Overview section. The system is required to do the following:

**Data Element Class - Book**

* Create a class with the given information (fields) in the overview section.
* Create a **constructor** that takes book’s title and author and creates a book instance with the provided information. The number of ratings, Total rating will be set to zero, price is set to random number between 1 and 10 and the book has not hardcover.
* Create another **constructor** that takes the information for the title, author, price and whether the book has hardcover or not and creates a book instance with the given information. The number of ratings, Total rating will be set to zero.
* A method **addRating** that takes the rating for the book and adds it to the total rating as well as incrementing the number of ratings for this book.
* A method called **findAvgRating** that returns the average rating for this book or 0 if there is no rating for the book.
* A method called **bookRecommendation** that returns a string based on the average rating of the book. The book is “strongly recommended” for average rating between 3 and 4, “Recommended” for average rating between 2 and 3(exclusive) , “Not Recommended” for average rating between 1 and 2 (exclusive) , and if there is no rating return “No Information Is Available For Recommendation”.
* A **toString** method that returns the string representation of a Book object: title, author, number of ratings, average rating, price and book recommendation.
* Add any necessary **getter** or **setter** methods.

**Driver Class - Amazon**

* This is the driver class for Book that contains a main method.
* Create a method **getInput()** that returns a Book object. This method allows the user to enter book information, uses the information to construct a Book object and returns it.
* This class contains a main method which continues asking the book information as long as user chooses to enter more information and displays the information of the book to the user. Refer to the program sample run for more clarification.
* Add any necessary methods to modularize your code.

**Deliverables / Submissions:**

Submit a compressed file containing the follow (see below): The Java application (it must compile and run correctly); Javadoc files in a directory; a write-up as specified below. Be sure to review the provided project rubric to understand project expectations.

* UML diagram of the Book class
* Any assumptions that you are making for this project
* In three or more paragraphs, highlights of your learning experience

Each student must submit two compressed (.zip) files to the assignment’s folder on Blackboard for this project with the following contents:

* LastNameFirstName\_Assignment3\_Complete.zip, a compressed file in the zip format, with the following:
  + Write up (Word document)
  + UML Diagram (Word or jpg document)
  + Javadoc (directory)
    - File1.html (example)
    - File2.html (example)
  + src (directory) 
    - File1.java (example)
    - File2.java (example)
* LastNameFirstName\_Assignment3\_Moss.zip, a compressed file containing one or more Java files:
  + File1.java (example)
  + File2.java (example)
  + This folder should contain Java source files only

**Grading Rubric**

**CMSC 203 Assignment #3**

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Overview:**

There are two parts to the rubric. First, the project is graded on whether it compiles, whether it runs without errors, and whether it satisfies the specifications. These points add up to 100. Second, the score is decremented if various additional requirements are not met, e.g., no UML diagram, uses constructs that are not allowed, etc.

**PROGRAMMING** (100 pts)

Compiles 40 pts \_\_\_\_\_

Accuracy

Passes private instructor tests 15 pts \_\_\_\_\_

Execution: runs without errors (either run-time or logic errors) 45 pts \_\_\_\_\_

Possible Sub-total 100 pts \_\_\_\_\_

**REQUIREMENTS** (Subtracts from Programming total)

**Documentation:**

Documentation within source code is missing or incorrect - 5 pts \_\_\_\_\_

Description of what class does is missing

Author’s Name, @author, is missing

Methods not commented properly using Javadoc @param, @return

UML diagram is not provided - 5 pts \_\_\_\_\_

Test cases are not provided - 5 pts \_\_\_\_\_

Lessons-learned paragraphs are not provided - 5 pts \_\_\_\_\_

**Programming Style:**

Incorrect use of indentation, statements, structures - 4 pts \_\_\_\_\_

User interface

Not clear to user how data is to be entered; UI does not follow requirements - 10 pts \_\_\_\_\_

Output is easy to understand - 4 pts \_\_\_\_\_

**Design:**

Does not use classes specified (Book and Amazon classes) - 6 pts \_\_\_\_\_

Classes do not have the functionality specified - 5 pts \_\_\_\_\_

No Existence of 2 class constructors as explained in overview section -4 pts \_\_\_\_\_

No check for invalid book rating -1 pts \_\_\_\_\_

No Loop for entering more book information -2 pts \_\_\_\_\_

No Loop for entering more than one rating for the book -2 pts \_\_\_\_\_

Average rating does not round to 2 decimal places -1 pts \_\_\_\_\_

Price is not generated randomly when the price is not provided -1 pts \_\_\_\_\_

Possible decrements: -60 pts \_\_\_\_\_

Possible total grade: 100 pts \_\_\_\_\_