**Assignment 2**

Josue Ponce

Montgomery College

2/23/2018

Author’s Note

This project report was prepared CMSC-203 CRN #30672, taught by professor Ahmed Tarek

**Table of Contents**

Book.Java Pseudo-Code…………………………………………………………………………2

Book.Java Copy of Code………………………………………………………………………...4

Amazon.Java Pseudo-Code……………………………………………………………………...9

Amazon.Java Copy of Code…………………………………………………………………….11

Test Cases……………………………………………………………………………………….14

Test #1………………………………………………………………………………………......14

Test 1 screenshots………………………………………………………………………………16

Test #2………………………………………………………………………………………......20

Test 2 Screenshots………………………………………………………………………………21

Test #3…………………………………………………………………………………………...23

Test 3 Screenshots………………………………………………………………………………24

**Book.Java** **Pseudo-Code**

* Declare public class book
* Declare private string title which will hold title of book
* Declare private String Author which will hold the book’s author.
* Declare private integer numberofRatings which will hold the number of ratings.
* Declare private integer totalRatings which will hold the total amount of ratings
* Declare double price which will hold the cost of the book.
* Declare private string hasHardCover which will hold if book has hardcover or not.
* Public book () Constructor class will initiate the title and author of the book with the value in its parameters. The remaining properties will be assigned default values such as price which will be random, numberOfRatings, totalRatings, and hasHardCover.
* Copy of public book() Constructor class with values such as title, author, price and hard cover (h) indicator.
* Declare setter method setTitle() to input the title of the book. String n in the parameters will contain the title of the book.
* Declare setter method setAuthor() to input the author of the book. String n in the parameters will contain the author’s name.
* Declare setter method setPrice() to set the price of the book. String n in the parameters will contain the book’s price.
* Declare setter method setCover() to determine if the book has a hardcover or not. String h in the parameters will contain information on whether the book has a hard cover or not.
* Declare addRating() method which will increase the number of ratings count and also calculate the totalRating.
* Declare findAvgRating() getter which will do the following:
  + - Declare double d;
    - Declare double result;
      * If numberOfRatings is equal to 0,

Then

Return 0.

* + - * Else
      * d will equal the totalRating/ NumberOfRatings.
      * Result will round by 2 decimal points and multiply d by 100.0 and divide by 100.0.
      * Return result.
* Declare getter String bookRecommendation() which will do the following:
* Double avgRating will call and be equal to method findAvgRating
* If average rating is >= to 3 && average rating is <= 4,

Then

Return “Strongly Recommended”;

* Else if average rating >=2 and average rating is <=3,

Then

Return “Recommended.

* Else if average rating >=1 and <= 2

Then

Return “Not Recommended”

* Else if average rating is equal to 0

Return “No information is available for recommendation.

* Else

Return “Unexpected Data found”

* Declare String toString() getter method which will return title, author numberOfRatings, findAvgRating, Price, hasHardCover, and recommendation comment. Method will call findAvgRating, and bookRecommendation methods.

**Book.Java Copy of Code**

/\*\*

\* Book class is a representation of a book with title, author, number of ratings received, calculates

\* total number of ratings, average rating, recommendation comment, price and whether the book has hard cover or not.

\*

\* @author Josue Ponce

\* @version 1.0

\*/

**public** **class** **Book** {

/\*\* Attribute title holds the title of the book. \*/

**private** String title;

/\*\*Attribute author holds the author of the book. \*/

**private** String author;

/\*\*Attribute numberOfRatings holds the number of ratings input by user. \*/

**private** **int** numberOfRatings;

/\*\*Attribute totalRating holds the total amount of ratings that were input by the user. \*/

**private** **int** totalRating;

/\*\*Attribute price holds the price of the book.\*/

**private** **double** price;

/\*\* Attribute hasHardCover holds if the book has a hard cover or not. \*/

**private** String hasHardCover;

/\*\*

\* Constructor class that initiates the title and author with the value in parameters.

\* The remaining properties are assigned default values.

\* @param title - represents the title of the book.

\* @param author - represents the author of the book.

\*/

**public** **Book**(String title, String author) {

**double** randy = (**float**)(Math.random() \* **10**);

**double** result;

result = Math.round(randy \* **100.00**) / **100.00**;

**this**.title = title;

**this**.author = author;

numberOfRatings = **0**;

totalRating = **0**;

price = result;

**this**.hasHardCover = "No";

}

/\*\*

\* Constructor class with values such as title, author, price and hard cover (h) indicator.

\* @param title - string - represents the title of the book.

\* @param author - string - represents the author of the book.

\* @param price - double - represents the price of the book.

\* @param h - String - indicates whether the book has hard cover or not.

\*/

**public** **Book**(String title, String author, **double** price, String h) {

**this**.title = title;

**this**.author = author;

numberOfRatings = **0**;

totalRating = **0**;

**this**.price = price;

h = "Yes";

**this**.hasHardCover = h;

}

/\*\*

\* setter method to input the title of the book.

\* @param n is a String containing the title of the book.

\*/

**public** **void** **setTitle**(String n) {

**this**.title = n;

}

/\*\*

\* setter method to input the author of the book.

\* @param n is a String containing the name of the book's author.

\*/

**public** **void** **setAuthor**(String n) {

**this**.author = n;

}

/\*\*

\* setter method to set the price of the book.

\* @param n is a double containing the price of the book.

\*/

**public** **void** **setPrice**(**double** n) {

**this**.price = n;

}

/\*\*

\* setter method determining whether the book has hard cover or not.

\* @param h is a String containing information on whether the book has hard cover or not.

\*/

**public** **void** **setCover**(String h) {

**this**.hasHardCover = h;

}

/\*\*

\* method that increases the number of ratings count and also calculates the rating total.

\* @param rating - int - represents the rating given for the book by a single entity and adds the total number of ratings input into the program.

\*/

**public** **void** **addRating**(**int** rating) {

totalRating += rating;

numberOfRatings++;

}

/\*\*

\* returns 0 if the number of ratings is zero else returns the average of ratings result.

\* @return average rating value - double.

\*/

**public** **double** **findAvgRating**() {

**double** d;

**double** result;

**if** (numberOfRatings == **0**) {

**return** **0**;

}

**else**

d = ((**double**) totalRating) / numberOfRatings;

result = Math.round(d \* **100.0**) / **100.0**;

**return** result;

}

/\*\*

\* returns a string value describing whether the book is recommended or not.

\*

\* @return string - recommendation comment.

\*/

**public** String **bookRecommendation**() {

**double** avgRating = findAvgRating();

**if** (avgRating >= **3** && avgRating <= **4**) {

**return** "Strongly Recomended";

}

**else** **if** (avgRating >= **2** && avgRating < **3**) {

**return** "Recommended";

}

**else** **if** (avgRating >= **1** && avgRating < **2**) {

**return** "Not Recommended";

}

**else** **if** (avgRating == **0**) {

**return** "No Information Is Available For Recommendation";

}

**else**

**return** "UNEXPECTED DATA FOUND";

}

/\*\*

\* returns the entire information of the book in a single string.

\*

\*@return title, author, numberofRatings,findAvgRating(),price, hasHardCover, and bookRecommendation().

\*/

**@Override**

**public** String **toString**() {

**return** "The Following book has been added to the Amazon Database: \n" +

"TITLE: " + title

+

"\nAUTHOR :" + author

+

"\nTOTAL # OF RATINGS: " + numberOfRatings

+

"\nAVERAGE RATING: " + findAvgRating()

+

"\nPRICE: $" + price + "\nHARDCOVER: " + hasHardCover

+

"\nRecommendations based on book ratings: " + bookRecommendation();

}

}

**Amazon.Java Pseudo-Code**

* import javax.swing.JOptionPane;
* Initialize Amazon class driver.
* Declare main method to create an entry point for the Amazon application which will be an array of command line arguments that will be passed to this method in order to obtain calculations from book.java and user input from amazon.java. Main method will print out the book information when user declines to enter another book.
* Declare Book getInput() which will obtain user inputs and return data to main method where all book information will be printed out.

The following will be done within book getInput() method:

* + Display welcome message.
  + Intialize String book2 and Book data.
  + Test expression loop will keep looping until the user declines to enter another book.
  + Prompt user to enter the title of the book
  + Prompt user to enter the book’s author
  + If user wants to enter more book data,

then

Prompt user to enter the price of the book

Ask user if the book has a hardcover

Create a new book with title, author, price, and hardcover.

* + Else create a new book with just the title and author.

o Test expression loop will keep looping until user enter rating(1-4) or 0 for no ratings

If rating is equal to 1,2,3,4,

then

Count the number of ratings per input

Add another test expression loop which will keep looping until the user declines to enter more ratings

Prompt user to enter more ratings(1-4) or 0 for no more ratings.

If more ratings are equal to 0, then

End loop and display book information

Else if more ratings >= to 1 && < = to 4, then

Count the number of ratings per input

Else display error message if input is incorrect

* + Else if rating is equal to 0
  + End loop and display book information
  + Else display error message if input is incorrect
  + Prompt user to enter another book
  + End loop until user declines to enter another book
  + Return data

**Amazon.Java Copy of Code**

**import** **javax.swing.JOptionPane**;

/\*\*

\* Amazon class tests the Book class by collecting information from the user.

\* @author Josue Ponce

\* version 1.0

\*/

//Declaration of a class Driver

**public** **class** **Amazon**

{

/\*\*

\* Amazon application entry point

\* @param args

\* array of command-line arguments passed to this method in order to print out all information about book.

\*/

//Declaration of a main method

**public** **static** **void** **main**(String[] args)

{

//Declaration of a book b

Book b = **new** Amazon().getInput();

JOptionPane.showMessageDialog(**null**, b.toString(), "Message", JOptionPane.INFORMATION\_MESSAGE);

}

/\*\*

\* Method Book getInput() asks the user for the title of book, author of book, more book information

\* input rating for book,

\* @return data to main where all book information will be printed out to user.

\*/

**public** Book **getInput**()

{

JOptionPane.showMessageDialog(**null**, "Welcome to Amazon Book Information Entry Application! \n" +

"\nLet's start by entering book information... ");

//Declaration of book data

/\*\* Attribute book2 asks if the user wants to input another book into the database\*/

String book2;

//Declaration of book data

Book data;

**do** {

String title = JOptionPane.showInputDialog("Enter the title of the book:");

String author = JOptionPane.showInputDialog("Enter the book's author: ");

String option = JOptionPane.showInputDialog("Do you have more information for the book ex.price,type of cover...(Y or N)?");

**if** (option.startsWith("y"))

{

String price = JOptionPane.showInputDialog("Enter the book's price");

String hasHardCover = JOptionPane.showInputDialog("Is the book hardcover? (Y or N)");

**if** (hasHardCover.startsWith("Y")) {

}

data = **new** Book(title, author, Double.parseDouble(price), hasHardCover);

} **else** {

data = **new** Book(title, author);

}

**int** rating;

**do** {

rating = Integer.parseInt(JOptionPane.showInputDialog("Enter Rating(1-4) or 0 for no rating:"));

**if** (rating == **1** || rating == **2** || rating == **3** || rating == **4**) {

data.addRating(rating);

**do** {

rating = Integer.parseInt(JOptionPane.showInputDialog("Enter more Rating(1-4) or 0 for no more rating:"));

**if** (rating == **0**) {

JOptionPane.showMessageDialog(**null**, data.toString(), "Message", JOptionPane.INFORMATION\_MESSAGE);

} **else** **if** (rating >= **1** && rating <= **4**) {

data.addRating(rating);

} **else**

JOptionPane.showMessageDialog(**null**, "Invalid rating", "Error", JOptionPane.ERROR\_MESSAGE);

} **while** (rating != **0**);

} **else** **if** (rating == **0**) {

JOptionPane.showMessageDialog(**null**, data.toString(), "Message", JOptionPane.INFORMATION\_MESSAGE);

} **else**

JOptionPane.showMessageDialog(**null**, "Invalid rating", "Error", JOptionPane.ERROR\_MESSAGE);

} **while** (rating != **0**);

book2 = JOptionPane.showInputDialog("Do you want to enter another book? ");

} **while** (book2.equalsIgnoreCase("y"));

**return** data;

}

}

**Test Cases**

The following tests were conducted to ensure that the program was free of logical errors and worked as intended. For each test, the mathematical output for the average rating, total amount of ratings, were calculated by hand and then recommendation comment was determined based on the average rating score that was calculated by hand. Number of invalid inputs were also added in hand calculations to ensure that the program only calculated valid inputs for book ratings. After hand calculating different values for each test, screenshots were taken for every step of the test to visually show the program’s output. The highlighted sections in the hand calculated section represent inputs and expected outputs. For the program sections it represents inputs as well as the program output or displayed results.

**Test #1: Hand Calculated Results**

**---------------------------------------------------------------------------------------------------------------------**

# of invalid inputs: 2

Invalid input numbers: -1,9

Input ratings for book: 4,3,4

Hand calculated Total # of ratings: 3

Hand Calculated average rating score: 3.67

Pre-Determined recommendation comment based off the average rating Score: Strongly Recommended.

Do you want to enter another book?

Yes

# of invalid inputs: 0

ratings for book: 0

Total number of ratings: 0

Hand calculated average rating score: 0.0

Pre-determined recommendation comment based off the average rating score:

No rating information is available

**Test #1 Program Inputs and Outputs**

**---------------------------------------------------------------------------------------------------------------------**

Enter the title of the book: Harry Potter and the Cursed Child

Enter the author of the book: J.K. Rowling

Do you have more information for the book ex.price, type of cover…(Y or N)?: n

Enter rating (1-4) or 0 for no rating: -1

Invalid rating

Enter rating (1-4) or 0 for no rating: 4

Enter more Rating(1-4) or 0 for no more rating: 3

Enter more Rating(1-4) or 0 for no more rating: 9

Invalid rating

Enter more Rating(1-4) or 0 for no more rating: 4

Enter more Rating(1-4) or 0 for no more rating: 0

The following book was added to the Amazon Database:

TITLE: Harry Potter and the Cursed Child

AUTHOR: J.K. Rowling

TOTAL # OF RATINGS: 3

AVERAGE RATING: 3.67

PRICE: $2.18

HARDCOVER: No

RECOMMENDATION BASED ON BOOK RATINGS: STRONGLY RECOMMENDED

Do you want to enter another book?

Y

Enter the title of the book: MoonGlow

Enter the author of the book: Michael Chabon

Do you have more information for the book ex.price, type of cover…(Y or N)?: Y

Enter the book’s price: 17.49

Is the book hardcover? (Y or N):

Y

Enter rating (1-4) or 0 for no rating:0

The following book was added to the Amazon Database:

TITLE: MoonGlow

AUTHOR: Michael Chabon

TOTAL # OF RATINGS: 0

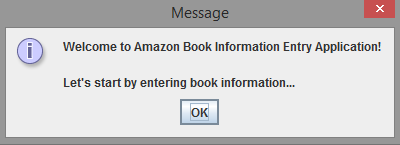
AVERAGE RATING: 0.0

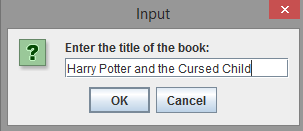
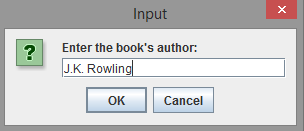
PRICE: $17.49

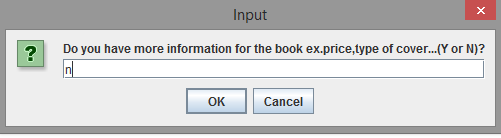
HARDCOVER: YES

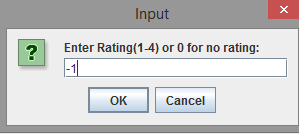
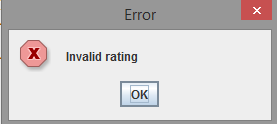
RECOMMENDATION BASED ON BOOK RATINGS: No Rating Information is Available

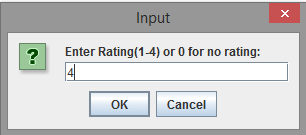
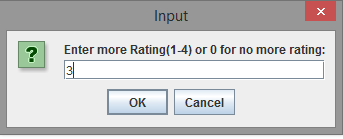
**Test #1 Screenshots**

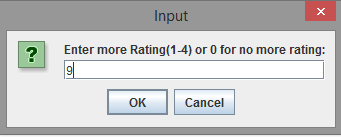
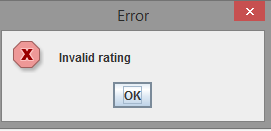


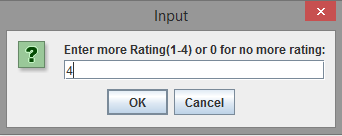
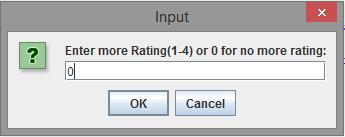
 

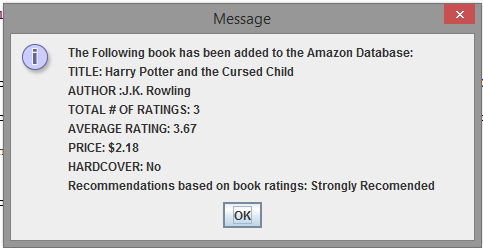


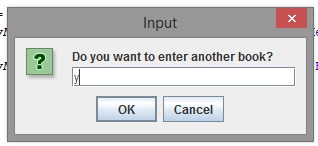
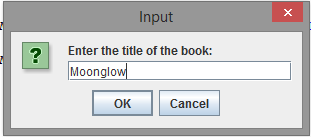
 

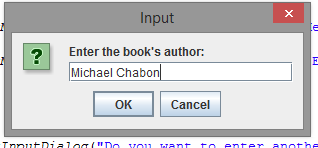
 

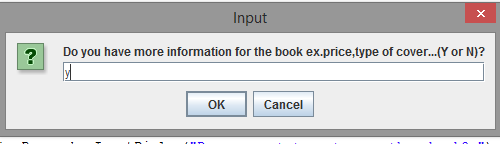
 

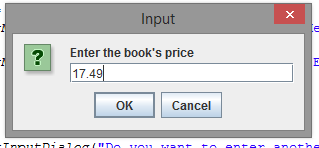
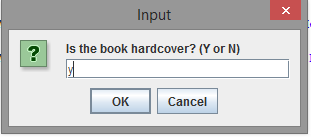
 

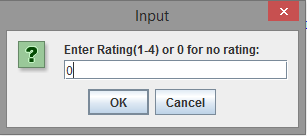


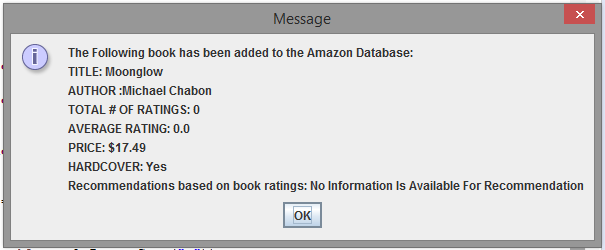
 

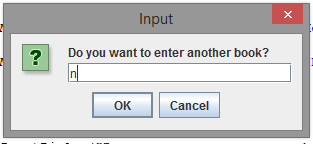


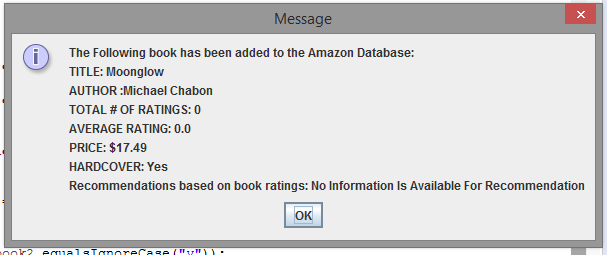










**Test # 2: Hand Calculated Results**

**---------------------------------------------------------------------------------------------------------------------**

# of invalid inputs: 0

Invalid input numbers: 0

Input ratings for book: 2,3,2,3

Hand calculated Total # of ratings: 4

Hand Calculated average rating score: 2.5

Pre-Determined recommendation comment based off the average rating Score: Recommended

Do you want to enter another book?

n

---------------------------------------------------------------------------------------------------------------------

**Test #2: Program Inputs and Outputs**

**---------------------------------------------------------------------------------------------------------------------**

Enter the title of the book: The Catcher in the Rye

Enter the author of the book: J. D. Salinger

Do you have more information for the book ex.price, type of cover…(Y or N)?: n

Enter rating (1-4) or 0 for no rating: 2

Enter more Rating(1-4) or 0 for no more rating: 3

Enter more Rating(1-4) or 0 for no more rating: 2

Enter more Rating(1-4) or 0 for no more rating: 3

The following book was added to the Amazon Database:

TITLE: The Catcher in the Rye

AUTHOR: J. D. Salinger

TOTAL # OF RATINGS: 4

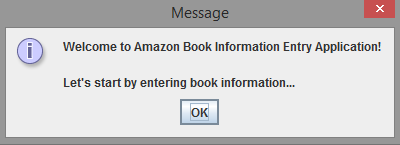
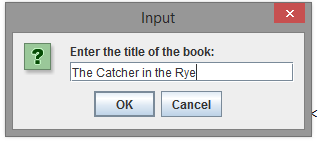
AVERAGE RATING: 2.5

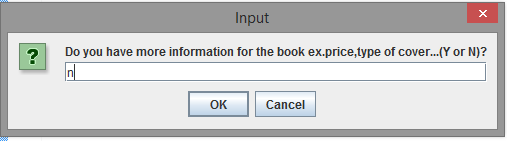
PRICE: $6.23

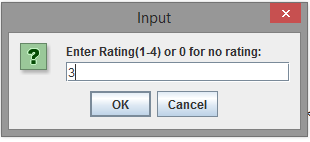
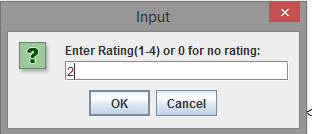
HARDCOVER: No

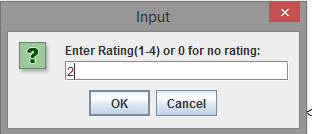
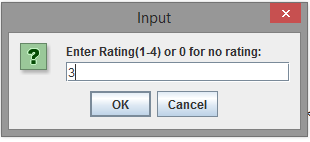
RECOMMENDATION BASED ON BOOK RATINGS: Recommended

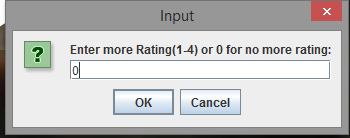
**Test #2 Screenshots**

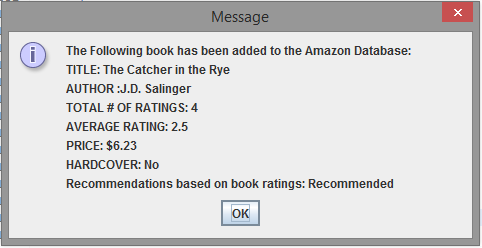
** **

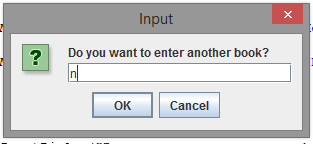
****

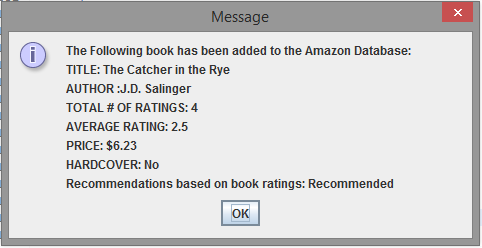
** **

** **

****

****

****

****

**Test #3: Hand Calculated Results**

**---------------------------------------------------------------------------------------------------------------------**

# of invalid inputs: 0

Invalid input numbers: 0

Input ratings for book: 1,2,1,2,1,2

Hand calculated Total # of ratings: 6

Hand Calculated average rating score: 1.50

Pre-Determined recommendation comment based off the average rating Score: Not Recommended

Do you want to enter another book?

N

---------------------------------------------------------------------------------------------------------------------

**Test #3: Program Inputs and Outputs**

---------------------------------------------------------------------------------------------------------------------

Enter the title of the book: Twilight

Enter the author of the book: Stephenie Meyer

Do you have more information for the book ex.price, type of cover…(Y or N)?: n

Enter rating (1-4) or 0 for no rating: 1

Enter more Rating(1-4) or 0 for no more rating: 2

Enter more Rating(1-4) or 0 for no more rating: 1

Enter more Rating(1-4) or 0 for no more rating: 2

Enter more Rating(1-4) or 0 for no more rating: 1

Enter more Rating(1-4) or 0 for no more rating: 2

The following book was added to the Amazon Database:

TITLE: Twilight

AUTHOR: Stephenie Meyer

TOTAL # OF RATINGS: 6

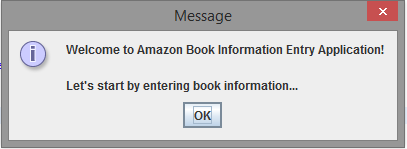
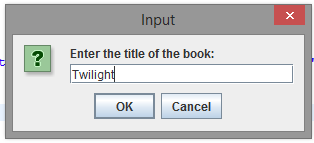
AVERAGE RATING: 1.5

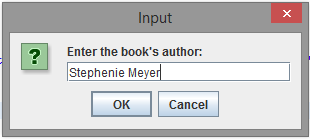
PRICE: $7.41

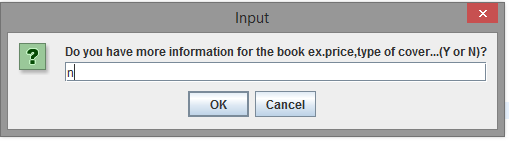
HARDCOVER: No

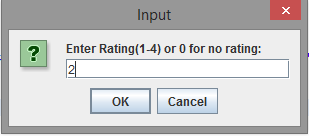
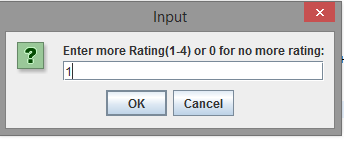
RECOMMENDATION BASED ON BOOK RATINGS: Not Recommended

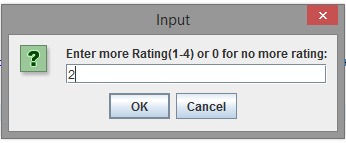
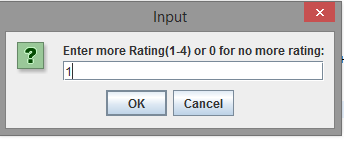
**Test #3 Screenshots**

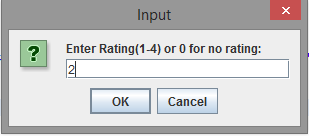
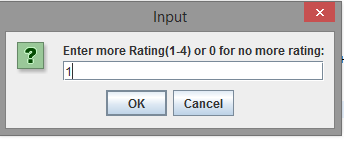
 

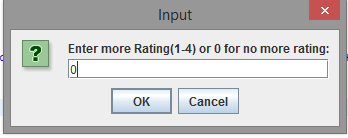


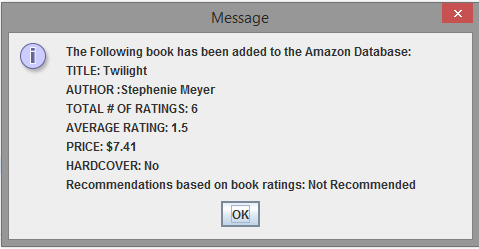


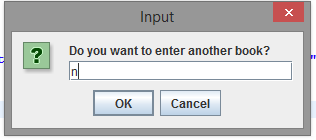
 

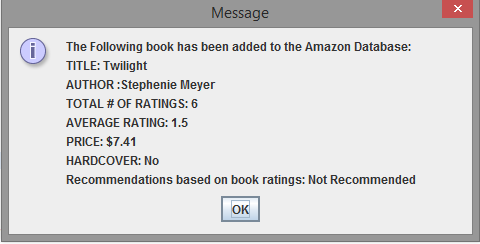
** **

** **

****

****

****

****