Assignment 2 - Requirements

In this assessment, you will create a program that stores information about a DVD collection.

The program must do the following:

1. Allow the user
   * + - 1. Add a DVD to the collection

Instantiate a DVD object

Add to ArrayList or Map

* + - * 1. Remove a DVD from the collection

Removing from the Array/ ArrayList or Map

* + - * 1. Edit the information for an existing DVD in the collection

Use the setter if needed, but form some connection

* + - * 1. List the DVDs in the collection

Print the entire Array/ArrayList using the Arrays.toString() method or …

For loop / foreach and use the objects toString() method to print individually

* + - * 1. Display the information for a particular DVD

Use a for loop and an if that checks against an inputted ID

**If using Strings, use .equals() rather than ==**

Find by ID

* + - * 1. Search for a DVD by title

Find by Title

1. Load the DVD library from a file … At the START of the program!!!
   1. File IO – DAO Class
      1. File object
      2. FileReader
      3. BufferedReader
   2. Data Marshalling – Cleaning, Parsing to allow for creation of DVD object
   3. Mapping … Creating objects
2. Save the DVD library back to the file **when the program completes –**
   1. **FileWriter**
   2. **BufferedWriter???**
3. Allow the user to add, edit, or delete many DVDs in one session (one at a time!!!)
4. Additionally, the program must follow the MVC design pattern and use dependency injection as shown in the course material.

You should follow the process outlined in the *Agile Approach Checklist for Console Applications* document elsewhere in this course.

DVD object should have the following fields:

|  |  |
| --- | --- |
| **Variable** | **Datatype** |
| Title | String |
| Release date | Date / String |
| MPAA rating | Int / String / char |
| Director's name | String |
| Studio | String |
| User rating or note | String |
| DVDS | Static Arraylist / Map |

# Marking Criteria

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Criteria** | Meets Expectations | Needs Improvement | No Credit | M.E. % (Roughly) |
| 1 | **Specifications** | 15 | 8 | 0 | 17.65% |
| 2 | **MVC** | 10 | 5 | 0 | 11.76% |
| 3 | **I/O** | 10 | 5 | 0 | 11.76% |
| 4 | **List/Map** | 10 | 5 | 0 | 11.76% |
| 5 | **Java Syntax** | 10 | 5 | 0 | 11.76% |
| 6 | **OOP** | 5 | 3 | 0 | 5.88% |
| 7 | **Interfaces** | 5 | 3 | 0 | 5.88% |
| 8 | **Inheritance** | 5 | 3 | 0 | 5.88% |
| 9 | **Composition** | 5 | 3 | 0 | 5.88% |
| 10 | **Code Style** | 10 | 5 | 0 | 11.76% |
|  | Total | 85 | 45 | 0 | 100.00% |