

# HW1: DVD Manager Application

## Background

*Unfortunately, this application was started by your co-worker that has since been fired. You need to finish only the console portion of the application before the first client deadline. Do not re-write or add any features beyond the following specifications. Do not create additional classes or impact the public interface. Your application MUST meet these requirements.*

## Abstract

Complete the DVD Manager project. The provided code is part of an application for cataloging a DVD collection. The application allows a user to keep track of all their DVDs. Basic information is kept for every DVD title in the collection. The application is currently a console application, but a GUI has been started for you.

## Activity

**TASK:** Complete the first two classes, DVD and DVDCollection, to create a correctly functioning console application.

## Provided files

Six Java files are provided and described here:

- DVD.java - A class to model a single DVD, including its title, rating and total running time.
- DVDCollection.java - A class to model a collection of DVDs using an array.
- DVDUserInterface.java - An interface that describes the operation required for any user interface to this DVD collection.
- DVDGUI.java - A class that implements the DVDUserInterface interface and provides a graphical user interface to the DVD collection.
- DVDConsoleUI.java - A class that implements the DVDUserInterface interface and provides a console user interface to the DVD collection.
- DVDManager.java - A class that contains a main method that launches one of the two user interfaces for the user to work with the DVD collection based on the user input.

## TASK:

Complete the DVD class by completing the given methods (constructor, toString, accessors and mutators).

The DVDCollection class uses an array to maintain the collection of DVDs. The DVDs should be stored in alphabetical order based on title starting at position 0 in the array, using adjacent cells. All titles should be stored in uppercase only and are assumed to be unique (no duplicates). You may assume titles do not contain commas. Complete the following methods in the DVDCollection class in the order shown below and test each one before moving on.

- toString - returns a string containing all of the DVDs in the collection, separated by newlines characters, along with the value of numdvds and the length of the DVD array for debugging purposes. The string should be formatted as shown in the example below:

```
numdvds = 3
```

```
dvdArray.length = 7
```

```
dvdArray[0] = ANGELS AND DEMONS/PG-13/138min
```

```
dvdArray[1] = STAR TREK/R/127min
```

```
dvdArray[2] = UP/PG/96min
```

- addOrModifyDVD - given the title, rating and running time of a DVD, add this DVD to the collection if the title is not present in the DVD collection or modify the DVD's rating and running time if the title is present in the collection. Do this operation only if the rating and running time are valid. If a new DVD is added to this collection, insert the DVD so that all DVDs are in alphabetical order by title. (NOTE: The collection should already be in alphabetical order when this method is called since this is the only method available to insert DVDs.)
- removeDVD - given the title, this method should remove the DVD with this title from the collection if present. The title must match exactly (in uppercase). If no title matches, do not change the collection.
- getDVDsByRating - given the rating, this method should return a string containing all DVDs that match the given rating in the order that they appear in the collection, separated by newlines.
- getTotalRunningTime - this method should return the total running time of all DVDs in the collection. If there are no DVDs in the collection, return 0.

- **loadData** - given a file name, this method should try to open this file and read the DVD data contained inside to create an initial alphabetized DVD collection. HINT: Read each set of three values (title, rating, running time) and use the **addOrModifyDVD** method above to insert it into the collection. If the file cannot be found, start with an empty array for your collection. If the data in the file is corrupted, stop initializing the collection at the point of corruption.
- **save** - save the DVDs currently in the array into the same file specified during the load operation, overwriting whatever data was originally there. Use the **DVD toString** output here.

## **Screenshots**

Create screenshots of your program output while testing the features. This should demonstrate all of the features of your application. The screenshots should be jpg or jpeg format. Please crop and format the images.

## **Deliverables**

Submit your source code, input files used, and screenshots of your program running. Submit all for full credit.

## **Original work only**

Upload your ORIGINAL work to Canvas before the due date and time. No GPT, Google, Cheg, 'collaborations', or any unoriginal work will be accepted.