|  |  |
| --- | --- |
| sait_icon_wordmark_horiz_text_black | **CPRG 251**  **Assignment 1 (Modules 1 – 2) Movie Management** |

**Name:**

**Marks:**  / 18

## **D2L Submission Instructions**

1. One ZIP file needs to submit to D2L with the following naming convention **CPRG251\_A1\_Firstname\_Lastname.zip** using your first and last name.

If working in a group of two (2), only one team member needs to submit to D2L (both can if you so wish). Both members will receive the same feedback. The file should have the following naming convention: **CPRG251\_A1\_Lastname of member 1\_Lastname of member 2.zip**

1. The ZIP file must contain the following:   
   1. The following directory structure:

* bin/ – Compiled Java files.
* src/ – Java source code files:
  + sait/mms/application/
  + sait/mms/managers/
  + sait/mms/problemdomain/
* doc/ – Generated Javadoc files.
  + Ensure the *private* option is checked and everything is included in the generated documentation.
* lib/ – Any third-party libraries. This folder can be empty.
* res/ – Any resource or data files.
* test/ – Unit test cases. This folder can be empty.

1. A text file named **Readme.txt** in the root folder of the ZIP archive and contain:

* A project title.
* What the program does.
* The date.
* The author
* How to run the program.

1. A runnable JAR file in the root folder of the ZIP archive.
   1. Use the naming convention: **FirstInitialLastname1.jar** (i.e.: JBlow1.jar).
   2. It is to be built using only Eclipse IDE and JDK 1.8x.
2. Text files containing the output of each completed runner’s main method.
   1. Use the name of the runner for the text file (i.e.: XYZRunner.txt)

## **Assignment Instructions**

Import the provided Java code into your Eclipse. Go through the provided runner classes (in the order below) to implement the related class.

1. MovieRunner
2. MovieManagementSystemRunner
3. MovieManagementSystemMenuRunner

Start at the top of the runner class then go down implementing each called method correctly. Once you have completed implementing the class, copy and paste the output from the runner’s main method into a text file called XYZRunner.txt (replacing *XYZRunner* with the corresponding runner).

Use the provided JavaDocs, and output formatting to implement the classes.

## **Assignment Guidelines**

1. You will use only Eclipse IDE.
2. The due date for this assignment is posted in D2L in the assignment submission area and in the provided calendar located in the *Course Information* area**.** Any assignment submitted after the due date will receive a mark of zero, but feedback maybe given.
3. Submissions must be student’s original work. Refer to the Academic Misconduct (AC.3.4) policies and procedures.

## **Problem**

Write a movie management system using object-oriented design principles. The program will read from the supplied data file into a single array list. The data file (movies.txt) contains information about the movies. Each movie record has the following attributes:

* Duration (in minutes)
* Title
* Year of release

Each record in the movies.txt file is formatted as follows:

* Duration,Title,Year
* e.g.: 91,Gravity,2013

Specifically, you must create an interactive menu driven application that gives the user the following options:

1. **Add a new movie.** 
   * The user will be prompted to enter the duration in minutes, title of the new movie, and the year the movie was released. Before the movie is added, the inputs provided by the user should be validated:
     + The duration and year of the movie should not be zero and the title of the movie should not be empty.
2. **Generate list of movies released in a year.** 
   * The user will input a year and the program will display a list of all the movies released in that year along with the duration (in minutes) of all the movies.
   * The list of movies does not have to be sorted.
3. **Generate list of random movies.**
   * The user will input the number of movies and the program will display a list containing the number of random movies along with the duration (in minutes) of all the movies.
   * There is no minimum or maximum duration for the movies in the list.
   * You can use *Collections.shuffle* in the *java.utils* package to randomize the movie list.
4. **Exit the program.**
   * Save the list of movies back into the data file “movies.txt” using the above format (Duration,Title,Year).
   * The process terminates cleanly.

## **Notes:**

## To follow the object-oriented principles, your project should contain ONLY the following classes and methods in their respective package.

|  |  |  |
| --- | --- | --- |
| **Package** | **Class** | **Methods** |
| sait.mms.application | AppDriver | main |
| sait.mms.managers | MovieManagementSystem | Default constructor,  displayMenu, addMovie, generateMoviesInYear, generateRandomMovieList, loadMovies, writeMoviesToFile |
| sait.mms.problemdomain | Movie | Non-default constructor, Accessor methods, toString |

You cannot use parallel and/or nested arrays in this assignment (you can use ArrayList).

## **Marking Guide**

The following criteria will be used to grade each implemented class:

|  |  |  |
| --- | --- | --- |
| Appropriate loops and test conditions are used. | / | 1 |
| Output matches what is shown in provided runner classes. | / | 1 |
| Java code is properly formatted. | / | 1 |
| Naming conventions are followed correctly. | / | 1 |
| Classes are well documented. | / | 1 |
| Provided runner class isn’t modified (except for removing comment delimiters). | / | 1 |
| **Total** | **/** | **6** |