Assignment 4

Reminder: Your code is to be designed and written by only you and not to be shared with anyone else. See the Course Outline for details explaining the policies on Academic Integrity. Submissions that violate the Academic Integrity policy will be forwarded directly to the Computer Science Academic Integrity Committee.
All materials provided to you for this work are copyrighted, these and all solutions you create for this work cannot be shared in any form (digital, printed or otherwise). Any violations of this will be investigated and reported to Academic Integrity.

Objectives

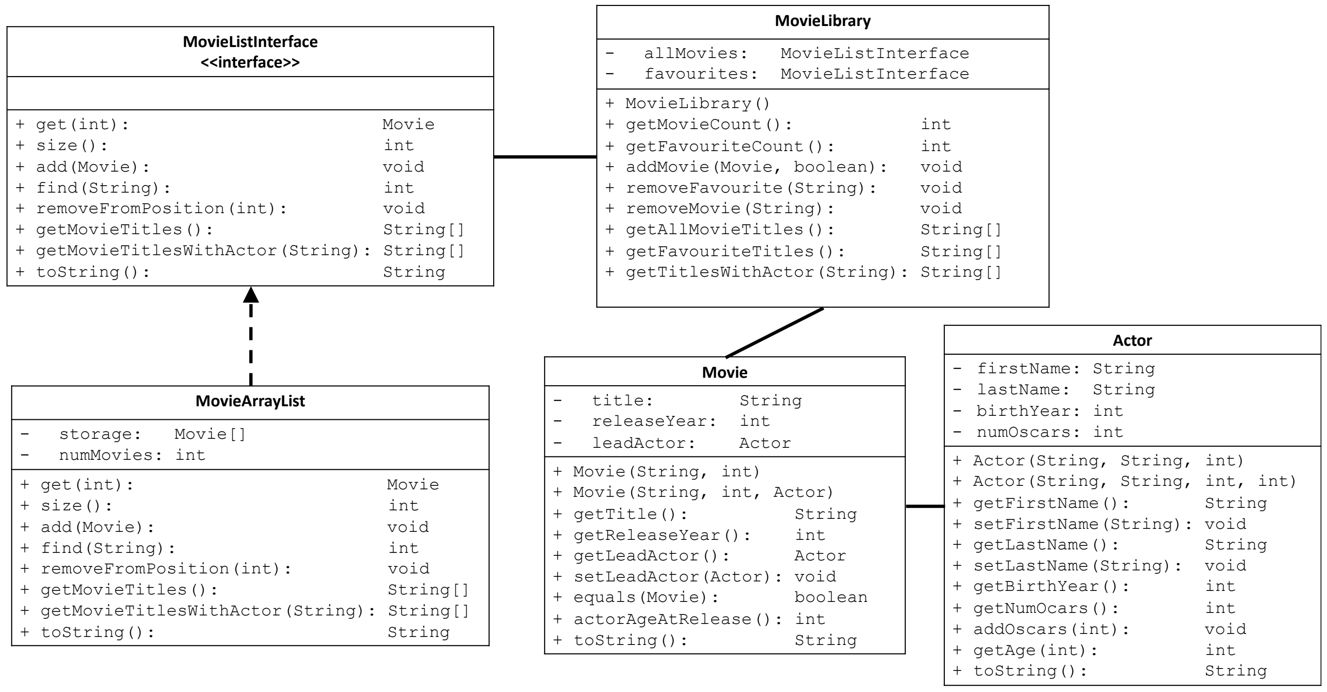
- Practice implementing classes
- Practice implementing to an interface
- Practice reading and understanding specifications
- Exposure to more complete testing

Introduction

You are creating a set of classes that can be used in a new *Movie Library* application. You are creating the infrastructure to allow a user to create a personalized *Movie Library* and add/remove movies to/from the library and look up information in the library.

You will implement the `MovieLibrary` class, where a library maintains two lists: a list of all the movies in the library and a sublist of the movies that are favourites. You are familiar with the fields and related classes of a movie from the past assignments but we have provided implementations of these for you.

The UML diagram on the following page provides the specification for the classes involved, showing the attributes contained in each class and a list of the methods in each class. The `MovieLibrary` class contains two fields/attributes that are of type `MovieListInterface`, which are each initialized as a `MovieArrayList` objects and which are subsequently made up of `Movie` objects. The complete implementation of the `MovieListInterface` and `Movie` and `Actor` classes have been provided to you but you will complete the `MovieArrayList` and `MovieLibrary` classes.



Submission and Grading

Submit your `MovieArrayList.java` and `MovieLibrary.java` with the completed methods through the assignment link in BrightSpace.

- You **must** name the methods in `MovieArrayList` and `MovieLibrary` as specified in the documentation and used in `A4Tester.java` or you will receive a **zero grade** as the tester will not compile.
- If you chose not to complete some of the methods required, you **must at least provide a stub for the incomplete method** in order for our tester to compile.
- If you submit files that do not compile with our tester (ie. an incorrect filename, missing method, etc) you will receive a **zero grade** for the assignment.
- Your code must **not** be written to specifically pass the test cases in the testers, instead, it must work on all valid inputs. We may change the input values when we run the tests and we will inspect your code for hard-coded solutions.
- **ALL late** and **incorrect** submissions will be given a **ZERO** grade.

Getting Started

1) Download all java files provided in the Assignment on BrightSpace

2) Try to compile `A4Tester.java`
   You will see it does not compile. You should see the following error:

```
MovieArrayList is not abstract and does not override abstract
method getMovieTitlesWithActor(String) in MovieListInterface
```

Add stubs for each of the methods `MovieArrayList` must implement:

a) Copy the method signatures from `MovieListInterface.java` to `MovieArrayList.java`

b) For each method signature, remove the “;” from the end of the signature and add an “{” and “}” to make it a method and make the method **`public`**

    c) For each method that has a return value, add a corresponding dummy return statement.
    NOTE: return null as a dummy return for functions that return an object or an array

    d) Recompile and edit until all compilation errors are gone

    DO NOT move on until you have the tester compiling with no errors!

3) Implement each method in `MovieArrayList.java`

    a) Uncomment the first movie test call within the `main` method of `A4Tester.java`

    b) Implement one method at time

    c) Compile and run the test program `A4Tester.java`

    d) Repeat step a-c until all methods are implemented and all movie tests pass

4) Repeat the process in Step 3 to implement each method in `MovieLibrary.java`

    Notice: The `MovieLibrary` class has the following 2 fields:

    ```
        private MovieListInterface      allMovies;
        private MovieListInterface      favourites;
    ```

    The implementation of each of the methods within the `MovieLibrary` class CANNOT access the private `MovieArrayList` fields (`storage` and `numMovies`) directly. They will however be able to call any method that is specified in the `MovieListInterface`

    ie. `String[] movieTitles = favourites.getMovieTitles();`