University of Newcastle
School of Electrical Engineering and Computing
SENG1110/6110 Programming Assignment 2 – Semester 1, 2017
Due: By electronic submission (Blackboard) by 11:59pm on **Fri 26/05.**

Movie database – part 2

Introduction

The objective of this assignment is to extend the implementation of Assignment 1 using arrays and external files.

Before you start

Carefully read the problem description below. Make sure that you have all the information necessary to create the program. Do not assume what is necessary. There is a discussion board forum: assignment 2. Post your questions there and check regularly. Start the assignment as soon as possible.

Problem Description

Your task in this assignment is to extend your movie database program from Assignment 1, by **adding** the following functionality:

1. Will allow the user to **save a movie database to a file**.
2. Will allow the user to **open an existing movie database from a file**.
3. Will allow the user to **enter any number of movies to the database**.
4. Will allow the user to **enter any number of movies into a playlist** (assuming they don't go over the size or time limit for a playlist).
5. Will allow the user to **create any number of playlists**.
6. Will prevent the user from entering a **movie that already exists** in the database.
7. Will allow the user to **request a list of all movies** whose duration is under a certain time (in minutes).
8. Will allow the user to **edit a movie in the database**.

Program Requirements:

Your program should still implement four classes, which store the following data:

- `Movie.java` – storing the following details about a movie.
  - `name` – the name of the movie.
  - `director` – the director of the movie.
  - `fileSize` – the size of the file in terms of memory (MB).
  - `duration` – the length of the movie in minutes
  - `numberOfMovies` – a static variable

- `Playlist.java` – stores up to any number of movies at a time.

  - `movies []` – An array of Movie objects, with initial size set to 4.
  - `totalTime` – the total playing time of all movies stored

- `MAX_TIME` – a constant which stores the maximum playing time for the playlist, set to a value of **1000** (minutes).
- `totalSize` – the current sum of all movie file sizes stored in this playlist.
- `MAX_SIZE` – a constant which stores the maximum files size allocation for this playlist, set to **20GB**.
- `logicalSize` – the current number of movies stored in the playlist.

- `MovieDatabase.java` – stores all current movies in the system.
  - `allMovies []` – an array of all current Movie objects stored in the system. This array's size should be initially set to 4.
  - `logicalSize` – the current number of movies stored in the database.

- `Interface.java` – provides the user interface.
  - `playlists []` – can store any number of Playlist objects, but has an initial size of 2.
  - `database` – stores a MovieDatabase object, holding all movies in the system.
  - `logicalSize` – the current number of playlists stored.

You should continue to apply the principles of encapsulation to your classes, and also write constructors for all classes. The '`numberOfMovies`' variable on the `Movie` class should be a static, and when a new movie object is created, this variable should be incremented, so as to indicate how many movies are currently stored. This variable should also be decremented when a `Movie` is deleted from the `MovieDatabase`.
In the `Movie` class:

- You need to have an additional Constructor method, which takes a movie name, director name, movie duration and file size as parameters and initialises the variables to these values.

In the `MovieDatabase` class:

- You should have a method for resizing the Movie object array. The formal parameters for this method are a boolean and a Movie array. When the boolean is true, we are increasing the physical size of the array by 4. When the boolean is false, we are reducing the array size by half its current physical size, while ensuring that the array is never smaller than its initial size of 4. This method will be used when the allMovies array becomes full, and the user wants to add a new movie. It will also be used when the logical size of the array is less than half of the physical size of the array, to cut down on the storage required for the array.
- You should also have a method that is passed a movie name, director name, fileSize and duration and returns true if that movie exists in the database. That way you can prevent duplicate movies being added into the movie database.

In the `Playlist` class:

- You should also have a method for resizing your Movie object array. The formal parameters for this method are a boolean and a Movie array. When the boolean is true, we are increasing the physical size of the array by 4. When the boolean is false, we are reducing the array size by half it's current physical size, while ensuring that the array is never smaller than its initial size of 4. This method will be used when the Movie array becomes full, and the user wants to add a new movie. It will also be used when the logical size of the array is less than half of the physical size of the array, to cut down on the storage required for the array.

Additionally, your classes need to have methods that provide the functionality outlined in the problem description. The only class which should have a main method is `Interface.java`, which should create an instance of the class Interface and call the `run()` method which will have the code to provide the user with

a menu to allow them to perform any of the tasks outlined in the problem description. The class Interface will also be the only one that will receive inputs and show output.

The format for your output of the MovieDatabase file is provided with the Assignment specifications. Your program should be able to read this file, and output to file in the same format. Note that it is just a text file.

Your solution must be your own work. Marks will be awarded for: layout, both visual (variable names, indentation) and structural (scope of variables, use of methods); documentation (comments); and ability of the submission to perform as specified. Note that you should particularly pay attention to the analysis and design phase of this assignment, to ensure that you have clarified the user requirements. A more detailed marking schema will be available later.

What to submit.

You should submit the Java program (`Movie.java`, `Playlist.java`, `MovieDatabase.java` and `Interface.java`) and the assignment cover sheet electronically under Assignment 2 link in Blackboard. Note: although you can save multiple versions of your program, you can only 'submit' your assignment once.

Extra Work for SENG6110 students

The functionality 7 of your program should give the option to the user to choose the key for ordering that it can be: name, artist, fileSize or duration.

SENG1110 students that implement the SENG6110 task will receive extra 10 marks (the total mark will not be more than 100).

In the Blackboard you will find a new forum in the discussion board: "assignment2". Any question about the assignment 2 you can post there. Check this forum regularly.

Prof Regina Berretta

Apr-2017