A "Java Bean" is a simple Java data wrapper class that has a no-argument constructor and public setters and getters for all fields.

1 Movie and Actor (3P)

Create a Java Bean `Movie` with fields `title`, `year` and `actors` and a Java Bean `Actor` with fields `firstName`, `lastName` and `movies`. An actor playing in a specific movie should appear in the `actors` list of the corresponding movie and vice versa. It is not required that these classes are as "bullet proof" as the ones from Exercise 1, i.e. you can skip `null` checks and be more trusting regarding data consistency. Both classes should also have a field `id` with a unique identifier. Assume that the uniqueness of identifiers is given "from the outside" and does not have to be checked by the classes themselves.

Write code so that two instances of a `Actor`/`Movie` are considered equal whenever their id is equal. Verify with a unit test.

2 Java Serialization (5P)

Make `Actor` and `Movie` serializable using standard Java serialization. Write a unit test that creates a movie list with three of your favorite movies, each with three top-listed actors. Serialize that list into a file. Re-create your movie list by deserializing from the file and verify that the list is equal to your initial movie list.

3 JSON Serialization (7P)

Make `Actor` and `Movie` serializable as JSON using the Jackson framework. Write a unit test that serializes your movie list as JSON string. Re-create your movie list by deserializing from the JSON string and verify that the list is equal to your initial movie list.

Serialization will run into problems caused by the bidirectional relationship between actors and movies. Find out how to solve these using means provided by the Jackson framework.