

# 2. PLF POS 1

## Nachtermin Filmedatenbank

4ACIF / Georg Graf (GRG)

2023-06-21 18:50 - 20:20 / B3.08

### WARNING

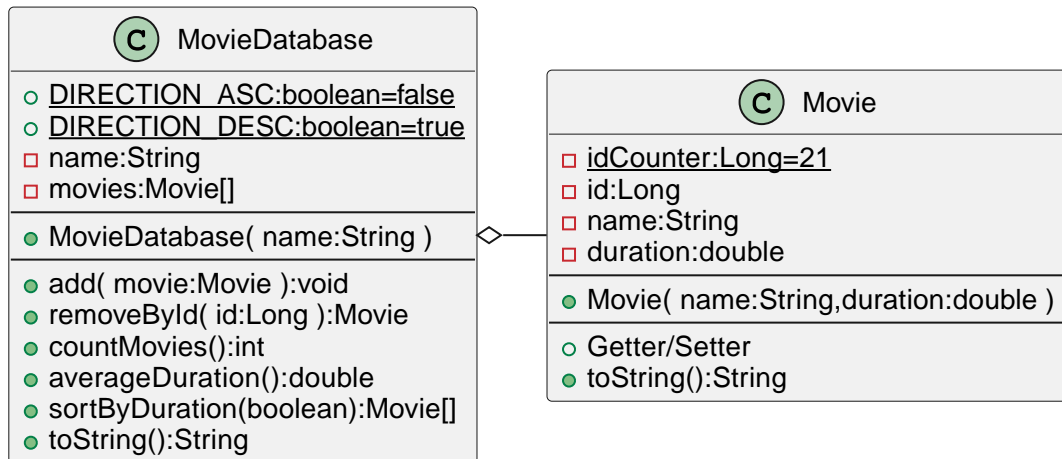
- Achten Sie darauf, nur kompilierfähige Programme abzugeben. Sollten Sie einen Compilerfehler nicht beheben können, kommentieren Sie das entsprechende Codestück aus. Nicht kompilierfähige Programme werden negativ bewertet.
- Halten Sie sich exakt an die Vorgaben, Formate sowie Klassen- und Methodennamen.
- Überall wo der Kommentar `// TODO` vorkommt, haben Sie Aufgaben zu lösen.

## Beschreibung

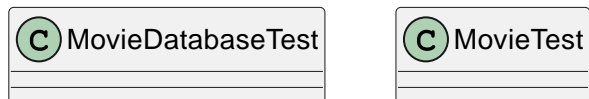
Es soll eine Filme-Datenbank entwickelt werden, in welcher Filme verwaltet werden können. Entwickeln Sie die Klassen auf Basis des UML-Diagramms und den weiter unten ausgeführten Anforderungen.

## UML-Diagramm

## model



## test



# Angabe und Abgabe Verfahren

## Angabe

- Auf **Z:\** finden Sie das Zipfile der Angabe, dieses auf den Desktop **verschieben**.
- Am Desktop - in den Desktop entpacken - und **die Aufgaben lösen**

## Abgabe

- In IntelliJ mittels **File** → **Export** → **Project to Zip File** auf das Laufwerk **Z:\** abgeben. (**Vorname\_Nachname.zip**)
- Vom Prüfungsrechner **abmelden** und diesen laufen lassen.

# 1. Aufgabe - Movie

Implementieren Sie die Model-Klasse `Movie` gemäß UML-Diagramm und folgender Klassenbeschreibung. Sie können die vorhandenen Testklassen für JUnit-Tests verwenden.

Table 1. Klassenbeschreibung

Attribut/Methode	Beschreibung
<code>idCounter</code>	<ul style="list-style-type: none"><li>• Counter, der für jede erzeugte <code>Movie</code>-Instanz eine eindeutige Id vergibt.</li><li>• Der Counter soll beim Wert 21 starten.</li></ul>
<code>id</code>	<ul style="list-style-type: none"><li>• Jede <code>Movie</code>-Instanz muss vom <code>idCounter</code> eine eindeutige Id erhalten, die nach ihrer Zuweisung <b>nicht</b> mehr verändert werden darf. Natürlich muß der <code>idCounter</code> (statisch) nach jeder ID-Vergabe inkrementiert werden, wie im Unterricht besprochen.</li></ul>
<code>name</code>	<ul style="list-style-type: none"><li>• Repräsentiert den Namen eines Filmes.</li><li>• Der Name des Movies darf nicht null sein.</li><li>• Der Name des Movies muss mindestens 3 Zeichen lang sein.</li><li>• Werfen Sie pro Fehlerfall eine <code>IllegalArgumentException</code> mit einer sprechenden Fehlermeldung.</li></ul>
<code>duration</code>	<ul style="list-style-type: none"><li>• Dauer des Filmes in Sekunden.</li></ul>
<code>toString (String)</code>	<ul style="list-style-type: none"><li>• Die Methode gibt einen String über alle Informationen eines Filmes zurück.</li></ul> <p>3 Beispiele:</p> <div><pre>21: Kill Bill: Vol. 1 (6660 sec) 22: From Dusk Till Dawn (6480 sec) 24: Pulp Fiction (9240 sec)</pre></div>

## 2. Aufgabe - MovieDatabase

Implementieren Sie die Klasse `MovieDatabase` gemäß UML-Diagramm und folgender Klassenbeschreibung. Sie können die vorhandenen Testklassen für JUnit-Tests verwenden.

Table 2. Klassenbeschreibung

Attribut/Methode	Beschreibung
<code>name</code>	Der Name der Filme Datenbank.
<code>movies</code>	<ul style="list-style-type: none"><li>• Alle Filme sollen im Array <code>movies</code> vom Typ <code>Movie[]</code> gespeichert werden.</li><li>• Insgesamt sollen 7 Filme gespeichert werden können.</li><li>• Achten Sie darauf, dass keine Methode eine Referenz auf das private Array zurückliefert.</li></ul>
<code>add( movie ) (void)</code>	<ul style="list-style-type: none"><li>• Die Methode fügt einen neuen Film in die Movie-Datenbase ein.</li><li>• Ein Film darf nicht doppelt aufgenommen werden. Wenn schon vorhanden, muß eine <code>IllegalArgumentException</code> geworfen werden.</li><li>• <code>null</code> ist kein Film und darf nicht aufgenommen werden. In dem Fall soll ebenfalls eine <code>IllegalArgumentException</code> fliegen.</li></ul>
<code>removeById( id ) (Movie)</code>	<ul style="list-style-type: none"><li>• Die Methode dient zum Entfernen eines Filmes.</li><li>• Wenn der Film mit der übergebenen <code>id</code> in der Movie-DB vorhanden ist, so soll er aus dem Array entfernt werden. Geben Sie das entfernte Objekt zurück.</li><li>• Ist der Film mit der gesuchten <code>id</code> nicht vorhanden, so soll der Wert <code>null</code> zurückgegeben werden.</li></ul>
<code>countMovies() (int)</code>	<ul style="list-style-type: none"><li>• Zählt alle in der Movie-DB vorhandenen Filme und liefert den Wert zurück. (Dh. Anzahl aller nicht-null Werte im Array)</li></ul>
<code>averageDuration() (double)</code>	<ul style="list-style-type: none"><li>• retourniert die durchschnittliche Dauer aller Filme in der Datenbank. (Arithmetisches Mittel: Gesamtlänge/Anzahl)</li></ul>
<code>sortByDuration(boolean) (Movie[])</code>	<ul style="list-style-type: none"><li>• Die Methode soll ein <b>neues</b> Array mit den sortierten Werten zurückliefern.</li><li>• Ist der boolean Parameter <code>DIRECTION_ASC</code> soll der kürzeste Film am Beginn sein.</li><li>• Ist der boolean Parameter <code>DIRECTION_DESC</code> soll der längste Film am Beginn sein.</li><li>• Alle <code>null</code> Werte sollen an das Ende des Arrays sortiert werden.</li><li>• Das bestehende Array <code>movies</code> soll unsortiert bleiben.</li></ul> <p><b>Hinweis</b></p> <p>Kopieren Sie die Daten vor dem Sortieren zuerst in ein <b>neues</b> Array mit der gleichen Größe wie <code>movies</code>.</p>

Attribut/Methode	Beschreibung
<code>toString()</code> <code>String</code>	<ul style="list-style-type: none"> <li>Liefert detaillierte Informationen über den Movie-DB zurück. Beispiel:</li> </ul> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;">         Filme-Datenbank: Quentin Lovers          3 Vorhandene Filme:          21: Kill Bill: Vol. 1 (6660 sec)          22: From Dusk Till Dawn (6480 sec)          24: Pulp Fiction (9240 sec)       </div> <ul style="list-style-type: none"> <li>Verwenden Sie eine Instanz der Klasse <code>StringBuilder</code> um die Informationen der Filme zu sammeln. Lassen Sie sich ggf. von IntelliJ dabei unterstützen!</li> </ul>

### 3. Aufgabe - 3 JUnit Tests

- Erstellen Sie, wie im Unterricht besprochen, 3 Testmethoden, welche das erwartete Verhalten der MovieDatabase überprüfen. Wählen Sie 3 Tests aus folgender Auswahl:
  - `testFill`
  - `testCapacity`
  - `testAddNull`
  - `testAddDoppelt`
  - `testRemove`
  - `testRemoveNull`
  - `testRemoveNichtVorhandenesMovie`
  - `testCount`
  - `testSort`
  - `testToString`

Viel Erfolg!!!