**Softwareontwerp & -architectuur 3**

**Casusopdracht Bioscoop – deel 1**

# Inleiding

In deze casusopdracht over een bioscoopsysteem ontwikkelen we stap voor stap een steeds complexer systeem voor de aanschaf van bioscoopkaartjes. We gebruiken deze casus om de theorie die je in de bijeenkomsten krijgt gelijk toe te passen in code. Je werkt aan de casusopdracht tijdens de bijeenkomsten en tussen de bijeenkomsten. In de bijeenkomsten kun je gevraagd worden door je docent om jouw uitwerking van een bepaald aspect te laten zien. Zo leren we van elkaars werk en heb je de nodige oefening en feedback gehad voordat je de nieuwe kennis in de Eindopdracht moet toepassen.

Voor ons bioscoopsysteem gaan we een paar klassen ontwikkelen. In de bioscoop draaien natuurlijk films. Elke film wordt minimaal 1, maar in veel gevallen vaker vertoond. Een bezoeker kan kaartjes kopen voor een voorstelling van een film en daarbij kan deze aangeven of hij/zij een premium zitplaats wil. De stoel is dan wat luxer uitgevoerd, zit natuurlijk comfortabeler en heeft net wat handiger vakjes voor eten en drinken dan de standaardstoel. Een bestelling kan bestaan uit zowel gewone- als premium-stoelen. Dat zal niet zijn wanneer je met twee personen gaat, maar als je met een groep gaat en je wilt bij elkaar in de buurt zitten, kan zo’n mix ontstaan.

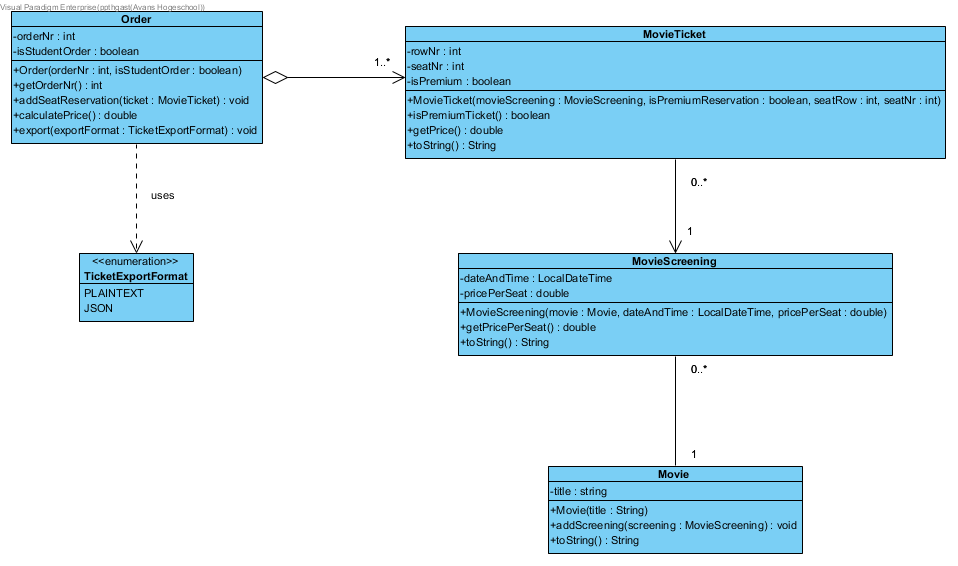
Wat de prijsberekening betreft, is er een aantal regels (die voor studenten voordelig uitpakken):

* Elk 2e ticket is gratis voor studenten (elke dag van de week) of als het een voorstelling betreft op een doordeweekse dag (ma/di/wo/do) voor iedereen.
* In het weekend betaal je als niet-student de volle prijs, tenzij de bestelling uit 6 kaartjes of meer bestaat, dan krijg je 10% groepskorting.
* Een premium ticket is voor studenten 2,- duurder dan de standaardprijs per stoel van de voorstelling, voor niet-studenten 3,-. Deze worden in de kortingen verrekend (dus bij een 2e ticket dat gratis is, ook geen extra kosten; bij 10% korting ook 10% van de extra kosten).
* Om de casus niet nog complexer te maken, gaan we ervan uit dat bij een studenten-order alle tickets voor studenten zijn; vandaar het isStudentOrder attribuut in de Order klasse en niet in MovieTicket.  
  (is wel een leuke uitdaging om deze eis er wél in op te nemen).

Voor de toekomst voorziet de bioscoop nog andere doelgroepen met andere kortingen.

Een Order kun je ook exporteren. Voorlopig in 2 formaten naar file, plain text en JSON formaat.

Het domeinklassendiagram hieronder toont de initiële opzet van de klassen van ons systeem.



# Opdracht

Implementeer de klassen uit het gegeven diagram en voer daarbinnen de volgende taken uit:

* Je programmeert het algoritme voor de prijsberekening in de calculatePrice methode van Order.
* Je programmeert de export-functionaliteit in de export methode van Order.

Je mag programmeren in een taal naar keuze; de docent werkt zijn/haar voorbeelden uit in Java. In de volgende deelopdracht gaan we aan de slag met testen en word je gevraagd je code in een development pipeline onder te brengen die automatisch je testen runt. Kijk alvast naar de opdracht van Les 2 als je hier een voorschot op wilt nemen.

Je werkt in duo’s - dezelfde duo’s als voor de eindopdracht.