FoP

De Bibliotheek

Inleiding

De bibliotheek-oefening werd in Fundamentals of Programming 1 al eens gemaakt. Hieronder vind je de opgave opnieuw, maar met kleine aanpassingen (groen), om de oefening zinvol te kunnen oplossen met `HashMap`s en `HashSet`s. Deze aanpassingen moeten nog uitgevoerd worden op het startproject.

In volgende oefening gaan we een bibliotheek simuleren, waar boeken kunnen ontleend worden voor een bepaalde tijd. We houden het relatief eenvoudig: de bibliotheek bevat alleen boeken, dus geen tijdschriften of video's. Om een boek te kunnen ontlenen moet men uiteraard lid zijn.
In wat volgt gaan we de nodige klassen één voor één beschrijven. Uit het bovenstaande blijkt dat we **leden** zullen hebben die **boeken ontlenen**. Om alles in goede banen te leiden moeten alle bibliotheekbewerkingen gebeuren aan de **balie**.

De klasse Lid

Van een lid houden we een *id* (String) dat het lid uniek identificeert, de *naam*, het *adres* en de *gemeente* bij. De gegevens worden als String bijgehouden.

- Het adres en de gemeente moeten kunnen **gewijzigd** worden.
- Een object van deze klasse wordt gemaakt aan de hand van een **4-argumenten-constructor**.
- De klasse heeft ook een **toString()- methode**.
- De klasse voorzien we van een equals() en hashCode()-methode (laat IntelliJ deze genereren) zodat Lid-objecten met elkaar kunnen vergeleken worden op gelijkheid. Aangezien het id uniek is, kan dit hiervoor dienen. Vink bij het genereren enkel het veld id aan.

De klasse Boek

Van een boek wordt de *titel*, de *auteur* en het *ISBN-nummer* telkens als String bijgehouden. Ieder boek heeft een uniek *identificatienummer* dat als een gewone integer wordt bijgehouden. Het is namelijk mogelijk dat we meerdere exemplaren van een boek in bezit hebben: deze exemplaren onderscheiden zich van elkaar via het identificatienummer. Een

laatste belangrijk veld houdt de *status* van een boek bij: ofwel is het boek aanwezig, ofwel niet.

- Om een Boek-object te creëren gebruiken we een **4-argumenten-constructor**.
- Het moet uiteraard mogelijk zijn om de status van het boek te **wijzigen**, alle andere velden blijven onveranderd na de creatie van het object.
- Schrijf een **toString()-methode**.

De klasse Ontlening

Aan de balie gaan we ontleningen moeten kunnen bijhouden. Nu we gebruik kunnen maken van HashMap, wordt de klasse **Ontlening overbodig**. Met een HashMap is het mogelijk om een boek en een lid aan elkaar te koppelen. Temeer omdat er in de klasse Ontlening, naast een Lid en een Boek, geen extra informatie werd bijgehouden.

In de klasse Ontlening houden we bij, welk *lid*, welk *boek* heeft ontleend. Om de complexiteit niet nodeloos aan te zwengelen, houden we voorlopig de datum waarop dat gebeurde niet bij.

- Een Ontlening-object wordt gecreëerd aan de hand van een **2-argumenten-constructor**.
- Schrijf een **toString()-methode**.

Verwijder de klasse Ontlening

De klasse Balie

Aangezien alle bewerkingen aan de balie dienen te gebeuren zal deze klasse ingewikkelder zijn dan de voorgaande. De balie houdt alle *boeken*, *leden* en *ontleningen* bij. Gebruik containers om deze objecten bij te houden.

Denk goed na over welke containers hier het meest geschikt zijn.
Enkel zaken waar je rekening mee moet houden:

- We moeten alle leden verzamelen. Elk lid mag maar één keer voorkomen. Zoeken op leden moeten we zelden doen.
- We moeten alle boeken verzamelen. Elk boek mag maar één keer voorkomen. Opgelet: een boek met hetzelfde ISBN-nr kan meerdere keren voorkomen. Van populaire boeken heeft de bib namelijk meerdere exemplaren. Een fysiek boek herken je dus niet aan z'n ISBN-nummer, maar aan z'n id. Boeken gaan we dikwijls moeten opzoeken op basis van dat id. Als iemand een boek ontleent, zal het boekid ingescand worden. Wanneer iemand een boek terugbrengt, zal op basis van het boekid de ontlening ongedaan gemaakt worden.
- Denk goed na over hoe je een lid en boek het beste koppelt aan elkaar bij een ontlening!

- Een Balie-object wordt gemaakt aan de hand van een **0-argumenten-constructor**.

Volgende methoden worden in de klasse Balie gemaakt:

- **toevoegenBoek**() waarbij een Boek-object als parameter wordt meegegeven; het boek wordt aan de betreffende container toegevoegd.

- **toevoegenLid**() waarbij een Lid-object als parameter wordt meegegeven; het lid wordt aan de betreffende container toegevoegd.

- **ontleen**() waarbij ~~een boek~~ het boekid en een lid als parameter worden meegegeven. Er wordt een ~~Ontlening object~~ ontlening toegevoegd. Indien het boek reeds uitgeleend was, geef dan een passende foutmelding. Vergeet ook niet de status van het boek aan te passen.

- **brengTerug**() waarbij het id van het uitgeleend boek als parameter wordt meegegeven. Vergeet niet de status van het boek aan te passen.

- **printOverzichtGeleendeBoeken**() geeft een overzicht van alle geleende boeken op het scherm.

- **printOverzichtAanwezigeBoeken**() geeft een overzicht van alle aanwezige boeken op het scherm.

- **printOntleendeBoekenVanPersoon**() waarbij de naam van het lid als parameter wordt meegegeven; de methode geeft alle boeken die het lid ontleend heeft op het scherm (Tip: maak gebruik van entrySet()).

De klasse Hoofdklasse

Zoals gewoonlijk gebruiken we deze klasse om de bovenstaande klassen uit te testen. We schrijven één enkele methode, namelijk main(), waarin we het volgende doen:

- eerst maken we een tiental boeken aan
- daarna een achttal leden
- na het aanmaken van het Balie-object worden de leden en de boeken toegevoegd
- daarna doen we een aantal ontleningen en geven vervolgens op het scherm een overzicht van de geleende boeken en de niet geleende boeken
- vervolgens laten we een aantal boeken terugbrengen
- tenslotte geven we een overzicht van de geleende boeken, de aanwezige boeken in de bibliotheek
- om af te sluiten geven we alle geleende boeken van een bepaald lid weer op het scherm