HO
GENT

H3 Klasse en object - Oefeningen

Table of Contents

1. Doelstellingen

- Oefening 2.1: klasse voorstellen in UML
- Oefening 2.2-2.3: applicatie uitwerken die gebruik maakt van een domeinobject, extra herhaling controlestructuren

2. Oefeningen

2.1. Klasse tekenen in UML

Stel een klasse Rekening voor in UML.

Eigenschappen van een object van Rekening

- een rekeningnummer voorgesteld door een geheel getal (tot 15cijfers lang)
- het saldo van de rekening
- naam van de houder van de rekening

Constructoren

- Constructor met 2 parameters, één voor het rekeningnummer en één voor de naam van de houder.
- Constructor met 1 parameter: enkel het rekeningnummer wordt doorgegeven.
- Constructor zonder parameters

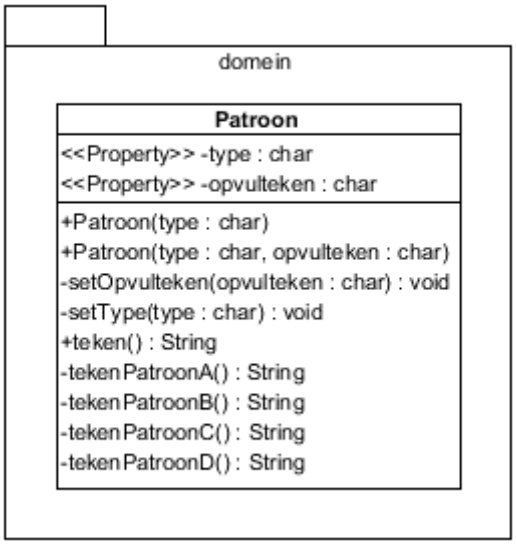
Bij creatie van een Rekening-object is het saldo altijd 0.0. Als het rekeningnummer niet meegegeven wordt, dan is de waarde 123456789, houder is “onbekend” als het niet meegegeven wordt.

Methodes

- De 3 eigenschappen hebben een getter.
- De naam van de houder van de rekening kan een nieuwe waarde krijgen via een setter, rekeningnummer ook (maar deze wijziging kan alleen maar van binnen in de klasse gebeuren, andere objecten kunnen dat niet doen). Voor saldo bestaat geen setter.
- Er is een methode om een bedrag te storten. De methode laat weten of het gelukt is of niet.
- Er is een methode om een bedrag af te halen. De methode laat weten of het gelukt is of niet.

2.2. Patroon

2.2.1. Implementeer de domeinklasse Patroon.



Implementeer de nodige **attributen, getters en setters**.

Attribuut type kan A, B, C of D zijn. Als een andere waarde doorgegeven wordt, dan zal type op de defaultwaarde A ingesteld worden.

Attribuut opvulteken kan de waarden *, +, -, ! of ? krijgen. Als een andere waarde doorgegeven wordt, dan zal opvulteken op defaultwaarde * ingesteld worden.

Constructoren

- met 1 parameter: het type wordt doorgegeven, opvulteken krijgt de defaultwaarde.
- met 2 parameters: zowel type als opvulteken worden doorgegeven.

Andere methodes

- hulpmethode tekenPatroonA (idem voor B, C en D): deze methode maakt een stuk tekst (titel + vorm gemaakt met juiste opvulteken). Deze tekst wordt teruggegeven dooor de methode.

De tekst kan zeker hier niet afgedrukt worden op het scherm, want we zitten in een domeinklasse. Deze bevat NOOIT uitvoer.

```
Patroon A
*
**
***
****
*****
******
*******
********
*********
**********
```

```
Patroon B
**********
 *********
  ********
   *******
    ******
     *****
      ****
       ***
        **
         *
```

```
Patroon C
**********
*********
********
*******
******
*****
****
***
**
*
```

```
Patroon D
         *
        **
       ***
      ****
     *****
    ******
   *******
  ********
 *********
**********
```

- methode teken(): op basis van het attribuut type wordt hier beslist welke hulpmethode aangeroepen wordt. Deze hulpmethode geeft een stuk tekst terug, de methode teken() zal

diezelfde tekst ook teruggeven.

Ben je klaar? Vergeet dan niet je domeinklasse te testen m.b.v. de gegeven testklasse!

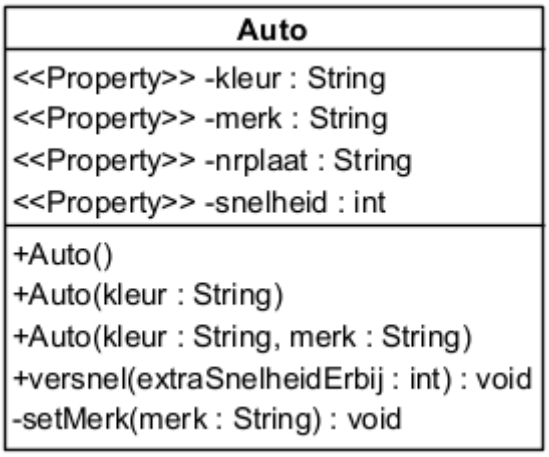
2.2.2. Applicatie

We maken nu een eenvoudige applicatie. De klasse PatroonApplicatie bevat alleen een main-methode.

Zorg ervoor dat je die 4 patronen onder elkaar afdrukt. Maak zeker gebruik van een object van de domeinklasse en een for-lus die de letters A tem D overloopt.

2.3. Auto

2.3.1. Implementeer de domeinklasse Auto



Attributen, getters en setters

Er zijn 4 attributen in de klasse Auto, ze hebben allemaal een getter en setter. Enkel het attribuut snelheid heeft GEEN setter.

Constructoren

Er zijn 3 constructoren. Bij creatie van een auto is snelheid altijd 0.

- Constructor met 2 parameters kleur en merk: nrplaat krijgt de waarde onbekend.
- Constructor met 1 parameter kleur: zowel nrplaat als merk krijgen de waarde onbekend.
- Constructor zonder parameters: zowel nrplaat als merk krijgen de waarde onbekend, de kleur is grijs.

Andere methodes

De methode versnel zorgt ervoor dat de waarde van de parameter toegevoegd wordt aan de huidige snelheid. Er is wel een maximum snelheid, nl. 220km/u. Als het versnellen ervoor zou zorgen dat die grens overschreden wordt, dan wordt snelheid op 220 ingesteld.

Klaar met implementeren? Testen maar!

2.3.2. Applicatie

Bouw een applicatie die volgende uitvoer heeft:

```
Registratie auto
 (1) Auto (alles standaard)
 (2) Auto met gekozen kleur

Wens je nog een auto te registreren?1
Registratie auto
 (1) Auto (alles standaard)
 (2) Auto met gekozen kleur

Wens je nog een auto te registreren?1
Registratie auto
 (1) Auto (alles standaard)
 (2) Auto met gekozen kleur

Wens je nog een auto te registreren?2
Geef een kleur:rood
Registratie auto
 (1) Auto (alles standaard)
 (2) Auto met gekozen kleur

Wens je nog een auto te registreren?5
Aantal geregistreerde auto's: 3
Aantal grijze auto's: 2
```

De klasse AutoApplicatie heeft alleen een main-methode.

1. Schrijf de tekst op het scherm

   ```
   Registratie auto
   (1) Auto (alles standaard)
   (2) Auto met gekozen kleur
   Wens je nog een auto te registreren?
   ```

2. Lees in welk cijfer de gebruiker ingeeft.
3. Maak een standaard auto aan bij keuze 1 (=constructor zonder parameters aanroepen) en bij keuze 2 vraag je eerst een kleur om daarna de auto aan te maken.
4. Als er een ander cijfer ingegeven wordt dan 1 of 2, eindigt de applicatie met een overzicht van hoeveel auto's geregistreerd werden en hoeveel er daarvan grijs waren.

Het attribuut kleur is een object van de klasse String. Als je de waarde van kleur wil vergelijken met "grijs" dan kan je dat niet met ==. Je kan dit als volgt vergelijken: auto.getKleur().equals("grijs")