PXLJavaAdvanced1819

Week 3: Inner classes

Opdracht 1:

Maak een klasse **Gearbox** (versnellingsbak). Voorzie een member variabele *maxGears*, het maximum aantal versnellingen, en een boolean member variabele *clutchIsIn* (clutch = ontkoppelingspedaal).

Voorzie een constructor waarbij je de waarde voor *maxGears* geeft.

Voorzie verder een methode *operateClutch(boolean in)* waarmee je de status van de ontkoppelingspedaal kan wijzigen.

Voorzie nu een publieke inner class **Gear** (versnelling) in de klasse **Gearbox**. Deze inner class heeft 2 member variabelen, *gearNumber*(int) en *ratio* (double). Voorzie de methode *driveSpeed(int revs)* die als resultaat de waarde van de parameter vermenigvuldigt met de *ratio*.

Voorzie in de klasse **Gearbox** een datastructuur (ArrayList of array) om **Gear**-objecten bij te houden. De methode *addGear(Gear gear)* voegt een Gear-object toe aan de datastructuur.

Voorzie nu een main-methode in een aparte klasse.
Maak een Gearbox aan met maxGears 2 en voeg vervolgens 2 Gear-objecten toe. Hoe maak je objecten van de inner class aan?

Vaak is het de verantwoordelijkheid van de outer class, om de objecten van de inner class aan te maken. De inner class **Gear** willen we in dit geval verborgen houden voor een programmeur die de klasse **Gearbox** gebruikt. Daarom maken we de inner class **Gear** private.

Voeg ook een member variabele *currentGear (int)* toe aan de klasse **Gearbox**. Deze member variabele geeft aan in welke versnelling de versnellingsbak momenteel staat.

Voorzie in de constructor van de klasse **Gearbox** dat alle (van 0 t.e.m. maxGears) **Gear**-objecten worden aangemaakt en m.b.v. de methode *addGear* worden bijgehouden in de datastructuur. De ratio van elke **Gear**-object is het gearNumber * 5,3.

Voorzie een methode *changeGear(int newGear)*, waarbij member variabele *currentGear* de waarde krijgt van newGear indien newGear een geldige waarde is en *clutchIsIn* true is. Indien je changeGear oproept met clutchIsIn false of met een ongeldige waarde, zal je versnellingsbak een vreemd geluid maken (System.out.println) en de *currentGear* wordt 0. Voorzie output “Gear “ + newGear + “ selected” wanneer het veranderen van versnelling is gelukt.

Voorzie tenslotte een methode *wheelSpeed(int revs)*. Indien *clutchIsIn* true is, krijg je weer vreemd geluid en is de snelheid 0. Anders wordt de driveSpeed van de huidige versnelling teruggegeven.

Voer even de testen uit en bekijk de werking van de **Gearbox** in de klasse **GearboxUsage**. Wat merk je op?

Verwachtte output van **GearboxUsage**:

```
Gear 1 selected.
```

```
5300.0
Grind! (of een ander vreemd geluid van je versnellingsbak)
0.0
Gear 3 selected.
95399.99999999999
```

Opdracht 2: HorrorShow

Voorzie eerst de interface **Monster** met methode *menace*() en de afgeleide interface **DangerousMonster** met de methode *destroy(Monster monster).*

Maak nu een klasse **HorrorShow** met een main-methode. In de main-methode voorzie je de local inner class **Vampire** die de interface **Monster** implementeert. Een **Vampire** bedreigt met de volgende woorden “I’ll drink your blood!!”.

Voorzie eveneens een local inner class **Werewolf** die de interface **DangeroursMonster** implementeert. Zijn bedreiging is “I’ll destroy you.”. Voor de implementatie van de methode *destroy* zorg je dat eerst het te vernietigen monster zijn bedreiging uitspreekt, daarna toon je de tekst “Big fight…” en tenslotte geef je de classname van het DangerousMonster gevolgd door “ killed one ” en dan de classname (getClass().getName()) van het te vernietigen monster.

Maak nu een object van de klasse **Werewolf** aan. “Destroy” eerst een vampier en daarna destroy je een anonymous inner class Monster. Dit laatste monster bedreigt de Werewolf met de tekst “You smell like wet dog.”.