Oefening 2: Weerstanden

2.1 Object = Data + Methodes

Maak een klasse *Weerstand* die de gelijknamige elektronische component voorstelt. Deze klasse moet als attribuut de weerstandswaarde (in ohm) van de weerstand bijhouden. Daarnaast moet deze klasse ook een aantal methoden aanbieden aan de buitenwereld:

- Een constructor met één argument die toelaat om een weerstand met een bepaalde weerstandswaarde aan te maken.
- Een "getter" methode voor het attribuut weerstand.
- Een "setter" methode voor het attribuut weerstand.
- Een methode *toString* die kan gebruikt worden om de weerstand af te printen in het formaat: "weerstand van x ohm ".
- Een methode *spanning* die als argument een stroom door de weerstand (in ampère) neemt en als resultaat de overeenkomstige spanning (in volt) over de weerstand teruggeeft.
- Een methode *stroom* die als argument een spanning over de weerstand neemt en als resultaat de overeenkomstige stroom door de weerstand teruggeeft.

Test je implementatie eens uit: maak een weerstand van 2Ω en gebruik je methodes om volgende conclusie af te printen:

> *Een spanning van 3 volt over een weerstand van 2 ohm produceert een stroom van 1.5 ampère. Een stroom van 3 ampère doorheen een weerstand van 2 ohm komt overeen met een spanning van 6 volt.*

2.2 Objecten gebruiken objecten

Schrijf nu een klasse *SerieSchakeling* die een serie van weerstanden voorstelt. Deze klasse heeft als attribuut een rij van weerstanden en biedt aan de buitenwereld volgende methoden aan:

- Een default constructor (zonder argumenten) die een rij met lengte 0 (nul) aanmaakt.
- Een constructor met een rij van weerstanden als argument
- Een "getter" methode voor deze rij van weerstanden.
- Een "setter" methode voor deze rij van weerstanden
- Een methode *toString* die kan gebruikt worden om de schakeling af te printen in het formaat:
  > *serieschakeling van 3 weerstand(en): weerstand van 2 ohm; weerstand van 3 ohm; weerstand van 5 ohm;*
- Een methode *voegToe* om een bijkomende weerstand toe te voegen aan de serieschakeling. Je zal hiervoor een nieuwe rij moeten aanmaken, die een grotere lengte heeft dan de oorspronkelijke rij, en dan de oorspronkelijke weerstanden hieraan toevoegen.
- Een methode *spanning* die als argument een stroom door de schakeling neemt en als resultaat de overeenkomstige spanning over de schakeling teruggeeft. Gebruik hiervoor een lus die de spanning over de individuele weerstanden (zoals berekend door de methode spanning uit je klasse Weerstand) optelt.
- Een methode *vervangingsweerstand* die de vervangingsweerstand voor deze schakeling berekent als de som van de weerstanden in de schakeling.
- Een methode *stroom* die als argument een spanning over de schakeling neemt en als resultaat de overeenkomstige stroom produceert. Gebruik hiervoor de methode vervangingsweerstand.

Test je implementatie eens uit: maak een serieschakeling van drie weerstanden (2 ohm, 3 ohm en 5 ohm). Maak deze serieschakeling eens op de 2 manieren aan : door gebruik te maken van de default constructor en ook via de constructor met een rij van weerstanden als argument.

Bereken de stroom die bij een spanning van 5 V hoort, en de spanning die bij een stroom 1 A hoort. Voeg dan een bijkomende weerstand van 1Ω toe en kijk of dit het verwachte effect heeft op de relatie tussen de stroom en de spanning.

2.3 Encapsulatie

In plaats van zijn weerstand, uitgedrukt in ohm, kunnen we ook zijn geleiding, uitgedrukt in Mho (of Siemens), gebruiken om een weerstand te beschrijven. De geleiding van een weerstand is 1/R.

Verander je klasse *Weerstand* zodanig dat deze nu niet langer de weerstandswaarde maar wel de geleiding opslaat in een attribuut. Let wel, bij het gebruik in andere klassen vb. de klasse Serieschakeling blijf je weerstanden aanmaken van x ohm; dus je constructor blijft als argument de weerstandswaarde hebben. Enkel intern in de klasse *Weerstand* wordt nu van die weerstand zijn geleiding bewaard i.p.v. zijn weerstandswaarde.

Ga zorgvuldig na wat de gevolgen zijn van deze wijziging. Waar moet je je bestaande code wijzigen? Overtuig jezelf ervan dat de objectgeoriënteerde **encapsulatie** je hier heel wat moeite bespaart: moest je overal rechtstreeks gebruik gemaakt hebben van het attribuut van de klasse *Weerstand*, in plaats van de methodes uit deze klasse te gebruiken, dan zou je op meer verschillende plaatsen je code hebben moeten wijzigen.

2.4 Overerving

Overschrijf (override) de *equals*-methode uit de Object-klasse (public boolean equals(Object o)) in de klasse Weerstand. Deze controleert of twee Weerstand-objecten gelijk zijn aan elkaar. Test deze methode uit op twee weerstanden. Wat is het verschil met het testen op gelijk aan (==) ? Controleer dit.

Implementeer een klasse *RegelbareWeerstand* die een regelbare weerstand voorstelt. Een regelbare weerstand heeft een minimum en een maximum, waartussen zijn weerstandswaarde kan variëren. Bij het maken van deze klasse gebruik je natuurlijk overerving. Je zal waarschijnlijk merken dat je bij je implementatie méér toegangsrechten nodig hebt tot de klasse *Weerstand* dan er momenteel worden aangeboden. In dat geval moet je deze toegangsrechten op een geschikte manier aanpassen.
Je klasse *RegelbareWeerstand* moet aanbieden:

- Een constructor met twee argumenten, nl. de minimum en de maximum waarde van de weerstand. We veronderstellen dat een pas aangemaakte weerstand een weerstandswaarde heeft die gelijk is aan het minimum (en dus een geleiding die gelijk is aan 1/min).
- Een methode *verhoog* om de weerstandswaarde van de weerstand met een bepaalde fractie van zijn totale bereik te verhogen. Als de weerstand bijvoorbeeld een minimale weerstand heeft van 2Ω, een maximale weerstand van 12Ω en een huidige weerstandswaarde van 4Ω, dan zal een verhoging met fractie 0.25 resulteren in een weerstand van 6,5Ω. Hierbij moet er wel mee worden rekening gehouden dat een weerstand nooit boven zijn maximale weerstand kan gaan (of lager dan zijn minimum) : een verhoging met fractie 0.25 als de weerstand al op 10Ω staat, zal bijvoorbeeld slechts resulteren in het maximum van 12Ω.
- Een *toString*-methode die de weerstand op een gepaste manier afprint.

Test je klasse uit door een serieschakeling van 3 weerstanden te maken, die bestaat uit een weerstand van 2Ω, een weerstand van 3Ω en een regelbare weerstand met bereik van 5Ω tot 10Ω.
Bereken eens de stroom die bij een spanning van 5V hoort, en kijk daarna hoe deze evolueert als je de weerstand van de regelbare weerstand met een percentage van 25% verhoogt. (Voor deze verhoging ga je gebruik moeten maken van het concept **typecasting**) .