Opdrachten

Bij iedere module horen enkele opdrachten. Deze opdrachten zijn het meest belangrijk van deze training om goed inzicht en begrip te krijgen van de besproken onderwerpen.

Neem de tijd om over de probleemstellingen na te denken. Gebruik bij voorkeur geen Google, tenzij je niet weet wat een bepaald begrip betekent.

Leren programmeren

Introductie IT: computersystemen

1. Bepaal je eigen leeftijd volgens het binaire cijferstelsel
   Tip: schrijf eerst de machten van 2 uit naast elkaar, te beginnen (rechts) bij 2 tot de macht 0 (dat is 1) en werk dan van rechts naar links. Schrijf de uitkomsten erboven en kijk welke cijfers je bij elkaar kunt optellen om tot jouw leeftijd te komen.

2. Bepaal je eigen leeftijd volgens het hexadecimale stelsel
   Tip: gebruik hiervoor eerst de binaire schrijfwijze en zet dit om naar 'paren' van hexadecimale tekens

Programmeren

Schrijf in stappen jouw ochtend (opsta) proces uit. Doe dit bijvoorbeeld als volgt:

1. Wekker gaat
2. Moet ik naar het toilet?
   a. Ja, ik ga naar het toilet
   b. Nee, ik ga gewoon opstaan
3. Aankleden.
4. Enzovoorts

Probeer jouw proces uit te modelleren in een zogeheten PSD; zie hiervoor Google en dit mag je gewoon op papier / in MSpaint doen.

Object oriëntatie: denken in objecten

1. Maak een klassediagram voor het weergeven van de volgende klassen:
   - Vorm
   - Vierkant
   - Cirkel
   - Rechthoek

   Gebruik hiervoor de StarUML software: https://staruml.io/

2. Voor alle vormen dient een omtrek te kunnen worden berekend middels een methode ‘omtrek()’. Deze rekent het uit en geeft een ‘double’ datatype terug. Hoe die omtrek daadwerkelijk wordt berekend komt in een later hoofdstuk aan bod.

   Denk bij het modelleren aan:
   - Uitdenken van de juiste klassen; wat zijn overeenkomende eigenschappen?
   - Zit er een hierarchie tussen de klassen?
   - Welke attributen en welke methoden kun je bedenken?

3. Bonus: voeg ook klassen toe voor:
   - Kubus
   - Balk
   - Cilinder

4. Een garage wilt graag de gegevens die ze gebruiken voor registratie van auto’s, motoren, bussen en vrachtauto’s vastleggen in een applicatie. Hiervoor is het van belang dat eerst een klasse diagram wordt gemaakt.

   Denk ook hierbij aan welke eigenschappen overeen komen. **Let op: je hoeft niet in detail te treden m.b.t. onderdelen van de verschillende voertuigen (al mag dat natuurlijk wel 😉).** Om het overzichtelijk te houden kun je o.a. denken aan: merk, model, prijs, kleur, vermogen en wat specifieke eigenschappen voor de verschillende klassen.

5. Breid je voertuigen klassediagram verder uit met:
   - Constructor(en)
   - Methoden voor: starten, rijden, stoppen, enzovoorts
   - Denk na of je ook setter en getter methoden wilt toevoegen
   - Geef alle attributen en methoden de juiste access modifiers (private / public)

6. Bedenk zelf een aantal voorbeelden van en bespreek dit:
   - Directe associatie (heeft / bevat)
   - Een aggregatie (kan hebben)
   - Compositie (bevat, maar geen los bestaandsrecht)
   - Tijdeljike associatie (even nodig zoals in een methode)

Object oriëntatie: overerving

1. Welke methoden van de vormen applicatie zou je kunnen/willen overriden? Denk hierover na, schrijf het op en bespreek dit.
2. Welke methoden van de vormen applicatie zou je kunnen/willen overloaden? Denk hierover na, schrijf het op en bespreek dit.

Object oriëntatie: polymorfisme en interfaces

1. Voeg voor de vormen applicatie nieuwe abstracte klasse toe (of maak een bestaande abstract). Denk hierbij ook aan of je bepaalde methoden abstract kunt maken.
2. Doe hetzelfde voor de voertuigen applicatie.

Bonus eindopdracht leren programmeren

1. Kies één van de twee applicaties (vormen of voertuigen). Schrijf hiervoor de benodigde klassen in Java code uit. Hier hoeven nog geen constructoren of uitgeschreven methoden bij te zitten. De applicatie hoeft nog niet te runnen, zolang hij maar compileert.

   Denk hierbij aan:
   a. Indien je een klasse 'public' maakt in Java, dient de filename ook zo te heten.
   b. Klassenamen beginnen in Java bij voorkeur met een hoofdletter
   c. Wil je een klasse laten overerven van een andere klasse, dan schrijven wij het woord 'extends' erachter
   d. In een klasse beginnen wij met het benoemen van de attributen (eigenschappen), met het datatype ervoor, daarna schrijven wij de methoden.

   Voorbeeldcode:

```
class Voedingsmiddel {
  String catagorie;
  void eten() {
   }
}

class Appel extends Voedingsmiddel {
 String kleur;
 String smaak;
}

class Brood extends Voedingsmiddel {
  String graansoort;
}
```