erasmus HOGESCHOOL BRUSSEL

Programming Essentials 2

Werkcollege 1

De bedoeling van de practica is de aangeleerde principes uit de theoriesessies iin praktijk te brengen. Dit doen we door middel van de programmeertaal Java. Omdat het de eerste kennismaking is met deze taal, starten we met een introductie over de taal en de gebruikte programmeeromgeving.

1 Java

We maken gebruik van een codevoorbeeld om enkele kenmerken van Java toe te lichten. *Het volledige voorbeeld kan je terug vinden op canvas.*

```
1  package voorbeeld;
```

Een Java package is een mechanisme binnen Java om klassen te organiseren in namespaces. Klassen die binnen eenzelfde categorie of functionaliteit vallen kunnen hierdoor gegroepeerd worden. We zullen een aparte package gebruiken voor elke oefening. Wanneer we toegang wensen tot een klasse in een andere package, zullen we deze package of de klasse moeten importeren. Een import doe je als volgt:

```
//Enkel klasse Kat importeren
import voorbeeld.Kat;
//Alle klassen uit voorbeeld package importeren
import voorbeeld.*;
```

```
2  /**
3   * @author Ruben Dejonckheere
4   * Klasse voor een hond met naam, vachtkleur, gewicht en leeftijd.
5   */
6  public class Kat {
7
8  }
```

Merk de javadoc documentatie op voor het begin van de class. Let er op dat je dit niet vergeet. De klassenaam begint met een hoofdletter. Achter de sluitaccolade hoort geen ; zoals in C++.

```
9      /**
10      * Naam van de Kat
11      */
12     private String naam;
13
14     /**
15      * @param naam Nieuwe naam van Kat
16      */
17     public void setNaam(String naam) {
18         this.naam = naam;
19     }
```

Merk op dat bij elke membervariabele of methode ook javadoc en een toegangsspecificatie hoort. Een membervariabele wordt helemaal in kleine letters geschreven. Een methodenaam begint met een kleine letter en gebruikt een hoofdletter bij elk nieuw woord. Bemerk ook de . na this. (Java kent geen -> operator.)

```
20 class Test {
21     public static void main(String[] args) {
22         Kat mijnKat = new Kat("Tijger",10,Kat.Kleur.GRIJS,7);
23         System.out.println(mijnKat);
24     }
25 }
```

Anders dan in C++, hoort de main() methode in Java thuis in een klasse.

```
26   public String toString() {
27       String info = getNaam() + " is " + getKleur().toString().toLowerCase() + ".\n";
28         info += naam + " is " + leeftijd + " jaar oud.\n";
29
30         if (getGewicht()>6)
31             info += naam + " weegt te veel.";
32         else
33             info += naam + " weegt " + getGewicht() + " kilo.";
34
35         return info + "\n";
36     }
```

Merk op dat de verschillen tussen de Java syntax en de C++ syntax miniem zijn wat betreft de selectie-structuren zoals if, switch, ... Ook herhalingsstructuren zoals de while- en for-loop kunnen in Java gebruikt worden.

Toch enkele verschillen:

a. cout wordt System.out.print
b. String (*met* hoofdletter) variabelen kan je concateneren met +
c. Java kent geen const

Je zal geen cin kunnen gebruiken. In plaats daarvan zullen we gebruik maken van de Scanner klasse. (Je zal java.util.Scanner moeten importeren.)

```
1 public static void main(String[] args) {
2         Scanner s = new Scanner(System.in);
3         System.out.println("Geef een getal in: ");
4         double getal = s.nextDouble();
5         System.out.println(getal);
6     }
```

2 IntelliJ

Je kan IntelliJ downloaden via https://www.jetbrains.com/idea/download/. Als je de Ultimate editie wil gebruiken kan je beroep doen op een studentenlicentie via: https://www.jetbrains.com/students.

- We maken in IntelliJ voor elk werkcollege een nieuw project aan. We bereiken dit door in het menu File, New, Project... te selecten. Je zal merken dat je een SDK moet selecteren. Indien deze nog niet is ingevuld kan je deze aanmaken door de folder te kiezen waar je de JDK hebt gedownload en uitgepakt. In de nieuwe versies van IntelliJ kan je dit in hetzelfde menu downloaden (Download JDK). Het is aangeraden de laatste versie te downloaden en te gebruiken.

  Klik twee maal op de "next" knop. Geef op het volgende scherm een projectnaam in (kies een duidelijke naam). Daarna kan je met de knop "Finish" het project aanmaken.

- Elke oefening zal je in een aparte package maken. Een package maak je door te rechterklikken op de src folder en New | Package te kiezen.

- De naamgeving nog eens kort samengevat: alle variabelen (dus ook membervariabelen) en methodenamen beginnen met kleine letter en elk nieuw woord in de naam met een hoofdletter; klassenamen idem maar deze beginnen ook met een hoofdletter. *Pas deze notatie zorgvuldig toe!*

- Merk op dat IntelliJ de Javadoc kan gebruiken om realtime informatie te bieden terwijl je code schrijft. Om dit te activeren kan je via File/Settings, Editor, General. Show quick documentation on mouse move selecteren. Je kan je eigen documentatie ook laten bundelen tot een documentatiewebsite. Dit doe je door in het menu Tools, Generate Javadoc te kiezen. Selecteer de scope waarvoor je de documentatie wil genereren. Vervolgens kies je voor welke visibility je Javadoc wil genereren: private genereert javadoc voor alle members, public enkel voor de public members. Na het selecteren van een output folder kan je met de "Finish" knop de generatiestap starten.

- Code uittesten doe je door je klasse te selecteren, te rechterklikken en Run te selecteren. In het console-venster zal je de uitvoer kunnen zien.

Oefening 1: klasse Cirkel

Maak een klasse Cirkel die als datamembers de straal van de cirkel en de kleur omvat. De mogelijke kleuren zijn wit, geel, rood, blauw en zwart. Zorg voor een aantal constructoren zodat je een Cirkel kan construeren zonder argumenten, met slechts één argument (straal, kleur of een andere cirkel) of met twee argumenten (straal en kleur). Tenzij anders opgegeven is by default de straal 0 en de kleur wit. Maak (uiteraard) ook getters en setters. Zorg ook voor de volgende extra methoden:

- toString() om de cirkel (straal en kleur) af te printen.
- geefOmtrek() : $2 \times \pi \times r$ met r als straal
- geefOppervlakte() : $\pi \times r \times r$

Voor de waarde van π kan je Math.PI gebruiken.

Probeer ook eens de javadoc te genereren.

Oefening 2: klasse Adres

Maak een klasse Adres. Een object van de klasse kan de adresgegevens van een persoon bijhouden. Een adres bevat minstens de volgende gegevens:

- straatnaam
- huisnummer (enkel cijfers)
- bus
- postcode (4 cijfers)
- woonplaats

Zorg ervoor dat bij de initializatie van een object alle gegevens kunnen meegegeven worden vanuit de constructoren, en ook kunnen nagekeken worden (anders mogen er in het object verzuimwaarden worden ingevuld). Zorg ook voor getters en setters. Geef het object de mogelijkheid zijn inhoud af te beelden op het scherm.

Oefening 3: klasse Waterton

Implementeer een klasse Waterton. Een waterton heeft een capaciteit, dit is de maximum hoeveelheid water (in liter) die de ton kan bevatten. De ton heeft ook een actuele inhoud: het aantal liter dat ze op het moment bevat.

Voorzie 2 constructoren, één op basis van de capaciteit, waarbij de inhoud op 0 gesteld wordt (leeg), en één op basis van capaciteit en inhoud. Voorzie een methode die het opvangen van water simuleert (als het regent). Als de ton vol is, loopt ze over. Zorg ervoor dat een programmeur die de functie gebruikt kan achterhalen of ze door de regen overloopt of niet.

Voorzie in een methode om water af te tappen. Hierbij geef je als parameter mee hoeveel water je probeert af te tappen. Gebruik de verkregen hoeveelheid als return waarde. Hou er rekening mee dat de inhoud van de ton ontoereikend kan zijn voor het gevraagde aantal liter.

Voorzie in een methode print om alle relevante gegevens (inhoud, capaciteit) van een Waterton object af te drukken.

Oefening 4: klasse Auto

Implementeer een klasse Auto. Een auto heeft een kilometerteller, een brandstoftank met een zekere inhoud en een verbruik (aantal liter per 100km). Gebruik voor deze gegevens een double als datatype.

Voorzie in minstens twee constructoren, één die toelaat aan alle attributen een beginwaarde te geven en één die werkt met een aantal default waarden (nieuwe auto, lege tank). Je hoeft geen bijkomende controles uit te voeren. Je mag er van uitgaan dat de gebruiker altijd realistische waarden gebruikt.

Voorzie een methode om de tank bij te vullen (altijd tot ze vol is).

Voorzie een methode om een gegeven aantal km te rijden. In dit geval telt de kilometerteller altijd vooruit (ook alsje een negatief getal ingeeft, watstaat voor achteruit rijden). Er wordt steeds brandstof verbruikt.

Hou er rekening mee dat er mogelijk te weinig brandstof kan zijn om de volledige afstand te rijden. In dat geval wordt er gereden tot de tank leeg is. In alle andere gevallen levert de methode het werkelijk aantal gereden km als returnwaarde.

Voorzie ook een methode print om alle relevante gegevens van een Auto object af te drukken.

Oefening 5: klasse BankRekening

Op een BankRekening staat een bepaald bedrag. Voorzie een default constructor die dat bedrag op 0.0 zet, en een tweede constructor die toelaat om een positieve double door te geven. Als er een negatief getal doorgegeven wordt, mag je het bedrag van de bankrekening op 0 zetten.

Voorzie een getter en setter voor het bedrag. Probeer de gebruiker van de klasse echter te verplichten om ipv de setter gebruik te maken van de methoden uit de volgende paragraaf.

Voorzie methoden om geld te storten en om geld af te halen. Bij een afhaling mag er in het rood gegaan worden, maar niet meer dan 1000 euro. Als dat wel het geval zou zijn, mag je de afhaling weigeren.

Zorg er natuurlijk ook voor dat het object zijn gegevens kan afprinten.