Project Kantine

1.4

Auteur: Teun de Jong, Stijn Wolthuis

Docent: Harald Rietdijk

Datum: 18-05-2020

## Week 1

### 1B. Leg uit wat de begrippen declaratie en initialisatie betekenen.

**Declaratie:** Het aanmaken van een variabele, zo weet de compiler dat we een variabele gaan gebruiken met een bepaalde naam en van welk type.

**Initialisatie:** Voor de eerste keer een waarde toekennen aan een variabele.

## Week 2

### 2A. Leg uit waarom het gebruik van een while lus in de methode verwerkRijVoorKassa() handiger is dan een for lus.

Er is minder code nodig om hetzelfde resultaat te bereiken. Werkt met Boolean, je hoeft geen extra teller bij te houden.

Met een for lus worden er personen overgeslagen. Door de teller

### 4B. Bij welke methodes in Kassa en Kantine komt dit voor(dubbele methode).

HoeveelheidGeldInKassa() en AantalArtikelen().

### 5A.Leg uit waarom het goed is om de methodes ArrayList<Artikel>getArrayList(Stringproductnaam)enArtikelgetArtikel(ArrayList<Artikel>)privatete maken.

Omdat het belangrijk is dat niet iedereen deze methode kan aanroepen.

### 5B.In welke situatie gebruik je een HashMap en wanneer een HashSet?

**Hasmap** is voor het opslaan als iets 2x in de map voor mag komen.

**Hashset** als het uniek moet zijn en een waarde niet dubbel mag zijn.

### 6A. Leg de werking van de constructor uit.

De constructor maakt een nieuwe kantine aan en een random getal. Dan een Array int[] hoeveelheden krijgt de waarden van getRandomArray(aantal\_artikelen, min\_artikelen\_per\_soort, max\_artikelen\_per\_soort). Vervolgens wordt er een nieuwe kantineaanbod aangemaakt en deze wordt gezet met het huidige kantineaanbod.

### 6B. Leg de implementatie van intgetRandomValue(int min, int max) uit en met name waarom er +1 in voorkomt.

De methode genereert een waarde binnen de opgegeven parameters. De reden waarom +1 wordt gebruikt is omdat je anders niet aan je minimale en maximale waarden kan komen.

## Week 3

### 2C. Er is geen constructor gedefinieerd voor Administratie terwijl je gewoon new Administratie() kan aanroepen. Leg uit waarom dat kan

Er wordt automatisch door Java een lege constructor gemaakt als deze niet word aan gemaakt.

### 2D. Leg uit waarom de twee al bestaande methoden van Administratie static kunnen zijn.

Deze methode kan static zijn omdat we deze simpele methoden voor het uitrekenen van het gemiddelde niet afhankelijk hoeven te zijn van een gemaakt Administratie object.

We kunnen het overal aanroepen en een static methode kunnen we aanroepen op basis van de klasse en niet het gemaakte object.

### 2E. Als je dat wil voorkomen kun je een private constructor voor Administratie maken. Doe dat en leg uit waarom je je doel nu bereikt.

Er veranderd niets omdat hij eerst zelf gemaakt werd en nu door ons.

### 2G.Leg uit wat final doet.

Een final waarde, methode of klasse kan niet worden veranderd. Nadat deze is aangemaakt kan deze niet worden gewijzigd.

### 2H. Als het goed is klaagt de compiler over zoiets als “Cannot make a static reference to the non-static field ...”. Leg uit waarom de compiler hierover klaagt.

De compiler klaagt over het maken van een static reference naar een non-static field omdat een static methode geldt voor de gehele klasse zonder een instantieobject. Omdat een non-static field alleen komt te bestaan bij het instantiëren van een object kan een static methode niet refereren naar een non-statisch veld omdat deze niet altijd hoeft te bestaan.

### 2I. Welk “probleem” heb je nu geïntroduceerd?

Nu staat het veld DAYS\_IN\_WEEK niet vast en zou het later opnieuw gedefinieerd kunnen worden en dat zou exact zijn wat we niet willen.

### 3B Waarom moet een super aanroep in de constructor altijd bovenaan staan?

Zorgt ervoor dat een parant klasse goed word geinisaloisserd zodat deze goed kan worden a angeroepen.