Java Programming Fundamentals

Oefeningen

INHOUDSTAFEL

Oef. bij hoofdstuk 2: Variabelen en operatoren

Oefening 1:

Maak een java-programma waarin verschillende data-types gedeclareerd en geïnitialiseerd worden.

Gebruik de instructie System.out.println() om het resultaat te controleren.

Oefening 2:

Een student behaalt op een examen volgende scores: 8, 6, 9 en 4.

Maak een programma dat het gemiddelde berekent (als decimaal getal) en het behaalde percentage (als je weet dat de maximum score op ieder vak 10 is).

Gebruik geen array's mocht je daar al kennis van hebben.

Oefening 3:

Gegeven : een te betalen bedrag voor een snoep uit een snoepautomaat. De kostprijs van de verschillende snoepen varieert tussen € 0,30 en € 1,20. De klant kan enkel betalen met een stuk van € 2. Het programma moet het wisselgeld uitrekenen : hoeveel stukken van € 1, € 0,50, € 0,20, € 0,10, € 0,05, € 0,02 en € 0,01 moeten er teruggegeven worden. Steeds met zo weinig mogelijk munten !

Test het programma voor een aankoop die € 0,42 kost, voor een aankoop die € 1,02 kost, …

Opgelet: bij bewerkingen met decimale getallen (float, double) geven bewerkingen niet altijd een correct resultaat: 10.0 / 2.0 = 4.9999993 (of zo iets). Gebruik dus de gepaste datatypes.

Oefening 4:

Maak een programma dat een geheel aantal seconden, bijvoorbeeld 5924, omrekent in uren, minuten en seconden.

Het resultaat: U:1 M:38 S:44

Oef. bij hoofdstuk 3: Arrays

Oefening 5:

We maken een array van 5 integers.

De waarden worden opgevuld met willekeurige getallen tussen 1 en 100 (grenswaarden inbegrepen). Gebruik hiervoor de method Math.random().

Opgelet: deze method geeft een *double* terug tussen 0 (inbegrepen) en 1 (niet inbegrepen).

Nadat de array gevuld is, wordt gevraagd om de vijf getallen, de som en de gemiddelde waarde (als decimaal getal) te tonen. Om het gemiddelde te berekenen gebruik je een property van de array om te weten hoeveel elementen er in de array aanwezig zijn.

Oef. bij hoofdstuk 4: Programmaverloop

Oefening 6:

Herneem vorige opdracht maar realiseer nu de oplossing door gebruik te maken van iteraties.

Oefening 7:

Controle op de random-generator.

Laat de randomgenerator 10.000 willekeurige gehele getallen genereren tussen 1 en 100 (grenswaarden inbegrepen).

Toon op het scherm hoe dikwijls ieder getal is gegenereerd.

Oefening 8:

Laat met de randomgenerator 100 getallen genereren tussen 1 en 1000 en stop ze in een array.

Sorteer de getallen en voer ze uit van klein naar groot.

Oefening 9:

Genereer lotto-getallen, d.w.z. 6 verschillende getallen tussen 1 en 42 (inbegrepen) + een reservegetal.

Publiceer in opklimmende volgorde.

Oefening 10:

Maak een programma waarin je de gebruiker vraagt hoeveel huisdieren hij/zij heeft. Breng voor de aantallen 0 t/m 3 telkens een gepaste melding op het scherm. Voor een aantal groter dan 3, breng een standaardbericht op het scherm.

Oef. bij hoofdstuk 5: OO, classes en objects

Oefening 11:

a) Maak een class Getal die enkel een public int-property x heeft (geen constructor).

   Maak een tweede class met een main-method. In deze method maak je een instance van de class Getal aan. Voer de waarde van x uit naar het scherm door rechtstreeks de property x aan te spreken.

b) Maak in class Getal de property x private. Wat stel je vast in je main-class en waarom ?

c) Voeg aan de class Getal de public method `print()` toe die de waarde van de membervariabele x op het scherm toont.

   Wijzig de method `main()` zodat deze gebruik maakt van de nieuwe method om de waarde van de membervariabele x op het scherm te tonen.

d) Voeg nu een constructor aan de class Getal toe. Deze constructor heeft een parameter (een int a). De constructor kent de waarde van a toe aan de membervariabele x.

   Herschrijf daarop de regel uit de `main()` die het object van Getal aanmaakt zodat deze nu de nieuwe constructor gebruikt. Merk op dat er nu geen default constructor meer wordt gemaakt.

e) Voeg aan Getal de method `absoluut()` toe. Deze method heeft als returnwaarde de absolute waarde van de membervariabele x (x zelf mag niet veranderen).

   Tip : om de absolute waarde te berekenen kan je de functie `Math.abs(x)` gebruiken.

   Wijzig de `main()` zodat hij gebruik maakt van deze method of anders gezegd: laat de `main()` bijvoorbeeld de absolute waarde van -3 op het beeldscherm tonen.

f) Voeg aan Getal een method `som(int a)` toe die de som van de member-variabele x en a teruggeeft. Test deze nieuwe method in de `main()`.

g) Voeg aan Getal een method `add(int a)` toe die de membervariabele x verhoogt met a. Wijzig de `main()` zodat die een object van de class Getal aanmaakt, de waarde van de membervariabele x op het scherm toont, deze waarde verhoogt met een int-getal d.m.v. de nieuwe method en tot slot deze nieuwe waarde van de membervariabele x op het scherm toont.

h) Nu gaan we de method `som(int a)` overloaden met nieuwe methods `float som(float a)` en `double som(double a)`. Test deze nieuwe methods in de main().

i) Maak een method `toDouble()` die als returnwaarde de waarde van membervariabele x als een double teruggeeft. Test deze nieuwe method in de main().

j) Maak een method `getX()` die de waarde van de membervariabele x teruggeeft en een method `setX(int a)` die de waarde van a toekent aan de membervariabele x.
Wijzig de constructor die dus gebruik zal gaan maken van deze set()-method.
Wijzig de main() zodat deze gebruik maakt van de get() method om de waarde op te vragen en te tonen op het scherm.

Oefening 12:

Maak eerst een class Waarnemer. Deze class kan een minimumtemperatuur, een maximumtemperatuur en een gemiddelde temperatuur bijhouden (property's) en weergeven (getter-methods). Verder heeft deze class een method die deze 3 statistieken bijwerkt wanneer er een nieuwe temperatuur geregistreerd werd.

Maak een tweede class (met main-method) die geregistreerde temperaturen inleest. Het programma stopt met de ingave van de temperatuur 999. Voer dan het aantal ingegeven waarden uit, de hoogste waarde, de laagste waarde en de gemiddelde waarde.

De in te geven waarden zijn integers. Het berekend gemiddelde is een decimaal getal. Vergeet niet in de class Waarnemer de temperaturen te initialiseren.

Oefening 13:

a) Maak een class *Student* waar je de ***naam*** (String) en ***score*** (int) van een *Student* kan bewaren. Zorg voor de nodige getters en setters (getNaam(), getScore(),…).

   Maak in een main-class *Examen* twee studenten aan en geef hen een naam en een score. Breng de gegevens op het scherm.

b) Maak constructors aan in de class *Student* die de naam of de naam en de score van de Student ontvangen. Wordt er geen score meegegeven dan wordt die ingesteld op nul.

c) Test de gemaakte classes in de main().

Oefening 14:

Schrijf een class Kaart.

Een kaart heeft:

- een **kleur** (harten, ruiten, klaveren of schoppen) en
- een **rang** (2, 3, 4, 5, 6, 7, 8, 9, 10, boer, vrouw, heer, aas).

Maak een constructor die een willekeurige kaart aanmaakt.

Maak een method `printKaart()` die de kleur en de rang van een kaart weergeeft.

Voeg aan de class Kaart een method `isHogerDan()` toe die de kaart vergelijkt met een andere kaart: wanneer de kaart hoger in kleur en rang is dan de andere kaart, geeft deze method de waarde `true` terug, anders `false`.

De volgorde van de kleur en rang, die tussen de haakjes vermeld staat, is van klein naar groot opgesomd.

Schrijf tevens een main()-programma waarin:

- een eerste kaart wordt aangemaakt
- de gegevens van deze kaart worden getoond op het scherm
- een tweede kaart wordt aangemaakt
- en ook hiervan de gegevens op het scherm worden getoond
- de eerste kaart vergeleken wordt met de tweede kaart en aangeeft of deze kaart hoger in rang is dan de tweede kaart.

Tip: Je kan deze oefening op 2 manieren oplossen:

a) **zonder** gebruik van **enum** voor kleur en rang

b) **met** gebruik van **enum** voor kleur en rang: dan kan je gebruik maken van de ingebouwde `compareTo()` method van een enum. Raadpleeg hiervoor eerst de documentatie.

Oef. bij hoofdstuk 6: Inheritance

Oefening 15:

a) de class Voertuig

Maak een class Voertuig met setter en getter methods voor:

- een property *polishouder* van het type String
- een property *kostprijs* van het type float
- een property *pk* van het type int
- een property *gemVerbruik* van het type float
- een property *nummerplaat* van het type String

De default polishouder en nummerplaat zijn “onbepaald” en de kostprijs staat op 0, evenals het aantal pk en het gemiddeld verbruik.

Maak een constructor zonder parameters.

Maak ook een constructor die de polishouder, de kostprijs, het aantal pk, het gemiddeld verbruik en een nummerplaat aanvaardt.

Return alle gegevens in één String in een toString() method. Eerst de polishouder dan de kostprijs, het aantal pk, het verbruik en een nummerplaat.

Voorzie ook een method toon(), die uiteraard alle gegevens toont op het scherm.

b) de class Vrachtwagen

Maak een class Vrachtwagen, die is afgeleid van Voertuig en setter en getter methods heeft voor:

- maxLading van het type float

De default maxLading is 10000 kg en mag niet negatief zijn.

Voorzie ook een constructor waarbij je alle waardes via parameters meegeeft.

Zorg ervoor dat de gegevens van de Vrachtwagen kunnen getoond worden op het scherm en override de toString()-method.

c) de class Personenwagen

Maak een class Personenwagen, die is afgeleid van Voertuig en setter en getter methods heeft voor:

- aantalDeuren van het type int
- aantalPassagiers van het type int

Het default aantal deuren is 4, het default aantal passagiers is 5.
Deze properties mogen niet negatief zijn.

Voorzie ook een constructor waarbij je alle waardes via parameters meegeeft.

Zorg ervoor dat de gegevens van de personenwagen kunnen getoond worden op het scherm en override de toString()-method.

d) de abstracte method getKyotoScore ()

Maak de class Voertuig abstract en voeg de volgende abstracte method toe :

```
public double getKyotoScore()
```

Deze method returnt een Kyoto-score. Voor een personenwagen is de Kyoto-score gelijk aan het verbruik vermenigvuldigd met het aantal pk, gedeeld door het aantal passagiers.

Voor een vrachtwagen is de Kyoto-score gelijk aan het verbruik vermenigvuldigd met het aantal pk, gedeeld door de lading in ton.

Werk getKyotoScore() uit in de diverse classes.

e) Hoofdprogramma

Test alles uit in een hoofdprogramma.

Oefening 16:

Probeer deze oefening op te lossen zonder computer.

Beschouw een class Punt. Deze bestaat uit :

- twee membervariabelen ***int x*** en ***int y*** die de coördinaten van het punt voorstellen.
- een method ***melding()*** die een bericht op scherm toont
- een method ***schrijf()*** voor het tonen van de waarde van deze coördinaten.

Beschouw een class GekleurdPunt, afgeleid van Punt. Deze heeft:

- een extra membervariabele ***String kleur***.
- een method ***schrijf()*** voor het tonen van de waarde van de coördinaten en het kleur.

Beschouw een class GekleurdPuntMetDikte, afgeleid van GekleurdPunt. Deze heeft:

- een extra membervariabele ***int dikte***.
- een method ***schrijf()*** voor het tonen van de waarde van de coördinaten, het kleur en de dikte van het punt.
- een method ***extra()*** voor het tonen van een extra bericht

Beschouw nu het volgende programma. Stel vast en verklaar wat er gebeurt.

```
public class Main {
    public static void main(String[] args) {
        Punt p1 = new Punt();
        Punt p2 = new Punt();
        Punt g1 = new GekleurdPunt() ;
        Punt g2 = new GekleurdPunt() ;
        GekleurdPuntMetDikte d1 = new
           GekleurdPuntMetDikte() ;
        GekleurdPuntMetDikte d2 = new
           GekleurdPuntMetDikte() ;
        Punt p3 = new GekleurdPuntMetDikte() ;
        p3.schrijf() ;
        d1.schrijf() ;
        g1 = d1;
        g1.schrijf() ;
        p3.extra();
        g1.melding();
        d1 = p1;
        d1.melding();
    }
}
```

Oef. bij hoofdstuk 7: Strings

Oefening 17:

De gebruiker tikt een zin in.
Breng op het scherm hoeveel klinkers er in de zin staan

Oefening 18:

Maak een class die een input onder de vorm

17 + 38 * 2 – 22

aanvaardt, dit uiteenrafelt in integers en bewerkingstekens, die de bewerkingen uitvoert en die het resultaat publiceert.
Het scheidingsteken is een spatie!

Tip: Zoek een geschikte method in de class String die hier handig gebruikt kan worden.

Ga er van uit dat de gebruiker geen fouten maakt bij het ingeven. Voer alle bewerkingen uit van links naar rechts en niet volgens de juiste wiskundige rekenvolgorde (* / + -).

Oefening 19:

Maak een class die in staat is om te onderzoeken of een ingevoerde tekst een palindroom is, hierbij al dan niet rekening houdend met hoofd- of kleine letters.
Voorbeelden van palindrooms: kok, lepel, parterretrap, snelmeetsysteemlens.

Maak een main waarin je een woord opvraagt en dit woord test.

Tip:
Gebruik een StringBuffer. Deze klasse heeft een method ***reverse()*** zodat men onmiddellijk kan vergelijken of een tekst en zijn omgekeerde gelijk zijn.

Oef. bij hoofdstuk 8: Interfaces

Oefening 20:

Voor deze oefening werk je verder met de oplossing van oefening 15. We breiden deze oefening uit met een interface Vervuiler :

```
public interface Vervuiler {
        public double geefVervuiling();
}
```

Implementeer deze interface in de classes Vrachtwagen en Personenwagen. Voor de personenwagen is de vervuiling gelijk aan de Kyotoscore maal 5, voor de vrachtwagen de Kyotoscore maal 20.

Maak ook een nieuwe class Stookketel. Deze class heeft een float-property *CONorm* (met bijhorende getter en setter). Laat deze class de interface Vervuiler implementeren. De vervuiling is de CONorm maal 100.

In het hoofdprogramma maak je vervolgens een array van Vervuiler. Je plaatst hierin een aantal objecten die de interface Vervuiler implementeren. Je overloopt de array en van elk object toon je het resultaat van de method geefVervuiling().

Oefening 21:

In deze oefening breiden we oefening 20 nog verder uit.

Maak twee interfaces : Privaat en Milieu. In de interface Privaat voorzie je een void-method geefPrivateData(), in de interface Milieu een void-method geefMilieuData().

Werk deze interfaces uit in Voertuig door in de method geefPrivateData () enkel de polishouder en de nummerplaat uit te voeren naar het scherm. Maak ook een method geefMilieuData() waarin je de properties pk, kostprijs en verbruik uitvoert.

In het hoofdprogramma maak je een array van Privaat. Stop een aantal voertuigen (vrachtwagens en personenwagens) in deze array en laat in een lus alle private gegevens zien. Probeer ook eens een array van Milieu te maken met eveneens een aantal voertuigen erin en ga eens na welke methods er nu beschikbaar zijn.

Oefening 22:

Maak een interface Voorwerp. Deze interface beschrijft een void-method gegevensTonen() en een double-method winstBerekenen().

Maak volgende classes die allen de interface Voorwerp implementeren :

- Een class boekenrek met properties hoogte, breedte, aankoopprijs en winst. Deze winst is gelijk aan de aankoopprijs x 2.
- Een class boek met properties titel, auteur, eigenaar, aankoopprijs en een genre. Alle boeken hebben als eigenaar “VDAB”.
- Een class leesboek, afgeleid van boek, met property onderwerp en een winst die gelijk is aan de aankoopprijs x 1,5
- Een class woordenboek, afgeleid van boek, met property taal en een winst gelijk aan de aankoopprijs x 1,75

Voorzie voor alle classes een toString()-method.

In het hoofdprogramma maak je 2 leesboeken, 2 woordenboeken en 2 boeken-rekken. Telkens één keer gebruik makend van een defaultconstructor en één keer aan de hand van een constructor met parameters.

Stop deze objecten in een array van het interfacetype Voorwerp. Doorloop deze array en toon alle gegevens van de arrayelementen. Totaliseer de winst en voer deze uit.

Oef. bij hoofdstuk 9: Packages

Oefening 23:

In deze oefening werk je de oplossing van oefening 22 verder uit.

Stop de objecten als volgt in een package-structuur:

Package **be.vdab.util** bevat:

- de interface Voorwerp

Package **be.vdab.voorwerpen** bevat:

- de class Boekenrek
- de class Boek
- de class Leesboek
- de class Woordenboek

Het hoofdprogramma zit in de package **be.vdab**.

Oef. bij hoofdstuk 10: Exception handling

Oefening 24:

Bestudeer het volgende programma en noteer wat de output is :

```
public class TestExceptions {
  public static void main(String[] args) {
    String test = "no";
    try {
      System.out.println("start try");
      doRisky(test);
      System.out.println("end try");
    }
    catch ( ScaryException se) {
      System.out.println("scary exception");
    }
    finally {
      System.out.println("finally");
    }
    System.out.println("end of main");
  }

  static void doRisky(String test) throws ScaryException {
    System.out.println("start risky");
    if ("yes".equals(test)) {
      throw new ScaryException();
    }
    System.out.println("end risky");
    return;
  }
}
```

Noteer vervolgens wat de output is indien de derde regel gewijzigd wordt in
`String test = "yes";`

Oefening 25:

In deze oefening werken we verder met rekeningen, spaar- en zichtrekeningen.

Voorzie een eigen exception RekeningException die gethrowd wordt wanneer er een foutief rekeningnummer wordt opgegeven.

Oef. bij hoofdstuk 12: Collections

Oefening 26 :

Maak een class Land. Deze class bevat de properties landcode, landnaam, hoofdstad, aantal inwoners, oppervlakte en één method die de bevolkingsdichtheid berekent.

Maak een tweede class (met main) waarin een arraylist gemaakt wordt van minstens 10 landen.

Voer minstens de gegevens van het land uit waarvan de bevolkingsdichtheid het dichtst aanleunt bij de gemiddelde bevolkingsdichtheid van alle landen.

Publiceer alle berekende aantallen met 2 cijfers na het decimaal teken.
Tip: zoek hiervoor in de documentatie bij de class String naar de method ***format()***.

Oefening 27 :

In deze oefening werken we verder met de voertuigen, vrachtwagens en personenwagens uit oefening 15 (of 20 en 21).

In de main creëren we telkens enkele voertuigen van verschillende types (bijvoorbeeld een personenwagen en twee vrachtwagens) en voegen die toe aan een TreeSet.

Doorloop de TreeSet volgens de sleutel, nl. in stijgende volgorde van nummerplaat en toon de informatie van de voertuigen.

Oefening 28 :

Bepaal hoeveel woorden beginnen met een bepaalde letter:

- Stockeer hiervoor eerst een aantal woorden in een array van Strings
- Bepaal dan door gebruik te maken van een HashMap hoeveel woorden beginnen met een letter.
  Deze hashMap zal dus key-value-paren bevatten (K=beginletter, V=aantal voorkomens)

Toon vervolgens volgende resulaten:

- aantal woorden per letter
- de grootte van de HashMap
- een afdruk van alle keys
- een afdruk van alle values
- een afdruk van alle map.entry – elementen

Oefening 29 :

Bewaar voor deze oef alles in package **be.vdab.winkel**

Schrijf een **klasse Product** met 2 membervariabelen:

- een omschrijving
- een prijs

Schrijf een **klasse Catalogus:**

- met 1 membervariabele: een verzameling van producten (List)
- een constructor om deze verzameling te vullen
- een method om de iterator op te vragen

Schrijf een **klasse Mandje** die een verzameling kan bevatten van de gekochte producten met hun aantallen (Gebruik een **HashMap** `HashMap <Product,Integer>`).
Verder heeft deze klasse ook volgende methods:

- method add (Product, int)
- method set (Product, int): om aantal van het gekochte product te wijzigen
- method remove(Product): verwijdert één product uit het mandje
- method clear(): maakt het mandje leeg
- method getSom: geeft totaal bedrag van de aankopen in het mandje
- method iterator()

Bedoeling is om via een main()-programma een overzicht te tonen van alle producten uit dit mandje (de inhoud van het mandje). Vul daarom eerst dit mandje programmatorisch mbv de methods add en set.
Tot slot toon je het totaal te betalen bedrag van de aankopen.

Oefening 30 :

Maak een class Tienkamper met de properties *naam* (String) en *punten* (int). Voeg de nodige getters en setters toe. Schrijf de toString(), de equals(), de hashCode() en de compareTo()-method en baseer deze op de *naam*.

Hoofdprogramma:

- Vul een collection (ArrayList) met een aantal atleten en toon ze.
- Maak een tweede collection (TreeSet) en vul deze met dezelfde objecten. Toon opnieuw de lijst.

Schrijf een eigen Comparator om de atleten op volgorde van hun punten te tonen. Gebruik deze comparator bij de arraylist en de treeset en toon de atleten.

Oef. bij hoofdstuk 13: Streams

Oefening 31:

In deze oefening gaan we een programma schrijven om een gastenboek bij te houden. Het gastenboek is een binair bestand *gastenboek.dat*. In het gastenboek worden entries weggeschreven, bestaande uit een tijdstip, een naam (van de schrijver) en zijn/haar boodschap. Het programma laat ook toe de lijst met alle entries te lezen en uit te voeren naar het scherm.

Maak eerst een GastenBoekEntry-class. Deze bevat properties datum, schrijver en boodschap. Vervolgens maak je een GastenBoek-class. Deze bevat één property, namelijk een ArrayList van GastenBoekEntries. Verder twee methods: één om een entry toe te voegen aan de ArrayList en één om alle entries op te vragen, eigenlijk een getMethod van de property (van de arraylist dus).

Maak een class GastenBoekManager die in staat is om in het gastenboek te schrijven. Per bericht in het gastenboek wordt een timestamp, de opsteller en de boodschap geregistreerd (omwille van praktische redenen is de boodschap beperkt tot één regel). Deze class heeft ook een tweede method waarmee je het gastenboek kan inlezen.

Maak een main-class die de keuze laat tussen lezen of schrijven en die de verdere input/output via de console afhandelt: wanneer gekozen is om te lezen wordt het gastenboek ingelezen en worden de gastenboekentries gepubliceerd. De meest recente items staan bovenaan, dus de volgorde van publicatie is omgekeerd t.o.v. de ingave. Wanneer gekozen is om te schrijven, wordt er een naam en boodschap gevraagd zodat er een gastenboekentry kan worden toegevoegd aan het gastenboek.

Oef. bij hoofdstuk 14: MultiThreading

Oefening 32:

Je maakt een class Main met een method public static void main(String args[]);

Je maakt in deze method een double array met 1.000.000 random getallen.

Je wil het gemiddelde kennen van deze getallen.

Je doet dit op volgende manier:

Één thread berekent het gemiddelde van de eerste 500.000 getallen.

Een tweede thread berekent gelijktijdig het gemiddelde van de laatste 500.000 getallen.

De method main vraagt deze gemiddeldes op en maakt het gemiddelde van deze gemiddeldes. Dit is gemiddelde van de 1.000.000 getallen

Oefening 33:

Je voegt aan de applicatie uit de theorie met de classes Kok en Stapel een class Klant toe. Deze class implementeert de interface Runnable.

Deze class stelt een klant voor die 50 iteraties uitvoert.
Bij iedere iteratie neemt de klant een pannenkoek van de stapel, op voorwaarde dat de stapel minstens één pannenkoek bevat.
Als volgende stap binnen de iteratie wacht de Klant 100 milliseconden.

Je maakt in de method main van de class Main naast de twee threads die een Kok voorstellen ook vier threads die een Klant voorstellen en je voert alle threads uit.

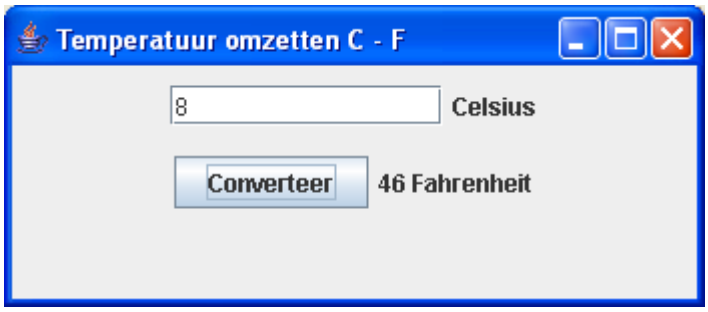
Je toont daarna hoeveel pannenkoeken er op de stapel liggen.

Gezien 2 koks elk 100 pannenkoeken bakten en 4 klanten elk 50 pannenkoeken bestelden, moet de stapel op het einde van de applicatie 0 pannenkoeken bevatten.

Oef. bij hoofdstuk 15: GUI

Oefening 34:

Temperatuurconversie Celsius – Fahrenheit



In deze toepassing kan de gebruiker een waarde ingeven in een tekstvak. Wanneer de gebruiker op de knop Converteer drukt, wordt de temperatuur in Fahrenheit berekend en het resultaat getoond.

Formule: °F = 9/5 °C + 32

Oefening 35:

Je kan met de knoppen links en rechts de middelste knop desactiveren / activeren.
Wanneer je klikt op de middelste knop krijg je een berichtvenster te zien.
De middelste knop heeft een icoontje.

Oefening 36:

Maak een class **GUIElementenSwingOef** waarbij volgende ingave mogelijk is:

Tracht te zorgen voor een goede layout.

Na een klik op de OK-knop, worden alle ingegeven en geselecteerde gegevens getoond.

Oefening 37:

Schrijf een toepassing **AnnuiteitSwing** voor de berekening van de constante periodieke afbetaling bij een op te geven kapitaal, een aantal perioden en een periodieke rentevoet.

Gebruik de volgende formule : $J = T.\frac{i}{1-\frac{1}{(1+i)^n}}$

Hierbij is J het jaarlijks te betalen bedrag (voor de maandaflossing nog te delen door 12), T is het ontleend kapitaal, i is de periodieke rentevoet en n is het aantal jaarlijkse termijnen (aantal perioden gedeeld door 12).

Oefening 38:

Maak een applicatie met volgende lay-out:

Via de knoppen kan men nieuwe elementen toevoegen, elementen verwijderen en de ordening van de elementen aanpassen.

Tip:

- Voor het toevoegen: gebruik een input-dialog box om de nieuwe waarde te vragen.
- Gebruik de API-documentatie om de methods van een JList en een DefaultListModel op te zoeken.

Oplossingen oef. bij hoofdstuk 2: Variabelen en operatoren

Oplossing oefening 1

```
/*
 * Main.java
 *
 * Created on 8 mei 2007, 8:39
 *
 * To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */

package oefening1;

/**
 *
 * @author slucas
 */
public class Main {

    /** Creates a new instance of Main */
    public Main() {
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        int geheelGetal = 5;
        long teller = 34521123L;
        float decimaalGetal = 1.5F;
        double grootDecGetal = 123456789.34;
        char eenTeken = 'a';
        boolean gehuwd = true;
        String verhaal = "Er was eens...";
        System.out.println(geheelGetal);
        System.out.println(teller);
        System.out.println(decimaalGetal);
        System.out.println(grootDecGetal);
        System.out.println(eenTeken);
        System.out.println(gehuwd);
        System.out.println(verhaal);
    }
}
```

Oplossing oefening 2

```
public static void main(String[] args) {
  int score1 = 8;
  int score2 = 6;
  int score3 = 9;
  int score4 = 4;
  int totaalScore = score1 + score2 + score3 + score4;
  float gemiddelde = totaalScore / 4f;
  float percentage = totaalScore * 100f / 40f;

  System.out.println("Het gemiddelde is " + gemiddelde +
                     " op 10");
  System.out.println("Het behaalde percentage : " + percentage +
                     "%");
}
```

Oplossing oefening 3

```
  public static void main(String[] args) {

      float teBetalen = 0.42F;
      int teBetalenInCent = (int)(teBetalen * 100);
      int terugTeGevenInCent = 200 - teBetalenInCent;
      System.out.println("Te betalen : " + teBetalen);
      System.out.println("Terug te geven : " +
                         terugTeGevenInCent / 100.);

      int stukkenVanEenEuro = terugTeGevenInCent / 100;
      System.out.println("Stukken van 1 euro : " +
                         stukkenVanEenEuro);
      terugTeGevenInCent -= stukkenVanEenEuro * 100;

      int stukkenVanVijftigCent = terugTeGevenInCent / 50;
      System.out.println("Stukken van 50 cent : " +
                         stukkenVanVijftigCent);
      terugTeGevenInCent -= stukkenVanVijftigCent * 50;

      int stukkenVanTwintigCent = terugTeGevenInCent / 20;
      System.out.println("Stukken van 20 cent : " +
                         stukkenVanTwintigCent);
      terugTeGevenInCent -= stukkenVanTwintigCent * 20;

      int stukkenVanTienCent = terugTeGevenInCent / 10;
      System.out.println("Stukken van 10 cent : " +
                         stukkenVanTienCent);
      terugTeGevenInCent -= stukkenVanTienCent * 10;
```

```
        int stukkenVanVijfCent = terugTeGevenInCent / 5;
        System.out.println("Stukken van 5 cent : " +
                           stukkenVanVijfCent);
        terugTeGevenInCent -= stukkenVanVijfCent * 5;

        int stukkenVanTweeCent = terugTeGevenInCent / 2;
        System.out.println("Stukken van 2 cent : " +
                           stukkenVanTweeCent);
        terugTeGevenInCent -= stukkenVanTweeCent * 2;

        int stukkenVanEenCent = terugTeGevenInCent;
        System.out.println("Stukken van 1 cent : " +
                           stukkenVanEenCent);
    }
```

Oplossing oefening 4

```
    public static void main(String[] args) {

        int totSec = 5924;
        int uren = totSec / 3600;
        int rest = totSec % 3600;
        int minuten = rest / 60;
        int seconden = rest % 60;

        System.out.println("U:" + uren +
                           " M:" + minuten +
                           " S:" + seconden);
    }
```

Oplossingen oef. bij hoofdstuk 3: Arrays

Oplossing oefening 5

```
public static void main(String[] args) {

    int[] getallen = new int[5];
    getallen[0] = (int)(Math.random() * 100) + 1;
    getallen[1] = (int)(Math.random() * 100) + 1;
    getallen[2] = (int)(Math.random() * 100) + 1;
    getallen[3] = (int)(Math.random() * 100) + 1;
    getallen[4] = (int)(Math.random() * 100) + 1;

    int som;
    float gemiddelde;
    som = getallen[0] + getallen[1] + getallen[2] +
          getallen[3] + getallen[4];
    gemiddelde = (float)som / getallen.length;
    System.out.println(getallen[0]);
    System.out.println(getallen[1]);
    System.out.println(getallen[2]);
    System.out.println(getallen[3]);
    System.out.println(getallen[4]);
    System.out.println("Som = " + som);
    System.out.println("Gemiddelde = " + gemiddelde);

 }
```

Oplossingen oef. bij hoofdstuk 4: Programmaverloop

Oplossing oefening 6

```
public static void main(String[] args) {
  int[] getallen = new int[5];
  int som = 0;
  for (int i=0; i != getallen.length ; i++){
    getallen[i] = (int)(Math.random()*100) + 1;
    som += getallen[i];
  }
  float gemiddelde = (float)som / getallen.length;
  for (int i = 0; i != getallen.length; i++) {
    System.out.println(getallen[i]);
  }
  System.out.println("Som = " + som);
  System.out.println("Gemiddelde = " + gemiddelde);
}
```

Oplossing oefening 7

```
public static void main(String[] args) {
  int[] getallen = new int[100];

  // genereren van 10000 willekeurige getallen
  for(int i = 0; i<10000; i++) {
    int randGetal = (int)(Math.random()*100);
    getallen[randGetal] +=1;
  }
  // publiceren van het resultaat
  for(int i = 0; i != getallen.length; i++) {
    System.out.println("getal " + (i+1) + " : " +
                       getallen[i]);
  }
}
```

Oplossing oefening 8

```
    public static void main(String[] args) {

        int[] getallen = new int[100];
        // genereren van 100 willekeurige getallen
        for(int i = 0; i != getallen.length; i++) {
           getallen[i] = (int)(Math.random()*1000 + 1);
        }

        // sorteren van de 100 getallen
        for(int i = 0; i != getallen.length - 1; i++) {
            for(int j = i+1; j != getallen.length; j++) {
                if (getallen[j] < getallen[i]) {
                    int tempGetal = getallen[i];
                    getallen[i] = getallen[j];
                    getallen[j] = tempGetal;
                }
            }
        }

        // tonen resultaat
        for (int i = 0; i != getallen.length; i++) {
            System.out.println(getallen[i]);
        }
    }
```

Oplossing oefening 9

```
public class Main {
    public static void main(String[] args) {
        // declaratie en initialisatie variabelen
        int[] getallen = new int[7];
        int aantalGetallen = 0;

        // genereer 7 getallen
        while (aantalGetallen < 7) {
            //genereer een getal
            int randGetal = (int) (Math.random() * 42 + 1);
            // test of het getal niet reeds gekozen is
            boolean dubbel = false;
            for (int i = 0; i != getallen.length && !dubbel; i++) {
                if (getallen[i] == randGetal) {
                    dubbel = true;
```

```
                }
            }
            //getal bewaren indien niet dubbel
            if (!dubbel) {
                getallen[aantalGetallen++] = randGetal;
            }
        }
        // er zijn nu 7 verschillende getallen gevonden
        // de eerste 6 worden gesorteerd, het zevende is
        // het reservegetal
        for (int i = 0; i != getallen.length - 2; i++) {
            for (int j = i + 1; j != getallen.length - 1; j++) {
                if (getallen[j] < getallen[i]) {
                    int tijdGetal = getallen[i];
                    getallen[i] = getallen[j];
                    getallen[j] = tijdGetal;
                }
            }
        }
        // uitvoer
        System.out.println("De winnende lotto getallen zijn : ");
        for (int i = 0; i != getallen.length - 1; i++) {
            System.out.println(getallen[i]);
        }
        System.out.println("\nHet reservegetal is :");
        System.out.println(getallen[getallen.length - 1]);
    }
}
```

Oplossing oefening 10

```
public static void main(String[] args) {

  System.out.print("Geef het aantal huisdieren : ");
  Scanner sc = new Scanner(System.in);
  int aantal = sc.nextInt();
  switch (aantal) {
      case 0 :
          System.out.println(
            "Bedankt voor de medewerking aan onze enquête.");
          break;
      case 1 :
          System.out.println("U heeft één huisdier.");
          break;
      case 2 :
          System.out.println("Een hond en een kat?");
          break;
      case 3 :
          System.out.println("Dat kan al tellen!");
          break;
      default :
          System.out.println("Wat een beestenboel!");
          break;
  }
}
```

Oplossingen oef. bij hoofdstuk 5: OO, classes en objects

Oplossing oefening 11

```
public class Getal {

    public Getal(int a) {
        setX(a);
    }

    private int x;

    public void print(){
        System.out.println("Waarde x: " + x);
    }
    public int absoluut(){
        return Math.abs(x);
    }
    public int som(int a) {
        return x+a;
    }
    public float som(float a) {
        return x+a;
    }

    public double som(double a) {
        return x+a;
    }
    public void add(int a) {
        x+=a;
    }
    public double toDouble() {
        return (double)x;
    }
    public int getX() {
        return x;
    }
    public void setX(int a) {
        x = a;
    }
}
```

Oplossing oefening 12

De class Waarnemer:

```
public class Waarnemer {

    private int minimumTemperatuur = 99;
    private int maximumTemperatuur = -99;
    private float gemiddeldeTemperatuur = 0F;
    private int aantalWaarnemingen = 0;

    public void nieuweRegistratie(int temperatuur) {
        float som = gemiddeldeTemperatuur * aantalWaarnemingen +
                    temperatuur;
        aantalWaarnemingen++;
        gemiddeldeTemperatuur = som / aantalWaarnemingen;
        if (temperatuur < minimumTemperatuur) {
            minimumTemperatuur = temperatuur;
        }
        if (temperatuur > maximumTemperatuur) {
            maximumTemperatuur = temperatuur;
        }
    }

    public int getAantalWaarnemingen() {
        return aantalWaarnemingen;
    }
    public int getMinimumTemperatuur() {
        return minimumTemperatuur;
    }
    public int getMaximumTemperatuur() {
        return maximumTemperatuur;
    }
    public float getGemiddeldeTemperatuur() {
        return gemiddeldeTemperatuur;
    }
}
```

De class Main :

```
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    Waarnemer w = new Waarnemer();

    System.out.print("Geef een temperatuur:");
    int ingelezenTemperatuur = sc.nextInt();
    while (ingelezenTemperatuur != 999) {
        w.nieuweRegistratie(ingelezenTemperatuur);
        System.out.print("Geef een nieuwe temperatuur:");
        ingelezenTemperatuur = sc.nextInt();
    }
    System.out.println("Het resultaat van " +
                       w.getAantalWaarnemingen() +
                       " waarnemingen:");
    System.out.println("Minimum:" + w.getMinimumTemperatuur());
    System.out.println("Maximum:" + w.getMaximumTemperatuur());
    System.out.println("Gemiddelde:" +
                      w.getGemiddeldeTemperatuur());
}
```

Oplossing oefening 13

```
public class Student {

    public Student(String naam, int score) {
        setNaam(naam);
        setScore(score);
    }

    public Student(String naam) {
        setNaam(naam);
        setScore(0);
    }

    private String naam;
    private int score;

    public String getNaam(){
        return naam;
    }

    public int getScore(){
        return score;
    }

    public void setNaam(String naam){
        this.naam = naam;
    }

    public void setScore(int score) {
       this.score = score;
    }

}
```

```
    public static void main(String[] args) {
        Student Ruben = new Student("Ruben", 85);
        Student Freya = new Student("Freya", 75);
        System.out.print("Eerste student : " + Ruben.getNaam() +
          ", score=" + Ruben.getScore());
        System.out.print("Tweede student : " + Freya.getNaam() +
          ", score=" + Freya.getScore());
    }
```

Oplossing oefening 14

a) **zonder** gebruik van **enum**:

```
package oefening14;

public class Kaart {

  private int kleur, rang ;

  private static String kleuren[] = {"harten", "ruiten", "klaveren",
                                     "schoppen" } ;
  private static String rangen[]  = {"2", "3", "4", "5", "6", "7",
                "8", "9", "10", "boer", "vrouw", "heer", "aas" } ;
  //static --> gelijk voor alle objecten!!

  public Kaart() {
    kleur = (int) (Math.random() * 4  ) ;
    rang = (int) (Math.random() * 13 ) ;
  }

  public String getKleur() {
     return kleuren[kleur];
  }

  public String getRang() {
     return rangen[rang];
  }

  public void printKaart() {
     System.out.print("Kleur= " + getKleur() +
                      "\tRang= " + getRang() );
  }

  public boolean isHogerDan( Kaart k2 ) {
      if ( (kleur >= k2.kleur) && (rang > k2.rang) )
       return true ;
    else
       return false ;
  }
}
```

b) **met** gebruik van **enum**:

```
package oefening14;

public class Kaart {

    public enum Kleur {
        HARTEN, RUITEN, KLAVEREN, SCHOPPEN
    }

    public enum Rang {
        TWEE, DRIE, VIER, VIJF, ZES, ZEVEN, ACHT, NEGEN, TIEN,
        BOER, VROUW, HEER, AAS
    }

    private Kleur kleur;
    private Rang rang;

    public Kaart() {
        Kleur[] alleKleuren = Kleur.values();
        kleur = alleKleuren[(int) (Math.random() *
                                   alleKleuren.length)];

        Rang[] alleRangen = Rang.values();
        rang = alleRangen[(int) (Math.random() *
                                   alleRangen.length)];
    }

    public String getKleur() {
        return kleur.toString();
    }

    public String getRang() {
        return rang.toString();
    }

    public void printKaart() {
        System.out.print("Kleur= " + getKleur() +
                         "\tRang= " + getRang());
    }

    public boolean isHogerDan(Kaart k2) {
        return kleur.compareTo(k2.kleur) >= 0 &&
               rang.compareTo(k2.rang) > 0;
    }
}
```

Het main()-programma kan in beide gevallen gebruikt worden:

```
public static void main(String[] args) {
    Kaart eersteKaart = new Kaart();
    eersteKaart.printKaart();
    System.out.println();

    Kaart tweedeKaart = new Kaart();
    tweedeKaart.printKaart();
    System.out.println();

    eersteKaart.printKaart();
    System.out.print(" is");
    if (! eersteKaart.isHogerDan(tweedeKaart))
        System.out.print(" niet");
    System.out.print(" groter dan ");
    tweedeKaart.printKaart();
}
```

Oplossingen oef. bij hoofdstuk 6: Inheritance

Oplossing oefening 15

De class Voertuig:

```
public abstract class Voertuig {
    private String polishouder = "onbepaald";
    private float kostprijs;
    private int pk;
    private float gemVerbruik;
    private String nummerplaat = "onbepaald";

    public Voertuig() {
    }
    public Voertuig(String polishouder, float kostprijs,
            int pk, float gemVerbruik, String nummerplaat) {
        setPolishouder(polishouder);
        setKostprijs(kostprijs);
        setPk(pk);
        setGemVerbruik(gemVerbruik);
        setNummerplaat(nummerplaat);
    }

    public String getPolishouder() {
        return polishouder;
    }
    public final void setPolishouder(String polishouder) {
        this.polishouder = polishouder;
    }

    public float getKostprijs() {
        return kostprijs;
    }
    public final void setKostprijs(float kostprijs) {
        if (kostprijs > 0)
            this.kostprijs = kostprijs;
    }
```

```
    public int getPk() {
        return pk;
    }
    public final void setPk(int pk) {
        if (pk > 0)
            this.pk = pk;
    }

    public float getGemVerbruik() {
        return gemVerbruik;
    }
    public final void setGemVerbruik(float gemVerbruik) {
        if (gemVerbruik > 0)
            this.gemVerbruik = gemVerbruik;
    }

    public String getNummerplaat() {
        return nummerplaat;
    }
    public final void setNummerplaat(String nummerplaat) {
        this.nummerplaat = nummerplaat;
    }
    @Override
    public String toString() {
        return polishouder + ";" + kostprijs + ";" +
               pk + ";" + gemVerbruik + ";" + nummerplaat;
    }
    public void toon() {
        System.out.println("polishouder: " + polishouder);
        System.out.println("kostprijs: " + kostprijs);
        System.out.println("pk: " + pk);
        System.out.println("gemVerbruik: " + gemVerbruik);
        System.out.println("nummerplaat: " + nummerplaat);
    }
    public abstract double getKyotoScore();
}
```

De class Vrachtwagen :

```
public class Vrachtwagen extends Voertuig {
    private float maxLading = 10000.0F;

    public Vrachtwagen() {
    }
    public Vrachtwagen(String polishouder, float kostprijs, int pk,
          float gemVerbruik, String nummerplaat, float maxLading) {
        super(polishouder,kostprijs,pk,gemVerbruik,nummerplaat);
        setMaxLading(maxLading);
    }

    public float getMaxLading() {
        return maxLading;
    }
    public final void setMaxLading(float maxLading) {
        if (maxLading > 0)
            this.maxLading = maxLading;
    }

    @Override
    public String toString() {
        return super.toString() + ";" + maxLading;
    }

    @Override
    public void toon() {
        System.out.println("\nVRACHTWAGEN");
        super.toon();
        System.out.println("max. lading: " + maxLading);
    }

    public double getKyotoScore() {
        return (getGemVerbruik() * getPk() / (maxLading / 1000.0) );
        //lading omzetten van kg naar ton
    }
}
```

De class Personenwagen :

```
public class Personenwagen extends Voertuig {
    private int aantalDeuren=4;
    private int aantalPassagiers=5;

    public Personenwagen() {
    }
    public Personenwagen(String polishouder, float kostprijs,
            int pk, float gemVerbruik, String nummerplaat,
            int deuren, int passagiers) {
        super(polishouder,kostprijs,pk,gemVerbruik,nummerplaat);
        setAantalDeuren(deuren);
        setAantalPassagiers(passagiers);
    }

    public int getAantalDeuren() {
        return aantalDeuren;
    }
    public final void setAantalDeuren(int aantalDeuren) {
        if (aantalDeuren > 0)
            this.aantalDeuren = aantalDeuren;
    }

    public int getAantalPassagiers() {
        return aantalPassagiers;
    }
    public final void setAantalPassagiers(int aantalPassagiers) {
        if (aantalPassagiers > 0)
            this.aantalPassagiers = aantalPassagiers;
    }

    @Override
    public String toString() {
        return super.toString() + ";" +
               aantalDeuren + ";" + aantalPassagiers ;    }
```

```
    @Override
    public void toon() {
        System.out.println("\nPERSONENWAGEN");
        super.toon();
        System.out.println("deuren: " + aantalDeuren + "\n" +
                           "passagiers: " + aantalPassagiers);
    }

    public double getKyotoScore() {
        return getGemVerbruik() * getPk() / aantalPassagiers;
    }
}
```

Het hoofdprogramma:

```
public class TestProgramma {

      public static void main(String[] args) {
        Voertuig opel1 = new Personenwagen();
        opel1.toon();
        System.out.println(opel1);

        Vrachtwagen volvo1 = new Vrachtwagen();
        volvo1.toon();
        System.out.println(volvo1);

        Personenwagen opel2 = new Personenwagen("Jan Klaasen",
                14599.0F, 105, 6.8F, "KLM099", 5, 5);
        opel2.toon();
        System.out.println(opel2);

        opel2.setKostprijs(-15000);
        opel2.setAantalDeuren(-7);
        opel2.setAantalPassagiers(0);
        System.out.println(opel2); //opel2 is niet gewijzigd

        Vrachtwagen volvo2 = new Vrachtwagen("Michel Dewolf",
                214599.0F, 440, 33.1F, "PRD441", 6000.0F);
        volvo2.toon();
        System.out.println(volvo2);

        System.out.println();
        System.out.println("Kyotoscore personenwagen 1: " +
           opel1.getKyotoScore());
        System.out.println("Kyotoscore personenwagen 2: " +
           opel2.getKyotoScore());
```

```
        System.out.println("Kyotoscore vrachtwagen 1: " +
           volvo1.getKyotoScore());
        System.out.println("Kyotoscore vrachtwagen 2: " +
           volvo2.getKyotoScore());

    }
}
```

Oplossing oefening 16

- De lijn p3.extra() geeft een compile-error. P3 is een punt-reference dat wijst naar een gekleurdpuntmetdikte maar heeft geen method extra(). Het volstaat een lege method extra() toe te voegen aan de class Punt om de polymorphism te laten werken.
- De lijn d1 = p1; geeft een compile-error. Omgekeerd zou wel kunnen (p1 = d1).
- Wanneer we de coderegels die compileerfouten geven weglaten krijgen we als uitvoer 3 keer de uitvoer van schrijf() uit een gekleurdpuntmetdikte en vervolgens twee keer de uitvoer van een melding.

Oplossingen oef. bij hoofdstuk 7: Strings

Oplossing oefening 17

```
import java.util.*;

public class Main {

    private static final String KLINKERS = "aeiou";
    public static void main(String[] args) {

    Scanner scan = new Scanner(System.in);
    String zin = scan.nextLine().toLowerCase();

    int teller = 0;
    for (int i=0; i != zin.length();i++) {
      if (klinkers.indexOf(zin.charAt(i)) >= 0) teller++;
    }

    System.out.println("Er zitten " + teller +
                       " klinkers in de zin:");
    System.out.println(zin);
  }
}
```

Oplossing oefening 18

```
import java.util.Scanner;


public class Main {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        String bewerking = scan.nextLine();
        String[] delen = bewerking.split(" ");
        //in geval van een lege expressie bevat de array delen 1
        //element, nl. een lege string: ""
        int getal = 0;
        // eerste cijfer in uitkomst zetten om te sommeren
        int uitkomst;
        if (!delen[0].isEmpty()) {
            uitkomst = Integer.parseInt(delen[0]);
```

```
        } else {
            uitkomst = 0;
        }
        int i = 1;
        while (i != delen.length) {
            char bewTeken = delen[i++].charAt(0);
            if (i != delen.length) {
                getal = Integer.parseInt(delen[i]);
                switch (bewTeken) {
                    case '+':
                        uitkomst += getal;
                        break;
                    case '-':
                        uitkomst -= getal;
                        break;
                    case '*':
                        uitkomst *= getal;
                        break;
                    case '/':
                        if (getal != 0) {
                            uitkomst /= getal;
                        }
                }
                i++;
            }
        }
        System.out.println(bewerking + " = " + uitkomst);
    }
}
```

Oplossing oefening 19

```
public class PalindroomTester {

    public boolean isPalindroom(String tekst,
            boolean hoofdlettergevoelig) {
        String omgekeerd =
```

```
          new StringBuffer(tekst).reverse().toString();
        if (hoofdlettergevoelig) {
            return tekst.equals(omgekeerd);
        } else {
            return tekst.equalsIgnoreCase(omgekeerd);
        }
    }
}
```

```
import java.util.*;
public class Main {
  public static void main(String[] args) {
    PalindroomTester palin = new PalindroomTester();
    Scanner scan = new Scanner(System.in);
    String tekst = scan.next();

    if (palin.isPalindroom(tekst, true)) {
      System.out.println("Het woord is een zuiver palindroom.");
    }
    else {
      if (palin.isPalindroom(tekst, false)) {
        System.out.println("Het woord is, los van de hoofdletters");
        System.out.println(", een palindroom");
      }
      else {
        System.out.println("Het woord is geen palindroom !");
      }
    }
  }
}
```

Oplossingen oef. bij hoofdstuk 8: Interfaces

Oplossing oefening 20

```
public interface Vervuiler {
    public double geefVervuiling();
}
```

```
public class Vrachtwagen extends Voertuig implements Vervuiler {

…
    public double geefVervuiling() {
        return getKyotoScore() * 20.0F;
    }
}
```

```
public class Personenwagen extends Voertuig implements Vervuiler {

…
    public double geefVervuiling() {
        return getKyotoScore() * 5.0F;
    }
}
```

```
public class Stookketel implements Vervuiler {
    private float CONorm;

    public Stookketel() {
    }
    public Stookketel(float CONorm) {
    setCONorm(CONorm);
    }

    public float getCONorm() {
        return this.CONorm;
    }
    public final void setCONorm(float CONorm) {
        this.CONorm = CONorm;
    }
```

```
    @Override
    public double geefVervuiling() {
        return getCONorm() * 100;
    }
}
```

```
public static void main(String[] args) {
      …
      System.out.println("\n*** Array van vervuilers ***");
      Vervuiler[] vervuilers = new Vervuiler[3];
      vervuilers[0] = opel2;
      vervuilers[1] = volvo2;
      vervuilers[2] = new Stookketel(7.5F);
      for (Vervuiler vervuiler:vervuilers) {
          System.out.println("vuil: "+ vervuiler.geefVervuiling() );
      }
}
```

Oplossing oefening 21

```
public interface Privaat {
    public void geefPrivateData();
}
```

```
public interface Milieu {
    public void geefMilieuData();
}
```

```
public abstract class Voertuig implements Milieu, Privaat {
   …

   public void geefPrivateData() {
       System.out.println("Polishouder :" + getPolishouder());
       System.out.println("Nummerplaat :" + getNummerplaat());
   }

      public void geefMilieuData() {
       System.out.println("PK :" + getPk());
       System.out.println("Kostprijs :" + getKostprijs());
       System.out.println("Gem. verbruik :" + getGemVerbruik());
   }
}
```

```
   public static void main(String[] args) {
        …
        Privaat auto[] = new Privaat[] {opel1, opel2,
                                        volvo1, volvo2};
        for (Privaat eenAuto:auto) {
           eenAuto.geefPrivateData();
        }

        Milieu auto2[] = new Milieu[] {opel1, opel2,
                                       volvo1, volvo2};
        for (Milieu eenAuto:auto2) {
           eenAuto.geefMilieuData();
        }
    }
```

Oplossing oefening 22

```
public interface Voorwerp {
    void gegevensTonen();
    double winstBerekenen();
}
```

```
public class BoekenRek implements Voorwerp {
    public BoekenRek() {
        setHoogte(0.0F);
        setBreedte(0.0F);
        setAankoopPrijs(0.0);
    }
    public BoekenRek(float hoogte, float breedte,
                     double aankoopPrijs) {
        setHoogte(hoogte);
        setBreedte(breedte);
        setAankoopPrijs(aankoopPrijs);
    }
    private float hoogte;
    public float getHoogte() { return this.hoogte; }
    public void setHoogte(float hoogte) { this.hoogte = hoogte; }

    private float breedte;
    public float getBreedte() { return this.breedte; }
    public void setBreedte(float breedte) { this.breedte = breedte;}

    private double aankoopPrijs;
    public double getAankoopPrijs() { return this.aankoopPrijs; }
    public void setAankoopPrijs(double aankoopPrijs) {
        this.aankoopPrijs = aankoopPrijs;
    }

    @Override
    public double winstBerekenen() {
        return this.getAankoopPrijs() * 2.0;
    }

    @Override
    public void gegevensTonen() {
        System.out.println(this.toString());
    }

    @Override
    public String toString(){
        return hoogte + " ; " + breedte + " ; " +
               aankooppprijs ;
    }
}
```

```
public abstract class Boek implements Voorwerp {
    public Boek() {
        setTitel("geen titel");
        setAuteur("geen auteur");
        setAankoopPrijs(0.0);
        setGenre("geen genre");
    }

    public Boek(String titel, String auteur, double aankoopPrijs,
                String genre) {
        setTitel(titel);
        setAuteur(auteur);
        setAankoopPrijs(aankoopPrijs);
        setGenre(genre);
    }

    private String titel;
    public String getTitel() { return titel; }
    public void setTitel(String titel) { this.titel = titel; }

    private String auteur;
    public String getAuteur() { return auteur; }
    public void setAuteur(String auteur) { this.auteur = auteur; }

    private static String eigenaar = "VDAB";
    public String getEigenaar() {
        return this.eigenaar;
    }

    private double aankoopPrijs;
    public double getAankoopPrijs() {
        return this.aankoopPrijs;
    }
    public void setAankoopPrijs(double aankoopPrijs) {
        this.aankoopPrijs = aankoopPrijs;
    }

    private String genre;
    public String getGenre() { return genre; }
    public void setGenre(String genre) { this.genre = genre; }


    @Override
    public void gegevensTonen() {
        System.out.println(this.toString());
    }

    @Override
    public String toString() {
        return titel + ", " + auteur + ", " + aankoopprijs + ", " +
               genre + ", " + eigenaar;
    }
}
```

```
public class LeesBoek extends Boek {

    public LeesBoek() {
        super();
        setOnderwerp("geen onderwerp");
    }
    public LeesBoek(String titel, String auteur,
        double aankoopPrijs, String genre,String onderwerp) {
        super(titel, auteur, aankoopPrijs, genre);
        setOnderwerp(onderwerp);
    }
    private String onderwerp;

    public String getOnderwerp() { return onderwerp; }

    public void setOnderwerp(String onderwerp) {
        this.onderwerp = onderwerp;
    }

    @Override
    public double winstBerekenen() {
        return this.getAankoopPrijs() * 1.5;
    }

    @Override
    public String toString() {
        return super.toString() + ", " + onderwerp;
    }
}
```

```
public class WoordenBoek extends Boek {

    public WoordenBoek() {
        super();
        setTaal("geen taal");
    }
    public WoordenBoek(String titel, String auteur,
           double aankoopPrijs, String genre,String taal) {
        super(titel, auteur, aankoopPrijs, genre);
        setTaal(taal);
    }
    private String taal;

    public String getTaal() { return taal; }

    public void setTaal(String taal) { this.taal = taal; }


    @Override
    public double winstBerekenen() {
        return this.getAankoopPrijs() * 1.75;
    }
    @Override
    public String toString() {
        return super.toString() + ", " + taal;
    }
}
```

```
  public static void main(String[] args) {

      Voorwerp voorwerpen[] = new Voorwerp[6];
      voorwerpen[0] = new LeesBoek();
      voorwerpen[1] = new LeesBoek("Java by example",
                      "J.P. Dupont",18.85,"Informatica","Java");
      voorwerpen[2] = new WoordenBoek();
      voorwerpen[3] = new WoordenBoek("Wolters N-F",
            "A.D.P. Vandervoort", 25.50, "Naslagwerken", "N-F");
      voorwerpen[4] = new BoekenRek();
      voorwerpen[5] = new BoekenRek(200.0F, 75.5F, 75.0);

      double totaleWinst = 0.0;
      for (Voorwerp eenVoorwerp:voorwerpen) {
          totaleWinst += eenVoorwerp.winstBerekenen();
          eenVoorwerp.gegevensTonen();
      }

      System.out.println("Totale winst : " + totaleWinst);
  }
```

Oplossingen oef. bij hoofdstuk 9: Packages

Oplossing oefening 23

```
package be.vdab.util;
public interface Voorwerp {
    void gegevensTonen();
    double winstBerekenen();
}
```

```
package be.vdab.voorwerpen;
import be.vdab.util.Voorwerp;
public class BoekenRek implements Voorwerp {
   …
}
```

```
package be.vdab.voorwerpen;
public abstract class Boek implements Voorwerp {
   …
}
```

```
package be.vdab.voorwerpen;
import be.vdab.util.Voorwerp;
public class LeesBoek extends Boek {
   …
}
```

```
package be.vdab.voorwerpen;
import be.vdab.util.Voorwerp;
public class WoordenBoek extends Boek {
   …
}
```

```
package be.vdab;
import be.vdab.util.Voorwerp;
import be.vdab.voorwerpen.LeesBoek;
import be.vdab.voorwerpen.WoordenBoek;
import be.vdab.voorwerpen.BoekenRek;
public static void main(String[] args) {
   …
}
```

Oplossingen oef. bij hoofdstuk 10: Exception handling

Oplossing oefening 24

Resultaat met “no” :

```
start try
start risky
end risky
end try
finally
end of main
```

Resultaat met “yes” :

```
start try
start risky
scary exception
finally
end of main
```

Oplossing oefening 25

```
public class RekeningException extends java.lang.Exception {

    private String rekeningNummer;

    public RekeningException() {
    }

    public RekeningException(String msg) {
        super(msg);
    }

    public RekeningException(String msg, String nummer) {
        super(msg);
        this.rekeningNummer = nummer;
    }

    public String getVerkeerdNummer() {
        return " => " + rekeningNummer;
    }
}
```

```
public abstract class Rekening {
    …
    public Rekening(String rnr) throws RekeningException {
        isRekNrOk(rnr);
        rekeningNr = rnr;
        saldo = 0;
    }

    …
    private void isRekNrOk (String reknr) throws RekeningException {
      if (reknr.length()==14) {
        int d1=Integer.parseInt(reknr.substring(0,3));
        int d2=Integer.parseInt(reknr.substring(4,11));
        int d3=Integer.parseInt(reknr.substring(12,14));

        long deeltal=d1 * 10000000L + d2;
        int rest=(int)(deeltal % 97);
        if (rest==0) rest=97;

        if (!(rest==d3))
            throw new RekeningException(
                "Het rekeningnummer is fout !!", reknr);
      }
      else {
          throw new RekeningException(
              "Het rekeningnummer is fout (te kort) !!", reknr);
      }
    }
}
```

```
public class SpaarRekening extends Rekening {

    public SpaarRekening(String rnr, double intrest) throws
      RekeningException {
        super(rnr);
        this.intrest = intrest;
    }
    …
 }
```

```
public class ZichtRekening extends Rekening {
    …
    public ZichtRekening(String rnr,  int bedrag) throws
      RekeningException {
        super(rnr);
        maxKrediet = bedrag;
    }
    …
 }
```

```
public static void main(String[] args) {
    DecimalFormat fmt = new DecimalFormat("#,##0.00");
    Rekening[] rekeningen = new Rekening[3];
    try {
      rekeningen [0] = new SpaarRekening ("035-0621094-44", 2.5);
      rekeningen [1] = new ZichtRekening ("784-5879305-64", 1000);
      rekeningen [2] = new SpaarRekening ("001-3456789-01", 5.5);

      rekeningen[0].storten(100.0);
      rekeningen[1].storten(200.0);
      rekeningen[1].afhalen(50.0);
    }
    catch (RekeningException re) {
      System.out.println(re.getMessage() + re.getVerkeerdNummer() );
    }

    for (int i = 0 ; i != rekeningen.length; i++) {
      if (rekeningen[i] != null)
          System.out.println(rekeningen[i].toString());
    }
}
```

Oplossingen oef. bij hoofdstuk 12: Collections

Oplossing oefening 26 :

```
public class Land {

    public Land(String landcode, String landnaam, String hoofdstad,
                long inwoners, float oppervlakte) {
        setLandCode(landcode);
        setLandNaam(landnaam);
        setHoofdstad(hoofdstad);
        setInwoners(inwoners);
        setOppervlakte(oppervlakte);
    }

    private String landCode;
    public String getLandCode() {
        return landCode;
    }
    public void setLandCode(String landCode) {
        this.landCode = landCode;
    }

    private String landNaam;
    public String getLandNaam() {
        return landNaam;
    }
    public void setLandNaam(String landNaam) {
        this.landNaam = landNaam;
    }

    private String hoofdstad;
    public String getHoofdstad() {
        return hoofdstad;
    }
    public void setHoofdstad(String hoofdstad) {
        this.hoofdstad = hoofdstad;
    }

    private long inwoners;
    public long getInwoners() {
        return inwoners;
    }
    public void setInwoners(long inwoners) {
        if (inwoners >= 0)
           this.inwoners = inwoners;
    }
```

```
    private float oppervlakte = 1.0F;

    public float getOppervlakte() {
        return oppervlakte;
    }

    public void setOppervlakte(float oppervlakte) {
        if (oppervlakte >0)
           this.oppervlakte = oppervlakte;
    }

    public float getBevolkingsDichtheid() {
        return getInwoners() / getOppervlakte();
    }
}
```

```
import java.util.*;

public class Landen {
  public static void main(String[] args) {
    List deLanden = new ArrayList(10);
    deLanden.add(new Land("B","Belgie","Brussel",10360000,30518));
    deLanden.add(new Land("D","Duitsland","Berlijn",82430000,
                          356974));
    deLanden.add(new Land("F","Frankrijk","Parijs",60660000,
                          547030));
    deLanden.add(new Land("I","Italie","Rome",58100000,301287));
    deLanden.add(new Land("L","Luxemburg","Luxemburg",470000,2586));
    deLanden.add(new Land("NL","Nederland","Amsterdam",16360000,
                          41528));
    deLanden.add(new Land("P","Portugal","Lissabon",10600000,
                          92082));
    deLanden.add(new Land("E","Spanje","Madrid",40340000,504750));
    deLanden.add(new Land("GB","Verenigd Koninkrijk","Londen",
                          60610000, 244820));
    deLanden.add(new Land("CH","Zwitserland","Bern",7490000,41286));

    //berekenen gemiddelde bevolkingsdichtheid
    double dichtheid = .0;
    for (Object obj:deLanden) {
      Land eenLand = (Land)obj;
      System.out.print(eenLand.getLandNaam());
      dichtheid += eenLand.getBevolkingsDichtheid();
      System.out.println(" : " +
          String.format("%.2f",eenLand.getBevolkingsDichtheid())
          + " inw./km²");
    }
```

```
    dichtheid /= deLanden.size();
    System.out.println("De gemiddelde bevolkingsdichtheid is : " +
                  String.format("%.2f",dichtheid) + " inw./km²");

    //zoeken land dichtst bij gemiddelde bevolkingsdichtheid
    int gemiddelde = 0;
    int teller = 0;
    double kleinsteVerschil = Double.MAX_VALUE;
    for (Object obj : deLanden) {
      Land eenLand = (Land) obj;
      double verschil = 0;
      teller++;
      if (eenLand.getBevolkingsDichtheid() > dichtheid)
        verschil = eenLand.getBevolkingsDichtheid() - dichtheid;
      else
        verschil = dichtheid - eenLand.getBevolkingsDichtheid();
      if (verschil < kleinsteVerschil) {
        kleinsteVerschil = verschil;
        gemiddelde = teller;
      }
    }
    System.out.println(((Land)deLanden.get(gemiddelde-1))
      .getLandNaam() + " (" + String.format("%.2f",
      ((Land)deLanden.get(gemiddelde-1)).getBevolkingsDichtheid()) +
      " inw./km²) leunt het dichtst aan bij het gemiddelde.");
  }
}
```

Oplossing oefening 27 :

```
  public static void main(String[] args) {
        …
        Set<Voertuig> lijst = new TreeSet<Voertuig>();
        lijst.add(opel2);
        lijst.add(volvo1);
        lijst.add(volvo2);

        for (Object obj : lijst) {
            Voertuig eenVoertuig = (Voertuig) obj;
            System.out.println(eenVoertuig);
        }
  }
```

Source van Voertuig :

```
public abstract class Voertuig implements Milieu, Privaat,
  Comparable {
   …
   public int compareTo(Object obj) {
     if (obj == null)
        throw new NullPointereException();

     return this.getNummerplaat().compareTo(
       ((Voertuig)obj).getNummerplaat());
   }
}
```

Oplossing oefening 28 :

```
 public static void main(String[] args) {
   String woorden[] = {"Banaan", "Perzik", "Kiwi", "Appel",
                       "Okkernoot", "Kriek", "Peer"};

   HashMap<Character,Integer> letters = new HashMap();
   int huidigAantal = 0;
   Integer resultaat;
   Character beginletter;

   for (String eenWoord:woorden) {
     beginletter = (Character) (eenWoord.charAt(0));
     if ((resultaat = letters.get(beginletter)) != null) {
         huidigAantal = resultaat.intValue();
         letters.put(beginletter, huidigAantal + 1);
     }
     else letters.put(beginletter, 1);
   }
   Set lettersKeys = letters.keySet();
   Iterator It = lettersKeys.iterator();
   while (It.hasNext()) {
     Character eenLetter = (Character)(It.next());
     System.out.println(eenLetter + " - " + letters.get(eenLetter));
   }
   System.out.println("De hashmap bevat " + letters.size() +
                      " elementen.");
 }
```

Oplossing oefening 29 :

```
package be.vdab.winkel;

import java.math.BigDecimal;

public class Product {
    private String omschrijving;
    private BigDecimal prijs;

    public Product() { }

    public Product(String oms, BigDecimal pr ) {
        setOmschrijving(oms);
        setPrijs(pr);
    }

    public String getOmschrijving() {
       return omschrijving;
    }

    public void setOmschrijving(String omschrijving) {
       this.omschrijving = omschrijving;
    }

    public BigDecimal getPrijs() {
       return prijs;
    }
    public void setPrijs(BigDecimal prijs) {
       this.prijs = prijs;
    }

    public boolean equals(Object object) {
        if (! (object instanceof Product)) {
              return false;
        }
        Product andere = (Product) object;
        return this.omschrijving.equals(andere.omschrijving);
    }
    public int hashCode() {
        return omschrijving.hashCode();
    }

    @Override
    public String toString() {
        return String.format ("%40s %.2f ",  omschrijving, prijs);
    }
}
```

```
package be.vdab.winkel;

import java.math.BigDecimal;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

public class Catalogus {
  private List <Product> lijst = new ArrayList<Product> (10);

  /* Opm.: List: nu kan 2x hetzelfde product in de catalogus zitten
  * --> wat wil je ermee kunnen doen, wat is bepaald in de analyse?
  */

  public Catalogus() {
    lijst.add(new Product("appelen", new BigDecimal("0.34") ) );
    lijst.add(new Product("peren", new BigDecimal("0.56")));
    lijst.add(new Product("citroenen", new BigDecimal("0.64" ) ) );
    lijst.add(new Product("aardbeien", new BigDecimal("2.85" ) ) );
    lijst.add(new Product("mandarijnen", new BigDecimal("1.60" ) ));
    lijst.add(new Product("noten", new BigDecimal("2.35" ) ) ) ;
    lijst.add(new Product("kastanjes", new BigDecimal("2.15" ) ) ) ;
    lijst.add(new Product("rozijnen", new BigDecimal("1.90" ) ) ) ;
  }

  public Iterator <Product> iterator() {
    return lijst.iterator();
  }
}
```

```
package be.vdab.winkel;

import java.math.BigDecimal;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;

public class Mandje {
    public Mandje() { }

    private Map <Product,Integer> mandje =
      new HashMap <Product,Integer> (20);

    public void add(Product product, int aantal) {
        if (! mandje.containsKey(product) )
            mandje.put(product, aantal);
        else
```

```
            set(product, aantal);
    }

    public void set(Product product, int aantal) {
        int oudeAantal=mandje.get(product);
        mandje.put(product,oudeAantal + aantal);
    }

    public void remove(Product product) {
        mandje.remove(product);
    }

    public void clear() {
        mandje.clear();
    }

    public BigDecimal getSom() {
        BigDecimal som = BigDecimal.ZERO;
        for (Map.Entry<Product,Integer> entry : mandje.entrySet()) {
            som = som.add(entry.getKey().getPrijs().multiply( new
                  BigDecimal (entry.getValue() ) ) );
        }
        return som;
    }

    /* iterator --> er zijn 3 zichten : alleen de keys, alleen de
     * values of beiden !
     * Dus hier een iterator van Map.Entry objecten
     */

    public Iterator <Map.Entry <Product,Integer>> iterator()
    {
        return mandje.entrySet().iterator();
    }
    // een entry is een interne class van Map
}
```

```
package be.vdab.winkel;

import java.math.BigDecimal;
import java.util.Iterator;
import java.util.Map;

public class TestProgramma {
    public static void main(String[] args) {
        Catalogus catalogus = new Catalogus();
          //wordt meteen gevuld door de constructor
```

```
        Iterator <Product> pi = catalogus.iterator();
        int i=1;

        Mandje mandje = new Mandje ();

        //mandje programmatorisch vullen
        while(pi.hasNext() ) {
            Product p = pi.next();
            System.out.println(p);
            i++;
            if (i%2==0)
                mandje.add(p,i);
        }

        System.out.println("u kocht:");
        Iterator <Map.Entry <Product,Integer>> mi =
          mandje.iterator();
        while(mi.hasNext() )
        {
            Map.Entry<Product,Integer> aankoop = mi.next();
            System.out.printf("%s %d \n",aankoop.getKey(),
              aankoop.getValue());
        }
        System.out.printf("u kocht voor een totaal van: %.2f \n",
          mandje.getSom() );
    }
}
```

Oplossing oefening 30 :

```
public class Tienkamper implements Comparable<Tienkamper> {

    private String naam;
    private int punten;

    public Tienkamper(String naam, int punten) {
        setNaam(naam);
        setPunten(punten);
    }

    public String getNaam() {
        return naam;
    }
    public void setNaam(String naam) {
        this.naam = naam;
    }
    public int getPunten() {
        return punten;
    }
    public void setPunten(int punten) {
        this.punten = punten;
    }

    @Override
    public String toString() {
        return naam + " - " + punten;
    }
    @Override
    public boolean equals(Object object) {
        if (! (object instanceof Tienkamper)) {
          return false;
        }
        Tienkamper andere = (Tienkamper) object ;
        return this.getNaam().equals(andere.getNaam());
    }
    @Override
    public int compareTo(Tienkamper object) {
        return this.getNaam().compareTo(object.getNaam());
    }

    @Override
    public int hashCode() {
        return this.getNaam().hashCode();
    }
}
```

```
import java.util.Comparator;

public class AltComparator implements Comparator<Tienkamper> {
    @Override
    public int compare(Tienkamper obj1, Tienkamper obj2) {
        if (obj1.getPunten() == obj2.getPunten())
            return 0;
            //return 1; wanneer wel dubbele punten zijn toegestaan
        else
            if (obj1.getPunten() > obj2.getPunten())
                return 1;
            else
                return -1;
    }
}
```

```
public static void main(String[] args) {
  Tienkamper Jürgen = new Tienkamper("Jürgen Hingsen",8832);
  Tienkamper Roman  = new Tienkamper("Roman Šebrle",8891);
  Tienkamper Daley  = new Tienkamper("Daley Thompson",8847);
  Tienkamper Dan    = new Tienkamper("Dan O'Brien",8891);

  List<Tienkamper> lijst = new ArrayList<Tienkamper>(4);
  lijst.add(Jürgen);
  lijst.add(Roman);
  lijst.add(Daley);
  lijst.add(Dan);

  System.out.println("Alle tienkampers uit de arraylist " +
                     "(=volgorde van toevoegen):");
  for (Tienkamper eenTienkamper:lijst)
     System.out.println(eenTienkamper);
  System.out.println();

  Set<Tienkamper> set = new TreeSet<Tienkamper>();
  set.add(Jürgen);
  set.add(Roman);
  set.add(Daley);
  set.add(Dan);

  System.out.println("Alle tienkampers uit de treeset " +
                     "(=gesorteerd op naam):");
  for (Tienkamper eenTienkamper:set)
     System.out.println(eenTienkamper);
  System.out.println();

  //de arraylist sorteren op punten
  Comparator comp = new AltComparator();
  Collections.sort(lijst, comp);

  System.out.println("Alle tienkampers uit de arraylist, " +
                     "gesorteerd op punten:");
  for (Tienkamper eenTienkamper:lijst)
    System.out.println(eenTienkamper);
  System.out.println();

  Set<Tienkamper> setOpPunten =
           new TreeSet<Tienkamper>(new AltComparator());
  setOpPunten.add(Jürgen);
  setOpPunten.add(Roman);
  setOpPunten.add(Daley);
  setOpPunten.add(Dan);
```

```
  System.out.println("Alle tienkampers uit de treeset, " +
                     "gesorteerd op punten:");
  for (Tienkamper eenTienkamper:setOpPunten)
     System.out.println(eenTienkamper);
  System.out.println();
}
```

Oplossingen oef. bij hoofdstuk 13: Streams

Oplossing oefening 31 :

```
package be.vdab;
import java.io.Serializable;
import java.util.Date;
public class GastenBoekEntry implements Serializable {
    private static final long serialVersionUID = 1L;
    private Date datum;
    private String schrijver;
    private String boodschap;
    public GastenBoekEntry(String schrijver, String boodschap) {
        this.datum = new Date();
        this.schrijver = schrijver;
        this.boodschap = boodschap;
    }
    public Date getDatum() {
        return datum;
    }
    public String getSchrijver() {
        return schrijver;
    }
    public String getBoodschap() {
        return boodschap;
    }
}
```

```
package be.vdab;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.List;
public class GastenBoek implements Serializable {
    private static final long serialVersionUID = 1L;
    List<GastenBoekEntry> entries =
      new ArrayList<GastenBoekEntry>();
```

```
    public List<GastenBoekEntry> getEntries() {
        return entries;
    }
    public void addEntry(GastenBoekEntry entry) {
        entries.add(entry);
    }
}
```

```
package be.vdab;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
public class GastenBoekManager {
    private static final String GASTENBOEK_PATH =
"/jpf/gastenboek.dat";
    public void save(GastenBoek gastenboek) {
        try {
            FileOutputStream fileOutputStream = new
FileOutputStream(GASTENBOEK_PATH);
            ObjectOutputStream objectOutputStream = new
ObjectOutputStream(fileOutputStream);
            objectOutputStream.writeObject(gastenboek);
        } catch (Exception ex) {
            System.out.println(ex);
        }
    }
    public GastenBoek load() {
        File file = new File(GASTENBOEK_PATH);
        if (file.exists()) {
            try {
                FileInputStream fileInputStream = new
FileInputStream(GASTENBOEK_PATH);
                ObjectInputStream objectInputStream = new
ObjectInputStream(fileInputStream);
                return (GastenBoek) objectInputStream.readObject();
            } catch (Exception ex) {
                System.out.println(ex);
```

```
                return null;
            }
        } else {
            return new GastenBoek();
        }
    }
}
```

```
package be.vdab;
import java.util.Collections;
import java.util.List;
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
         GastenBoekManager gastenBoekManager =
           new GastenBoekManager();
        GastenBoek gastenBoek;
        Scanner scanner = new Scanner(System.in);
        System.out.print("1=lezen, 2=toevoegen:");
        switch (scanner.nextInt()) {
            case 1:
                gastenBoek = gastenBoekManager.load();
                List<GastenBoekEntry> entries =
                  gastenBoek.getEntries();
                Collections.reverse(entries);
                for (GastenBoekEntry entry : entries) {
                    System.out.println(entry.getDatum() +
                     ":"+entry.getSchrijver() + ":" +
                     entry.getBoodschap());
                }
                break;
            case 2:
                scanner.nextLine();
                System.out.print("Schrijver:");
                String schrijver = scanner.nextLine();
                System.out.print("Boodschap:");
                String boodschap = scanner.nextLine();
```

```
                GastenBoekEntry entry = new
GastenBoekEntry(schrijver, boodschap);
                gastenBoek = gastenBoekManager.load();
                gastenBoek.addEntry(entry);
                gastenBoekManager.save(gastenBoek);
                break;
            case 3:
                System.out.println("Verkeerde keuze");
        }
    }
}
```

Oplossingen oef. bij hoofdstuk 14: Multithreading

Oplossing oefening 32 :

```
package be.vdab;

public class GemiddeldeRekenaar implements Runnable {
  private double[] getallen;
  private int vanafIndex;
  private int totIndex;
  private double gemiddelde;
  public GemiddeldeRekenaar(double[] getallen, int vanafIndex,
    int totIndex) {
    this.vanafIndex = vanafIndex;
    this.totIndex = totIndex;
    this.getallen = getallen;
  }
  @Override
  public void run() {
    double totaal = 0;
    for (int i = vanafIndex; i != totIndex; i++) {
      totaal += getallen[i];
    }
    gemiddelde = totaal / (totIndex - vanafIndex + 1);
}
  public double getGemiddelde() {
    return gemiddelde;
  }
}
```

```
package be.vdab;

public class Main {
  public static void main(String[] args) {
    double getallen[] = new double[1000000];
    for (int i = 0; i != getallen.length; i++) {
      getallen[i] = Math.random();
    }
    GemiddeldeRekenaar gemiddeldeRekenaar1 =
      new GemiddeldeRekenaar(getallen, 0, getallen.length/2-1);
    GemiddeldeRekenaar gemiddeldeRekenaar2 =
    new GemiddeldeRekenaar(getallen, getallen.length / 2,
      getallen.length);
    Thread thread1 = new Thread(gemiddeldeRekenaar1);
    Thread thread2 = new Thread(gemiddeldeRekenaar2);
    thread1.start();
    thread2.start();
    try {
      thread1.join();
      thread2.join();
    } catch (InterruptedException ex) {
      System.err.println(ex);
    }
```

```
    System.out.println((gemiddeldeRekenaar1.getGemiddelde() +
      gemiddeldeRekenaar2.getGemiddelde()) / 2);
  }
}
```

Oplossing oefening 33 :

De gewijzigde class Stapel

```
package be.vdab;

public class Stapel {
  private int aantalPannenKoeken;
  synchronized public void voegPannenkoekToe() {
    ++aantalPannenKoeken;
  }
  synchronized public boolean neemPannenkoekWeg() {
    if (aantalPannenKoeken > 0) {
      --aantalPannenKoeken;
      return true;
    }
      return false;
    }
  public int getAantalPannenkoeken() {
    return aantalPannenKoeken;
  }
}
```

De nieuwe class Klant

```
package be.vdab;

public class Klant implements Runnable {
  private Stapel stapel;
  public Klant(Stapel stapel) {
    this.stapel = stapel;
  }
  @Override
  public void run() {
    int aantalPannenKoekenGegeten = 0;
    while (aantalPannenKoekenGegeten != 50) {
      if (stapel.neemPannenkoekWeg()) {
        aantalPannenKoekenGegeten++;
      }
      try {
        Thread.sleep(100);
      } catch (InterruptedException ex) {
        System.err.println(ex);
      }
    }
  }
}
```

De gewijzigde class Main

```
package be.vdab;

import java.util.ArrayList;
import java.util.List;

public class Main {

  public static void main(String[] args) {
    Stapel stapel = new Stapel();
    List<Thread> threads = new ArrayList<Thread>();
    for (int i = 0; i != 2; i++) {
      threads.add(new Thread(new Kok(stapel)));
    }
    for (int i = 0; i != 4; i++) {
      threads.add(new Thread(new Klant(stapel)));
    }
    for (Thread thread : threads) {
      thread.start();
    }
    try {
      for (Thread thread : threads) {
        thread.join();
      }
    } catch (InterruptedException ex) {
      System.err.println(ex);
    }
    System.out.println(stapel.getAantalPannenkoeken());
  }
}
```

Oplossingen oef. bij hoofdstuk 15: GUI

Oplossing oefening 34 :

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class TempConversie extends JFrame implements ActionListener
{
  private JPanel panel1 = new JPanel();
  private JPanel panel2 = new JPanel();

  private JTextField textTempCelsius;
  private JLabel labelCelsius;
  private JButton knopConverteer;
  private JLabel labelFahrenheit;

  public TempConversie() {
    setSize(350,150);
    setTitle("Temperatuur omzetten C - F");
  }

  public static void main(String[] args) {
    TempConversie frame = new TempConversie();
    frame.createGUI();
    frame.setVisible(true);
  }

  private void createGUI() {
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setLayout(new FlowLayout());
    labelCelsius    = new JLabel("Celsius");
    textTempCelsius = new JTextField(12);
    knopConverteer  = new JButton ("Converteer");
    labelFahrenheit = new JLabel("Fahrenheit");

    panel1.add(textTempCelsius);
    panel1.add(labelCelsius);
    panel2.add(knopConverteer);
    panel2.add(labelFahrenheit);
    add(panel1);
    add(panel2);
    knopConverteer.addActionListener(this);
  }

  @Override
  public void actionPerformed(ActionEvent e) {
    int tempFahr = (int)(
     (Double.parseDouble(textTempCelsius.getText()) ) * 1.8 + 32);
    labelFahrenheit.setText(tempFahr + " Fahrenheit");
  }
}
```

Oplossing oefening 35 :

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class DrieButtons extends JFrame implements ActionListener
{
  private JButton buttonLinks ;
  private JButton buttonMidden ;
  private JButton buttonRechts ;

  public static void main (String []args) {
    DrieButtons frame = new DrieButtons();
    frame.setSize(600,200);
    frame.createGUI();
    frame.setVisible(true);
    frame.setTitle("Spelen met 3 buttons");
  }

  private void createGUI() {
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setLayout(new FlowLayout() );
    JPanel panel = new JPanel();
    buttonLinks = new JButton("Disable knop -->");
    buttonRechts = new JButton("<--Enable knop");
    buttonRechts.setEnabled(false);
    ImageIcon icon = new ImageIcon("C:\\gelukkig.gif");
    buttonMidden = new JButton("Middelste knop", icon);
    panel.add(buttonLinks);
    panel.add(buttonMidden);
    panel.add(buttonRechts);
    add(panel);
    buttonLinks.addActionListener(this);
    buttonMidden.addActionListener(this);
    buttonRechts.addActionListener(this);
  }

  @Override
  public void actionPerformed(ActionEvent e) {
    if (e.getSource()== buttonLinks) {
      buttonMidden.setEnabled(false);
      buttonLinks.setEnabled(false);
      buttonRechts.setEnabled(true);
    }
    else
      if (e.getSource()== buttonRechts) {
        buttonMidden.setEnabled(true);
        buttonRechts.setEnabled(false);
        buttonLinks.setEnabled(true);
      }
      else
        JOptionPane.showMessageDialog(null,
          "U koos voor de middelste knop");
    }
}
```

Oplossing oefening 36 :

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;

public class GUIElementenSwingOef extends Jframe
                                  implements ActionListener {
  private JButton buttonOK;
  private JComboBox cboMaanden;
  private JTextField txtJaar, txtNaam, txtUren ;
  private JRadioButton radioButtonArb, radioButtonBed,
                       radioButtonKad ;
  private JList keuzelijstAfdelingen;

  public static void main(String[] argv) {
      GUIElementenSwingOef frame = new GUIElementenSwingOef();
      frame.setSize(500, 350);
      frame.createGUI();
      frame.setTitle("Taakinvulling werknemer");
      frame.setVisible(true);
  }

  private void createGUI(){
    setDefaultCloseOperation(EXIT_ON_CLOSE)  ;
    setLayout(new FlowLayout() );
    JPanel panelDatum = new JPanel();
    JPanel panelNaam  = new JPanel();
    JPanel panelFunctieAfdeling = new JPanel();
    JPanel panelUren  = new JPanel();
    JLabel lblMaanden = new JLabel("maand:") ;
    String[] cboElementen = {
      "Januari","Februari","Maart","April","Mei","Juni", "Juli",
      "Augustus", "September","Oktober","November", "December" };
    cboMaanden = new JComboBox(cboElementen);
    JLabel lblJaar = new JLabel("jaar:") ;
    txtJaar = new JTextField(4); // 4 pos voor grootte tekstveld
    panelDatum.add(cboMaanden);
    panelDatum.add(lblJaar);
    panelDatum.add(txtJaar);
    JLabel lblNaam = new JLabel("naam:") ;
    txtNaam = new JTextField(30);
    panelNaam.add(lblNaam);
    panelNaam.add(txtNaam);
    radioButtonArb = new JRadioButton("arbeider");
    radioButtonBed = new JRadioButton("bediende");
    radioButtonKad = new JRadioButton("kaderlid");
    ButtonGroup groepering = new ButtonGroup();
    groepering.add(radioButtonArb);
    groepering.add(radioButtonBed);
    groepering.add(radioButtonKad);
```

```
    panelFunctieAfdeling.add(radioButtonArb);
    panelFunctieAfdeling.add(radioButtonBed);
    panelFunctieAfdeling.add(radioButtonKad);
    JLabel lblAfdeling = new JLabel("afdeling:") ;
    String[] lijstAfdelingen = {"productie","verkoop","inkoop",
      "marketing" };
    keuzelijstAfdelingen = new JList(lijstAfdelingen);
    keuzelijstAfdelingen.setVisibleRowCount(2);
    JScrollPane lijstPaneel = new JScrollPane(keuzelijstAfdelingen);
    panelFunctieAfdeling.add(lblAfdeling);
    panelFunctieAfdeling.add(lijstPaneel);
    JLabel lblUren = new JLabel("aantal uren:") ;
    txtUren = new JTextField(20);
    panelUren.add(lblUren);
    panelUren.add(txtUren);
    buttonOK = new JButton("OK");
    buttonOK.addActionListener(this);
    add(panelDatum);
    add(panelNaam);
    add(panelFunctieAfdeling);
    add(panelUren);
    add(buttonOK);
  }

  @Override
  public void actionPerformed(ActionEvent e){
    if (e.getSource()==buttonOK){
      String ingave ="wanneer: " + cboMaanden.getSelectedItem()
            + " " + txtJaar.getText()
            + "\nnaam: " + txtNaam.getText()
            + "\nfunctie: "
            + (radioButtonArb.isSelected()?
               radioButtonArb.getText():"")
            + (radioButtonBed.isSelected()?
               radioButtonBed.getText():"")
            + (radioButtonKad.isSelected()?
               radioButtonKad.getText():"")
            + "\nafdeling: "
            + (keuzelijstAfdelingen.isSelectionEmpty()?
              "":keuzelijstAfdelingen.getSelectedValue())
            + "\nuren: " + txtUren.getText() ;

      JOptionPane.showMessageDialog(null, ingave);
     }
  }
}
```

Oplossing oefening 37 :

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class AnnuiteitSwing extends JFrame implements ActionListener
{
  private JTextField txtRentevoet;
  private JTextField txtPerioden;
  private JTextField txtKapitaal;
  private JLabel lblAnnuiteit;
  private JButton btnBereken;

  public AnnuiteitSwing() {
    super("Constante annuïteiten");
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setLayout(new GridLayout(4, 2));

    JLabel lblRentevoet = new JLabel("periodieke rentevoet");
    add(lblRentevoet);
    txtRentevoet = new JTextField();
    add(txtRentevoet);

    JLabel lblPerioden = new JLabel("aantal perioden");
    add(lblPerioden);
    txtPerioden = new JTextField();
    add(txtPerioden);

    JLabel lblKapitaal = new JLabel("ontleend kapitaal");
    add(lblKapitaal);
    txtKapitaal = new JTextField();
    add(txtKapitaal);

    btnBereken = new JButton("Bereken afbetaling");
    add(btnBereken);

    lblAnnuiteit = new JLabel();
    lblAnnuiteit.setHorizontalAlignment(SwingConstants.RIGHT);
    add(lblAnnuiteit);

    btnBereken.addActionListener(this);

    setSize(300, 120);
    show();
  }
```

```
  @Override
  public void actionPerformed(ActionEvent evt) {
    double dRentevoet = Double.parseDouble(txtRentevoet.getText());
    int iPerioden = Integer.parseInt(txtPerioden.getText());
    double dKapitaal = Double.parseDouble(txtKapitaal.getText());

    // tussenresultaat: v = (1 + rentevoet)^jaarperioden
    double v = 1.0;
    iPerioden /= 12;
    for (int k = 1; k <= (iPerioden) ; k++)
      v *= 1 + dRentevoet;
    double dAnnuiteit = dKapitaal * dRentevoet * v / (v - 1) / 12;
    lblAnnuiteit.setText(String.valueOf(dAnnuiteit));
  }
  public static void main(String[] argv) {
    AnnuiteitSwing hetVenster = new AnnuiteitSwing();
  }
}
```

Oplossing oefening 38 :

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class Keuzelijst extends JFrame implements ActionListener  {
   private JButton cmdVerwijderen, cmdOmhoog,
                   cmdOmlaag, cmdNieuw;
   private JPanel paneel;
   private JList keuzelijst;
   private DefaultListModel eenModel;
   public static void main (String[] args) {
      Keuzelijst frame = new Keuzelijst();
      frame.setSize(240,200);
      frame.createGUI();
      frame.setVisible(true);
   }
   private void createGUI() {
     setDefaultCloseOperation(EXIT_ON_CLOSE);
     paneel = new JPanel();
     cmdVerwijderen = new JButton("Verwijderen");
     cmdOmhoog = new JButton("Omhoog");
     cmdOmlaag = new JButton("Omlaag");
     cmdNieuw = new JButton("Nieuw");
     paneel.setLayout(new GridLayout(4,1));
     paneel.add(cmdVerwijderen);
```

```
    paneel.add(cmdOmhoog);
    paneel.add(cmdOmlaag);
    paneel.add(cmdNieuw);

    eenModel = new DefaultListModel();
    eenModel.addElement("Water");
    eenModel.addElement("Wijn");
    eenModel.addElement("Bier");
    eenModel.addElement("Koffie");
    eenModel.addElement("Thee");
    eenModel.addElement("Soep");
    keuzelijst = new JList(eenModel);

    keuzelijst.setFixedCellWidth(124);
    keuzelijst.setSelectionMode(
       ListSelectionModel.SINGLE_SELECTION);
    add(keuzelijst,"West");
    add(paneel,"East");
    cmdVerwijderen.addActionListener(this);
    cmdOmhoog.addActionListener(this);
    cmdOmlaag.addActionListener(this);
    cmdNieuw.addActionListener(this);
  }

  @Override
  public void actionPerformed(ActionEvent e) {

      if (e.getSource()== cmdVerwijderen) {
           int dewelke = keuzelijst.getSelectedIndex();
           if (dewelke != -1) eenModel.removeElementAt(dewelke);
      }
      if (e.getSource()== cmdOmhoog) {
           int dewelke = keuzelijst.getSelectedIndex();
           Object dedrank = keuzelijst.getSelectedValue();
           if (dewelke > 0) {
               eenModel.removeElementAt(dewelke);
               eenModel.add(dewelke-1,dedrank);
           }
      }
      if (e.getSource()== cmdOmlaag) {
           int dewelke = keuzelijst.getSelectedIndex();
           Object dedrank = keuzelijst.getSelectedValue();
           if (dewelke != -1 && dewelke < eenModel.getSize()-1) {
               eenModel.removeElementAt(dewelke);
               eenModel.add(dewelke+1,dedrank);
           }
      }
      if (e.getSource()== cmdNieuw) {
          String deNieuwe = JOptionPane.showInputDialog(
```

```
              null,"Nieuwe drank:");
            if (! deNieuwe.equals("")) {
                eenModel.addElement(deNieuwe);
            }
        }
    }
}
```

Colofon

Sectorverantwoordelijke	Ortaire Uyttersprot
Cursusverantwoordelijk	Jean Smits
Medewerkers	Steven Lucas Brigitte Loenders Hans De Smet
Versie	3-10-2013