Java Essentials

# Hoofdstuk 8

## Eenvoudige klassen: Deel 2

DE HOGESCHOOL MET HET NETWERK

Hogeschool PXL – Elfde-Liniestraat 24 – B-3500 Hasselt
www.pxl.be - www.pxl.be/facebook

# Inhoud

1. Inleiding
2. Wrappers voor primitieve datatypes
3. ArrayLists
4. Datums en tijden
5. Samenvatting

# 1. Inleiding

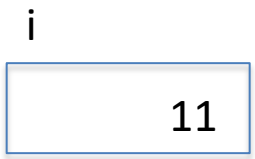
Doel:

- eenvoudige klassen uit de Java API gebruiken
- Java API documentatie leren gebruiken

# 2. Wrappers voor primitieve datatypes

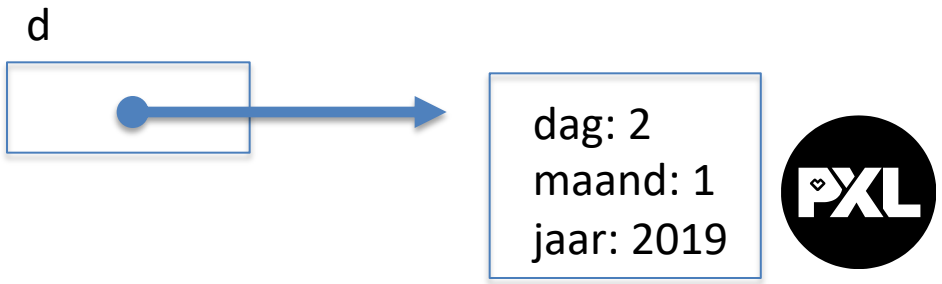
Primitieve variabele:

```
int i;
i = 10;
i++;
System.out.println("de waarde van i " + i);
```




Referentievariabele:

```
Datum d;                          // declaratie van de variabele
d = new Datum (2, 1, 2019);       // aanmaak van de variabele
d.printDatum();                   // gebruik van de variabele
```



# 2.1 Wrapperklassen

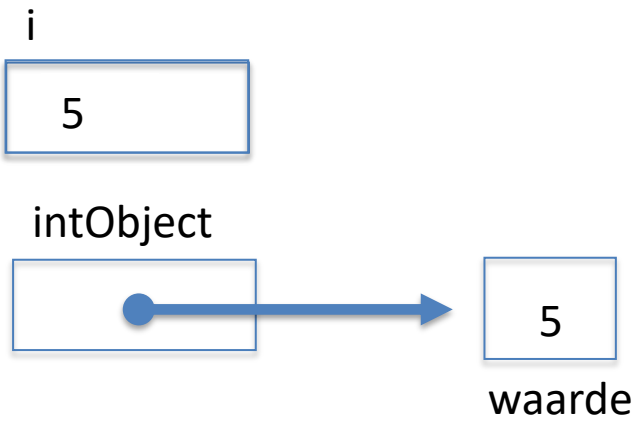
Voor elk primitief datatype bestaat er een object variant
= wrapperklasse

primitief datatype → wrapperklasse


Wrapperklassen
Integer, Long, Double, Boolean, Character

# Voorbeeld:

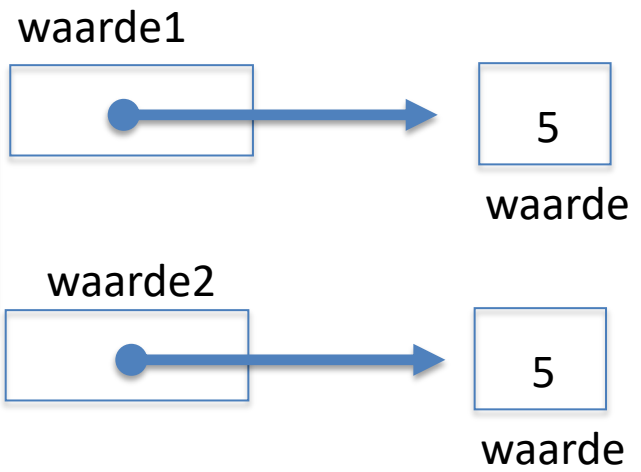
```
int i = 88;
Integer intObject = new Integer(5);
System.out.println(intObject);
i =  intObject.intValue();
System.out.println(i);
```




Output?

**5**
**5**

```
Integer waarde1 = new Integer(5);
Integer waarde2 = new Integer(5);
System.out.println(waarde1 == waarde2);
System.out.println(value1.equals(waarde2));
```




Output?

**false**
**true**

@Deprecated(since="9")
public Integer(int value)

# 2.2 Autoboxing

**autoboxing** primitief datatype → wrapper object

```
Integer intObject = 5;
```

**auto-unboxing** wrapper object → primitief datatype

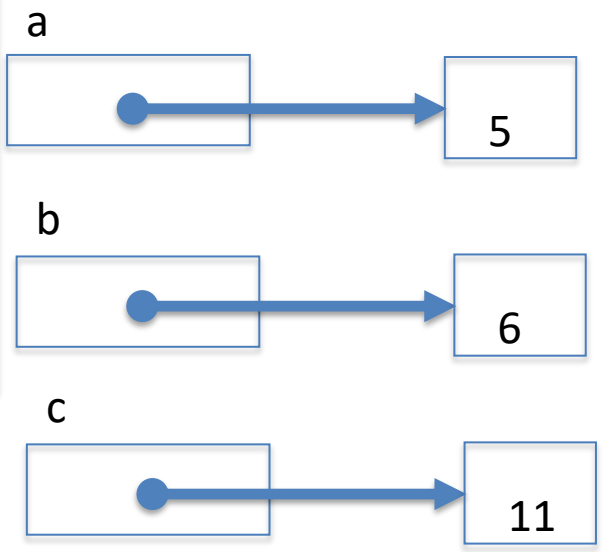
```
int intPrimitive =  new Integer(6);
```

Omzettingen gebeuren *automatisch* door de compiler.

=> wrapper objecten en primitieve datatypen mogen door elkaar gebruikt worden

# Voorbeelden

```
public class SlechtAutoBoxing {
    public static void main(String[] args) {
        Integer a = 5;
        Integer b = 6;
        Integer c = a + b;
        System.out.println(c);
    }
}
```



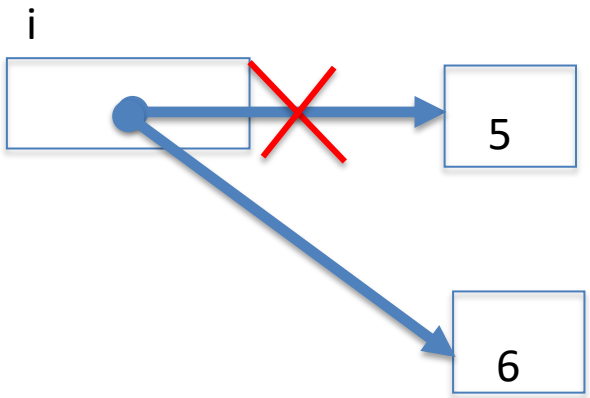

**wanneer er veel gerekend moet worden**
**=> zeker niet met objecten werken!!!!**

```
public class AutoBoxVerhoging {
    public static void main(String[] args) {
        Integer i = 5;
        i++;
        System.out.println(i);
    }
}
```

## Wat gebeurt er in het intern geheugen?

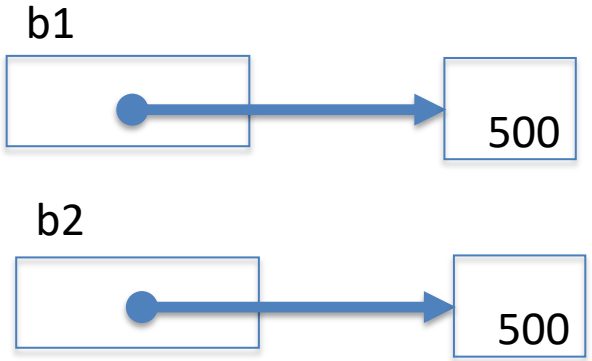
1. autoboxing

2. auto – unboxing
   primitieve waarde verhogen
   autoboxing




# Autoboxing en de == en != operatoren

```
public class Test {
    public static void main(String[] args) {
        Integer b1 = 500;
        Integer b2 = 500;
        System.out.println(b1 == b2); //false
        System.out.println(b1 != b2); //true
    }
}
```



# Method overloading en autoboxing

```
public class AutoBoxingOverloading{
    public static void main(String[] args) {
        method(5);
    }

    private static void method(long i) {
        System.out.println("long");
    }

    private static void method(Integer i) {
        System.out.println("Integer");
    }
}
```

**Output?** long

**Eerst steeds widening**
**Reden: compatibiliteit met vorige Java versies**

# Method overloading en autoboxing

```
public class Test {
    public static void main(String[] args) {
        method(5);
    }

    private static void method(Long l) {
        System.out.println("Long");
    }
}
```

**Output?** foutmelding

**Autoboxing alleen naar overeenkomstig wrapper type**

# 2.3 static members: static variabelen

WrapperKlassen: Integer, Long, Double

| | |
|---|---|
| MIN_VALUE | minimale waarde |
| MAX_VALUE | maximale waarde |

WrapperKlassen: Double

| | |
|---|---|
| NaN | resultaat van 0/0 |
| NEGATIVE_INFINITY | resultaat van getal(<0)/0 |
| POSITIVE_INFINITY | resultaat van getal (>0)/0 |

**Zie Javadoc voor andere klasse (static) variabelen**

- Conversie methoden String => primitief datatype

| | | |
|---|---|---|
| bv | String -> int | Integer.parseInt () |
| | String -> double | Double.parseDouble() |

## Andere static methoden:

- Double.isNAN() nagaan getal ongeldige waarde (0/0)
- Double.isInfinite() nagaan of de waarde oneindig is

```
System.out.println(Double.isInfinite(5.0 / 0));   //true
```

Opgepast: er is ook een niet-static versie van de methode!

Vb.

```
Double d = new Double(5);
d = d / 0;
System.out.println(d.isInfinite());          //true
```

**Zie Javadoc voor andere klasse (static) methoden**

# Opdracht 1: *Werken met de klasse Integer*

- Open de API-documentatie van de klasse Integer.
- Maak een programma dat aan de gebruiker een waarde vraagt en deze inleest als een String.
- Zet de string-waarde om in een Integer-object.
- Tel bij dit object een waarde op en druk het resultaat af.
- Druk het aantal bits voor een integer af.
- Druk het aantal bytes voor een integer af.

# Opdracht 2:

*Wat is de output van onderstaand programma?*

- Bekijk de uitvoer van Opdracht2.java.
- Gebruik de java doc om de // - lijnen van de nodige commentaar te voorzien.
- Voer het programma opnieuw uit met de waarde 253, -45 voor k. Ontleed de bekomen resultaten.

# Opdracht 3: *De klasse BigInteger en BigDecimal.*

- Open de API documentatie van de klasse BigInteger en de klasse BigDecimal.

- Schrijf een programma voor het berekenen van faculteit van een getal (bv? 5! = 5 ×4 ×3 ×2 ×1).

  Test je programma uit => wat merk je?

- Herschrijf je programma door gebruik te maken van de klasse BigInteger.

# 3. ArrayLists

## 3.1 Inleiding

ArrayList
= geordende verzameling van objecten waarbij het aantal objecten op voorhand niet vast ligt (zoals een list in Python)

De klasse ArrayList bevindt zich in het pakket java.util
➔ volledige naam van de klasse = java.util.ArrayList.

Creatie van een Arraylist

```
import java.util.ArrayList;

ArrayList list = new ArrayList();
```

```
ArrayList  list = new ArrayList ();
```

Je hebt een nu een lege ArrayList waarin om het even welk object geplaatst kan worden.
Wil je alleen objecten van een bepaalde klasse in je arraylist, dan dien je dit bij het aanmaken te vermelden.

Bv alleen objecten van de klasse Student

```
ArrayList<Student> list = new ArrayList<>();
```

# !! Uitleg van alle methoden → Java API-documentatie !!

## Enkele nuttige methoden

```
ArrayList<E> list = new ArrayList<>();
```

| Methode | Omschrijving van de methode |
|---|---|
| boolean add(E element) | voegt element toe achteraan in de rij |
| void add(int index, E element) | voegt element toe op de gespecifieerde index |
| E get(int index) | geeft het element dat zich op index bevindt terug |
| E set (int index, E element) | vervangt het element dat zich op index bevindt door het opgegeven element |
| boolean remove(Object o) | verwijdert het eerste gespecifieerde element uit de rij |
| E remove(int index) | verwijdert het element op de gespecifieerde index |

```
ArrayList<E> list = new ArrayList<>();
```

| Methode | Omschrijving van de methode |
|---|---|
| int indexOf(E element) | geeft de index van het eerste voorkomen van het element (gebruikmakend van de methode equals), -1 indien het element niet voorkomt |
| void clear() | verwijdert alle elementen |
| int size() | geeft het aantal element in de arrayList |

**!! Andere methoden → Java API-documentatie !!**

# Opdracht 4: *De klasse ArrayList*

1. Maak een arrayList lijst aan waar je alleen Strings in kan plaatsen.
2. Voeg 3 String-objecten voorbeeld1, voorbeeld2, voorbeeld3 toe aan de lijst.
3. Voeg een 4$^{de}$ String toe, tussen 1$^{ste}$ en 2$^{de}$ String.
4. Verander de juist toegevoegde String en druk alles af.
5. Verwijder het juist tussengevoegde element. Druk alles af, maak hierbij gebruik van een for each lus.
6. Op welke plaats komt het object voorbeeld2?
7. Verwijder alle elementen uit de arrayList.

# Opdracht 5: *De klasse ArrayList*

- Importeer de klassen Student en Klas. Overloop deze klassen.
- Hoe kan je ervoor zorgen dat studenten op basis van hun naam kunnen opgezocht worden in de arrayList?

# 4. Datums en tijden
## 4.1 Inleiding

- Sinds Java 8 nieuwe klassen in package **java.time**

- Enkele moeilijkheden
  - 2 manieren om naar tijd te kijken
    mens => jaren, maanden, dagen, uren, …
    computer => aantal tijdseenheden vanaf bepaald punt
  - Tijdszones => tijd verschilt op ieder punt op de aarde

| | |
|---|---|
| Europe/London | Z Greenwich |
| Europe/Brussels | +01:00 |
| America/New_York | -05:00 |

  - Schrikkeljaren
  - Zomertijd

# 4.2 Computertijden: de klasse Instant

- Computertijd = aantal tijdseenheden voor of na een nulpunt

  nulpunt = 1 januari 1970 om 00:00:00 (EPOCH)

  tijdseenheid = 1 nanoseconde = $10^{-9}$ sec


- Klasse variabelen EPOCH, MAX, MIN

- Klasse Instant geen publieke constructor
  => onmogelijk om zelf instanties van klasse Instant aan te maken

- Statische methoden die instantie van klasse teruggeven

  Bvb. Instant now = Instant.now();

- Methoden om bewerkingen te doen op een tijdstip (zie Javadoc)

```java
import java.time.Instant;

public class InstantApp {

    public static void main(String[] args) {
        System.out.println(Instant.EPOCH);
        System.out.println(Instant.MIN);
        System.out.println(Instant.MAX);

        Instant now = Instant.now();
        System.out.println(now);
        System.out.println(now.getEpochSecond());
        System.out.println(now.getNano());

        Instant earlier = now.minusSeconds(20);
        System.out.println(earlier);

        Instant later = now.plusSeconds(20);
        System.out.println(later);

        System.out.println(now.isAfter(earlier));
        System.out.println(now.isBefore(later));
    }
}
```

Greenwich

1970-01-01T00:00:00Z
-1000000000-01-01T00:00:00Z
+1000000000-12-31T23:59:59.999999999Z

# Opdracht 6: *De klasse Instant*

Maak een programma dat de gebruiker vraagt zijn naam in te geven.
Toon hoe lang dit geduurd heeft (in seconden).

# 4.3 Menselijke datums en tijden

## 4.3.1 Enumeraties voor weekdagen en maanden

### Enumeratie Month

- bevat 12 instanties voor de maanden: Month.JANUARY, …
- interessante methoden (zie Javadoc)

…

### Enumeratie DayOfWeek

- bevat 7 instanties voor de dagen: DayOfWeek.MONDAY, …
- interessante methoden (zie Javadoc)

```java
import java.time.DayOfWeek;
import java.time.Month;

public class EnumMontDay {
    public static void main(String[] args) {
        Month maand = Month.FEBRUARY;
        DayOfWeek dag = DayOfWeek.FRIDAY;
        System.out.println(dag + " " + maand);

        System.out.print("Het aantal dagen in deze maand ");
        System.out.println(maand.length(true));

        System.out.println("5 maanden verder is de maand " +
                                                maand.plus(5));
        System.out.print("3 dagen eerder is ");
        System.out.println(dag.minus(3));
    }
}
```

```
FRIDAY FEBRUARY
Het aantal dagen in deze maand 29
5 maanden verder is de maand JULY
3 dagen eerder is TUESDAY
```

# Opdracht 7: *Weekdagen en maanden*

Maak een programma dat de gebruiker vraagt een weekdag in te geven (1-7) en een aantal dagen om erbij op te tellen.
Druk af welke dag van de week dit is.

# 4.3.2 Lokale datums en tijden

- Lokaal = geen rekening houdend met de tijdszones of het zomeruur
- Tijden => klasse LocalTime

  Datums zonder tijden => klasse LocalDate

  Datum en tijd => klasse LocalDateTime

- Deze klassen hebben geen constructors
  => statische methoden om instanties te bekomen

  bvb methode now()
  methoden of()

```java
import java.time.*;

public class LocalDateTimeApp {

    public static void main(String args[]) {
        LocalDate nowDate = LocalDate.now();
        LocalTime nowTime = LocalTime.now();
        LocalDateTime nowDateTime = LocalDateTime.now();

        LocalDate otherDate = LocalDate.of(2015, 6, 23);
        LocalTime otherTime = LocalTime.of(10, 25, 2);
        LocalDateTime otherDateTime = LocalDateTime.of(otherDate, otherT

        System.out.println(nowDate);
        System.out.println(nowTime);
        System.out.println(nowDateTime);

        System.out.println(otherDate);
        System.out.println(otherTime);
        System.out.println(otherDateTime);
    }
}
```

```
2018-12-28
10:08:43.621
2018-07-12T10:08:43.621
2015-06-23
10:25:02
2015-06-23T10:25:02
```

# Opdracht 8: *Datums en tijden*

Maak een programma dat van je geboortedatum het volgende afdrukt:

- De hoeveelste dag van het jaar het was
- De dag van de week
- Of dit al dan niet een schrikkeljaar was

# 4.4 Tijdsduur

Te gebruiken klassen

- Duration

  tijdsverschillen tussen machinetijden (= instanties van de klasse Instant)

- Period

  tijdsverschillen tussen “datebased–time” (= menselijk tijden)

- ChronoUnit: enumeratie
  - verschillende tijdseenheden: ChronoUnit.HOURS, ChronoUnit.DAYS, ….
  - tijdsverschil kan met methode between() omgezet worden in de gekozen tijdseenheid

```
…
public class DurationApp {
    public static void main(String
        Instant now = Instant.now()
        System.out.println("nu " + n
        Instant later =  now.plusSec
        System.out.println("xx secon
        Duration duration =  Duratio
        System.out.println(duration

        long milliseconds = ChronoUn

        System.out.println(millisec

        LocalDate nowDate = LocalDate.now();
        LocalDate thenDate = LocalDate.of(1980, 2, 15);

        System.out.println("Datum1: " + nowDate);
        System.out.println("Datum2: " + thenDate);

        Period period = Period.between(thenDate, nowDate);
        System.out.println("jaar: " + period.getYears() + " maanden:" +
                                  period.getMonths() + " dagen: " + period.getDays());

        long days = ChronoUnit.DAYS.between(thenDate, nowDate);
        System.out.println("aantal dagen tussen 2 datums " + days);
    }
}
```

```
nu 2018-12-28T08:21:01.140Z
xx seconden later 2018-12-28T09:04:46.838456398Z
2625
2625698
Datum1: 2018-12-28
Datum2: 1980-02-15
jaar: 38 maanden:10  dagen: 13
aantal dagen tussen 2 datums 14196
```

# Opdracht 9: *Tijdsduur*

Maak een programma dat de periode berekent sinds je geboorte. Druk van deze periode het volgende af

- het aantal dagen
- het aantal maanden
- het aantal jaren
- Bereken je leeftijd uitgedrukt in maanden

# 3.5 Formattering van datums en tijden

- Conversie tussen tekst-voorstelling en Java datum/tijd-objecten
  => klasse DateTimeFormatter

- Conversie van tekenreeks naar datum/tijd-object (parse)

- Conversie van datum/tijd-object naar een tekenreeks (format)

- Conversie kan volgens eigen opgegeven patroon of volgens standaard patronen (statische constanten van de klasse). Verdere uitleg tekens in patroon zie Javadoc.

```
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;

public class FormatterApp {
    public static void main(String[] args) {
        DateTimeFormatter myFormatter1 =
              DateTimeFormatter.ofPattern("dd/MM/yyyy HH:mm:ss");
        DateTimeFormatter myFormatter2 =
              DateTimeFormatter.ofPattern("d/M/yy H:m:s");
        DateTimeFormatter myFormatter3 =
              DateTimeFormatter.ofPattern("yyyy-MM-dd");
        DateTimeFormatter myFormatter4 =
              DateTimeFormatter.ofPattern("EEEE dd MMMM yyyy");


        LocalDateTime dt1 =
              LocalDateTime.parse("03/08/2018 13:45:03",myFormatter1);

        System.out.println(myFormatter1.format(dt1));
        System.out.println(myFormatter2.format(dt1));
         System.out.println(myFormatter3.format(dt1));
         System.out.println(myFormatter4.format(dt1));

    }
}
```

03/08/2018 13:45:03

3/8/18 13:45:3

2018-08-03

vrijdag 03 augustus 2018

```java
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;

public class FormatterApp {
    public static void main(String[] args) {
        DateTimeFormatter myFormatter1 =
              DateTimeFormatter.ofPattern("dd/MM/yyyy HH:mm:ss");
        DateTimeFormatter myFormatter2 =
              DateTimeFormatter.ofPattern("d/M/yy H:m:s");

        LocalDateTime dt1 =
              LocalDateTime.parse("03/08/2018 13:45:03",myFormatter2);

        System.out.println(myFormatter1.format(dt1));
        System.out.println(myFormatter2.format(dt1));
    }
}
```

```
Exception in thread "main" java.time.format.DateTimeParseException: Text
'03/08/2018 13:45:03' could not be parsed at index 8
      at
java.time.format.DateTimeFormatter.parseResolved0(DateTimeFormatter.java:1949)
      at java.time.format.DateTimeFormatter.parse(DateTimeFormatter.java:1851)
      at java.time.LocalDateTime.parse(LocalDateTime.java:492)
      at voorbeeldenTijd.FormatterApp.main(FormatterApp.java:11)
```

- Afdrukken van datum/tijd-objecten

  Gebruik maken van printf ()

```
import java.time.LocalDateTime;

public class DatumPrintf {
    public static void main(String[] args) {
        LocalDateTime datum = LocalDateTime.of(2018, 10, 1, 17, 45, 7);

        System.out.printf("%1$td/%1$tm/%1$ty %1$tH:%1$tM:%1$tS \n", datum);
        System.out.printf("%1$td/%1$tb/%1$tY %1$tI:%1$tM:%1$tS \n", datum);
        System.out.printf("%1$tA %1$td %1$tB %1$tY %1$tI:%1$tM:%1$tS \n", datum);
    }
}
```

```
01/10/18 17:45:07
01/okt/2018 05:45:07
maandag 01 oktober 2018 05:45:07
```

Meest gangbare conversie-karakters voor datums en tijden (zie ook Javadoc)

| Conversie | Omschrijving |
|---|---|
| td | dag in Twee cijfers |
| ta | dag in tekst afgekort |
| tA | dag in tekst voluit |
| tm | maand in 2 cijfers |
| tb | maand in tekst afgekort |
| tB | maand in tekst voluit |
| ty | jaar in twee cijfers |
| tY | jaar in vier cijfers |
| tH | uren (24 uur) |
| tI (hoofletter i) | uren (12 uur) |
| tM | minuten |
| tS | seconden |

# Opdracht 10: *Formattering*

- Maak een programma dat de gebruiker vraagt een datum in te geven in het formaat DD/MM/YYYY.
- Zet de bekomen tekst om in een object van de klasse LocalDate en druk de inhoud af in het formaat YYYY-MM-DD.
- Druk van de dag en de maand van deze datum in het Nederlands af.

# 5. Samenvatting

In dit hoofdstuk werden veel gebruikte klassen besproken

- Wrapperklassen
  => om primitieve datatypen te kunnen gebruiken als objecten
- ArrayLists
  => om een verzameling van objecten te maken waarvan we het aantal op voorhand niet kennen
- Klassen voor Datum/Tijd
  - ✓ Computertijd => Instant
  - ✓ Datum (zonder tijdzone) => LocalDate
  - ✓ Tijd (zonder tijdzone) => LocalTime
  - ✓ Tijdsduur (computertijd) => Duration
  - ✓ Tijdsduur (menselijke tijd) => Period
- enumeraties voor Datum/Tijd
  =>Month, DayOfWeek , ChronoUnit