

Informatica

les 1

Java versus Python

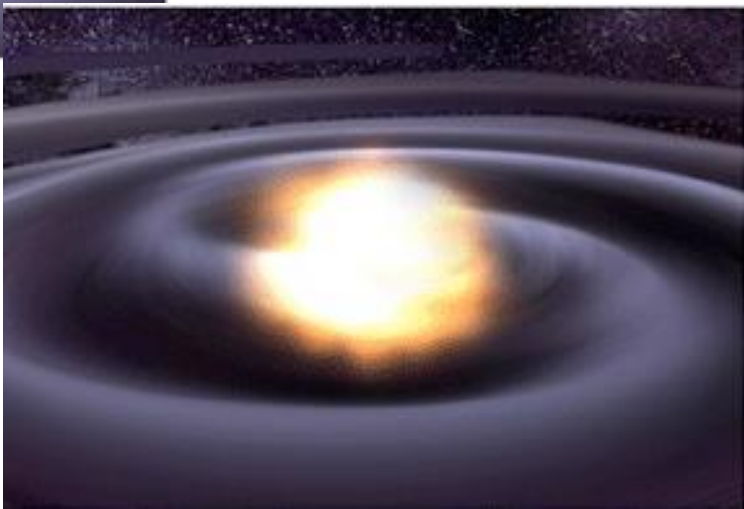
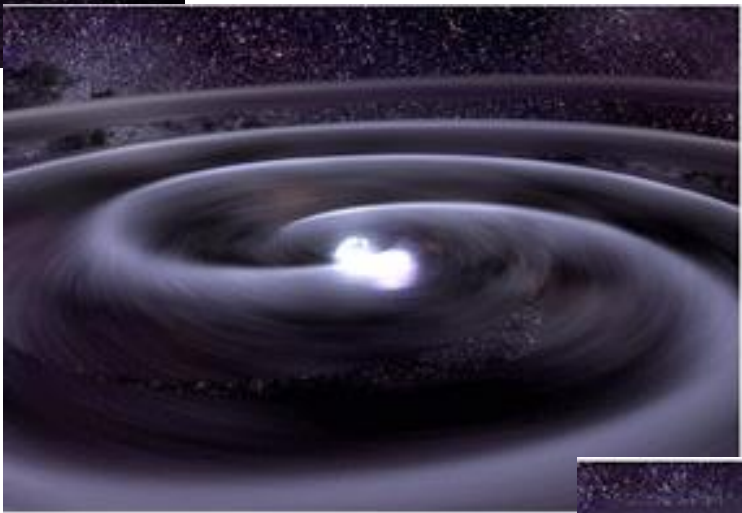
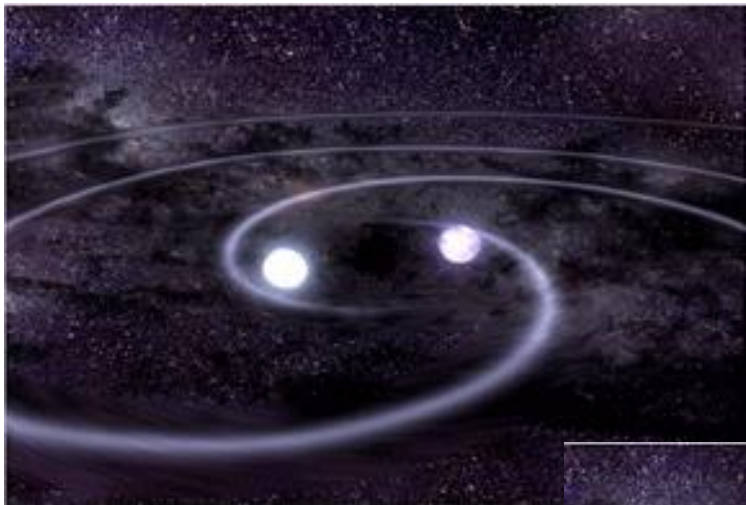
Jan Lemeire

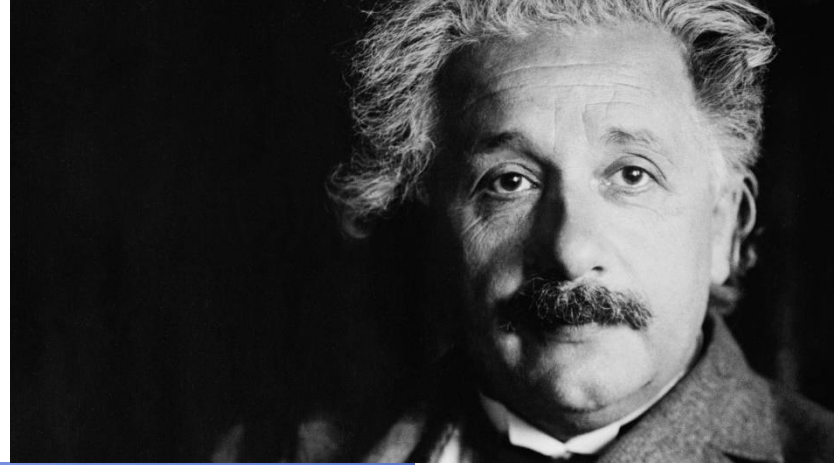
Informatica IR 2^e semester

februari – mei 2020



Vrije Universiteit Brussel

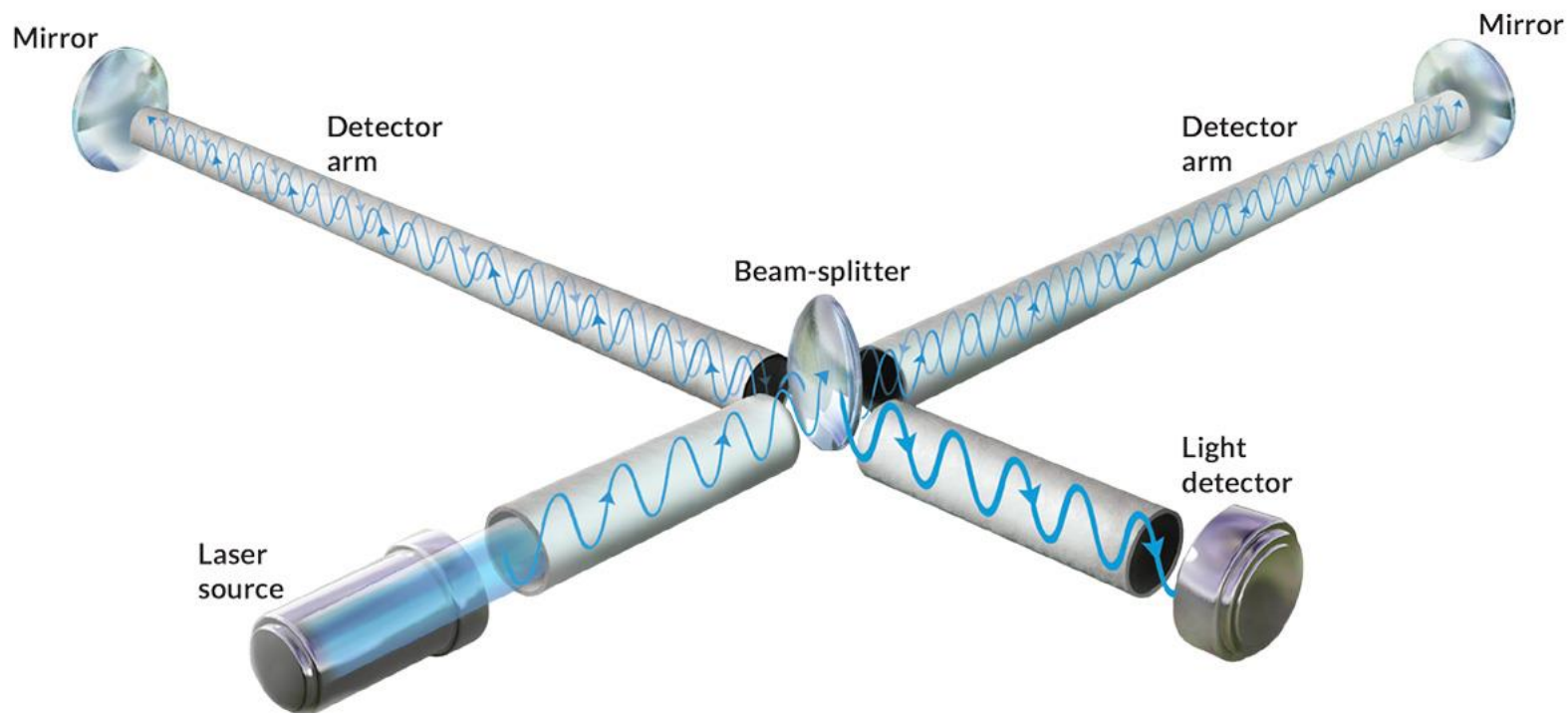
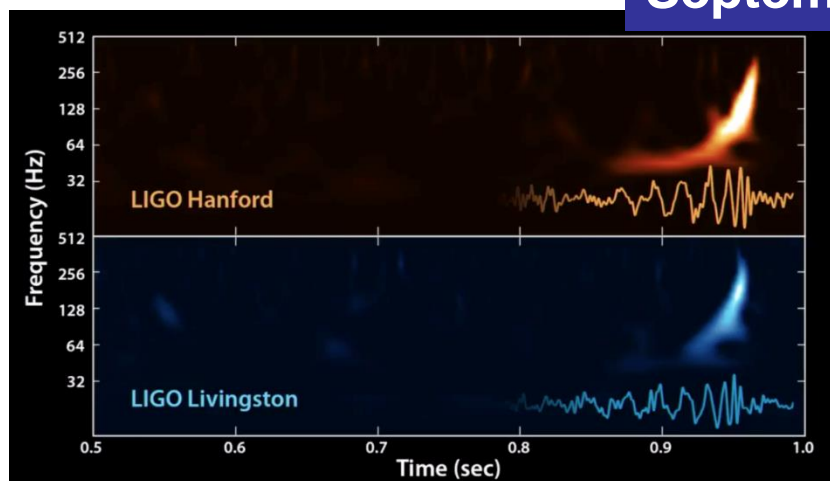




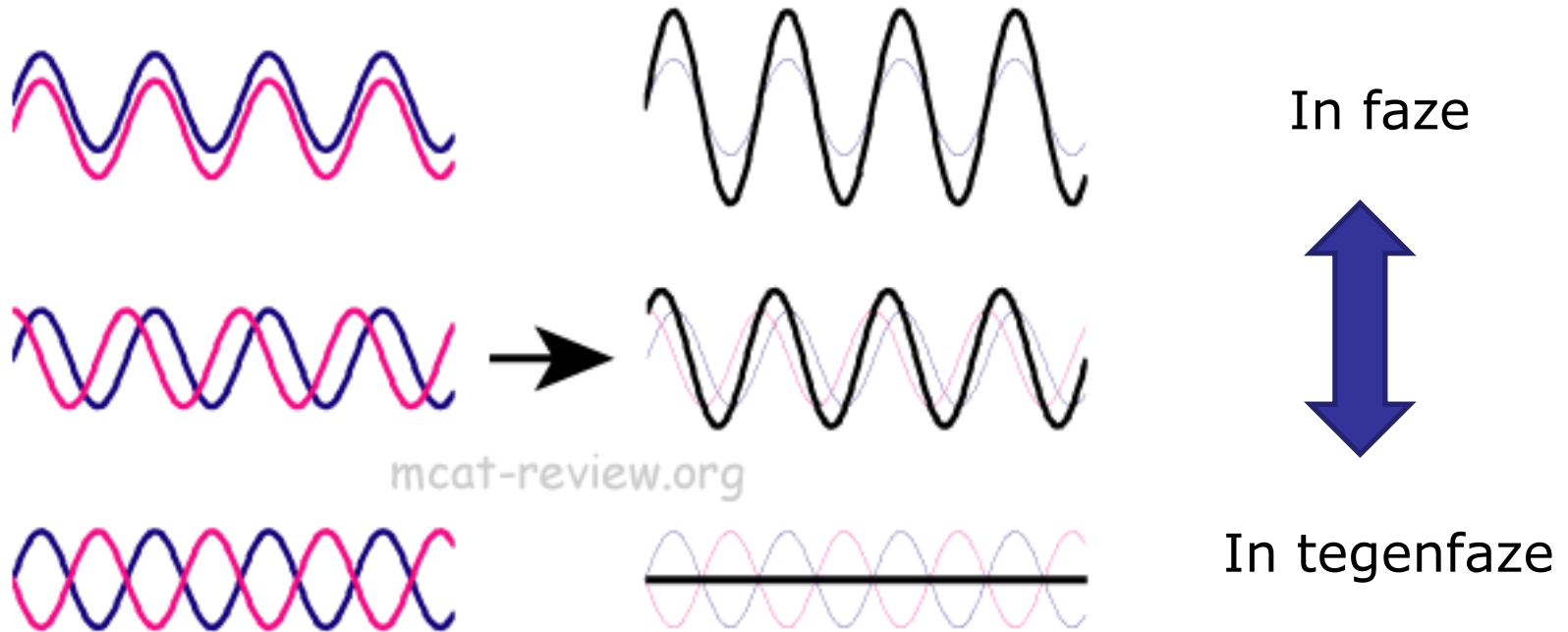
LIGO

The First Observation
of Gravitational Waves

September 14, 2015



Superpositie van signalen



UPDATE: August 14, 2017

**Detection of a gravitational wave by 3 detectors,
also by the European Virgo.**

**1.8 billion years ago, collision of black holes
of 25 and 31 times the mass of the sun**

14 augustus 2017 om 12.30.43 uur
Belgische tijd.

Het verschijnsel is voor het eerst tegelijk waargenomen door drie detectoren, en twee soorten van detectoren: de twee LIGO's in de VS en de Europese Virgo. Voor de Europese Virgo was de waarneming een primeur. Virgo, die een uitgebreide update heeft gekregen en nu de Advanced Virgo heet, was nog maar twee weken opnieuw bezig met het verzamelen van wetenschappelijke gegevens.

De botsing vond plaats op zowat 1,8 miljard lichtjaar afstand.

GW170814, zoals de nieuw ontdekte zwaartekrachtgolf is gedoopt (Gravitational Wave en de datum), is veroorzaakt in de laatste ogenblikken van de versmelting van twee zwarte gaten die 31 en 25 keer zo zwaar waren als de zon

Albert Einstein had in 1915 geopperd dat het heelal bestaat uit ruimte en tijd, en dat die één geheel vormen. Na een heftige gebeurtenis kan die "ruimtetijd" trillen. De schokgolven, zwaartekrachtgolven, gaan dan door het heelal als rimpelingen na een steen in een vijver. Bij zo'n rimpeling rekt de ruimte iets uit of krimpt zij iets. Hoe groter de massa en hoe sneller de beweging van een massarijk object, des te sterker de zwaartekrachtgolf.



06 wetenschap

DE STANDAARD
VRIJDAG 19 JULI 2019

ok Vlaanderen zet zijn schouders onder de Einstein Telescope

Drielandenregio wil 'kosmisch afluistercentrum'

t zuiden van

gaten en gravitatiegolven voortvloei-
en) te huisvesten. 'Telescope' is overi-



De Einstein Telescope komt 200 meter diep te liggen en wordt een gelijkzijdige driehoek met zijden van
10 kilometer. © rr



2010

350 Million triangles/second
3 Billion transistors GPU

1995

5.000 triangles/second
800.000 transistors GPU

2016

14.000 Million triangles/second
15 Billion transistors GPU

[Beta - MikeTheSempai]





B SYNCHRONIZE





We zitten met
de noepers

[illegible]

De buurtpolitie speelt wel vaker een geloopt spel met fictie en realiteit.

Comenius' roman van de kunststof van de *Deusopvoeding*, die zich in tegenstelling tot zijn eerste roman *De Thaumtopoeia* niet meer aan enkele thema's van de magische wereld maar aan de algemene opvoeding van de kindergemeenschap in de 17e eeuw richt. Het is een roman die de opvoeding van een kind in de 17e eeuw als een proces van kennisoverdracht (didacticus) van de *Deusopvoeding* aan de werkers van de eerste school met veel succes en succesvol met een aantal kinderen van de 17e eeuw. Het is een roman die de opvoeding van een kind in de 17e eeuw als een proces van kennisoverdracht (didacticus) van de *Deusopvoeding* aan de werkers van de eerste school met veel succes en succesvol met een aantal kinderen van de 17e eeuw.

Goldrush

op de grafische chips

Gamers zijn boos. Door het succes van bitcoin en artificiële intelligentie raken zij niet meer aan grafische kaarten. De prijs ervan is met 30 procent gestegen en sommige winkels zouden maar één kaart per klant verkopen.

DOMINIQUE DECENTYN, ILLUSTRATE JORDEN WARR

De symptomen kunnen ook wel geringe zijn, bijvoorbeeld een lichte ontsteking van de keel of een lichte pijn in de borst. Het kan ook zijn dat de symptomen helemaal ontbreken. Het is daarom belangrijk om de arts hiervan in kennis te stellen.

1,000 cases

Was eren een GPU? Je kunt spreken over een GPU, niet alleen maar de centrale processor van je pc (CPU of Central Processing Unit). Maak je die een chip van Intel. Maar een GPU of Graphics Processing Unit is een chip die een beeld van een 3D-omgeving op het scherm toont. Het is een chip die een beeld van een 3D-omgeving op het scherm toont. Het is een chip die een beeld van een 3D-omgeving op het scherm toont.

Om de beste grafische kaarten wordt nu haast letterlijk gevocht.

[illegible]

De Intel-cip op de pc GPU of Central Processing Unit) bestaat uit twee tot zes processoren die simultaan kunnen werken. Een GPU heeft er duizenden. De chips kunnen ook anders schakelen. Er zijn computers met een heel kleine processor: De embedded. Het is een afgeleide op het andere. Micro, integreert alle belangrijke onderdelen van een computer op één chip. Het is een procesor met een klein stukje van het systeem.

"What was more," says Auerbach, "we were processing information not just for our own clients, but for other clients. In 2002, suddenly our machine data hit valuable AI-algorithms on GPU's and machine learning has become a thing."

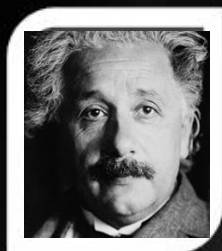
As it happens, it has also become a screaming, warlike task to accomplish. By AI world leaders, he'd seen this in the services such as Apple's Siri and Google's Assistant, in which he'd seen how they'd been built.

By 2010, he'd seen that of course,

Cryptomani
 en lussman is of nog een rieve
 korpsing omden. cryptomani
 word de blinde 'rijen'
 die blinde blinde word 're'

**The third pillar
of the scientific world:
computational science**

1



THEORY



2

EXPERIMENTATION



Informatica = tool

Jan Lemeire (jan.lemeire@vub.be)

- ◆ Burgerlijk ingenieur, 1994, VUB
 - + bijkomende masters in de computerwetenschappen (1995)
- ◆ Werkte 4 jaar in de privé, voor 2 IT-consultancybedrijven
- ◆ 2000-2007: doctoreerde aan de VUB als assistent
 - ✦ Gaf oefeningen informatica
- ◆ Sinds 2008: professor aan VUB
 - ✦ Vak 'parallel systems' in de masters
 - ✦ Sinds 2011: titularis 'Informatica' eerste bachelors
- ◆ Sinds oktober 2013: geeft ook les aan *industriële ingenieurs*
 - ✦ Bachelors: Informatica en Basiselektronica
 - ✦ Masters: Computerarchitectuur
- ◆ **Onderzoeksdomeinen:**
 - ✦ GPU computing
 - ✦ Robot vision

<http://parallel.vub.ac.be>

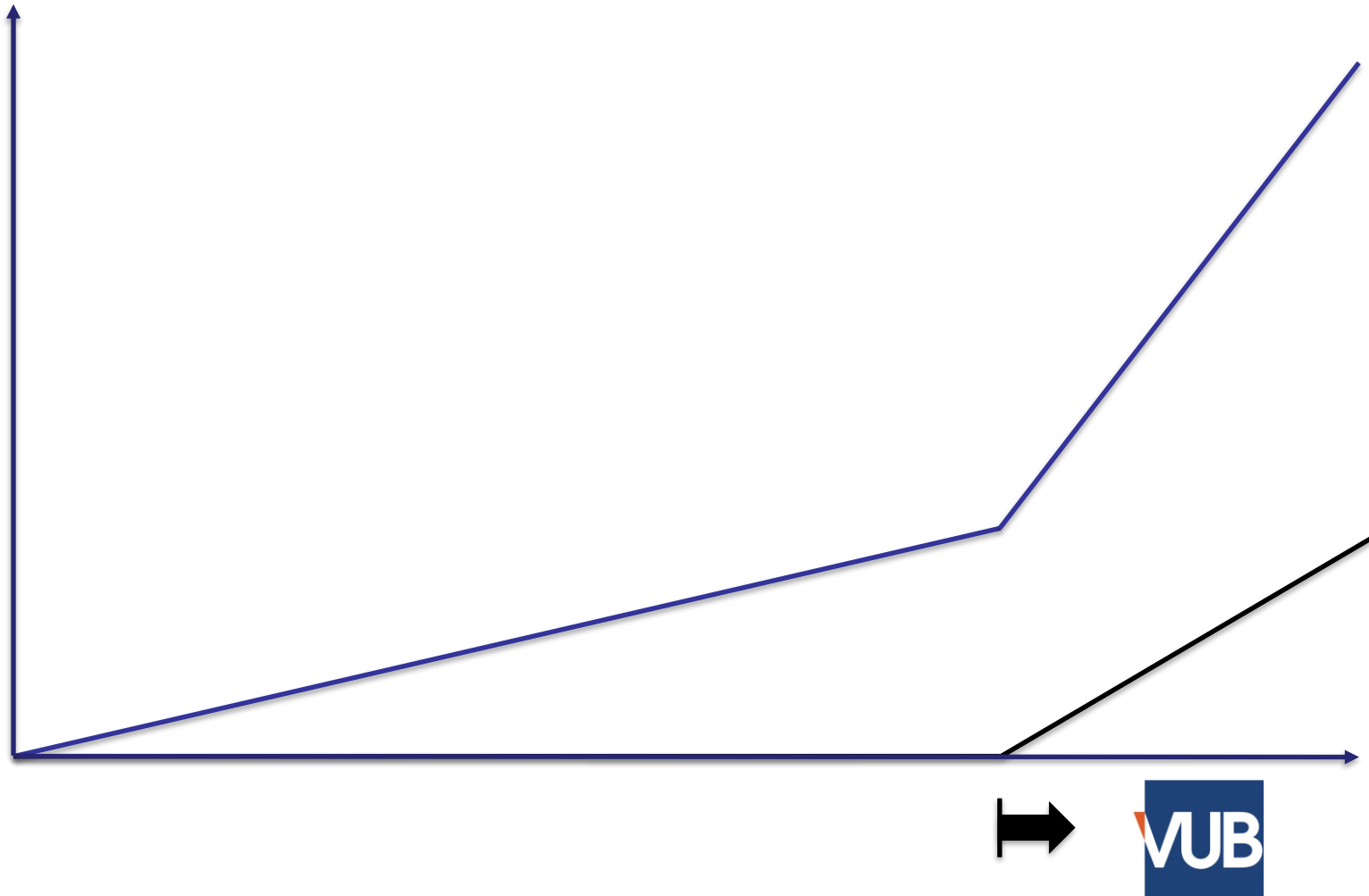


Eerste semester

Python: procedureel programmeren



Universiteit zonder voorkennis



Basis-
programmeervaardigheden

While – for
Lijsten/arrays
Functies
Recurisie

Java & Object-georiënteerd
programmeren

Regels
Organisatie van de code
Gebruik klassen

Datastructuren

Arrays – ArrayLists
Stack, Queue, Linked lists
Trees - Maps

CURSUS II

Algoritmen

Basisalgoritmen
Slimme algoritmen
Sorteren

Beheers het gebruik van bibliotheken:

- Programmeervaardigheden
- Klassestructuur
- Voorbeelden
- Documentatie

PROJECT

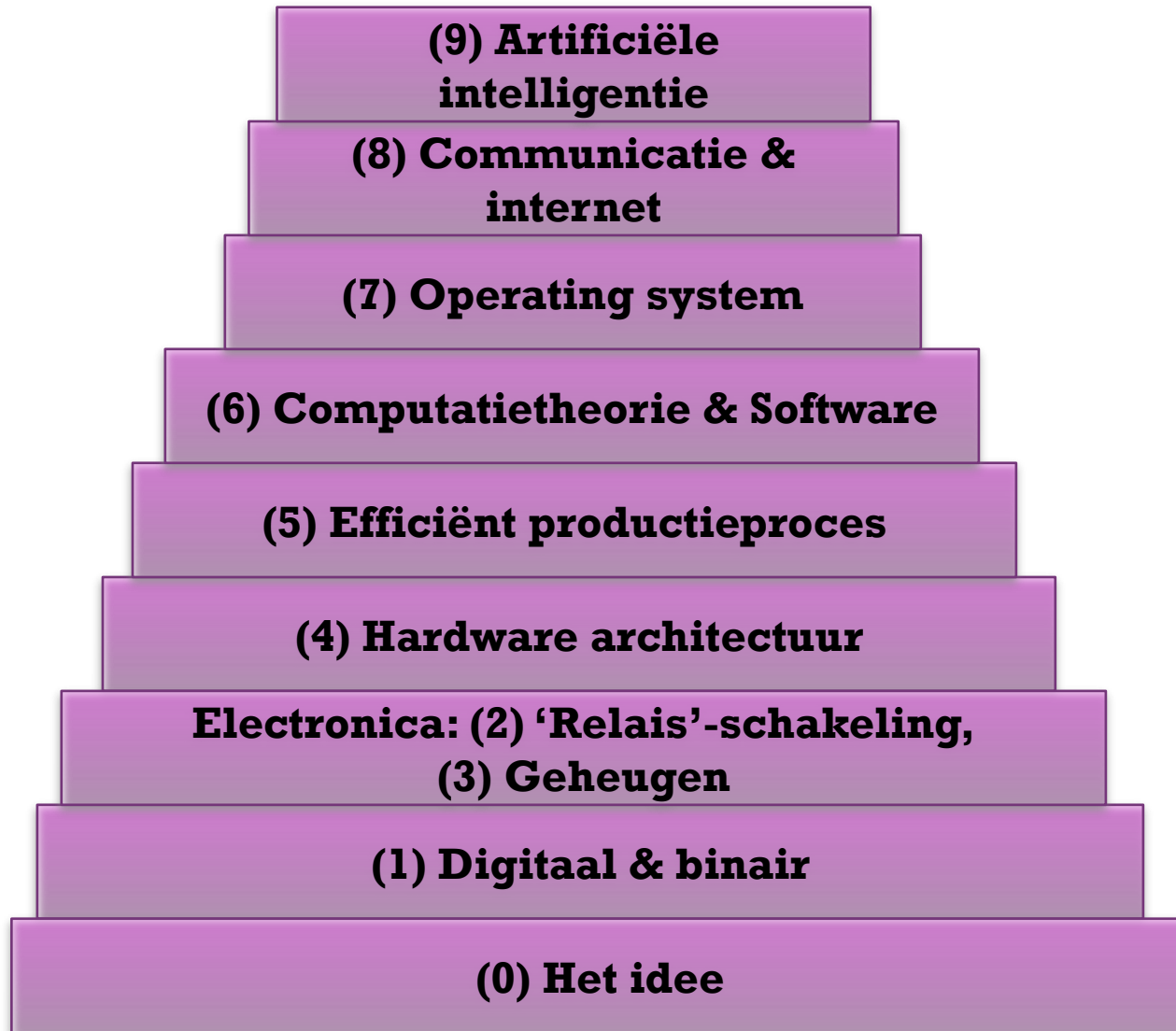
GUI-klassen

JFrames – Jpanels – Jbuttons - ...
Events – Eventlisteners - mouse
Paint, layout, JApplet

Utility-klassen

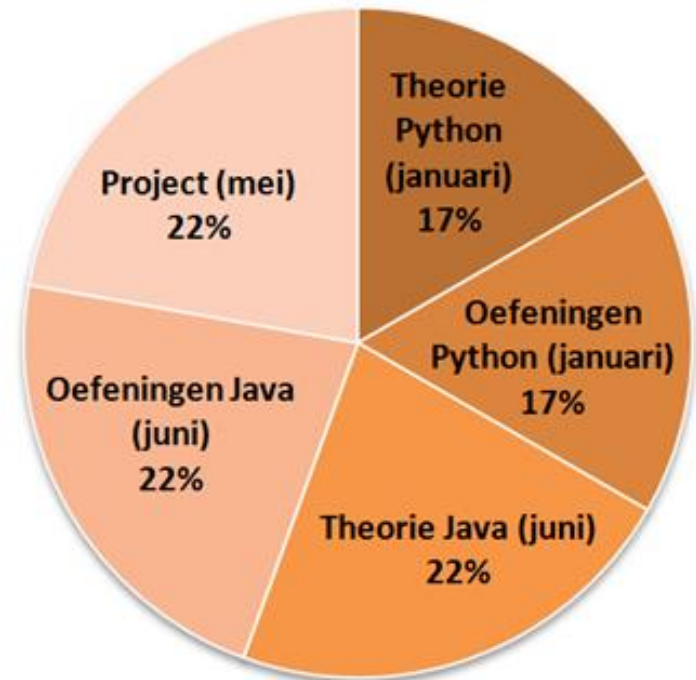
Timer, Thread, Random, File, ...

Informatica deel III: technologie, historiek en economische aspecten



2^e semester

- Cursustekst deel I
- Cursustekst deel II
 - Verschillen met vorig jaar: op website
- Cursustekst deel III
- Slides: alles staat in cursus
 - Indien niet: staat duidelijk aangegeven
- ***<http://parallel.vub.ac.be>***
Code & alle info



2e zit en vrijstellingen

Zie parallel.vub.ac.be

- ♦ Vanaf **12/20** heb je een (officieuze) vrijstelling voor één van de 5 onderdelen: Python theorie, Python oefeningen, Java project, Java theorie of Java oefeningen. De vrijstelling geldt van 1e naar 2e zit, **maar ook van 1 jaar op een ander**.
- ♦ Er wordt niet afgerond.
- ♦ In juni bieden we de mogelijkheid om facultatief je punten van het 1e semester te verbeteren, indien je een 7, 8 of 9 behaalde in het eerste semester (gemiddelde over theorie en oefeningen) en je slaagt voor je mondeling examen met minstens een 12/20.

Deel I

- ◆ 4 pagina's referentie java
- ◆ Java's spelregels: *van buiten kennen*
- ◆ Pijlers van object-georiënteerde programmeertalen

TIOBE Index for February 2019

Feb 2019	Feb 2018	Change	Programming Language	Ratings	Change
1	1		Java	15.876%	+0.89%
2	2		C	12.424%	+0.57%
3	4	⬆	Python	7.574%	+2.41%
4	3	⬇	C++	7.444%	+1.72%
5	6	⬆	Visual Basic .NET	7.095%	+3.02%
6	8	⬆	JavaScript	2.848%	-0.32%
7	5	⬇	C#	2.846%	-1.61%
8	7	⬇	PHP	2.271%	-1.15%
9	11	⬆	SQL	1.900%	-0.46%
10	20	⬆	Objective-C	1.447%	+0.32%
11	15	⬆	Assembly language	1.377%	-0.46%
12	19	⬆	MATLAB	1.196%	-0.03%
13	17	⬆	Perl	1.102%	-0.66%
14	9	⬇	Delphi/Object Pascal	1.066%	-1.52%
15	13	⬇	R	1.043%	-1.04%
16	10	⬇	Ruby	1.037%	-1.50%
17	12	⬇	Visual Basic	0.991%	-1.19%
18	18		Go	0.960%	-0.46%
19	49	⬆	Groovy	0.936%	+0.75%
20	16	⬇	Swift	0.918%	-0.88%



http://www.tiobe.com/tiobe_index



Java versus Python



p. 8

1. Object-georiënteerde taal.
2. `public static void main`
3. Puntkomma's en accolades
4. `System.out.println`
5. Typeren: sterk en statisch
6. Arrays en ArrayLists
7. Verschil tussen letters en woorden
8. De for-lus
9. Varia

Python2java

Hetzelfde in Python

```
'Functions for working with temperatures.'  
  
from math import sqrt  
  
def to_celsius(t):  
    'Convert Fahrenheit to Celsius.'  
    return (t - 32.0) * 5.0 / 9.0  
  
x = 100  
x = x / 3  
x = sqrt(x)  
  
y = to_celsius(x)  
  
print str(x)+" Fahrenheit = "+str(y)+" Celsius"
```

Java bestaat louter uit klassen

Statements eindigen met ';'

File zelfde naam als klasse

2 manieren om
commentaar te
geven

JavaVoorbeeld.java

// een klein voorbeeld met de belangrijkste java-elementen

import java.lang.Math;

public class JavaVoorbeeld {

/* PROGRAMMA */

public static void main(String[] args) {

// declaratie en initialisatie van x

int x = 100;

// enkele berekeningen

x = x / 3;

x = (int) Math.sqrt(x);

// omzetting naar Celsius

double y = toCelsius(x);

// printen

System.out.println(x+" Fahrenheit =" +y+"Celsius");

Printen van gegevens

/** Convert Fahrenheit to Celsius */

public static double toCelsius(double gradenFahrenheit){

return (gradenFahrenheit - 32) * 5 / 9;

}

}

Commentaar die gebruikt
wordt voor documentatieType van
teruggeefwaardeHet programma begint met de **main**.
De eerste lijn moet er zo uit zien.Functie van Math-klasse, een java-klasse met handige
mathematische functies.sqrt geeft een double terug, met (int) zet je om
(‘casting’) naar integer.

Oproep van zelfgemaakte functie.

Een functie duidt je aan met static

Parameter van methode

Rekenen met integers!

Python code	Waarde x	Java code	Waarde x
<code>x = 100</code>	100	<code>int x = 100;</code>	100
<code>x = x / 3</code>	33.33333 (a)	<code>x = x / 3;</code>	33 (b)
<code>x = sqrt(x)</code>	5.773502	<code>x = (int) Math.sqrt(x);</code>	5 (c)

- (a) Alhoewel x eerst als integer werd opgeslagen, wordt het type verandert naar float om zo de cijfers achter de komma weer te geven. (Enkel in Python 3, niet in Python 2)
- (b) In de meeste programmeertalen blijft x hetzelfde type behouden. Bij integerbewerkingen wordt alles na de komma weg gegooit.
- (c) De sqrt-functie neem een float/double als input. X wordt automatisch omgezet naar double (typeverruiming). De teruggeefwaarde van sqrt is een double, maar x is een integer. Omdat het hier over typeverarming gaat moet je expliciet aangeven dat je het type verandert en akkoord bent met verliezen van alle cijfers achter de komma.

Rekenen met integers!

Java code	Waarde x	
<code>int x = 100; x = x * 2 / 3;</code>	(a) $x = ?$	x = 66
<code>int x = 100; x = x * (2 / 3);</code>	(b) $x = ?$	x = 0
<code>int x = 100; x = 2 / 3 * x;</code>	(c) $x = ?$	x = 0
<code>double x = 2 / 3;</code>	(d) $x = ?$	x = 0.0
<code>double x = 2.0 / 3;</code>	(e) $x = ?$	x = 0.66667

- ◆ Integers: alles naar de komma verdwijnt!
- ◆ (d) de omzetting naar double gebeurt pas na de berekening!
(e) is een oplossing.

Wat berekent *GHI*?"



```
import java.util.Scanner;

public class GHI {

    /** PROGRAMMA */
    public static void main(String[] args) {

        System.out.println("Geef 2 getallen: ");

        Scanner scanner = new Scanner(System.in); // om input in te lezen
        int a = scanner.nextInt();
        int b = scanner.nextInt();

        int ghi = berekenGHI(a, b);
        System.out.println("Het resultaat na het ingeven van "+a+" en "+b+" is "+ghi);
    }

    public static int berekenGHI(int x, int y){
        while (x != y){
            if (x > y)
                x = x-y;
            else
                y = y-x;
        }
        return x;
    }
}
```

Hetzelfde in python

```
def berekenGHI(x, y):  
    while x != y:  
        if x > y:  
            x = x-y  
        else:  
            y = y-x  
    return x  
  
s = input("Geef 2 getallen: ")  
a = int(s)  
s = input()  
b = int(s)  
ggd = berekenGHI(a, b)  
print("GGD van", a, "en", b, "is", ggd)
```

Getypeerde parameters

```
public static int berekenGrootstGemeneDeler(int x, int y)
```

- ◆ Java: Enkel op te roepen met integers
- ◆ In python: ook reëel getal mogelijk! Elk type, dus ook string...
 - Parameters 4.5 & 5.5 werkt
 - Parameters 4.5 en 5.1 werkt het niet: gaat oneindig door, x en y worden nooit gelijk
=> NOK!

Type checking nuttig

- ◆ Je weet als gebruiker wat mee te geven
 - ✦ In python: input documenteren
- ◆ Geruststelling, je kan 'niets fouts' doen
 - ✦ Hier: negatieve waarden geeft fouten: checken
 - ✦ In Python: steeds extra checks doen op type
- ◆ Parameters kunnen ingewikkelde objecten zijn

Pythonvoorbeeld audio

```
sample_rate, signal = scipy.io.read("mymusic.wav")  
freq = scipy.fft(signal))
```

Inlezen van audio-file en omzetten naar frequentiespectrum.

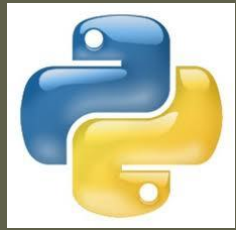
Gemakkelijk, maar hoe zien de gegevens er uit? Wat zijn de types? ***Niet duidelijk.***

Functie: maar 1 teruggeefwaarde

- ◆ Java, zoals de meeste programmeertalen
- ◆ Functie
 - ✦ kan maar 1 waarde teruggeven
 - ✦ Voorbeeld: celsius-conversie, ggd
- ◆ Indien meerdere waarden: 'truken' nodig



Voorlopige conclusie



Python

- ◆ Snel, voor kleine programma's
- ◆ High-level, handige basisfunctionaliteiten
 - ✦ **Cf Matlab**

Java

- ◆ Voor grote programma's en bibliotheken, te delen met andere programmeurs
- ◆ Iets meer low-level
- ◆ Meest-gebruikte industriële taal



Java versus Python



1. Object-georiënteerde taal.
2. `public static void main`
3. Puntkomma's en accolades
4. `System.out.println`
5. Typeren: sterk en statisch
6. Arrays en ArrayLists
7. Verschil tussen letters en woorden
8. De for-lus
9. Varia

Doelstellingen object-georiënteerde talen

- ◆ Een project zo structureel mogelijk opbouwen met objecten
- ◆ Code opdelen in logische componenten, modulariteit
- ◆ Code makkelijker te **hergebruiken** en **uit te breiden**

Objecten

Persoon

<i>voornaam</i>	<i>naam</i>
<input type="text" value="Rik"/>	<input type="text" value="Vermeulen"/>
<i>emailadres</i>	
<input type="text" value="rikiki@gmail.com"/>	

Persoon

<i>voornaam</i>	<i>naam</i>
<input type="text" value="Jana"/>	<input type="text" value="Laplace"/>
<i>emailadres</i>	
<input type="text"/>	

Persoon

<i>voornaam</i>	<i>naam</i>
<input type="text" value="Victoria"/>	<input type="text" value="Lemeire"/>
<i>emailadres</i>	
<input type="text"/>	

```
public class Persoon {
```

```
//===== ATTRIBUTEN =====//
```

```
String voornaam, naam;
```

```
String emailadres;
```

```
//===== CONSTRUCTORS =====//
```

```
Persoon(String voornaam, String naam){
```

```
    this.voornaam = voornaam;
```

```
    this.naam = naam;
```

```
}
```

```
Persoon(String voornaam, String naam, String emailadres){
```

```
    this.voornaam = voornaam;
```

```
    this.naam = naam;
```

```
    this.emailadres = emailadres;
```

```
}
```

```
//===== METHODES =====//
```

```
public void maakDefaultEmailadres(String domeinVanProvider){
```

```
    emailadres = voornaam+"."+naam+"@"+domeinVanProvider;
```

```
}
```

```
public String toString(){
```

```
    return voornaam+" "+naam+(emailadres==null?"": " ["+emailadres+"]");
```

```
}
```

```
}
```

Ternary operator: ... ? ... : ...

Persoonobjecten

```
public class PersoonMain {  
  
    /** PROGRAMMA */  
    public static void main(String[] args) {  
  
        Persoon rik = new Persoon("Rik", "Vermeulen", "rikiki@gmail.com");  
        Persoon jana = new Persoon("Jana", "Laplace");  
        Persoon victoria = new Persoon("Victoria", "Lemeire");  
  
        victoria.voornaam = "Vicky";  
        jana.maakDefaultEmailadres("hotmail.com");  
        System.out.println("Adres aangemaakt: "+jana.emailadres);  
  
        System.out.println("Rik: "+rik);  
        System.out.println("Jana: "+jana);  
        System.out.println("Victoria: "+victoria);  
    }  
}
```

Objecten

- ◆ Gedefinieerd met een **klasse** (*type object*)
- ◆ Heeft **attributen**
 - ✦ De eigenschappen van het object
- ◆ Aangemaakt met een **constructor**
 - ✦ Object is *instantiatie* van een welbepaalde klasse
- ◆ Heeft **methodes**
 - ✦ Methode heeft toegang tot de attributen, kan deze veranderen

Overerving

Een student is een persoon

```
public class Student extends Persoon{
    enum Faculteit {IR, WE, GF, LK, LW, ES, RC, PE};

    int rolnummer;
    Faculteit faculteit = Faculteit.IR; // default waarde voor
    alle nieuwe objecten

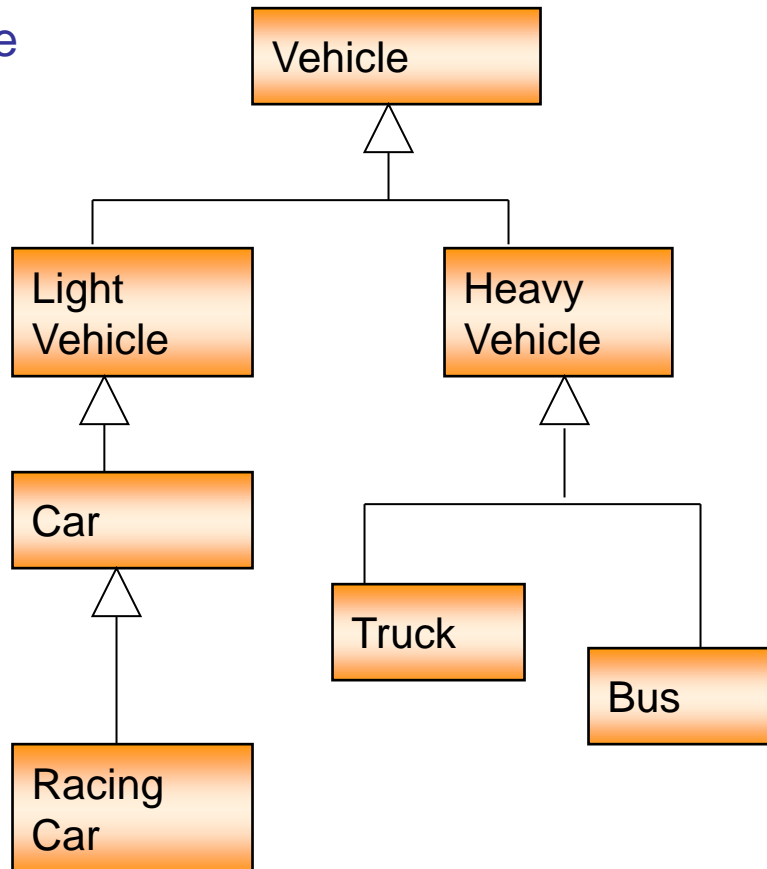
    Student(String voornaam, String naam, int
    rolnummer) {
        super(voornaam, naam);
        this.rolnummer=rolnummer;
    }
}
```

Overerving (*inheritance*)

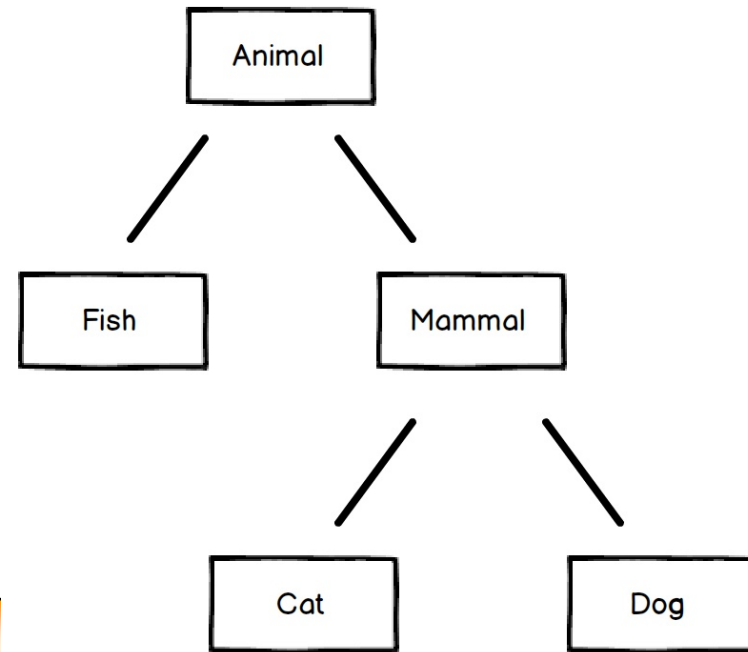
- ◆ Subklasse erft alle attributen en methodes over van moederklasse
- ◆ Subklasse kan attributen toevoegen
- ◆ Subklasse kan methodes toevoegen of methodes *overschrijven*
- ◆ Constructor van subklasse moet een constructor van moederklasse oproepen (superconstructor)

Overerving leidt tot een klassehiërarchie

Generalizatie



Specializatie




Pijlers van object-georiënteerde programmeertalen

p. 7

I. Encapsulatie

- 2.3 ArrayList p. 12
- 3.1 Stapel-datastructuur p. 13
- 6.2 Java's LinkedList p. 56

II. Overerving (inheritance)

- 1.1.3 Studentvoorbeeld p. 20
 - 1.3 Vriendenvoorbeeld p. 36
 - 1.4.1 MyPanel p. 41
 - 1.4.3 PainComponent overschrijven p. 44
 - 4.3 FunctieMetAfgeleide-interface p. 25
- 

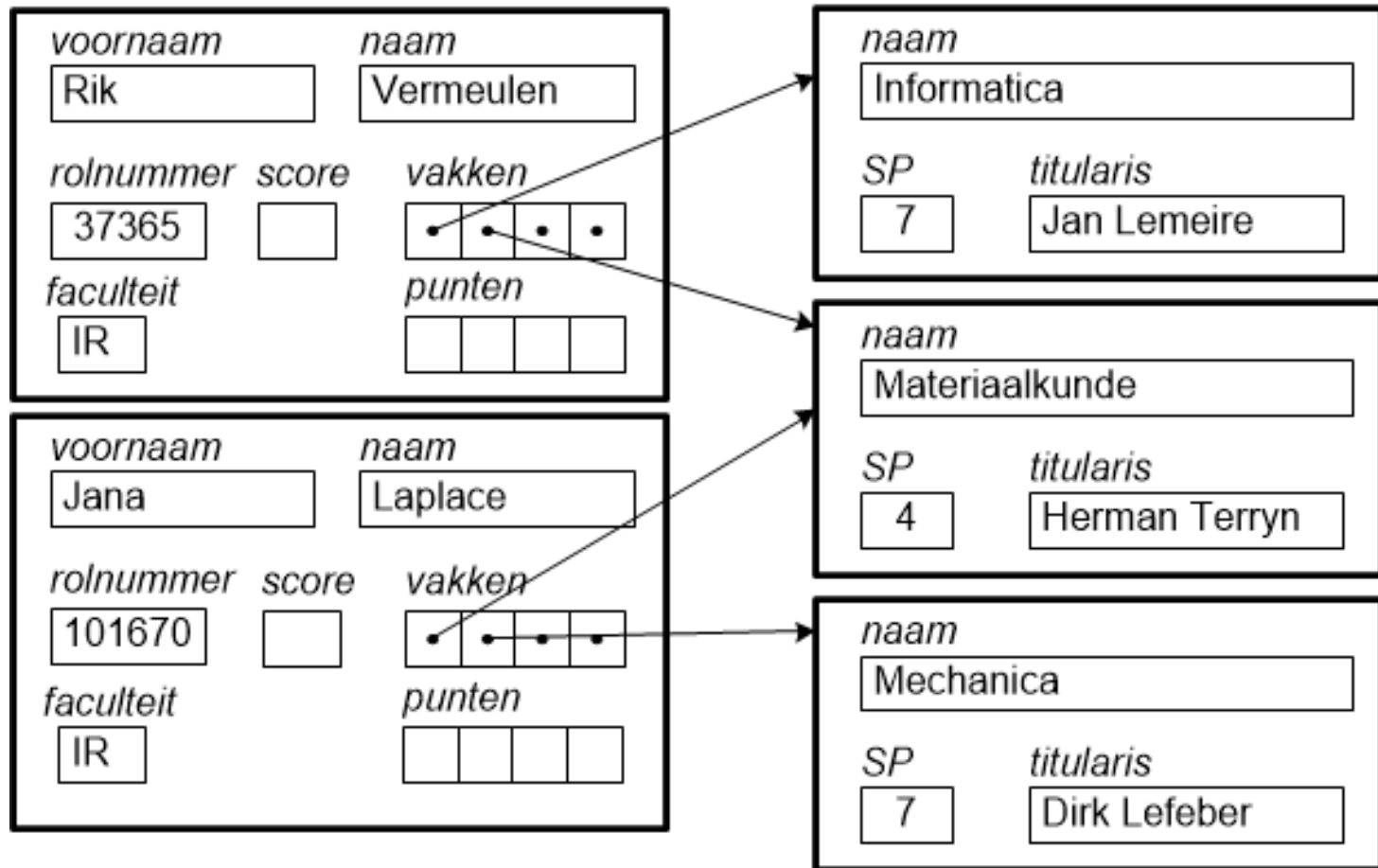
III. Polymorfisme en abstractie

- 1.2.4 Set p. 31
- 1.2.5 Map p. 33
- 1.4.2 EventListener p. 43
- 1.6.4 Abstracte klassen p. 58
- 4.2 Functie-interface p. 22
- 5.2.2 Backtracking & Breadth-first p. 32
- Addendum bij hoofdstuk 5 (zie website, is optioneel)
 - Abstract zoekalgoritme
 - Vergelijking van zoekalgoritmes

Super-constructor

- ◆ Het aanmaken van een Student-object vergt ook het aanmaken van een Persoon-object
- ◆ Op de eerste lijn van de Student-constructor roep je een constructor van Persoon op met `super(voornaam, naam);`

Objectattributen



Persoon

String voornaam, naam;
String emailadres;

void maakDefaultEmailadres(String
domeinVanProvider);

PersoonMetVriend

PersoonMetVriend vriend;

boolean kenJeDiePersoonViaVia
(PersoonMetVriend persoon)

Student

Faculteit faculteit;

int rolnummer, score;

ArrayList<Vak> **vakken**;

ArrayList<Integer> **punten**;

float score;

int berekenTotaalScore()

void voegVakToe(Vak vak, **int** score)

Ontkenner

boolean kenJeDiePersoonViaVia(persoon)

Leugenaar

boolean kenJeDiePersoonViaVia(persoon)

```
public class Student extends Persoon
{
    enum Faculteit {IR, WE, GF, LK, LW, ES, RC, PE};

    int rolnummer;
    Faculteit faculteit = Faculteit.IR; // default waarde voor alle nieuwe objecten
    ArrayList<Vak> vakken;
    Map<Vak, Integer> punten;
    float score;

    Student(String voornaam, String naam, int rolnummer, Faculteit fac){
        super(voornaam, naam);
        this.rolnummer=rolnummer;
        this.faculteit = fac;
        vakken = new ArrayList<Vak>();
        punten = new HashMap<Vak, Integer>();
    }
}

public class Vak {
    String naam, titularis;
    int SP;
    Vak(String naam, String titularis, int SP){
        this.naam = naam;
        this.titularis = titularis;
        this.SP = SP;
    }
}
```

```

public static void main(String[] args) {
    Student rik = new Student("Rik", "Vermeulen", 37365, Faculteit.IR);
    Student jana = new Student("Jana", "Laplace", 101670, Faculteit.WE);

    Vak informatica = new Vak("Informatica", "Jan Lemeire", 7);
    Vak materiaalkunde = new Vak("Materiaalkunde", "Herman Terryn", 4);
    Vak mechanica = new Vak("Mechanica", "Dirk Lefeber", 7);

    rik.vakken.add(informatica);
    rik.punten.put(informatica, 14);
    rik.vakken.add(materiaalkunde);
    rik.punten.put(materiaalkunde, 12);
    rik.vakken.add(mechanica);
    rik.punten.put(mechanica, 17);

    jana.vakken.add(informatica);
    jana.punten.put(informatica, 16);
    jana.vakken.add(mechanica);
    jana.punten.put(mechanica, 13);
}

```