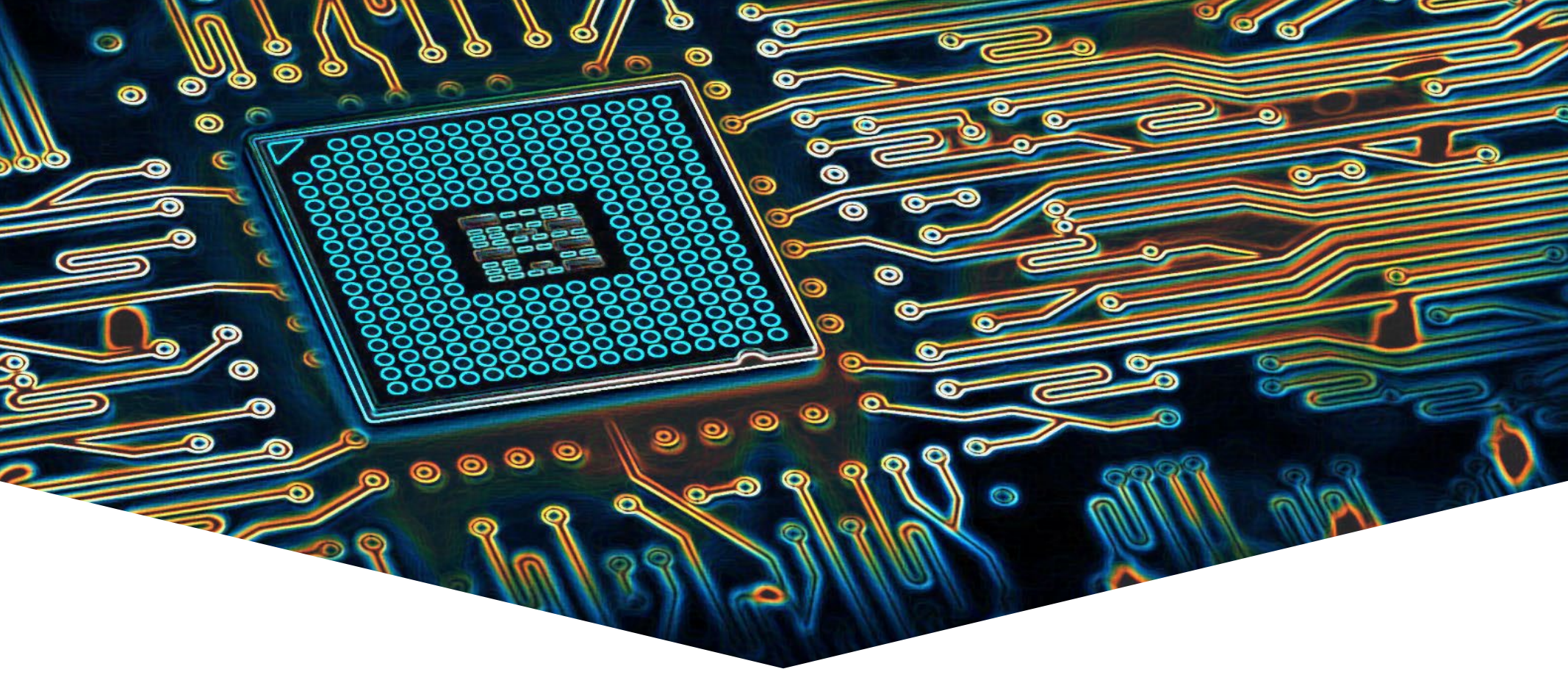


EMS31 Introductie kwartaal 4

Leerdoelen EMS31

Leerdoelen EMS31.

Je leert in kwartaal **3**:

- hoe je een betere **C**-programmeur wordt.

Je leert in kwartaal **4**:

- de basisbeginselen van objectgeoriënteerd programmeren in **C++**;
- de basisbeginselen van objectgeoriënteerd ontwerpen met **UML**.

Leerdoelen EMS31 kwartaal 4

#	Niveau	Weging	De student is in staat om …
4	C	30 %	… gebruik te maken van objectgeoriënteerde programmeertechnieken in C++ zoals encapsulation, templates, overerving en polymorfisme zodanig dat de student programmatuur kan ontwikkelen die uitbreidbaar, aanpasbaar en herbruikbaar is.
5	C	10 %	… op grond van zijn of haar kennis van de big-O-notatie effectief gebruik te maken van de standaard in C++ aanwezige datastructuren en algoritmen.
6	C	10 %	… een objectgeoriënteerd ontwerp te maken van een programma in UML door middel van usecase-, klassen-, sequentie-, toestand- en activiteitendiagrammen.

Inhoud

Objectgeoriënteerd Programmeren in **C++**.

- responsibility driven design (ontwerpen uitgaande van verantwoordelijkheden).
- information hiding (het afschermen van informatie door middel van het scheiden van interface en implementatie).
- abstraction (het afschermen van complexiteit door middel van het scheiden van interface en implementatie).
- inheritance (het mogelijk maken van een vorm van hergebruik, ... is een ... in plaats van ... heeft een ...).
- polymorphism (verschijningsvorm mogelijk gemaakt door dynamic binding).

Objectgeoriënteerd Ontwerpen met **UML**.

- klasse- en objectdiagram.
- use-case diagram.
- sequence- en communicatiediagram.
- toestands- en activiteitendiagram.

Herbruikbaarheid

Aanpasbaarheid

Uitbreidbaarheid

Toetsing en studielast EMS31

Toets	Leerdoelen	Weging	Deadline
Opdracht 1	1, 2 en 3	50 %	Lesweek T3, zondag 23.59 uur
Opdracht 2	4, 5 en 6	50 %	Lesweek T4, zondag 23.59 uur

Je werkt in **tweetallen** samen m.b.v. git (*je krijgt van ons een repository*) en levert je werk ook in op dat repository.

EMS31 levert je **5** studiepunten op dat betekent dat je 5/30 = 1/6 van de week
→ **6.5 uren / week** aan EMS31 dient te besteden.

Planning kwartaal 4

Week	Werkvorm	Beschrijving
4.1	Les	Van C naar C++
	Zelfstudie	Zie paragraaf 2
4.2	Les	User-defined Data Type in C++
	Zelfstudie	Zie paragraaf 2
4.3	Les	Templates
	Zelfstudie	Zie paragraaf 2
4.4	Les	Overerving en polymorfisme
	Zelfstudie	Zie paragraaf 2
4.5	Les	Datastructuren
	Zelfstudie	Zie paragraaf 2
4.6	Les	Algoritmen, performance en big-O-notatie
	Zelfstudie	Zie paragraaf 2
4.7	Les	UML use cases en klassediagram
	Zelfstudie	Werken aan opdracht 2
4.8	Les	UML sequentie-, toestands- en activiteitendiagram
	Zelfstudie	Werken aan opdracht 2
T4	**Opdracht 2**	Deadline zondag 23.59 uur
HT4	**Herkansing opdracht 1 en 2**	Deadline zondag 23.59 uur

Studiemateriaal vind je op de wiki:

HR_ELEKTRO / Embedded / EMS31

Wiki

Create page

Clone wiki

EMS31 / Home

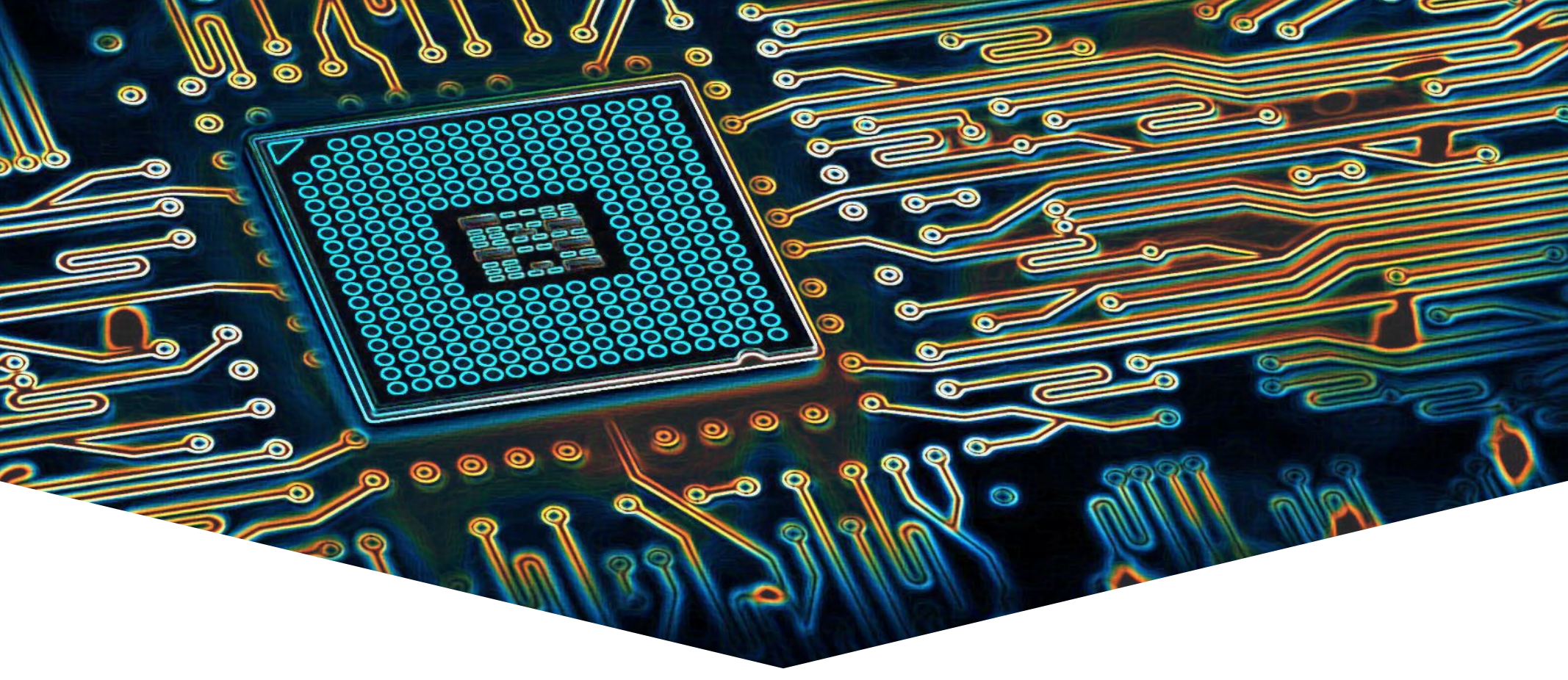
View | History | Edit

EMS31 - Embedded Systems 3

Dit repository is bedoeld voor studenten en docenten van de opleiding Elektrotechniek van de Hogeschool Rotterdam en wordt gebruikt het studiemateriaal voor de cursus "EMS31 - Embedded Systems 3" te verspreiden.

Let op! Deze wiki is nog niet volledig voor studiejaar 2023-2024.

De informatie in dit repository is zoals alle mensenwerk niet foutloos, verbeteringen en suggesties zijn altijd welkom! Maak als je ons feedback wilt geven een issue aan.



EMS31 Kwartaal 4 Week 1: Van C naar C++

Van C naar C++

Leerdoelen kwartaal 4 week 1.

In het **theoriedeel** leer je wat:

- het verschil is tussen **objectgeoriënteerd** programmeren in **C++** in plaats van **gestructureerd** programmeren in C.

In het **practicumdeel** leer je hoe:

- in C++ iets naar een **output**device (bijvoorbeeld je beeldscherm) kunt sturen en iets van een **input**device (bijvoorbeeld je toetsenbord) kunt inlezen;
- in C++ standaard classes (bijvoorbeeld **string**) kunt gebruiken.

Een stapje verder...

...met programmeren en ontwerpen.

- Van gestructureerd naar objectgeoriënteerd.
- C++ is een uitbreiding op C:
  - alles wat in C kan, kan ook in C++.
  - veel wat in C kan, kan in C++ beter (I/O, struct, array, c-string enz).

C is het fundament
voor C++.
Zie inleiding dictaat.

Bron: https://www.facebook.com/pg/CleverKidsEducare/photos

Een stapje verder...

...met programmeren en ontwerpen.

Waarom?

Dat is een lang verhaal...

Gestructureerde programmeertalen:

- ±1945 assembler, ±1957 FORTRAN, ±1960 Algol60, ±1972 C (1989 std ANSI C)

Software crisis:

- Software niet op tijd geleverd.
- Software duurder dan afgesproken.
- Software niet foutloos.

Idee voor de oplossing:

- Herbruikbare software componenten maken.
- Deze componenten gebruiken bij maken van grote uitbreidbare en onderhoudbare software systemen.

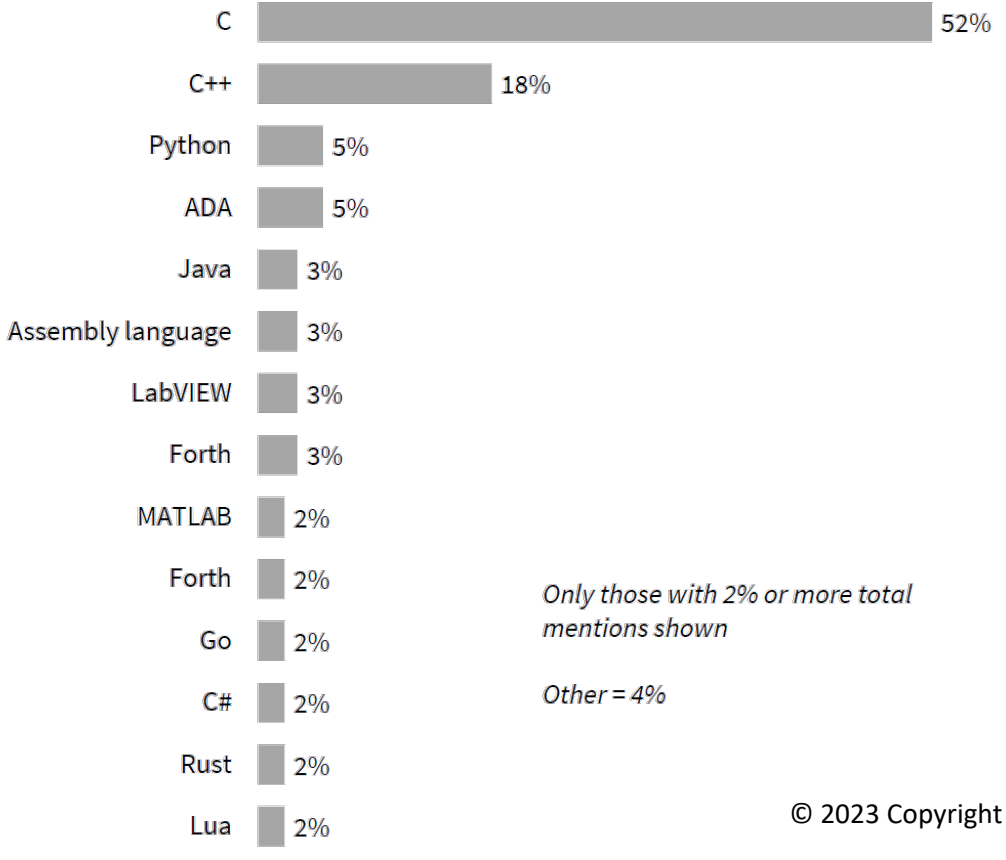
Bron: https://www.publicdomainpictures.net/nl/view-image.php?image=6884&picture=kind-met-laptop

Objectgeoriënteerde programmeertalen:

- ±1967 Simula, ±1976 Smalltalk, ±1983 C++ (1998 std C++), ±1995 Java (Sun/Oracle), ±2000 C# (Microsoft), ±2014 Swift (Apple).

… maar ook een kort verhaal

“C” dominates other languages for embedded software programming



Only those with 2% or more total mentions shown

Other = 4%

Inleiding C++

Bjarne Stroustrup:

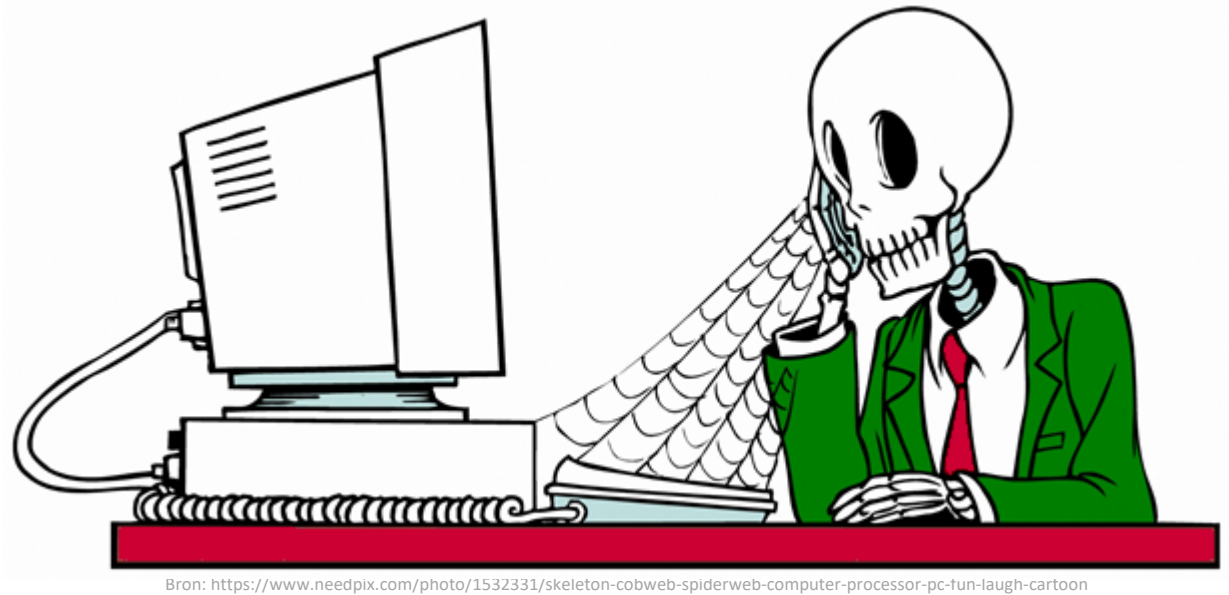
“C++ is a programming language that:

- is a better C,
- supports data abstraction,
- supports object-oriented programming,
- and supports generic programming.”

Bron: https://www.waterstechnology.com/people/4005396/coding-in-the-trenches-bjarne-stroustrup-morgan-stanley

Vraag

Welke software hoeft **nooit** uitgebreid of veranderd te worden?



Bron: https://www.needpix.com/photo/1532331/skeleton-cobweb-spiderweb-computer-processor-pc-fun-laugh-cartoon

Software die niemand (meer) gebruikt.

Nieuw!

Objectgeoriënteerd programmeren is een nieuwe manier van denken ...

... over hoe we code en data in een computerprogramma kunnen structureren.

Programmeer paradigma's:

- **imperative** (C, Pascal).
- functional (LISP, Haskell).
- logic (Prolog).
- **object oriented** (C++, Java, C#).
- **generic** (ADA, C++).

Objectgeoriënteerd Denken

De manier van probleem oplossen die gebruikt wordt bij de objectgeoriënteerde programmeertalen en ontwerpmethoden lijkt vaak op de manier van probleem oplossen die mensen in het dagelijks leven ook gebruiken.

Voorbeeld

Ik wil mijn oma een bosje bloemen sturen.

http://www.pngall.com/?p=33394

- object
- message + arguments
- receiver's responsibility
- method (information hiding)

Vraag

Wat is het verschil tussen een message en een functie?

- Een message heeft een bepaalde receiver.
- De method die bij de message hoort is afhankelijk van de receiver.
- De receiver van een message kan ook tijdens run-time worden bepaald.

  Dynamic binding between the message (function name) and method (code).

Vraag

Waarom weet ik zoveel van mijn bloemiste?

Bron: https://pages.rediff.com/photos/preview/phoolwala/5987/4/florist_girl

- class
- instance (object)
- hierarchy
- inheritance (base and derived)

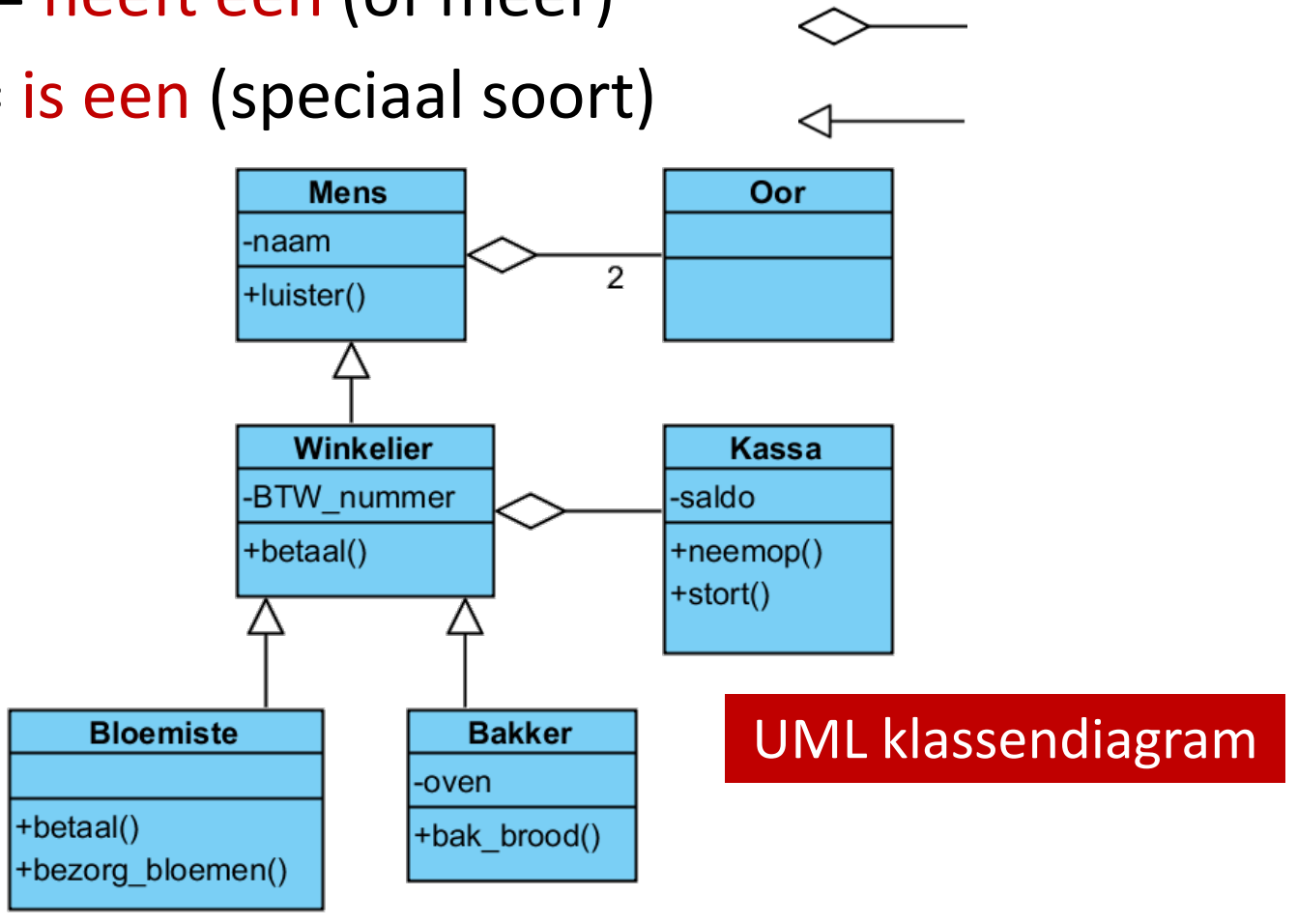
Method binding

- Zoek in class van het receiver object.
- Als daar geen method is zoek dan in de base class van de class van het receiver object.
- Als daar geen method is zoek dan in de base class van de base class van de class van het receiver object.
- Enzovoort.

Een method uit de base class kan **overriden** worden door een method in een derived class.

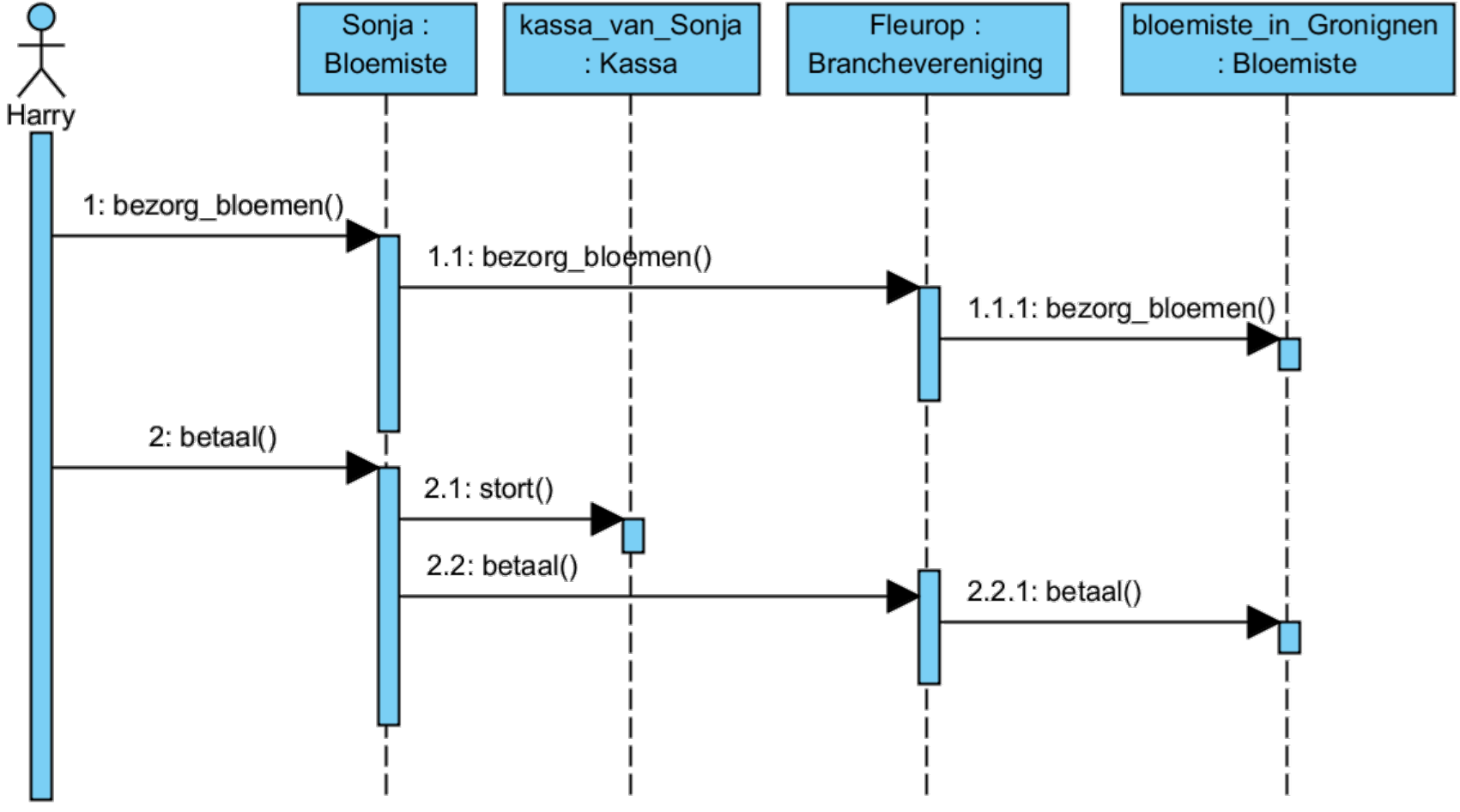
Verband tussen classes

- **Aggregation** = heeft een (of meer)
- **Inheritance** = is een (speciaal soort)



UML klassendiagram

UML sequentiediagram



Steeds meer abstractie

Functions.

- Avoid duplicating code.
- Information hiding.

Modules.

- Data and information hiding.

User-defined data types (UDT's).

- Instantiation.

Generic functions en generic UDT's.

- Templates.

Classes.

- Messages.
- Inheritance.
- Polymorphism.

Geschiedenis van programmeertechnieken

Doel van object oriëntatie

- Construeren van herbruikbare software componenten.
- Gebruiken van deze componenten bij het construeren van grote aanpasbare en uitbreidbare systemen.

Software IC

Huiswerk:
Bestudeer hoofdstuk 2 t/m 2.1
van het dictaat.

Bron: https://upload.wikimedia.org/wikipedia/commons/b/b2/TEXAS_INSTRUMENTS_TI_IC_Integrated_Circuits.jpg

Volgende les…

Definiëren van een UDT in C++ d.m.v. een `class`.

Aan de slag!

Aan de slag met [Opdrachten_Week_4.1.pdf]

https://www.modekoninginmaxima.nl/mode-maxima-mode-algemeen/nieuwjaarsontvangsten-2020-dag-1