

# Maak je eigen programmeertaal

**YOP = [Y]our [O]wn [Programming] language**

**7 maart 2023**

**Jurgen Vinju, Tijs van der Storm**



cwi-gast  
gastheer\_CWI

**Riemer van Rozen, Paul Klint, Koen Groenland (Q),  
Amber Moet, Marije Tolsma, Emil Gorter, Carla van Asperen, Robin Augustijn**

# Dagindeling

## L120, L016, Forum

Tijd	Items		Extra
10:00-10:50	Introducties ( <b>L120</b> )	Vertaalspel	Installatie Rascal
10:50-11:05	Pauze		
11:05-11:35	Quantum Quest ( <b>L016</b> ) + Rascal Practicum ( <b>L120</b> )	Rascal Practicum ( <b>L120</b> )	
11:45-12:15	Rascal Practicum ( <b>L120</b> )	Quantum Quest ( <b>L016</b> )	
12:15-13:00	Pauze	Lunch	
12:15-15:00	<b>Stap-voor-stap YOP Hackathon (<b>L120</b>)</b>	Snacks, Demo, Drankjes	Prijsuitreiking

# Introducties



Centrum Wiskunde & Informatica

<http://www.cwi.nl>

[75 jaar CWI](#)

[Constance van Eeden](#)

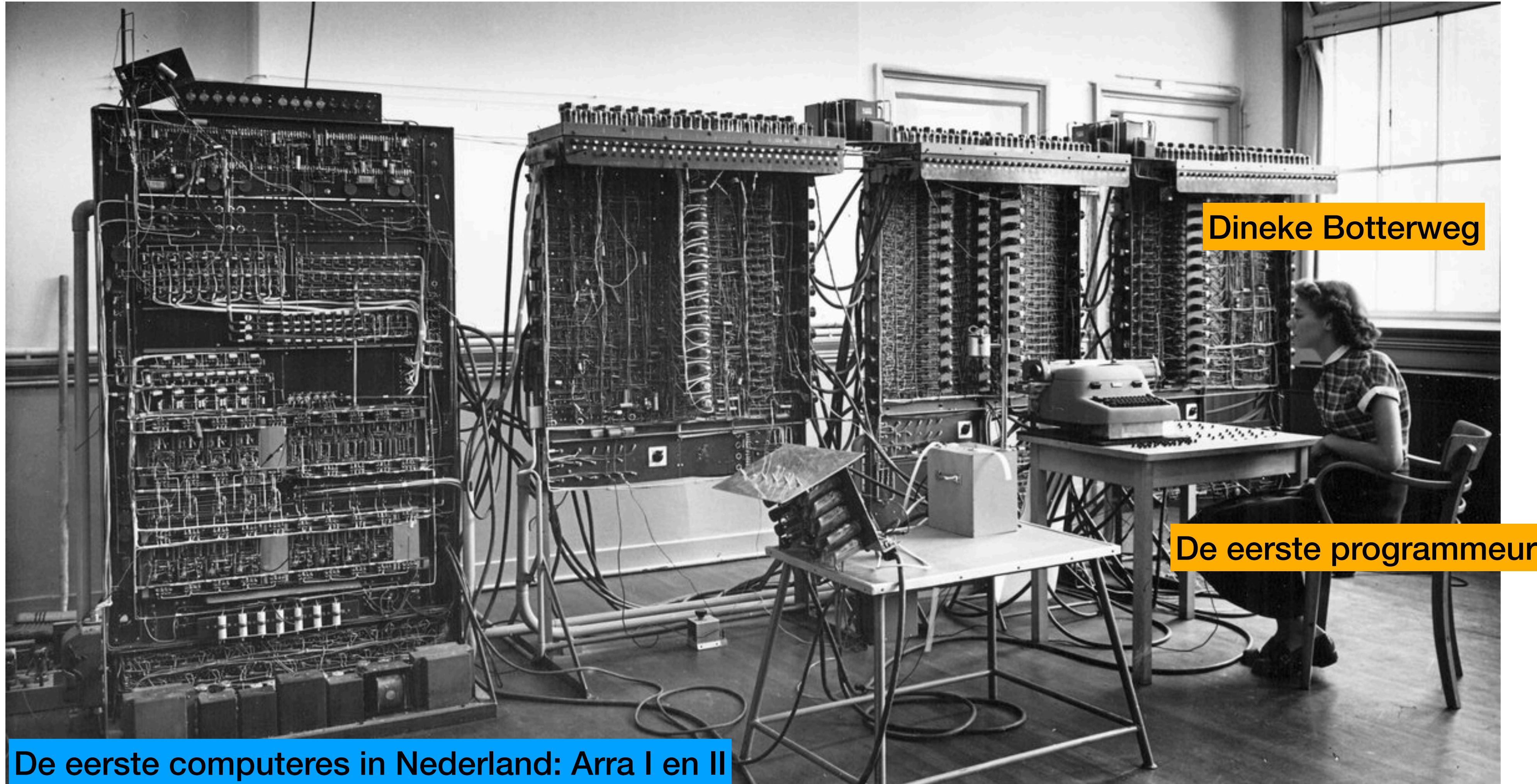


rascal

<http://www.rascal-mpl.org>

# Het Mathematisch Centrum

sinds 1946



@ van Wijngaarden

# Het internet in Europa

## Piet Beertema en zijn 1 april grap



**From chernenko@kremvax.UUCP Sun Apr 1 15:02:52 1984**

**"We have been informed that on this network many people have given strong anti-Russian opinions, but we believe they have been misguided by their leaders, especially the American administration, who is seeking for war and domination of the world."**

**From: chernenko@kremvax.UUCP**

From: Step  
Sent: Thursday, November 17, 1988 8:28 AM  
To: HOSTMASTER@SRI-NIC.ARPA; rick@seismo.CSS.GOV  
Subject: Re: [HOSTMASTER@SRI-NIC.ARPA: Re: mcvax internet connection]

> Thanks for the additional information re: CWI-ETHER, net  
> #192.16.184.  
> This is to let you know that we have changed the status of this  
> network to connected.

Sue - Thanks!

Rick - Go!

-s

Een idee van Piet:

.nl  
.fr  
.uk  
.de  
.?

From chernenko@kremvax.UUCP Sun Apr 1 15:02:52 1984  
Relay-Version: version B 2.10.1 6/24/83 (MC840302); site mcvax.UUCP  
Posting-Version: version B 2.10.1 4/1/84 (SU840401); site kremvax.UUCP  
Path: mcvax!moskvax!kremvax!chernenko  
From: chernenko@kremvax.UUCP  
Newsgroups: net.general,eunet.general,net.politics,eunet.politics  
Subject: USSR on Usenet  
Message-ID: <0001@kremvax.UUCP>  
Date: Sun, 1-Apr-84 15:02:52 GMT  
Article-I.D.: kremvax.0001  
Posted: Sun Apr 1 15:02:52 1984  
Date-Received: Mon, 1-Apr-84 12:26:02 GMT  
Organization: MIIA, Moscow  
Lines: 41

<.....>

Well, today, 840401, this is at last the Socialist Union of Soviet Republics joining the Usenet network and saying hallo to everybody.

One reason for us to join this network has been to have a means of having an open discussion forum with the American and European people and making clear to them our strong efforts towards attaining peaceful coexistence between the people of the Soviet Union and those of the United States and Europe.

work many people have given strong  
they have been misguided by their  
stration, who is seeking for war

side we hope to have a possibility  
and ideas.

believe in the truth of what we  
is network; to them we are very  
ly give your comments and opinions.

e:

International Affairs

Contact: K. CHERENKO  
Phone: +7 095 840401  
Postal-Address: Moscow, Soviet Union  
Electronic-Address: mcvax!moskvax!kremvax!chernenko  
News: mcvax kremvax kgbvax  
Mail: mcvax kremvax kgbvax

And now, let's open a flask of Vodka and have a drink on our entry on  
this network. So:

NA ZDAROVJE!

--  
K. Chernenko, Moscow, USSR  
...{decvax,philabs}!mcvax!moskvax!kremvax!chernenko

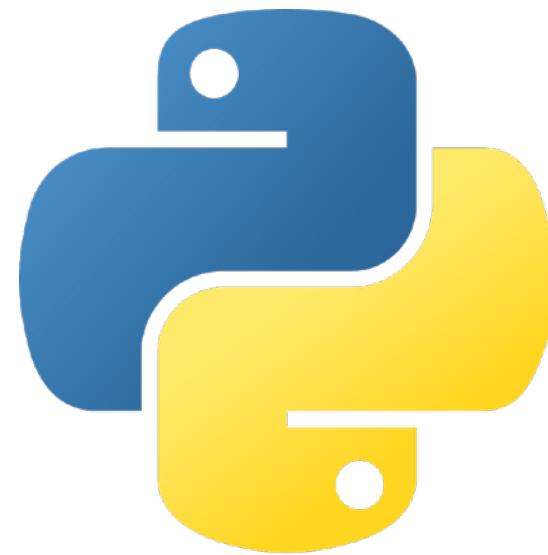
# Python uitgevonden

## Guido van Rossum

De rebelse uitvinder



1991 Amoeba @ CWI groep  
legt zich toe op de volgende generatie Algol-achtige  
programmeertalen, serieus, “correct”, “netjes”



1991 Guido @ CWI bouwt een “scripting” taal  
dynamisch, flexibel, creatief, uitbreidbaar



2020 Felienne Hermans (in Leiden) maakt “Hedy”  
leerbaar, graduateel, fun



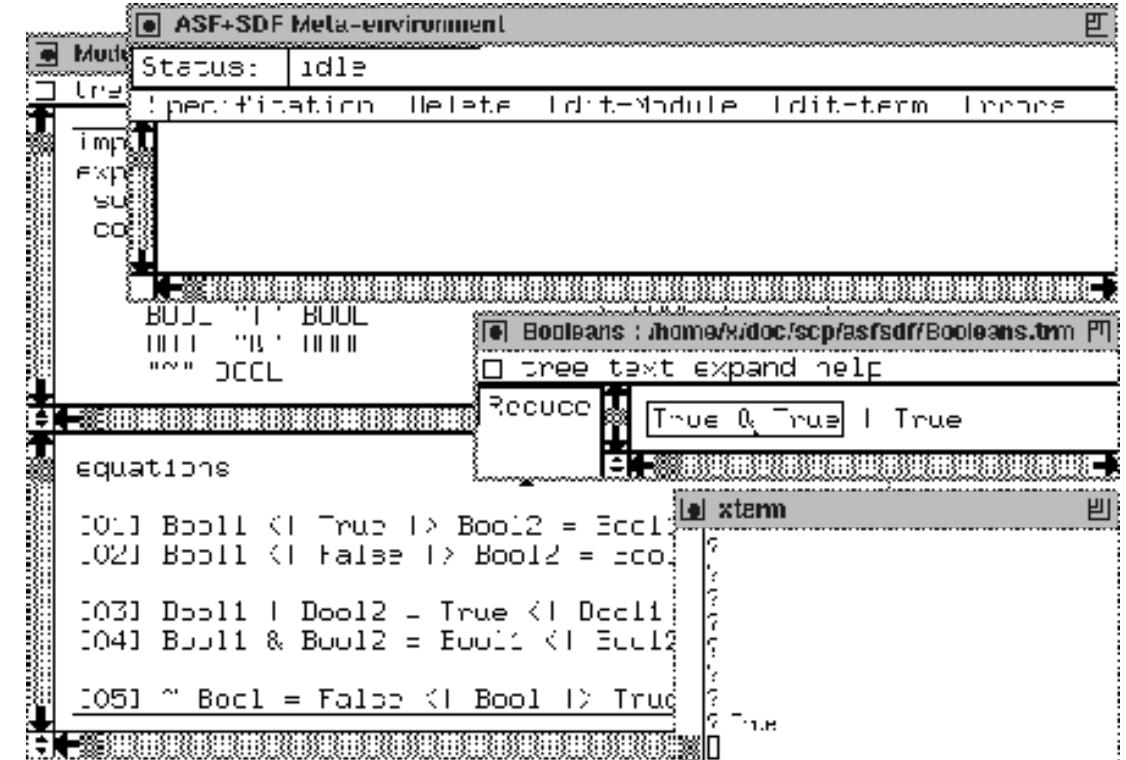
Oct 2021	Oct 2020	Change	Programming Language	Ratings	Change
1	3	▲	Python	11.27%	-0.00%
2	1	▼	C	11.16%	-5.79%
3	2	▼	Java	10.46%	-2.11%
4	4		C++	7.50%	+0.57%
5	5		C#	5.26%	+1.10%
6	6		Visual Basic	5.24%	+1.27%
7	7		JavaScript	2.19%	+0.05%
8	10	▲	SQL	2.17%	+0.61%
9	8	▼	PHP	2.10%	+0.01%
10	17	▲	Assembly language	2.06%	+0.99%

Tiobe Index 2022: Python meest populaire taal  
in de hele wereld!

# Rascal Metaprogrammeertaal

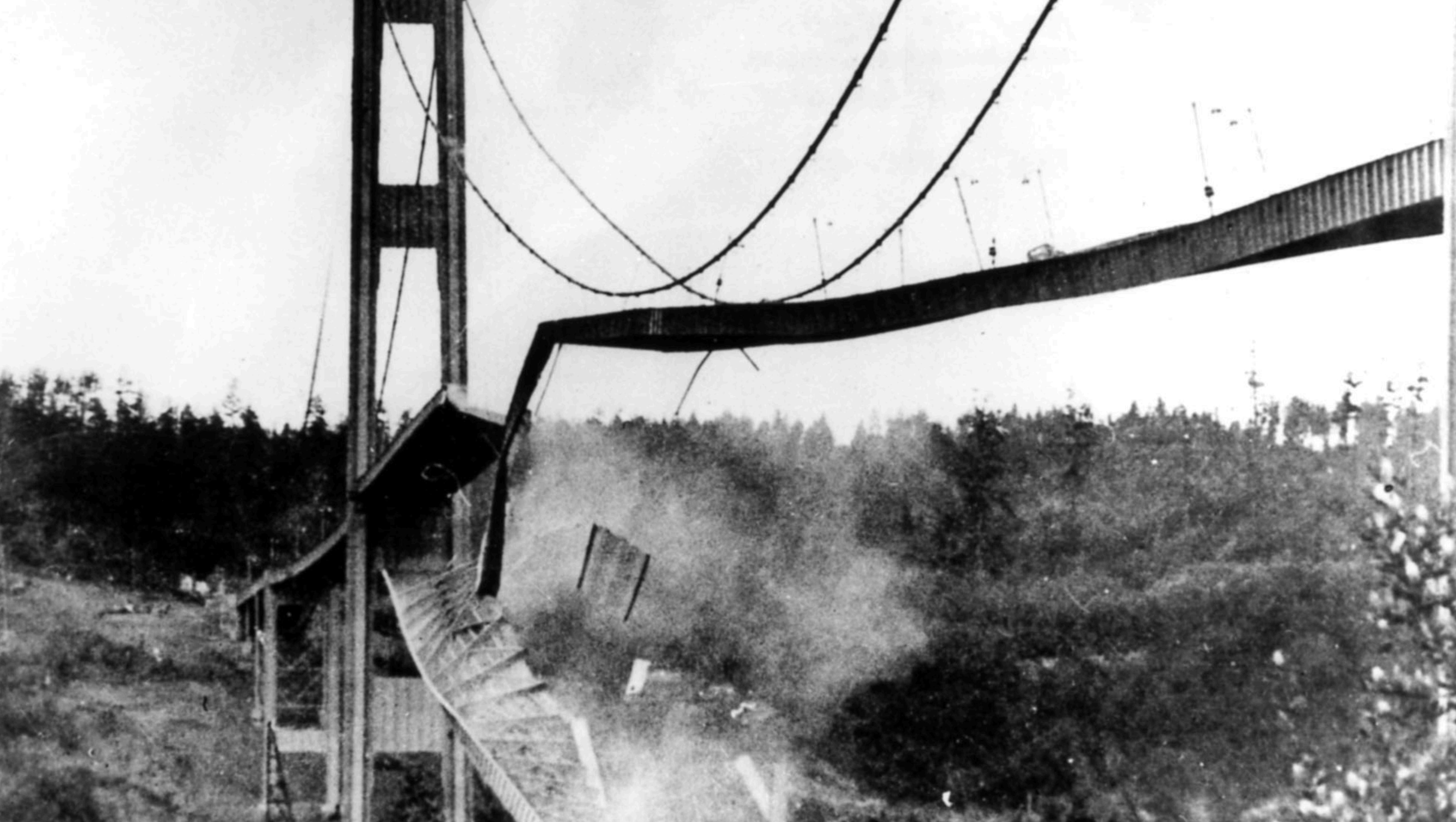
2009

- 1984 ASF+SDF Meta-Environment
- 2002 ASF+SDF Meta-Environment 2.0
- 2009 Rascal metaprogramming Language
- “Meta”: talen voor talen = code voor code
  - Maak je eigen programmeertaal (*dat doen we vandaag!*)
  - Analyseer (en verander) miljoenen regels broncode
- Vakken: Compilerbouw, Model Driven Engineering, Software Onderhoud



# Wat doen taalingenieurs?

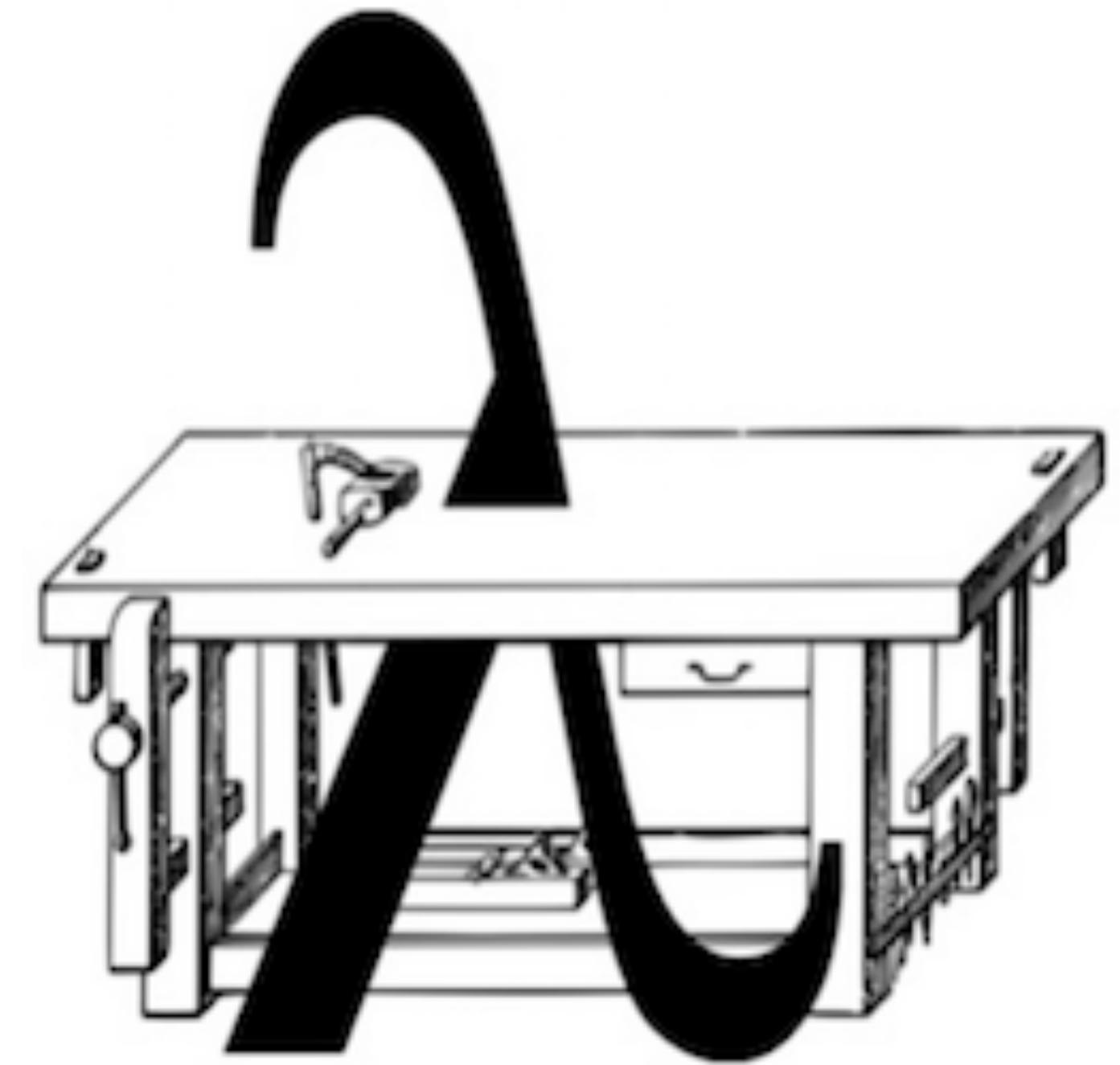




# Language engineering

## de “werktuigbouwkunde” van computertalen

- Hoe bouw je een computertaal?
- Op solide **principes** en **technieken** gestoeld
- We **weten** wat we doen, en **waarom**
- Bewust van **tradeoffs** en **omgevingsfactoren**
- We kunnen **uitspraken** doen over het resultaat
- Met de juiste **gereedschappen** op de juiste manier



# Spelen met je eigen programmeertaal

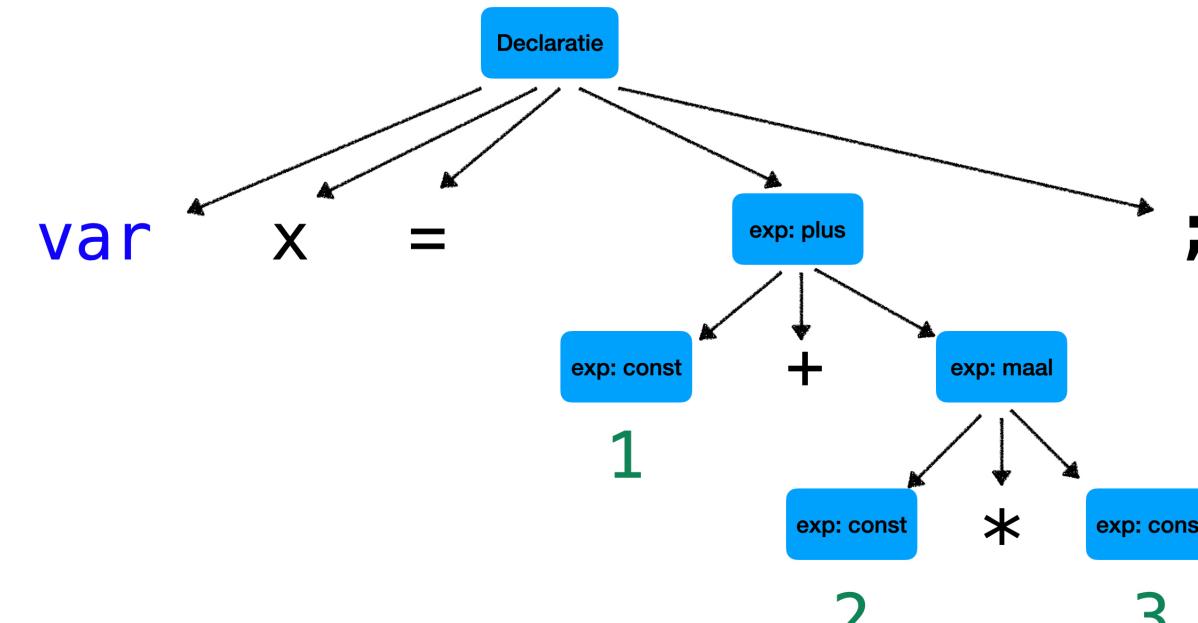
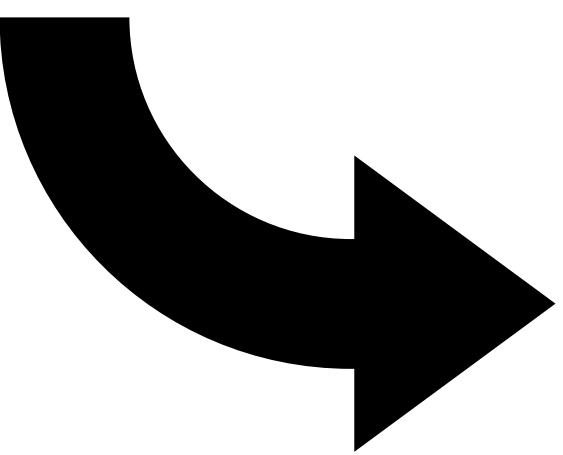
- Vertrouwen op Rascal
- Ontwerpen is ook *uitproberen*, “*fouten*” maken, verbeteren
- Zo effectief mogelijk, grote stappen snel thuis!
- Stel je een doelgroep voor



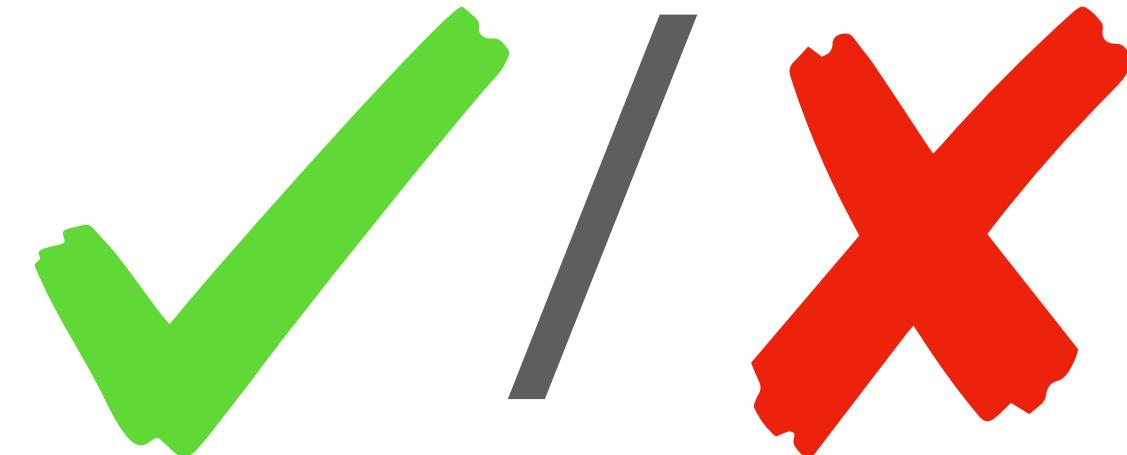
# **Wat gebeurt er allemaal? onder de motorkap**

```
var x = 1 + 2 * 3;
```

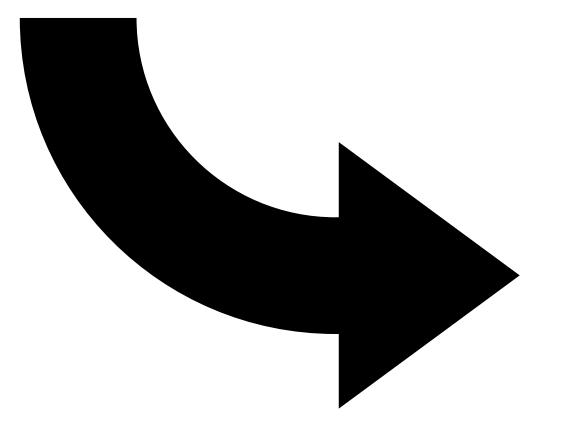
Ontleden



Valideren



Vertalen



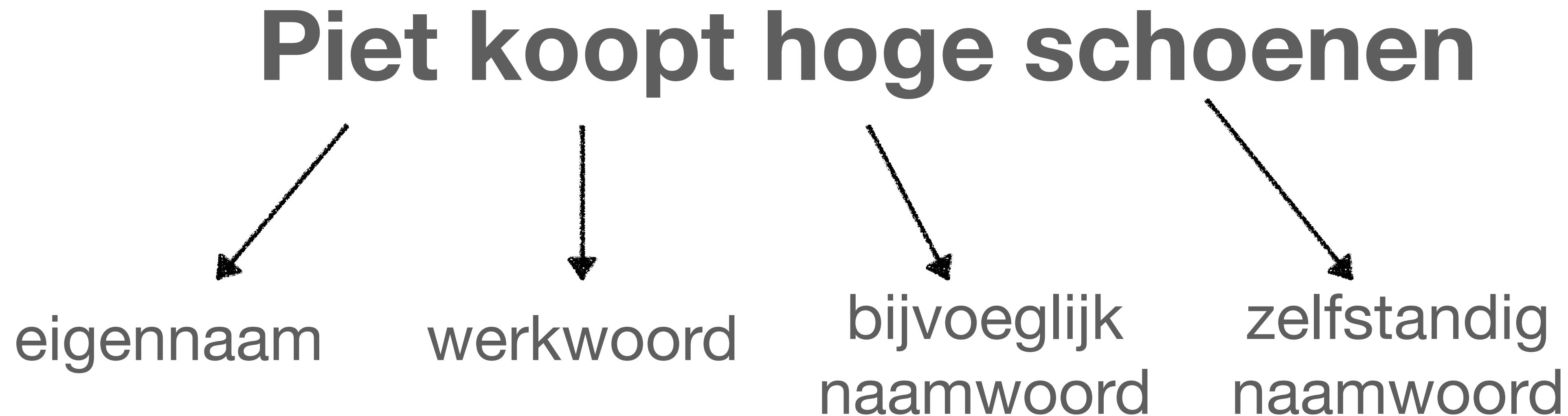
Resultaat

# Ontleden

Piet koopt hoge schoenen

# Ontleden

## syntax: woorden



# Ontleden

## syntax: zinnen

Piet koopt hoge schoenen

onderwerp      persoons-  
                      vorm

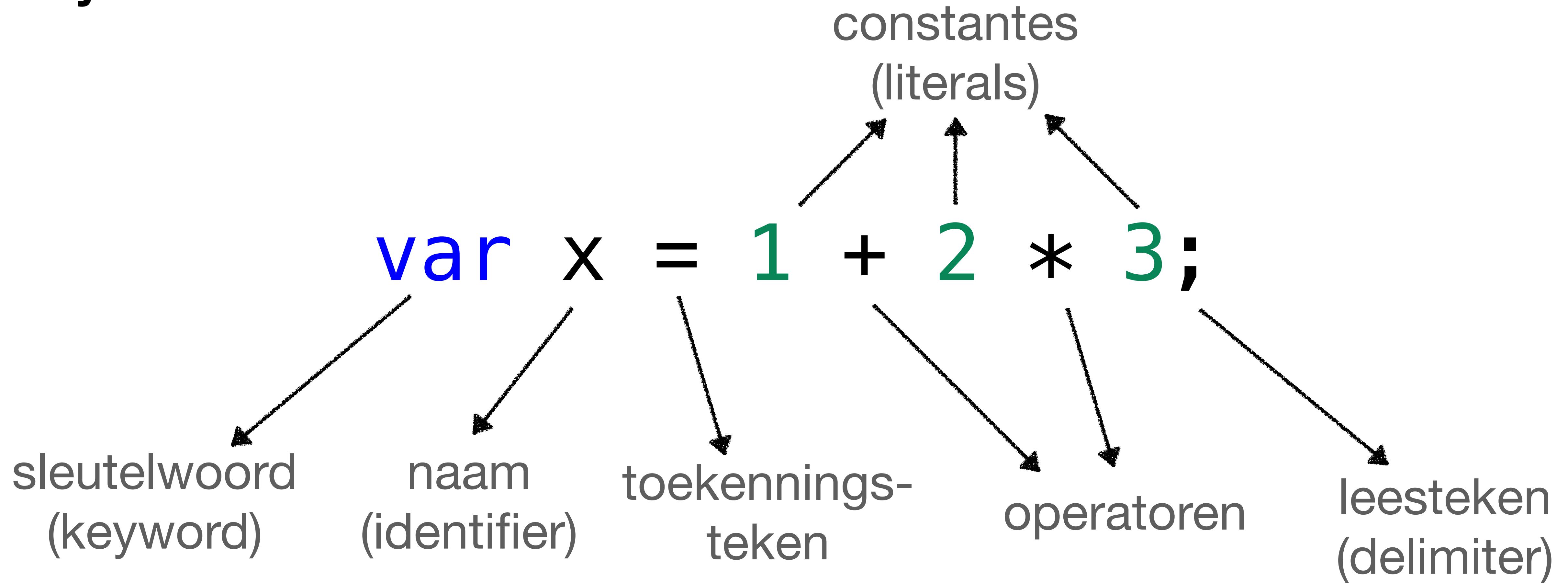
lijdend voorwerp

# Ontleden van computertaal syntax analyse

```
var x = 1 + 2 * 3;
```

# Ontleden van computertaal

## syntax: woorden



# Ontleden van computertaal

syntax: zinnen

var x = 1 + 2 \* 3;

getal

getal

getal

som

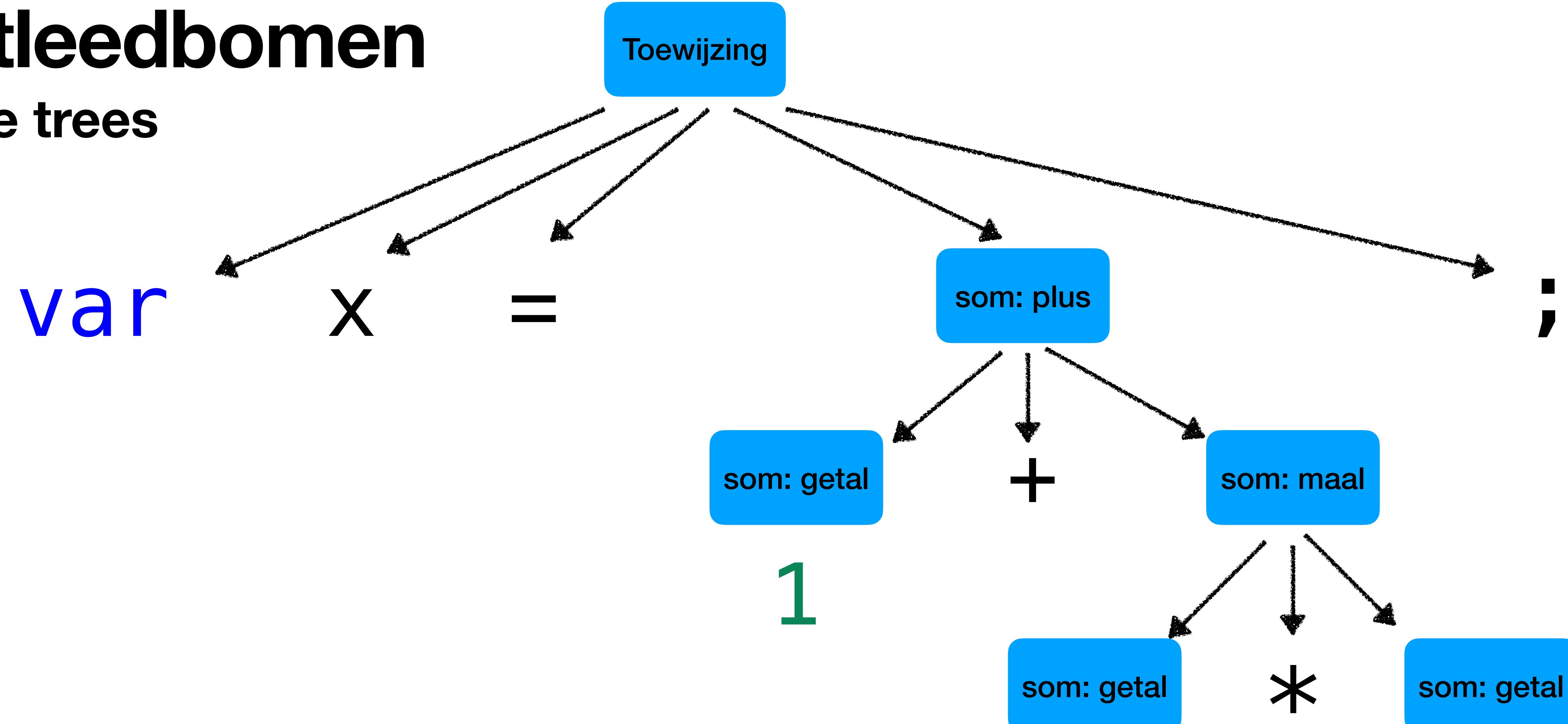
som

toewijzing



# Ontleedbomen

parse trees



```
var x = 1 + 2 * 3;
```

2

3

# Language engineering onderzoek

## syntax analyse: automatisch ontleden!

```
syntax Toewijzing  
= "var" Naam "=" Som ";"
```

```
syntax Som  
= Getal  
| Naam  
| left Som "*" Som  
> left Som "+" Som;
```

```
lexical Getal = [0-9]+;
```

```
lexical Naam = [a-zA-Z]+;
```

Je programmeert in Rascal  
wat de regels van de grammatica zijn.  
ff wachten...  
automatisch ontleden!  
Dus het **hoe** hoef je niet eens te weten :-)

Vandaag: niets onthouden, geen toetsen  
Alleen maar spelen met “YOP”

# Zin en onzin: geldigheid

# Beyoncé is een priemgetal



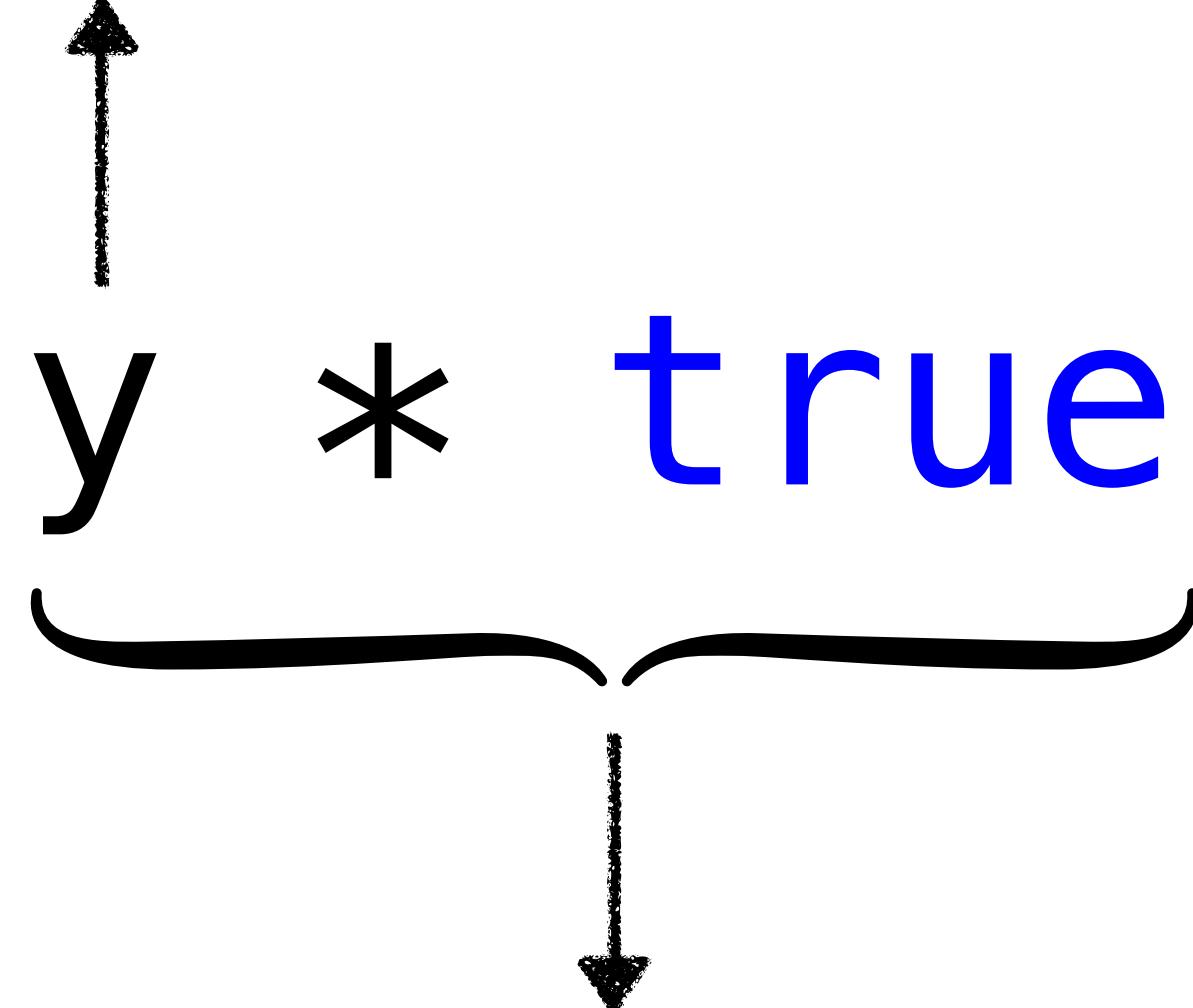
	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

# Geldigheid validiteit, zinnigheid

```
var x = 1 + y * true;
```

# Geldigheid validiteit / “wellformedness” / naamanalyse / “type checking”

**Variabele y bestaat helemaal niet**



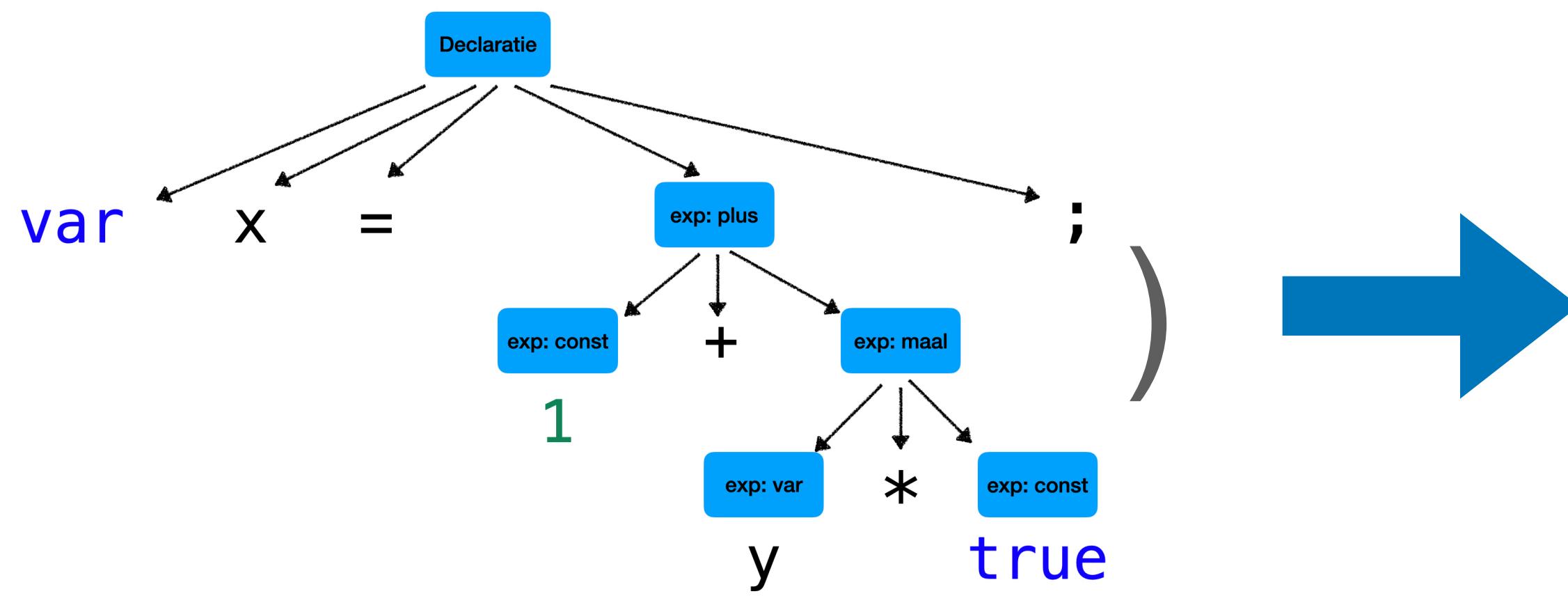
```
var x = 1 + y * true;
```

**wat is onzin keer “waar”?**

# Controleren op geldigheid

validiteit / “wellformedness” / naamanalyse / “type checking”

check(



- Variabele y is niet gedeclareerd
- Vermenigvuldiging ongeldig op waarheidswaarden

# Maar wat betekent het?

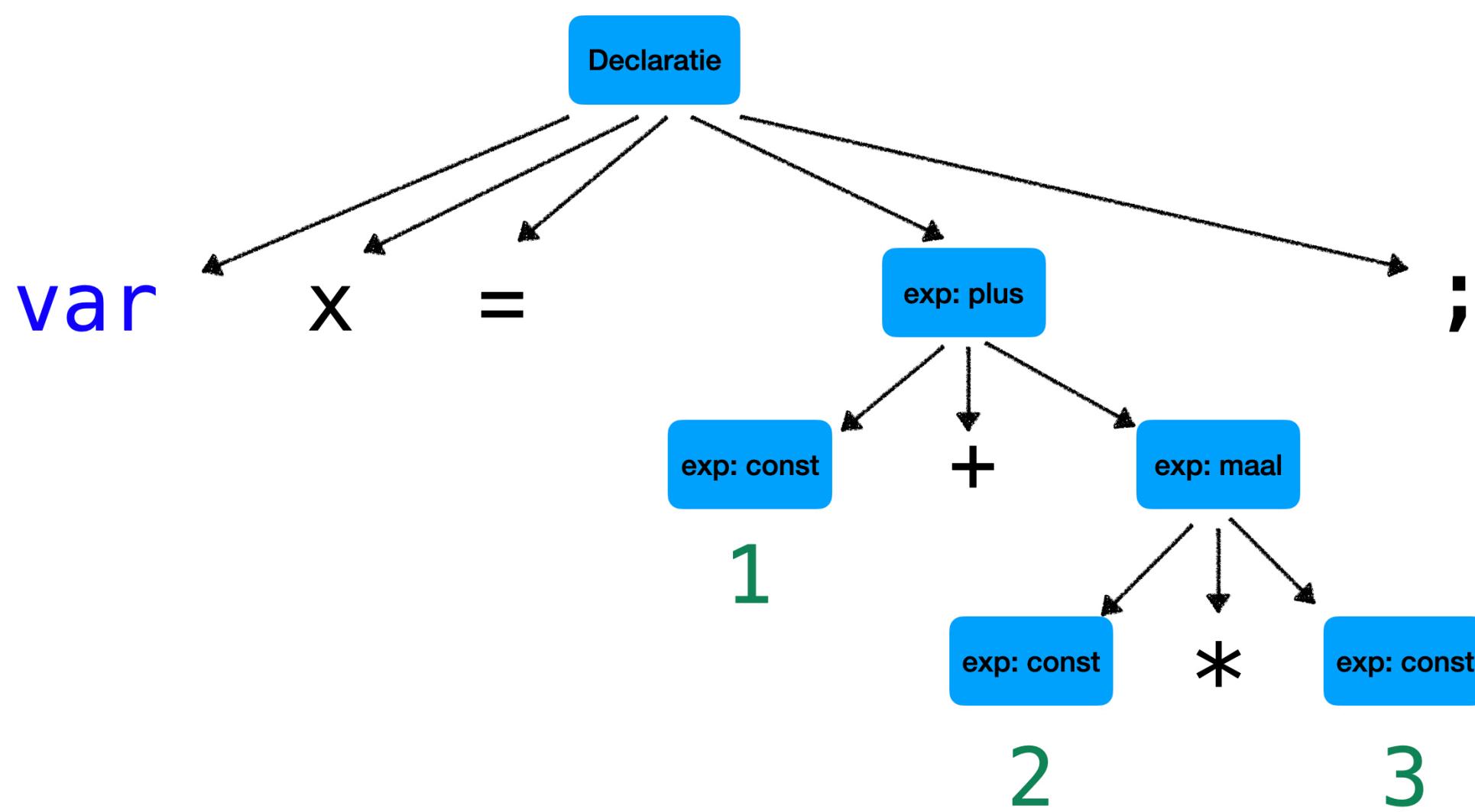
semantiek: vertalen



var x = 1 + 2 \* 3;

# Semantiek = Vertalen, vertalen, vertalen

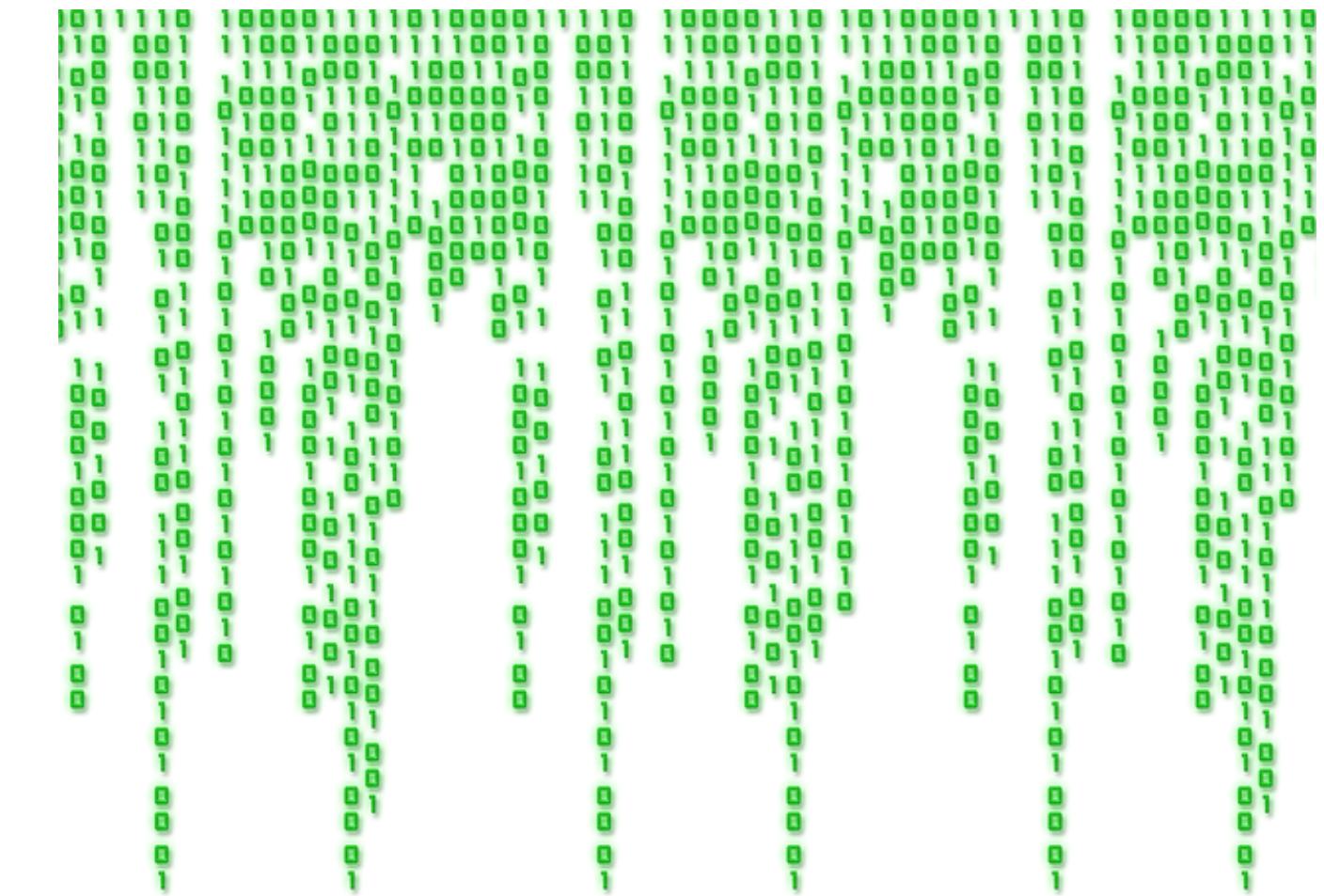
## compilers / interpreters



Je ne parle pas français



dezelfde formule, maar in een taal die de hardware kan uitvoeren



Ik spreek geen frans.

# Vertaling van rekensommen: 1-op-1

```
int vertaal(`n`, env) =.toInt(n);  
  
int vertaal(`x`, env) = zoekInTabel(env, x);  
  
int vertaal(`l + r`, env)  
= vertaal(l, env) plus vertaal(r, env);  
  
int vertaal(`l * r`, env)  
= vertaal(l, env) maal vertaal(r, env);
```

```
syntax Som  
= Getal  
| Naam  
| left Som "*" Som  
> left Som "+" Som;  
  
lexical Getal = [0-9]+;
```

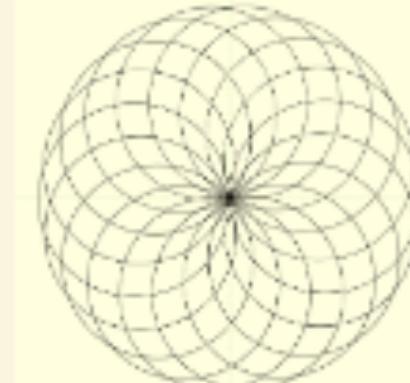
```
lexical Naam = [a-z]+;
```

De sommetjes grammatica

# YOP

## [Y]our [O]wn [P]rogramming Language

- **eerst:** woordenschat en grammaticaregels bedenken
- **dan:** vertalen naar een taal die de computer al kent

Taal	Voorbeeld	
YOP	cirkel 50	Nederlands, relatieve coordinaten
miniSVG	circle(0,0,50,\stroke-color="orange")	Engels, veel haakjes, absolute coordinaten
SVG	</circle cx=0 cy=0 r=50>	Engels, XML, vishaken, absolute coordinaten
Pixels		Beeldtaal, pixels

# YOP

## Vertaalspel

Als je snel klaar bent, doe je het nog een keer  
Doordraaien: nummer 4 wordt nummer 1, etc.

- Teams van vier personen, naast elkaar. En 4 talen.
- Nummer 4 **tekent** straks iets
- Wat nummer 3 als **gebarentaal** uitlegt, gehoord van nummer 2
- Die in het **{frans,engels,arabisch,papiaments,...}** zei, wat nummer 1
- had bedacht in het **Nederlands**, gekozen uit:
  - cirkel, ellipse, ruit, vierkant, rechthoek, ster
  - met een grootte (in cm)  $> 0$  en  $\leq 10$

# Vertalen is betekenis geven

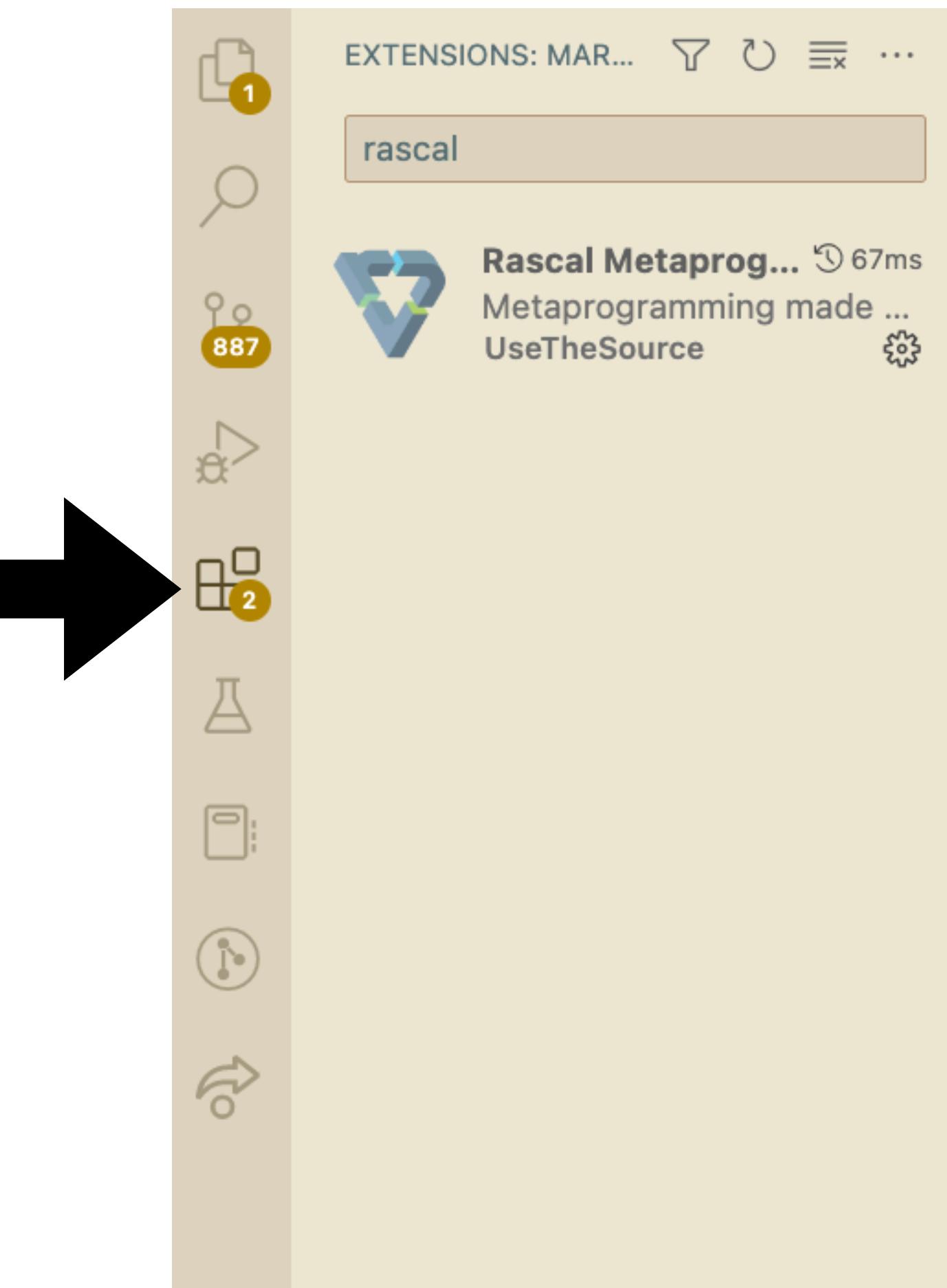
- 你好嗎 -> Nǐ hǎo ma -> Hoe gaat het?
- “Tussentalen” zijn fijn.
- Computerhardware kent van huis uit alleen een soort **binaire gebarentaal**
  - **01 = vermenigvuldig, 10=sla op in geheugen, 11=haal op uit geheugen**
- In welke taal werden de eerste automatische vertalers geschreven?
- In welke taal werden de meest bekende programmeertalen geschreven?
- In welke taal gaan jullie YOP maken?

# Voorbereiding practicum

- Eigen laptop mee, geleend van CWI, of een scherm achterin de zaal
- Microsoft VScode geïnstalleerd
- Rascal extensie geïnstalleerd, en automatisch Java
- yop-rascal-nl project toegevoegd
- `Gebruiker.rsc` geopend
- Wachten tot de “status” balk is uitgeraast
- **Hulp van Jurgen, Tijs, Riemer, Paul**



# Rascal extensie

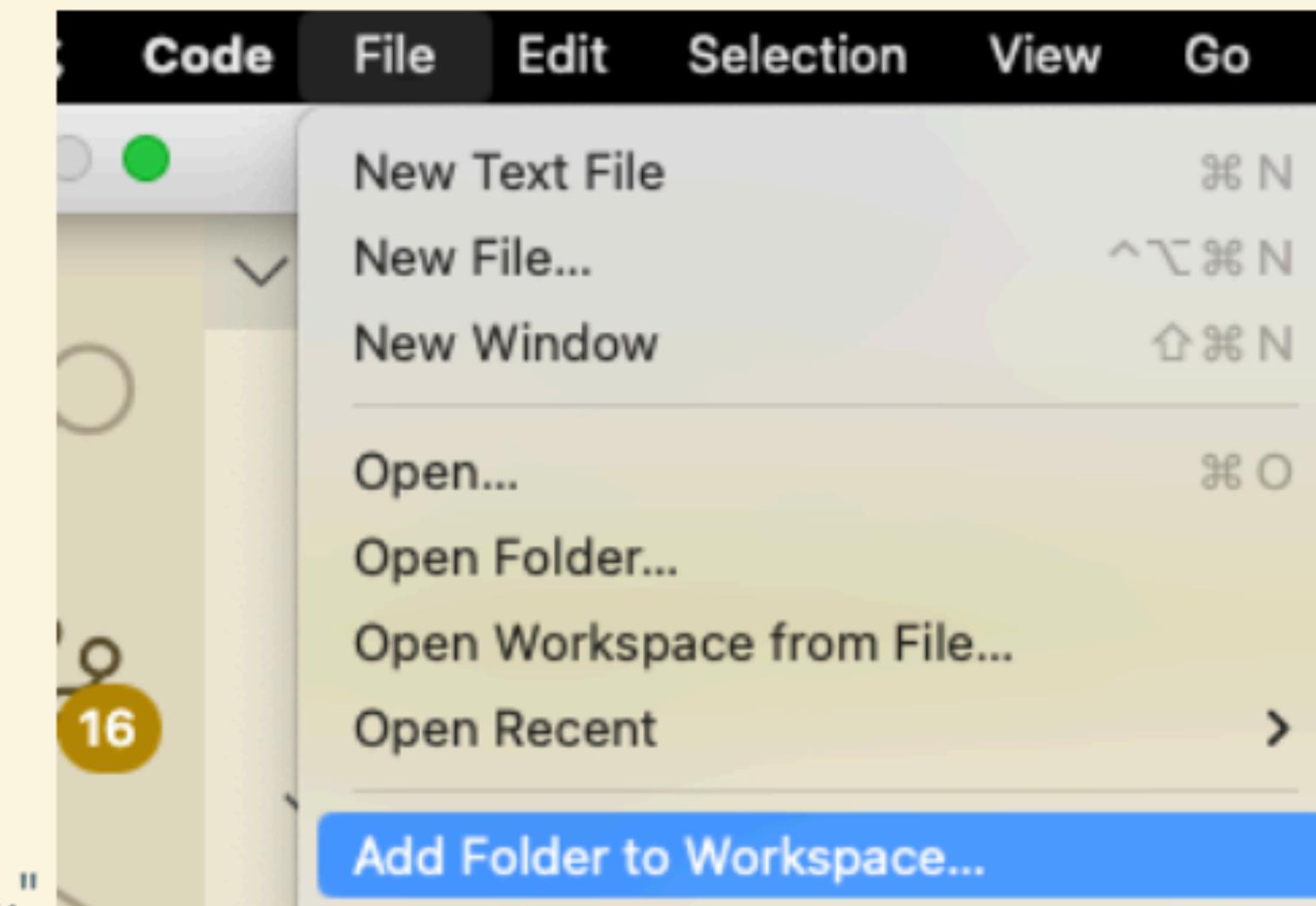


# YOP project zip bestand

Je hebt het YOP project nodig.

- óf download de zip hier:

- <https://github.com/cwi-swat/yop-rascal-nl/releases/download/2023-03-03/yop-rascal-nl.zip>
- **kortere URL** voor hetzelfde: <https://tinyurl.com/2yfup2rr>
- Pak de zip ergens uit. Onthoudt waar ergens is. Bijvoorbeeld in **Mijn Documenten/yop-rascal-nl**
- de folder **moet yop-rascal-nl** heten.
- in die folder staat **src** en **META-INF**, etc.
- als dit niet klopt, vraag even hulp!
- Voeg **Mijn Documenten/yop** toe aan je VScode "workspace":



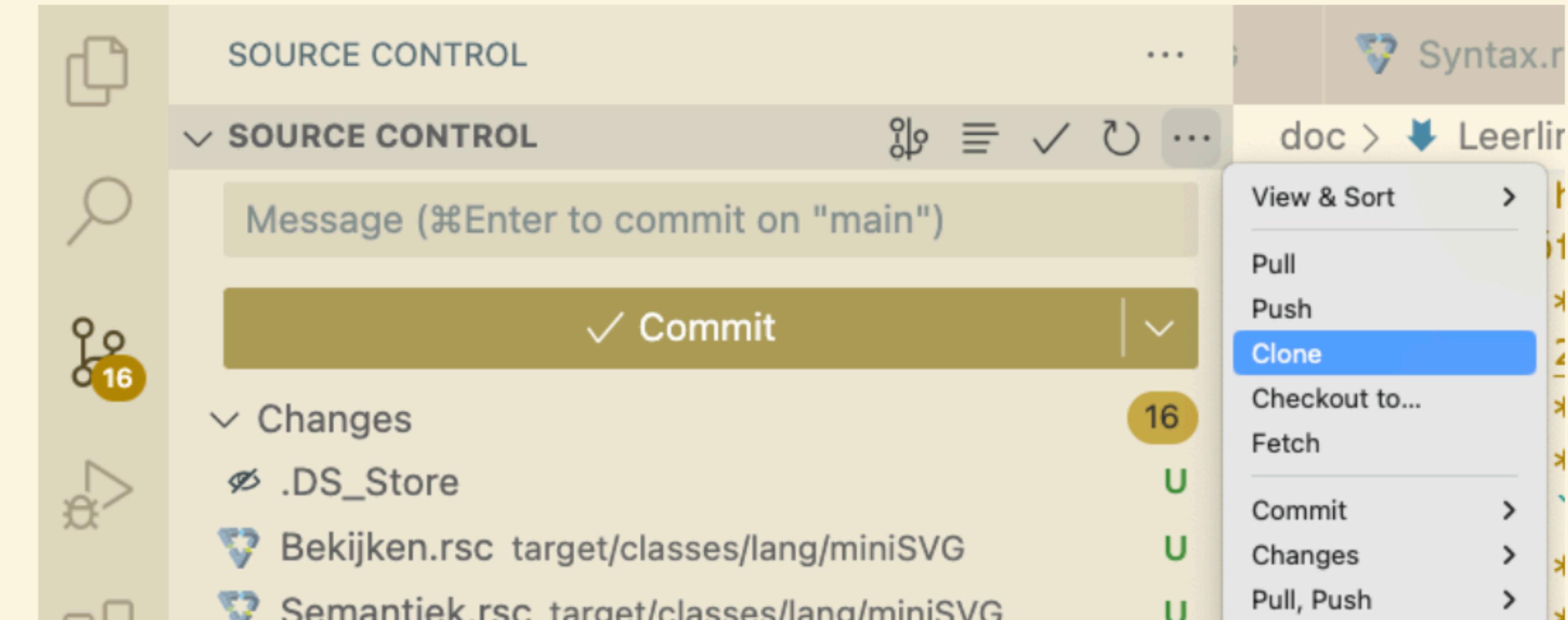
- "File" menu, "Add folder to workspace..."

# YOP project

## git clone

óf je "cloned" hem met "git"

- CMD+P op de mac of CTRL+SHIFT+P op Windows, dan **clone** typen en ENTER, of vindt die feature via dit menu:

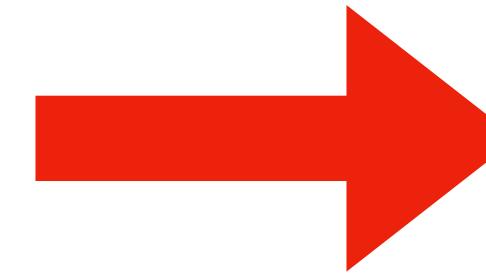


The screenshot shows the Source Control panel of an IDE. On the left is a sidebar with icons for file operations (copy/paste), search, changes (with 16 pending), and other options. The main area shows a commit message input field with placeholder text "Message (⌘Enter to commit on 'main')". Below it is a large green "Commit" button with a checkmark icon. To the right, there's a list of changes: ".DS\_Store" (marked U), "Bekijken.rsc target/classes/lang/miniSVG" (marked U), and "Semantiek.rsc target/classes/lang/miniSVG" (marked U). A context menu is open on the right, listing "View & Sort", "Pull", "Push", **Clone** (which is highlighted in blue), "Checkout to...", "Fetch", and other options like "Commit", "Changes", and "Pull, Push".

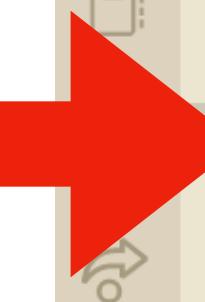
- CMD+P op de mac of CTRL+SHIFT+P op Windows, dan **clone** typen en ENTER.
- Kies <https://github.com/cwi-swat/yop-rascal-nl> als repository (precies overtypen)
- kies een locatie: Bijvoorbeeld in **Mijn Documenten/yop-rascal-nl**
- dan gaat de rest vanzelf

# Gebruiker.rsc

1



2



De status balk onderin raast af en toe even

⌚ Loading evaluator for YOP: printing the source code of the parser class

```
55     }
56 }
57
58 value exec(mini(Programma p)) {
59     try {
60         MiniSVG mini = vertaal(p);
61         bestand = (p.src.top.parent + "svg" + p.src.top.file)[extension="mini"];
62         iprintToFile(bestand, mini);
63         edit(bestand);
64         return ("result": true);
65     }
66     catch loc src : {
67         registerDiagnostics([error("Delen door nul is flauwkeul", src)]);
68         return ("result": false);
69     }
70 }
71
72 value exec(svg(Programma p)) {
73     try {
74         str svg = toSVG(vertaal(p));
75         bestand = (p.src.top.parent + "svg" + p.src.top.file)[extension="svg"];
76         writeFile(bestand, svg);
77         edit(bestand);
78         return ("result": true);
79     }
80     catch loc src : {
81         registerDiagnostics([error("Delen door nul is flauwkeul", src)]);
82         return ("result": false);
83     }
84 }
85
86 Run in new Rascal terminal
86 void main() {
87     registerLanguage(yopLang);
88 }
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    GITLENS

# Terminal

The screenshot shows a VS Code interface with the 'TERMINAL' tab selected in the top navigation bar. The terminal window displays the following Rascal code and its execution:

```
|lib://rascal-lsp|,  
|target://yop-rascal-nl|,  
|system:///|  
],  
libs=[|lib://rascal|],  
srcs=[  
    |lib://rascal-lsp|,  
    |file:///Users/jurgenv/git/yop-rascal-nl/src|  
])INFO: resolved |lib://rascal| at |jar+file:///Users/jurgenv/.vscode/extensions/usethesource.rascalmpl-0.6.4-head/assets/jars/rascal.jar!/  
rascal>import lang::yop::Gebruiker;  
ok  
rascal>main()  
ok  
rascal>main()      Pijltje omhoog + ENTER  
ok  
rascal>          Pijltje omhoog + ENTER  
rascal>■
```

De status balk onderin raast af en toe even

Loading evaluator for YOP: printing the source code of the parser class

# YOP editor

The screenshot illustrates the YOP editor interface, divided into three main sections:

- EXPLORER** (Left): Shows the project structure with a tree view. Red arrows point to the 'target' folder and the 'dahlia.yop' file under 'voorbeelden'.
- EDITOR** (Middle): Displays the code for 'dahlia.yop'. The code defines a recursive drawing pattern:

```
1 herhaal 8 {  
2     rechts 45  
3     herhaal 6 {  
4         herhaal 90 {  
5             vooruit 2  
6             rechts 2  
7         }  
8     }  
9 }  
10 rechts 90
```

Annotations with red arrows explain the workflow:
  - A red arrow points from the 'vertalen' (translate) link in the header to the right margin of the code editor.
  - A red arrow points from the 'schrijven' (write) link in the header to the left margin of the code editor.
  - A red arrow points from the 'klik=terug!' (click=return!) text to the generated SVG preview.
- PREVIEW** (Right): Shows the resulting SVG output, which is a complex, multi-layered flower-like fractal pattern.

# YOP vertalingen, met alle tussenstappen

## Bekijk MiniSVG code

☰ dahlia.mini × ☰ http://localhost:9052/

voorbeelden > svg > ☰ dahlia.mini

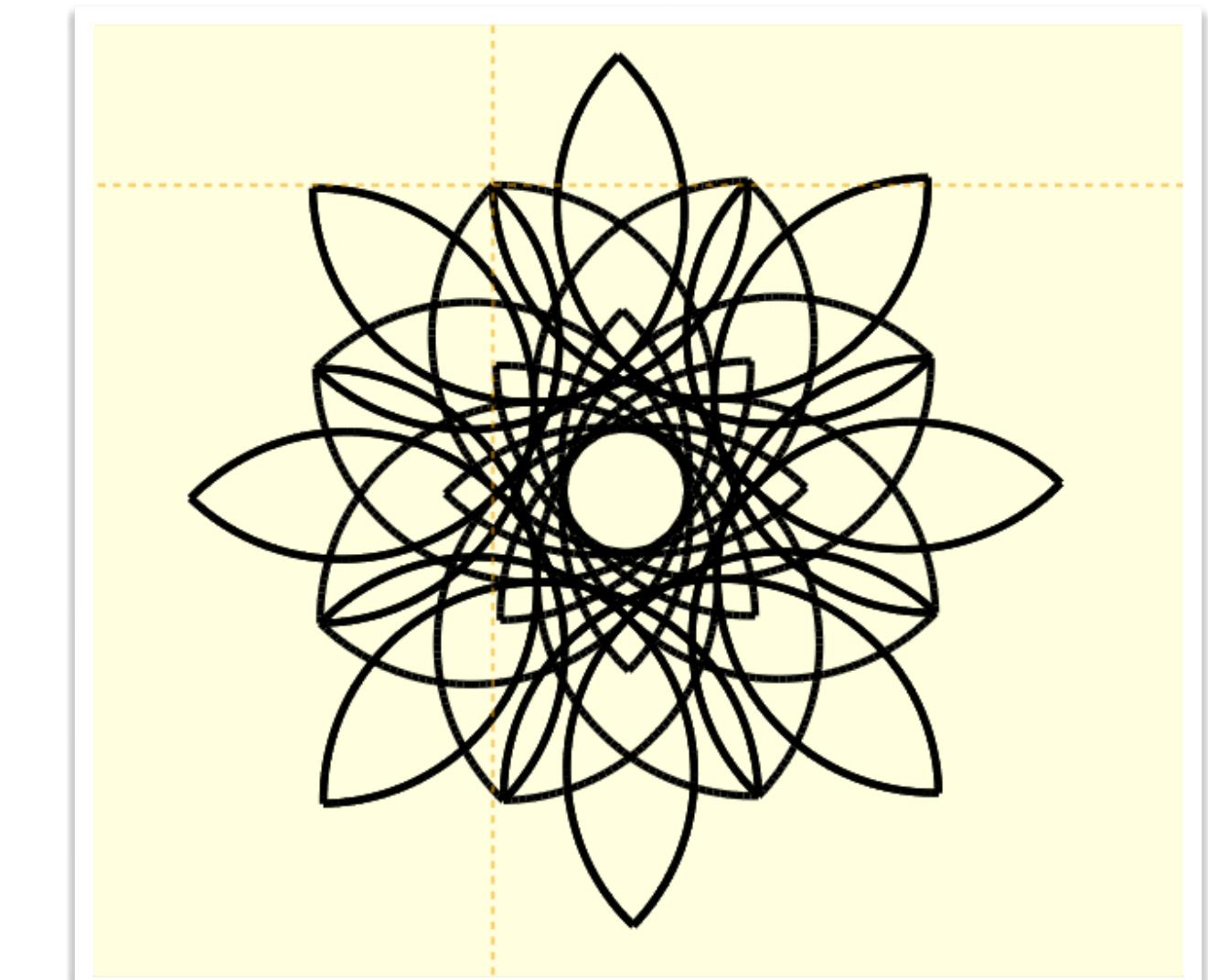
```
1 miniSVG( [move(
2   | 0.,
3   | 0.,
4   |
5   |   comment("🐢 rechts 45; x: 0., y: 0., richting: -45., pen: true 🐢"),
6   | move(
7   |   | 0.,
8   |   | 0.,
9   |   |
10  |   | move(
11  |   |   | 0.,
12  |   |   | 0.,
13  |   |   |
14  |   |   | link(
15  |   |   |   |file:///Users/jurgenv/git/yop-rascal-nl/voorbeelden/dahlia
16  |   |   |   | line(0.,1.41421356236,0.,-1.414213562380)),
17  |   |   |   | comment("🐢 rechts 2; x: 1.41421356236, y: -1.414213562380,
18  |   |   |   | link(
19  |   |   |   |   |file:///Users/jurgenv/git/yop-rascal-nl/voorbeelden/dahlia
20  |   |   |   |   | line(1.41421356236,2.77821028254,-1.414213562380,-2.876920
21  |   |   |   |   | comment("🐢 rechts 2; x: 2.77821028254, y: -2.87692096557018
22  |   |   |   |   | link(
23  |   |   |   |   |   |file:///Users/jurgenv/git/yop-rascal-nl/voorbeelden/dahlia
24  |   |   |   |   |   | line(2.77821028254,4.09032834062,-2.87692096557018682992,-
25  |   |   |   |   |   | comment("🐢 rechts 2; x: 4.09032834062, y: -4.38634012593056
26  |   |   |   |   |   | link(
27  |   |   |   |   |   |   |file:///Users/jurgenv/git/yop-rascal-nl/voorbeelden/dahlia
28  |   |   |   |   |   |   | line(4.09032834062,5.34896912292,-4.38634012593056048976,-
29  |   |   |   |   |   |   | comment("🐢 rechts 2; x: 5.34896912292, y: -5.94063204874112
30  |   |   |   |   |   |   | link(
31  |   |   |   |   |   |   |   |file:///Users/jurgenv/git/yop-rascal-nl/voorbeelden/dahlia
```

| Bekijk SVG code

voorbeelden > svg >  dahlia.svg

```
1  <svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" vi
2   <rect x="0" y="0" width="1000" height="1000" style="fill: lightyellow"/>
3   <g transform="matrix(1 0 0 -1 500 500)" >
4
5     <g transform="matrix(1 0 0 1 0.0 0.0)" >
6
7       <!-- 🌸 rechts 45; x: 0., y: 0., richting: -45., pen: true 🌸 -->
8       <g transform="matrix(1 0 0 1 0.0 0.0)" >
9
10      <g transform="matrix(1 0 0 1 0.0 0.0)" >
11
12        <a href="#" onclick="fetch('/editor?' + new URLSearchParams({src: '|fil
13          <line x1="0.0" x2="1.414213562360" y1="0.0" y2="-1.4142135623800" s
14            fill="rgb(230,230,230,1.0)"
15            stroke-width="2.0"
16            fill-opacity="0.80"
17            stroke-opacity="1.0" />
18        </a>
19        <!-- 🌸 rechts 2; x: 1.41421356236, y: -1.414213562380, richting: -45.0 -->
20        <a href="#" onclick="fetch('/editor?' + new URLSearchParams({src: '|fil
21          <line x1="1.414213562360" x2="2.778210282540" y1="-1.4142135623800" s
22            fill="rgb(230,230,230,1.0)"
23            stroke-width="2.0"
24            fill-opacity="0.80"
25            stroke-opacity="1.0" />
26        </a>
27        <!-- 🌸 rechts 2; x: 2.77821028254, y: -2.87692096557018682992, richting: -45.0 -->
28        <a href="#" onclick="fetch('/editor?' + new URLSearchParams({src: '|fil
29          <line x1="2.778210282540" x2="4.090328340620" y1="-2.87692096557018682992" s
30            fill="rgb(230,230,230,1.0)"
31            stroke-width="2.0"
32            fill-opacity="0.80"
33            stroke-opacity="1.0" />
34        </a>
35        <!-- 🌸 rechts 2; x: 4.09032834062, y: -4.38634012593056048976, richting: -45.0 -->
36        <a href="#" onclick="fetch('/editor?' + new URLSearchParams({src: '|fil
```

Bekijk plaatje



# Opdracht 0: Spelen met YOP

- YOP is een schildpadtaal zoals “LOGO”
  - de schildpad loopt en houdt een pen vast
  - “pen op”, dan tekent hij niets
  - “pen neer”, dan tekent hij iets
  - “links 50” draai 50 graden naar links (rechts) werkt ook
- Assenstelsel:  $x$  (horizontaal) en  $y$  (verticall) van -500 tot 500 met de oorsprong in het midden
- Rekenen:  $x$ ,  $-$ , / zit er in
- Formule letters/namen zitten er al in: “ $a = 1$ ”, “links  $3 \times a$ ”
- Herhalen zit er al in: “herhaal  $3 \times a \{ \text{links } 2 \text{ vooruit } a \}$ ”
- Verder mist er aardig wat :-)
- **Opdracht:** probeer de voorbeelden, schrijf zelf YOP programma's bedenk wat je wel zou willen verbeteren

# **tot 15:00**

- Opdracht 1 `vooruit` -> `loop`
- Opdracht 2 `tekst Blablabla` toevoegen
- Opdracht 3 vrije creativiteit
- Opdracht 4 demo!! met prijsuitreiking

# Opdracht 1

## De eerste vertaler naar miniSVG aanpassen

### ### Deel 1

- \* `vooruit <getal>` naar `loop <getal>`
  - \* Vervang in `src/lang/yop/Syntax` `vooruit` door `loop`
  - \* Vervang in `src/lang/yop/Semantiek`, `vooruit` door `loop`
- 
- \* Probeer een yop programma uit te voeren:
    - \* eerst de `main` functie van `Gebruiker` nog eens draaien. Daarna ziet VScode de wijzigingen.
    - \* `vooruit` zit niet meer in de woordenschat.
    - \* pas dus `vooruit` aan naar `loop`

### ### Deel 2

- \* Nu zijn alle oude programma's stuk
- \* Laten we de oude `vooruit` **\*ook\*** toestaan!
- \* Voeg in `src/lang/yop/Syntax` een kopie van de `loop` regel toe.
- \* Kopiëer in `src/lang/yop/Semantiek` de vertaalregel voor `loop`
- \* Probeer yop programma's waar zowel `loop` als vooruit in `voorkomen`

# Opdracht 2

## Een handige uitbreiding

\* Voeg toe aan `src/lang/yop/Syntax.rsc`

```
syntax Tekening = "schrijf" Tekst;
lexical Tekst = ![\n]+ !>> ![\n];
```

\* Voeg daarna toe aan `src/lang/yop/Semantiek.rsc`

```
Element vertaal((Tekening) `schrijf <Tekst zin>`) = text(huidigeX, huidigeY, "<zin>");
```

\* Uitproberen! Wat gaat er mis?

\* Repareren:

```
```
Element vertaal((Tekening) `schrijf <Tekst zin>`)
  = rotate(huidigeRichting, [text(huidigeX, huidigeY, "<zin>")]);
```

```

Gelukkig zat er nog een `rotate` in miniSVG...

# Opdracht 3

## Creatieve veranderingen en uitbreidingen

- \* maak nog vettere YOP programma's
- \* voeg leuke, rare, handige taal constructies toe aan YOP en probeer ze uit
  - \* eerst Syntax
  - \* dan Semantiek
  - \* dan testen
  - \* dan fixen
  - \* niet vergeten steeds `main()` te draaien uit `Gebruiker`
- \* denk **niet te moeilijk.** simpel=goed

Stel veel vragen!

Kleine stapjes!

Kijk de kunst af bij elkaar!

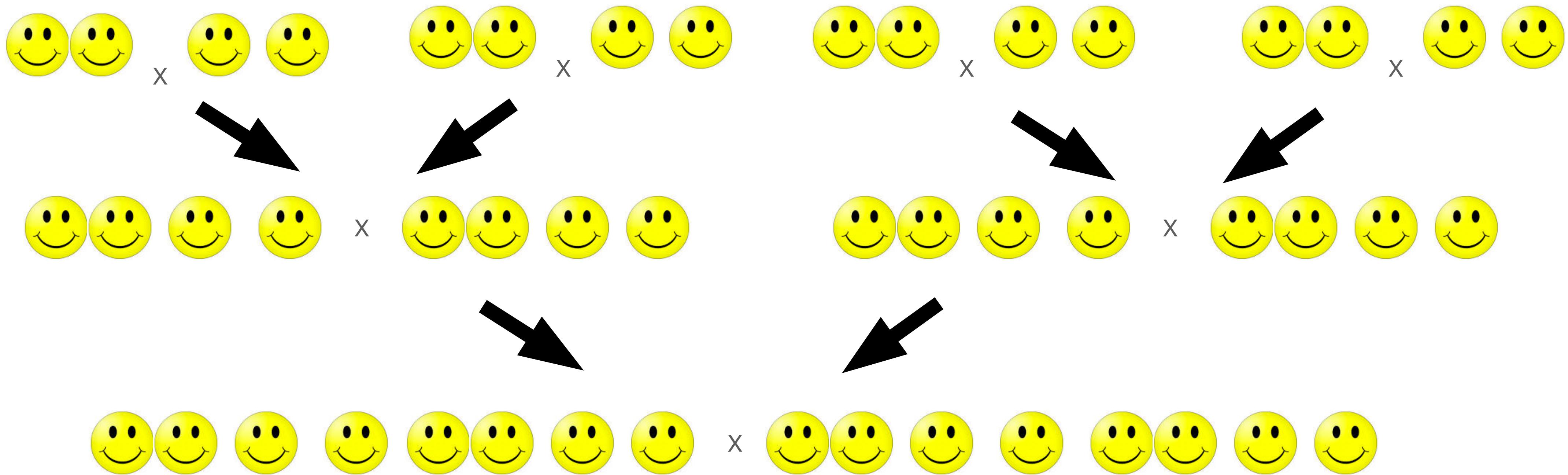
# Opdracht 3

## Tips en ideeën

- \* Er zit al een `Kleur` concept in YOP maar dat wordt nergens gebruikt.
  - \* Je kan `Tekening` uitbreiden met iets van kleur.
  - \* Er is al een `vertaal` voor `Kleur` die je kunt toepassen.
  - \* Of, je zou kunnen denken aan andere soorten vormen toevoegen: ellipse, vierkant, rechthoek.
  - \* Of, je kan gewoon alles veranderen in de taal en het beter maken, volgens je eigen smaak.
  - \* Vertaling naar Straatstaal, Spaans, Papiaments, Javaans, Frans, Marrokaans, Chinees, of Turks?
- 
- \* Verander stap-voor-stap, woord voor woord, concept voor concept:
  - \* Eerst de syntax
  - \* Dan de semantiek (vertaling)
  - \* Dan proberen en testen
  - \* Dan het volgende woord of concept veranderen.
  - \* **\*\*NIET IN 1 KEER ALLES VERANDEREN\*\***

# Demo time!

**De waterval methode: kies een winnaar in steeds grotere groepjes**



Er blijven twee of drie groepjes over voor een korte klassikale demonstratie  
en dan stemmen we!

# Afsluiting

## YOP; wat hebben we geleerd?



Centrum Wiskunde & Informatica

- Bedankt voor jullie komst en jullie inzet!
- Je kan zelf een programmeertaal maken
- De computer het zware werk laten doen
- Veel proberen in kleine stapjes
- Rascal is handig voor het maken van programmeertalen
- **Software en programmeertalen? misschien iets voor jou?**

LYCEUM  
KRALINGEN

