

Cursus Programmeren en Dataverwerking



<http://hay.github.io/codecourse>

Vanavond (18.00 - 21.30)

- Introductierondje
- Algemene introductie (60 - 90m)
 - Iets over bits en bytes
 - Iets over programmeurs en programmeertalen
 - Programmeren en de terminal
- Aan de slag!
- Huiswerk :)

- Introduceer jezelf
 - Je eerste computerervaring
 - Heb je wel eens iets geprogrammeerd? Zo ja, wat?
 - Heb je wel eens een terminal (zoals MS-DOS) gebruikt?
 - Waar zou je je nieuwe kennis willen inzetten?





dalogo.gif (6735 bytes)

John Kranen
Baer Hornix

Each artist (from the primitive through Rembrandt and van Gogh until now) abstracts his/her theme and creates in that way a new reality. This process of abstraction is a unique and essential part of art. The distinction that one makes in visual art is that between figurative and non-figurative art. The tools used by artists in this abstraction-process have always been the same throughout the centuries: originality, individualism, knowledge, imagination, integrity, creativity, inspiration, etc.

Click [here](#) to continue

For more artwork

This page is © 1998-2003 by [John Kranen](#)

AS SEEN IN

CART

THE INTERNET ART DATABASE

ON



- 3.033 foto's
- 65 filmpjes
- >17 GB aan data
- 3 apparaten
- Canon heeft verkeerde tijd op 1/3 van de foto's
- Wat ik wil:
 - Alle foto's bij elkaar
 - Met een genummerde bestandsnaam
 - Met de correcte tijden
 - Zowel in de bestandsnaam als in de metadata



```
import os, glob, datetime

os.chdir("../Ongesorteerd/")

FILENAME = "JP2015_%04d_%s_%s.%s"

index = 1

files = glob.glob("*")
files.sort(key = os.path.getmtime)

for f in files:
    timestamp = os.stat(f).st_mtime
    date = str(datetime.datetime.fromtimestamp(timestamp)).replace(" ", "T").replace(":", "")[0:-2]
    extension = f[-3:]
    source = f[7:9]

    filename = FILENAME % (index, source, date, extension)
    print filename

    os.rename(f, filename)

    index = index + 1
```

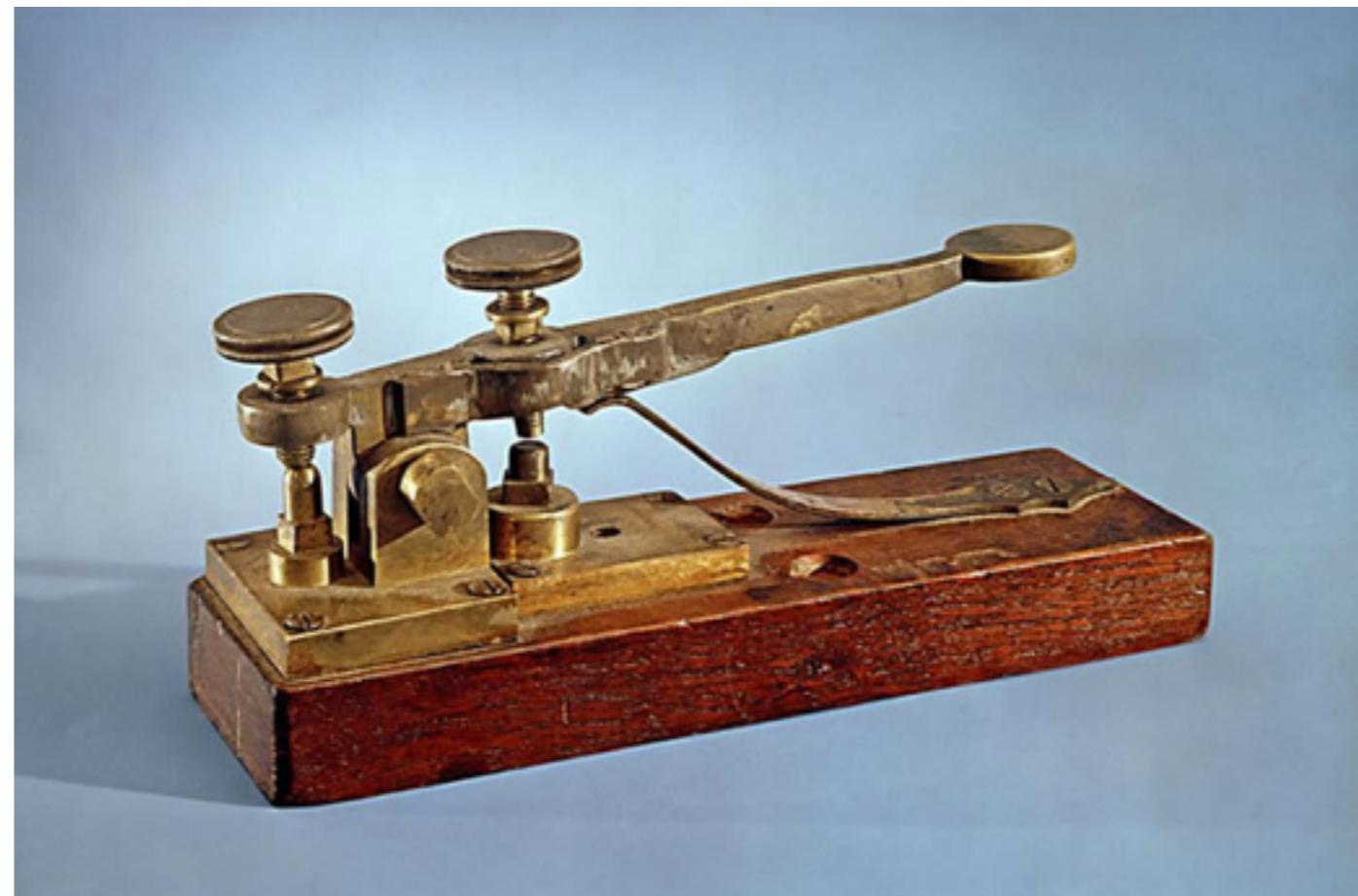


lets over bits en bytes

Wat is een bit?

0025300328004230000000000000000000000000000

02









Lichtsterkte = A + B
(0 = donker, 3 = vol licht)

A



2

B



1

Hoe kun je 109 maken met alleen deze lichtknopjes?



128



64



32



16



8



4



2



1

$$64 + 32 + 8 + 4 + 1 = 109$$

01101101 binair

6D hexadecimaal



128



64



32



16



8



4



2



1

0

1

1

0

1

1

0

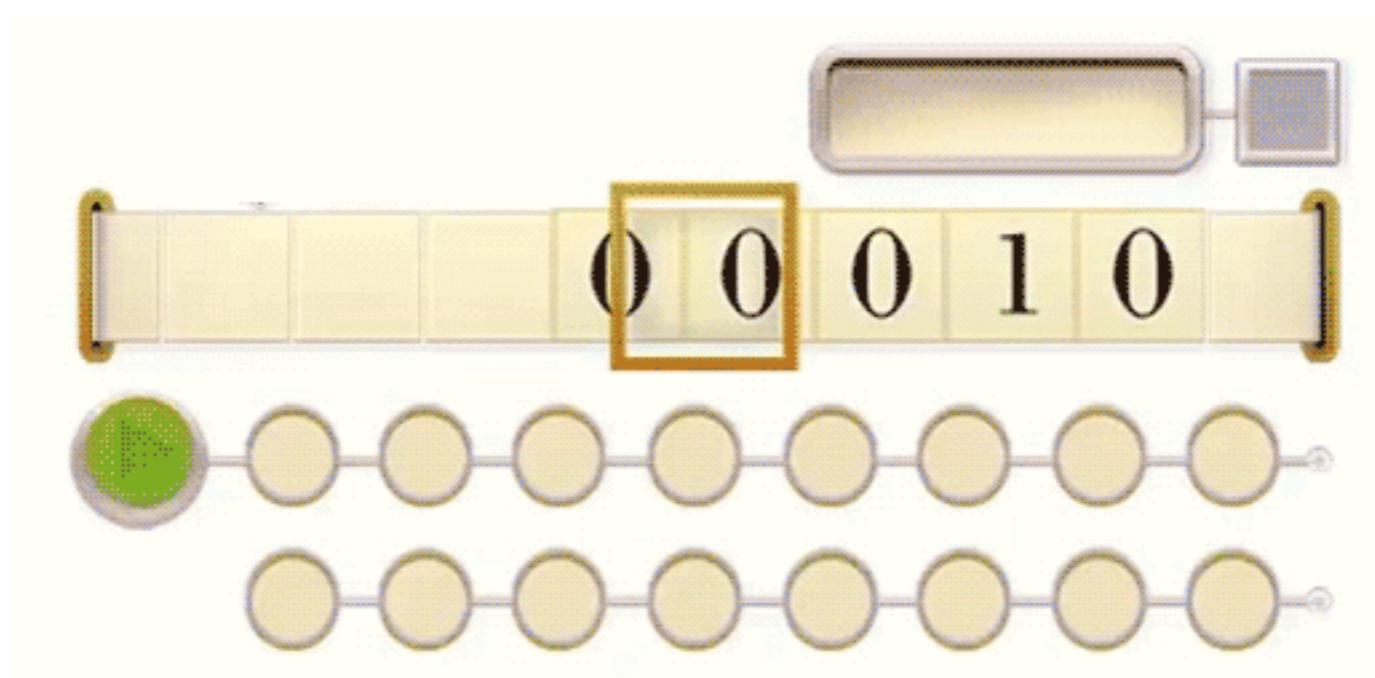
1

ASCII value	Character	Control character	ASCII value	Character	ASCII value	Character	ASCII value	Character
000	(null)	NUL	032	(space)	064	@	096	
001	☺	SOH	033	!	065	A	097	ä
002	☻	STX	034	"	066	B	098	ß
003	♥	ETX	035	#	067	C	099	ç
004	♦	EOT	036	\$	068	D	100	đ
005	♣	ENQ	037	%	069	E	101	é
006	♠	ACK	038	&	070	F	102	í
007	(beep)	BEL	039	'	071	G	103	g
008	■	BS	040	(072	H	104	h
009	(tab)	HT	041)	073	I	105	i
010	(line feed)	LF	042	*	074	J	106	j
011	(home)	VT	043	+	075	K	107	k
012	(form feed)	FF	044	-	076	L	108	í
013	(carriage return)	CR	045	-	077	M	109	m
014	♫	SO	046	.	078	N	110	n
015	☼	SI	047	/	079	O	111	ö
016	▬	DLE	048	0	080	P	112	p
017	▬	DC1	049	1	081	Q	113	q
018	↑	DC2	050	2	082	R	114	r
019	!!	DC3	051	3	083	S	115	s
020	π	DC4	052	4	084	T	116	t
021	§	NAK	053	5	085	U	117	u
022	▬	SYN	054	6	086	V	118	v
023	↑	ETB	055	7	087	W	119	w
024	↑	CAN	056	8	088	X	120	x
025	↓	EM	057	9	089	Y	121	y
026	→	SUB	058	:	090	Z	122	z
027	←	ESC	059	;	091	[123	{
028	(cursor right)	FS	060	<	092	\	124	:
029	(cursor left)	GS	061	=	093]	125	}
030	(cursor up)	RS	062	>	094	^	126	-
031	(cursor down)	US	063	?	095	-	127	▀



Data

Instructies





Apollo Guidance Computer (1966)

Net zoveel techniek als een broodrooster



Deep Blue (1996)

11 miljard berekeningen per seconde
> \$1.000.000



iPhone 6 (2014)

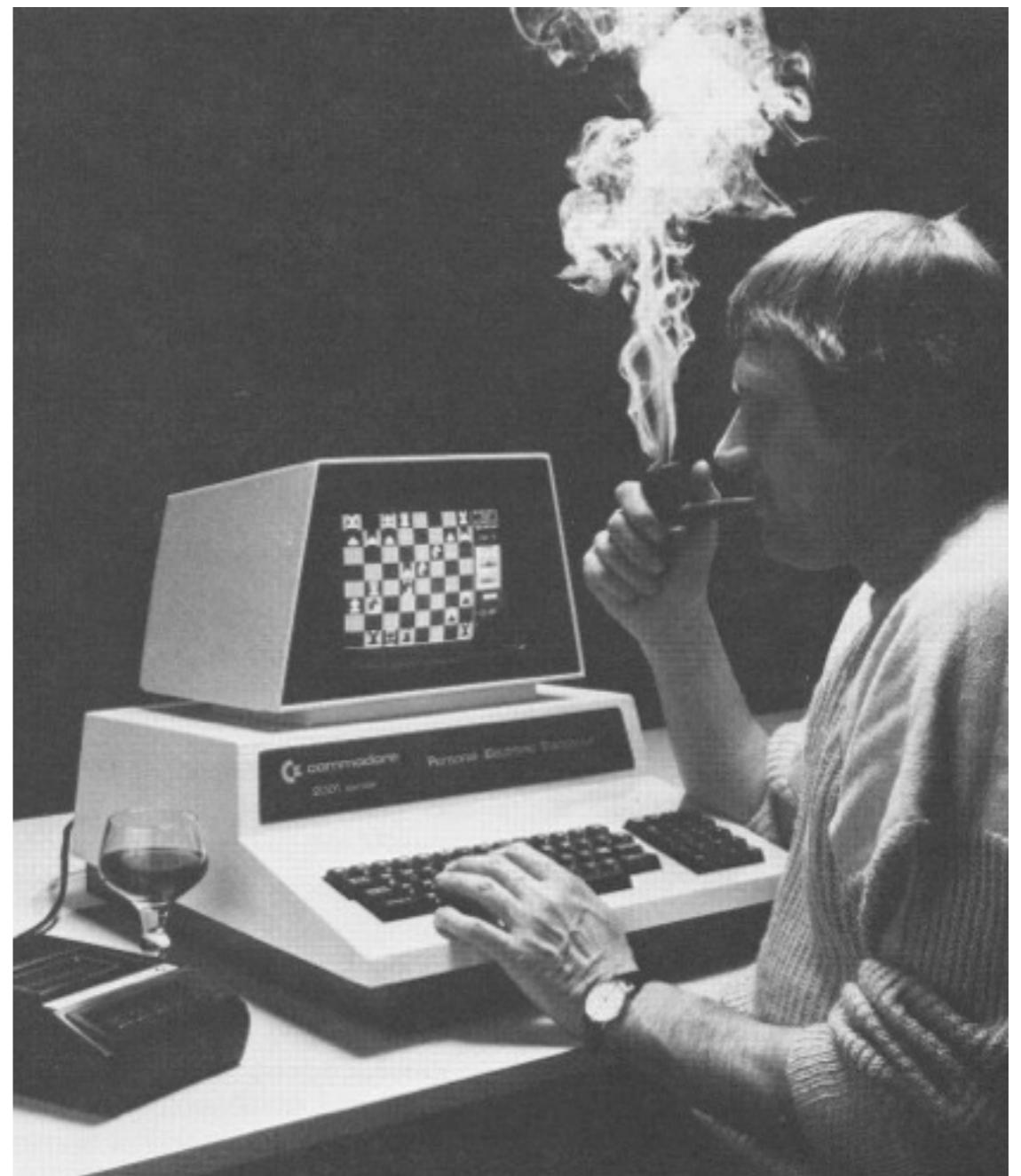
>100 miljard berekeningen per seconde
\$700

Moore's Law

De snelheid en capaciteit van processoren verdubbeld elke 12 tot 18 maanden.

Programmeren

- De instructies die de computer vertellen wat er met al die data moet gebeuren.
- Het in abstracties vatten van algoritmes en processen.
- Alles is gebouwd op een laag van abstracties.
- Talen zijn verschillend, maar alles wordt uiteindelijk machinecode.



```
1 print "Hallo, wereld!"  
2 print "Leuk dat jullie er zijn."  
3 print "Zijn jullie nog niet in slaap gevallen?"|
```



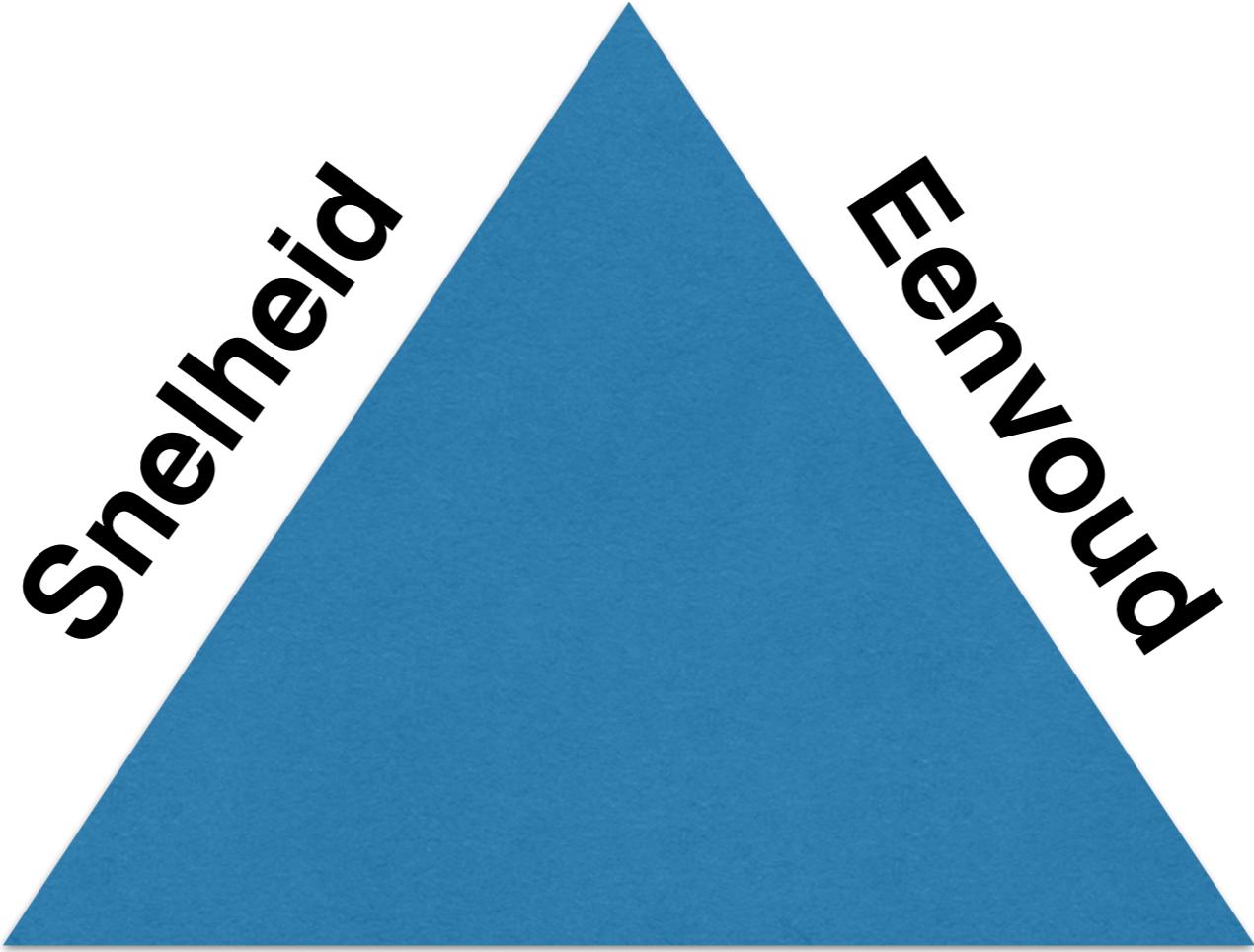
```
111100011010101100111011011110010011110111111111111111  
11001111110011000000000011011111010010110011111101111  
1111111100000111000111000111110011110000000110101111110  
000011101001100100111110111110000111111001100110001011  
100111110000110001100110101111001111100010111010111111  
100100111111110011101111000111111000110111110001111110  
1101111011101011110111001111110011111110011111000100111  
1111100010010111100011000111110001111111111111110111  
1110111111100001110000010111100111111110000000111001100  
10100000111001111101111111110000000110001000011000  
11100111011011111111001011111011110111100000011111111  
1100110011000100001000111111110001111100100000100001000  
0000111110111001001110000111111011111111111000100111  
100001100110010111001000110001001101111000011000111111
```

```
1 print "Hallo, wereld!"  
2 print "Leuk dat jullie er zijn."  
3 print "Zijn jullie nog niet in slaap gevallen?"
```

Compilatie

```
111100011010101100111011011110010011110111111111111111  
1100111111001100000000001101111010010110011111101111  
1111111100000111000111000111110011110000000110101111110  
000011101001100100111110111110000111111001100110001011  
100111110000110001100110101111001111100010111010111111  
100100111111110011101111000111111000110111110001111110  
110111101110101111011001111110011111110011111000100111  
1111100010010111100011000111110001111111111111110111  
1110111111100001110000010111100111111110000000111001100  
10100000111001111101111111110000000110001000011000  
11100111011011111111001011111011110111100000011111111  
1100110011000100001000111111110001111100100000100001000  
0000111110111001001110000111111011111111111000100111  
100001100110010111001000110001001101111000011000111111
```

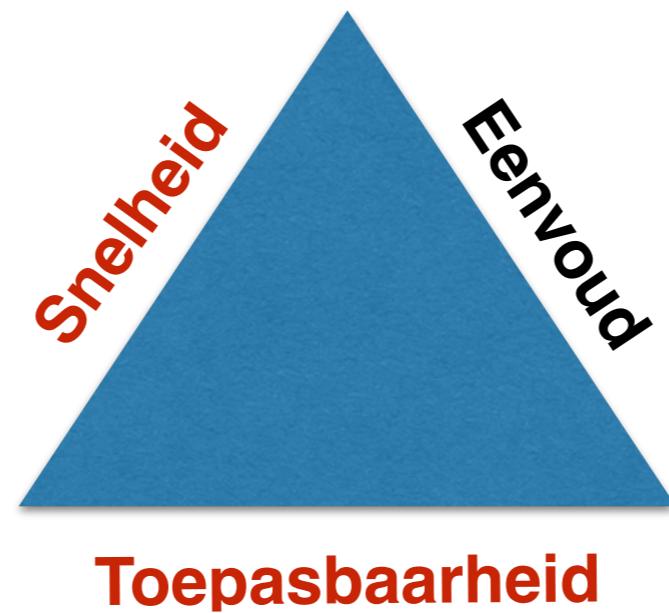




Toepasbaarheid

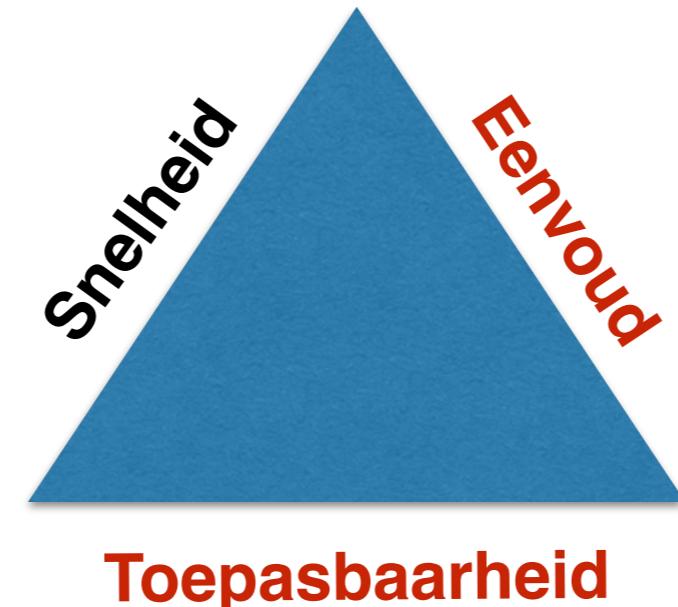
C

```
1 #include<stdio.h>
2
3 int main()
4 {
5     printf("Hello, World!\n");
6     return 0;
7 }
```



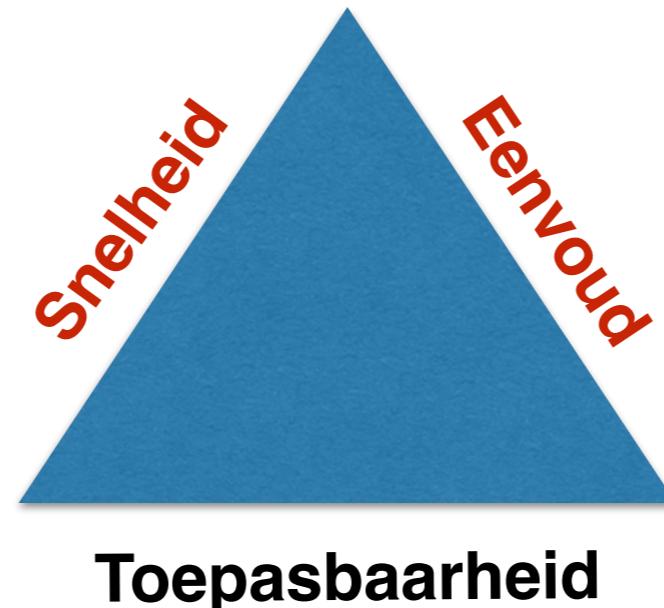
Python

```
1 print "Hello, World!"
```



Mathematica

```
1 Print["Hello, World!"]
```



Meer talen

- **Java**: snel, zeer breed inzetbaar, complex en ‘verbose’, ‘enterprise’
- **C++**: “C met meer”, nog sneller dan Java (games), ‘dicht op het metaal’, draait overal, maar nog complexer en makkelijk om fouten in te maken. Geen memory management.
- **Ruby**: vergelijkbaar met Python, bekend van Ruby on Rails.
- **Javascript**: voor webbrowsers, maar ook ‘op de server’. Zeer populair, maar niet altijd even eenvoudig. Jonge gemeenschap, dus niet altijd even volwassen en lastig om bij te houden.
- **PHP**: voor websites. Relatief makkelijk om snel mee te beginnen, lastig om ‘mooi’ in te schrijven, je schiet jezelf snel in de voet. Slechte naam.
- **Objective-C / Swift**: taal binnen Apple-wereld. Gebaseerd op C.

Waarom Python + Dataverwerking?

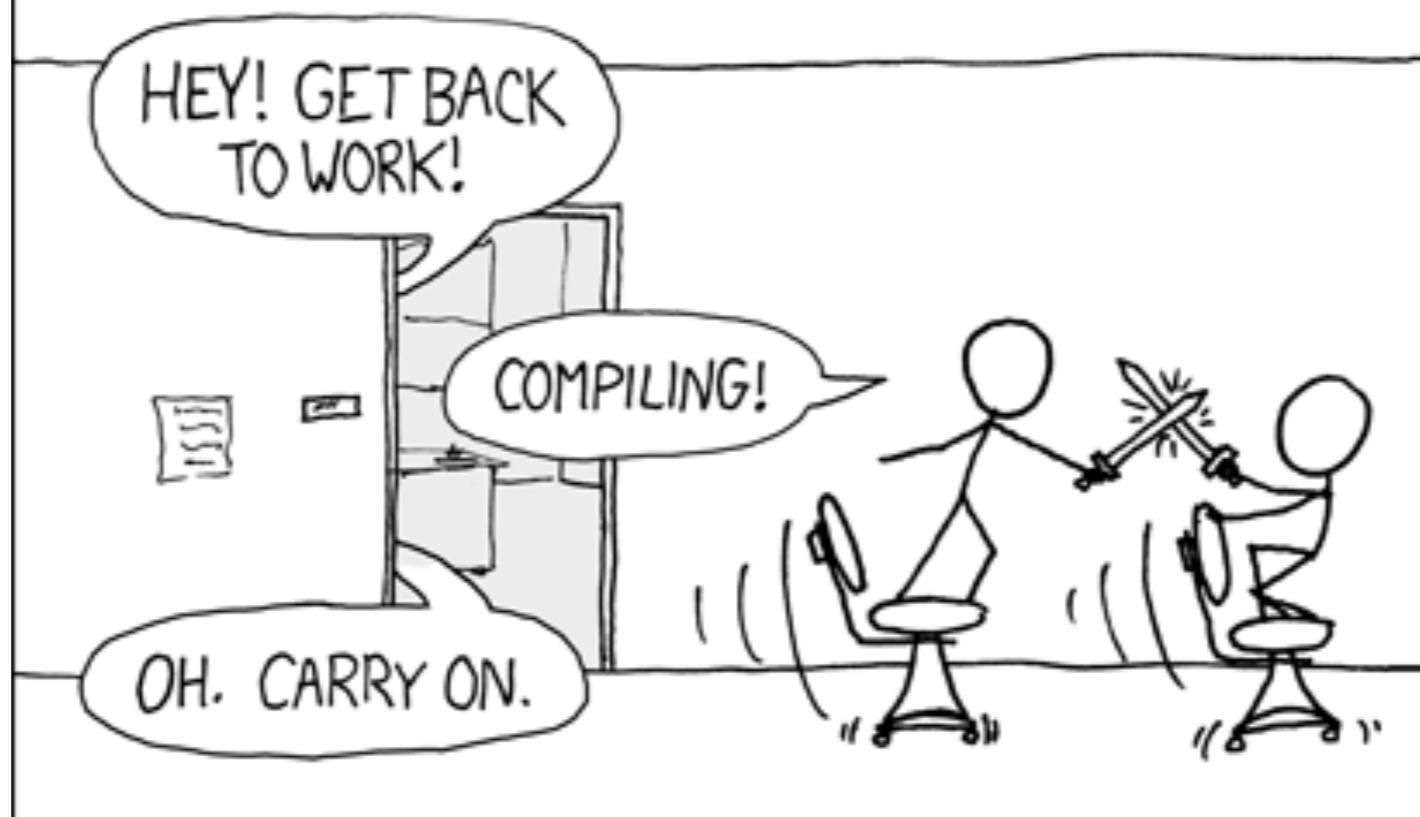
- Eenvoudig, duidelijke syntax.
- Weinig manieren om jezelf in de voet te schieten.
- Niet snel, maar dat maakt voor ons weinig uit.
- Geen gebruikers! Hoera!
- Data gaat in en uit.
- Goede standaard library.
- Veel documentatie en voorbeelden te vinden (StackOverflow)
- Let op voor Python 2 vs Python 3
- Hollandse glorie!



Het grote geheim over programmeurs

THE #1 PROGRAMMER EXCUSE
FOR LEGITIMATELY SLACKING OFF:

"MY CODE'S COMPILING."



Programmeurs zijn lui, ongeduldig en verwaand

Three Virtues

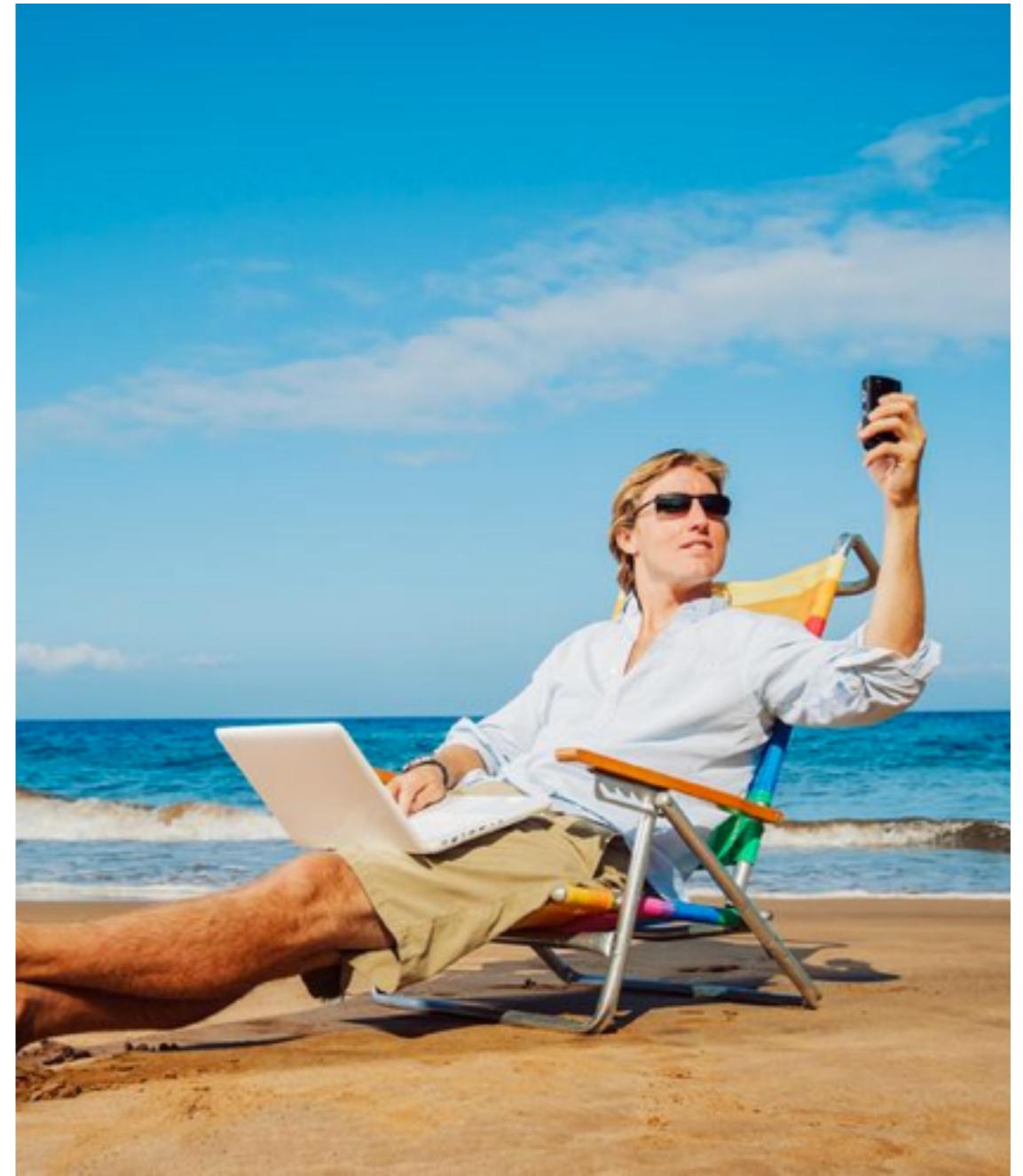
According to Larry Wall⁽¹⁾, the original author of the Perl programming language, there are **three great virtues of a programmer**; Laziness, Impatience and Hubris

1. **Laziness**: The quality that makes you go to great effort to reduce overall energy expenditure. It makes you write labor-saving programs that other people will find useful and document what you wrote so you don't have to answer so many questions about it.
 2. **Impatience**: The anger you feel when the computer is being lazy. This makes you write programs that don't just react to your needs, but actually anticipate them. Or at least pretend to.
 3. **Hubris**: The quality that makes you write (and maintain) programs that other people won't want to say bad things about.
-

(1) Quoted from "Programming Perl", 2nd Edition, O'Reilly & Associates, 1996

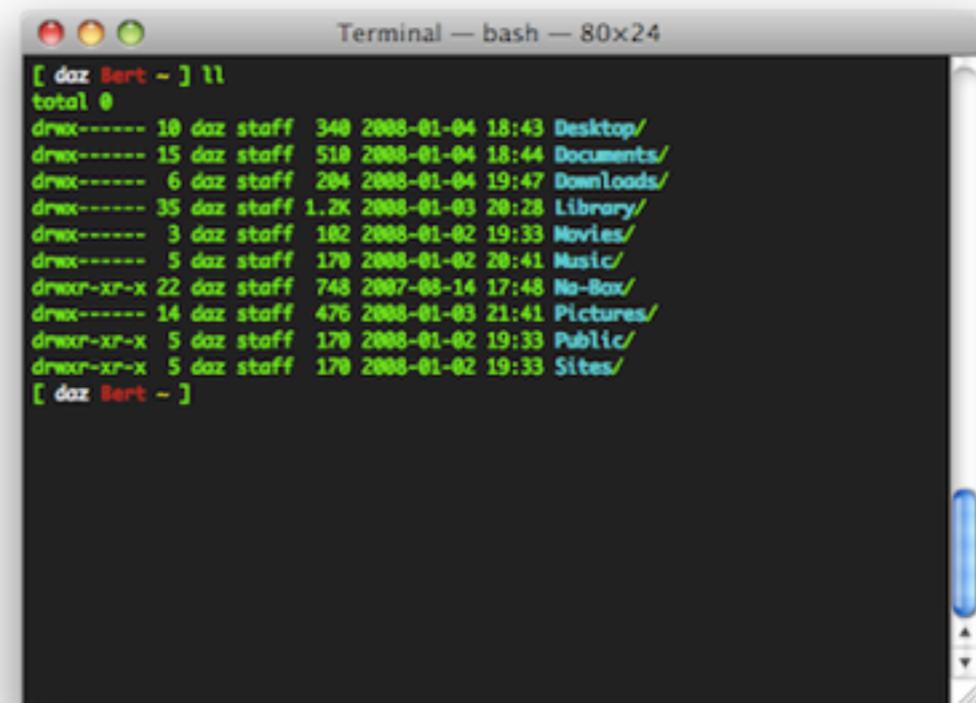
Luiheid

- Nooit jezelf herhalen.
- Alles automatiseren.
- Geen code schrijven die al door anderen is geschreven.
- Geen tools maken die al bestaan.
- Hoe minder regels code, hoe beter.
- Wat je niet weet opzoeken met Google.



Terminaal...

- Je kan alles automatiseren met de terminal.
- Automatisch dingen laten draaien.
- Combinatie Python + terminal is ideaal voor dataverwerking.
- Je kan alles in Python doen, maar sommige dingen gaan beter met tools via de terminal.
- Behalve op Windows.



A screenshot of a Mac OS X Terminal window titled "Terminal — bash — 80x24". The window shows a command-line interface with the following output:

```
[daz Bert ~] 11
total 0
drwx----- 10 daz staff 340 2008-01-04 18:43 Desktop/
drwx----- 15 daz staff 510 2008-01-04 18:44 Documents/
drwx----- 6 daz staff 204 2008-01-04 19:47 Downloads/
drwx----- 35 daz staff 1.2K 2008-01-03 20:28 Library/
drwx----- 3 daz staff 182 2008-01-02 19:33 Movies/
drwx----- 5 daz staff 170 2008-01-02 20:41 Music/
drwxr-xr-x 22 daz staff 748 2007-08-14 17:48 Na-Box/
drwx----- 14 daz staff 476 2008-01-03 21:41 Pictures/
drwxr-xr-x 5 daz staff 170 2008-01-02 19:33 Public/
drwxr-xr-x 5 daz staff 170 2008-01-02 19:33 Sites/
[daz Bert ~]
```



Aan de slag!

Download en unzip het zipje wat je hier kan vinden:

<http://hay.github.io/codecourse/>

helloworld.py

```
1 print "Hallo, welkom bij ons programma"
2 print "Eerst wil ik je naam weten"
3 name = raw_input("Wat is je naam?")
4 print name
5 print "Hallo!"
6 print "Nu wil ik je leeftijd weten."
7 age = raw_input("Hoe oud ben je?")
8 oldage = age * 2
9 print oldage
10 print "Oud zeg!"
```

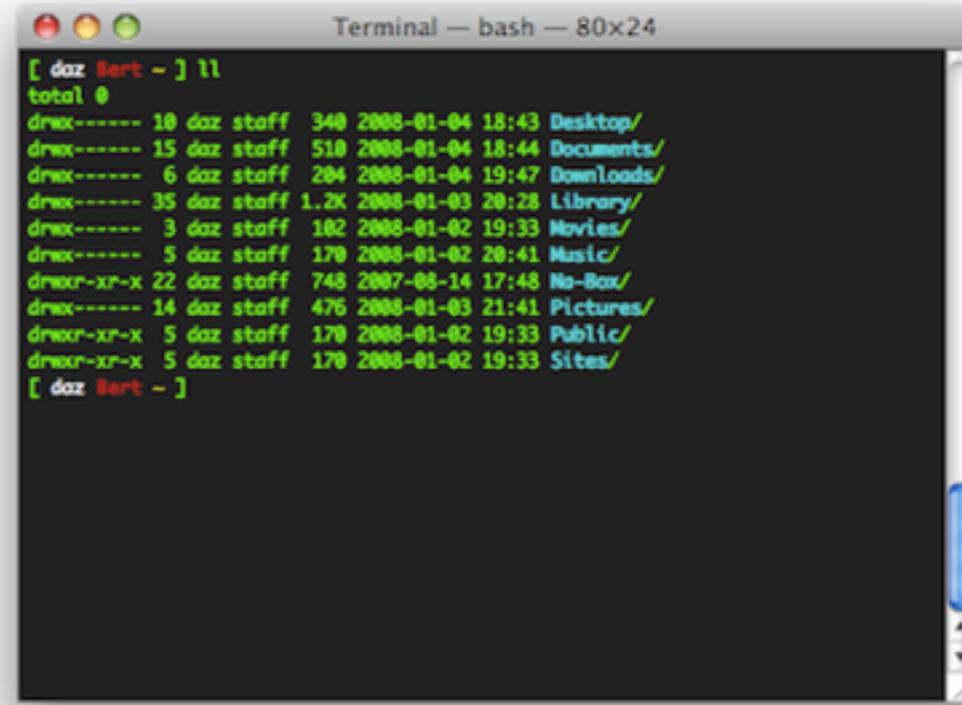


Binair



Tekst

ASCII value	Character	Control character	ASCII value	Character	ASCII value	Character	ASCII value	Character
000	(null)	NUL	032	(space)	064	@	096	
001	☺	SOH	033	!	065	A	097	ä
002	☻	STX	034	"	066	B	098	ß
003	♥	ETX	035	#	067	C	099	ç
004	♦	EOT	036	\$	068	D	100	đ
005	♣	ENQ	037	%	069	E	101	é
006	♠	ACK	038	&	070	F	102	í
007	(beep)	BEL	039	'	071	G	103	g
008	■	BS	040	(072	H	104	h
009	(tab)	HT	041)	073	I	105	i
010	(line feed)	LF	042	*	074	J	106	j
011	(home)	VT	043	+	075	K	107	k
012	(form feed)	FF	044	-	076	L	108	í
013	(carriage return)	CR	045	-	077	M	109	m
014	♫	SO	046	.	078	N	110	n
015	☼	SI	047	/	079	O	111	ö
016	▬	DLE	048	0	080	P	112	p
017	▬	DC1	049	1	081	Q	113	q
018	↕	DC2	050	2	082	R	114	r
019	!!	DC3	051	3	083	S	115	s
020	π	DC4	052	4	084	T	116	t
021	§	NAK	053	5	085	U	117	u
022	▬	SYN	054	6	086	V	118	v
023	↑	ETB	055	7	087	W	119	w
024	↑	CAN	056	8	088	X	120	x
025	↓	EM	057	9	089	Y	121	y
026	→	SUB	058	:	090	Z	122	z
027	←	ESC	059	;	091	[123	{
028	(cursor right)	FS	060	<	092	\	124	:
029	(cursor left)	GS	061	=	093]	125	}
030	(cursor up)	RS	062	>	094	^	126	-
031	(cursor down)	US	063	?	095	-	127	▀



A screenshot of a Mac OS X Terminal window titled "Terminal — bash — 80x24". The window shows a directory listing in green text on a black background. The listing includes the root directory and several subdirectories like Desktop, Documents, Downloads, Library, Movies, Music, Na-Box, Pictures, Public, and Sites. The output ends with "[daz Bert ~]".

```
[daz Bert ~] 11
total 0
drwx----- 10 daz staff 340 2008-01-04 18:43 Desktop/
drwx----- 15 daz staff 510 2008-01-04 18:44 Documents/
drwx----- 6 daz staff 204 2008-01-04 19:47 Downloads/
drwx----- 35 daz staff 1.2K 2008-01-03 20:28 Library/
drwx----- 3 daz staff 102 2008-01-02 19:33 Movies/
drwx----- 5 daz staff 170 2008-01-02 20:41 Music/
drwxr-xr-x 22 daz staff 748 2007-08-14 17:48 Na-Box/
drwx----- 14 daz staff 476 2008-01-03 21:41 Pictures/
drwxr-xr-x 5 daz staff 170 2008-01-02 19:33 Public/
drwxr-xr-x 5 daz staff 170 2008-01-02 19:33 Sites/
[daz Bert ~]
```

De tekstgebaseerde terminal
Alles is tekst in UNIX

1. Python

```
[hay@haybook examples] (gh-pages)$ cd ~/htdocs/git/codecourse/examples/
[hay@haybook examples] (gh-pages)$ ls
advancedtypes.py          files.py           people.txt
basictypes.py              functions.py      string.py
compare.py                 helloworld.py
convert.py                 ifloop.py
[hay@haybook examples] (gh-pages)$ python helloworld.py
Hallo, welkom bij ons programma
Eerst wil ik je naam weten
Wat is je naam?■
```

```
cd                      # Change directory
ls                      # List
ls -l                  # List als een lijst
pwd                     # Print working directory
python bestand.py      # Voer bestand.py uit
<TAB>                 # Vul aan (handig voor bestandsnamen en paden)
<Pijltje omhoog>     # Ga door vorige commando's heen
<ENTER>                # Voer uit
reset of <CTRL-L>     # Maak scherm leeg
```

```
1 print "Hallo, welkom bij ons programma"
2 print "Eerst wil ik je naam weten"
3 name = raw_input("Wat is je naam?")
4 print name
5 print "Hallo!"
6 print "Nu wil ik je leeftijd weten."
7 age = raw_input("Hoe oud ben je?")
8 oldage = age * 2
9 print oldage
10 print "Oud zeg!"
```

Basistypes

```
1  a = 42          # Integer, een heel getal
2  b = 20          # Nog een integer
3  c = 3.1415      # Float, een breuk
4  d = "Hallo"     # String, een reeks karakters
5  e = False        # Boolean, True of False
6  f = None         # None, niks :)

7
8  print a + b     # Optellen
9  print b - a     # Aftrekken
10 print c * a      # Vermenigvuldigen
11 print b % 3       # Modulo (rest)
12 print a / b       # Delen, hier gebeurd iets geeks...
13 print d + a       # Kan je ook strings optellen?
14
```

Conversies

```
1  a = 42
2  b = "42"
3
4  print a + int(b)
5
6  c = "Hay is zo oud: "
7  d = 32
8
9  print c + str(d)
```

Opdracht #1

- Schrijf een programma dat berekent hoe oud iemand is op basis van geboortejaar (hoeft niet heel precies)

```
1  print "Hallo, welkom bij ons programma"
2  print "Eerst wil ik je naam weten"
3  name = raw_input("Wat is je naam?")
4  print name
5  print "Hallo!"
6  print "Nu wil ik je leeftijd weten."
7  age = raw_input("Hoe oud ben je?")
8  oldage = age * 2
9  print oldage
10 print "Oud zeg!"
```

Vergelijkingen

```
1  a = 20
2  b = 30
3  c = 40
4  d = "Hay Kranen"
5  e = None
6
7  print a == a          # Is gelijk aan
8  print a != b          # Niet gelijk aan
9  print a > b           # Groter dan
10 print a < b            # Kleiner dan
11 print d == "Hay Kranen" # Vergelijkingen werken ook met strings
12 print b < c and b > a   # AND met twee vergelijkingen
13 print a < c or b < c    # OR met twee vergelijkingen
14 print not e             # 'Inverteer' de uitkomst
15
```

Opdracht #2

- Schrijf een programma dat van twee mensen de leeftijd vraagt en de volledige naam, en print wie er ouder is, en of iemand “Barrie” heet.

```
1  a = 20
2  b = 30
3  c = 40
4  d = "Hay Kranen"
5  e = None
6
7  print a == a          # Is gelijk aan
8  print a != b          # Niet gelijk aan
9  print a > b           # Groter dan
10 print a < b           # Kleiner dan
11 print d == "Hay Kranen" # Vergelijkingen werken ook met strings
12 print b < c and b > a  # AND met twee vergelijkingen
13 print a < c or b < c   # OR met twee vergelijkingen
14 print not e            # 'Inverteer' de uitkomst
15
```

Control structures

```
1  a = 20
2  b = 10
3
4  if a + b == 30:
5      print "Het getal is dertig!"
6  else:
7      print "Het getal is geen dertig"
8
9  for x in range(0, b):
10     print "Hallo"
```

Let op indenten en dubbele punten!

```
1  a = 20
2  b = 10
3
4  if a + b == 30:
5      print "Het getal is dertig!"
6  else:
7      print "Het getal is geen dertig"
8
9  for x in range(0, b):
10     print "Hallo"
```

Opdracht #3

- Schrijf een programma dat iemands naam vraagt en een getal. Als die persoon “Barrie” heet print je zo vaak “Welkom” als het getal. Als die persoon niet “Barrie” heet print je “Helaas, tot ziens”.

```
1  a = 20
2  b = 10
3
4  if a + b == 30:
5      print "Het getal is dertig!"
6  else:
7      print "Het getal is geen dertig"
8
9  for x in range(0, b):
10     print "Hallo"
```

Huiswerkopdracht

- * Introduceer de Swiebertje quiz
- * Vraag de naam van de gebruiker en heet 'm van harte welkom
- * De eerste vraag is: noem een willekeurig jaar waarin Swiebertje op tv was (elk antwoord tussen 1955 en 1975 is goed)
- * De tweede vraag is: Wie speelde Swiebertje? Zowel de voornaam ("Joop") als de achternaam ("Doderer") is goed, maar niet de hele naam!
- * Print vervolgens het aantal goede en foute antwoorden (die moet je dus bij het hele programma bijhouden!)
- * Print dan of de gebruiker gewonnen, verloren, of evenveel goede als foute antwoorden had met een bijpassend compliment of verwensing.

Je kunt deze quiz maken met alleen de kennis die je deze les hebt opgedaan bij opdrachten #1, #2 en #3. Wat je zeker nodig zult hebben:

- * Input van de gebruiker via `raw_input`
- * Booleaanse vergelijkingen (`and` / `or` / `groter dan` / `kleiner dan`)
- * `if` / `else`
- * Converteren tussen `string` en `integer` en vice versa
- * Concateneren van `strings`