hcc! programmeren

16 september 2017

Een rondleiding langs een aantal programmeertalen

Inleiding

Johan Volkers


ALGOL
UP
COMIT
APT
TRIC
MAD
FORTRANSIT
IPL V
COLASL
FORTRAN III
SOAP
COBOL
MCP
MATHMATIC
FACT
IPL
XZ
BIOR
VIPP
GAT
X1
SHADOW III
FLEXIMATIC
IT
SURGE
UNICODE
MYSTIC
API
ADES II
SLANG
CL-I
COMMERCIAL TRANSLATOR
APS III
CLIP
FORTRAN II
ARGUS
MISHAP
SOS
FLIP
SALE
NUIT
XTRAN
ABC
ALMACO
GOOFUS
STRAP II
LISP
ALGO
THREADED LISTS
NELIAC
BALGOL
TAC
ALTAC
MADCAP
CAGE
MAGIC
APX III
PRINT
ALTRAN
ACT
BACAIC
STRAP I
PACT I
SMAC
FORTRAN
FORAST
UNCOL
9 PAC
SAP
USE
IVY
BABEL
1960

Keuze genoeg...

Aargh! ABAP ABC ActionScript Ada Algol Algol-58 Algol-60 Algol 68 Algol-W Amiga APL AppleScript Arduino Argh! Assembleertalen AutoHotkey AutoIt AWK BASIC BBC BASIC BCPL Blitz Basic Brainfuck C C# C++ CFML Clean Clipper Clojure COBOL D DarkBASIC Decimal BASIC Delphi Delta COBOL DIV Games Studio EGL Eiffel Erlang Euphoria F# Fenix Project Forth Fortran Gambas GCL GML Go Groovy Haskell HyperTalk (de scripttaal van HyperCard) Inform IronPython Java JavaScript JScript Just Another Language Just BASIC Kotlin LabVIEW Leet Liberty BASIC Lisp Logo Lua m4 Machinetaal Malbolge M-code ML Modula-2 Mondrian mSL Oberon Objective-C ObjectPAL Object Pascal Ocaml Oz Pascal Pawn Perl PHP Piet PL/1 PL/pgSQL PL/SQL PostScript PowerBASIC PowerBuilder Processing Progress Prolog ProvideX PureBasic Python QBasic R Rexx RPG Ruby Rust SAS Scala Scheme Scratch sed Seed7 Self Simula SmallBASIC Smalltalk SNOBOL Swift Tcl TeX TI-BASIC T-SQL Turbo BASIC Turbo C Turbo Pascal UBasic Vala VBScript Visual Basic Visual C Visual Objects Whitespace XQuery XSLT YAML

En dat zijn ze echt niet allemaal…..

Doel van programmeren

Zorgen dat een computer doet wat *jij* wilt

Geschiedenis

1e en 2e generatie

Machinetaal

```
03067153215        # kopieer inhoud geheugenlokatie 6715 naar 3215
12190233215        # verhoog met de constante op plaats 9023
```

Assembleertalen – symbolische weergave machineinstructies
Namen voor geheugenlokaties

```
MOV     VAR1,TELLER
ADD     CONST1,TELLER
```

3e generatie programmeertalen: hoger abstractieniveau

Cobol

```
MOVE WAARDE1 TO TELLER.
ADD 1 TO TELLER.
```

Fortran

```
TELLER = WAARDE + 1
```

Algol

```
teller := waarde1 + 1;
```

4e generatie programmeertalen

Hoger abstractieniveau tbv speciaal soort problemen

SQL: benaderen van relationele databases

```
SELECT aantal,omschrijving FROM artikelen
    WHERE artikelnummer > 1000 AND
    Aantal != 0;
```

Postgress 4GL

Snelle ontwikkeling van business-applicaties

5e generatie: probleemoplossende programmeertalen

Beschrijf hoe het probleem opgelost moet worden

vs

Beschrijf het probleem

Kunstmatige intelligentie: Prolog

Compileren vs interpreteren

Compileren
Programma wordt in een aantal stappen omgezet in een programma in machinecode
Bijv. C, C++, Fortran

Interpreteren
Programma wordt rechtstreeks uitgevoerd
Bijv. Python, Perl, Ruby

```
edit myprog.c
# compileer en link
gcc -output=myprog myprog.c
# run
myprog
```

```
edit myprog.py
# run
python myprog.py
```

Voordeel van gecompileerde programma's

Snelheid van uitvoering
– hoe belangrijk is dat nog?

Voordeel van geïnterpreteerde talen

– taal aanmerkelijk krachtiger
– veel snellere ontwikkelcyclus
  edit run
  vs
  edit compile/link run ...

Java

– wordt vertaald naar machinecode voor een virtuele machine
– kan daardoor op alle platforms draaien waar een VM beschikbaar is

Programmeerparadigma

Imperatief programmeren

C, Fortran, etc

Object-georienteerd programmeren

Gegevens en de bewerkingen op die gegevens worden verpakt in objecten

Puur object-georienteerd: Smalltalk, Eiffel
Bij voorkeur OO: C++, Ruby, Python
Ondersteunen OO: Perl, PHP

Functioneel programmeren

Lisp, Haskell, F#

Logisch programmeren

Prolog

Niet in de presentaties: Perl

Geïnterpreteerde scripttaal
Krachtig en efficient
Vooral sterk in het verwerkend van tekstfiles etc.

Motto: Perl maakt eenvoudige dingen eenvoudig
maar maakt ingewikkelde dingen niet onmogelijk

Heeft de naam onleesbare code op te leveren

Een slechte programmeur schrijft in iedere taal onleesbare code

Een goede programmeur houdt rekening met mensen die later nog eens naar de code moeten kijken.

Dat zou hij zelf wel eens kunnen zijn….

```
#!  /usr/bin/perl

open(INVOER,"<",$invoerbestand)
    or die "Openen $invoerbestand lukt niet!\n";
while(<INVOER>) {
    VerwerkRegel($_) if /^invoer/;
}
close INVOER;
```

Niet in presentaties: PHP

Speciaal ontwikkeld voor web-applicaties.
Biedt de mogelijkheid om code en HTML te mengen
(al is het beter om daar terughoudend in te zijn)

```
<?php
$nu=HaalTijdOp();
?>
<H1>Het is nu <?php echo $nu;?></H1>
```

Waarmee beginnen?

Waarmee beginnen?

Incidenteel programmeren: Basic, Scratch

Waarmee beginnen?

Incidenteel programmeren: Basic, Scratch
Opstap naar professioneel programmeren: Python

Marius Smits

Scratch

een taal om mee te beginnen

Spreek jij Scratch?

Scratch

een taal om mee te beginnen

Ontstaan van de taal Scratch

– programmeertaal voor onderwijsdoeleinden

– geschreven The Lifelong Kindergarten Group (MIT)

– gericht op jonge mensen vanaf 8 jaar

– scratching techniek gebruikt door disc-jockeys

Scratch

een taal om mee te beginnen

Probleemdomein

– omgaan met geautomatiseerde systemen

– bijbrengen van vaardigheden

– zoals creatief denken en probleemanalyse

– trainen in het doelgericht werken en samenwerken

Scratch

een taal om mee te beginnen

Voordelen en nadelen t.o.v. andere talen

– Scratch primair bestemd voor jonge mensen

– taal wordt visueel samengesteld

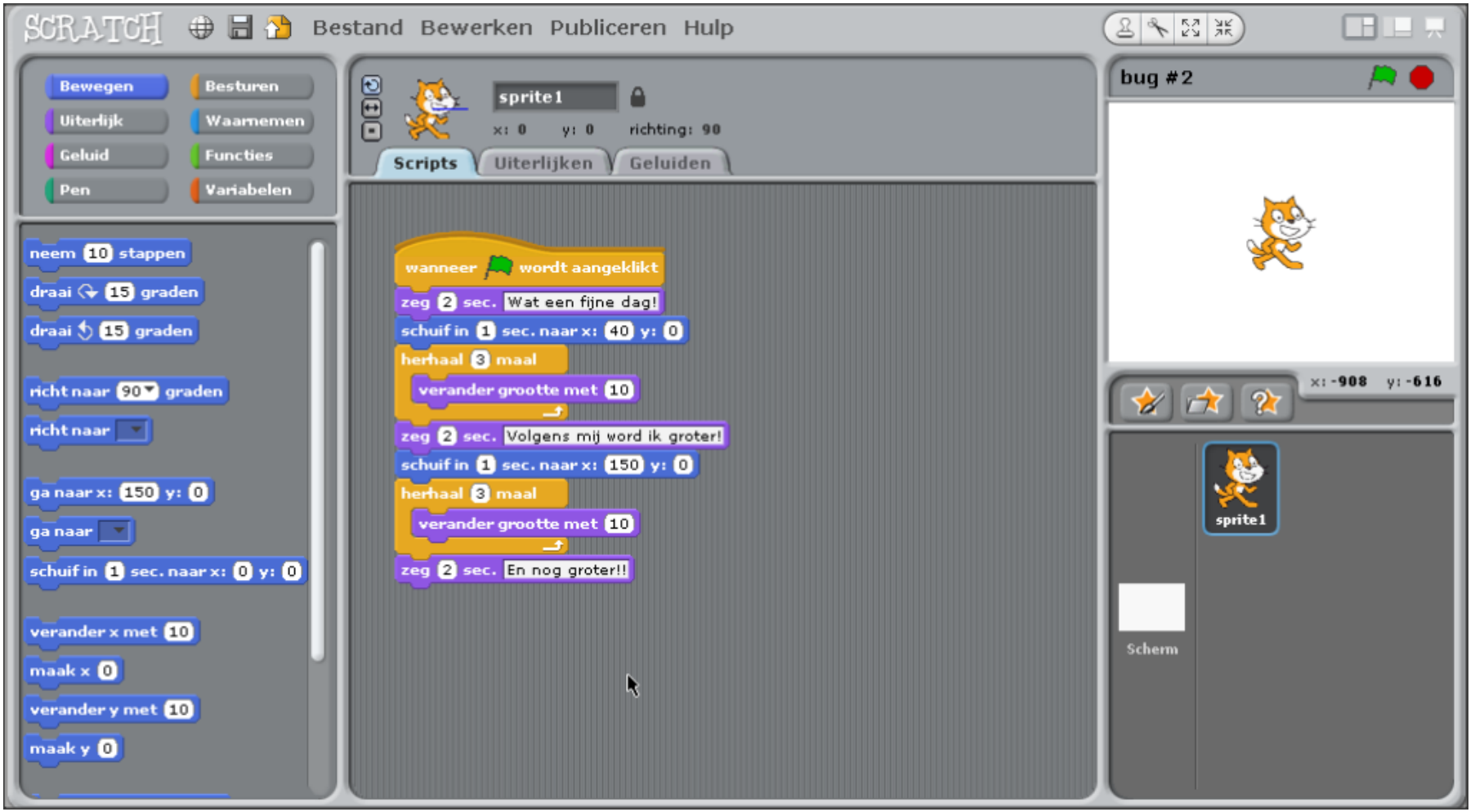
– bijbrengen gevoel voor de mogelijkheden en beperkingen

– ontwikkelen creatief denken en probleemanalyse

Scratch

een taal om mee te beginnen

Een eenvoudig voorbeeld



Scratch

een taal om mee te beginnen

URL’s voor meer informatie

– https://scratch.mit.edu

– https://www.google.nl/search?q=scratch

Scratch

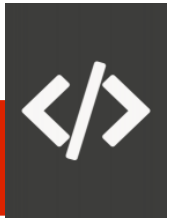
een taal om mee te beginnen

Spreek jij Scratch?

Vloeiend ...

hcc programmeren

Python



Theo van Haandel

Python

Inhoud

- Ontstaansgeschiedenis
- Kenmerken
- Voordelen
- Nadelen
- Voorbeelden
- Beschikbaarheid
- Versienummers
- Verschillen versies 2 en 3
- Referenties

Python

Ontstaansgeschiedenis

- Python is eind 80'er jaren bedacht en geïmplementeerd in C
- is gebaseerd op de ABC programmeertaal
- door Guido van Rossum bij het CWI in Amsterdam

- hij is nog steeds de Benevolent Dictator for Life (BDFL)
- eerste release versie 0.9.0 in 1991 met de CWI licentie
- kende al classes, inheritance, exceptions, list, dict, str

- in 2001 werd de Python Software Foundation (PSF) opgericht
- regelt de voortgang van de ontwikkeling van Python
- tegenwoordig grote Python community enthousiaste gebruikers

Python

Versie historie A

eigenaar	jaar	v1	v2	v3	Enkele features
CWI	1991	0.9.0			eerste release: classes, modules, exceptions, list, dict, str
CWI	1994	1.0			
CNRI	1999	1.5.2			

Python

Versie historie A

eigenaar	jaar	v1	v2	v3	Enkele features
CNRI, BeOpen	2000	1.6	2.0		unicode, garbage collection
CNRI, PSF	2001	1.6.1	2.1, 2.2		geneste scopes, weak references, descriptors, generators
PSF	2003		2.3		set datatype, source encoding, zip file import
PSF	2004		2.4		decorators, decimal datatype
PSF	2006		2.5		conditionele expressies, with statement
PSF	2008		2.6	3.0	unicode, print functie, string functie, migratie

Python

Versie historie B

eigenaar	jaar	v1	v2	v3	Enkele features
PSF	2009			3.1	ordered dictionary
PSF	2010		2.7		sommige features van Python 3
PSF	2011			3.2	stabiele ABI voor C modules
PSF	2012			3.3	yield from, virtual environments, exception hierarchy
PSF	2014			3.4	enum, asyncio, ensurepip: pip voor externe packages installatie
PSF	2015			3.5	coroutines: async await, matrix @ operator
PSF	2016			3.6	literals: formatted string en underscore
PSF	2017		2.7.14	3.6.2	security

Python

Kenmerken A

- script taal
- ontwikkeld om de productiviteit van programmeurs te verhogen
- interpreter, minder snel dan gecompileerde talen

- geen declaraties nodig, types worden afgeleid uit de context
- type-safe, met sterke, dynamische types
- een ongeldige bewerking resulteert in een exception

- ondersteunt verschillende manieren van programmeren: procedureel, object geörienteerd, functioneel

Python

Kenmerken B

Gebruikt voor:

- systeembeheer, (in batch) verwerken van bestanden
- wetenschappelijke software: NumPy, SciPy, Jupyter notebooks
- websites, WSGI interface, Django, Flask, Apache met mod_wsgi

Python

Kenmerken B

De naam komt van Monty Python's Flying Circus, verwijzingen:

- the Cheeseshop
- Unladen Swallow
- Spam and Eggs

Er zijn ook verwijzingen naar de soort slang:

- het logo van Python:
- Anaconda
- Boa Constructor

Python

Voordelen (A)

- code eenvoudiger, leesbaarder, compacter
- geen declaraties
- geen braces { } of begin/end, indendatie met spaties en tabs
- datatypes met weinig beperkingen
- bijvoorbeeld int, gehele getallen, kan veel groter dan 128 bits
- verschil maken tussen 32/64 bits of signed/unsigned is niet nodig

Python

Voordelen (B)

- "batteries included", standaard meer dan 200 modules/packages
- veel meer packages, meer dan 160.000 in Python Package Index
- portable, voor Windows, Mac OS X, Linux, ook Raspberry Pi
- bestaat ook voor Android en Apple iOS, minder gebruikelijk

- documentatie is uitgebreid en van hoge kwaliteit
- veel aanvullende informatie : boeken, websites

Python

Nadelen (A)

- geen goede multithreading vanwege Global Interpreter Lock

- geen goede standaard GUI module, Tcl/Tk toolkit niet ideaal
- alternatieve GUI modules, gebaseerd op GTK, Qt of wxWidgets, zijn minder 'Pythonic'
- interpreter, niet zo snel, geen hoge performance, behalve bij sommige C modules zoals NumPy

Python

Nadelen (B)

- te veel informatie, te veel keuzes, luxe probleem

- niet zo veel informatie in het Nederlands
- wel gratis Nederlandstalig boek "De Programmeursleerling: Leren coderen met Python 3" geschreven door Pieter Spronck, gratis te downloaden in pdf formaat, ook in het Engels

Python

Voorbeelden prompt / IDLE

```
>>> print("Hello, world!")
Hello, world!

>>> print(hex(2 ** 142 + 1))
0x40000000000000000000000000000001

>>> (3 + 2j) * (2 + 3j)
13j

>>> txt = "z # t 3 A 5 7 2"
>>> lijst = txt.split()
>>> lijst.sort()
>>> txt2 = '.'.join(lijst)
>>> print(txt2)
#.2.3.5.7.A.t.z
```

Python

Voorbeeld programma run

In bestand test.py:

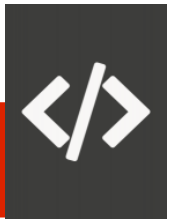
```
with open('test.py') as infile:
  for line in infile:
      print(line.upper(), end = '')
```

Resultaat uitvoeren test.py op Windows 10:

```
D:\Test>py test.py

WITH OPEN('TEST.PY') AS INFILE:
    FOR LINE IN INFILE:
        PRINT(LINE.UPPER(), END = '')
```

Python



Beschikbaarheid

Downloads voor:

- Windows, 32 bits (x86) en 64 bits (x86-64)
- Mac OS X
- source code, voor Linux ook in meeste Linux distributies

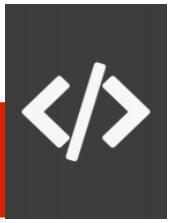
Andere Python distributies, onder andere:

- Anaconda
- Python(x, y)
- ActivePython

Alternatieve implementaties, bijvoorbeeld:

- PyPy (JIT implementatie, geschreven in Python)
- MicroPython (voor microcontollers, geschreven in C)
- IronPython (voor .NET, geschreven in C#)
- Jython (voor de Java Virtual Machine (JVM), geschreven in Java)

Python



Versienummers

- aangeduid door: major.minor.micro, zoals 3.6.2
- soms met a (alpha), b (beta) of rc (release candidate): 2.7.14rc1
- major versie 2 bijvoorbeeld 2.3 of 2.7, major versie 3 o.a. 3.6

- verschillende major.minor versies tegelijk te installeren

- een hogere micro versie is een bug fix release
- overschrijft een lagere micro versie: 3.6.2 is een verbeterde versie, gaat over 3.6.1

Python

Verschillen versies 2 en 3

- code voor Python 3 meestal niet uitwisselbaar met Python 2
- nog steeds veel code voor Python 2 in gebruik
- daarom Python 2.7 nog steeds ondersteund, tot 2022

- beter om Python 3 te gebruiken indien mogelijk
- Python 2 alleen voor modules die geen Python 3 ondersteunen
- meestal voor deze modules geen updates meer

Veel verschillen tussen Python 2 en 3, belangrijkste:

- print is statement in Python 2, functie in Python 3
- verschillen in string types, Unicode support

Python

Referenties

- https://www.python.org/ de standaard Python website
- https://docs.python.org/ de uitgebreide Python documentatie
- https://www.python.org/downloads/release/python-362/ de nieuwste Python 3.6.2 downloads
- https://pypi.python.org/pypi de Python Package Index, met meer dan 116.000 packages
- http://www.spronck.net/pythonbook/dutchindex.xhtml gratis boek "De Programmeursleerling: Leren coderen met Python 3" geschreven door Pieter Spronck.

einde van dit onderwerp

Forth

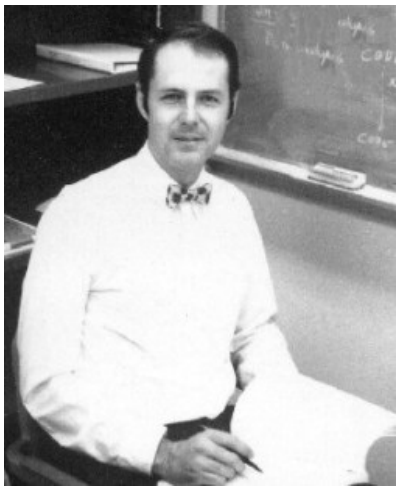
door Gerard Vriens

Forth: ontstaan en ontwikkeling

- Bedacht eind jaren '60 door Charles H. (“Chuck”) Moore
- Had eigenlijk “Fourth” moeten heten
- Duidelijke invloeden van de AI-taal Lisp
- Vanaf 1970 verder ontwikkeld, en gepopulariseerd

m.m.v. Elisabeth D. (“Bess”) Rather

- Standaarden:
  - Forth79
  - Forth83
  - ANS-Forth (ANSI 1994)



Forth: nadelen

- *Command line interface*, dus niet grafisch
- *Reverse Polish* notatie: “even wennen”
- Andere denkwijze (o.a. door stack)
- Geen “strakke” syntax
- Veel “woordkennis” nodig
- Geen ingebouwde *floating point*
- Weinig (standaard)bibliotheken
- Veel “eigen” Forth-versies

Forth: voordelen

- Ook een (primitief) operating systeem
- Compact
  - Qua systeem
  - Qua programma's
- Snel
- Interactief
- Interpretatie en (automatische) compilatie
- Er bestaan speciale Forth processoren

Forth: toepassingsgebieden

Forth is met succes gebruikt voor o.a.

- Hardwarebesturing
  - Eerste toepassing: *Kitt Peak* radiotelescoop
  - Philae komeetlander (ESA 1992-2015)
- Embedded systems
  - Microcontrollers (TI Launchpad)
- Systeemprogrammering
  - Vaak gebruikt voor eerste programmering nieuwe hardware:
    Intel 8086 chip (1978), Apple MacIntosh (1984)
- Numeriek en wetenschappelijk programmeren
- Games
  - Starflight (DOS)

Forth: voorbeeld

Bereken de grootste gemene deler (GGD) van twee gehele getallen:

```
80386 ciforth 5.1
119 544 GGD .
119 544 GGD ? ciforth ERROR # 12 : NOT RECOGNIZED

: GGD
  BEGIN
    DUP 0= INVERT
  WHILE
    OVER OVER MOD ROT DROP
  REPEAT
  DROP ;
 OK
119 544 GGD .
17  OK
```

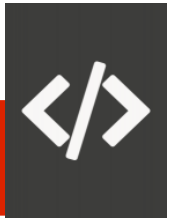
Forth: nadere informatie

- Wikipedia
  - Volg desgewenst ook de weblinks!
- HCC Forth interessegroep:
  - Bijeenkomsten in Bilthoven, op de tweede zaterdag van elke *even* maand
  - forth.hcc.nl
  - Weblinks onder “Contact ⟶ Websites”
  - Vijgeblad-archief onder “Contact ⟶ Partnersites”

Dank voor uw aandacht!

Vragen?

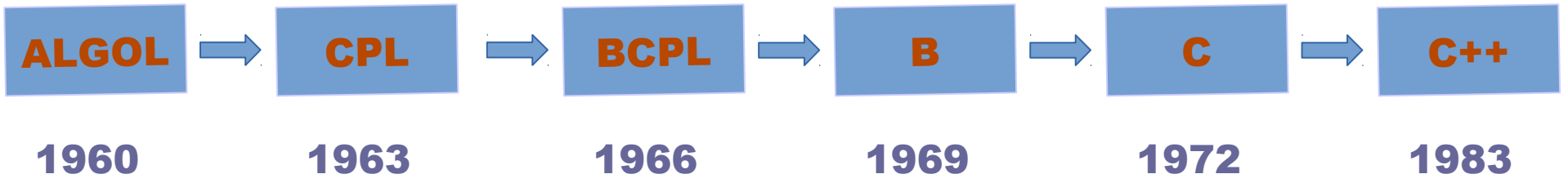
C - de standaard programmeertaal



Daniel von Asmuth

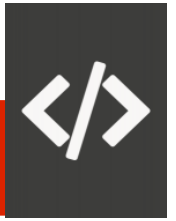
C - de standaard programmeertaal

Het ontstaan van de taal



C is de taal waarin het grootste deel van het UNIX besturingssysteem is geschreven - zowel de kernel als de utilities.

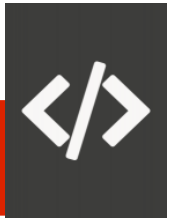
C - de standaard programmeertaal



Toepassingen

- computers en elektronische apparaten
- industrie
- ruimtevaart
- wetenschap en onderwijs
- financien en administratie
- Internet en telefonie
- spelletjes en simulaties
- gezondheidszorg
- defensie, politie, spionage en cybercriminaliteit
- media
- overheid
- logistiek
- land- en tuinbouw

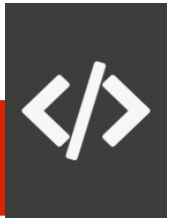
C - de standaard programmeertaal



Kenmerken

- scheiding tussen macroprocessor en compiler
- imperatief (sequentieel, iteratief)
- gebaseerd op functies (recursief)
- blok-gestructureerd en modulair
- statische typen - lakse typecontrole
- standaard bibliotheek optioneel

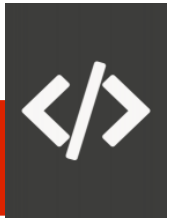
C - de standaard programmeertaal



Voordelen!

- **groen:** zuinig met geheugen en rekenkracht
- **portabel**: draait op alle moderne processoren
- **geschikt** voor alle toepassingen, high- en low-level, dankzij OS integratie
- **ondersteunt** meta-programmeren
- **stabiel** (geen problemen met versies) en gestandaardiseerd
- goede en goedkope compilers van **diverse** leveranciers
- **compact**: gemakkelijk te leren en implementeren
- dankzij Open Source veel **software** beschikbaar

C - de standaard programmeertaal

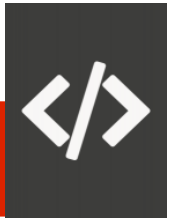


Nadelen?

- **beknoptheid** gaat boven leesbaarheid
- er bestaan veel ver**schillende** cod**eerst**ijlen
- **geheugenbeheer:** meer werk voor de programmeur, minder voor de processor
- **portabiliteit:** code die op het ene systeem goed werkt, geeft problemen op een ander platform
- C geeft je veel **vrijheid** en weinig controles, zodat slordige programmeurs code met **bugs** erin schrijven

Er wordt al decennia gewerkt aan een betere programmeertaal, maar er is er nog geen die helemaal voldoet.

C - de standaard programmeertaal

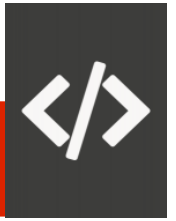


1-voudig voorbeeld

```
int main( void)
{
   1,2,3,4;
   "Hoedje van papier";
   return 0;
}
```

Een C programma heeft altijd een functie die ‘main’ heet en meestal andere functies aanroept. Hier wordt een vijftal expressies uitgerekend, waarna de functie 0 retourneert om aan te geven dat het programma eindigde zonder fouten.

C - de standaard programmeertaal



2-voudig voorbeeld

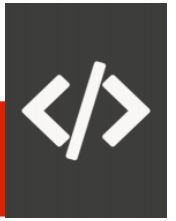
```
#include <stdio.h>
#include <math.h>

void main( void)
{
   float        pi, (*f) (float x);

   f = atanf, pi = 4.0 * f( 1.0);
   printf( "De waarde van pi is ongeveer %f.\n", pi);
}
```

Hier wordt variabele *f* gedeclareerd als een floating point functie, krijgt de arctangens als waarde toegewezen, die vervolgens wordt aangeroepen en het resultaat afgedrukt.

C - de standaard programmeertaal



3-voudig voorbeeld

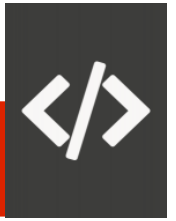
```
#include <stdio.h>
#include <math.h>

void main( void)
{
  float        pi = 4.0 * atanf( 1.0);
  float        *pipo = &pi;

  printf( "De waarde van pi is ongeveer %f.\n", *pipo);
}
```

Een eenvoudige illustratie van het gebruik van pointers.

C - de standaard programmeertaal

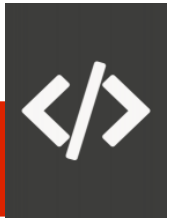


4-voudig voorbeeld

```
typedef          struct _node  /* linked list element */
{ void           *info;
  struct _node   *next;
} NODE, *NODE_P;
```

Recursieve datatypes worden element voor element verwerkt met behulp van pointers; merk op dat het type ‘struct _node’ een element bevat van het type ‘poiner naar struct _node’.

C - de standaard programmeertaal

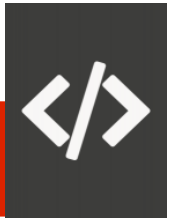


5-voudig voorbeeld

```
{
    char            hallo[] = {'W','e','l','k','o','m',
                               ' ','b','i','j',' ','d','e',' '};
    char            hcc[] = "Hobby Computer Club";
    char            b = 'a' + 1;
}
```

Het kleinste integer type kan een karakter uit het ASCII alfabet bevatten. Hier worden twee strings gedeclareerd; de lengte volgt uit de waarde waarmee ze worden geinitialiseerd. Een string bevat altijd een NUL als laatste element.

C - de standaard programmeertaal



beginnen met C

C Compilers zijn beschikbaar voor de meeste 4-bits, 8-bits, 16-bits, 32-bits, 64-bits architecturen, zowel Open als Closed Source; voor embedded targets bestaan cross-compilers. De meeste compilers ondersteunen zowel C als C++.

Om C te leren is de tweede editie van 'The C programming language' van Kernighan & Ritchie nog steeds een aanrader voor wie al enige programmeerkennis heeft. De officiële C11 standaard is te koop voor wie het exact wil weten. Voor beginners wordt “Een methode van programmeren” van Dijkstra & Feijen (ook wel: “A method of programming”) aangeraden.

einde van dit onderwerp