

PROGRAMMEREN VOOR KINDEREN

LEER STAP VOOR STAP PROGRAMMEREN
EN JE EIGEN COMPUTERGAMES MAKEN



DK Londen

Redactie Sam Priddy, Sam Atkinson, Lizzie Davey, Daniel Mills, Ben Morgan, Maud Whatley

Design Fiona Macdonald, Simon Murrell, Laura Brim, Sophia MTT

Hoofdredactie Paula Regan, Owen Peyton Jones

Productie Ben Marcus, Mary Slater

Uitgever Sarah Larter

Art director Phil Ormerod

Associate publishing director Liz Wheeler

Publishing director Jonathan Metcalf

DK India

Redactie Devika Dwarkadas, Suefa Lee, Neha Pande, Sanjay Chauhan,
Shreya Anand Virmani, Vanya Mittal

Design Sachin Gupta, Suhita Dharamjit, Harish Aggarwal

Hoofdredactie Rohan Sinha, Sudakshina Basu

Productie Balwant Singh

Oorspronkelijke titel *Computer Coding for Kids*

Oorspronkelijke uitgever Dorling Kindersley Limited, Londen

Copyright © 2014 Dorling Kindersley Limited, Londen

Copyright © Nederlandse vertaling Uitgeverij Lannoo, Tiel

Vertaling Ronald Schrijber, www.groenlingua.nl

Redactie Nederlandse editie Jaap Verschoor / Kantoor Verschoor Boekmakers, Heemstede

Zetwerk Peter Verwey Grafische Producties, Heemstede

ISBN 978 94 014 1965 9

D/2014/45/359

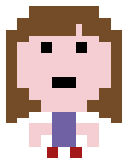
NUR 211 / 241

www.lannoo.com

Gedrukt en gebonden in China bij South China Printing Company

Alle rechten voorbehouden. Niets uit deze uitgave mag worden verveelvoudigd,
opgeslagen in een geautomatiseerd gegevensbestand en/of openbaar gemaakt in enige vorm
of op enige wijze, hetzij elektronisch, mechanisch of op enige andere manier
zonder voorafgaande schriftelijke toestemming van de uitgever.

Registreer u op onze website en we sturen u regelmatig een nieuwsbrief
met informatie over nieuwe boeken en met interessante, exclusieve aanbiedingen.



CAROL VORDERMAN is een van de populairste Britse tv-presentatrices en een bekend wiskundige. Ze heeft technische wetenschappen gestudeerd in Cambridge. Carol pleit ervoor dat elk kind de kans moet krijgen de waardevolle wereld van het coderen te leren kennen. Ze heeft talloze tv-shows gepresenteerd over wetenschap en techniek op de BBC, ITV en Channel 4. Ze was 26 jaar lang medepresentatrice van *Countdown* op Channel 4 en werd daardoor bijna de bestverkopende vrouwelijke non-fictieauteur vanaf 2000. En ook uit haar adviserende rol aan de Britse premier David Cameron over de toekomstige potentie van wiskundeonderwijs bleek overduidelijk haar passie en toewijding om op een spannende en voor iedereen te begrijpen manier wiskunde, wetenschap en techniek over te brengen op het grote publiek.



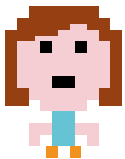
DR. JON WOODCOCK heeft natuurwetenschappen gestudeerd aan de University of Oxford en is doctor *Computational Astrophysics* aan de University of London. Hij begon met coderen toen hij acht was heeft alle soorten computers geprogrammeerd, van enkelchips microcomputers tot wereldwijde supercomputers. Tot zijn vele projecten behoren gigantische ruimtesimulators, onderzoek voor hightechbedrijven en intelligente robots die gemaakt zijn van afvalmaterialen. Jon heeft een passie voor wetenschappelijke en technologische educatie en geeft op scholen lezingen over ruimtecomputertechnologie en computerprogrammeren. Hij heeft aan talloze boeken over wetenschap en technologie meegewerkt.



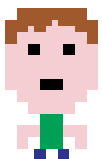
SEAN McMANUS leerde programmeren toen hij negen was. Zijn eerste programmeertaal was Logo, nu is hij een ervaren schrijver en journalist op technologisch gebied. Tot zijn boeken behoren *Scratch Programming in Easy Steps*, *Web Design in Easy Steps* en *Raspberry Pi For Dummies*. Op zijn website www.sean.co.uk vind je Scratch-games en handleidingen.



CRAIG STEELE is gespecialiseerd in computerwetenschappelijk onderwijs. Hij is projectmanager bij CoderDojo Scotland, dat gratis toegankelijke ruimtes beheert waar jongeren kunnen coderen. Hij heeft gewerkt bij de Scottish Qualification Authority, Glasgow Science Centre en de University of Glasgow. Craigs eerste computer was een ZX Spectrum.



CLAIRE QUIGLEY studeerde computerwetenschappen in Glasgow, waarin ze bachelor en doctor werd. Ze heeft gewerkt op de computerafdeling van de Cambridge University en aan een project over het ontwikkelen van computervaardigheden bij basisschoolkinderen. Ze is mentor bij CoderDojo Scotland, een codeerclub voor jongeren.



DANIEL McCAFFERTY heeft computerwetenschappen gestudeerd aan de University of Strathclyde. Daarna heeft hij software ontwikkeld voor de grootste investeringsbanken ter wereld. In zijn vrije tijd begeleidt hij kinderen bij CoderDojo in Schotland, een codeerclub voor jongeren.

Inhoud

8 **VOORWOORD** door Carol Vorderman

10 **ZO WERKT DIT BOEK**

1 **WAT IS CODEREN?**

14 Wat is een computerprogramma?

16 Denk als een computer

18 Word een codeerder

2 **SCRATCH**

22 Wat is Scratch?

24 Scratch installeren en opstarten

26 Scratch-interface

28 Sprites

30 Kleurblokken en scripts

32 **Project 1: Ontsnap aan de draak!**

38 Dingen laten bewegen

40 Uiterlijken

42 Verstoppertje

44 Gebeurtenissen

46 Eenvoudige loops

48 Pennen en turtles

50 Variabelen

52 Rekenen

54 Strings en lijsten

56 Coördinaten

58 Maak wat lawaai!

60 **Project 2: Gooi de dobbelsteen**

62 Goed of fout?

64 Beslissingen, vertakkingen

66 Waarnemen, detecteren

68 Ingewikkelde loops

70 Berichten versturen

72 Blokken maken

74 **Project 3: Apenkooien**

82 Tijd om te experimenteren

3 **PYTHON**

86 Wat is Python?

88 Python installeren

92 Kennismaking met IDLE

94 Fouten

96 **Project 4: Spokenspel**

98 Spokenspel gedecodeerd

100 Programmastroom

102 Eenvoudige commando's

104 Moeilijkere commando's

106 Welk venster?

108 Variabelen in Python

110 Soorten data

112 Rekenen in Python

114 Strings in Python

116 Input en output

118 Beslissingen nemen

120 Vertakken

- 122 Loops in Python
- 124 'While'-of 'terwijl'-loops
- 126 Ontsnappingsloops
- 128 Lijsten
- 130 Functies
- 132 **Project 5: Gekke zinnen**
- 134 Tupels en woordenboeken
- 136 Lijsten in variabelen
- 138 Variabelen en functies
- 140 **Project 6: Tekenmachine**
- 148 Fouten en foutopsporing
- 150 Algoritmes
- 152 Bibliotheken
- 154 Vensters maken
- 156 Kleur en coördinaten
- 158 Vormen maken
- 160 Dingen veranderen
- 162 Reageren op events
- 164 **Project 7: Bellenschiet**
- 176 Wat nu?

4 COMPUTERS VANBINNEN

- 180 Het hart van de computer
- 182 Binair en basisgetallen
- 184 Symbolen en coderingen
- 186 Logische poorten

- 188 Processors en geheugen
- 190 Basisprogramma's
- 192 Data opslaan in bestanden
- 194 Internet

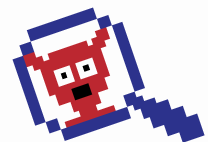
5 PROGRAMMEREN IN HET ECHT

- 198 Computertalen
- 200 Computerhelden
- 202 Nijvere programma's
- 204 Computergames
- 206 Apps ontwerpen
- 208 Programmeren voor internet
- 210 JavaScript gebruiken
- 212 Schadelijke programma's
- 214 Minicomputers
- 216 Word een meesterprogrammeur

- 218 Woordenlijst
- 220 Register
- 224 Dankwoord

Lees meer op:

www.dk.com/computercoding
(Engelstalig)



Voorwoord

Nog maar een paar jaar geleden leek programmeren iets geheimzinnigs dat alleen maar door specialisten kon worden gedaan. Veel mensen konden zich niet voorstellen dat programmeren leuk kon zijn. Maar ineens werd alles anders. Internet, e-mail, sociale netwerken, smartphones en apps overvielen ons als een wervelwind en veranderden ons leven compleet.

Computers zijn niet meer weg te denken uit ons leven. We bellen nauwelijks meer iemand op, maar sturen een sms of gebruiken de sociale media. Shoppen of vermaak, nieuws of games, alles doen we tegenwoordig via de computer. Maar we kunnen nog veel meer dan alleen gebruik maken van de computertechniek, we kunnen die ook zelf maken. Als we leren coderen, kunnen we zelf onze digitale meesterwerken maken.

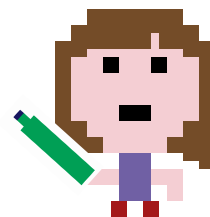
Alles wat computers doen wordt bepaald door regels met codes die iemand op een toetsenbord heeft ingevoerd. Het lijkt dan wel een vreemde taal, maar het is een taal die iedereen snel kan leren. Volgens velen is coderen een van de belangrijkste dingen die je in de 21ste eeuw kunt aanleren.

Leren coderen is ontzettend leuk, want je ziet meteen resultaat, ook al heb je nog veel te leren. Het is zelfs zo leuk om games en programma's te maken dat je niet eens merkt dat je er moeite voor moet doen als je er eenmaal mee bezig bent. En het is creatief – misschien wel de eerste wetenschap die kunst, logica, verhalen en business combineert.

Bovendien is coderen geweldig voor je ontwikkeling. Je leert logisch te denken en je kunt sneller problemen oplossen – heel belangrijk in veel verschillende gebieden in je leven, van wetenschap en techniek tot medicijnen en rechten, maar ook in creatieve beroepen. Het aantal banen waarvoor kennis van coderen nodig is, zal in de toekomst enorm toenemen; er is nu al een groot tekort aan goede codeerders. Leer coderen en de digitale wereld ligt aan je voeten!

Carol Vorderman

CAROL VORDERMAN



Zo werkt dit boek

In dit boek leer je alles wat nodig is om te kunnen coderen. Met allerlei leuke projecten breng je dit in de praktijk. We hebben alles in kleine stukken opgedeeld, zodat je het gemakkelijk kunt volgen en begrijpen.

Elk onderwerp wordt in detail beschreven met voorbeelden en oefeningen

In 'Zie ook'-kaders vind je andere onderwerpen over het thema

Pixelpoppetjes geven je onderweg aanwijzingen en tips



BELLENSCHIETER

De afstand tussen twee punten
In dit spel en in vele andere is het handig objecten te kennen. Hier leer je een beetje gebruiken die de computer kan berekenen.

11 De functie berekent de afstand tussen twee punten. Voeg dit coderingsfragment toe direct na de code die je hebt gemaakt in stap 9.

```
from math import sqrt
def distance(id1, id2):
    x1, y1 = get_coords(id1)
    x2, y2 = get_coords(id2)
    return sqrt((x2 - x1)**2 + (y2 - y1)**2)
```

Verstoppertje

Welkom in de studio van de speciale effecten! Ontdek met de paarze 'Uiterlijken'-blokken hoe je sprites kunt laten verdwijnen en weer verschijnen, groeien en krimpen en vervagen en weer opdoemen.

Sprites laten verdwijnen

Met het blok 'verdwijn' kun je een sprite laten verdwijnen. De sprite is nog wel op het speelveld en kan nog rondlopen, maar je kunt hem niet zien totdat je hem met het blok 'verschijn' weer zichtbaar maakt.

Met het blok 'verdwijn' laat je sprites in games verdwijnen



Verdwijn en verschijn

Gebruik het blok 'verdwijn' als je een sprite wilt laten verdwijnen, met het blok 'verschijn' haal je hem weer tevoorschijn. Je vindt deze blokken in de menu's van het blokkenpalet.



Verdwijnende kat

Probeer dit script uit met de kat. Hij verdwijnt en verschijnt, maar blijft bewegen, ook al kun je hem niet zien.

als wordt aangeklikt

- herhaal
 - wacht 1 sec.
 - verdwijn
 - draai 90 graden
 - neem 100 stappen
 - wacht 1 sec.
 - verschijn

Met dit blok verberg je de kat

Door dit blok draait de kat met de klok mee

De kat beweegt nog, ook al is hij verborgen

Met dit blok maak je de kat weer zichtbaar

ZIE OOK

(38-39) Dingen laten bewegen
Berichten (70-71) versturen

Grootte en effecten

Met scripts kun je de grootte van een sprite wijzigen en speciale effecten toevoegen.

Geef positieve getallen in om de sprites groter te maken en negatieve om ze kleiner te maken

Hogere getallen zorgen voor grotere sprites, terwijl lagere ze juist kleiner maken. 100 is normale grootte

Verander de grootte van je sprite
Met deze twee blokken kun je een sprite groter of kleiner maken door het getal te wijzigen of door een ander groottepercentage te kiezen.

Zet alle effecten uit

Grafische effecten
Met de grafische effecten kun je het uiterlijk van je sprite wijzigen of vervagen.

Laat je sprites opduiken

Als je een geest-sprite kiest uit de categorie 'Fantasie' in de bibliotheek kun je het script maken zoals hieronder weergegeven. Je kunt de geest laten verdwijnen en weer opduiken als je erop klikt.

als op deze sprite wordt geklikt

- zet alle effecten uit
- herhaal 20 keer
 - verander geest -effect met 5
- schuif in 0.1 sec. naar x: een getal tussen -150 en 150 y: een getal tussen -150 en 150
- herhaal 20 keer
 - verander geest -effect met 5

Door het 'geest'-effect zal de sprite enigszins vervagen. Als je dit blok 20 keer herhaalt, zal de sprite helemaal vervagen

Met het blok 'Functie' kun je een willekeurige horizontale houding geven

Met dit blok verschijnt de geest weer langzaam in beeld

Kleurrijke illustraties wijzen je op verschillende programmeer-ideeën

Programmeer-scripts en codes worden stap voor stap uitgelegd

Instructies om te klikken, slepen of selecteren

Labels helpen je elke stap te begrijpen

Via zeven projecten bouw je je codevaardigheden op. Projectpagina's vallen op door de blauwe balk

Via eenvoudige instructies word je stap voor stap door elk project geleid

BELLENSCHIETER 171

13

Update nu de hoofdloop van het spel om de functies te installeren die je zojuist hebt aangemaakt. Onthoud dat de volgorde belangrijk is, dus let erop dat je alles op de juiste plek zet. Voer de code dan uit. De bellen moeten knappen als ze de duikboot raken. Kijk in het Shell-venster wat je score is.

```

score = 0
#MAIN GAME LOOP
while True:
    if randint(1, BUB_CHANCE) == 1:
        create_bubble()
    move_bubbles()
    clean_up_bubs()
    score += collision()
    print(score)
    window.update()
    sleep(0.01)
    
```

Zet de score aan het begin van het spel op nul

Maakt nieuwe bellen aan

Telt de score op bij het totaal


Toont de score in het Shell-venster - het zal later correct worden weergegeven

Dit pauzeert de actie korte tijd - probeer dit te verwijderen en kijk wat er gebeurt

TIPS VAN EXPERTS

Python-snelkoppeling

De code 'score += collision()' is een snelkoppeling om 'score = score + collision()' te schrijven. Het voegt de score door botsing toe aan de totale score en updatet die daarna. Dit soort codering komt veel voor, dus is een snelkoppeling handig. Je kunt hetzelfde ook doen met het symbool '-'. Zo is 'score -= 10' bijvoorbeeld hetzelfde als 'score = score - 10'.



Vergeet niet je werk op te slaan

Elke coderingsregel heeft een duidelijk label, dus je kunt je niet vergissen

Dit icoontje geeft aan dat het project op de volgende bladzijde verder gaat

Free punten berekenen
 Het is handig om de afstand tussen twee punten te berekenen met een bekende wiskundige formule te berekenen.

tussen twee objecten.
 direct na de p.9.
 Laadt de functie 'sqrt' vanuit de Math-bibliotheek
 Krijgt de positie van het eerste object
 ds(id1)
 Krijgt de positie van het tweede object
 ds(id2)
 Geeft de afstand tussen beide

$$\sqrt{(x2 - x1)**2 + (y2 - y1)**2}$$

VERSTOPPERTJE 43

Wijzig de getallen in de blokken om te zien hoe krachtig het effect is

Verander pixeleren -effect met 25

Verander kleur op 0

Verander het nummer om de kleur te wijzigen

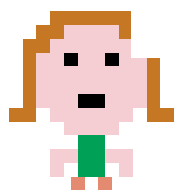
Met het blok 'Functies' kun je een willekeurige verticale houding kiezen

Met het blok 'Functies' kun je een willekeurige horizontale houding kiezen

Dit blok laat de geest langzaam bewegen, onzichtbaar

In kaders vind je extra informatie: tips, definities en dingen om te onthouden

Lees door en begin met coderen!



TIPS VAN EXPERTS

Wanneer opslaan?

Dit opslagsymbool verschijnt op de projectspreads om je eraan te herinneren dat je je werk moet opslaan. Zo gaat niets verloren als de computer crasht. Denk eraan regelmatig je werk op te slaan.



Vergeet niet je werk op te slaan

Wat is een computerprogramma?

ZIE OOK

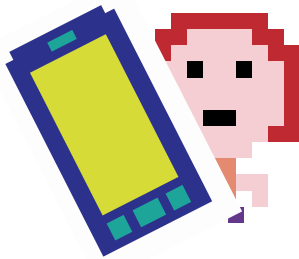
Denk als een **16-17** >
computer

Word een **18-19** >
codeerder

Een computerprogramma is een reeks instructies die een computer volgt om een taak uit te voeren. 'Coderen' of 'programmeren' wil zeggen dat je de stap-voor-stapinstructies schrijft die de computer vertellen wat hij moet doen.

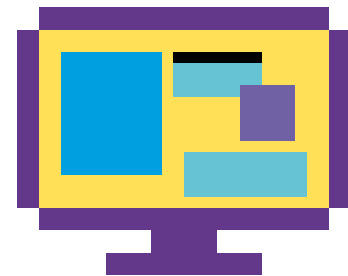
Computerprogramma's zijn overal

We worden omgeven door computerprogramma's. Veel apparaten die we elke dag gebruiken worden erdoor aangestuurd. Al deze machines volgen stap-voor-stap-instructies die een computerprogrammeur heeft geschreven.



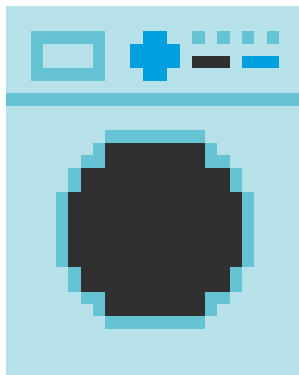
< Mobiele telefoons

Dankzij programma's kun je bellen en berichten versturen. Als je in je contactenlijst zoekt, vindt een programma het juiste nummer voor je.



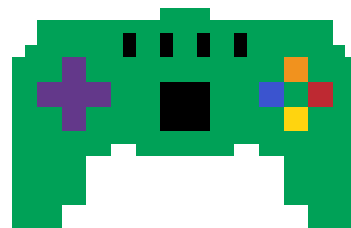
△ Computersoftware

Alles wat een computer doet, van browsen op het internet tot tekstverwerken of muziek afspelen, werkt door coderingen die een computerprogrammeur heeft ingevoerd.



△ Wasmachines

Wasmachines zijn geprogrammeerd om verschillende wasprogramma's uit te voeren. Via computercoderingen worden de temperatuur van het water en de duur van de wasbeurt geregeld.

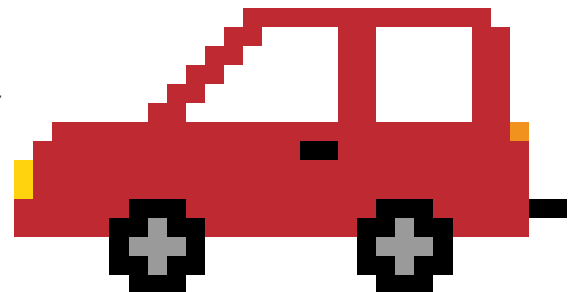


< Games

Consoles zijn gewoon ook een soort computers en alle games die je erop speelt, zijn programma's. Alle afbeeldingen, geluiden en besturingen zijn in computercodes geschreven.

> Auto's

In veel auto's controleren computerprogramma's de snelheid, temperatuur en het benzinepeil in je tank. Ze kunnen voor de veiligheid van de bestuurder zelfs checken of de remmen nog goed werken.

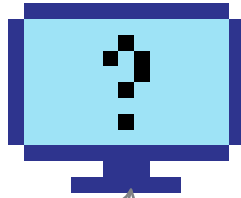


Zo werkt een computerprogramma

Computers lijken dan wel heel slim, maar het zijn in feite gewoon dozen die heel snel en heel precies instructies opvolgen. Omdat mensen intelligent zijn, gebruiken ze computers om bepaalde taken te verrichten aan de hand van geschreven programma's, lijsten of instructies.

1 Computers kunnen niet denken

Een computer zal niets uit zichzelf doen. Het is aan de programmeur om de computer te instrueren.



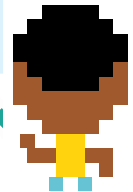
Zonder instructies is een computer stuurloos

Dit computerprogramma telt af om van start te gaan

2 Schrijf een programma

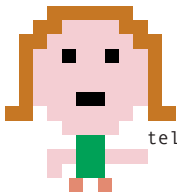
Je kunt een computer zeggen wat hij moet doen door hem een gedetailleerde reeks instructies te geven, een programma. Elke instructie moet zo beperkt zijn dat de computer hem kan begrijpen. Kloppen de instructies niet, dan zal de computer niet doen wat jij wilt.

```
tellen binnen bereik (10, 0, -1):
print ('aftellen', tellen)
```



3 Programmeertalen

Computers kunnen alleen instructies opvolgen die in een taal geschreven zijn die ze begrijpen. De programmeur bepaalt welke taal voor een bepaalde taak het geschiktst is.



```
tellen binnen bereik (10, 0, -1):
print ('aftellen', tellen)
```

Alle programma's worden uiteindelijk geconverteerd in een 'binaire code', een standaardcomputertaal die uit nullen en enen bestaat

```
0010 0011 1000 1100
1000 0110 0100 1001
0100 1001 0001 0101
```

AAN DE SLAG!



LINGO

Hardware en software

'Hardware' wil zeggen de materiële delen van de computer die je kunt zien of aanraken (alle bedrading, de stroomkring, het circuit, het toetsenbord, het scherm enz.). De 'software' zijn de programma's die in de computer draaien en regelen hoe de computer werkt. Software en hardware zorgen er samen voor dat de computer nuttige dingen kan doen.

Denk als een computer

Een programmeur moet leren denken als een computer. Alle taken moeten in kleine stukjes worden opgedeeld, zodat ze makkelijk te volgen zijn en onmogelijk verkeerd te begrijpen.

Denk als een robot

Stel je voor dat de ober in het café een robot is. De robot heeft een simpel computerbrein en je moet hem vertellen hoe hij de bestelling vanuit de keuken naar de gasten aan tafel kan brengen. Als eerste moet je nu het proces in eenvoudige taken verdelen die de computer kan begrijpen.

1 Ober-robot programma 1

Met dit programma grijpt de robot het eten van het bord, ramt dwars door de keukenmuur naar de eetzaal en zet het voedsel op de grond. Het algoritme was niet gedetailleerd genoeg.

1. Pak het eten op

2. Loop van de keuken naar de eettafel

3. Zet het eten neer

2 Ober-robot programma 2

Deze keer hebben we de ober-robot verteld dat hij de deur moet gebruiken. Dat doet hij nu ook, maar dan struikelt hij over de kat en valt het bord op de grond.

1. Pak een bord met eten op

2. Loop van de keuken naar de eettafel via:

Loop naar de deur tussen keuken en eetzaal

Loop van de deur naar de tafel

3. Zet het bord vóór de gast op tafel

ZIE OOK

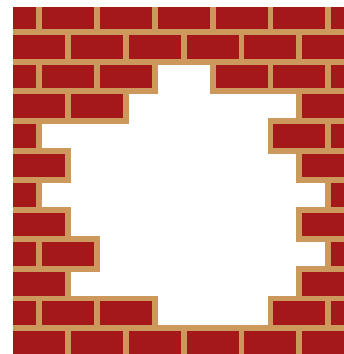
◀ 14-15 Wat is een computerprogramma?

Word 18-19 ▶
een codeerder

LINGO

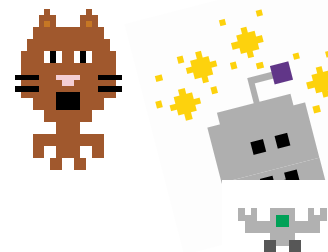
Algoritme

Een algoritme is een reeks eenvoudige instructies om een taak uit te voeren. Een programma is een algoritme dat vertaald is in een taal die computers kunnen begrijpen.



◁ Een ramp!

De instructies waren niet duidelijk: we zijn vergeten de robot te vertellen dat hij door de deur moet lopen. Voor mensen mag dat duidelijk zijn, maar computers kunnen niet zelf denken.



△ Nog niet perfect

De robot weet niet hoe hij met obstakels als een kat moet omgaan. Dus moet het programma nog preciezere instructies geven, zodat hij veilig kan lopen.

3 Ober-robot programma 3

Met deze versie van het programma brengt de robot het eten succesvol bij de gast en omzeilt alle obstakels. Maar nadat hij het bord heeft neergezet, blijft hij staan terwijl andere borden zich ophopen in de keuken.

1. Pak een bord met eten op en houd dit de hele tijd omhoog

2. Loop van de keuken naar de eettafel via:

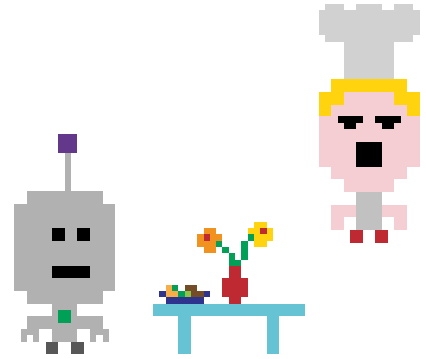
Loop naar de deur tussen keuken en eetzaal

Kijk of er obstakels zijn en loop eromheen

Loop van de deur naar de tafel

Kijk of er obstakels zijn en loop eromheen

3. Zet het bord vóór de gast op tafel



△ Eindelijk goed zo?

Eindelijk lukt het de robot het eten succesvol af te leveren. Maar we zijn vergeten hem te instrueren terug te gaan naar de keuken om het volgende bord te pakken.

Voorbeeld uit de echte wereld

De ober-robot mag dan fantasie zijn, maar dit soort algoritmes doen hun werk overal om ons heen. Een computergestuurde lift bijvoorbeeld heeft met precies dezelfde problemen te maken. Moet hij omhoog of omlaag? Naar welke verdieping moet hij hierna?

1. Wacht tot de deuren dicht zijn

2. Wacht tot er op een knop is gedrukt

Als de gekozen verdieping hoger ligt dan de huidige:

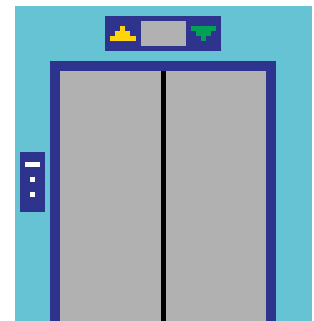
Beweeg de lift omhoog

Als de gekozen verdieping lager ligt dan de huidige:

Beweeg de lift omlaag

3. Wacht tot verdieping gelijk is aan gekozen bestemming

4. Open deuren



◁ Liftprogramma

Om de lift correct en veilig te laten werken moet elke stap precies en duidelijk zijn en elke andere mogelijkheid uitsluiten. De programmeurs moeten zeker weten dat ze een geschikt algoritme maken.