

# Les 12 - Toekomst van programmeren

[« Vorige les](#)

## Huiswerk

### Programmeren voor Beginners - Les 12

Je hebt je huiswerk nog niet ingediend.

[Huiswerk inzenden](#)

Je hebt 1 vraagmoment. 1 moment over.

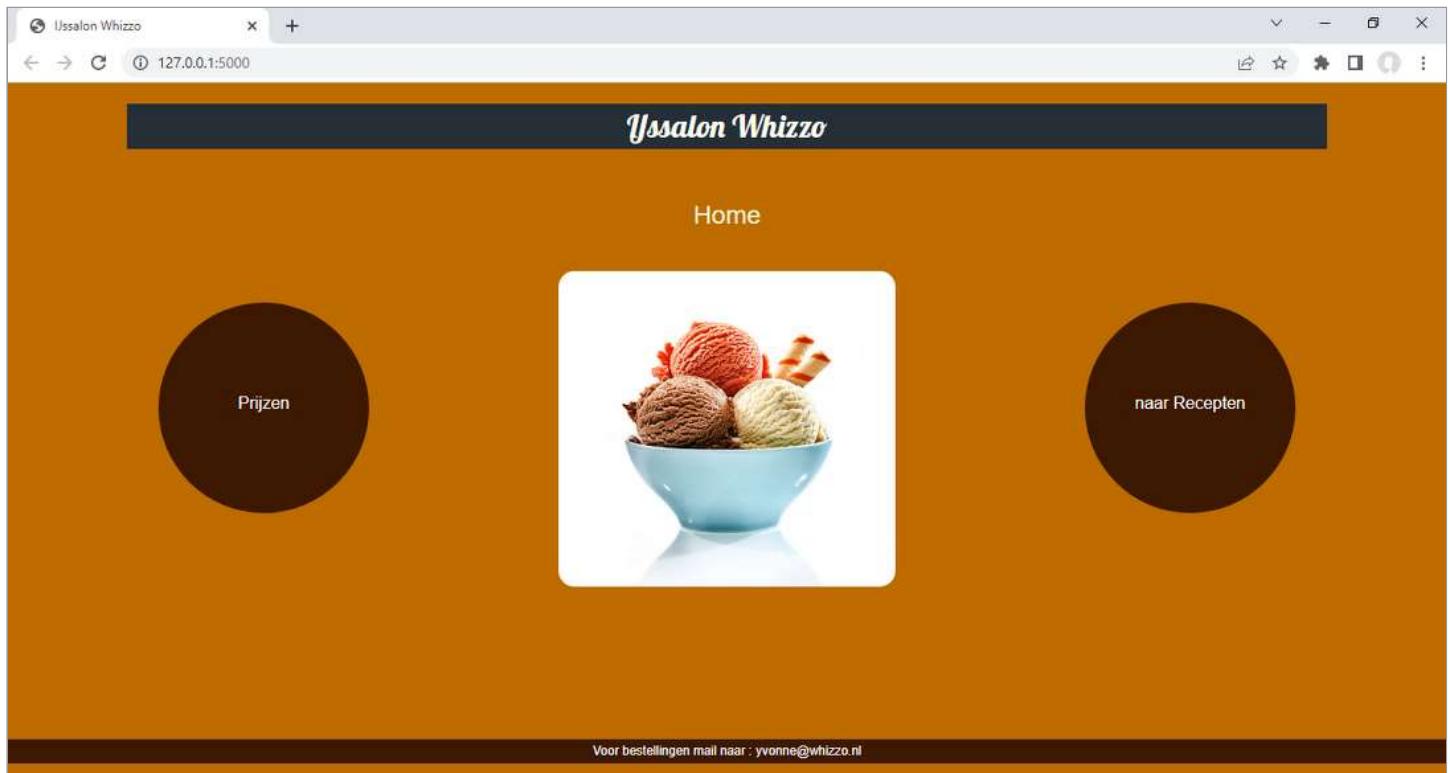
[Ga naar stel een vraag](#)[!\[\]\(cf531ed27e91483460120fcc057b3901\_img.jpg\) Markeer als deelgenomen aan deze les](#)

## Inhoudsopgave

- 12.1** [Inleiding](#)
- 12.2** [Ontwikkeling 1: Editors](#)
- 12.3** [Ontwikkeling 2: Internet of Things](#)
- 12.4** [Ontwikkeling 3: Cloud-based werken](#)
- 12.5** [Ontwikkeling 4: Veranderingen op taalgebied](#)
- 12.6** [Vervolgtaal kiezen](#)
- 12.7** [Samenvatting](#)
- 12.8** [Vraagmoment](#)
- 12.9** [Oefenopgaven](#)
- 12.10** [Huiswerkopgaven](#)
- 12.11** [Afsluiting](#)
- 12.12** [Bijlage](#)

## 12.1 Inleiding

Dit is de laatste les van deze serie. Gefeliciteerd! U heeft in de voorgaande lessen een gedegen basis gelegd voor diverse programmeerprojecten. Een mooi voorbeeld daarvan was de website die we maakten voor de ijssalon van Ellie van der Steen:



De weg staat voor u open om nog meer van zulke projecten op te pakken. Om dat te kunnen (blijven) doen, moet u zich voortdurend ontwikkelen. De programmeerwereld doet dat namelijk ook.

Computers en het internet spelen een steeds grotere rol en ontwikkelen zich continu. In deze les houden we ons daarom bezig met de vraag wat dit betekent voor u als programmeur.

Daarnaast besteden we aandacht aan een eventuele vervolgstap na deze laatste les. Gaat u zich specialiseren en zo ja, in welke taal? Wij geven u informatie die u kan helpen met het nemen van deze beslissing.

#### Leerdoelen

Aan het einde van deze les weet u:

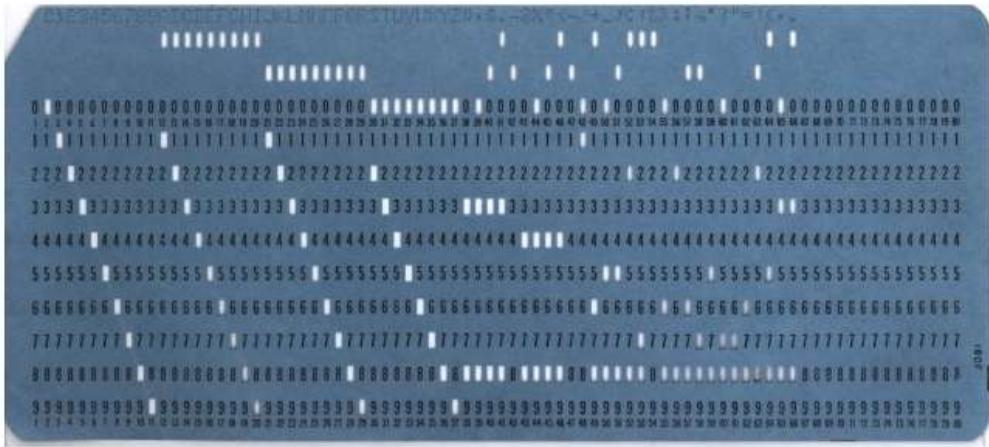
- hoe *code editors* zich hebben ontwikkeld en wat momenteel de trend is;
- wat *live parsing* is;
- wat we bedoelen met *code completion*;
- wat *Artificial Intelligence* is en welke rol het speelt in de programmeerwereld;
- hoe digitalisering heeft geleid tot *the Internet of Things*;
- wat *IoT*-programmeurs doen;
- wat de *cloud* is en wat *cloud computing* inhoudt;
- welke *cloud services* interessant kunnen zijn voor een beginner;
- welke veranderingen op taalgebied we zoal onderscheiden;
- hoe u het beste een vervolgentaalkunt selecteren.



Enkele afbeeldingen in deze les zijn moeilijk leesbaar in de printversie van deze les. Wilt u een afbeelding uitvergroten? Bekijk dan de digitale versie van de les op Plaza. Klik op de afbeeldingen om ze te vergroten.

## 12.2 Ontwikkeling 1: Editors

In de eerste les gaven we informatie over de geschiedenis van het programmeren. U maakte kennis met de eerste computerprogramma's, die de vorm van ponskaarten hadden.



De eerste computerprogramma's waren ponskaarten.

Het maken van zo'n ponskaart kon gemakkelijk fout gaan. De 'code' moest goed gecontroleerd werden voordat de ponskaart definitief gemaakt werd, want eenmaal gemaakt kon de kaart niet meer aangepast worden.

Tegenwoordig werken we gelukkig niet meer met ponskaarten, maar kunt u uw code invoeren in een **code editor**. Dit hulpmiddel is specifiek ontworpen voor het bewerken van de broncode van computerprogramma's door programmeurs. Bij een *editor* is het dus wél mogelijk om een code achteraf aan te passen.

De *editors* die wij vandaag de dag kennen, zijn veel meer ontwikkeld dan de eerste *editors*. In deze paragraaf gaan we kort in op de geschiedenis van de *code editor*.

## 12.2.1 LEXX

Rond 1985 ontwikkelde Mike Cowlishaw een redelijk moderne *editor*, genaamd LEXX. Die naam werd gekozen omdat de editor oorspronkelijk bedoeld was voor *lexicografen* die meewerkten aan de *New Oxford English Dictionary*.

Deze editor en gelijksoortige editors die halverwege de jaren tachtig van de vorige eeuw werden ontwikkeld, werden **source-code editors** genoemd. Men kon er de code mee schrijven en wijzigen, maar ook niet veel meer. Toch waren deze twee hulpmiddelen al verwerkt in dit type *editors*:

- **Live parsing**

Dit is het proces waarbij de *parser* (een component van de *compiler*) gegevens in real time analyseert en interpreteert. Dit gebeurt terwijl de gegevens worden gemaakt of ontvangen.

Bij *live parsing* breekt de *parser* de gegevens in kleinere stukjes, zoekt naar patronen en structuren en probeert betekenisvolle informatie te vinden. Dit is vaak belangrijk als er snel moet worden gereageerd op inkomende gegevens, bijvoorbeeld bij het werken met computers.

- **Color-coding**

Deze *editor* was al voorzien van *color-coding*. De kleuren gaven niet alleen informatie over de structuur van de code, maar informatie over de structuur van de code geven. Ook wordt door *color-coding* duidelijk waar eventueel fouten zijn gemaakt.

The screenshot shows a window titled 'Document: Bungler DED' with the text 'At: "<entry>"'. The XML code is as follows:

```
<entry>
  <hwsec>
    <hwgp>
      <hwlem>bungler</hwlem>
      <pron>b<I>n</I>nglər</pron>. </hwgp>
    <vfl>Also <vd>b</vd> <vf>bungler</vf>,
    </vfl>
  <etym>f. as prec. + <xra>(xlem)-er</xle</etym>
  <sen>One who bungles; a clumsy unskillful person.
  <quot>
    <qdat>1533 </qdat>
    <auth>MORE </auth>
    <wk>Assw. Payson. ZK. </wk>Wks. (1557)
    <qtxt>He is even but a very bungler.
  </quot>

```

Screenshot van editor LEXX.

## 12.2.2 IDE's

In de jaren negentig van de vorige eeuw werden zogenaamde IDE's populair. IDE staat voor *Integrated Development Environment*. Deze naam geeft aan dat dit soort applicaties tot meer in staat was dan alleen het aanmaken en wijzigen van broncode. Zo bevatten deze editors diverse hulpmiddelen die het leven van een programmeur vergemakkelijken, zoals mogelijkheden voor foutopsporing (*debugging*), een geavanceerde zoekfunctie en geïntegreerde versiecontrole.

Een IDE is vaak bedoeld voor het programmeren in een bepaalde taal. Zo is *Eclipse* een IDE voor Java. IDE's kunnen echter ook taalonafhankelijk zijn.



Op Plaza vindt u een video van de meest gebruikte code editors tussen 1991 en 2019.

In de video wordt duidelijk dat *Visual Studio* in 1997 het levenslicht zag. Later kwam er concurrentie van editors als *Sublime Text* en *PyCharm*. De editor die wij in deze cursus gebruiken (*Visual Studio Code*), is een afgeleide van *Visual Studio* en werd gelanceerd in 2015. Het werd al snel een van de meest populaire editors ter wereld, met name vanwege zijn veelzijdigheid, flexibiliteit en uitbreidingsmogelijkheden.

## 12.2.3 Opkomst van kunstmatige intelligentie

Veel IDE's, waaronder *Visual Studio Code*, werken met **automatische code-aanvulling** (*code completion*). Hierbij voorspelt de editor wat u wilde invoeren op basis van enkele tekens die u al heeft ingevoerd. U kunt uw invoer dan met één druk op de knop compleet maken. Ook mogelijk: als u een 'open-teken' invoert, wordt automatisch het 'sluit-teken' ingevoerd (hierbij kunt u bijvoorbeeld denken aan "" of {}).

Op dit moment vind er een nieuwe revolutie in de programmeerwereld plaats: kunstmatige intelligentie (ki) doet zijn intrede. Deze ontwikkeling is beter bekend onder de Engelse benaming: **Artificial Intelligence** (AI).

*Artificial Intelligence* is het vermogen van een computer of een computergestuurd apparaat om taken uit te voeren waarvoor normaal gesproken menselijke intelligentie nodig is.

Stelt u zich eens voor: een 'slim' algoritme dat de beschikking heeft over alle code die ooit is opgeslagen op *GitHub*, daar patronen uit kan halen en u op basis daarvan hints kan geven terwijl u aan het programmeren bent. Dat kan u flink wat tijd schelen!

Het is niet zo dat het algoritme al het werk voor u kan doen: u zult zelf nog de nodige beslissingen moeten nemen. Maar: de regels code die voor de hand liggend zijn, kunt u door het algoritme laten doen.

Een voorbeeld van zo'n hulpmiddel is **GitHub Copilot**. Meer informatie over dit hulpmiddel (inclusief een demo) vindt u op <https://github.com/features/copilot>. U zult zien dat *GitHub Copilot* erg ver gaat: het is zelfs mogelijk om in commentaarregels aan te geven wat u van plan bent om te programmeren en het programma komt met kant-en-klare suggesties.

In 2022 werd een alternatief voor *GitHub Copilot* gelanceerd, genaamd **Ghostwriter** (<https://replit.com/site/ghostwriter>). De makers van *Ghostwriter* claimen dat Ghostwriter dubbel zo snel is als *GitHub Copilot*.



Voor het gebruik van zowel **GitHub Copilot** als **Ghostwriter** moet u betalen. Er zijn gratis alternatieven, maar de vraag is of die net zo veel toegevoegde waarde hebben als hun commerciële tegenhangers.

## 12.3 Ontwikkeling 2: Internet of Things

Programmeren wordt steeds belangrijker en gelukkig ook populairder. Dat komt door **digitalisering**.

Digitalisering verandert onze samenleving in snel tempo. Digitale technologieën hebben grote invloed, onder andere op ons huishouden. Uw pc, tablet en smartphone zijn daardoor niet langer de enige 'computers' in huis.

In een aantal apparaten zit een **microcontroller**. Dit is een geïntegreerde schakeling met een **microprocessor**. (Een processor is het centrale onderdeel van de computer waar programmacode wordt verwerkt.) U kunt een microcontroller vergelijken met een kleine, eenvoudige computer, hij wordt namelijk gebruikt om elektronische apparatuur te besturen.

Er zijn veel apparaten die een microcontroller bevatten. Het gaat veel verder dan de eerdergenoemde pc, tablet en smartphone: ook apparaten zoals uw wasmachine, stofzuiger, koelkast, oven en thermostaat zijn voorzien van een microcontroller.

Het is relatief eenvoudig om een microcontroller te voorzien van een internetaansluiting. Daardoor zijn tegenwoordig niet alleen mensen maar ook dingen online. Alle aan het internet verbonden apparaten worden **Internet of Things (IoT)** genoemd.



Alle aan het internet verbonden apparaten worden 'Internet of Things' genoemd. En dat zijn er steeds meer!

Door deze internetverbinding worden onze apparaten steeds slimmer:

- Veel mensen hebben een slimme energiemeter, waardoor monteurs niet of nauwelijks nog langs hoeven te komen. De meterstanden worden automatisch uitgelezen en doorgegeven.
- Een geavanceerde tandenborstel analyseert uw poetsgedrag en geeft mogelijke verbeteringen.
- Een slimme thermostaat weet wanneer u thuis bent en wanneer niet, zodat de verwarming niet onnodig hoeft aan te staan.
- Er bestaan printers die zelf actie ondernemen als zij een nieuwe toner of servicebeurt nodig hebben.

Deze en andere vergaande koppelingen van (huishoudelijke) apparaten en het internet bieden oneindig veel mogelijkheden. Ze maken onze omgeving slimmer en meetbaarder, maar ook privacygevoeliger.

Doordat het aantal slimme apparaten blijft toenemen, stijgt ook de vraag naar **IoT-programmeurs**. Dit zijn programmeurs die gespecialiseerd zijn in het programmeren van zulke apparaten. Zij gebruiken daarvoor programmeertalen die we eerder in deze lessen noemden (zoals Python, Java of C++), maar ook hulpmiddelen die we nog niet eerder vermeldden (zoals Arduino of Raspberry Pi).



*Er is steeds meer vraag naar IoT-programmeurs, die apparaten zoals deze programmeren.*

## 12.4 Ontwikkeling 3: Cloud-based werken

De laatste jaren is **cloud computing** in opmars. Maar wat is dat eigenlijk?

Simpel gezegd is de *cloud* het internet. *Cloud computing* is daarom het werken en opslaan via het internet. Dat wat u opslaat, staat niet op de computer, maar online (op een **server**). Deze server staat altijd aan en is altijd verbonden met het internet.



*Een server is een computer in een datacentrum.*

De kans is groot dat u momenteel al aan *cloud computing* doet, bijvoorbeeld voor de opslag van foto's. U bewaart ze dan niet op de harde schijf van uw computer of op een usb-stick, maar gebruikt een service als *Dropbox*, *OneDrive*, *iCloud* of *Google Drive*.

Ook *GitHub* is een voorbeeld van *cloud computing*. Dergelijke services kunt u vaak gratis gebruiken, maar voor extra opslag moet u betalen.

Deze manier van werken heeft grote voordelen. Niet alleen zijn uw data goed beveiligd, maar ook kunt u uw data op iedere moment en op elk met het internet verbonden apparaat openen.

Tegenwoordig is het niet alleen mogelijk om *cloud computing* te gebruiken voor opslag. U gebruikt dan niet alleen de opslagruijtmte van een derde partij, maar bijvoorbeeld ook de processor en andere apparatuur en/of software.

Een voorbeeld daarvan is **Software as a Service** (SaaS), waarbij u software gebruikt zonder dat u die op uw eigen computer hoeft te installeren. Een bekend voorbeeld daarvan is Office 365 (van Microsoft).

Ook programmeurs doen steeds meer aan *cloud computing*. Daarvoor zijn professionele services beschikbaar. Enkele grote spelers op deze markt zijn Microsoft (Azure), Amazon (AWS) en Oracle (Gen 2).

Voor u als beginnend (Python-)programmeur zijn zulke services nog een stap te ver. Wel is het interessant om eens een kijkje te nemen op deze drie websites:

- **Replit** ([www.replit.com](https://www.replit.com))

Dit is een programmeergeoriënteerde cloud-IDE's die een ingebouwde compiler heeft die werkt vanuit elk besturingssysteem. Replit ondersteunt tientallen bibliotheken, waaronder Python, C, C++ en Java.

- **CodePen** (<https://codepen.io>)

Dit is een online community voor het testen en presenteren van door gebruikers gemaakte HTML-, CSS- en JavaScript-codefragmenten. Het functioneert als een online code-editor en een open-source leeromgeving, waar u codefragmenten kunt maken en testen.

- **PythonAnywhere** (<https://eu.pythonanywhere.com>)

Dit is een onlinedienst die het mogelijk maakt een Python-code uit te voeren in de *cloud*.



Ga naar Plaza en bekijk een video over PythonAnywhere.

Bij al deze aanbieders moet u een (in beginsel gratis) account aanmaken, waarna u hun *editor*, *interpreter* en *compiler* kunt gebruiken voor het vervaardigen, compileren en beschikbaar maken van uw zelfgemaakte programma's.

Ook goed om te weten: er is een *cloud*-versie van VS Code beschikbaar! Deze kunt u vinden op <https://vscode.dev>. U gebruikt dan niet de rekenkracht van uw eigen computer, maar die van de Microsoft-servers. Let wel: u heeft in deze onlineversie van VS Code niet alle mogelijkheden ter beschikking die u wel in de reguliere versie heeft.

A screenshot of the Microsoft Visual Studio Code interface running in a web browser. The title bar says "main.js – default (Workspace)". The left sidebar shows a file tree with a "DEFAULT (WORKSPACE)" folder containing ".github", ".vscode", "assets", "static" (with "images" and "javascript" subfolders), and "main.js". The "main.js" file is selected and shown in the center editor pane. The right editor pane shows "style.css" and "main.js" side-by-side. The bottom status bar shows "Ln 11, Col 1 Spaces: 4 UTF-8 LF (JavaScript Layout: U.S.)".

```
theCatSaidNo > static > stylesheets > # style.css > body
1 body {
2   font-family: 'Montserrat', 'Lato', '0
3   color: #6b7381;
4   background: #white;
5   -webkit-touch-callout: none;
6   -webkit-user-select: none;
7   -khtml-user-select: none;
8   -moz-user-select: none;
9   -ms-user-select: none;
10  user-select: none;
11  transition: background-color 0.25s;
12 }
13 .jumbotron {
14   background: #6b7381;
15   color: #ddc18;
16 }
17 .jumbotron h1 {
18   color: #fff;
19 }
20 .example {
21   margin: 4rem auto;
22 }
23 .example > .row {
24   margin-top: 2rem;
25   height: 5rem;
26   vertical-align: middle;
27   text-align: center;
28   border: 1px solid #rrba(189, 193, 20
29 }
30 .example > .row:first-of-type {
31   border: none;
32   height: auto;
33   text-align: left;
34 }
35 .example h3 {
36   font-weight: 400;
37 }
```

```
theCatSaidNo > static > javascript > JS main.js > ...
1 pawToggled = false;
2 var myTimeout;
3
4 function callbackToggle() {
5   return function () {
6     if (pawToggled) {
7       document.getElementById(
8         ...
9     }
10 }
11
12 function togglePaw() {
13   if (!pawToggled) {
14     // Runs when we toggle the b
15     document.getElementsByClassName(
16       ...
17     );
18     myTimeout = setTimeout(callbackTogg
19     ...
20   }
21
22 pawToggled = !pawToggled;
23 }
```

De onlineversie van VS Code lijkt veel op de 'reguliere' versie van het programma.

## 12.5 Ontwikkeling 4: Veranderingen op taalgebied

In de eerste lessen kreeg u al een overzicht van programmeertalen door de jaren heen. We begonnen bij de talen van Ada Lovelace en Conrad Zuse, bespraken de *Assembly Language* uit de jaren vijftig van de vorige eeuw, noemden talen als Fortran en Lisp en sloten af met talen die steeds meer leken op de reguliere Engelse taal.

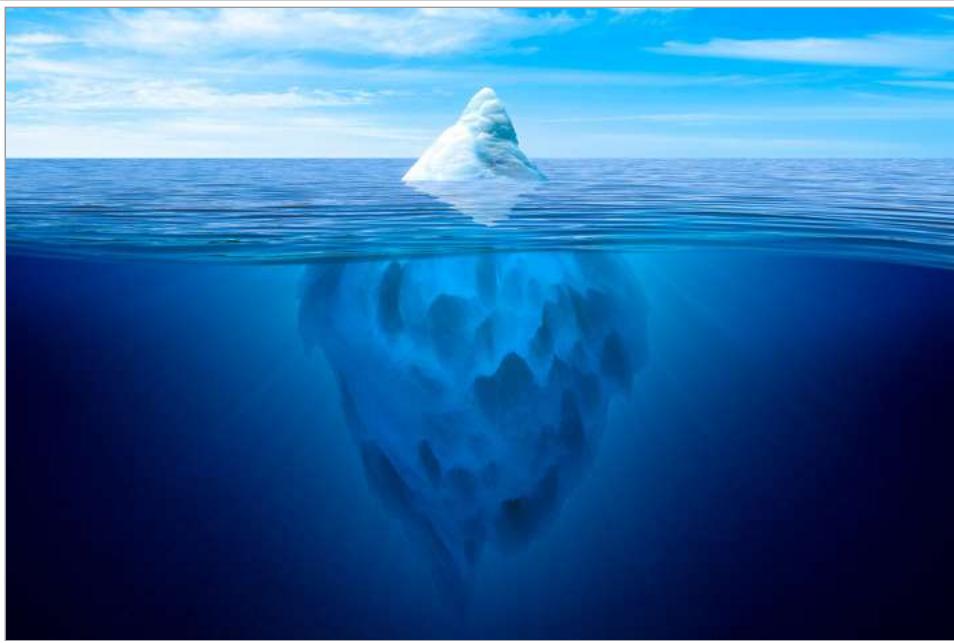
De ontwikkeling van programmeertalen is niet gestopt: ze ontwikkelen zich nog altijd. Op taalgebied kunnen we de volgende ontwikkelingen noemen:

- In de eenentwintigste eeuw verschenen onder meer C# (2000), Go (2009), Rust (2010), Dart (2011) en Swift (2014). Als u kijkt naar de diverse populariteitsindexen (waarover later in deze les meer), zult u zien dat deze talen inmiddels een plekje in de top 20 veroverd hebben.
- Veel nieuwe talen zijn **functionele programmeertalen**. De ‘functies’ die u heeft geleerd in Python, vloeien hieruit voort. Functies worden gebruikt om het computerprogramma in logische delen (in functies) op te splitsen en dan de functies aan te roepen in de hoofdcode.
- Het is niet zo dat zulke nieuwe talen de oude talen verdrijven. Sterker nog: diverse oude talen maken een comeback! Voorbeelden hiervan zijn Assembler (1947), Fortran (1957), COBOL (1959) en SQL (1972). De taal van Ada Lovelace wordt nog steeds gebruikt door de NAVO (zij het in een doorontwikkelde versie).

## 12.6 Vervolgtaal kiezen

In de voorgaande lessen heeft u op basisniveau programmeerkennis opgebouwd. Zo bent u onder meer bekend met begrippen als ‘variabele’, ‘statement’, ‘iteraties’ (*loops*) en ‘functies’.

Deze basis is echter slechts het topje van de ijsberg: er is nog veel meer te leren over programmeren.



*Er valt nog veel meer te leren over programmeren:  
uw kennis tot nu toe is slechts het ‘topje van de ijsberg’.*

Uw opgedane kennis is een goede basis voor verdere stappen in de programmeerwereld:

- U begrijpt ingewikkeldere stof pas als u de basis beheert.
- In welke taal u zich ook gaat verdiepen: u zult deze taal gemakkelijker onder de knie krijgen doordat u de basis beheert.

In deze cursus heeft u vooral gewerkt met de taal Python. Een verdieping in deze taal lijkt daarom een logische vervolgstap. Dat kan ook prima: met alle modules en extensies die momenteel al bestaan en alle nieuwe modules die vrijwel dagelijks uitkomen, is er weinig dat u niet met Python kunt doen.

Toch kan het ook interessant zijn u in een andere taal dan Python te verdiepen. Of u dit wilt, ligt vooral aan **uw programmeerdeelen**.

In een van de eerste lessen hebben we kort elf verschillende talen besproken. Daarbij horen grofweg de volgende doelen:

Taal	Toepassing
HTML	Webdevelopment
CSS	Webdevelopment
JavaScript	Webdevelopment
PHP	Webdevelopment
C++	Games / Mobiele apps / Computersoftware / Webapplicaties
C#	Games / Webapplicaties
Java	Mobiele apps
SQL	Databases
Ruby	Webdevelopment / Webapplicaties
Swift	Webapplicaties
Python	Data-analyse / Webapplicaties / Games / Machinelearning / ...

In de bijlage vindt u nogmaals de informatie per programmeertaal die u in een eerdere les kreeg. U zult zien dat u, na het doorlopen van deze lessenserie, deze informatie beter zult begrijpen.

Bij de keuze voor een ‘vervolgtaal’ is het ook interessant om te kijken naar de populariteit van talen. Daarbij biedt de **PYPL Index** hulp. PYPL staat voor *PopularitY of Programming Language*. Het is een lijst die is samengesteld op basis van zoekgedrag in zoekmachine Google.

Omdat de lijst geregelde wordt bijgewerkt, plaatsen we de lijst niet in deze les. In plaats daarvan vragen we u om de actuele lijst te bekijken op <https://pypl.github.io/PYPL.html>.



Andere interessante indexen zijn de **TIOBE Index**, **IEEE Spectrum's Top Programming Languages** en de resultaten van de **Annual Developer Survey** van Stack Overflow (kopje “Technology - Most loved, dreaded, and wanted”).

## 12.7 Samenvatting

In de allereerste les leerde u over de eerste computerprogramma's. Deze hadden de vorm van ponskaarten en konden niet meer worden aangepast nadat ze waren gemaakt. Er is sindsdien veel veranderd:

- Tegenwoordig gebruiken we *code editors* om een code te schrijven en wijzigen. De huidige *code editors* zijn veel geavanceerder dan de eerste *editors*, die in de jaren tachtig van de vorige eeuw werden ontwikkeld. Zo zijn onder meer *live parsing* en *color coding* als *features* toegevoegd.
- In de jaren negentig van de vorige eeuw werden IDE's populair. Deze applicaties boden meer dan alleen het aanmaken en wijzigen van broncode. Ze bevatten diverse hulpmiddelen voor programmeurs.

In de afgelopen decennia zijn verschillende IDE's op de markt gebracht. *Visual Studio Code*, dat in 2015 werd gelanceerd, is wereldwijd een van de populairste *editors*.

- Tegenwoordig heeft kunstmatige intelligentie (ki) haar intrede in de programmeerwereld gemaakt. Door middel van bijvoorbeeld *code completion* en automatische invoer van tekens kunt u daardoor veel tijd besparen.

Er zijn ook hulpmiddelen, zoals *Github Copilot*, die in staat zijn om op basis van commentaarregels te voorspellen wat u gaat programmeren en u op basis hiervan code aanbieden.

- *Cloud computing* is het opslaan en werken via het internet, in plaats van op de computer of een USB-stick. Dit kan voor opslag of voor het gebruik van software en andere apparatuur en/of software. Het heeft diverse voordelen, zoals goede beveiliging en toegang op elk met internet verbonden apparaat.

Er zijn professionele services beschikbaar, maar ook diverse services die interessant zijn voor een beginner. Ook is er een cloud-versie van VS Code beschikbaar.

## 12.8 Vraagmoment

Heeft u vragen over deze les? Dan kunt u die stellen via een vraagmoment.

U herkent een vraagmoment aan het spreekballonnetje dat op Plaza in het overzicht "Lessen" bij een les staat. Door op deze spreekballoon te klikken, komt u op een nieuwe pagina, waar u uw vraag in het tekstvak kunt typen.



Het tekstvak mag maximaal vijfhonderd tekens bevatten. Zorg er dus voor, dat u uw vraag/vragen bondig formuleert.

Nadat u op "Verstuur vraag" heeft geklikt, wordt uw vraag automatisch naar uw docent verstuurd. Zodra uw docent de vraag heeft beantwoord, verschijnt er op uw dashboard bij de opleiding een spreekballoon.

U kunt per les één keer vragen stellen. Verzamel dus uw vragen over de les, voordat u deze instuurt. Als u gebruik heeft gemaakt van het vraagmoment, verdwijnt het spreekballonnetje bij de les.

## 12.9 Oefenopgaven

De volgende opgaven werkt u als oefenopgaven voor uzelf uit. De uitwerkingen van deze opgaven kunt u meteen zelf controleren.

Hoewel *code editors* tientallen jaren geleden een stuk eenvoudiger waren, hadden ze toch al twee erg handige hulpmiddelen. Op welke twee hulpmiddelen doelen we?

→ Bekijk antwoord

Hoe verschilt een IDE van eerdere versies van *code editors*?

→ Bekijk antwoord

Wat is *code completion*?

→ Bekijk antwoord

Wat is *Artificial Intelligence*?

→ Bekijk antwoord

Hoe kan AI u als programmeur van dienst zijn?

→ Bekijk antwoord

Wanneer spreken we van een 'slim apparaat'?

→ Bekijk antwoord

Wat bedoelen we met *Internet of Things*?

→ Bekijk antwoord

Wat doet een IoT-programmeur?

→ Bekijk antwoord

Wat is *cloud computing*?

→ Bekijk antwoord

Ook voor een beginnend (Python-)programmeur is er SaaS-software beschikbaar. Wat kan een aanbieder van zulke software voor u betekenen?

→ Bekijk antwoord

Wat is de belangrijkste factor bij het bepalen van een vervolgtaal? En waarop kunt u nog meer letten?

→ Bekijk antwoord

## 12.10 Huiswerkopgaven

Maak deze huiswerkopgaven en stuur ze via Plaza naar uw docent. Deze opgaven worden nagekeken en voorzien van een cijfer. Mocht u een onvoldoende cijfer behalen, dan krijgt u een herkansing.

1. In een eerdere les stelden we u de volgende vraag:

*Heeft u zelf toekomstige programmeerplannen? Omschrijf die plannen dan kort en benoem welke taal volgens u het beste bij deze plannen past. Licht uw keuze kort toe.*

We vragen u nu nogmaals om deze vraag te beantwoorden.

(Heeft u nog steeds geen concrete programmeerplannen? Geef dan aan of het geleerde op een andere manier voor u nuttig is en zo ja, hoe.)

2. Maak een computerprogrammaatje dat twee getallen als invoer accepteert en True uitvoert als de getallen gelijk aan elkaar zijn en False als deze niet gelijk aan elkaar zijn.

Maak vervolgens een testtabel, waarbij u van tevoren nadenkt over de waardes die u wil invoeren en de te verwachten uitkomst. Bijvoorbeeld:

Invoer	Verwachte uitvoer	Uitvoer	Opmerkingen
1, 2	False		
3, 3	True		
...	...		
...	...		

Zorg ervoor dat uw tabel uit minimaal tien rijen bestaat en vul de tabel compleet in. (Lees: vul niet alleen een verwachting in, maar controleer ook of uw verwachting klopt.)

## 12.11 Afsluiting

U bent aan het einde gekomen van de lessenserie *Programmeren voor beginners*. Er zijn in deze lessen verschillende belangrijke basisbeginselen aan bod gekomen. Hopelijk bent u met deze cursus gekomen op een niveau waarop u gemakkelijk zelf verder kunt.

Het leren stopt natuurlijk niet met deze laatste les. Nu is het aan u om het geleerde in praktijk te brengen en steeds efficiënter met Python of een andere taal te leren omgaan.

Indien u verder wilt maar nog problemen ondervindt, kunt u de antwoorden op vragen vaak vinden op internet:

- Via een zoekmachine als Google vindt u veel informatie.
- Ook op websites, zoals *Stack Overflow*, *Python.org* en *W3Schools*, en/of *communities* staat veel waardevolle informatie. U kunt uiteraard ook nog eens terugbladeren in deze cursus.
- Daarnaast is *YouTube* een goede bron van informatie, ondanks dat u steeds meer advertenties te zien krijgt voordat u bij de inhoud komt en de inhoud van wisselende kwaliteit is.

Wilt u verder studeren bij NHA? Overweeg dan een cursus over een specifieke programmeertaal. NHA biedt uitgebreidere cursussen over onder meer Python, JavaScript, PHP en Java.

Wij wensen u veel succes met uw vervolgstappen in de programmeerwereld!

## 12.12 Bijlage

### Talen en hun toepassingen

In deze bijlage gaan we nogmaals dieper in op de programmeertalen die tegenwoordig veel gebruikt worden. U zult zien dat ze allemaal zo hun eigen toepassingen hebben.

### 12.12.1 HTML

Hoog of laag?	Hoog
Lineair of object-georiënteerd?	n.v.t.
Programmeer- of scripttaal?	n.v.t. (markuptaal)
Framework(s) op basis van HTML	n.v.t.

Om op het internet informatie te bekijken, gebruikt u een internetbrowser, ook wel 'webbrowser' of kortweg 'browser' genoemd. Door in de adresregel van het programma een webadres in te typen, roept u de startpagina van de website op. De webpagina is geschreven in een speciale opmaaktaal: **HTML**, wat staat voor *HyperText Markup Language*. De browser kan deze codering vertalen naar een grafische weergave op uw beeldscherm.

HTML ondersteunt **hypertekst**. Dat wil zeggen dat het mogelijk is om documenten en bestanden te verbinden door volgbare verwijzingen, die ook wel bekend zijn onder de naam **hyperlinks**.

Daarnaast is HTML een opmaaktaal. Door bepaalde codes kunt u aangeven waar u bijvoorbeeld koppen, tabellen en alinea's wilt plaatsen. Ook afbeeldingen (of andere vormen van multimedia) kunt u dankzij HTML opnemen in uw webpagina.

```

1 <html>
2
3   <head>
4     <title>DevOps Demo 01</title>
5     <link rel="stylesheet" href="style.css">
6   </head>
7   <body>
8     <h1>DevOps Demo 01</h1>
9     <p>Made by [name]</p>
10    </body>
11
12 </html>

```

Voorbeeld van HTML.

## 12.12.2 CSS

Hoog of laag?	Hoog
Lineair of object-georiënteerd?	n.v.t.
Programmeer- of scripttaal?	n.v.t.
Framework(s) op basis van CSS	Bootstrap, Foundation, Tailwind, ...

Een pagina die alleen in HTML-taal is geschreven, ziet er erg simpel en eenvoudig uit. Om een aantrekkelijke pagina te creëren, wordt daarom ook gebruikgemaakt van **CSS**.

CSS staat voor *Cascading Style Sheets*. Terwijl u HTML gebruikt om een webdocument te structureren, helpt CSS bij het specificeren van de stijl van uw document. Met CSS kunt u de vormgeving van elk element in een webpagina bepalen (zoals lay-out, kleur, uitlijning, vorm, lettertype of animatie).

```

1 body{
2   text-align: center;
3   font-family:'Trebuchet MS', sans-serif;
4 }
5
6 h1{
7   font-size: 200%;
8   margin-top: 20px;
9 }

```

Voorbeeld van CSS.

## 12.12.3 JavaScript

Hoog of laag?	Hoog
Lineair of object-georiënteerd?	Object-georiënteerd mogelijk
Programmeer- of scripttaal?	Scripttaal
Framework(s) op basis van JavaScript	Vue.js, React.js, jQuery)

Het gebruik van HTML en CSS is nog maar het begin. Wie een website 'slim' wil maken en als *content management-systeem* wil laten functioneren, moet andere programmeertalen toevoegen die deze acties vergemakkelijken. De meest voorkomende oplossingen voor deze taken zijn te vinden in programmeertalen als JavaScript, PHP, C#. In deze subparagraph lichten we de taal JavaScript toe. Informatie over PHP en C# volgt verderop.

Terwijl HTML vooral de structuur van een pagina beschrijft, is JavaScript bedoeld om een pagina interactiever te maken. Enkele populaire toepassingen zijn bijvoorbeeld animaties, formuliervalidatie, uitrolmenu's en fotocarrousels. Veel bekende websites maken gebruik van

JavaScript, zoals Office 365, Facebook, Instagram en Gmail. JavaScript is zo populair, dat het door alle browsers uitgevoerd kan worden.

JavaScript werd in eerste instantie geschreven om functionaliteiten toe te voegen aan websites, vandaar dat het vaak genoemd wordt in combinatie met HTML en CSS. De taal is echter zo populair geworden, dat er inmiddels ook manieren bedacht zijn om JavaScript te gebruiken voor andere toepassingen.

```
function getRecommendations(products, purchasedProducts, numRecommendations) {
    const weights = [];
    for (let i = 0; i < products.length; i++) {
        const product = products[i];
        const weight = getWeight(product);
        weights.push({id: product.id, weight});
    }
    weights.sort((recommendation1, recommendation2) =>
recommendation2.weight - recommendation1.weight);
    return weights.slice(0, Math.min(numRecommendations,
weights.length));
}
```

Voorbeeld van JavaScript.

#### 12.12.4 PHP

Hoog of laag?	Hoog
Lineair of object-georiënteerd?	Object-georiënteerd mogelijk
Programmeer- of scripttaal?	Scripttaal
Framework(s) op basis van PHP	Laravel, Symfony, CakePHP

Zonder te diep in te gaan op de werking van het internet, moet u begrijpen dat er bij een internetverbinding dingen gebeuren aan de kant van de server (de server van YouTube bijvoorbeeld) en aan de kant van de *client* (uw computer of mobiele telefoon). PHP is gemaakt om programmatuur te schrijven voor de kant van de server. De taal genereert HTML-pagina's en de toepassingen voor PHP zijn dus vooral te vinden op het gebied van webpagina's.

PHP is een concurrent van Python. Beide talen worden gebruikt voor de achterkant van een website (*back-end*). Een grappig detail is dat Python tegenwoordig vooral in de VS erg populair is en in Nederland op veel plaatsen PHP wordt gebruikt, terwijl Python is uitgevonden door een Nederlander (Guido van Rossum).

```

1 <?php
2 // single line comment 1
3 # single line comment 2
4 /* multi line comment - ?> */
5 /** doc-style comment - ?> */
6 function printNumber()
7 {
8     $number = 1234;
9     print "The number is $number,\n<?xml
10    for ($i = 0; $i <= $number; $i++) {
11        $x++;
12        $x--;
13        $x += 1.0;
14        $z = 2.; // error
15    }
16    if (PRINT_SUMMARY_CONSTANT)
17        echo <<< custom_HEREDOC
18 <br />
19 Summary:<br />
20 The \$x variable has value: $x
21 <br />
22 custom_HEREDOC;
23 }
24 ?>

```

Voorbeeld van PHP.

## 12.12.5 C++

Hoog of laag?	Laag
Lineair of object-georiënteerd?	Object-georiënteerd mogelijk
Programmeer- of scripttaal?	Programmeertaal
Framework(s) op basis van C++	Qt, Boost, Visual C++

Deze taal is een afgeleide van de taal C. C++ kan nog altijd beschouwd worden als een 'lagere' taal en wordt gebruikt voor het maken van games, mobiele apps, computersoftware en webapplicaties.

```

1 #include<iostream>
2 using namespace std;
3
4 int main() {
5     cout<<"Enter The Size Of Array:   ";
6     int size;
7     cin>>size;
8
9
10    int array[size], key,i;
11
12    // Taking Input In Array
13    for(int j=0;j<size;j++) {
14        cout<<"Enter "<<j<<" Element: ";
15        cin>>array[j];
16    }

```

Voorbeeld van C++.

## 12.12.6 C#

Hoog of laag?	Hoog
Lineair of object-georiënteerd?	Object-georiënteerd
Programmeer- of scripttaal?	Programmeertaal
Framework(s) op basis van C#	.NET, Visual C#, WPF

Ook deze programmeertaal is een afgeleide van C, maar deze variant is ontwikkeld tot een 'hogere' taal. Voor Microsoft is het de belangrijkste taal om zijn applicaties in te programmeren. Ook als u games wilt maken in de bekende game-engine *Unity*, is het leren van C# aan te bevelen.

```
void MyFunction () {  
  
    myLight.enabled = !myLight.enabled;  
}  
  
int AddTwo (int var1, int var2) {  
    int returnValue = var1 + var2;  
    return returnValue;  
}
```

Voorbeeld van C#.

## 12.12.7 Java

Hoog of laag?	Hoog
Lineair of object-georiënteerd?	Object-georiënteerd mogelijk
Programmeer- of scripttaal?	Programmeertaal
Framework(s) op basis van Java	Spring, Apache Wicket, Grails

Het voordeel van Java is dat de applicaties die u met deze taal maakt, op verschillende soorten devices uitgevoerd kunnen worden (dus op mobiele telefoons, tablets en pc's). Veel apps op mobiele telefoons zijn geschreven in Java.

```
package rentalStore;  
import java.util.Enumeration;  
import java.util.Vector;  
  
class Customer {  
    private String _name;  
    private Vector<Rental> _rentals = new Vector<Rental>();  
  
    public Customer(String name) {  
        _name = name;  
    }  
    public String getMovie(Movie movie) {  
        Rental rental = new Rental(new Movie("", Movie.NEW_RELEASE), 10);  
        Movie m = rental._movie;  
        return movie.getTitle();  
    }  
    public void addRental(Rental arg) {  
        _rentals.addElement(arg);  
    }  
    public String getName() {  
        return _name;  
    }  
}
```

Voorbeeld van Java.

## 12.12.8 SQL

Hoog of laag?	n.v.t.
Lineair of object-georiënteerd?	n.v.t.
Programmeer- of scripttaal?	n.v.t
Framework(s) op basis van SQL	-

Deze taal is een beetje 'vreemde eend in de bijt': hij wordt gebruikt om bepaalde typen databases aan te maken en te bewerken. (Een database is een gestructureerde verzameling data.)

```
SELECT TOP (1000) [id]
    ,[code]
    ,[firstName]
    ,[lastName]
    ,[locationID]
FROM [SampleDB].[dbo].[employees]
```

Voorbeeld van SQL.

## 12.12.9 Ruby

Hoog of laag?	Hoog
Lineair of object-georiënteerd?	Object-georiënteerd mogelijk
Programmeer- of scripttaal?	Scripttaal
Framework(s) op basis van Ruby	(Ruby on) Rails, Scorched, Cuba

Ook deze taal wordt toegepast bij het maken van websites/webapplicaties. Ruby kenmerkt zich door het feit dat een programmeur snel tot resultaat kan komen. De taal wordt veel gebruikt met het framework *Rails*.

Ruby is een **opensourcetaal**. Dat wil zeggen dat deze niet in beheer is van een bepaald bedrijf, maar in principe door iedereen kan worden uitgebreid en mede ontwikkeld.

```
1      invoke active_record
2      create  db/migrate/20151212160304_create_books.rb
3      create  app/models/book.rb
4      invoke resource_route
5          route resources :books
6      invoke scaffold_controller
7      create  app/controllers/books_controller.rb
8      invoke  erb
9      create  app/views/books
10     create   app/views/books/index.html.erb
11     ...
```

Voorbeeld van Ruby.

## 12.12.10 Swift

Hoog of laag?	Hoog
Lineair of object-georiënteerd?	Object-georiënteerd mogelijk
Programmeer- of scripttaal?	Programmeertaal
Framework(s) op basis van Swift	Quick, Vapor, SwiftMonkey

Dit is de taal waarmee de meeste applicaties voor *Apple-devices* worden geschreven (zoals de *iPhone* en *iMac*). Het is een relatief nieuwe taal met veel elementen uit Python en Ruby.

```

struct Player {
    var name: String
    var highScore: Int = 0
    var history: [Int] = []

    init(_ name: String) {
        self.name = name
    }
}

var player = Player("Tomas")

```

Voorbeeld van Swift.

## 12.12.11 Python

Hoog of laag?	Hoog
Lineair of object-georiënteerd?	Object-georiënteerd mogelijk
Programmeer- of scripttaal?	Scripttaal
Framework(s) op basis van Python	Django, Flask, Pyramid, Enthought Tool Suite

Python is een hogere programmeertaal die werd ontwikkeld door de Nederlander Guido van Rossum. Deze taal is zeer leesbaar en relatief eenvoudig te leren. Vandaar ook dat we in het vervolg van deze cursus met Python zullen werken.

Toepassingen van deze taal zijn *machine learning*, *datamining* en allerhande computerapplicaties. Ook gebruiken wetenschappers deze taal veel voor het maken van berekeningen.

Door gebruik te maken van frameworks als *Flask* en *Django* kan Python zelfs gebruikt worden als *back-end language* voor websites. Een bekend voorbeeld van een website die dat doet, is *YouTube*.

```

num = int(input("Enter the value of n: "))
hold = num
sum = 0

if num <= 0:
    print("Enter a whole positive number!")
else:
    while num > 0:
        sum = sum + num
        num = num - 1;
    # displaying output
    print("Sum of first", hold, "natural numbers is: ", sum)

```

Voorbeeld van Python.