Wat is Programmeren?

01 Inleiding

Inleiding

Wat is programmeren?

image-1681546166494.png

Programmeren is het het schrijven van instructies in een programmeertaal waarmee een computer taken kan uitvoeren.

Het is als het schrijven van een recept waarbij je de computer vertelt wat het moet doen in kleine stappen, zoals het uitvoeren van berekeningen, het weergeven van informatie op het scherm, het opslaan van gegevens of het uitvoeren van bepaalde acties.

Het si dus belangrijk dat we een opdracht in stapjes leren opdelen.

In deze module leren we:

1. Hoe delen we een taak op in stapjes?
2. Wat is een programmeertaal?
   We gebruiken hiervoor LUA (maar dat is eigenlijk niet belangrijk want alle programmeertalen lijken heel erg veel op elkaar).
3. Hoe schrijf je code en waar bestaat code uit?
4. We leren hoe een computer een programma uitvoert (spoiler stap voor stap).
5. We leren wat variabelen zijn en hoe we hier mee kunnen werken.
6. We leren hoe we de volgorde van instructies die de computer moet uitvoeren kunnen beïnvloeden.
7. We leren een aantal basis elementen van de computertaal en we gaan hiermee aan de slag.

Aan het eind van deze module kun je zelf een paar eenvoudige programmaatjes maken.

Opdracht

We beginnen met een oefening op in stapjes te leren denken.

Stel dat jij pauze hebt en bij de Albert Heijn een broodje gaat kopen. Welke stappen moet je dan uitvoeren. Noem er minimaal 8.

Denk aan betalen, broodje kiezen, naar de AH lopen, terug naar school lopen, de AH inlopen, betalen, naar de broodjes lopen, ...

Inleveren

Een PDF-document *opdracht1-<jouw-naam>.pdf* , waarin je alle stappen beschrijft in de juiste volgorde.

Voorbeeld: Stel je heet Merouane Alhidi dan heet jouw document dus " *opdracht1-MerouaneAlhidi.pd* f".
Zorg ervoor dat je jouw Word document ook bewaard als pdf.

--

02 Stappen

Stappen

Stap voor stap in de juiste volgorde

image-1681546405543.png

Bij programmeren moet je de computer stap voor stap vertellen wat er moet worden gedaan, omdat computers niet dezelfde manier van denken hebben als mensen. Computers begrijpen alleen heel specifieke en exacte instructies, en ze voeren deze instructies uit in de volgorde waarin ze zijn geprogrammeerd. Dit betekent dat de programmeur heel precies moet zijn en alle stappen die nodig zijn om een bepaalde taak uit te voeren, moet beschrijven.

Als de programmeur een taak wil automatiseren, moet hij of zij deze taak in kleinere stappen opdelen en voor elke stap specifieke instructies voor de computer schrijven. Deze instructies worden in de code geschreven in de programmeertaal van keuze, en elke regel code vertelt de computer wat er moet gebeuren in een bepaalde stap van het proces. De computer voert deze stappen vervolgens uit in de volgorde waarin ze in de code zijn geschreven, en produceert zo het gewenste resultaat.

Daarom is het essentieel om bij het programmeren de computer stap voor stap te vertellen wat er moet worden gedaan, om ervoor te zorgen dat de code correct wordt uitgevoerd en het gewenste resultaat wordt geproduceerd. Zelfs kleine fouten in de code of het ontbreken van een enkele stap kunnen leiden tot ernstige problemen en bugs in de software.

Oefenen

Stel je bent thuis en je wilt naar school, welke stappen voer je dan uit?

1. Check hoe laat ik op school moet zijn.
2. Bepaal hoe laat ik moet vertrekken van huis.
3. Begin 5 minuten voor vertrek mijn spullen in te pakken; laptop, lader, schoolpas, tas, telefoon en natuurlijk mijn OV kaart en jas.
4. Vertrek van huis met de spullen op de tijd die is bepaald in stap 2.
5. Pak de fiets uit de schuur.
6. Sluit het huis en de schuurdeur af.
7. Ga naar het busstation (bijvoorbeeld).
8. Ik kan 5 minuten voordat de bus vertrek aan op het busstation en zet mijn fiets op slot.
9. ....

En ga zo maar door. Snap je?

Opdracht A

Stel, je wilt gaan douchen, welke stappen moet je dan uitveren?

Neem de onderstaande stappen over en zet ze in de juiste volgorde.

1. Spoel de shampoo uit je haar met water.
2. Spoel je lichaam volledig af met water.
3. Maak je haar en lichaam volledig nat met water.
4. Zet de douche aan en zorg ervoor dat de temperatuur van het water naar wens is.
5. Stap in de douchecabine en sluit de deur of het gordijn.
6. Breng shampoo aan op je haar en masseer het in je hoofdhuid.
7. Droog jezelf af met een handdoek.

8. Breng douchegel of zeep aan op een washandje of spons en was je lichaam.
9. Schakel de douche uit en stap uit de douchecabine.
10. Spoel het washandje of de spons uit met water en herhaal stap 6 indien nodig.

Maak een document en zet de 10 stappen in de juiste volgorde.

Inleveren (1 bestand)

stappen02a-<jouw-naam>.pdf, met de 10 stappen, in de juiste uitvoeringsvolgorde uit **opdracht A** .

--

03 Eerste code

Eerste code

Computer code

image-1681546689394.png

Voor deze opdracht gaan we code maken die we via een website kunnen testen. De taal die we gebruiken is LUA, maar dat is niet zo belangrijk omdat de meeste computertalen heel erg veel op elkaar lijken.

Ga naar: https://www.lua.org/cgi-bin/demo

Typ het volgende programma van twee regels op de website in en druk op run:

print ("Hallo") print ("Ik ben de computer en ik spreek LUA")

Druk op run en je ziet het resultaat:

image-1681475595555.png

Het programma wordt regel voor regel uitgevoerd. Eerst regel 1 en dan regel 2.

Opdracht

Voeg zelf een regel toe aan deze code. Verzin zelf een tekst en zorg ervoor dat de tekst op de derde regel wordt afgedrukt.

Inleveren

Een schermafdruk van je gehele browser met de LUA-code en het resultaat. Noem dit document opdracht03-<jouw-naam>.

--

04 Flow control (goto)

Flow control (goto)

Flow contro l is een moeilijk woord voor de code die ervoor zorgt in welke volgorde je code wordt uitgevoerd.

Standaard worden de regels code één voor één uitgevoerd. Eerst de eerste regel, dan de tweede, enzovoorts.

Maar we kunnen met de code de volgorde ook aanpassen. Je kunt bijvoorbeeld aan het eind van het programma zeggen dat de code door moet gaan op regel 1.

Dat ziet er zo uit:

```
::begin::
print("Hallo")
print("Ik ben de computer en ik spreek LUA")
goto begin
```

Op regel 1 zetten we een label, we geven deze regel een naam. Dan voeren we de code op regel 2 en 3 uit. En als we klaar zijn dan staar er op regel 4 dat de code door moet gaan op de regel met de naam ::begin::

Opdracht

Wat doet deze code? Voer de code uit. Als je de code uitvoert dan staat er de tekst "Your program was aborted" (jou programma is afgebroken), waarom is dat?

Dus leg uit wat je code doet en waarom die wordt afgebroken.

Inleveren

Een PDF-document (opdracht04-<jouwnaam>.pdf) waarin je in eigen woorden uitlegt wat het programma doet en waarom het programma is afgebroken.

Je mag overleggen met elkaar, maar je moet je ***eigen*** PDF-document maken.

--

05 Variabelen

Variabelen

Een variabele in een programmeertaal wordt gebruikt om een waarde op te slaan die later kan worden gebruikt of gewijzigd. Het is een soort container (of doosje) waarin je gegevens kan opslaan, zoals een getal, een tekst, of een object.

image-1681476608665.png

Stel je wilt de computer laten onthouden dat de jouw naam Bob is en dat jij 35 jaar oud bent. Dan kun je twee variabelen maken, *name* en *age,* in de variabele *name* stop je dan de waarde Bob en in de variabele *age* stop je dan de waarde 35.

De naam (*name* en *age*) kan je in je code zelf kiezen.

```
name="Max Bisschop"
age=35
```

Als je deze code uitvoert dan gebeurt er niets. Dat komt omdat er geen output is. Je drukt niets af zoals in de vorige opdrachten.

Opdracht

Bedenk twee eigen variabele en geef die een waarde.

Druk vervolgens de twee variabelen af.

Inleveren

Screenshot van je *gehele* browser met de opdracht. Noem het bestand *opdracht5-<jouw-naam>.png*

Voorbeeld

image-1681477008155.png

06 Rekenen met variabelen

We kunnen ook rekenen met variabelen. Kijk naar de volgende code:

```
getal=1
print(getal)
getal=getal+2
print(getal)
```

Tip: als je in een programmeertaal een = ziet dan kun je dit beter lezen als 'wordt' en dus niet als 'is'.

Regel 1: Deze code maakteen variabele genaamd *getal* en geeft deze de waarde 1.
In het kort staat er dus: de variabele *getal* wordt 1

Regel 2: druk de variabel getal af.

Regel 3: Getal wordt getal plus getal. Getal was 1 dus getal wordt 1+2. Getal wordt dus 3.

Regel 4: druk de variabel getal af.

Regel 2 en 4 zijn hetzelfde, maar er wordt toch wat anders afgedruk. Dat komt dus omdat de waarde van de variabele op regel 3 wordt veranderd.

Opdracht

Gebruik de volgende code en vul aan op de plaats van de ....

Zorg ervoor dat op regel 7 het laatste cijfer van jouw studentennummer wordt afgedrukt.

Dat doe je dus door de code over te nemen en alleen op de plaats van de puntjes de code aan te passen.

```
getal=10
getal=getal-5
print(getal)
getal=getal*2
```

```
print(getal)
getal=getal....
print(getal)
```

Inleveren

Een screendump *opgave6-<jouw-naam>.png* van je gehele browser met oplossing.

--

07 if-then-else-end

Check deze code:

```
leeftijd = 18

if leeftijd >= 18 then
    print("Je bent volwassen!")
    print("Ben er maar trot op!")
else
    print("Je bent minderjarig.")
    print("Jammer dan.")
    print("Maar je wordt vanzelf volwassen ;)")
end
```

Deze code controleert de waarde van de variabele *leeftijd* en beslist op basis daarvan welke boodschap moet worden afgedrukt op het scherm.

De if-then-else begint met "if", gevolgd door de voorwaarde die wordt gecontroleerd.

De if-then-else-end heeft 3 delen.

1. De voorwaarde op de erste regel
2. De stappen die worden uitgevoerd als de voorwaarde waar is, dit staat tussen *then* en *else* .
3. De stappen die worden uitgevoerd als de voorwaarde niet waar is, dit staat tussen *else* en *end* .

Als de voorwaarde waar is, dan wordt de code in de "then"-blok uitgevoerd. In dit geval, als de leeftijd groter of gelijk is aan 18, dan wordt de boodschap "Je bent volwassen!" afgedrukt op het

scherm.

Als de voorwaarde onwaar is, wordt de code in de "else"-blok uitgevoerd. In dit voorbeeld, als de leeftijd kleiner is dan 18, dan wordt de boodschap "Je bent minderjarig." afgedrukt op het scherm.

Het "end"-trefwoord sluit de hele if-then-else af.

Opdracht

Maak een variabele *temperatuur* en geeft deze een waarde.

Maak een if-then-else-end. Als de temperatuur < 19 is dan druk je af dat de kachel aan moet. Als dit niet zo is dan druk je af dat het nog niet zo koud is dat de kachel aan moet.

Inleveren

Een screendump, *opdracht7-<jouw-naam>.png* van je gehele browser met oplossing.

--

08 Herhalen totdat

Kijk naar het volgende programmaatje:

```
a=1


::begin::
a=a+1
print(a)
goto begin
```

Deze code zet de variabele op 1 en telt er dan 1 bij op.

De variabele wordt afgedruk en dan gaar de code weer naar regel 3. Dit gaat eindeloos door. Stel we willen stoppen als a 15 is.

```
a=1
::begin::
a=a+1
```

```
print(a)
if (a == 15) then
  print("klaar")
else
  goto begin
end
```

Het wordt al wat ingewikkelder, maar laten dit programma stap voor stap bekijken.

1. de variabele a krijgt de waarde 1
2. Regel 2 krijgt de naam *begin*
3. de variabele a krijgt de waarde van a plus 1, dus de variabele a wordt 1 hoger.
4. de waarde van de variabele a wordt afgedrukt
5. De regel is een if-then-else en regel 5 tot en met 9 horen bij elkaar. Op deze regel staan als de waarde van a 15 doe dan wat er na de *then* staan (dus op regel 6).
6. Doe dit als de if op regel 5 waar is. Druk dan *klaar* af.
7. Na de else staat wat er moet gebeuren als de vergelijking op regel 5 NIET waar is.
8. Doe dit als de if op regel 5 niet waar is. Ga nu naar de regel met de naam begin (dat is regel 2).
9. De end geeft aan dat dit het einde is van het if-then-else-end.

Opdracht

Gebruik het voorbeeld, maar pas het aan:

- gebruik je eigen variabele naam in plaats van *a* . Bedenk een naam van tenminste 5 letters.
- druk alle getallen van 10 tot en met 20 af.

Inleveren

Een screendump, *opdracht8-<jouw-naam>.png* van je gehele browser met oplossing.

--

09 While

De vorige opdracht kan je mooier en eenvoudiger uitvoeren met een *while* .

De *while* ziet zo uit.

```
a=1
while( a < 15 )
do
   print(a)
   a=a+1
end
```

Alles wat tussen de do en end staat wordt uitgevoerd zolang (while) de vergelijking op regel 2 waar is.

Programmablok en inspringen

De code tussen regel 3 en 6 wordt een programmablok genoemd. Een programmablok wordt altijd ingesprongen. Dat is technisch niet in alle talen verplicht, maar dit maakt de code wel een stuk leesbaarder.

Variabelenamen

De variabelenaam *a* is slecht gekozen. a zegt niets en in een programma heb je wel 100den variabelen, als die allemaal a, b, c of zo heten dan weet je niet waar dit voor staat. Je zou bijvoorbeeld veel beter de naam *optellen* of *teller* of *getallen,* of *....* kunnen gebruiken. Kun jij een goede logische naam bedenken?

Je kunt een while-loop ook in een soort grafiekje weergeven, we noemen dat een flow chart (een diagram die de flow laat zien).

WhileloopinC2.png

Opdracht

De volgende (hieronder) code is slecht geschreven. Er zit een fout in en de code springt niet juist in.

Pas de code aan zodat de getallen 20 tot en met 1 in aflopende volgorde wordt afgedrukt. Dus de output moet worden 20,19,18,17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1 (maar dan wel onder elkaar.

Vervang de variabele a door een eigen gekozen variabele waarbij de naam van de variabele tenminste 5 lang is.

Dus je doet drie dingen:

1. Pas de code aan zodat de getallen 20 tot en met 1 wordt afgedrukt (tip: het plus teken op regel 6 is waarschijnlijk niet juist!).
2. Laat de code inspringen zoals het hoort.
3. Vervang de variabele-naam a in een zelfgekozen naam van tenminste 5 lang. Gebruik een logische naam.

```
a=20


while (a>1)
do
print(a)
a=a+1
end
```

Inleveren

Een screendump, opdracht9-<jouw-naam>.png van je gehele browser met oplossing.

--

10 Tafel

Screenshot 2023-04-19 144156.png

Bekijk de onderstaande code

```
tafel=8
teller=0


teller=teller+1
print(teller.."X"..tafel.."="..teller*tafel)

teller=teller+1
print(teller.."X"..tafel.."="..teller*tafel)

teller=teller+1
print(teller.."X"..tafel.."="..teller*tafel)

teller=teller+1
```

```
print(teller.."X"..tafel.."="..teller*tafel)
```

Voer deze code (hierboven) uit en bekijk de output.

```
1X8=8
2X8=16
3X8=24
4X8=32
```

Het doel is om de code uit te breiden zodat de gehele tafel van 8 wordt afgedrukt, dus:

```
1X8=8
2X8=16
3X8=24
4X8=32
5x8=40
6*8=48
7*8=56
8*8=64
9*8=72
10*8=80
```

We kunnen natuurlijk de regel 13 en 14 nog een paar keer kopiëren, Maar stel we zouden door willen gaan tot aan 100*8=8000, dat zou heel veel (stom) werk zijn.

Da moet dus anders kunnen en daarvoor kunnen we de while gebruiken die we in de vorige opdracht hebben geleerd.

Opdracht

Maak een tafel van 12, dus 1x12=12, 2x12=24 enzovoorts. Gebruik daarvoor (delen van) de code die hierboven staat. Gebruik alleen *while* die je in de vorige les hebt leren gebruiken zodat je efficiëntere code krijgt (= minder regels code).

Inleveren

Een screendump, *opdracht10-<jouw-naam>.png* van je gehele browser met oplossing.
Je laat de tafel van 12 zien en je gebruikt daarbij een *while* loop.

--

11 eindopdracht 1

Bekijk de volgende code:

```
num = 10


-- Test of het getal even is
if num % 2 == 0 then
  print("Het getal is even")
else
  print("Het getal is oneven")
end
```

Commentaar

Op regel 3 zie je in het begin --, dat is een manier in LUA om commentaar toe te voegen. Alles na de -- wordt dus niet gezien als code, maar is er om uit te leggen in 'gewone niet-code-taal' wat er gebeurt.

Opdracht

Maak LUA-code die alle even getallen tussen 0 en 100 afdrukt. het eerste getal dat wordt afgedrukt is dus 2 en het laatste getal is 98.

Tip: gebruik delen van de code hierboven en gebruik de *while* .

Eisen:

1. spring goed in (programma blokken)
2. maak gebruik van logische variabele namen
3. gebruik commentaar in je code.

Elk programmablok moet inspringen en een blok in een blok moet je dus twee maal laten inspringen!

Inleveren (1 bestand)

* Screendump, *eindopdracht1-<jouw-naam>.png* van je gehele browser met oplossing.

12 eindopdracht 2

Screenshot 2023-04-19 145429.png

Voor deze opdracht heb je alle kennis nodig van de voorgaande opdrachten.

Opdracht

Kies zelf een oneven getal tussen 0 en 100.
Zet in het commentaar bij je code welk getal jij hebt gekozen.

Maak een LUA-programma die alle oneven getallen tussen 0 en 100 (dus eerste is 1 en laatste is 99), bij elkaar optelt behalve het oneven getal dat jij hebt gekozen.

Eisen:

1. spring goed in (programma blokken)
2. maak gebruik van logische variabele namen
3. gebruik commentaar in je code

De output moet er als volgt uit zien:

```
Als je alle oneven getallen tussen 0 en 100 optelt, behalve het door mij gekozen getal, dan krijg je XX
```

Op de plaats van de XX komt natuurlijk het antwoord.

Zorg dat je de code begrijpt en dat je alleen dingen gebruikt die zijn uitgelegd!

Inleveren (1 bestand)

Screendump, *eindopdracht2-<jouw-naam>.png* van je gehele browser met oplossing.

Revision #4
Created 2024-06-08 13:29:06 UTC by Max
Updated 2024-06-08 18:34:00 UTC by Max