|  |  |
| --- | --- |
| Biostatica  Door Alistair Vardy, Bart van Trigt en Mark Schrauwen | Matlab Wk 4.1 |

Inhoudsopgave

[Versiebeheer 2](#_Toc497997503)

[1 Inleiding 3](#_Toc497997504)

[2 Het gestructureerd ontwerpen van een programma 4](#_Toc497997505)

[3 Het interpreteren van fouten 5](#_Toc497997506)

[3.1 Te veel of te weinig haakjes 5](#_Toc497997507)

[3.1.1 Opdracht: 5](#_Toc497997508)

[3.2 Problemen bij het vermenigvuldigen van matrices of vectoren 6](#_Toc497997509)

[3.2.1 Opdrachten 6](#_Toc497997510)

[3.3 Verkeerde index van een matrix of een vector 7](#_Toc497997511)

[3.3.1 Opdrachten 7](#_Toc497997512)

[3.4 Het gebruik van een enkel ‘=’- teken i.p.v. twee 8](#_Toc497997513)

[3.4.1 Opdrachten 8](#_Toc497997514)

[3.5 Toewijzing van een verkeerd aantal elementen 9](#_Toc497997515)

[3.5.1 Opdrachten 9](#_Toc497997516)

[3.6 Verkeerd gebruik van een operator 10](#_Toc497997517)

[3.6.1 Opdrachten 10](#_Toc497997518)

[3.7 Het niet afmaken van een commando 10](#_Toc497997519)

[3.7.1 Opdrachten 11](#_Toc497997520)

[3.8 Het vergeten van aanhalingstekens om een string 11](#_Toc497997521)

[3.9 Verkeerd schrijven van een functienaam 11](#_Toc497997522)

[3.9.1 Opdrachten 12](#_Toc497997523)

[3.10 Gebruiken van een functie die niet in het *path* of in dezelfde map staat 12](#_Toc497997524)

[3.10.1 Opdracht 12](#_Toc497997525)

[3.11 Het niet afsluiten van een if- of while statement 13](#_Toc497997526)

[3.11.1 Opdrachten 13](#_Toc497997527)

[3.12 Combinatie Opdrachten 13](#_Toc497997528)

[3.12.1 Opdracht 1 13](#_Toc497997529)

[3.12.2 Opdracht 2 14](#_Toc497997530)

[4 Google is je grootste vriend 15](#_Toc497997531)

[4.1.1 Opdracht 15](#_Toc497997532)

[4.2 Matlab forum voor vragen aan andere gebruikers 15](#_Toc497997533)

[4.3 Docenten 15](#_Toc497997534)

# Versiebeheer

|  |  |  |  |
| --- | --- | --- | --- |
| Versie | Datum | Beschrijving | Door |
| 0.0 | 11-07-2017 | Eerste versie | Alistair Vardy |
| 0.1 | 17-09-2017 | Veelvoorkomende fouten toegevoegd. | Alistair Vardy |
| 0.2 | 27-10-2017 | Opdrachten toegevoegd en reader doorgewerkt en verbeterd. | Bart van Trigt |
| 0.3 | 08-11-2017 | Reader afgemaakt en kan voorzien worden van feedback. | Bart van Trigt |
| 0.4 | 16-11-2017 | Feedback van Herre Faber verwerkt | Bart van Trigt |

# Inleiding

In dit deel gaan we verder in op het ontwerpen van een programma. Zoals je de afgelopen weken wel hebt gemerkt krijg je vaak genoeg zo’n vervelende foutmelding. Gelukkig dat we die foutmeldingen krijgen want anders krijg je fouten in de berekeningen. Toch kunnen die foutmeldingen aardig wat frustraties veroorzaken. Daarom zullen we in deze reader de nadruk leggen op het interpreteren van veel voorkomende foutmeldingen en hoe we die kunnen oplossen. Daarnaast laten we zien hoe we Matlab kunnen gebruiken om hulp te zoeken bij functies en problemen.

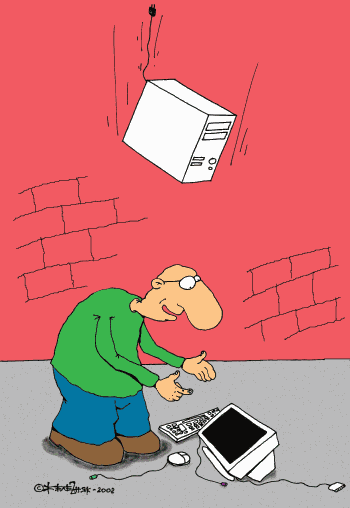
De volgende leerdoelen zijn hiervoor gemaakt. Na het doorwerken van deze reader ben je in staat:

1. elf veel voorkomende syntaxfouten (taalfouten) te interpreteren en op te lossen.
2. syntaxfouten in een bestaand Matlab bestand aan te passen.
3. zelf fouten op te lossen met behulp van de Matlab help functie.
4. fouten op te lossen door externe hulpbronnen, zoals Mathworks en Google, te benaderen.

Je hebt in reader 1.2 al wat geleerd over fouten, zo weet je dat er syntaxfouten zijn en programmeerfouten. In deze reader gaan we verder op deze fouten in.

**Zie je een fout? Of heb je een aanbeveling, dan horen we dat graag! Stuur dan een e-mail naar** [**mjschrau@hhs.nl**](mailto:mjschrau@hhs.nl) **en wij passen het dan z.s.m. aan.**

[**Je mag ook hier je suggesties doen.**](http://www.bewegingstechnologie.com/weblinks/curr17/Biostatica/bitbucket)



# Het gestructureerd ontwerpen van een programma

We hebben eerder gesproken over het maken van een programma. Naast tips over het gebruik van commentaar en netjes variabelen en functies een naam geven, hebben we ook gesproken over de fouten die je kunt maken. We gaan daar nu wat verder op in.

We herhalen eerst een aantal “best practices” voor het programmeren:

1. Beschrijf eerst wat het programma (of de functie) doet.
2. Geef aan wat de inputs en outputs zijn.
3. Geef het programma of de functie een naam die de lading dekt.
4. Geef variabelen een duidelijke naam zodat deze ondubbelzinnig is.
5. Gebruik nooit getallen die niet direct te verklaren zijn. Definieer deze eerst vooraf zodat jij of iemand anders meteen kan zien wat het is. Bijvoorbeeld het aantal proefpersonen of de samplefrequentie.
6. Geef een variabele **nóóit** dezelfde naam als van een functie, want je krijgt dan vervelende fouten. Bijvoorbeeld mean=mean(snelheid). Gebruik hier i.p.v. mean bijvoorbeeld meanSnelheid als ouput. De functie aanroep komt er dan zo uit te zien:

meanSnelheid=mean(snelheid).

# Het interpreteren van syntax fouten

We zullen een aantal veel voorkomende fouten demonstreren aan de hand van eenvoudige voorbeelden. Veel informatie in deze reader is gebruikt van twee websites. Wil je meer weten over errors kijk dan naar deze twee websites:

* <https://en.wikibooks.org/wiki/MATLAB_Programming/Error_Messages>
* <https://www.physicsforums.com/insights/5-common-matlab-error-messages-fix/>

## Te veel of te weinig haakjes

t = 1;

x = sin(t));

y = cos(t;

z = tan(t});

kolommen = [1 2 3]];

Deze fouten worden door de editor al opgemerkt en door een rood streepje aan de rechterkant gemarkeerd. Als deze commando’s toch uitgevoerd worden, volgt de volgende foutmelding:

x = sin(t));

x = sin(t));

↑

Error: Unbalanced or unexpected parenthesis or bracket.

Merk op dat de eerste fout al gemaakt wordt in de regel x = sin(t)). Daar moet 1 haakje staan in plaats van 2. Soms geeft de editor pas in een regel later een foutmelding hier bijvoorbeeld bij y = cos(t . Dat gebeurt wel vaker. Dus als je de fout niet direct kunt vinden op de plek waar Matlab het aangeeft, kijk dan ook eens wat verder terug of (mogelijk) verderop.

### Opdracht:

t = 1;

x = sin(t));

y = cos(t;

z = tan(t});

kolommen = [1 2 3]];

**Run het bovenstaande commando en maak alle regels kloppend, zodat je geen foutmeldingen meer krijgt.**

## Problemen bij het vermenigvuldigen van matrices of vectoren

Als we twee vectoren  en  willen vermenigvuldigen, dan moeten de afmeting n en k aan elkaar gelijk zijn. Dit zijn de binnenste afmetingen van de matrices als ze vermenigvuldigd worden. Als deze afmetingen niet overeenkomen dan is de foutmelding:

A = [1 2 3];

B = [4 5 6];

A\*B

Error using \*

Inner matrix dimensions must agree.

De andere optie is dat we de matrices of vectoren elementsgewijs willen vermenigvuldigen. In dit geval gebruiken we de operator ‘.\*’ i.p.v. ‘\*’.

### Opdrachten

**Vraag1**

A = [1 2 3];

B = [4 5 6];

Vermenigvuldiging=A\*B

**Je wilt vector A en vector B elementsgewijs vermenigvuldigen. Kopieer het bovenstaande commando en maak hem kloppend, zodat je geen foutmeldingen meer krijgt.**

**Vraag 2**

A = [7 5 3 4];

B = [4 5 6];

Vermenigvuldiging=A.\*B

**Run het bovenstaande commando.**

Je krijgt nu de foutmelding:

matrix dimensions must agree.

Dit komt omdat de 4 (het laatste element) van vector A niet elementsgewijs vermenigvuldigd kan worden, omdat vector B maar een 1 bij 3 vector is.

**Je wilt toch de eerste 3 elementen van beide vectoren vermenigvuldigen en er uiteindelijk een 1 bij 4 vector uitkrijgen. Bedenk een slimme manier om deze error op te lossen (TIP: je mag hier de vectoren aanpassen)**

## Verkeerde index van een matrix of een vector

Als we van een matrix of vector een element opvragen dat niet bestaat, dan geeft Matlab een foutmelding zoals dat het geval is bij de volgende commando’s:

A = [1 2 3;4 5 6];

v = [1 2 3 4];

A(10)

v(5)

Index exceeds matrix dimensions.

Matrix A heeft maar 6 elementen, dus het opvragen van het 10e element kan niet. Matlab geeft terecht een foutmelding. Hetzelfde geldt voor de vector v.

Als we een index opvragen die negatief is, gelijk aan nul is of niet gelijk aan een geheel getal is, krijgen we de volgende foutmelding

>> A = [1 2 3;4 5 6];

A(-1)

Subscript indices must either be real positive integers or logicals.

Dus onthoudt: indexen mogen alleen maar uit positieve gehele getallen bestaan. Soms heb je niet in de gaten dat daar iets misgaat, bijvoorbeeld als je i als index gebruikt en i = 2.00000000000000001. In dat geval moet je i even afronden met de function **round**.

### Opdrachten

**Vraag1**

Jaap wilt het tweede element opvragen van de vector kniehoek. Hij heeft het volgende stukje code geschreven. Run de onderstaande code:

kniehoek = [70 89 82 82 87 83 82 87 90];

kniehoek(0)

**Jaap krijgt een foutmelding. Los deze op.**

**Vraag2**

Jaap wilt nu het derde element opvragen van de vector kniehoek, hij denkt geleerd te hebben van zijn vorige fout en heeft het volgende stukje code geschreven. Run de onderstaande code:

kniehoek = [70 89 82 82 87 83 82 87 90];

kniehoek(3,1)

**Jaap krijgt een foutmelding, los deze op.**

## Het gebruik van een enkel ‘=’- teken in plaats van twee

Als we willen bepalen of een variabele gelijk is aan een bepaalde waarde, dan gebruiken we twee ‘=’-tekens. Als we één ‘=’-teken gebruiken, dan is het een toewijzing zoals in de eerste regel van de volgende code:

n = 5;

if n = 4

n = n^2;

end

if n = 4

↑

Error: The expression to the left of the equals sign is not a valid target for an assignment.

In de tweede regel gaat dus fout!

### Opdrachten

**Vraag1**

n = 5;

if n = 4

n = n^2;

end

**Maak het bovenstaande commando kloppend.**

**Vraag 2**

snelheid = 7;

if snelheid == 4

snelheidNieuw = snelheid^2;

elseif snelheid = 7

snelheidNieuw=snelheid^2

else

snelheidNieuw=0

end

**Maak het bovenstaande commando kloppend.**

## Toewijzing van een verkeerd aantal elementen

Als we een aantal elementen in een matrix of vector willen plaatsen, dan moet het aantal elementen (en de vorm daarvan) overeenkomen met het aantal (en de vorm) van die elementen. In het volgende voorbeeld gaat het dan ook fout, omdat we op een enkele plek drie elementen willen zetten:

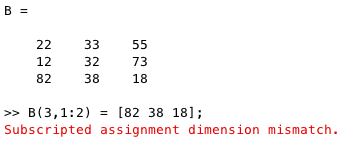
A = [1 2 3;4 5 6];

A(1) = [2 3 4];

Dit geeft de volgende foutmelding:

In an assignment A(:) = B, the number of elements in A and B must be the same.

Dit kan ook voorkomen:



of dit:



**Ga bij de twee voorgaande voorbeelden zorgvuldig na wat er mis is en hoe je het zou kunnen oplossen.**

### Opdrachten

**Vraag 1**

Carlo wil een mooie rijvector krijgen waarvan de waardes met gelijke stapjes oplopen. Hiervoor moet hij de eerste drie cijfers van vector A veranderen in: 1 2 3.

Run het volgende commando:

A = [4 2 7 4 5 6];

A(1) = [1 2 3];

**Hij krijgt een foutmelding. Wat moet hij doen zodat het wel goed gaat?**

**Los de foutmelding op.**

**Vraag 2**

Andreas is druk bezig om zijn data in orde te maken. Hij heeft de onderstaande matrix B gemaakt. Hij wilt de laatst rij vervangen door nieuwe waardes: 82 38 18. Hij doet het op de volgende manier. Run het volgende commando:

B = [22 33 55; 12 32 73; 28 17 32];

B(3,1:2) = [82 38 18];

**Hij krijgt een foutmelding. Wat moet hij doen zodat het wel goed gaat ?**

**Los de foutmelding op.**

## Verkeerd gebruik van een operator

Als we in een commando een operator te veel of te weinig zetten, zoals in het volgende voorbeeld, dan zal Matlab een foutmelding geven:

>> A(1:, 2)

A(1:, 2)

↑

Error: Unexpected MATLAB operator.

Matlab geeft aan waar de onverwachte operator staat.

### Opdrachten

**Vraag 1**

snelheid = 7;

if snelheid ~= 4

snelheidNieuw = snelheid^2;

elseif snelheid = 7

snelheidNieuw=snelheid^2

else

snelheidNieuw=0

end

**Run de bovenstaande code en maak kloppend.**

## Het niet afmaken van een commando

Als we vergeten een commando af te maken dan zal dat ook zichtbaar zijn in de editor. Aan de rechterkant krijg je een rood streepje zien en het commando krijgt een rood kringeltje onder de tekst. Als we het commando uitvoeren dan krijgen we een foutmelding:

>> str = 'test

str = 'test

↑

Error: Character vector is not terminated properly.

>> A = 1+2+

A = 1+2+

↑

Error: Expression or statement is incomplete or incorrect.

**Ga bij de twee voorgaande voorbeelden zorgvuldig na wat er mis is en hoe je het zou kunnen oplossen.**

### Opdrachten

**Vraag 1**

**Voer het onderstaande commando in en maak het kloppend, het antwoord moet 7 worden.**

A = +1+2+

**Vraag 2**

snelheid = 7;

if' snelheid == 4

Tempo='langzaam'

elseif snelheid == 7

Tempo=''Snel'

end

**Maak de bovenstaande code kloppend.**

## Het vergeten van aanhalingstekens rondom een string

Als we bijvoorbeeld een file willen inladen op de volgende manier:

data = load(datafile.txt);

Dan is het essentieel dat er om de filenaam aanhalingstekens worden geplaatst.

data = load(‘datafile.txt’);

Zo niet, dan krijgen we de volgende foutmelding:

Undefined variable "datafile" or class "datafile.txt".

## Verkeerd schrijven van een functienaam

Als we per ongeluk een functie (of variabele) verkeerd schrijven, zoals hieronder gegeven, dan zal Matlab ons vertellen dat de functie of variabele ongedefinieerd is:

>> A = [1 2 3;4 5 6];

gemiddelde = meam(A);

Undefined function or variable 'meam'.

**Nog een voorbeeld van verkeerde spelling**

onderbeenLengte = 40;

bovenbeenLengte = 50;

beenLengte = onderbeenLengte+bovenbeenlengte

Probeer dit uit en je ziet deze foutmelding:

>> beenLengte = onderbeenLengte+bovenbeenlengte

Undefined function or variable 'bovenbeenlengte'.

**Wat gaat er mis?**

Het kan ook zo zijn dat je denkt dat je een variabele hebt aangemaakt, maar hij staat niet in je Workspace of je hebt hem per ongeluk verwijderd.

>> A = [1 2 3;4 5 6];

clear A;

gemiddelde = mean(A);

Undefined function or variable 'A'.

### Opdrachten

**Vraag 1**

Bepaal de maximale waarde van de vector Versnelling.

**Run de onderstaande code en maak het kloppend.**

clearvars;

Versnelling = [3 7 18 9 12 1]

maximaleVersnelling = max(snelheid)

**Vraag 2**

Bereken de maximale waarde van de vector Versnelling.

**Run de onderstaande code.**

Versnelling = [3 7 18 9 12 1]

clear all;

maximaleVersnelling = max(Versnelling)

**Je krijgt nu een foutmelding. Los deze op.**

## Gebruiken van een functie die niet in het *path* of in dezelfde map staat

Wat was ook alweer het path? Dat is de complete reeks van mappen en submappen waar je in aan het werk bent. Deze kan je aanpassen in Matlab. Als je dat verkeerd doet, krijg je dezelfde foutmelding als in het probleem hierboven. Er zijn twee oplossingen:

* Zet de functie in dezelfde map.
* Voeg de map waar de functie wel staat toe aan het path.

### Opdracht

**Vraag1**

* **Maak een functie met de naam 'optellen'.**
* **Sla deze functie op in het mapje desktop.**
* **De input van deze functie bestaat uit twee scalaire waardes.**
* **De eerste scalaire waarde geef je de naam 'bovenbeen' en de tweede scalaire waarde de naam ‘onderbeen’.**
* **De output is de opgetelde waarde van het bovenbeen met het onderbeen.**

**De functie komt er als volgt uit te zien:**

function opgeteldeWaarde = optellen(bovenbeen, onderbeen)

opgeteldeWaarde= bovenbeen + onderbeen;

end

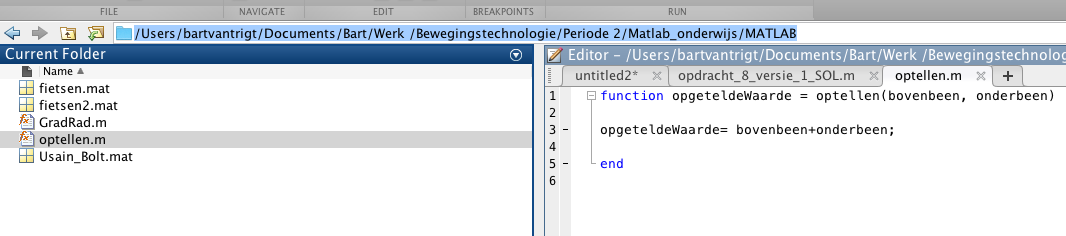
**Geef in je command window de waardes bovenbeen=9 en onderbeen=5 mee.**

**Roep je functie aan in je Command Window. Je krijgt nu waarschijnlijk een foutmelding.**

Undefined function or variable 'optellen'.

**Zet de functie in dezelfde map als waar het Path naartoe wijst. Je kan deze map vinden door te kijken naar de balk bovenin.**

**Run je functie opnieuw vanuit je Command Window en nu is de fout, als het goed is, verdwenen.**

****

Figuur . Dit figuur laat jou zien wat de map is waar je path naar toe wijst. LET OP: dit path is voor iedereen anders dus neem niet de map over zoals je hier ziet!

## Het niet afsluiten van een if- of while statement

Als we vergeten aan het eind van een if- of while statement het commando end te plaatsen, dan weet Matlab nog niet dat we er klaar mee zijn. Dit kan leiden tot de situatie waarin Matlab wacht op een volgend commando, dat dus helaas nooit komt.

### Opdrachten

**Vraag1**

**Run het onderstaande commando.**

snelheid = 7;

if snelheid == 4

Tempo='langzaam'

elseif snelheid == 7

Tempo='Snel'

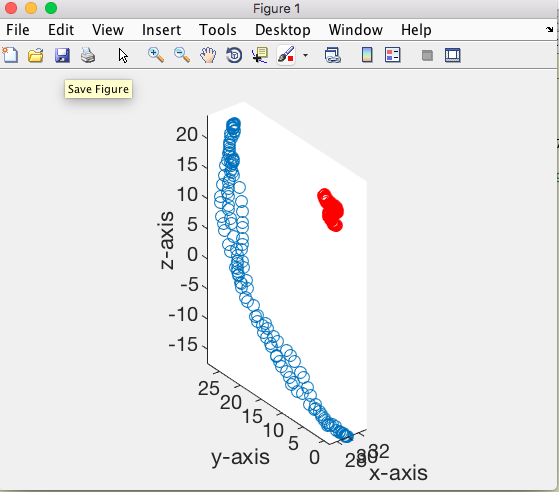
**Krijg je een foutmelding of niet?**

**Maak de code kloppend.**

## Combinatie Opdrachten

### Opdracht 1

In het zipbestand wk4\_1\_opdracht1.zip zit een ‘.mat’ file met data en een script met de naam: ‘maak\_de\_code\_correct.m’. In dit script staan ongeveer acht fouten. Het is de bedoeling dat jij de code kloppend gaat maken. Kijk goed naar de error die je krijgt en zoek die error in de hierboven behandelde paragrafen op. Aan de hand daarvan kan je de code oplossen en weet je waar de fout zit. Daarnaast staat er op de plek van de fout nuttige informatie. Deze informatie kan je gebruiken om de code kloppend te maken. Maak de code kloppend, zodat je uiteindelijk het onderstaande figuur eruit krijgt:



### Opdracht 2

In het zipbestand wk4\_1\_opdracht2 zit een script met de naam: ‘maak\_de\_code\_correct\_opdracht2.m’. In dit script staan fouten. Het is de bedoeling dat jij de code kloppend gaat maken. Kijk goed naar de error die je krijgt en zoek die error in de hierboven behandelde paragrafen op. Aan de hand daarvan kan je de code oplossen en weet je waar de fout zit. Je kan nuttige informatie vinden op de plek waar de error zit. Gebruik deze informatie om de code kloppend te maken.

Je komt er vanzelf achter wanneer je code kloppend is. Hiervoor moet je wel het geluid van je computer aanzetten!

# Programmeerfouten

Programmeerfouten zorgen dat het programma iets anders doet dan je zou willen. Hierbij kan het zelfs gebeuren dat er geen foutmelding komt. Dat betekent dat jij, de programmeur, denkt dat jouw programma werkt, terwijl er in werkelijkheid niets van klopt. Dat is hetzelfde als denken dat je naar huis reist, maar aan het einde van je reis aan de andere kant van het land bent. Je bent dan van A naar B gereisd, de reis is technisch gesproken goed gegaan, je hebt geen ongelukken gehad, maar je had bij C moeten uitkomen.

We gaan niet heel erg diep in op programmeerfouten maar we bespreken de drie meest voorkomende.

## Het niet sluiten van een if-statement

Je hebt geleerd dat je een if of while- statement altijd moet sluiten met een end. Doe je dit niet dan ga je problemen krijgen. Maar je krijgt geen foutmelding. We gaan het eens onder de loop nemen. **Voer de volgende code in.**

balSnelheid = 75;

if balSnelheid > 75

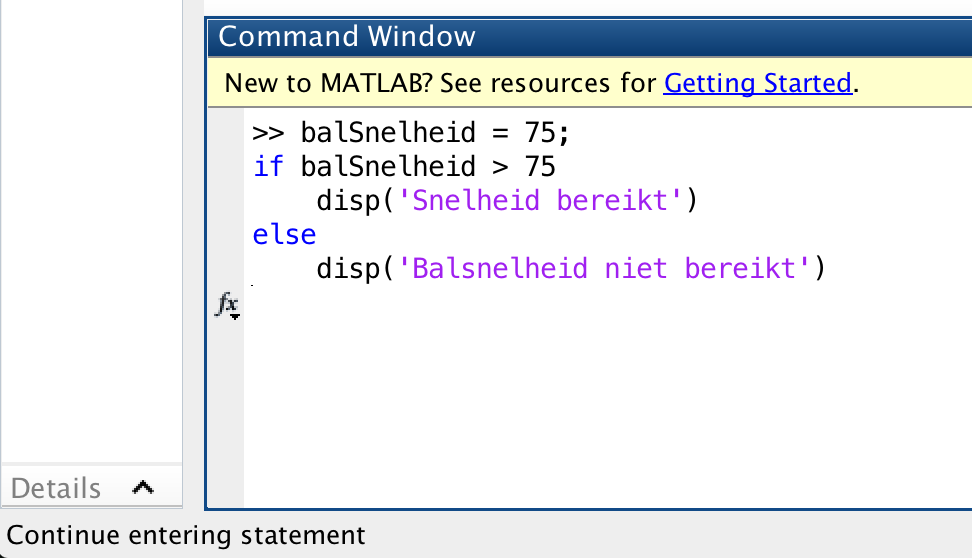
disp('Snelheid bereikt')

else

disp('Balsnelheid niet bereikt')

**Wat merk je dat er gebeurd? Probeer nu eens 1+1 in je Command Window uit te voeren. Komt hier een waarde uit?**

Waarschijnlijk niet, dit komt omdat je de if-statement niet hebt afgesloten met een end, vervolgens zie je in onder in matlab het volgende staan: ‘Continue entering statement’. Je kan door gaan door eerst end in te typen.

****

## Programmeerfouten for-loop

### Overschrijven van een variabele binnen een for-loop

Stel we hebben twee vectoren en die willen we bij elkaar optellen. Dat kunnen we snel doen met een for-loop.

**Run de onderstaande code.**

VectorB = 0:1:10;

VectorC = 10:1:20;

for i = 1:length(VectorC)

VectorA = VectorB(i) + VectorC(i);

end

**Kreeg je een foutmelding wanneer je de code liet runnen? Waarschijnlijk niet maar toch is er iets wat niet klopt. Wat valt je op als je kijkt naar de aangemaakte variabele VectorA?**

**Laat nu de onderstaande Code runnen.**

VectorB = 0:1:10;

VectorC = 10:1:20;

for i = 1:length(VectorC)

VectorA(i) = VectorB(i) + VectorC(i);

end

**Heb je de programeer fout kunnen vinden?**

### Een verkeerd element aanroepen in een for-loop

Net hebben we gezien dat een variabele telkens wordt overschreven. Het kan ook zijn dat je in een for-loop voor je gevoel alles netjes hebt geprogrammeerd maar toch een foutje hebt gemaakt. Bijvoorbeeld door niet de i ergens te gebruiken.

In het onderstaande voorbeeld wil je vectorB element per element vermenigvuldigen met vectorC.

VectorB = 0:1:10;

VectorC = 0:1:10;

for i = 1:length(VectorC)

VectorA(i) = VectorB(i).\*VectorC(1);

end

**Wat zie je nu als je kijkt naar VectorA?**

Is dit wat je ook echt wilde bereken?

**Laat nu de onderstaande code runnen.**

VectorB = 0:1:10;

VectorC = 0:1:10;

for i = 1:length(VectorC)

VectorA(i) = VectorB(i).\*VectorC(i);

end

**Komt hier iets anders uit? Heb je de programmeerfout kunnen vinden?**

## Een variabele overschrijven

Wanneer je in een programma ergens een variabele aanmaakt en deze vervolgens dezelfde variabele aanmaakt krijg je problemen. Dus zorg ervoor dat je nooit twee keer dezelfde variabele aanmaakt. Hieronder een voorbeeld. We willen van de positie het eerste element optellen bij het tweede element en het tweede bij het derde element etc. **Laat de onderstaande code runnen**

clc; clear all;

positie = 0:9;

forloopPositie = ones(9,1);

for i = 1:length(positie)-1

forloopPositie(i) = positie(i) + positie(i+1);

end

forloopPositie = 7:10;

**Wat is de waarde die je krijgt voor de variabele forloopPositie? Klopt deze met wat je wilde doen?**

# Google is je grootste vriend

Wanneer een vraagstuk tegenkomt en je weet niet hoe je het moet oplossen, bedenk dan altijd: ‘ik ben vast niet de enige met dit probleem’. Als je niet de enige bent, dan is daar Google voor je om te kijken of iemand kan helpen.

### Opdracht

Plot een cirkel met een straal van 2 en het centrum is (2,3).

Geen idee hoe het moet? Google het! Je krijgt het beste resultaat als je het in het Engels googelt. Gebruik dus niet cirkel, straal en centrum, maar: circle, radius en centre of center. Gebruik niet ‘tekenen’, maar ‘draw’.

## Matlab forum voor vragen aan andere gebruikers

Waarschijnlijk ben je bij de laatste opgaven uitgekomen bij het Matlab forum. Dit is het forum waar Matlab gebruikers elkaar helpen:

<https://nl.mathworks.com/matlabcentral/answers/index>

Hier staat een community van experts en gebruikers klaar om elkaar te helpen. Kijk naar de eerder gestelde vragen.

## Docenten

Mocht je er echt niet meer uitkomen, dan zijn er natuurlijk nog de Matlab Goeroes, namelijk jullie docenten: Bart, Mark en Herre. Daarnaast zijn er meerdere docenten op de opleiding die ook wel weg weten met Matlab: Caroline, Aad en Hubert zijn hier voorbeelden van.