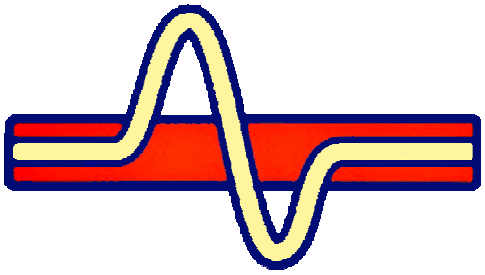
Coach6



Handboek CoachTaal

Vragen en oplossen van problemen
Heeft u, bijv. n.a.v. de installatie nog vragen of problemen, kijk dan in de FAQ Software op de CMA-website (http://www.cma-science.nl rubriek 'Ondersteuning'), of stuur uw vraag naar helpdesk@cma-science.nl

Hardware en software worden ontwikkeld en gedistribueerd door de Stichting CMA.

Tekst: Vincent Dorenbos, Ewa Kędzierska
Revisie 6.32, 15 februari 2011

Stichting CMA
Van Leijenberghlaan 124 (unit B),
1082 DB Amsterdam
Telefoon: 020 7600920
Fax: 020 7600929
E-mail: info@cma-science.nl
Internet: http://www.cma-science.nl/

(Lege pagina)

Inhoudsopgave

CoachTaal: Introductie

CoachTaal is een eenvoudige programmeertaal die in Coach 6 gebruikt wordt:

- Voor formules, om berekeningen uit te voeren met gegeven in diagrammen of tabellen;
- Bij modelleren voor de formules in grafische modellen of voor het schrijven van tekst-modellen.
- Om programma's te maken in stuuractiviteiten.

Coach controleert tijdens de uitvoering van een model, programma of formule of de expressies en opdrachten correct zijn. Zo niet kunnen deze niet worden uitgevoerd er verschijnt dan een foutmelding op het scherm.

In de volgende onderwerpen worden alle elementen van de CoachTaal behandeld in volgorde van toenemende moeilijkheid:

- De spelling van namen en getallen;
- Gereserveerde woorden en karakters;
- Het vormen van Expressies (inclusief variabelen);
- Het vormen van Opdrachten (toekenningen, voorwaardelijke - en lusopdrachten);
- Standaard CoachTaalfuncties en -procedures, Standaard wiskundige functies en speciale wiskundige functies;
- Een overzicht van fouten en foutmeldingen.

Zie ook:

- Een vergelijking van CoachTaal met andere programmeertalen

CoachTaal: Namen en getallen

In eenmodel, programma of formule gebruik je namen voor het aanduiden van constanten, startwaarden en variabelen (grootheden), procedures en functies. Je hebt een grote vrijheid in het bedenken van geschikte namen. Op deze pagina worden de regels voor het vormen van correcte namen uiteengezet.

Namen

Bij het samenstellen van namen *is het toegestaan* om karakters uit de volgende verzamelingen te gebruiken:
- { **A B C .. Z** } hoofdletters;
- { **a b c .. z** } kleine letters;
- { **1 2 3 .. 0** } cijfers;
- { **£ _ & ~ ! | { } [ ]** } symbolen;

De CoachTaal-interpreter van Coach 6 maakt wél onderscheid tussen hoofd- en kleine letters. Bijvoorbeeld de variabelen BevolkingsGROEI en bevolkingsgroei zijn herkend als twee verschillende namen. Bij het samenstellen van namen zijn bepaalde karakters en woorden gereserveerd voor Coach. Deze mogen niet gebruikt worden. Bij deze woorden maakt Coach geen onderscheid tussen hoofd- en kleine letters.

N.B.:

- Dit is een nieuw aspect in Coach 6. Een bijkomstigheid is dat als de gebruiker in een Coach 5model, programma of formule hoofd- en kleine letters in namen door elkaar heeft gebruikt, deze in Coach 6 niet langer zullen werken. Corrigeer na openen eerst de spelling m.b.t. hoofd/kleine letters, daarna zal het programma, tekstmodel of de formule weer correct werken.
- Het is niet toegestaan een getal te gebruiken als het eerste karakter van een naam.

Getallen

Bij het samenstellen van getallen kun je karakters uit de volgende verzameling gebruiken:

{ **0 1 2 3 4 5 6 7 8 9 + - . e E** }

Gebruik de letter **e** of **E** om machten van 10 aan te duiden (voor wetenschappelijke notatie), bijv.:
Het grootste aantal significante cijfers is elf. Het kleinste geaccepteerde positieve getal is $1{,}0e^{-36}$, het grootste positieve getal 1,0e+36. Een foutmelding verschijnt als deze grenzen overschreden worden.
1,5e-3 = 1,5E-3 = 0,0015 = 1,5 * 10^-3.

N.B.:

- Decimale cijfers worden weergegeven afhankelijk van de Windows instelling van het decimale scheidingsteken.
- Een getal mag niet beginnen met een decimaal scheidingsteken.

CoachTaal: Gereserveerde woorden en karakters

Gereserveerde karakters

Sommige karakters hebben een speciale betekenis in Coach. Het is niet toegestaan om deze te gebruiken in namen van constanten en variabelen, procedures en functies. Deze karakters zijn:

- Karakters met ASCII-codes: **0 .. 31**
- Tekens voor de rekenkundige operatoren: **( ) - + * / , ^ = < >**
- **Spaties** en de tekens: **' . , % " : ; $ # @**

N.B.:

- Door een naam tussen vierkante haken te plaatsen gelden bovenstaande beperkingen niet. Het eerste karakter van de naam is dan "[" en het laatste karakter "]", bijv. [2πr] is toegestaan als variabelenaam.
- De volgende karakterparen hebben een speciale betekenis voor de syntax van expressies en condities (en mogen alleen gebruikt worden in de juiste context): **<= >= := <>**

Gereserveerde woorden

Sommige woorden hebben een speciale betekenis in Coach. Het is niet toegestaan om ze als namen voor constanten en variabelen, procedures of functies te gebruiken. Deze woorden staan in de tabel hieronder:

Aan	EindDoe	Niet	Teken
Abs	EindFunctie	Niveau	Tel
Afgeleide	EindProcedure	Of	Teller
AfgeleideGlad	En	Pi	TotDat
Als	Entier	Procedure	TotHier
Anders	Exp	Puls	TussenTijd
ArcCos	Fac	PulsHerhaald	TweedeAfgeleide
ArcSin	Filter	Rand	TweedeAfgeleideGlad
ArcTan	Functie	Repeteer	Uit
Bezier	Geluid	ResetTellers	Wacht
Bit	Herhaal	Round	WasBitHoog
Cos	Histogram	Sin	WasBitLaag
Dan	Index	SlaOp	WisData
Delta	Integraal	Som	Wordt
DeltaFil	Kolom	Spline	ZetAan
Doe	Ln	Sqr	ZetAanAbsoluut
Domein	Log	Sqrt	ZetNiveau
DrukAf	LoopTijd	Stop	ZetUit
EenheidStap	Max	Stopwatch	ZetUitAbsoluut
EindAls	Min	Tan	Zodra
			Zolang

N.B.: De gereserveerde woorden **Aan**, **Uit** en **Pi** [π] zijn de namen van constantes. De waarden van deze constantes is respectievelijk 255, 0 en 3,141592654… Ze kunnen niet worden veranderd.

CoachTaal: Variabelen en constanten

Gewone variabelen

Een variabele kan worden gezien als een naam voor een veranderende numerieke waarde (getal). De naam van de variabele kan vrij gekozen worden, maar moet voldoen aan de regels voor namen en getallen van CoachTaal.
De waarde van een rekenkundige expressie toekennen aan een variabele betekent dat de waarde wordt opgeslagen in een geheugenplak van de computer onder die naam. Zo betekent "Totaal := 5 + 6" dat de waarde "11" wordt opgeslagen onder de naam "Totaal".
Zoals het woord al aangeeft zal de opgeslagen waarde onder de naam kunnen variëren tijdens de uitvoering van het programma of model. Een geheugenplaats met een naam en een waarde die niet verandert noemen we een **constante**.
Variabelen in CoachTaal zijn altijd *globaal*. Dit betekent dat ze beschikbaar zijn in het gehele programma dat wordt geschreven, inclusief alle functies en procedures.

Logische variabelen

Het resultaat van een logische expressie is ook een waarde en deze wordt ook opgeslagen in het computergeheugen. Dit wordt een *logische variabele* (of Boole'se variabele) genoemd.
Is het resultaat van een logische expressie **Aan** (Waar/Hoog), dan wordt de geheugenplaats gevuld met de waarde **255**. Is het resultaat **Uit** (Onwaar/Laag), dan wordt de geheugenplaats gevuld met de waarde **0**. De voorkeur voor de namen **Aan** en **Uit** voor resp. Waar en Onwaar in CoachTaal heeft te maken met de stuurmogelijkheden die beschikbaar zijn.
Tip: In formules kun je zowel de kolomgrootheden als de kolomnamen zelf (**C1** .. **C8**) als variabelen gebruiken.

Rekenkundige operatoren

Operator	Werking	Prioriteit
-	Teken omkeren	1
^	Machtsverheffen	1
*	Vermenigvuldigen	2
/	Delen	2
+	Optellen	3
-	Aftrekken	3

N.B.:

- Teken omkeren en machtsverheffen hebben onderling gelijke prioriteit (prioriteit 1), net zoals vermenigvuldigen en delen (prioriteit 2) en optellen en aftrekken (prioriteit 3).
- Indien de operatoren in een expressie gelijke prioriteit hebben wordt de expressie van links naar rechts geëvalueerd. De volgorde van de operatoren in een expressie kan altijd door de gebruiker worden bepaald via het gebruik van haakjes.

Voorbeelden

Zie ook:

- Standaard wiskundige functies;

- Speciale wiskundige functies;
- Een formule maken

Logische operatoren

Logische operatoren (of Boole'se operatoren) werken op logische waarden, bijvoorbeeld op het resultaat van relationele operatoren. In CoachTaal zijn er drie logische operatoren:

Operator	Werking	Prioriteit
NIET	logische waarde omkeren	1
EN	logische EN	2
OF	logische OF	3

Waarheidstabellen voor de logische operatoren

N.B.: S1 en S2 in deze tabel staan voor uitspraken zoals bijv. "A>B" of "Aantal<1".

	NIET
S1	**NIET** S1
AAn	**Uit**
Uit	**Aan**

	EN	
S1	S2	S1 **EN** S2
Aan	**Aan**	**Aan**
Aan	**Uit**	**Aan**
Uit	**Aan**	**Aan**
Uit	**Uit**	**Uit**

	OF	
S1	S2	S1 **OF** S2
Aan	**Aan**	**Aan**
Aan	**Uit**	**Aan**
Uit	**Aan**	**Aan**
Uit	**Uit**	**Uit**

Voorbeelden

CoachTaal: Relationele operatoren

Relationele operatoren werken altijd op twee waarden. Deze waarden worden met elkaar vergeleken. Het resultaat is een logische variabele (Boole'se variabele): één van de logische constantes **Aan** (**Waar**) of **Uit** (**Onwaar**).
Door gebruik te maken van logische operatoren kunnen complexe relationele expressies samengesteld worden.
In zulke gevallen moeten de relationele expressies altijd tussen haakjes geplaatst worden.

Operator	Vergelijking	Prioriteit
=	is gelijk aan	4
<>	is niet gelijk aan	4
<	is kleiner dan	4
>	is groter dan	4

<=	is kleiner dan of gelijk aan	4
>=	is groter dan of gelijk aan	4

N.B.:

- Alle relationele operatoren hebben gelijke prioriteit (prioriteit 4).
- Als de operatoren in een expressie gelijke prioriteit hebben, dan wordt de expressie geëvalueerd van links naar rechts. De prioriteit van een operator in een expressie kan altijd door de gebruiker worden herroepen door het gebruik van haakjes.

Voorbeelden

CoachTaal: Waar en Onwaar

Een voorwaarde is ofwel **Waar** of **Onwaar** (respectievelijk overeenkomend met de logische waarde **1** en de logische waarde **0**).

Voor **Waar** wordt in CoachTaal de term **Aan** gebruikt.
Aan komt overeen met de numerieke waarde **255**.

Voor **Onwaar** wordt in CoachTaal de term **Uit** gebruikt.
Uit komt overeen met de numerieke waarde **0**.

CoachTaal: Expressies

Onder een expressie verstaan we elke uitdrukking die **een waarde oplevert**. Een expressie kan een combinatie zijn van variabelen, operatoren, constanten en functies. Er zijn twee groepen expressies:

1. Eenvoudige expressies

Een eenvoudige expressie is:

- Een los getal, zoals **6,13** of **105**
- Een losse constante, zoals ; **Pi**, **Aan**, **Uit**
- Een variabele zoals **x**, of **tijd**
- Een aanroep van een standaard wiskundige functie, een standaard CoachTaalfunctie of een speciale wiskundige functie. Een functie levert ook **altijd** een waarde op.

2. Expressies met operatoren (Voorbeelden)

Een expressie met operatoren bestaat uit waarden en operatoren. Een operator werkt op één of meerdere waarden. Een operator is een voorschrift waarmee waarden worden omgezet in andere waarden. Er zijn drie verschillende soorten operatoren: rekenkundige operatoren, logische operatoren en relationale operatoren.

Voorbeelden:
In de expressie **3*(V+1)**, werkt de operator "*" op de waarden "**3**" en "**(V+1)**".

N.B.: In expressies met meer dan één operator, bepaalt de prioriteit van elke operator de volgorde van uitvoering. De prioriteiten zijn aangeduid op de pagina over Operatoren (zie link hierboven). De gebruiker kan de volgorde vastleggen door het juiste gebruik van haakjes.

Voorwaardelijke expressies
Expressies met logische en/of relationele operatoren worden ook gebruikt om voorwaardelijke opdrachten te definiëren. Een expressie kan worden gebruikt als een voorwaarde, als het resultaat één van de Boole'se waarden Waar (**Aan**) of Onwaar (**Uit**) is.
Bijvoorbeeld:

- Temperatuur > 15
- Helderheid <> (l+1)/l
- (Helderheid > 20) EN (Kleur <> Rood)

Voorbeelden van voorwaardelijke expressies

(x>1) EN (x<2)
Bijvoorbeeld, als x=1,3 dan heeft de voorwaarde (conditie) de waarde **Waar** (**Aan**).
Als x=4 dan heeft de voorwaarde de waarde **Onwaar** (**Uit**).

NIET (Temperatuur >100)
De conditie heeft de waarde **Waar** (**Aan**) als de temperatuur kleiner of gelijk is aan 100, en **Onwaar** (**Uit**) in alle andere gevallen.

(y<-1) OF (y>1)
De conditie heeft de waarde **Waar** (**Aan**) als bijv. y = 2, en **Onwaar** (**Uit**) als bijv. y = 0,2.

(z>1) EN (z<2) OF (z>5) EN (z<6)
De conditie heeft de waarde **Waar** (**Aan**) als z een waarde tussen 1 en 2 heeft, of tussen 5 en 6, en **Onwaar** (**Uit**) in alle andere gevallen.

Zie ook:
- CoachTaal-opdrachten;
- Gereserveerde woorden en karakters

De syntax van expressies

Algemene regels
Gebruik van spaties en <Enter> (zgn. 'linebreaks' (harde regelafbrekingen))
- In een **logische expressie** moeten spaties worden geplaatst om de logische operatoren te scheiden van de variabele(n) of waarde(n).
- In een **rekenkundige expressie** of een **relationele expressie** is het gebruik van spaties optioneel.
- Een expressie hoeft niet op één regel te staan. Omwille van de duidelijkheid kan het bijvoorbeeld gewenst zijn om een expressie met meer dan één functie over meerdere regels te schrijven.
- Expressies moeten van elkaar worden gescheiden door spaties of door <Enter> ('hard return'). Bij scheiding door spaties kunnen twee of meer expressies op één regel geplaatst worden. Bij gebruik van <Enter> als scheiding tussen expressies wordt elke expressie op een nieuwe regel geplaatst.

Gebruik van haakjes ( )
- In een expressie kunnen haakjes gebruikt worden om de prioriteit van een operatie te veranderen.
- Haakjes kunnen ook gebruikt worden om een expressie te verduidelijken.

Expressies met operatoren
Indien een operator op een losse waarde werkt, wordt de waarde achter de operator geplaatst. Werkt een operator op twee waarden, dan wrodt de operator tussen de waarden in geplaatst. Operatoren met priosriteit 1 worden eerst uitgevoerd, dan operatoren met prioriteit 2, enz.

Voorbeeld: Interpretatie van expressies

Expressies met rekenkundige operatoren	Interpretatie
A*B^C	A*(B^C)
A^-B*C	(A^(-B))*C
A^-B-C	(A^(-B))-C
A/B-C	(A/B)-C
A+B*C	A+(B*C)
A-B*-C	A-(B* (-C))
a3*x^3 + a2*x^2 + a1*x + a0	(a3*x^3)+(a2*x^2)+(a1*x)+a0
Expressies met logische operatoren	**Interpretatie**
NIET Droog+1	(**NIET**(Droog))+1
Aan OF NIET(Beschikbaar)	**Aan OF** (**NIET**(Beschikbaar))
Droog **EN** Wind **OF** Zon	(Droog **EN** Wind) **OF** Zon
Zon **OF** Droog **EN** Wind	Zon **OF** (Droog **EN** Wind)
Expressies met relationele operatoren	**Interpretatie**
X<Z/(Z-1)	X<(Z/(Z-1))
X>=1+Y	X>=(1+Y)
X*2<=Y+5	(X*2)<=(Y+5)
(A<=B) **EN NIET**(C>B)	(A<=B) **EN** (**NIET**(C>B))

N.B.:

- Houd in gedachten, bij het vergelijken van een logische variabele met een waarde, dat de logische waarden **Aan** en **Uit** overeenkomen met de numerieke waarden 255 en 0. Een logische waarde heeft echter automatisch de waarde **Aan** indien zijn waarde niet gelijk is aan 0.
- Mogelijk geef je er de voorkeur aan om te werken met de namen *Waar* en *Onwaar* in plaats van de namen **Aan** en **Uit**. Dit is alléén toegestaan indien je de volgende toekenningen in je programma opneemt:
  Waar := Aan
  Onwaar := Uit

CoachTaal: Opdrachten

Een opdracht is een programma-element dat onafhankelijk kan worden uitgevoerd. We maken onderscheid tussen enkelvoudige opdrachten en structuuropdrachten.

Enkelvoudige opdrachten

1. Toekenningen
Met een toekenning wordt de waarde van een variabele vervangen door het resultaat van een expressie. Het symbool voor de toekenningsoperator is **':='** (dubbele punt is-gelijk). In CoachTaal kun je ook het symbool '=' of het woord **wordt** gebruiken. Het is niet nodig om de operator van de overige symbolen te scheiden met spaties.

Syntax:

variabele := expressie

Voorbeelden

- X := Y + Z
- Nat **wordt** (regen) **EN** (**NIET**(paraplu))

2. Procedure-aanroep
Procedures worden gebruikt om een programma structuur te geven. Een procedure bestaat uit één of meer opdrachten en draagt een naam. Het resultaat van een procedure is de uitvoering van de ingesloten opdrachten.
De opdrachten worden uitgevoerd op de positie waar een aanroep naar de procedurenaam in het programma staat. In Coach zijn er verschillende typen van procedures mogelijk:

- Standaard CoachTaalprocedures.
- Eigen Commando's, d.w.z. procedures gemaakt door de eindgebruiker.
- Een serie vooraf gemaakte procedures door de maker van de activiteit (zgn. Microwerelden).

Structuuropdrachten

Een structuuropdracht is een opdracht met een voorgeschreven structuur. Er zijn twee typen:

- voorwaardelijke opdracht (Als..Dan of Zodra..Doe);
- Lusopdrachten om een aantal opdrachten te herhalen in een lus, al dan niet bestuurd door een voorwaarde.

Voorwaardelijke opdrachten

Een voorwaardelijke opdracht is een opdracht van de vorm 'Als .. Dan ..' of 'Zodra .. Doe'. Deze opdracht voert een reeks opdrachten uit, afhankelijk van de waarde van een voorwaardelijke expressie. Deze voorwaardelijke expressie moet één van de Boole'se waarden opleveren:Waar (**Aan**) of Onwaar (**Uit**).

Er zijn twee verschillende typen 'Als .. Dan'-opdrachten:

Als .. Dan .. EindAls
Als de voorwaarde **Waar** (**Aan**) is, dan worden de opdrachten tussen **Dan** en **EindAls** uitgevoerd. Is de voorwaarde **Onwaar** (**Uit**), dan worden de opdrachten niet uitgevoerd.

Syntax:
Als voorwaardelijke expressie **Dan**
    Opdrachten
EindAls

Als .. Dan .. Anders .. EindAls
Als de voorwaarde **Waar** (**Aan**) is, dan worden de opdrachten tussen **Dan** en **Anders** uitgevoerd.
Is de voorwaarde **Onwaar** (**Uit**), dan worden de opdrachten tussen **Anders** en **EindAls** uitgevoerd.

Syntax:
Als voorwaardelijke expressie **Dan**
    Opdrachten
Anders
    Opdrachten
EindAls

Zodra .. Doe .. EindDoe
Als de conditie **Waar** (**Aan**) is, dan worden de opdrachten tussen **Doe** en **EindDoe** uitgevoerd. Deze opdracht wordt gebruikt in modelleren voor een Voorval. Met een Voorval kan een plotselinge verandering van een toestandsvariabele gemodelleerd worden, bijv.

het stuiteren van een bal (tijdens de stuitering verandert de richting en/of grootte van de snelheid plotseling).

Syntax:
Zodra voorwaardelijke expressie **Doe**
Opdrachten
EindDoe

Voorbeelden van voorwaardelijke opdrachten

1. Als .. Dan .. EindAls:
```
Als x>0 Dan
    y:= sqrt(x)
EindAls
```

2. Als .. Dan .. Anders .. EindAls:

```
Als (a<2) EN (b>5) Dan
    a:=a + 1
    b:=b – 5
Anders
    a:=a –1
    b:=b + 3
EindAls
```

3. Zodra .. Doe .. EindDoe:

```
Zodra x>0 Doe
    v:= -0,9*v
EindDoe
```

CoachTaal: Lusopdrachten

In een lusopdracht wordt een groep opdrachten een aantal keren uitgevoerd. Er zijn drie verschillende typen lusopdracht, waarvan Herhaal..TotDat en Zolang..Doe bestuurd worden door een voorwaardelijke expressie. Repeteer..TotHier is een onvoorwaardelijke lusopdracht.

Herhaal .. TotDat
Herhaal .. TotDat herhaalt de opdrachten tussen **Herhaal** en **TotDat** zolang de voorwaarde **Onwaar** (**Uit**) is. De opdrachten worden tenminste één keer uitgevoerd.

Syntax:
Herhaal
Opdrachten
TotDat voorwaardelijke expressie

Voorbeeld: Herhaal .. TotDat

```
Herhaal
    ZetAan(1)
    ZetUit(1)
TotDat LoopTijd > 125
```

```
Herhaal
    Als x>1 Dan
        ZetAan(2)
    Anders
        ZetUit(2)
    EindAls
TotDat x>2
```

Zolang .. Doe .. EindDoe
Zolang .. Doe .. EindDoe herhaalt de opdrachten tussen **Doe** en **EndDoe** zolang de voorwaarde achter Zolang **Waar** (**Aan**) is.
Indien de voorwaarde **Onwaar** (**Uit**) is bij de start, dan worden de opdrachten ***niet*** uitgevoerd.

Syntax:

```
Zolang voorwaardelijke expressie Doe
    Opdrachten
EindDoe
```

Voorbeeld: Zolang .. Doe

```
Zolang t>1 Doe
    ZetAan(1)
    ZetUit(1)
EindDoe
```

```
Zolang r<> 3 Doe
    Als x>1 Dan
        ZetAan(2)
    Anders
        ZetUit(2)
    EindAls
EindDoe
```

Repeteer Aantal **.. TotHier**
Repeteer .. TotHier herhaalt de opdrachten tussen **Repeteer** en **TotHier** een vast aantal keren (bepaald door de parameter 'Aantal').
De opdrachten worden tenminste één keer uitgevoerd.

Syntax:

```
Repeteer Aantal
    Opdrachten
TotHier
```

Voorbeeld: Repeteer .. TotHier

```
Repeteer 10
    Wacht(0,5)
    ZetAan(1)
    Wacht(0,5)
```

```
    ZetUit(1)
TotHier

Repeteer 1000
    Als x>1 Dan
        ZetAan(2)
    Anders
        ZetUit(2)
    EindAls
TotHier
```

Foutmeldingen

Bij het schrijven van programma's of (tekst)modellen kunnen twee soorten fouten optreden:

- Syntax-fouten, die gedetecteerd worden tijdens de interpretatie van het progamma of model in de fase voor uitvoering, of net voor de toewijzing van variabelen aan diagramassen.
- Uitvoeringsfouten tijdens de uitvoering van het programma of model.

Foutmeldingen tijdens de interpretatie (Voorbeelden)

Indien een fout wordt gedetecteerd tijdens de interpretatie, verschijnt het programma of model op het scherm, tezamen met een foutmelding die aangeeft wat er aan de hand is. De cursor knippert achter de plaats waar de fout ontdekt is.

Foutmelding	Reden
"Onverwacht karakter"	Er is een karakter geplaatst waar het niet wordt verwacht.
"Geen geldig getal"	De interpreter kan een getal niet lezen.
"De naam is al in gebruik. Voer een andere naam in."	Verschijnt wanneer twee variabelen, functies of procedures een identieke naam hebben.
"Type-conflict"	Verschijnt wanneer een waarde is toegekend aan de naam van een procedure, of wanneer het resultaat van de functie toegekend wordt buiten de functie.
"... verwacht"	De interpreter verwacht een naam of een symbool.
"... NIET verwacht"	De interpreter verwacht het ingevoerde symbool niet. Vaak is de reden voor deze melding niet ogenblikkelijk duidelijk.
"Teveel variabelen"	Het beschikbare geheugen voor namen van variabelen is vol. Oplossing: gebruik kortere namen.
"Functieresultaat niet toegekend"	De functiedefinitie bevat niet de toekenning: functienaam := functie_resultaat

Foutmeldingen tijdens de uitvoer van een programma of model (Voorbeelden)

Wanneer een programma of model is geschreven, is er nog geen garantie dat er geen fouten zullen optreden tijdens de uitvoering. Treedt een fout op, dan wordt de uitvoering ge-

stopt en verschijnt een melding op het scherm. Mogelijk uitvoeringsfouten en bijbehorende oorzaken zijn:

Foutmelding	Reden
"Deling door nul"	Een variabele in de noemer van een breuk is gelijk aan 0.
"Getal buiten bereik"	Een getal wordt té groot of té klein.
"Waarde niet in domein"	Het argument van een standaardfunctie ontvangt een waarde die niet is toegestaan.

Tip: Nadat alle fouten eruit zijn gehaald, komt het moeilijkste van het schrijven van een programma of model: het zo te maken dat het precies doet wat je wilt dat het doet. Hierbij kan Coach je echter niet helpen. Kom je er niet uit, vraag dan je medeleerlingen of docent om hulp. Iemand anders heeft vaak een frisse kijk op jouw programma/model en ziet vaak sneller wat er 'fout' gaat (in jouw redenatie).
Heel vaak zal de ontwikkeling ook cyclisch verlopen: fout verbeteren, testen, weer iets verbeteren, weer testen, enz. Een computer is geduldig en er kan niet veel mee fout gaan, dus kun je gemakkelijk dingen uitproberen.

Voorbeeld van foutmeldingen tijdens uitvoering

Voorbeeld van een programma dat een foutmelding geeft tijdens uitvoering

Programma	Foutmeldingen
a := 100	
w := 100	
Repeteer 100	
a := a - 1	
y := 12/a	"Deling door nul" (in de laatste lus (als a=0))
y := **ln**(a)	"Waarde niet in domein" (in de laatste lus (als a=0))
w := w*w	"Waarde buiten bereik" (in de 5e lus)
TotHier	

Voorbeeld van een tekstmodel dat een foutmelding geeft tijdens uitvoering

Model	Opmerkingen
t := t + dt	De cursor knippert niet altijd precies op de plek waar de fout is.
Als t > 13 **Dan**	Let daarom goed op waar je de fout verbetert.
a := F/m	In het onjuiste model links, ontbreekt het laatste commando
Anders	('EindAls') van de 'Als'-opdracht.
a := F/(2 * m)	De CoachTaal interpreter kan deze fout pas ontdekken nadat de
v := v + a * dt	laatste opdracht is uitgevoerd. Dan verschijnt de foutmelding
x := x + v * dt	"EindAls verwacht" en knippert de cursor na het laatste opdracht
	in plaats van na de opdracht "a := F/(2 * m)".

Voorbeelden van syntaxfouten en foutmeldingen

Voorbeelden van syntaxfouten en de bijbehorende foutmeldingen

Opdracht(en)	Foutmelding
a : 1	Onverwacht karakter
a 1	"Toekenning" verwacht
a := / 1	"/" niet verwachtot expected
a := 1e=12	Getal is niet correct
Als a>0 **Dan** a:=1e12	"EindAls" verwacht
Herhaal a:=1e12	"TotDat" verwacht
Repeteer a := 1e12	"Toekenning" niet verwacht
Repeteer a 1e12	Getal niet verwacht
Functie a(x;y) **EindFunctie**	"EindFunctie" niet verwacht
Functie a(x;y) z=x*y **EindFunctie**	Functieresultaat niet toegekend (het zou juist zijn indien de tweede lijn a=x*y was)
Functie a(x;y) a=x∗y **EindFunctie** **Procedure** a(e) w = **sqrt**(e) **EindProcedure**	De naam is al in gebruik. Voer een andere naam in. [uitleg: functie en procedure hebben dezelfde naam "a"]
Functie a(x;y) a=x∗y **EindFunctie** a = 12	Type-conflict [uitleg: het is onmogelijk om een functienaam te vergelijken met een getal. Bovendien verwacht de functie-aanroep twee parameters].

Procedure a(x;y) z = x+y **EindProcedure**	
s = a(1;2)	Type-conflict [uitleg: zo'n opdracht kan gedaan worden met een functie-aanroep maar niet met een procedure-aanroep].
a(1;2;3)	")" verwacht [uitleg: er staan teveel parameters in de procedure-aanroep].
a(1)	";" verwacht [uitleg: er staan te weinig parameters in de procedure-aanroep].
Procedure b b = 12 **EindProcedure**	"=" niet verwacht

Vergelijking met andere programmeertalen

CoachTaal is een relatief eenvoudige programmeertaal, afgeleid van Pascal. Indien je enige ervaring hebt met talen zoals BASIC of Pascal, dan zul je veel van de expressies en opdrachten wel herkennen.
Programma's en tekstmodellen, die in CoachTaal zijn geschreven, kunnen alleen worden uitgevoerd binnen Coach.
Om een programma of tekstmodel in CoachTaal te kunnen schrijven, moet je iets weten over de volgorde waarin de woorden, de expressies en de opdrachten moeten staan (**syntax**) en over de betekenis ervan (**semantiek**). Hoe meer expressies je kent, hoe beter je programma's en modellen kunt schrijven, en hoe beter je stuurmodellen kunt besturen met de programma's. Als je CoachTaal gaat leren, zul je snel ontdekken dat je de computer een hoop rekenwerk kunt laten doen, en hoe je CoachTaal kunt gebruiken om processen en apparaten te besturen (bijv. robots).
Anders dan in meer ingewikkelde programmeertalen, is het niet mogelijk om arrays en karaktervariabelen (chars) te definiëren, en ook niet om waarden via het toetsenbord in te voeren (invoer en uitvoer) of bestandsbewerkingen te doen tijdens het draaien van een programma.
Aan de andere kant geven de expressies en standaardprocedures en -functies van CoachTaal je directe controle over de ingangen, uitgangen en tellers van de interfaces (voor zover ze op die interface beschikbaar zijn, natuurlijk) en over formules en modellen.

Standaard wiskundige functies

N.B.: De maximale waarde van x is 10^35 (tenzij anders vermeld).

Functie	Beschrijving	Opmerkingen
Sin(x)	Sinus van x	x in deg/rad*
Cos(x)	Cosinus van x	x in deg/rad*
Tan(x)	Tangens van x	x in deg/rad*
Arcsin(x)	Inverse sinusfunctie (radialen)	x in [-1;1]
Arccos(x)	Inverse cosinusfunctie (radialen)	x in [-1;1]
Arctan(x)	Inverse tangensfunctie (radialen)	
Exp(x)	e-macht van x (e^x)	
Ln(x)	Natuurlijke logarithme (grondtal e)	
Log(x)	Briggse logarithme (grondtal 10)	
Sqr(x)	Kwadraat van x	x in [-max, 81)
Sqrt(x)	Vierkantswortel van x	
Abs(x)	De absolute waarde van x	
Entier(x)	Afronding van x naar lagere gehele getal	
Round(x)	Afronding van x	
Fac(x)	Faculteit van x (round(x))!	x in [0;+33,5)
Max(x_1;x_2;.. x_i)	Selecteert de grootste van i parameters	
Min(x_1;x_2;..x_i)	Selecteert de kleinste van i parameters	
Rand	Genereert een willekeurig getal op het interval [0;1>	
Teken(x)	Genereert -1 voor x < 0; 0 voor x = 0; +1 voor x > 0	
EenheidStap(X,b)	Genereert 0 voor x < b; en 1 voor x = b	
Puls(x;b;l;h)	Genereert een puls van variabele x die start bij de beginwaarde b en lengte l en hoogte h heeft	
PulsHer-haald(x;b;l;i;h)	Genereert pulsen van variabele x met herhaalinter-val i. De eerste puls start bij beginwaarde b, heeft een lengte l en een hoogte h	

Zie ook:

- Wiskundige operaties;
- Speciale wiskundige functies;
- Een formule maken

Speciale wiskundige functies

Speciale wiskundige functies maken gebruik van meer dan één cel om een nieuwe waarde te berekenen.
Deze functies worden gebruikt in de formule-editor voor berekeningen in diagrammen en tabellen.

Functie	**Beschrijving**
Bezier(C)	Bezier berekent een polynoombenadering volgens de Bezier-methode, gebasseerd op de waarden in de kolom op de x-as en kolom **C**. In tegenstelling tot de optie **Analyse/Verwerking>Benadering**, is het niet mogelijk het aantal rijen te verhogen bij gebruik van de functie Bezier. (Zie, voor details: Benadering).
Delta(C)	De Delta-functie berekent het verschil tussen twee opeenvolgende waarden in bronkolom **C**. Cel(n) van de doelkolom wordt gevuld met de waarde van de expressie: Cel(n+1) - Cel(n) uit de bronkolom. De laatste cel in de doelkolom blijft onbepaald.
Deltafil(C)	Deltafil is een combinatie van Delta(C) en Filter(C;n). Deze functie is alléén zinvol indien de gegevens op gelijke afstand van elkaar liggen in de tijd. Voor **n** wordt de vaste waarde 2 genomen.
Afgeleide(C)	Afgeleide voert een numerieke differentiatie uit van de waarden in bronkolom **C**.
AfgeleideGlad(C;n)	De gladde afgeleide benadert de gegevens in bronkolom **C** met een Spline-functie met factor **n**. Daarna wordt een exacte numerieke differentiatie berekend van de spline-functie.
TweedeAfgeleide(C)	Numerieke tweede-orde differentiatie van de waarden in kolom **C**.
TweedeAfgeleideGlad(C;n)	De gladde tweede-orde afgeleide benadert eerst de gegevens in bronkolom **C** met een Spline-functie met factor **n**. Daarna wordt een exacte numerieke tweede-orde differentiatie berekend van de spline-functie.
Domein(b;e)	Domein vult alle cellen van de (lege) doelkolom met gelijkverdeelde waarden over het interval [b;e](**b**: beginwaarde; **e**: eindwaarde) volgens de formule: cel(index) = **b** + [(**e** - **b**) * (index - 1)]/(**n** – 1), waarbij **n** het aantal rijen is.

Filter(C;n)	De functie Filter filtert de waarden in bronkolom C met filterinterval **n**. De waarde in elke cel wordt vervangen door de gemiddelde waarde van de cel zelf, de **n** voorgangers en de **n** opvolgers (d.w.z. het gemiddelde over 2**n**+1 punten). Aan de randen wordt een spiegeling uitgevoerd om voldoende waarden te verkrijgen.
Integraal(C;r)	Integraal voert een numerieke integratie uit van de waarden in bronkolom **C**. **r** is de integratieconstante (de beginwaarde).
Spline(C;n)	Spline berekent een polynoombenadering volgens de Spline-methode, gebasseerd op waarden in de kolom langs de x-as en kolom **C**. **n** staat voor de waarde van de parameter **Interval**. In tegenstelling tot de optie **Analyse/Verwerking>Benadering**, is het niet mogelijk het aantal rijen te verhogen bij gebruik van de functie Spline. (Zie, voor details: Benadering).
Som(C)	Som sommeert de waarden van cellen. Cel(n) bevat de som van de waarden in alle cellen van bronkolom **C**, die een index kleiner of gelijk aan **n** hebben.

N.B.: Vervang **C** in deze operaties door grootheden van de kolommen of de kolomnamen zelf (**C1** .. **C8**).

Zie ook:

- Rekenkundige operatoren;
- Standaard Wiskundige functies;
- Overzicht van verwerkings- en analysemogelijkheden van Coach

CoachTaal: Standaardprocedures en -functies

De volgende standaardprocedures en - functies kunnen in programma's gebruikt worden.

Standaardprocedures

- Geluid
- SlaOp en WisData
- Stop
- Stopwatch
- Tel
- Wacht
- ZetAan en ZetUit
- ZetAanAbsoluut en ZetUitAbsoluut
- ZetNiveau

Standaardfuncties

- Bit
- LoopTijd
- Niveau
- Teller en ResetTellers
- Tussentijd
- WasBitHoog en WasBitLaag

CoachTaal: Geluid

De standaardprocedure **Geluid(f; t)** produceert een toon met een frequentie **f** Hertz gedurende **t** seconden.

Zie ook:

- Overzicht van standaardprocedures en -functies

CoachTaal: SlaOp

N.B.: Gebruik van **SlaOp** in een programma is alléén nodig in activiteiten van het type 'Sturen met Programma'. In andere activiteit-typen worden programma-variabelen automatisch opgeslagen, maar zijn ze niet beschikbaar om uit te zetten in diagrammen en tabellen.

De standaardprocedure **SlaOp** is nodig om diagrammen en tabellen goed te vullen met programmavariabelen. Iedere keer dat **SlaOp** gepasseerd wordt, worden de waarden van de variabelen die in diagrammen of tabellen voorkomen één keer weggeschreven. Indien **SlaOp** *niet wordt gebruikt* of *op de verkeerde plek wordt gebruikt* (zie voorbeeld) kan dat leiden tot verkeerde of lege diagrammen of tabellen.

SlaOp is het complement van de standaardprocedure **WisData**.

Voorbeeld:
Meet gedurende een half uur, en schrijf alleen de waarden als de temperatuur hoger is dan 20° C:

Correcte oplossing:
Zolang de temperatuur onder de 20 ºC is worden geen waarden weggeschreven in het diagram (SlaOp binnen de Als-opdracht die de temperatuur test).

```
Herhaal
   Als Temperatuur > 20 Dan
     Wacht(1)
     SlaOp
   EindAls
TotDat LoopTijd >1800
```

Foutieve oplossing:
Waarden worden ook geschreven in het diagram als de temperatuur lager is dan 20 ºC. (SlaOp buiten de Als-opdracht die de temperatuur test).

```
Herhaal
   Als Temperatuur > 20 Dan
     Wacht(1)
   EindAls
    SlaOp
TotDat LoopTijd >1800
```

Zie ook:

- Coach-Language lusopdrachten
- Overzicht van standaardprocedures en -functies

CoachTaal: WisData

De standaardprocedure **WisData** wist alle gegevens (in alle diagrammen en tabellen) in de huidige activiteit.

WisData is het complement van de standaarprocedure **SlaOp**.

Zie ook:

- Overzicht van standaardprocedures en -functies

CoachTaal: Stop

De standaardprocedure **Stop** stopt de uitvoering van het programma op de positie van deze procedure.

Zie ook:

- Overzicht van standaardprocedures en -functies

CoachTaal: Stopwatch

De standaardprocedure **StopWatch(B)** start een tijdmeting (in secondes) indien de Boolese variabele **B** de waarde **Aan** (Waar) heeft. De verstreken tijd na de start van de tijdmeting wordt gemeten door de standaardfunctie **TussenTijd**.
TussenTijd retourneert de waarde 0 ogenblikkelijk na de uitvoering van **StopWatch(Aan)**. De functie TussenTijd blijft 0 na de uitvoering van **Stopwatch(Uit).**

Voorbeeld:

```
Stopwatch(Aan)
Herhaal
  …
  …
TotDat TussenTijd > 10
```

Zie ook:

- Overzicht van standaardprocedures en -functies

CoachTaal: Tel

De standaardprocedure **Tel(n; p)** telt pulsen via Teller-ingang **n**. Het programma wacht totdat het aantal pulsen **p** is bereikt. De maximale waarde van **p** is 65535.

- Door een willekeurige toets (behalve **<Esc>**) in te drukken, forceer je het programma om door te gaan met de volgende opdracht, ook al is het aantal pulsen nog niet bereikt (hiermee voorkom je dat het programma 'blijft hangen').
- **<Esc>** is gereserveerd om de programma-uitvoer te onderbreken.

Het gebruik van de standaarprocedure **Tel** is analoog aan het gebruik van de Standaarprocedure **Wacht**. **Tel** wacht totdat het opgegeven aantal pulsen is geteld, **Wacht** wacht de opgegeven tijd in seconden.

Zie ook:

- Overzicht van standaardprocedures en -functies

CoachTaal: Wacht

De standaardprocedure **Wacht(t)** 'telt' tijd. Het programma wacht totdat het aantal opgegeven seconden **t** verstreken is.

- Door indrukken van een willekeurige toets (behalve **<Esc>**) wordt het programma geforceerd om door te gaan met de volgende opdracht, zelfs als de opgegeven wachttijd nog niet verstreken is.
- **<Esc>** is gereserveerd om de programma-uitvoering af te breken.

Het gebruik van de standaardprocedure **Wacht** is analoog aan het gebruik van de standaardprocedure **Tel**. **Wacht** pauzeert een tijdsinterval, terwijl **Tel** het programma pauzeert totdat een opgegeven aantal pulsen gedetecteerd is.

Zie ook:

- Overzicht van standaardprocedures en -functies

CoachTaal: ZetAan

De standaardprocedure **ZetAan(i, j, ..)** zet de digitale uitgangen **i, j, ..** '**Hoog**' en laat de andere uitgangen ongemoeid.
De lijst met parameters kan net zo lang zijn als het aantal uitgangen van de interface.

ZetAan is het complement van de standaardprocedure **ZetUit** en verwant aan**ZetAanAbsoluut** en **ZetUitAbsoluut**.

Voorbeelden:

- Voor een interface met vier digitale uitgangen is **ZetAan(1;2;3;4)** toegestaan.
- Worden geen parameters genoemd, dan zet **ZetAan** alle digitale uitgangen 'Hoog'. Dus met vier uitgangen doet het commando **ZetAan** hetzelfde als **ZetAan(1;2;3;4)**.

Zie ook:

- Overzicht van standaardprocedures en -functies

CoachTaal: ZetUit

De standaardprocedure **ZetUit(i, j, ..)** zet de digitale uitgangen **i**, **j**, .. '**Laag**' en laat de toestand van de andere uitgangen ongemoeid.
De lijst parameters kan net zo lang zijn als het aantal uitgangen van de interface.

ZetUit is het complement van de standaardprocedure **ZetAan** en verwant met **ZetUitAbsoluut** en **ZetAanAbsoluut**.

Voorbeelden:

- Voor een interface met vier digitale uitgangen is **ZetUit(1;2;3;4)** toegestaan.
- Worden geen parameters genoemd, dan zet **ZetUit** alle digitale uitgangen 'Laag'. Dus met vier uitgangen doet het commando **ZetUit** hetzelfde als **ZetUit(1;2;3;4)**.

Zie ook:

- Overzicht van standaardprocedures en -functies

CoachTaal: ZetAanAbsoluut

De standaardprocedure **ZetAanAbsoluut(i, j, ..)** zet de digitale uitgangen **i, j, ..** '**Hoog**' en alle andere '**Laag**'.
De lijst met parameters kan net zo lang zijn als het aantal digitale uitgangen van de interface.

ZetAanAbsoluut is het complement van de standaardprocedure **ZetUitAbsoluut** en verwant met **ZetUit** en **ZetAan**.

Voorbeelden:

- Voor een interface met vier digitale uitgangen is **ZetAanAbsoluut(1;2;3;4)** toegestaan.
- Worden geen parameters genoemd, dan zet **ZetAanAbsoluut** alle digitale uitgangen 'Hoog'. Dus met vier uitgangen doet het commando **ZetAanAbsoluut** hetzelfde als **ZetAanAbsoluut(1;2;3;4)**.

Zie ook:

- Overzicht van standaardprocedures en -functies

CoachTaal: ZetUitAbsoluut

De standaardprocedure **ZetUitAbsoluut(i, j, ..)** zet de digitale uitgangen **i**, **j**, .. '**Laag**' en alle andere uitgangen op '**Hoog**'.
De lijst parameters kan net zo lang zijn als het aantal uitgangen van de interface.

ZetUitAbsoluut is het complement van de standaardprocedure **ZetAanAbsoluut** en verwant met **ZetUit** en **ZetAan**.

Voorbeelden:

- Voor een interface met vier digitale uitgangen is **ZetUitAbsoluut(1;2;3;4)** toegestaan.
- Worden geen parameters genoemd, dan zet **ZetUitAbsoluut** alle digitale uitgangen 'Laag'. Dus met vier uitgangen doet het commando **ZetUitAbsoluut** hetzelfde als **ZetUitAbsoluut(1;2;3;4)**.

Zie ook:

- Overzicht van standaardprocedures en -functies

CoachTaal: ZetNiveau

N.B.: *Deze procedure werkt alleen met CMA CoachLab II/II+, CMA €Sense en de LEGO DACTA interfaces.*

De standaardprocedure **ZetNiveau(i, p)** zet het vermogensniveau van digitale uitgang **i** op de waarde **p** (**i** en **p** zijn beide gehele getallen).

- Indien gebruikt bij CoachLab II of II+ moet **i** liggen tussen 1 en 4, waarbij 1 overeenkomt met uitgang A, 2 met uitgang B, enz., en **p** moet liggen tussen 1 en 16, waarbij 1 overeenkomt met het laagste vermogensniveau en 16 met het hoogste.
- Indien gebruikt bij €Sense moet **i** liggen tussen 1 en 2, waarbij 1 overeenkomt met de LED en 2 met de zoemer, en **p** moet liggen tussen 1 en 16, waarbij 1 overeenkomt met het laagste vermogensniveau en 16 met het hoogste.
- Indien gebruikt bij de LEGO DACTA Interface B moet **i** liggen tussen 1 en 8, waarbij 1 overeenkomt met uitgang A, 2 met uitgang B, ..., 8 met uitgang H, enz. en **p** moet liggen tussen 1 en 8, waarbij 1 overeenkomt met het laagste vermogensniveau en 8 met het hoogste.

- Indien gebruikt bij de LEGO DACTA RCX moet **i** liggen tussen 1 en 3 waarbij 1 correspondeert met uitgang A, 2 met uitgang B en 3 met uitgang C, en **p** moet liggen tussen 1 en 8, waarbij 1 correspondeert met het laagste vermogensniveau en 8 met het hoogtste.

N.B.

- Het hoogste vermogensniveau dat met ZetNiveau verkregen kan worden wordt autoamtsch beperkt door het ingestelde bereik van de Vermogensschuif in de Actuator-eigenschappen.
- Hogere waarden voor **p** dan de bovenstaande maximumwaarde leiden niet tot een foutmelding en ze stellen de actuatore in op het maximale vermogensniveau..

Zie ook:

- Overzicht van standaardprocedures en -functies

CoachTaal: Bit

De standaardfunctie **Bit(n)** geeft de waarde **Aan** (Waar) indien de digitale ingang **n** "**Hoog**" is, en **Uit** (Onwaar) indien de ingang "**Laag**" is.
N.B.: De numerieke waarden **Aan** en **Uit** zijn 255 en 0.

Voorbeelden:

- In een toekenning: **Toestand = Bit(1)**
- In een conditionele expressie: **Als Bit(2) = Aan Dan ...** of korter: **Als Bit(2) Dan ...**

Zie ook:

- Overzicht van standaardprocedures en -functies

CoachTaal: LoopTijd

De standaardfunctie **LoopTijd** geeft de tijd (in seconden) terug, die verstreken is sinds de start van het programma.

Voorbeelden:

- Herhaal … TotDat LoopTijd > *100*
- Zolang LoopTijd < *100* Doe … EindDoe

Zie ook:

- CoachTaal lusopdrachten
- Overzicht van standaardprocedures en -functies

CoachTaal: Niveau

De standaardfunctie**Niveau(n)** geeft de waarde terug van de sensor die aangesloten is op analoge ingang **n**.
Indien de sensor geijkt is, wordt de waarde uitgedrukt in de geijkte eenheid.

Voorbeelden:

- In een toekenning: **Helderheid = Niveau(1)**
- In een voorwaardelijke expressie: **Als Niveau(2) < 0,5 Dan ...**

Zie ook:

- Overzicht van of standaardprocedures en -functies

CoachTaal: Teller en ResetTellers

De standaardfunctie **Teller(n)** geeft het aantal pulsen terug dat gedetecteerd is op tellering-gang **n**.
Alle tellers worden op 0 gezet met de standaardprocedure **ResetTellers**. De tellers kunnen niet individueel op 0 gezet worde, dus **ResetTellers** heeft geen parameters.

Voorbeelden:
- In een toekenning: **Toestand=Teller(1)**
- In een voorwaardelijke expressie: **Als Teller(2) < 100 Dan…**

Zie ook:
- Overzicht van standaardprocedures en -functies

CoachTaal: Tussentijd

De standaardfunctie **Tussentijd** geeft de tijd (in seconden) terug die verstreken is na uitvoering van de Standaardprocedure **Stopwatch(Aan)**.

Tussentijd geeft 0 terug na de uitvoering van **Stopwatch(Uit)**.

Voorbeeld:

```
Stopwatch(Aan)
Herhaal
   …
   …
TotDat Tussentijd > 10
```

Zie ook:
- Overzicht van standaardprocedures en -functies

CoachTaal: WasBitHoog

De standaardfunctie **WasBitHoog(n)** geeft de waarde **Aan** (Waar) terug, als de digitale ingang **n** "**Hoog**" is geweest na de laatste aanroep van **WasBitHoog(n)**, en anders geeft de functie **Uit** (Onwaar).
N.B.: De numerieke waarden van **Aan** en **Uit** zijn 255 en 0.

WasBitHoog is het complement van de standaardfunctie **WasBitLaag**.

Voorbeelden:
- In een toekenning: **Toestand = WasBitHoog(1)**
- In een voorwaardelijke expressie: **Als WasBitHoog(2) = Aan Dan …**
  of korter: **Als WasBitHoog(2) Dan …**

Zie ook:
- Overzicht van standaardprocedures en -functies

CoachTaal: WasBitLaag

De standaardfunctie **WasBitLaag(n)** geeft de waarde **Aan** (Waar) terug, als de digitale ingang **n** "**Hoog**" is geweest na de laatste aanroep van **WasBitHoog(n)**, en anders geeft de functie **Uit** (Onwaar).

N.B.: De numerieke waarden van **Aan** en **Uit** zijn 255 en 0.

WasBitLaag is het complement van de standaardfunctie **WasBitHoog**.

Voorbeelden:

- In een toekenning: **Toestand = WasBitLaag(1)**
- In een conditionele expressie: **Als WasBitLaag(2) = Aan Dan …**of korter: **Als WasBitLaag(2) Dan …**

Zie ook:

- Overzicht van standaardprocedures en -functies