Masterclass programmeren op de GR TI-84 (les 2)

*
Kevin van As

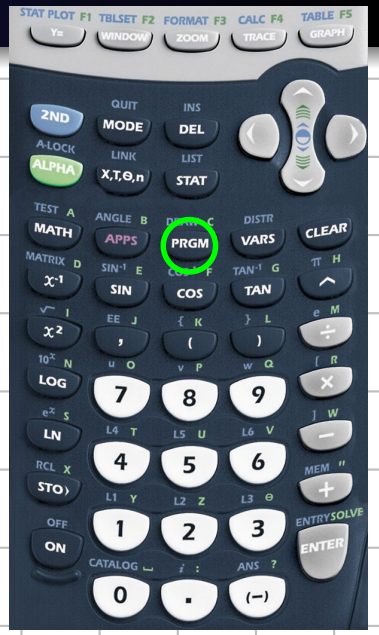
December 13, 2015

Recap!

We hebben gekeken naar:

- Waar kun je prgms vinden? *
- Wat zijn variabelen?
- Wat zijn datatypen?
  Met namen: getallen en strings.
- Basic I/O: `Prompt` en `Disp`.

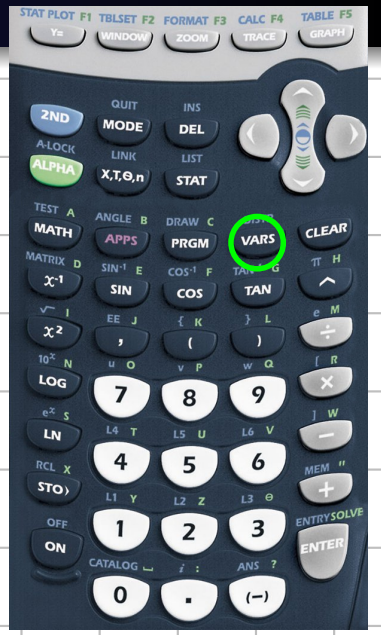
Recap!



We hebben gekeken naar:

- Waar kun je prgms vinden? *
- Wat zijn variabelen?
- Wat zijn datatypen?
  Met namen: getallen en strings.
- Basic I/O: `Prompt` en `Disp`.

Recap!



We hebben gekeken naar:

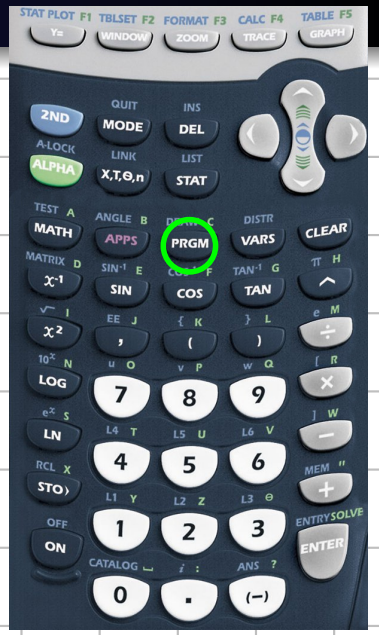
- Waar kun je prgms vinden? *
- Wat zijn variabelen?
- Wat zijn datatypen?
  Met namen: getallen en strings.
- Basic I/O: `Prompt` en `Disp`.

Recap!

We hebben gekeken naar:

- Waar kun je prgms vinden? *
- Wat zijn variabelen?
- Wat zijn datatypen?
  Met namen: getallen en strings.
- Basic I/O: `Prompt` en `Disp`.

Recap!



We hebben gekeken naar:

- Waar kun je prgms vinden? *
- Wat zijn variabelen?
- Wat zijn datatypen?
  Met namen: getallen en strings.
- Basic I/O: `Prompt` en `Disp`.

Vooruitzicht!

Vandaag zullen we kijken naar:

- Boolean datatype
- Control: "`If` statements"
- Pseudo Code
- Algoritmes

Vooruitzicht!

Vandaag zullen we kijken naar:

- Boolean datatype
- Control: "`If` statements"
- Pseudo Code
- Algoritmes

Vooruitzicht!

Vandaag zullen we kijken naar:

- Boolean datatype
- Control: "`If` statements"
- Pseudo Code
- Algoritmes

Vooruitzicht!

Vandaag zullen we kijken naar:

- Boolean datatype
- Control: "`If` statements"
- Pseudo Code
- Algoritmes

Vooruitzicht!

Vandaag zullen we kijken naar:

- Boolean datatype
- Control: "`If` statements"
- Pseudo Code
- Algoritmes

Outline

Outline

Wat is een Boolean?

- Een Boolean datatype beschrijft een waarheid, of of er aan een conditie voldaan wordt: “Juist” of “Onjuist”
  - In het Engels, “True” of “False”
- Een Boolean kan dus slechts twee waarden aannemen: Hij is “Binair”, en is dus ook te beschouwen als een Binair getal: 1 (juist) of 0 (onjuist).
- De Boolean wordt ook gebruikt in de Logica om op een formele manier te redeneren over de juistheid van argumenten.

Wat is een Boolean?

- Een Boolean datatype beschrijft een waarheid, of of er aan een conditie voldaan wordt: “Juist” of “Onjuist”
  - In het Engels, “True” of “False”
- Een Boolean kan dus slechts twee waarden aannemen: Hij is “Binair”, en is dus ook te beschouwen als een Binair getal: 1 (juist) of 0 (onjuist).
- De Boolean wordt ook gebruikt in de Logica om op een formele manier te redeneren over de juistheid van argumenten.

Wat is een Boolean?

- Een Boolean datatype beschrijft een waarheid, of of er aan een conditie voldaan wordt: “Juist” of “Onjuist”
  - In het Engels, “True” of “False”
- Een Boolean kan dus slechts twee waarden aannemen: Hij is “Binair”, en is dus ook te beschouwen als een Binair getal: 1 (juist) of 0 (onjuist).
- De Boolean wordt ook gebruikt in de Logica om op een formele manier te redeneren over de juistheid van argumenten.

Wat is een Boolean?

- Een Boolean datatype beschrijft een waarheid, of of er aan een conditie voldaan wordt: “Juist” of “Onjuist”
  - In het Engels, “True” of “False”
- Een Boolean kan dus slechts twee waarden aannemen: Hij is “Binair”, en is dus ook te beschouwen als een Binair getal: 1 (juist) of 0 (onjuist).
- De Boolean wordt ook gebruikt in de Logica om op een formele manier te redeneren over de juistheid van argumenten.

Wat is het nut van een Boolean?

Booleans kunnen gebruikt worden om je programma te conditioneren:
Onder bepaalde voorwaarden (condities) heeft je programma een andere output.

*

Neem bijvoorbeeld de ABC formule:
Afhankelijk van het teken van de discriminant zijn er nul, één of twee oplossingen.
$D = b^2 - 4ac$

Wat is hier de conditie?

Wat is het nut van een Boolean?

Booleans kunnen gebruikt worden om je programma te conditioneren:
Onder bepaalde voorwaarden (condities) heeft je programma een andere output.

*

Neem bijvoorbeeld de ABC formule:
Afhankelijk van het teken van de discriminant zijn er nul, één of twee oplossingen.
$D = b^2 - 4ac$

Wat is hier de conditie?

Wat is het nut van een Boolean?

Booleans kunnen gebruikt worden om je programma te conditioneren:
Onder bepaalde voorwaarden (condities) heeft je programma een andere output.

*

Neem bijvoorbeeld de ABC formule:
Afhankelijk van het teken van de discriminant zijn er nul, één of twee oplossingen.
$D = b^2 - 4ac$

Wat is hier de conditie?

Wat is het nut van een Boolean?

Booleans kunnen gebruikt worden om je programma te conditioneren:
Onder bepaalde voorwaarden (condities) heeft je programma een andere output.

*

Neem bijvoorbeeld de ABC formule:
Afhankelijk van het teken van de discriminant zijn er nul, één of twee oplossingen.
$D = b^2 - 4ac$

Wat is hier de conditie?

Wat is het nut van een Boolean?

Booleans kunnen gebruikt worden om je programma te conditioneren:
Onder bepaalde voorwaarden (condities) heeft je programma een andere output.

*

Neem bijvoorbeeld de ABC formule:
Afhankelijk van het teken van de discriminant zijn er nul, één of twee oplossingen.
$D = b^2 - 4ac$

Wat is hier de conditie?

Voorbeelden van statements

Condities zijn statements zoals:

`C=2` `π≠3` `A>6` `A<B` `X≥42` `0≤A`

Vraag: Wanneer zijn deze 6 condities "true" (juist)?

*

Voorbeelden van statements

Condities zijn statements zoals:

`C=2` `π≠3` `A>6` `A<B` `X≥42` `0≤A`

Vraag: Wanneer zijn deze 6 condities “true” (juist)?

*

Voorbeelden van statements

Condities zijn statements zoals:

C=2 | π≠3 | A>6 | A<B | X≥42 | 0≤A

Vraag: Wanneer zijn deze 6 condities "true" (juist)?

De 6 mogelijke boolean operatoren zijn:

- = Gelijk aan.
- ≠ Ongelijk aan.
- > Groter dan.
- < Kleiner dan.
- ≥ Groter dan of gelijk aan.
- ≤ Kleiner dan of gelijk aan.

Voorbeelden

Welke conditie hebben we nodig om te eisen dat iemand minstens 12 jaar oud is? (A is de leeftijd van de gebruiker.)

```
PROGRAM:OUDERDAN
:Disp "HOE OUD BEN JE?"
:Prompt A
```

*

Voorbeelden

Welke conditie hebben we nodig om te eisen dat iemand minstens 12 jaar oud is? (A is de leeftijd van de gebruiker.)

```
A≥12
```

```
PROGRAM:OUDERDAN
:Disp "HOE OUD BEN JE?"
:Prompt A
```

Voorbeelden

Welke conditie hebben we nodig om te eisen dat iemand minstens 12 jaar oud is? (A is de leeftijd van de gebruiker.)

```
A≥12
```

```
PROGRAM:OUDERDAN
:Disp "HOE OUD BEN JE?"
:Prompt A
```

Stel dat R een getal in een programma is, die slechts gelijk is aan 0 of 1. Het beschrijft of het wel (R=1) of niet (R=0) regent.
Welke conditie kunnen we gebruiken om te kijken of het regent?

Voorbeelden

Welke conditie hebben we nodig om te eisen dat iemand minstens 12 jaar oud is? (`A` is de leeftijd van de gebruiker.)

`A≥12`

```
PROGRAM:OUDERDAN
:Disp "HOE OUD BEN JE?"
:Prompt A
```

*

Stel dat `R` een getal in een programma is, die slechts gelijk is aan `0` of `1`. Het beschrijft of het wel (`R=1`) of niet (`R=0`) regent.
Welke conditie kunnen we gebruiken om te kijken of het regent?
`R≠0` "Het regent niet niet"; Of `R=1` "Het regent wel".

Outline

Logica

Booleans kunnen gebruikt worden om op een formele manier te redeneren: Logica.

De tabel laat zien wat de operatoren 'and', 'or', 'xor' (exclusive or: A óf B, maar niet allebei) en 'not' doen met de booleans A en B.

A	B	A and B	A or B	A xor B	not(A)
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

Table: Logische operatoren

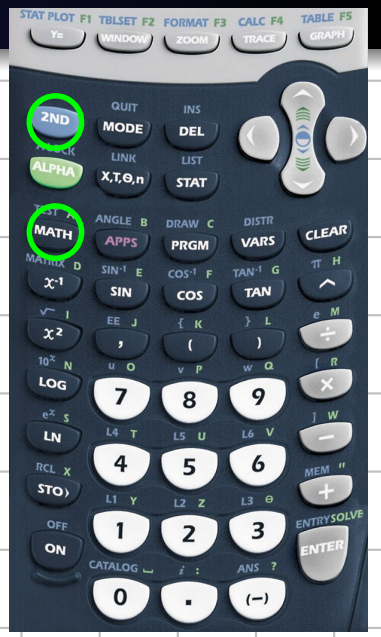
Logica

Booleans kunnen gebruikt worden om op een formele manier te redeneren: Logica.
De tabel laat zien wat de operatoren 'and', 'or', 'xor' (exclusive or: A óf B, maar niet allebei) en 'not' doen met de booleans A en B.

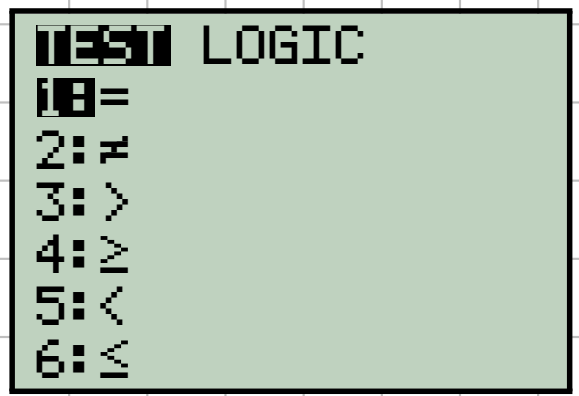
A	B	A and B	A or B	A xor B	not(A)
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

Table: Logische operatoren

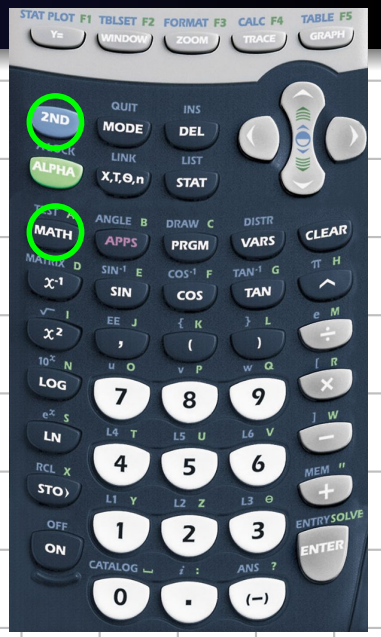
Logica op de GR



- Druk op [TEST]=[2nd][MATH].
- Blader naar rechts voor het [LOGIC]*-menu
- Merk op dat op je GR 0 'onjuist/false' is, terwijl ieder ander getal (bijv. π) 'juist/true' is.

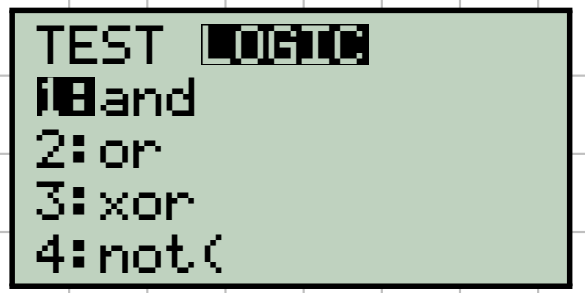


Logica op de GR

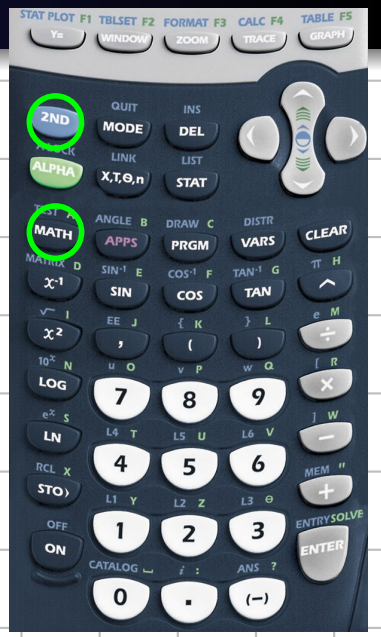


- Druk op [TEST]=[2nd][MATH].
- Blader naar rechts voor het [LOGIC]-menu

- Merk op dat op je GR 0 'onjuist/false' is, terwijl ieder ander getal (bijv. π) 'juist/true' is.

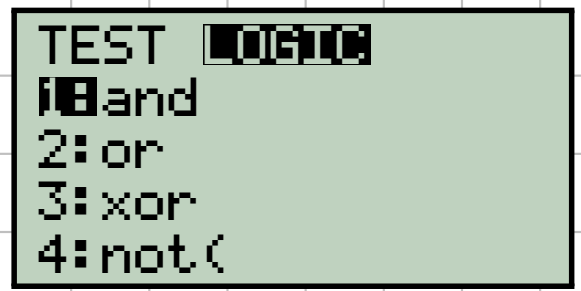


Logica op de GR



- Druk op [TEST]=[2nd][MATH].
- Blader naar rechts voor het [LOGIC]-menu

- Merk op dat op je GR 0 'onjuist/false' is, terwijl ieder ander getal (bijv. π) 'juist/true' is.



Outline

Hoe kunnen we condities gebruiken in een programma?

Allereerst...Hoe gebruiken we condities in ons taalgebruik?

*

Hoe kunnen we condities gebruiken in een programma?

Allereerst...Hoe gebruiken we condities in ons taalgebruik?

"Als het regent, dan blijf ik binnen."

Hoe kunnen we condities gebruiken in een programma?

Allereerst...Hoe gebruiken we condities in ons taalgebruik?

"Als het regent, dan blijf ik binnen."
"Als ik later groot ben, dan wordt ik een brandweerman."

Hoe kunnen we condities gebruiken in een programma?

Allereerst...Hoe gebruiken we condities in ons taalgebruik?

"Als het regent, dan blijf ik binnen."
"Als ik later groot ben, dan wordt ik een brandweerman."
"Als ik die toets haal, dan ga ik over, of anders blijf ik zitten."

Hoe kunnen we condities gebruiken in een programma?

Allereerst...Hoe gebruiken we condities in ons taalgebruik?

"Als het regent, dan blijf ik binnen."
"Als ik later groot ben, dan wordt ik een brandweerman."
"Als ik die toets haal, dan ga ik over, of anders blijf ik zitten."

In het algemeen:
"**Als** «CONDITIE», **dan** «ACTIE», **of anders** «ACTIE»"

Hoe kunnen we condities gebruiken in een programma?

"**Als** «CONDITIE», **dan** «ACTIE», **of anders** «ACTIE». **Klaar**."

*

Hoe kunnen we condities gebruiken in een programma?

"**Als** «CONDITIE», **dan** «ACTIE», **of anders** «ACTIE». **Klaar**."

"**If** «CONDITION», **then** «ACTION», **else** «ACTION». **End**."

*

Hoe kunnen we condities gebruiken in een programma?

"**Als** «CONDITIE», **dan** «ACTIE», **of anders** «ACTIE». **Klaar**."

"**If** «CONDITION», **then** «ACTION», **else** «ACTION». **End**."

```
PROGRAM:FIRST IF
:If SOME CONDITION
:Then
:DO SOMETHING
:Else
:DO OTHER THING
:End
```

*

Hoe kunnen we condities gebruiken in een programma?

"**Als** «CONDITIE», **dan** «ACTIE», **of anders** «ACTIE». **Klaar**."

"**If** «CONDITION», **then** «ACTION», **else** «ACTION». **End**."

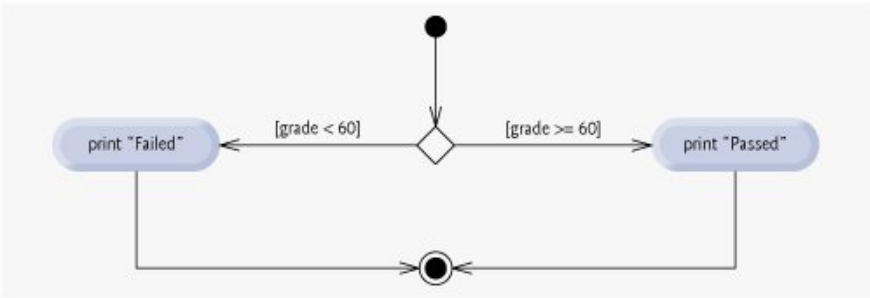
```
PROGRAM:FIRST IF
:If SOME CONDITION
:Then
:DO SOMETHING
:Else
:DO OTHER THING
:End
```

```
PROGRAM:FIRST IF
:If G≥60
:Then
:Disp "PASSED."
:Else
:Disp "FAILED."
:End
```

Hoe kunnen we condities gebruiken in een programma?

"**Als** «CONDITIE», **dan** «ACTIE», **of anders** «ACTIE». **Klaar**."

"**If** «CONDITION», **then** «ACTION», **else** «ACTION». **End**."



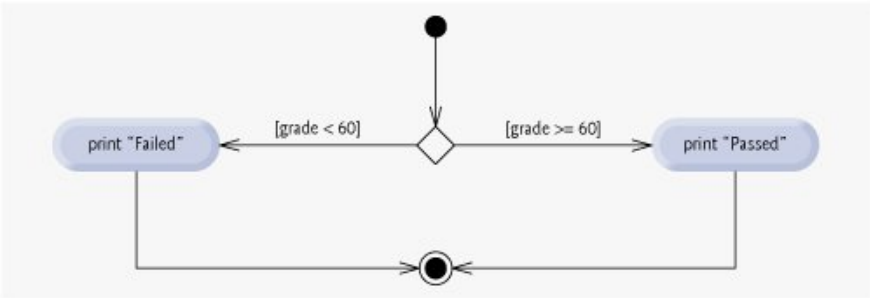

```
PROGRAM:FIRST IF
:If G≥60
:Then
:Disp "PASSED."
:Else
:Disp "FAILED."
:End
```

Er ontstaan nu twee paden in het programma:

Hoe kunnen we condities gebruiken in een programma?

"**Als** «CONDITIE», **dan** «ACTIE», **of anders** «ACTIE». **Klaar**."

"**If** «CONDITION», **then** «ACTION», **else** «ACTION». **End**."



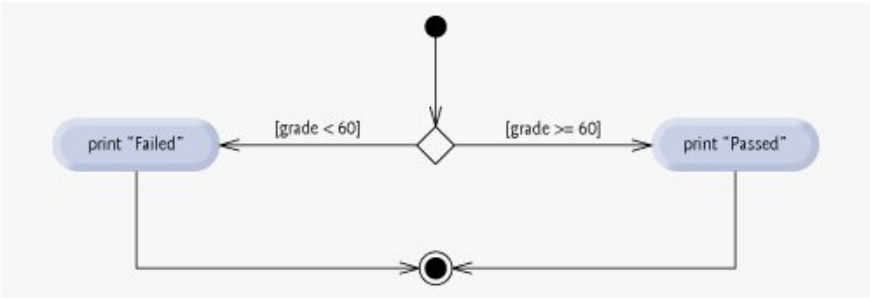

```
PROGRAM:FIRST IF
:If G≥60
:Then
:Disp "PASSED."
:Else
:Disp "FAILED."
:End
```

Er ontstaan nu twee paden in het programma:
Slechts de statement(s) tussen `Then` en `Else` wordt uitgevoerd, of de statement(s) tussen `Else` en `End`.

Hoe kunnen we condities gebruiken in een programma?

"**Als** «CONDITIE», **dan** «ACTIE», **of anders** «ACTIE». **Klaar**."

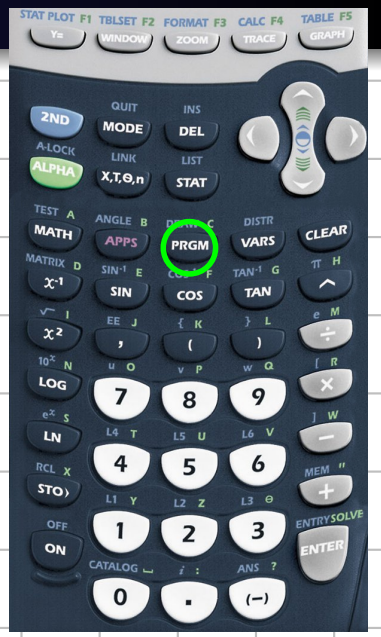
"**If** «CONDITION», **then** «ACTION», **else** «ACTION». **End**."




```
PROGRAM:FIRST IF
:If G≥60
:Then
:Disp "PASSED."
:Else
:Disp "FAILED."
:End
```

Er ontstaan nu twee paden in het programma:
Slechts de statement(s) tussen `Then` en `Else` wordt uitgevoerd, of de statement(s) tussen `Else` en `End`. Daarna gaat het programma verder met statements die onder `End` staan.

Waar staat het op de GR?



- Maak een nieuw programma: `PROGRAM:FIRSTIF`.
- Druk op [PRGM] en kies `If`.
- Type een letter.
- Druk op [TEST] — [2nd] [MATH] en kies `=`.
- Type `5` gevolgd door [ENTER].
- Druk op [PRGM] en kies `Then`, daarna [ENTER].
- Druk op [PRGM] en kies `Else`, daarna [ENTER].
- Druk op [PRGM] en kies `End`, daarna [ENTER].
- Dit is de structuur van het `if`-statement.

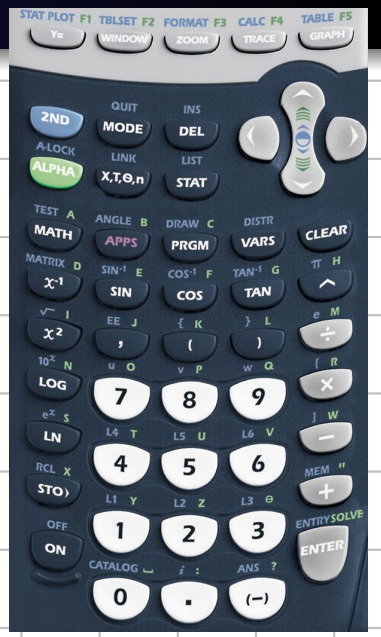
```
PROGRAM:FIRSTIF
:■
```

Waar staat het op de GR?

- Maak een nieuw programma: `PROGRAM:FIRSTIF`.
- Druk op PRGM en kies **If**.
- Type een letter.
- Druk op [TEST] — 2nd MATH en kies =.
- Type 5 gevolgd door ENTER.
- Druk op PRGM en kies **Then**, daarna ENTER.
- Druk op PRGM en kies **Else**, daarna ENTER.
- Druk op PRGM en kies **End**, daarna ENTER.
- Dit is de structuur van het **if**-statement.

```
CTL I/O EXEC
1:If
2:Then
3:Else
4:For(
5:While
6:Repeat(
7↓End
```

Waar staat het op de GR?

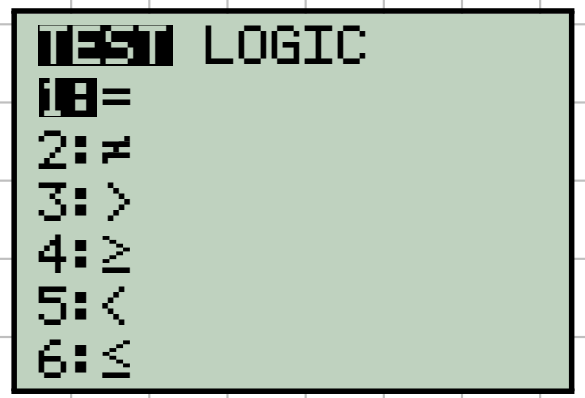


- Maak een nieuw programma: `PROGRAM:FIRSTIF`.
- Druk op [PRGM] en kies **If**.
- Type een letter.
- Druk op [TEST] — [2nd][MATH] en kies =.
- Type 0 gevolgd door [ENTER].
- Druk op [PRGM] en kies **Then**, daarna [ENTER].
- Druk op [PRGM] en kies **Else**, daarna [ENTER].
- Druk op [PRGM] en kies **End**, daarna [ENTER].
- Dit is de structuur van het **if**-statement.

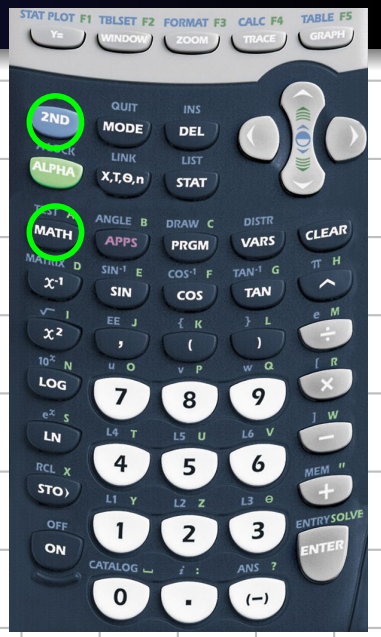
```
PROGRAM:FIRSTIF
:If A
```

Waar staat het op de GR?

- Maak een nieuw programma: `PROGRAM:FIRSTIF`.
- Druk op `PRGM` en kies `If`.
- Type een letter.
- Druk op `[TEST]` = `2nd` `MATH` en kies `=`.
- Type `0` gevolgd door `ENTER`.
- Druk op `PRGM` en kies `Then`, daarna `ENTER`.
- Druk op `PRGM` en kies `Else`, daarna `ENTER`.
- Druk op `PRGM` en kies `End`, daarna `ENTER`.
- Dit is de structuur van het `if`-statement.



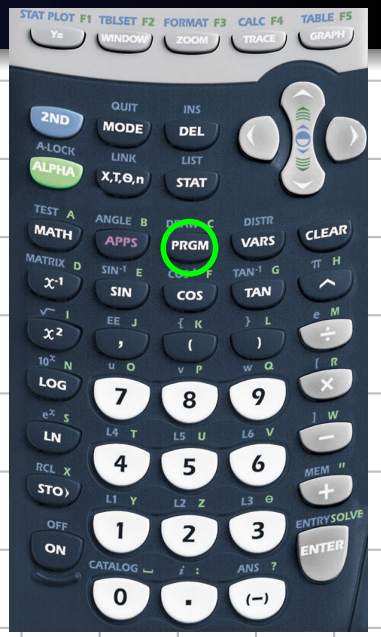
Waar staat het op de GR?



- Maak een nieuw programma: `PROGRAM:FIRSTIF`.
- Druk op PRGM en kies **If**.
- Type een letter.
- Druk op [TEST]=2nd MATH en kies **=**.
- Type **0** gevolgd door ENTER.
- Druk op PRGM en kies **Then**, daarna ENTER.
- Druk op PRGM en kies **Else**, daarna ENTER.
- Druk op PRGM en kies **End**, daarna ENTER.
- Dit is de structuur van het **if**-statement.

```
PROGRAM:FIRSTIF
:If A=0
:■
```

Waar staat het op de GR?



- Maak een nieuw programma: `PROGRAM:FIRSTIF`.
- Druk op PRGM en kies **If**.
- Type een letter.
- Druk op [TEST]=2nd MATH en kies **=**.
- Type **0** gevolgd door ENTER.
- Druk op PRGM en kies **Then**, daarna ENTER.
- Druk op PRGM en kies **Else**, daarna ENTER.
- Druk op PRGM en kies **End**, daarna ENTER.
- Dit is de structuur van het **if**-statement.

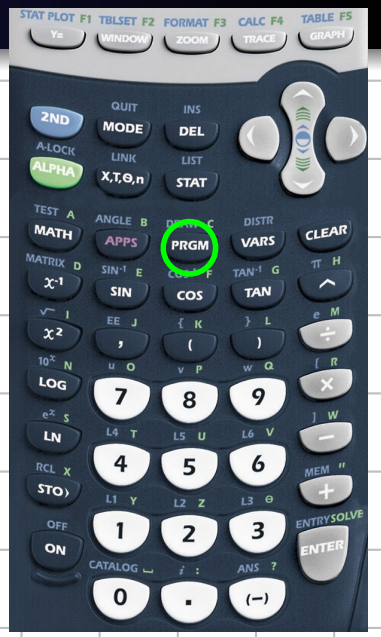
```
CTL I/O EXEC
1:If
2:Then
3:Else
4:For(
5:While
6:Repeat(
7↓End
```

Waar staat het op de GR?

- Maak een nieuw programma: `PROGRAM:FIRSTIF`.
- Druk op PRGM en kies **If**.
- Type een letter.
- Druk op [TEST]=2nd MATH en kies **=**.
- Type **0** gevolgd door ENTER.
- Druk op PRGM en kies **Then**, daarna ENTER.
- Druk op PRGM en kies **Else**, daarna ENTER.
- Druk op PRGM en kies **End**, daarna ENTER.
- Dit is de structuur van het **if**-statement.

```
CTL I/O EXEC
1:If
2:Then
3:Else
4:For(
5:While
6:Repeat(
7↓End
```

Waar staat het op de GR?



- Maak een nieuw programma: `PROGRAM:FIRSTIF`.
- Druk op PRGM en kies **If**.
- Type een letter.
- Druk op [TEST]=2nd MATH en kies *.
- Type **0** gevolgd door ENTER.
- Druk op PRGM en kies **Then**, daarna ENTER.
- Druk op PRGM en kies **Else**, daarna ENTER.
- Druk op PRGM en kies **End**, daarna ENTER.
- Dit is de structuur van het **if**-statement.

```
CTL I/O EXEC
1:If
2:Then
3:Else
4:For(
5:While
6:Repeat(
7:End
```

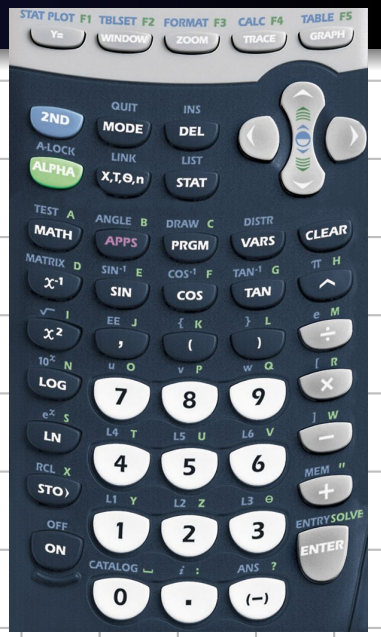
Waar staat het op de GR?

- Maak een nieuw programma: `PROGRAM:FIRSTIF`.
- Druk op [PRGM] en kies **If**.
- Type een letter.
- Druk op [TEST]=[2nd][MATH] en kies *.
- Type **0** gevolgd door [ENTER].
- Druk op [PRGM] en kies **Then**, daarna [ENTER].
- Druk op [PRGM] en kies **Else**, daarna [ENTER].
- Druk op [PRGM] en kies **End**, daarna [ENTER].
- Dit is de structuur van het **if**-statement.

```
PROGRAM:FIRSTIF
:If A=0
:Then
:Else
:End
```

Outline

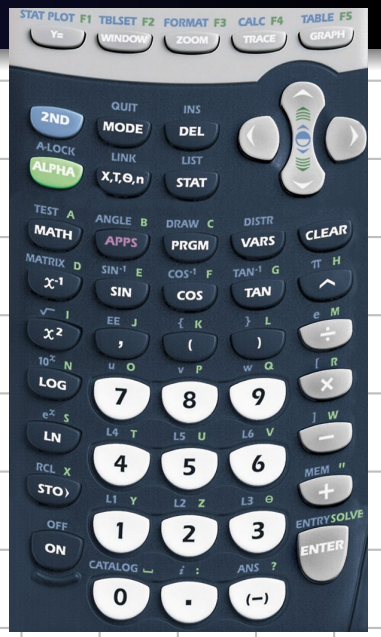
Intermezzo: Invoegen/Insert



- Plaats je cursor op `Else`.
- Druk op [INS]=[2nd][DEL]. Je cursor is nu verandert.
- Druk op [ENTER]. Je hebt nu een regel *boven* ingevoegd.
- [INS]=[2nd][DEL] of de cursor bewegen beëindigd de insert-modus.

```
PROGRAM:FIRSTIF
:If A=0
:Then
:Else
:End
```

Intermezzo: Invoegen/Insert



- Plaats je cursor op `Else`.
- Druk op [INS]=[2nd][DEL]. Je cursor is nu verandert.
- Druk op [ENTER]. Je hebt nu een regel *boven* ingevoegd.
- [INS]=[2nd][DEL] of de cursor bewegen beëindigd de insert-modus.

```
PROGRAM:FIRSTIF
:If A=0
:Then
:■lse
:End
```

Intermezzo: Invoegen/Insert

- Plaats je cursor op `Else`.
- Druk op [INS]=[2nd][DEL]. Je cursor is nu verandert.
- Druk op [ENTER]. Je hebt nu een regel *boven* ingevoegd.
- [INS]=[2nd][DEL] of de cursor bewegen beëindigd de insert-modus.

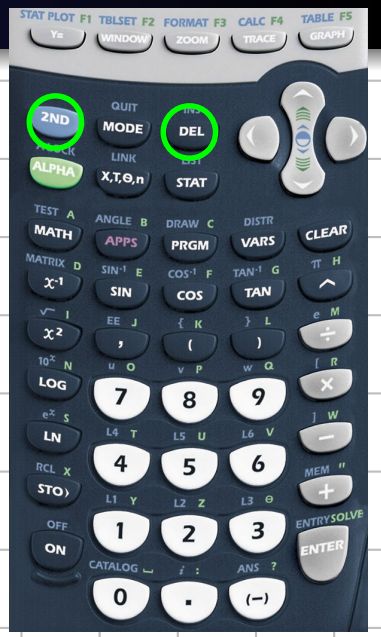
```
PROGRAM:FIRSTIF
:If A=0
:Then
:_lse
:End
```

Intermezzo: Invoegen/Insert

- Plaats je cursor op `Else`.
- Druk op [INS]=[2nd][DEL]. Je cursor is nu verandert.
- Druk op [ENTER]. Je hebt nu een regel *boven* ingevoegd.
- [INS]=[2nd][DEL] of de cursor bewegen beëindigd de insert-modus.

```
PROGRAM:FIRSTIF
:If A=0
:Then
:
:_lse
:End
```

Intermezzo: Invoegen/Insert



- Plaats je cursor op `Else`.
- Druk op [INS]=[2nd][DEL]. Je cursor is nu verandert.
- Druk op [ENTER]. Je hebt nu een regel *boven* ingevoegd.
- [INS]=[2nd][DEL] of de cursor bewegen beëindigd de insert-modus.

```
PROGRAM:FIRSTIF
:If A=0
:Then
:
:Else
:End
```

Outline

Oefening!

Schrijf het volgende Prgm op je GR (bedenk zelf een leuke naam):

- Vraag de gebruiker om een cijfer, *G*.
- Daarna: **Als** «Cijfer 6 of hoger», **dan** «Weergeef OK», **of anders** «Weergeef FAIL».
- Breid dit programma uit zodat een cijfer boven 8 GOOD en boven 9.5 PERFECT zegt.
- Controleer of *G* wel een cijfer is! (Waar moet een cijfer aan voldoen?) Geef een foutmelding indien *G* geen cijfer kan zijn.
  Tip: Je kunt gebruik maken van Stop uit het PRGM CTL menu.

Oefening!

Schrijf het volgende Prgm op je GR (bedenk zelf een leuke naam):

- Vraag de gebruiker om een cijfer, *G*.
- Daarna: **Als** «Cijfer 6 of hoger», **dan** «Weergeef OK», **of anders** «Weergeef FAIL».
- Breid dit programma uit zodat een cijfer boven 8 GOOD en boven 9.5 PERFECT zegt.
- Controleer of *G* wel een cijfer is! (Waar moet een cijfer aan voldoen?) Geef een foutmelding indien *G* geen cijfer kan zijn.
  Tip: Je kunt gebruik maken van Stop uit het PRGM CTL menu.

Merk op dat we het programma in stappen schreven, dit is volgens het 'Scrum' programmeer-paradigma: "Maak eerst een werkend programma, voeg daarna nieuwe features toe!"

Oefening! ÷ Antwoord

```
PROGRAM:GRADER
:Input "GRADE?
",G
:If G<1 or G>10
:Then
:Disp "INVALID
GRADE"
:Stop
:End
:If G≥6:Then
:If G≥8:Then
:If G≥9.5:Then
:Disp "PERFECT"
:Else
:Disp "GOOD"
:End
:Else
:Disp "OK"
:End
:Else
:Disp "FAIL"
:End
```

Dit is een mogelijk antwoord. Uiteraard zijn er meerdere mogelijkheden. Zolang het resultaat maar hetzelfde is!

Merk op dat ik hier `Input` gebruik, i.p.v. `Prompt`. Het werkt hetzelfde, maar heeft als eerste argument een string, hier: "`Grade?` ".

Outline

Hoe zet je een probleem om in een programma?

- Begin met het probleem te versimpelen.
- Denk vervolgens als een computer:
  - Niets is "duidelijk" of "obvious".
  - De computer is dom.
  - Je moet alles letterlijk uitspellen voor de computer!
- Het helpt om gebruik te maken van "pseudocode".

Hoe zet je een probleem om in een programma?

- Begin met het probleem te versimpelen.
- Denk vervolgens als een computer:
  - Niets is “duidelijk” of “obvious”:
    - De computer denkt niet zelf na!
  - Je moet alles letterlijk uitspellen voor de computer!
- Het helpt om gebruik te maken van “pseudocode”.

Hoe zet je een probleem om in een programma?

- Begin met het probleem te versimpelen.
- Denk vervolgens als een computer:
  - Niets is “duidelijk” of “obvious”:
    - De computer denkt niet zelf na!
    - Je moet alles letterlijk uitspellen voor de computer!
- Het helpt om gebruik te maken van “pseudocode”.

Hoe zet je een probleem om in een programma?

- Begin met het probleem te versimpelen.
- Denk vervolgens als een computer:
  - Niets is “duidelijk” of “obvious”:
    - De computer denkt niet zelf na!
  - Je moet alles letterlijk uitspellen voor de computer!
- Het helpt om gebruik te maken van “pseudocode”.

Hoe zet je een probleem om in een programma?

- Begin met het probleem te versimpelen.
- Denk vervolgens als een computer:
  - Niets is “duidelijk” of “obvious”:
    - De computer denkt niet zelf na!
  - Je moet alles letterlijk uitspellen voor de computer!
- Het helpt om gebruik te maken van “pseudocode”.

Hoe zet je een probleem om in een programma?

- Begin met het probleem te versimpelen.
- Denk vervolgens als een computer:
  - Niets is “duidelijk” of “obvious”:
    - De computer denkt niet zelf na!
  - Je moet alles letterlijk uitspellen voor de computer!
- Het helpt om gebruik te maken van “pseudocode”.

Wat is pseudocode?

- Pseudocode ligt tussen “echte” code en mensentaal in.
  - “Echte” code heeft een strakke syntax waar je je aan moet houden.
    - Een kleine typefout en de computer begrijpt je niet!
  - Pseudocode heeft niet zo’n strakke syntax, maar het is “in eigen woorden”
- Pseudocode heeft echter wel de **structuur** van een stukje code

Wat is pseudocode?

- Pseudocode ligt tussen “echte” code en mensentaal in.
  - “Echte” code heeft een strakke syntax waar je je aan moet houden.
    - Een kleine typefout en de computer begrijpt je niet!
  - Pseudocode heeft niet zo’n strakke syntax, maar het is “in eigen woorden”
- Pseudocode heeft echter wel de **structuur** van een stukje code

Wat is pseudocode?

- Pseudocode ligt tussen “echte” code en mensentaal in.
  - “Echte” code heeft een strakke syntax waar je je aan moet houden.
    - Een kleine typefout en de computer begrijpt je niet!
  - Pseudocode heeft niet zo’n strakke syntax, maar het is “in eigen woorden”
- Pseudocode heeft echter wel de **structuur** van een stukje code

Wat is pseudocode?

- Pseudocode ligt tussen “echte” code en mensentaal in.
  - “Echte” code heeft een strakke syntax waar je je aan moet houden. *
    - Een kleine typefout en de computer begrijpt je niet!
  - Pseudocode heeft niet zo’n strakke syntax, maar het is “in eigen woorden”
- Pseudocode heeft echter wel de **structuur** van een stukje code

Wat is pseudocode?

- Pseudocode ligt tussen “echte” code en mensentaal in.
  - “Echte” code heeft een strakke syntax waar je je aan moet houden. *
    - Een kleine typefout en de computer begrijpt je niet!
  - Pseudocode heeft niet zo’n strakke syntax, maar het is “in eigen woorden”
- Pseudocode heeft echter wel de **structuur** van een stukje code

Wat is pseudocode?
Voorbeeld

Stel dat we een programma willen schrijven die ons vertelt of we vandaag een paraplu nodig hebben.

Wat is pseudocode?
Voorbeeld

Stel dat we een programma willen schrijven die ons vertelt of we vandaag een paraplu nodig hebben. In pseudocode:

Algorithm 1 Pseudocode "paraplu nodig?"

```
1: function ParapluNodig(regenkans)
2:     if regenkans = hoog then
3:         Paraplu nodig
4:     else if regenkans = laag then
5:         Geen paraplu nodig
6:     else
7:         Paraplu optioneel.   ▷ Je smelt niet!
8:     end if
9: end function
```

Wat is pseudocode?
Voorbeeld

Stel dat we een programma willen schrijven die ons vertelt of we vandaag een paraplu nodig hebben.

Algorithm 1 Pseudocode "paraplu nodig?"

```
1: function ParapluNodig(regenkans)
2:     if regenkans = hoog then
3:         Paraplu nodig
4:     else if regenkans = laag then
5:         Geen paraplu nodig
6:     else
7:         Paraplu optioneel.   ▷ Je smelt niet!
8:     end if
9: end function
```

```
PROGRAM:PARAPLU
:Prompt R
:If R>40
:Then
:Disp "PARAPLU
NODIG"
:Else:If R<10
:Then
:Disp "PARAPLU
ONNODIG"
:Else
:Disp "PARAPLU
OPTIONEEL"
:End:End
```

Wat is het nut van pseudocode?

Wanneer een programma zeer ingewikkeld wordt, kun je soms honderden regels code samenvatten in slechts 1 zin pseudocode. Bijv. “bereken het elektrisch veld”, zoals in de figuur hieronder.

Wat is het nut van pseudocode?

Wanneer een programma zeer ingewikkeld wordt, kun je soms honderden regels code samenvatten in slechts 1 zin pseudocode. Bijv. “bereken het elektrisch veld”, zoals in de figuur hieronder.

Algorithm 3: Global structure of the multiScatter algorithm

Input: SphereManager
Result: Computed all $\vec{E}$'s of all spheres. $\vec{E}_{ji}^{\text{accum}}$ will be scattered to the camera

```
1  forall the scattering orders, p do
2      forall the 'via' particles, i do
3          Call sphere_i.nextIteration()
4      end
5      forall the 'via' particles, i do
6          forall the 'target' particles, j do
7              forall the 'source' particles, l do
8                  Use [S]_jil to scatter the field: from E_il^{p-1} to E_jil^p, cf. Alg. 5
9                  Accumulate to E_ji^p cf. (3.3).
10             end
11         end
12     end
13     forall the i, j do
14         Accumulate to E_ji^accum cf. (3.8).
15     end
16     If converged, then break the p-loop. Else continue.
17 end
```

Deze figuur komt uit mijn master afstudeerproject: http://www.kevinvanas.nl/TUDelft/MSc/Report.pdf

Wat is een algoritme?

- Op voorgaande slides kwam het woord "**Algorithm**" voor (NL: Algoritme). Wat is dat?
- Een algorithme is "een stukje code".
- Dat "stukje code" heeft als doel om iets specifieks uit te rekenen.
  - Bijvoorbeeld het uittekenen van het zesde priemgetal.
- Dus overal waar "Algorithm" staat, kun je simpelweg denken "*een stukje computercode om iets uit te rekenen*".

Wat is een algoritme?

- Op voorgaande slides kwam het woord "**Algorithm**" voor (NL: Algoritme). Wat is dat?
- Een algorithme is "een stukje code".
- Dat "stukje code" heeft als doel om iets specifieks uit te rekenen.
  - Bijvoorbeeld het uitrekenen van het zesde priemgetal.
- Dus overal waar "Algorithm" staat, kun je simpelweg denken "*een stukje computercode om iets uit te rekenen*".

Wat is een algoritme?

- Op voorgaande slides kwam het woord "**Algorithm**" voor (NL: Algoritme). Wat is dat?
- Een algorithme is "een stukje code".
- Dat "stukje code" heeft als doel om iets specifieks uit te rekenen.
  - Bijvoorbeeld het uitrekenen van het zesde priemgetal.
- Dus overal waar "Algorithm" staat, kun je simpelweg denken "*een stukje computercode om iets uit te rekenen*".

Wat is een algoritme?

- Op voorgaande slides kwam het woord "**Algorithm**" voor (NL: Algoritme). Wat is dat?
- Een algorithme is "een stukje code".
- Dat "stukje code" heeft als doel om iets specifieks uit te rekenen.
  - Bijvoorbeeld het uitrekenen van het zesde priemgetal.
- Dus overal waar "Algorithm" staat, kun je simpelweg denken "*een stukje computercode om iets uit te rekenen*".

Wat is een algoritme?

- Op voorgaande slides kwam het woord "**Algorithm**" voor (NL: Algoritme). Wat is dat?
- Een algorithme is "een stukje code".
- Dat "stukje code" heeft als doel om iets specifieks uit te rekenen.
  - Bijvoorbeeld het uitrekenen van het zesde priemgetal $(= 13)$.
- Dus overal waar "Algorithm" staat, kun je simpelweg denken "*een stukje computercode om iets uit te rekenen*".

Outline

Oefening: Wat doet deze pseudocode?

Algorithm 2 Pseudocode "WhatDoIDo?"

```
1:  function WhatDoIDo(a,b,c)
2:      D = √(b² − 4ac)
3:      if D > 0 then
4:          Er zijn twee oplossingen
5:          x = (−b ± D)/2a
6:      else if D = 0 then
7:          Er is exact één oplossing
8:          x = −b/2a
9:      else                                   ▷ D < 0
10:         Er zijn geen reële oplossingen!
11:     end if
12: end function
```

Oefening: Wat doet deze pseudocode?

Algorithm 3 Pseudocode "WhatDoIDo2?"

```
1:  function WhatDoIDo2(N_att)
2:      Maak N_att random getallen tussen 1 en 6
3:      Vraag of N_def 1 of 2 is
4:      Maak N_def random getallen tussen 1 en 6
5:      Sorteer beide setten van hoog naar laag
6:      if Getal 1 van set 1 > Getal 1 van set 2 then p_1 = p_1 + 1
7:      else    p_2 = p_2 + 1
8:      end if
9:      if Getal 2 van set 1 > Getal 2 van set 2 then p_1 = p_1 + 1
10:     else    p_2 = p_2 + 1
11:     end if
12:     Display de scores: p_1 en p_2
13: end function
```

Oefening: Wat doet deze pseudocode?

Algorithm 4 Pseudocode "WhatDoIDo3?"

```
1: function WhatDoIDo3(geluid g)    ▷ Niet voor GR geschikt
2:     Knip g op in losse klanken, k.
3:     Laad een database in.
4:     Verwijder de stiltes uit k.*
5:     Correleer k met alle items in de database.
6:     Bereken een score voor alle klanken en vind daarmee de
   meest waarschijnlijke letter.
7:     Maak een lijst voor alle mogelijke woorden die g kan zijn.
8:     Vergelijk de lijst met een woordenboek en kies het meest
   waarschijnlijke woord.
9: end function
```

Nu jullie: Maak je eigen pseudocode!

1. Bepaal of iemand genoeg geld, g, op zijn rekening heeft staan, indien hij een product voor x euro wil kopen en r euro in het rood mag staan.
2. Gegeven de huidige datum, (y_t, m_t, d_t), en iemand's geboortedatum, (y_g, m_g, d_g),*bepaal zijn/haar leeftijd.
3. Uit een set van 5 kaarten, bepaal wat de pokerset is.
   - Bijvoorbeeld: "pair of 5's, K high" or "straight, 9 high"
4. Er zijn vier spelers die elk punten hebben. Bepaal de winnaar, i.e. degene met de meeste punten.
5. Bereken de 'rest' van y/x.
   - Bijvoorbeeld: $5/2 = 2$ 'rest' 1
   - Tip: Kijk naar MATH NUM int(of MATH NUM fPart(

Zet pseudocode om in een Prgm!

Implementeer deze twee opdrachten van de vorige slide op je rekenmachine:

1. Bepaal of iemand genoeg geld, g, op zijn rekening heeft staan, indien hij een product voor x euro wil kopen en r euro in het rood mag staan.
2. Gegeven de huidige datum, (y_t, m_t, d_t), en iemand's geboortedatum, (y_g, m_g, d_g), bepaal zijn/haar leeftijd.
   - Tip: Het is handig om voor jezelf duidelijk 6 variabelen te kiezen *én* op te schrijven wat alles betekent. Immers, je kunt slechts één letter gebruiken per variabele! Duidelijkheid is zeer belangrijk: Als je het overzicht verliest, dan weet je zeker dat je je Prgm niet kunt schrijven.

Exercises

- Redeneer m.b.v. formele logica:
  1. Het regent. Ik heb een paraplu. Regent het `and` heb ik een paraplu?
  2. Het is bewolkt. Het is droog. Regent het `or` is het bewolkt?
  3. Het is bewolkt. Het regent. Regent het `xor` is het bewolkt?
  4. Het regent. Regent het niet niet niet?
- Maak de pseudocodes en implementeer alle algoritmes van slide 32 (dat zijn de opdrachten aan het einde van "pseudocode & algoritmes"), voor in hoeverre je kunt.
- De pokeropdracht is optioneel. Ik raad aan om wél de pseudocode te maken, maar het niet te implementeren. Dat is gigantisch veel werk op je GR.
- Breid je `ABCPD` programma uit voor de drie situaties van `D`.

Antwoorden

1. Het regent. Ik heb een paraplu. Regent het `and` heb ik een paraplu?
   - `R=1` `P=1` Gevraagd: `R and P`
   - Dit geeft `1` (true)!
2. Het is bewolkt. Het is droog. Regent het `or` is het bewolkt?
   - `B=1` `R=0` Gevraagd: `B or R`
   - Dit geeft `1` (true)!
3. Het is bewolkt. Het regent. Regent het `xor` is het bewolkt?
   - `B=1` `R=1` Gevraagd: `B xor R`
   - Dit geeft `0` (false)!
4. Het regent. Regent het niet niet niet?
   - `R=1` Gevraagd: `not(not(not(R)))`
   - Redeneer van binnen naar buiten: `not(R)=0` `not(not(R))=1` en dus is het antwoord `0` (false)!

Antwoorden

```
PROGRAM:NUFMONEY
:Input "HOEVEEL
GELD? ",G
:Input "HOEVEEL
KOST HET? ",X
:Input "HOEVEEL
KREDIET? ",R
:
:If G-X≥0:Then
:Disp "JE KUNT
HET BETALEN"
:Else
:If X-G>R:Then
:Disp "TE DUUR"
:Else
:Disp "OK, KOST
KREDIET"
:End
```

```
PROGRAM:BIRTHDAY
:Input "YOU Y",Y
:Input "YOU M",M
:Input "YOU D",D
:Input "CUR Y",Z
:Input "CUR M",N
:Input "CUR D",E
:
:Z-Y→A
:If M>N:Then
:A+1→A
:Else
:If M=N and D>E:Then
:A-1→A
:End
:End
:Disp A
```

Antwoorden

Dit negeert gelijkspel...En is slecht leesbaar. Later meer over dit soort gevallen (m.b.v. Lists).

```
PROGRAM:WIEWINT
:Prompt A
:Prompt B
:Prompt C
:Prompt D
:
:If A>B:Then
:If A>C:Then
:If A>D:Then
:Disp "A WINS"
:Else
:Disp "D WINS"
:End
:Else
:If C>D:Then
:Disp "C WINS"
```

```
:Else
:Disp "D WINS"
:End
:End
:Else
:If B>C:Then
:If B>D:Then
:Disp "B WINS"
:Else
:Disp "D WINS"
:End
:Else
:If C>D:Then
:Disp "C WINS"
:Else
:Disp "D WINS"
:End
:End
:End
```

Antwoorden

```
PROGRAM:RESTYDX
:Disp "REST(Y/X):"
:Input "Y=",Y
:Input "X=",X
:If X=0
:Then
:Disp "X CANNOT BE 0"
:Stop
:End
:Disp fPart(Y/X)*X
:Disp (Y/X-int(Y/X))*X
```

```
PROGRAM:ABC
:Disp "SOLVING AX²+BX+C=0"
:Prompt A,B,C
:B²-4AC→D
:If D>0:Then
:Disp "THERE ARE TWO
SOLUTIONS:"
:(-B+√(D))/(2A)→X
:Disp X
:(-B-√(D))/(2A)→X
:Disp X
:Else:If D=0:Then
:Disp "THERE IS ONE
SOLUTION:"
:(-B)/(2A)→X
:Disp X
:Else
:Disp "THERE IS NO
SOLUTION!"
:End
:End
```