Antwoorden S&C

Ofja, ongeveer dan

Iemand

Januari 2018

Contents

Introduction

Lieve lezer,

Hierbij antwoorden op de leertaken van de cursus Semantiek & Correctheid. Er is geen garantie dat ze correct zijn, gezien ik zeker niet naar alle responsiecolleges ben geweest. Mocht je iets tegenkomen wat niet klopt, laat het me weten.

Succes met het tentamen!

Chapter 1

Leertaak 1

1.1

Lees shit

1.2

Lees meer shit

1.3

$$\mathcal{B}[\![\neg(x = 1)]\!]s$$
$$\mathcal{B}[\![x = 1]\!]s$$
$$\mathcal{A}[\![x]\!]s = \mathcal{A}[\![1]\!]s$$
$$sx = \mathcal{N}[\![1]\!]$$
$$3 = 1$$

Nu lezen we van onderen naar beneden om de truth value te bepalen

$$3 = 1 \qquad \texttt{ff}$$
$$sx = \mathcal{N}[\![1]\!] \qquad \texttt{ff}$$
$$\mathcal{A}[\![x]\!]s = \mathcal{A}[\![1]\!]s \qquad \texttt{ff}$$
$$\mathcal{B}[\![x = 1]\!]s \qquad \texttt{ff}$$
$$\mathcal{B}[\![\neg(x = 1)]\!]s \qquad \texttt{tt}$$

Dus het antwoord is `tt`.

1.4

Lees meer shit.

Voor de syntax geldt dat b wordt uitgebreid tot

$$b ::= \texttt{true}|\texttt{false}|a_1 = a_2|a_1 \leq a_2|\neg b|b_1 \wedge b_2|b_1\&\&b_2|b_1||b_2$$

Voor de semantiek voegen we de volgende twee regels toe:

$$\mathcal{B}[\![b_1\&\&b_2]\!]s = \begin{cases} \mathcal{B}[\![b_2]\!]s & \text{if } \mathcal{B}[\![b_1]\!]s = \text{tt} \\ \text{ff} & \text{otherwise} \end{cases}$$

$$\mathcal{B}[\![b_1||b_2]\!]s = \begin{cases} \text{tt} & \text{if } \mathcal{B}[\![b_1]\!]s = \text{tt} \\ \mathcal{B}[\![b_2]\!]s & \text{otherwise} \end{cases}$$

De gewone And en de conditional-And zijn equivalent in While.

1.5

`result1` = false. `result2` = true.

Het resultaat verandert bij het vervangen niet. Je kan op veel manieren zorgen dat het resultaat wel verandert. De makkelijkste is denk ik !f en f omdraaien, dus

```
result1 = !f() || f()
```

Dan wordt result1 true en result2 false.

Chapter 2

Leertaak 2

2.1

Lees shit

2.2

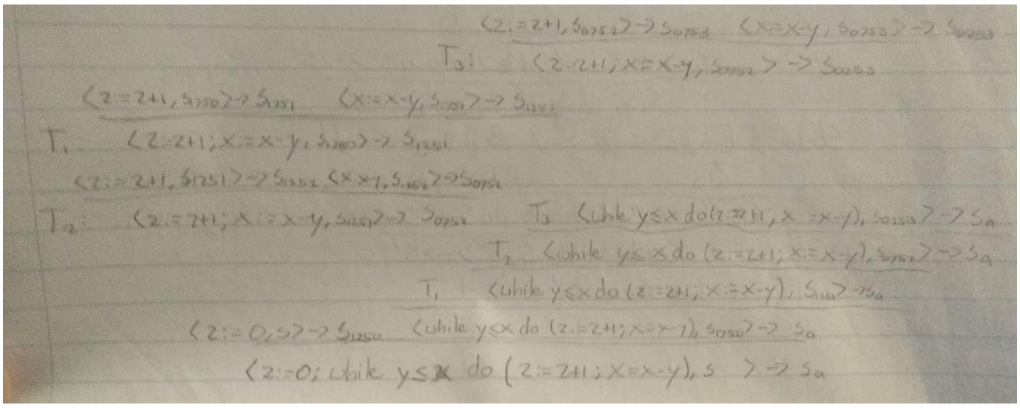


Figure 2.1: Een boom.

2.3

Lees meer shit

2.4

Voeg aan S een nieuwe optie toe, dus:

```
S ::== .... | do S while b
```

2.5

$$[\mathrm{do}_{\mathrm{ns}}^{\mathtt{tt}}] \quad \frac{\langle S, s\rangle \to s' \qquad \langle\ \mathtt{do}\ S\ \mathtt{while}\ b, s'\rangle \to s''}{\langle\ \mathtt{do}\ S\ \mathtt{while}\ b, s\rangle \to s''} \quad \text{if } \mathcal{B}[\![b]\!]s = \mathtt{tt}$$

$$[\mathrm{do}_{\mathrm{ns}}^{\mathtt{ff}}] \quad \frac{\langle S, s\rangle \to s'}{\langle\ \mathtt{do}\ S\ \mathtt{while}\ b, s\rangle \to s'} \quad \text{if } \mathcal{B}[\![b]\!]s = \mathtt{ff}$$

2.6 E2.4

- `while -(x=1) do (y:=y*x; x:=x-1)`. This terminates for x ≥ 1 and loops for x < 1.
- `while 1 <=x do (y:=y*x; x:=x-1`. This terminates for all x.
- `while true do skip`. This loops foreverrrr...

Ik weet niet zo goed hoe je dit aantoont met de semantiek regels, als iemand nog een idee heeft of nog aantekeningen hiervan heeft, zijn die welkom.

Chapter 3

Leertaak 3

3.1

Lees shit

3.2

3.2.1

Als in statement S a_1 de hele tijd op $a_1 - 1$ wordt gezet, bijvoorbeeld, dan loopt de for loop voor eeuwig.

3.2.2

De eerste regel wordt:

$$\frac{\langle S, s[x \mapsto \mathcal{A}[\![a_1]\!]s\rangle \to s' \qquad \langle\ \texttt{for x:=}\ \mathcal{N}^{-1}(\mathcal{A}[\![x]\!]s) + 1\ \texttt{to}\ \mathcal{N}^{-1}(\mathcal{A}[\![a_2]\!]s)\ \texttt{do}\ S, s'\rangle \to s''}{\langle\ \texttt{for x:=}\ a_1\ \texttt{to}\ a_2\ \texttt{do}\ S, s\rangle \to s''}$$

En de tweede blijft hetzelfde.

Het idee is om in plaats van a_1 te gebruiken je x gebruikt en dan wel met s en niet s'. Echter, om x te gebruiken moet je een beetje moeilijk doen met $\mathcal{A}$ en $\mathcal{N}$ omdat x een variabele is en dus geen getal.

3.2.3

Zo'n voorbeeldregel zou zijn:

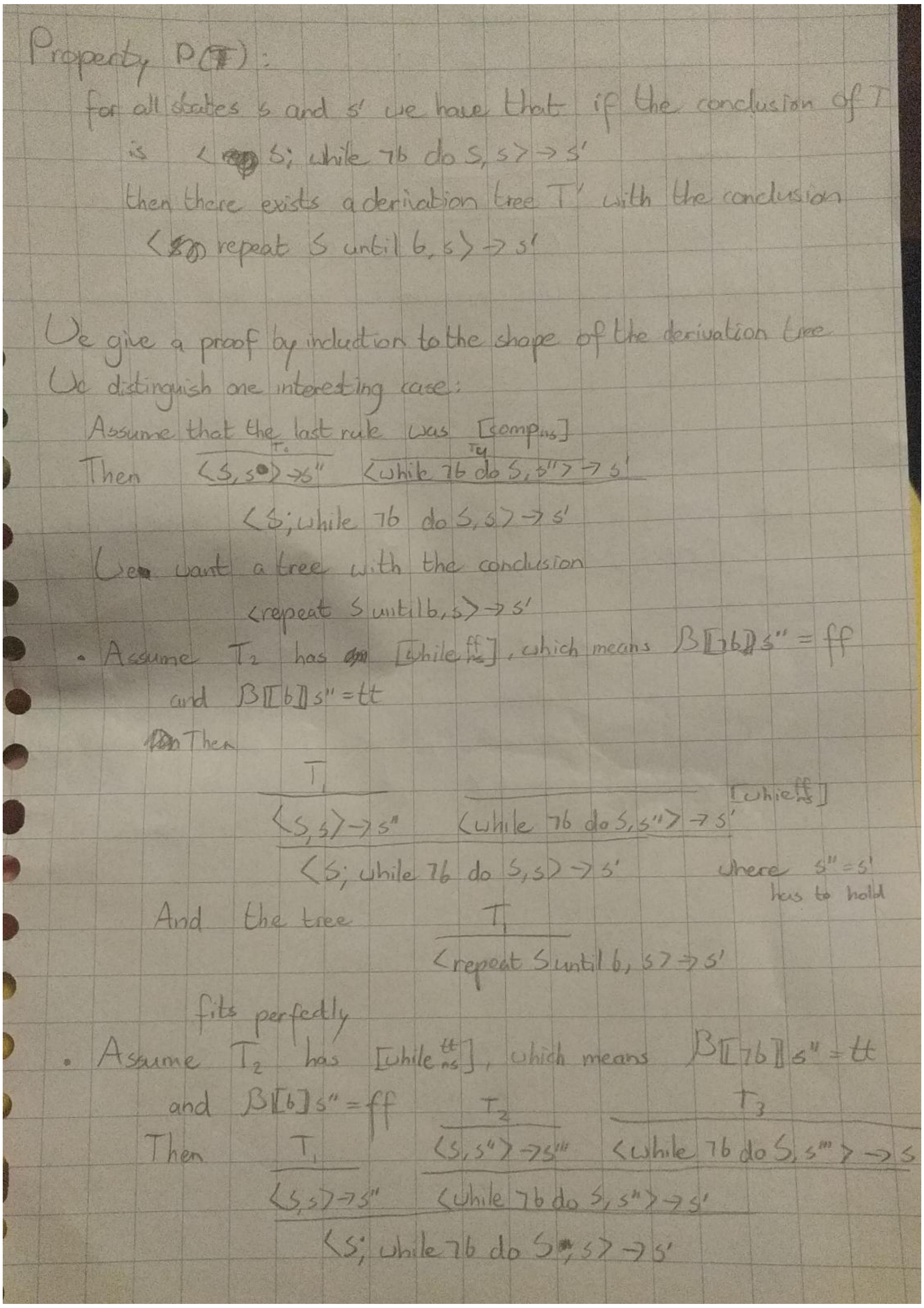
```
for x:=0 to 2 do x:=0
```

3.3

Zoals de opdracht zegt hebben we dit dus al een keer gedaan

3.4

Te bewijzen: $\langle\ S;\ \texttt{while}\ \neg b\ \texttt{do}\ S, s\rangle \to s \Rightarrow \langle\ \texttt{repeat}\ S\ \texttt{until}\ b, s\rangle \to s'$



Property $P(T)$:

for all states s and s' we have that if the conclusion of T is $\langle S;\ \text{while } \neg b \text{ do } S, s\rangle \to s'$

then there exists a derivation tree T' with the conclusion $\langle \text{repeat } S \text{ until } b, s\rangle \to s'$

We give a proof by induction to the shape of the derivation tree.

We distinguish one interesting case:

Assume that the last rule was $[\text{comp}_{ns}]$

Then

$$\frac{\overset{T_1}{\overline{\langle S, s\rangle \to s''}} \quad \overset{T_2}{\overline{\langle \text{while } \neg b \text{ do } S, s''\rangle \to s'}}}{\langle S;\text{while } \neg b \text{ do } S, s\rangle \to s'}$$

We want a tree with the conclusion

$$\langle \text{repeat } S \text{ until } b, s\rangle \to s'$$

- Assume T_2 has $[\text{while}^{ff}_{ns}]$, which means $\mathcal{B}[\![\neg b]\!]s'' = ff$ and $\mathcal{B}[\![b]\!]s'' = tt$

Then

$$\frac{\overset{T_1}{\overline{\langle S, s\rangle \to s''}} \quad \overline{\langle \text{while } \neg b \text{ do } S, s''\rangle \to s'}\ [\text{while}^{ff}_{ns}]}{\langle S;\ \text{while } \neg b \text{ do } S, s\rangle \to s'}$$

where $s'' = s'$ has to hold

And the tree

$$\frac{T_1}{\langle \text{repeat } S \text{ until } b, s\rangle \to s'}$$

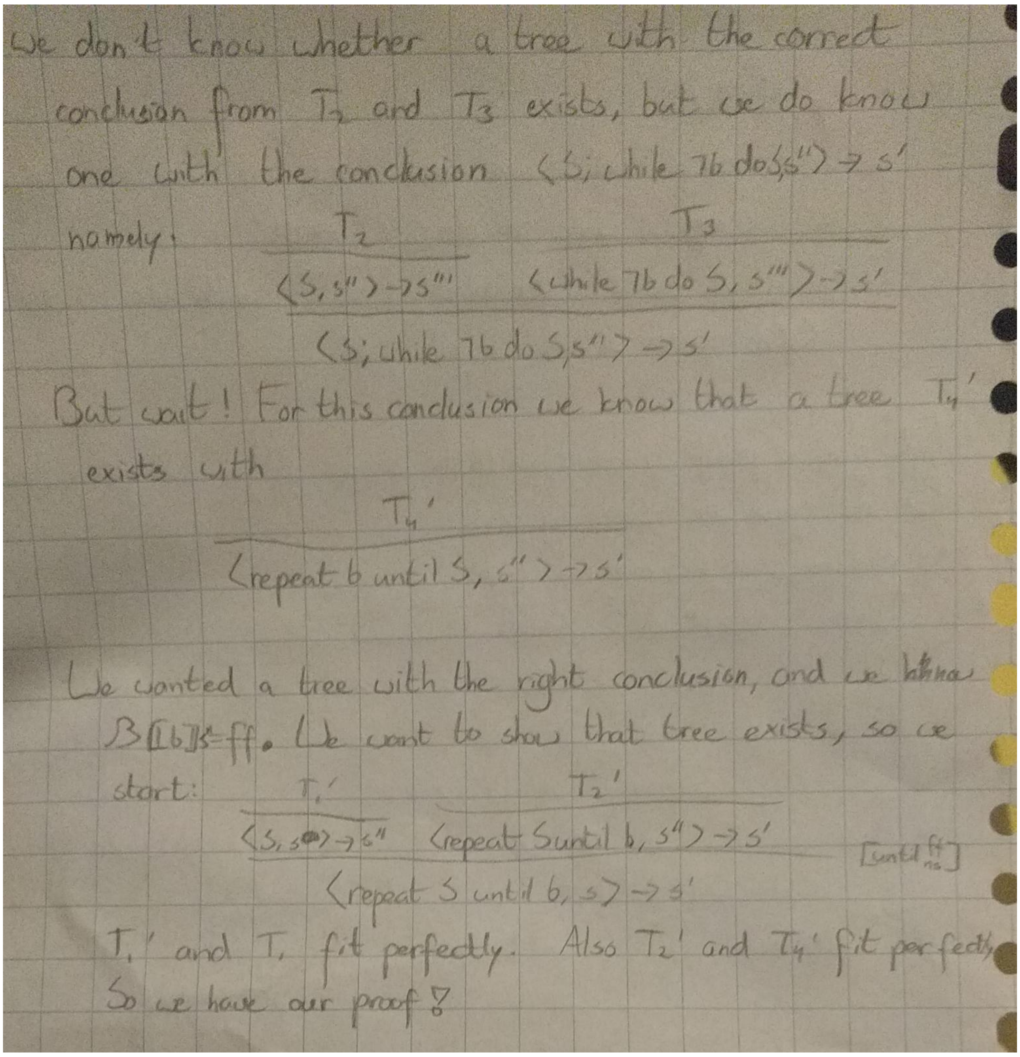
fits perfectly

- Assume T_2 has $[\text{while}^{tt}_{ns}]$, which means $\mathcal{B}[\![\neg b]\!]s'' = tt$ and $\mathcal{B}[\![b]\!]s'' = ff$

Then

$$\frac{\overset{T_1}{\overline{\langle S, s\rangle \to s''}} \quad \dfrac{\overset{T_2}{\overline{\langle S, s''\rangle \to s'''}} \quad \overset{T_3}{\overline{\langle \text{while } \neg b \text{ do } S, s'''\rangle \to s'}}}{\langle \text{while } \neg b \text{ do } S, s''\rangle \to s'}}{\langle S;\ \text{while } \neg b \text{ do } S, s\rangle \to s'}$$

Figure 3.1: Deel 1.



we don't know whether a tree with the correct conclusion from T_2 and T_3 exists, but we do know one with the conclusion $\langle S; \text{while } \neg b \text{ do } S, s'' \rangle \to s'$

namely:

$$\dfrac{\dfrac{T_2}{\langle S, s'' \rangle \to s'''} \quad \dfrac{T_3}{\langle \text{while } \neg b \text{ do } S, s''' \rangle \to s'}}{\langle S; \text{while } \neg b \text{ do } S, s'' \rangle \to s'}$$

But wait! For this conclusion we know that a tree T_4' exists with

$$\dfrac{T_4'}{\langle \text{repeat } b \text{ until } S, s'' \rangle \to s'}$$

We wanted a tree with the right conclusion, and we know $\mathcal{B}[\![b]\!]s = ff$. We want to show that tree exists, so we start:

$$\dfrac{\dfrac{T_1'}{\langle S, s \rangle \to s''} \quad \dfrac{T_2'}{\langle \text{repeat } S \text{ until } b, s'' \rangle \to s'}}{\langle \text{repeat } S \text{ until } b, s \rangle \to s'} \; [\text{until}^{ns}_{ff}]$$

T_1' and T_1 fit perfectly. Also T_2' and T_4' fit perfectly. So we have our proof!

Figure 3.2: Deel 2.

Chapter 4

Leertaak 4

4.1

Lees shit

4.2

lees ˜ als ¬

```
{x=n}
{x=n /\ 1=1}
y:=1;
{x=n /\ y=1}
{x>0->x!=n! /\ y=1 /\ x=n}
{x>0->y*x!=n! /\ n>=x}
while ~(x=1) do (
    {~(x=1) /\ (x>0->y*x!=n! /\ n>=x)}
    {x>0->y*x!=n! /\ n>=x)}
    {(x-1)>0->y*x!=n! /\ n>=(x-1)}
    {(x-1)>0->(x*y)*(x-1)!=n! /\ n>=(x-1)}
    y = x*y;
    {(x-1)>0->y*(x-1)!=n! /\ n>=(x-1)}
    x = x-1
    {x>0->y*x!=n! /\ n>=x}
)
{~~(x=1) /\ (x>0->y*x!=n! /\ n>=x)}
{(x=1) /\ (x>0->y*x!=n! /\ n>=x)}
{(1>0->y*1!=n! /\ n>=1)}
{y*1!=n! /\ n>=1}
{y*1=n! /\ n>0}
{y=n! /\ n>0}
```

4.3

```
{x=n}
{x=n /\ 1=1 /\ 1=1 /\ 1=1}
{x=n /\ 1 = (0+1)^2 /\ 1= 2*0+1 /\ 1 = 0^2 + 1}
z := 0;
{x=n /\ 1 = (z+1)^2 /\ 1 = 2z+1 /\ 1 = z^2 + 1}
odd := 1;
{x=n /\ 1 = (z+1)^2 /\ odd = 2z+1 /\ 1 = z^2 + odd}
sum := 1;
{x=n /\ sum = (z+1)^2 /\ odd = 2z+1 /\ sum = z^2 + odd}
while sum<=x do (
    {x=n /\ sum<=x /\ sum = (z+1)^2 /\ odd = 2z+1 /\ sum = z^2 + odd}
    {x=n /\ sum<=x /\ sum = (z+1)^2 /\ odd = 2z+1 /\ sum = z^2 + odd}
    {x=n /\ sum = (z+1)^2 /\ odd = 2z+1 /\ sum = z^2 + odd}
    {x=n /\ sum = (z+2)^2-(2z+1)-2 /\ odd = 2z+1 /\ sum = z^2 + odd }
    {x=n /\ sum = (z+2)^2-odd-2 /\ odd = 2z+1 /\ sum = z^2 + odd }
    {x=n /\ sum+odd+2 = (z+2)^2 /\ odd = 2z+1 /\ sum = z^2 + odd }
    {x=n /\ sum+odd+2 = (z+2)^2 /\ (odd+2) = 2z+3 /\ (sum+odd+2) = (z+1)^2 + odd+2 }
    {x=n /\ (sum+(odd+2)) = ((z+1)+1)^2 /\ (odd+2) = 2(z+1)+1 /\ (sum+(odd+2)) = (z+1)^2 + (odd+2
    z := z+1;
    {x=n /\ (sum+(odd+2)) = (z+1)^2 /\ (odd+2) = 2z+1 /\ (sum+(odd+2)) = z^2 + (odd+2)}
    odd := odd+2
    {x=n /\ (sum+odd) = (z+1)^2 /\ odd = 2z+1 /\ (sum+odd) = z^2 + odd}
    sum := sum+odd
    {x=n /\ sum = (z+1)^2 /\ odd = 2z+1 /\ sum = z^2 + odd}
)
{x=n /\ ~(sum<=x) /\ sum = (z+1)^2 /\ odd = 2z+1 /\ sum = z^2 + odd}
{x=n /\ ~((z+1)^2<=x) /\ odd = 2z+1 /\ (z+1)^2 = z^2 + odd}
{x=n /\ ~((z+1)^2<=x) /\ (z+1)^2 = z^2 + 2z+1}
{x=n /\ (z+1)^2>x /\ (z+1)^2 = z^2 + 2z+1}
{(z+1)^2>n /\ (z+1)^2 = z^2 + 2z+1}
{(z+1)^2>n /\ z^2=(z+1)^2 - 2z -1}
{(z+1)^2>n /\ z^2<=(z+1)^2 - 2z -1}
??? Hier mist informatie, de invariant moet sterker, iets met z^2<=n toevoegen???
{(z+1)^2>n /\ z^2 <= n}
{z^2 <= n /\ (z+1)^2 > n}
```

Chapter 5

Leertaak 5

5.1

Lees shit

5.2

$$\frac{\{P\}S_1\{Q\} \qquad \{P\}S_2\{R\}}{\{P\}S_1 \texttt{ or } S_2\{Q \vee R\}}$$

Of: $$\frac{\{P\}S_1\{Q\} \qquad \{P\}S_2\{Q\}}{\{P\}S_1 \texttt{ or } S_2\{Q\}}$$

5.3

We willen soundness bewijzen, dus we moeten bewijzen dat: $\forall s, s'(Ps = \texttt{tt} \wedge \langle s_1 \texttt{ or } S_2, s\rangle \to s') \to Qs' = \texttt{tt}$

1. Vanwege inductie kunnen we aannemen dat $\models_p \{P\}S_1\{Q\}$ and $\models_p \{P\}S_2\{Q\}$
2. Laat $Ps = \texttt{tt}$
3. Laat s' zijn zodat $\langle S_1 \texttt{ or } S_2, s\rangle \to s'$
4. Uit 3 volgt dat or1 regel of or2 regel gebruikt is
5. Geval or1: uit 5 volgt dat $\langle S_1, s\rangle \to s'$. Daaruit in combinatie met 1 en 2 bewijst dat $Qs' = \texttt{tt}$
6. Het geval or2 is hetzelfde

5.4

5.5

5.6

1. `{ P } x := x+10 { x>0 }`

   De wlp is `x>-10`. Namelijk:

```
{x>-10}
{x + 10 > -10 + 10}
{x + 10 > 0}
x:= x+10
{x > 0}
```

2. `{ P } x := x+x { x>0 }`
   De wlp is `x>0`. Namelijk:

```
{x>0}
{2x > 0}
{x + x > 0}
x:= x + x
{x > 0}
```

3. `{ P } x := x+y { x=2y }`
   De wlp is `x=y`. Namelijk:

```
{x = y}
{x + y = y + y}
{x + y = 2y}
x:= x+y
{x = 2y}
```

4. `{ P } x := x+y { x=-1 en y=1 }`
   De wlp is `x = -2 /\ y = 1`. Namelijk:

```
{x = -2 /\ y = 1}
{x + 1 = -2 + 1 /\ y = 1}
{x + 1 = -1 /\ y = 1}
{x + y = -1 /\ y = 1}
x:= x+y
{x = -1 /\ y = 1}
```

5. `{ P } x := 2*y; w := -x { x>y en y>0 en wx>=0 }`
   De wlp hier is `wlp(x:=2*y, wlp (w:=-x, x>y>0 en w x>= 0))`

   Dit is gelijk aan: `wlp ( x:=2*y, x>y>0 en -xx>=0`

   Dan ook: `w*y> y> 0 en -(2y) 2y >=0`

   `2y > y > 0 en -4y^2 >= 0`

   `2y>y>0 en y=0`, dus `2 * 0 > 0 > 0 en y = 0`, dus `0>0 en y=0`.

   Maar dan `false en y=0`, dus de wlp is `false`

6. `{ P } x := x+1 { true }`

7. `{ P } x := (m+n) div 2 { m<=x<=n }`
8. `{ P } x := x div y { x = 7 }`
9. `{ P } z := 0 { er bestaat w zodat x^2 + y^2 = w^2 en z=0 }`
10. `{ P } if x = 2*y then x := 3 else y := x div 3 { x=3y }`
11. `{ P } if x = y then x := 0 else y := 0 { x = y }`
12. `{ P } if x < 0 then x := x+2 else skip { x>0 }`

    De wlp van een if statement is `B[b] /\ wlp(S1, Q)` (B[b] / wlp(S2, Q)). Dus in ons geval is dat: `(x<0 /\ wlp(x:=x+2, x>0)) /\ (x>=0 /\ wlp (skip, x>0))` Dus:

    ```
    = (x<0 /\ x + 2 > 0) \/ (x >= 0 /\ x > 0)
    = (x < 0 /\ x > -2) \/ x > 0
    = (-2 < x < 0) \/ x > 0
    = x = -1 \/ x > 0 (we assumed that x is an integer)
    ```

Chapter 6

Leertaak 6

6.1

Lees shit

6.2 E2.16

Bedenk dat x de waarde 17 heeft en y de waarde 5 aan het begin.

```
<z:=0; while y<= x do (z:=z+1; x:=x-y), s>
<while y<= x do (z:=z+1; x:=x-y), s[z->0]>
<if y<=x then ((z:=z+1; x:=x-y); while y<=x do (z:=z+1; x:=x-y)) else skip, s[z->0]>
<(z:=z+1; x:=x-y); while y<=x do (z:=z+1; x:=x-y), s[z->0]>
<x:=x-y; while y<=x do (z:=z+1; x:=x-y), s[z->1]>
<while y<=x do (z:=z+1; x:=x-y), (s[z->1])[x->12]>
```

etc.

6.3 E2.17 en E2.18

```
<repeat S until b, s> =>
<if b then skip else (S; repeat S until b), s>
```

Een andere mogelijkheid is (waar `-` de negatie aangeeft):

```
<repeat S until b, s> =>
<S; if -b then (repeat S until b) else skip, s>
```

Voor de for-loop:

```
<for x:=a1 to a2 do S, s> =>
<if a1<=a2 then (x:=a1; S; for x:= A^(-1)(A[a+1]s) to a2 do S else skip, s>
```

waar `A` $= \mathcal{A}$

6.4

Lees meer shit

6.5

Te bewijzen: als $\langle S_1, s\rangle =>^k s'$ dan $\langle S_1; S_2, s\rangle =>^k \langle S_2, s'\rangle$

Inductie naar k.

Neem $k = 0$. Dan willen we bewijzen dat als $\langle S_1, s\rangle =>^0 s'$ dan $\langle S_1; S_2, s\rangle =>^0 \langle S_2, s'\rangle$

Dan ga je in 0 stappen van configuratie $\langle S_1, s\rangle$ naar s'. Maar $\langle S_1, s\rangle$ is geen eindtoestand en s' wel. Dit kan dus niet. Dan geldt wat we willen bewijzen dus op triviale wijze.

Inductiestap: Zij $k_0 \in N$ zodat $k_0 \geq 0$. Neem aan dat TB geldt voor alle $k_0 \in N$ met $k \leq k_0$ (IH). Dan is wat we willen bewijzen in de inductiestap dat TB ook geldt voor $k_0 + 1$. Dus: als $\langle S_1, s\rangle \Rightarrow^{k_0+1} s'$ dan $\langle S_1; S_2, s\rangle \Rightarrow^{k_0+1} \langle S_2, s'\rangle$

Neem aan dat $\langle S_1, s\rangle \Rightarrow^{k_0+1} s'$. Dan zijn er S_1' en s'' zodat $\langle S_1, s\rangle \Rightarrow \langle S_1', s''\rangle \Rightarrow^{k_0} s'$. Dus in het bijzonder $\langle S_1', s''\rangle \Rightarrow^{k_0} s'$. Pas dan de IH hierop toe. Dus $\langle S_1'; S_2, s''\rangle \Rightarrow^{k_0} \langle S_2, s'\rangle$. De regel $[\text{comp}^1_{\text{ns}}]$ geeft dan

$$\frac{\langle S_1, s\rangle \Rightarrow \langle S_1', s''\rangle}{\langle S_1; S_2, s\rangle \Rightarrow \langle S_1'; S_2, s''\rangle}$$

Uit $\langle S_1; S_2, s\rangle \Rightarrow \langle S_1'; S_2, s''\rangle$ en $\langle S_1'; S_2, s''\rangle \Rightarrow^{k_0} \langle S_2, s'\rangle$ volgt $\langle S_1; S_2, s\rangle \Rightarrow^{k_0+1} \langle S_2, s'\rangle$

6.6

Stel dat je mhet zou vervangen door een 'of', dan zouden alle statements die een eindige afleidingsrij hebben equivalent zijn, wat niet echt handig is.

6.7

6.8

6.9

Lees meeeer

6.10

E2.33

De mogelijke final states zijn alle positieve waardes voor x (niet 0).

E2.35

Volgens mij is het gewoon de volgende regel, maar dat zal wel te simpel zijn gedacht:

$$\frac{\langle S, s\rangle \Rightarrow s'}{\langle \texttt{protect}\ S\ \texttt{end}, s\rangle \Rightarrow \text{s'}}$$

Dit kan ook in natuurlijke semantiek, aangezien die er al vanuit gaat dat alles een blok is wat in een keer uitgevoerd moet worden.

Chapter 7

Leertaak 7