DD1351 Logik för dataloger

Fö 12 – Repetition, några problem med predikatlogik samt generaliseringar av logik

Repetition

Vi repeterar huvudpunkterna i det vi gått igenom

- Satslogikens semantik
- Naturlig deduktion i satslogik
- Predikatlogikens semantik
- Naturlig deduktion i predikatlogik
- Sundhet och fullständighet

Konnektiv

Konnektiv	Namn	Textuell form	Motsvarighet i Java
$\wedge$	Konjunktion	och	`&&`
$\vee$	Disjunktion	eller	`||`
$\neg$	Negation	inte	`!`
$\rightarrow$	Implikation	om…så	

Prioriteter

Vi inför prioriteter mellan konnektiv för att minska antalet parenteser.

$\neg$ har högst prioritet

sedan kommer $\wedge$ och $\vee$

sedan kommer $\rightarrow$

dvs i stället för $(p \wedge (\neg\, q)) \rightarrow r$ skriver vi $p \wedge \neg\, q \rightarrow r$

Formalisering

p : tåget kommer för sent
q : det finns en taxi vid stationen
r : John blir försenad till sitt möte

$\neg\ q$: det finns inte någon taxi vid stationen

$p \land \neg q$: tåget kommer för sent och det finns inte någon taxi vid stationen

$p \land \neg q \rightarrow r$:
Om tåget kommer för sent och det inte finns någon taxi vid stationen, så blir John försenad till sitt möte.

Valueringar

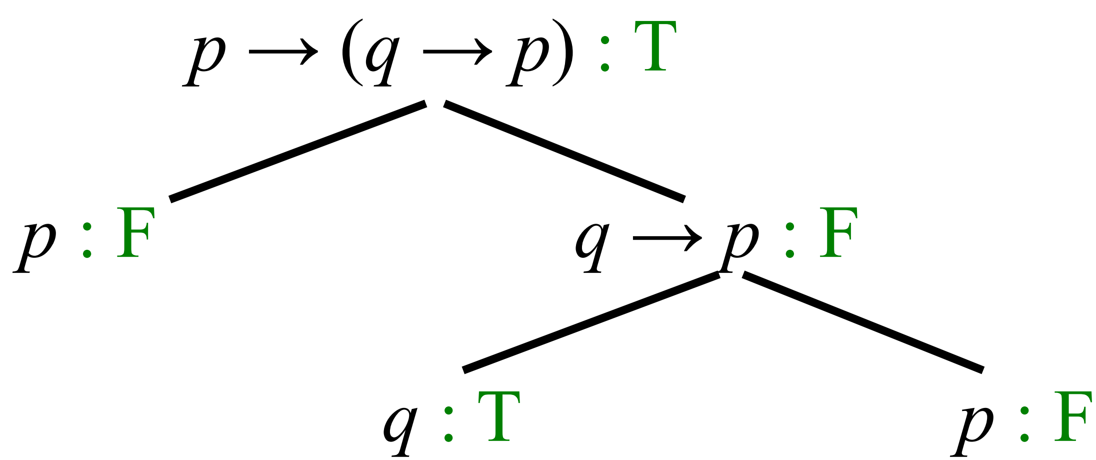
En **valuering** är en tilldelning av sanningsvärden till varje variabel, t.ex.:

$$\{p : \mathrm{F}, q : \mathrm{T}\ , r : \mathrm{F}\}$$

Givet en valuering kan vi beräkna sanningsvärdet på hela formler med hjälp av **sanningsvärdestabeller** (en tabell per konnektiv)

Formelns sanningsvärde

- En formels sanningsvärde kan nu beräknas rekursivt.
- Exempel: $p \to (q \to p)$ och $\{p : \mathrm{F}, q : \mathrm{T}\}$



Logisk konsekvens

Logisk konsekvens **i satslogik** kan undersökas med hjälp av sanningsvärdestabeller.

T.ex. är det sant att $p \wedge q \models p \vee q$?

p	q	$p \wedge q$	$p \vee q$
T	T	T	T
T	F	F	T
F	T	F	T
F	F	F	F

Alla valueringar som gör premisserna sanna

Alla valueringar som gör
slutsatsen sann

Validitet och satisfierbarhet

En formel är **valid** om den är sann i alla modeller.

Enklaste exemplet: $p \vee \neg p$

En formel är **satisfierbar** om den är sann i någon modell.

Enklaste exemplet: p

En formel är **osatisfierbar** om den är falsk i alla modeller.

Enklaste exemplet: $p \wedge \neg p$

Exempel

Gäller följande eller inte?

$$p \wedge q \to \neg r \models r \to (p \to q)$$

Lösning finns i KS från 2015

$p \wedge q \to \neg r \models r \to (p \to q)$?

Vi kan göra en fullständig undersökning med sanningstabeller.

Men vi kan också se om konklusionen går att falsifiera. Om vi ska falsifiera den måste $r \to (p \to q)$ vara falsk. Enda möjligheten att få det är att sätta $\{p: T, q: F, r: T\}$

I den modellen är faktiskt $p \wedge q \to \neg r$ sann.

Konklusionen är inte giltig.

Bevisexempel

Sekvent: $p \to (q \to r) \quad \vdash \quad p \wedge q \to r$

Bevis:

1	$p \to (q \to r)$	premiss
2	$p \wedge q$	antagande
3	p	$\wedge e_1$ 2
4	$q \to r$	$\to e$ 3, 1
5	q	$\wedge e_2$ 2
6	r	$\to e$ 5, 4
7	$p \wedge q \to r$	$\to i$ 2-6

Predikatlogik

- Predikatlogik utökar det satslogiska språket med:
  - variabler
  - konstanter
  - funktionssymboler
  - relationssymboler
  - kvantifierare

Variabler och konstanter

- **Variabler** kan bindas till termer (definieras snart), som representerar objekt
  - Som variabler använder vi typiskt *u*, *v*, *w*, *x*, *y*, *z*
- **Konstanter** representerar objekt
  - typiskt *a*, *b*, *c*, *anna*, *eva*, *sverige*, *stefan_löfven*

Termer

- **Funktionssymboler** är t.ex. f, g, +, •
  - Varje symbol har en **aritet** (antal argument)
- En **term** definieras rekursivt:
  - en variabel är en term
  - en konstant är en term
  - om f är en funktionssymbol av aritet n, och $t_1, \ldots, t_n$ är termer, så är $f\,(t_1, \ldots, t_n)$ en term
  - inget annat är en term

Formler

- **Relationssymboler**, t.ex P, Q, Känner, Primtal, Udda
- Formler definieras rekursivt:
  - om P är en relationssymbol med aritet n, och $t_1, \ldots, t_n$ är termer, så är $P(t_1, \ldots, t_n)$ en formel
  - om φ och ψ är formler så är följande formler: $\neg \varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$, $\varphi \rightarrow \psi$, $\forall x\, \varphi$, $\exists x\, \varphi$
  - inget annat är en formel

Räckvidd

Vilka *kvantifierare* binder vilka *förekomster* av vilka variabler?

$$\forall x\ \exists y\ (P(x,y) \wedge \exists x\ Q(x,y))$$

Räckvidden (eng: scope) för en kvantifierare i en formel ...$(\forall x\ \phi)$... eller ...$(\exists x\ \phi)$... omfattar alla förekomster av variabeln x i delformeln ϕ, med undantag av de som i sin tur tillhör en delformel $\forall x\ \psi$ eller $\exists x\ \psi$ av ϕ

Fria och bundna variabelförekomster

- En förekomst av en variabel x i en formel ϕ är **bunden** om den ligger inom räckvidden för någon kvantifierare, och är **fri** annars
- Exempel: i formeln $\exists y\,(P(x,y) \wedge \exists x\, Q(x,y))$ är första förekomsten av variabeln x fri, och den andra bunden
- En kvantifierare $\forall x$ eller $\exists x$ **binder** alla (fria) förekomster av variabeln x inom sin räckvidd
- Vi kan byta namn på bundna variabler men inte på fria!

Substitution

För en variabel x, en term t och en formel ϕ, betecknar $\phi[t \,/\, x]$ formeln som är resultatet av substitutionen av alla fria förekomster av x i ϕ med t

Exempel:

$$\exists y\, (P(x,y) \wedge \exists x\, Q(x,y))\, [x{+}1/x]$$

$$= \exists y\, (P(x{+}1,y) \wedge \exists x\, Q(x,y))$$

Variabelinfångande

- Problem: $\phi(x) \equiv \exists y\ (x < y)$
  - vad är då $\phi(y)$? $\exists y\ (x < y)[y/x]$?

    men $\exists y\ (y < y)$ har *inte* samma mening!
  - **variabelinfångand***e* (eng: variable capture)
  - när vi gör en substitution måste vi **undvika variabelinfångande!**
- Substitutionen $\phi[t\ /\ x]$ **undviker infångande** om **t är fri för x** *i* ϕ

Variabelinfångande

Termen t är ***fri* för variabeln x i formeln** ϕ om ingen fri förekomst av x i ϕ är inom räckvidden för någon kvantifierare $\forall y$ eller $\exists y$ för någon variabel y i t

Detta kan *alltid* åstadkommas - genom *omdöpning* i formeln, t.ex:

$\phi(x) \equiv \exists y\,(x < y)$ y kan döpas om till z

$\phi(x) \equiv \exists z\,(x < z)$ med samma mening

$\phi(y) \equiv \exists z\,(y < z)$

Substitution som undviker variabelinfångande

OBS: I kursen kommer vi låta

$$\phi[t / x]$$

beteckna resultatet av substitutionen som undviker variabelinfångande!

1. i formeln ϕ, döp om alla kvantifierare som fångar någon variabel i termen t vid någon fri förekomst av variabeln x i formeln ϕ
2. utför substitutionen som vanligt, dvs substituera alla fria förekomster av x i ϕ med t

Slutna och öppna formler

En **sluten** formel (= en **sats**, eng. *sentence*) är en formel utan fria variabler, t.ex. $\forall x\, (U(x))$.

Given en modell M, så är en sats sann eller falsk i M.

En **öppen** formel innehåller fria variabler, t.ex. $U(x)$.

- För att kunna veta om $U(x)$ är sann eller falsk i M måste vi veta vad x är bundet till.
- Därför införs begreppet **omgivning** (def 2.17), vilket är en funktion från variabler till värden, t.ex. $[x \mapsto a]$.
- Detta är inte svårt men blir rätt tekniskt.

I denna kurs kommer vi fokusera på **slutna formler**.

Modeller

En **modell** till en predikatlogiska formel Φ specificerar:

- en mängd *A* (universum, alla objekt vi talar om)
- ett element i *A* för varje konstant i Φ
- en funktion $f : A^n \rightarrow A$ för varje funktionssymbol (med *n* argument) i Φ
- en *n*-ställig relation över *A* för varje predikatsymbol (med *n* argument) i Φ

Logisk konsekvens

Formeln ψ är en **logisk konsekvens** av $\varphi_1, \ \varphi_2, \ \ldots, \ \varphi_n$ om ψ är sann i alla modeller i vilka $\varphi_1, \ \varphi_2, \ \ldots, \ \varphi_n$ är sanna.

Detta skrivs

$$\boldsymbol{\varphi_1, \ \varphi_2, \ \ldots, \ \varphi_n \ \models \ \psi}$$

(Vi förutsätter att $\varphi_1, \ \varphi_2, \ \ldots, \ \varphi_n$, ψ är slutna formler).

Exempel

Ge ett informellt semantiskt resonemang som visar att

$$\exists x \, \forall y \, P(x,y) \models \forall y \, \exists x \, P(x,y)$$

För lösning, se KS 2013

Vi kan resonera så här:

Låt M vara en godtycklig modell

Vi har någon grundmängd A.

Om $\exists x \forall y\, P(x,y)$ är sann i M så finns det något $b \in A$ så att $\forall y\, P(b,y)$ är sann. Då gäller $P(b,y)$ för alla y. Men då gäller också $\exists x\, P(x,y)$ för alla val av y (med b som x). Då gäller $\forall y \exists x\, P(x,y)$

Eftersom det gäller för alla modeller där $\exists x \forall y\, P(x,y)$ är sann så gäller också konsekvensen $\exists x \forall y\, P(x,y) \models \forall y \exists x\, P(x,y)$

Naturlig deduktion

Vi utökar naturlig deduktion till att även gälla för predikatlogik.

Bevisregler:

- alla regler från satslogiken, plus
- introduktions- och elimineringsregler för
  - likhet
  - all-kvantifiering $\forall x$
  - existens-kvantifiering $\exists x$

Bevisexempel

Sekvent: $\forall x\,(P(x) \rightarrow Q(x)),\ \exists x\,P(x)\ \vdash\ \exists x\,Q(x)$

Bevis:

1		$\forall x\,(P(x) \rightarrow Q(x))$	premiss
2		$\exists x\,P(x)$	premiss
3	x_0	$P(x_0)$	antagande
4		$P(x_0) \rightarrow Q(x_0)$	$\forall x$ e 1
5		$Q(x_0)$	$\rightarrow$e 3, 4
6		$\exists x\,Q(x)$	$\exists x$ i 5
7		$\exists x\,Q(x)$	$\exists x$ e 2,3-6

Exempel

Visa med naturlig deduktion att

$$\forall x\,(\neg I(x) \vee S(x)) \vdash \neg \exists x\,(I(x) \wedge \neg S(x))$$

För lösning, se KS 2014

$\forall x\,(\neg I(x) \vee S(x)) \vdash \neg \exists x\,(I(x) \wedge \neg S(x))$

1.	$\forall x\,(\neg I(x) \vee S(x))$	Premiss
2.	$\exists x\,(I(x) \wedge \neg S(x))$	Antagande
3.	$I(x_0) \wedge \neg S(x_0)$	Antagande
4.	$I(x_0)$	$\wedge e_1$ 3
5.	$\neg S(x_0)$	$\wedge e_2$ 3
6.	$\neg I(x_0) \vee S(x_0)$	$\forall x e$ 1
7.	$\neg I(x_0)$	Antagande
8.	$\perp$	$\neg e$ 4, 7
9.	$S(x_0)$	Antagande
10	$\perp$	$\neg e$ 9, 5
11.	$\perp$	$\vee e$ 6, 7-8, 9-10
12.	$\perp$	$\exists x e$ 2, 3-11
13.	$\neg \exists x\,(I(x) \wedge \neg S(x))$	$\neg i$ 2-12

Sundhet och fullständighet

Precis som för satslogik kan man visa:

Naturlig deduktion är ett **sunt** och **fullständigt** bevissystem för predikatlogik

dvs $\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \psi$ omm $\varphi_1, \varphi_2, \ldots, \varphi_n \models \psi$

Detta visades 1929 av den österrikiske logikern **Kurt Gödel**.

Kurt Gödel (1906-1978)

Begränsningar hos predikatlogik

Vi kan beskriva en modell M genom att ange formler som skall vara sanna i M.

Vi kan t.ex. ta Peanos axiom. De är tänkta att vara giltiga i modellen N (de naturliga talen). Det bästa vore om de bara vore giltiga i N. Detta mål nås nästan. Det finns vissa oväntade konstiga modeller som också uppfyller axiomen. Gödel visade att det är oundvikligt.

Gödels ofullständighetssats

Kurt Gödel visade 1931 att Peano-aritmetik **inte** är negationsfullständigt.

Mer specifikt lyckades han konstruera en formel G som man kan inse är **sann,** men sådan att P*eano* $\nvdash$ G och *Peano* $\nvdash \neg G$.

Beviset är gjort så att för **varje tänkbar uppsättning axiom** för heltalen kan vi konstruera en sådan formel G. Alltså **går det inte att axiomatisera aritmetiken**.

Detta är **Gödels** berömda **första ofullständighetssats**.

Gödels bevis på 2 bilder (1)

Insikt 1: Formler och bevis kan kodas som tal.

Insikt 2: Eftersom Peano-aritmetik kan representera alla beräkningsbara funktioner, och beviskontroll är beräkningsbart, så finns det en aritmetisk formel $Prf(x,y)$ som är sann exakt när y (kodat som ett tal) är ett bevis för x (kodat som ett tal).

(x är ett mycket stort tal, y är ännu mycket större.)

Formeln $\neg\exists y(\ Prf(n,y))$ uttrycker då att ”det finns inget bevis för formeln med koden n”.

Gödels bevis på 2 bilder (2)

Formeln $\neg\exists y(\ Prf(n,y))$ uttrycker att ”det finns inget bevis för formeln med koden *n*”.

Gödel lyckades nu konstruera en formel G med kod *m* och som han kunde bevisa var ekvivalent med $\neg\exists y(\ Prf(m,y))$. Det betyder att *G* säger ”jag är inte bevisbar”.

Om nu *G* är falsk, så är det också falskt att *G* inte är bevisbar, dvs vi kan bevisa något falskt utifrån våra axiom. Då måste axiomen vara falska eller innehålla en motsägelse, men vi vet att axiomen är sanna i ASM.

Alltså är *G* sann men inte bevisbar. Underbart!

Mängdlära och ZF-axiomen

Ännu har man inte funnit någon sann men icke bevisbar formel som uttrycker något ”naturligt” samband mellan naturliga tal.

Annat är det i mängdläran, där flera viktiga utsagor har visat sig vara oberoende av de sk **Zermelo-Fraenkel-axiomen.**

T.ex. både **kontinuumhypotesen** (se Fö 6) och dess negation är bägge konsistenta med axiomen.

Problem: Hur definieras oändlighet?

Vi kan ange en formel F som säger att alla modeller den är sann i innehåller exakt 3 element.

$$E_3 : \ \exists x \, \exists y \, \exists z \, (x \neq y \wedge x \neq z \wedge y \neq z \wedge \forall u \, (u = x \vee u = y \vee u = z))$$

På samma sätt kan vi definiera E_n för alla n.

Antag nu att vi skulle vilja definiera en formel som är sann enbart i modeller som är *ändliga*, d.v.s. modeller som har ett ändligt universum.

Om vi kunde säga att exakt en av formlerna

$$E_1, E_2, E_3, E_4, \ldots$$

är sann i modellen så skulle det fungera. Men vi kan inte uttrycka detta i en enda formel. Det går att visa att en sådan formel inte kan finnas.

Men om vi tar formeln

$$\forall x\,(c \neq f(x)) \wedge \forall x\,\forall y\,(f(x) = f(y) \rightarrow x = y)$$

Det går att visa att denna formel är sann endast i oändliga modeller.

Det går också att inse att negationen av formeln trots detta inte karakterisera ändliga modeller.

Rekursivt uppräkningsbara mängder

En mängd M är **rekursivt uppräkningsbar** om man kan skriva ett datorprogram P som givet ett objekt x ger resultatet ”ja” precis när $x \in M$.

Om $x \notin M$ så kan P svara ”nej” eller **inte terminera** alls.

Alla dessa mängder är rekursivt uppräkningsbara (och t.om. **rekursiva**; se nästa bild): {alla Java-program}, {alla predikatlogiska formler}, N, $\mathrm{N}_{\text{Jämn}}$, Q

Rekursiva mängder

En mängd M är **rekursiv** om om man kan skriva ett datorprogram P som givet ett objekt x ger resultatet ”ja” precis när $x \in M$, och ”nej” precis när $x \notin M$.

Dvs, P terminerar **alltid** och svarar alltid rätt.

Om M är rekursiv så säges problemet att bestämma huruvida $x \in M$ eller ej vara **avgörbart**.

M är rekursiv om och endast om både M och M' (=komplement-mängden till M) är rekursivt uppräkningsbara.

Ett oavgörbart problem

Betrakta alla möjliga Javaprogram J_1, J_2, ... som tar ett heltal som resultat och returnerar ett heltal som svar (eller aldrig terminerar).

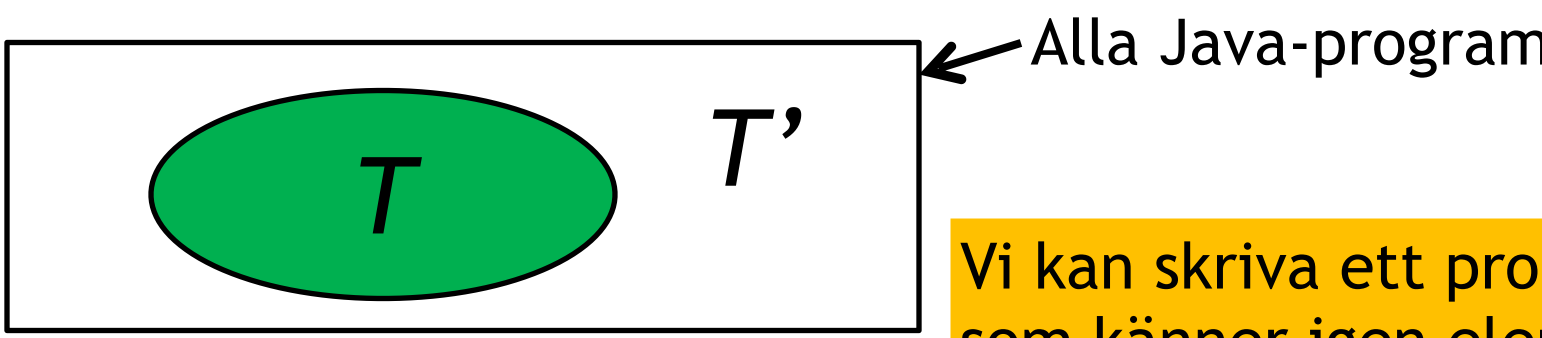
$$\text{Definiera } D(n) = \begin{cases} 1 & \text{om } J_n(n) \text{ inte terminerar} \\ J_n(n)+1 & \text{annars} \end{cases}$$

Antag att D kan implementeras (dvs $D = J_k$ för något k). Men då kommer $D(k)$ antingen både returnera 1 och inte terminera, eller returnera både x och x+1. **Motsägelse:** alltså finns inget program som beräknar D.

Termineringsproblemet

Diagonaliserings-resonemanget på föregående bild visar att **termineringsproblemet** (eng. *halting problem*) är ett oavgörbart problem.

Mängden *T* av terminerande Java-program är rekursivt uppräkningsbar men **inte rekursiv**.



Vi kan skriva ett progra
som känner igen elemen
i *T* men inte i *T*' !

Vad innebär termineringsproblemet?

Att termineringsproblemet är oavgörbart betyder **inte** att vi **aldrig** kan säga om ett program terminerar.

Program 1:

```
print( "hej" );
```

Terminerar

Program 2:

```
while (true )
    { print("hej"); }
```

Terminerar inte

Men det innebär att vi **inte kan hitta en systematisk metod** som **alltid** avgör, givet ett program P och indata x, ifall P terminerar på x eller ej.

Olika utvidgningar av logik

Det finns en del varianter på klassisk logik. Några exempel är:

- Högre ordningens logik
- Intuitionistisk logik
- Modal logik
- Temporal logik

Högre ordningens logik

Det går att visa att om vi tar formeln

$$\neg\exists f\,(\forall x(c \neq f(x)) \wedge \forall x\,\forall y\,(f(x) = f(y) \to x = y))$$

så kommer den faktiskt att vara sann precis i alla ändliga modeller. Men det nya är att den existentiella kvantifieraren här löper över *funktioner*. Om man tillåter det har vi ett exempel på *högre ordningens logik*. Ett problem med den typen av logik är att naturlig deduktion inte längre kommer att vara effektivt avgörbar.

Intuitionistisk logik

Vet vi säkert att $P \vee \neg P$ nödvändigtvis måste vara sant? I intuitionistisk logik godkänner vi inte det antagandet. I intuitionistisk logik använder vi naturlig deduktion med ett begränsat antal regler.

Speciellt får vi inte använda reglerna $\vee_e, \neg\neg_e$

Detta leder till att antalet valida formler kommer att minska. Intuitionistisk logik är knutet till något som kallas *konstruktivists matematik*.

Modal logik

Modal logik går ut på att vi i viss mening graderar sanning genom att lägga på *modaliteter*.
En möjlighet är att skilja mellan att säga att p är sant (bara) och att säga att p är *nödvändigt sant*. Den senare typen av sanning brukar skrivas som

$$\Box\, p$$

Det finns olika typer av modala logiker som kan beskrivas med olika axiom. Ett exempel är systemet S4 som definieras av axiomen:

$1.\ \Box\, A \to A$

$2.\ \Box\,(A \to B) \to (\Box\, A \to \Box\, B)$

$3.\ \Box\, A \to \Box\Box\, A$

En speciell variant av modal logik är *temporal logik*.

Resten av kursen

Matematisk induktion och strukturell induktion

Hoare-logik

Temporal logik (i ett specialfall)

Exempel på bevis i Hoare-logik

$(\!| \; x = x_0 \; |\!)$ Precondition

`if (x > 0) {`

$(\!| \; x = x_0 \wedge x > 0 \; |\!)$ If

$(\!| \; x = |x_0| \; |\!)$ Implied (➽)

`y = x;`

$(\!| \; y = |x_0| \; |\!)$ Assignment

`} else {`

$(\!| \; x = x_0 \wedge \neg(x > 0) \; |\!)$ If

$(\!| \; -x = |x_0| \; |\!)$ Implied (➽)

`y = -x;`

$(\!| \; y = |x_0| \; |\!)$ Assignment

`}`

$(\!| \; y = |x_0| \; |\!)$ Postcondition

Några fler exempel:

$0 \cdot 0 = 0$

Hur visar vi det. Jo:

$0 \cdot 1 = 0(0+1)$ [Def av "+1"]

$0(0+1) = 0 \cdot 0 + 0$ [4:de axiomet]

$0 \cdot 0 + 0 = 0 \cdot 0$ [3:dje axiomet]

Så $0 \cdot 1 = 0 \cdot 0$

Men $0 \cdot 1 = 0$ [5:te axiomet

Så $0 \cdot 0 = 0$