



## Musterlösungen zum Übungsblatt 8

### Lösung

#### 1. Striktheit und Rechenstruktur BOOL

Teilaufgabe 1)

Eine Funktion heißt strikt, falls sie das Ergebnis "undefiniert" ( $\perp$ ) liefert, wenn nur ein Eingabeparameter undefiniert ( $\perp$ ) ist.

Formal: Sei  $f: M_1 \times \dots \times M_n$  eine strikte Funktion. ( $\perp \in M_i \forall 1 \leq i \leq n$ ) Dann gilt:

Falls  $\exists x_i: x_i = \perp, 1 \leq i \leq n$ , dann  $f(x_1, \dots, x_n) = \perp$ .

Beispiel:

Sei  $f$  eine strikte Abbildung, die die Summe der Quadratwurzeln von 4 Zahlen berechnet:

$f: \mathbb{N} \times \mathbb{N} \times \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}_{\perp}$ ,  $f(a,b,c,d) = \sqrt{a} + \sqrt{b} + \sqrt{c} + \sqrt{d}$ . (+ und  $\sqrt{\phantom{x}}$  sind hier als strikte Verknüpfungen anzusehen.)

Definierter Fall ohne  $\perp$ :  $f(1, 4, 9, 16) = 10$

Jedoch:  $f(3, 4, -1, 1) = \perp$

( $-1 \notin \mathbb{N}$ , daher ist  $-1$  bezüglich der Menge  $\mathbb{N}$  Bottom. Der Ergebnis Typ ist  $\mathbb{N}_{\perp} := \mathbb{N} \cup \{\perp\}$ , um Fälle, wie den soeben behandelten auch abdecken zu können)

Teilaufgabe 2)

Signatur:

$fct \cdot || = (bool, bool) bool$

(Bemerkung: Die obige Schreibweise, z.B.  $f: \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$  ist keine Signatur sondern lediglich die Zuordnung einer Funktionalität zu einem Funktionssymbol. Diese Schreibweise wird allgemein für Funktionen (also auch für Rechenstrukturen) verwendet. Signaturen verwendet man, wenn man explizit Rechenstrukturen behandelt.)

Sei  $b \in \{true, false, \perp\}$  beliebig.

Dann gilt:

$$a || b = \begin{cases} true, & \text{falls } a = true \\ b, & \text{falls } a = false \\ \perp, & \text{falls } a = \perp \end{cases}$$

Wertetabelle:

a	b		$a \dot{\cup} b$	$a \parallel b$
true	true		true	true
true	false		true	true
false	true		true	true
false	false		false	false
$\perp$	$\perp$		$\perp$	$\perp$
true	$\perp$		$\perp$	<b>true</b>
false	$\perp$		$\perp$	$\perp$
$\perp$	true		$\perp$	$\perp$
$\perp$	false		$\perp$	$\perp$

### Teilaufgabe 3)

```
import info1.*;
```

```
public class SDis{ //Verwendung von strikter Disjunktion
    public static void main(String args[]) {
        int a;
        int b;

        //Zwei Zahlen einlesen
        System.out.println("Bitte zwei Zahlen eingeben:");
        a=Console.in.readInt();
        b=Console.in.readInt();

        //Vergleich mit striktem Oder
        if (true | a/b==0)
            System.out.println("Wenn man das hier liest, ist b!=0");
    }
}
```

```
import info1.*;
```

```
public class NSDis{ //Verwendung von nicht strikter Disjunktion
    public static void main(String args[]) {
        int a;
        int b;

        //Zwei Zahlen einlesen
        System.out.println("Bitte zwei Zahlen eingeben:");
        a=Console.in.readInt();
        b=Console.in.readInt();

        //Vergleich mit NICHT striktem Oder
        if (true || a/b==0)
```

```

        System.out.println("Diese Anweisung wird immer erreicht.");
    }
}

```

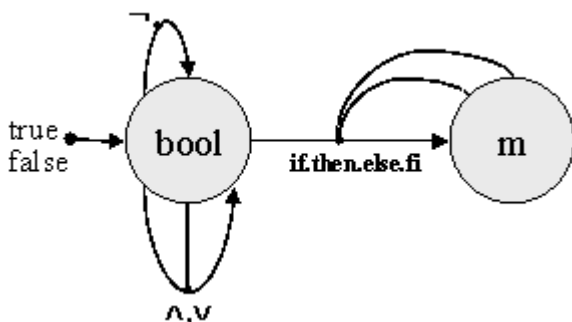
Wird für b 0 eingegeben, so wird der Unterschied zwischen strikter und nicht-strikter Disjunktion deutlich. Das erste Programm bricht mit einer Ausnahme ab, das zweite terminiert trotzdem korrekt, da der Quotient a / b nicht mehr ausgewertet wird.

Wenn in Java eine Funktion (hier a / b) Bottom als Ergebnis hat, so bricht das Programm mit einer Ausnahme ab.

## Lösung

### 2. Rechenstruktur BOOLIF

a)



b) Eine strikte Funktionsspezifikation von if . then . else . fi muß für alle a, b und c die Belegungen  $\perp$  , true und false berücksichtigen. Die strikte Funktionsspezifikation lautet:

$$\text{if } a \text{ then } b \text{ else } c \text{ fi} = \begin{cases} a = \perp \vee b = \perp \vee c = \perp \\ b, & a = \text{true}^{\text{BOOLIF}}, c \neq \perp \\ c, & a = \text{false}^{\text{BOOLIF}}, b \neq \perp \end{cases}$$

c)

Behauptung: Es gilt  $\text{if } x \text{ then } y \text{ else } z \text{ fi} = (x \wedge^{\text{BOOLIF}} y) \vee^{\text{BOOLIF}} (\neg^{\text{BOOLIF}} x \wedge^{\text{BOOLIF}} z) =: f(x,y,z)$

Beweis:

Man zeigt, dass f für jeden der 4 Fälle aus b) das Gleiche liefert wie fct if.then.else.fi.

Man beachte, dass sämtliche Funktionen aus BOOLIF strikt sind.

Zur besseren Übersichtlichkeit setzen wir  $\text{false}^{\text{BOOLIF}} := \text{false}$  und  $\text{true}^{\text{BOOLIF}} := \text{true}$ .

Fall:  $x = \perp, y, z \in B^+$ :

$$\begin{aligned} & \text{if } \perp \text{ then } y \text{ else } z \text{ fi} = \perp \\ & f(\perp, y, z) = (\perp \wedge^{\text{BOOLIF}} y) \vee^{\text{BOOLIF}} (\neg^{\text{BOOLIF}} \perp \wedge^{\text{BOOLIF}} z) \end{aligned}$$

$$\begin{aligned}
&= \perp \vee \text{BOOLIF } (\perp \wedge \text{BOOLIF } z) \\
&= \perp \vee \text{BOOLIF } \perp \\
&= \perp
\end{aligned}$$

Fall:  $x = \text{true}, z \neq \perp$ :

$$\begin{aligned}
&\text{if true then y else z fi} = y \\
&f(\text{true}, y, z) = (\text{true} \wedge \text{BOOLIF } y) \vee \text{BOOLIF } (\neg \text{BOOLIF } \text{true} \wedge \text{BOOLIF } z) \\
&= (\text{true} \wedge \text{BOOLIF } y) \vee \text{BOOLIF } (\text{false} \wedge \text{BOOLIF } z) \\
&= (\text{true} \wedge \text{BOOLIF } y) \vee \text{BOOLIF } \text{false} \\
&= (\text{true} \wedge \text{BOOLIF } y) \\
&= y
\end{aligned}$$

Fall:  $x = \text{false}, y \neq \perp$ :

$$\begin{aligned}
&\text{if false then y else z fi} = z \\
&f(\text{false}, y, z) = (\text{false} \wedge \text{BOOLIF } y) \vee \text{BOOLIF } (\neg \text{BOOLIF } \text{false} \wedge \text{BOOLIF } z) \\
&= (\text{false} \wedge \text{BOOLIF } y) \vee \text{BOOLIF } (\text{true} \wedge \text{BOOLIF } z) \\
&= \text{false} \vee \text{BOOLIF } (\text{true} \wedge \text{BOOLIF } z) \\
&= \text{false} \vee \text{BOOLIF } (\text{true} \wedge \text{BOOLIF } z) \\
&= (\text{true} \wedge \text{BOOLIF } z) \\
&= z
\end{aligned}$$

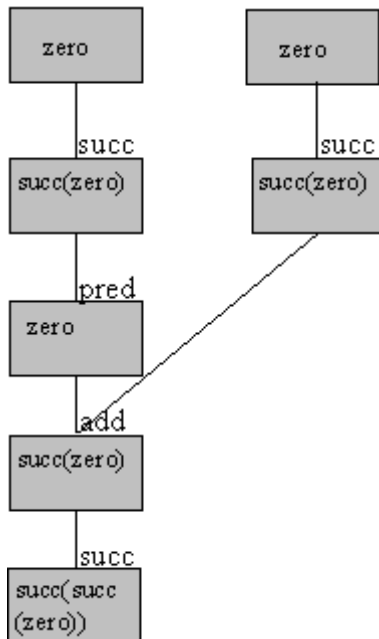
Fall  $y = \perp, z = \perp, x \in B^+$ :

$$\begin{aligned}
&\text{if x then } \perp \text{ else } \perp = \perp \\
&f(x, \perp, \perp) = (x \wedge \text{BOOLIF } \perp) \vee \text{BOOLIF } (\neg \text{BOOLIF } x \wedge \text{BOOLIF } \perp) \\
&= \perp \vee \text{BOOLIF } \perp \\
&= \perp
\end{aligned}$$

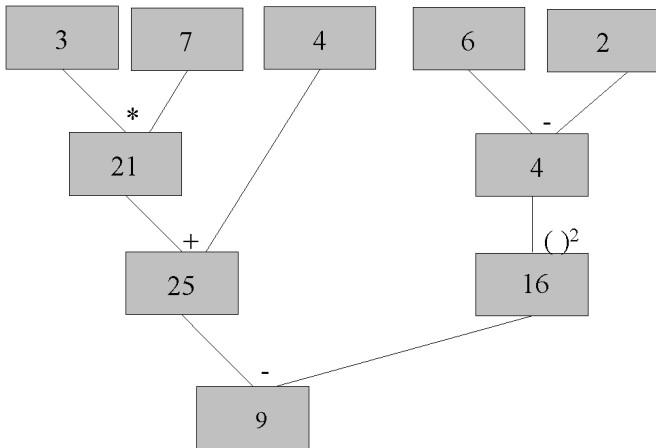
## Lösung

### 3. Berechnungsformulare

a)  $\text{succ}(\text{add}(\text{pred}(\text{succ}(\text{zero})), \text{succ}(\text{zero})))$



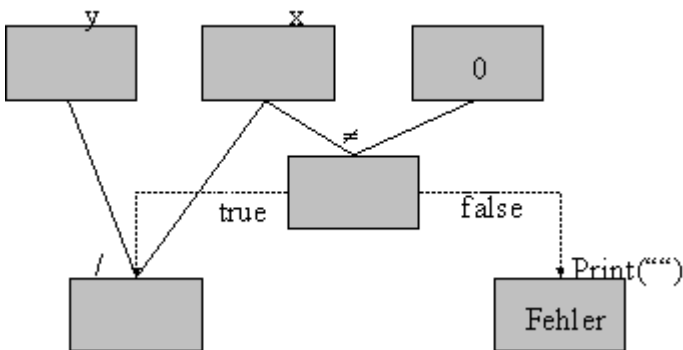
b)  $(3*7)+4 - (6-2)^2$



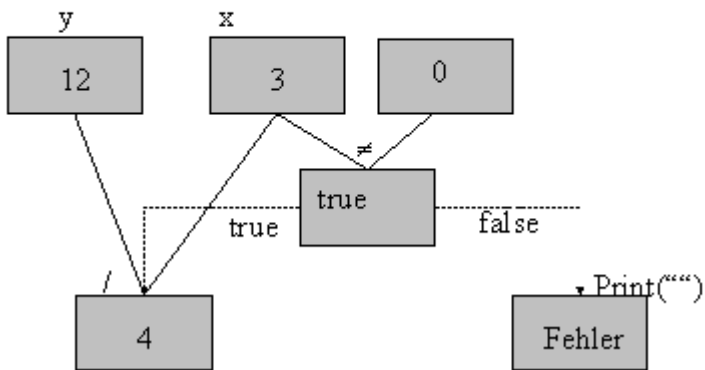
c. `if x ≠ 0 then y / x else print(`Fehler`)`

Das Problem bei dieser Aufgabe ist, dass durch die Fallunterscheidung zwei mögliche Berechnungsschemata zu einem Term existieren, da die Bedingung der if-Anweisung wahr oder falsch sein kann.

Das Problem löst man, indem man die Berechnung, je nach Ergebnis der Fallunterscheidung, in einem weiteren Rechenformular fortsetzt, zu welchem gestrichelte Pfeile führen.



d) Einsetzen der x- und y-Werte:  $x = 12, y = 3$



## Lösung

### 4. Java: Ganzzahlige Quadratwurzel

```
import info1.*; class Quad { public static void main( String[] args ) { int i = 0; int n; System.out.print("Bitte eine Zahl eingeben:"); n = Console.in.readInt(); while((i+1)*(i+1) <= n) // Bedingung für i+1 prüfen, da i noch einmal i++; // erhöht wird, //wenn die Bedingung gerade noch richtig ist. System.out.println("Quadratwurzel von "+n+" ist >= "+i+ "."); } }
```

---