

Algorithmen und Programmierung

2. Aufgabenblatt

Aufgabe 2.1

- a) Jedes Konto besitzt einen Betrag: **total**
Es gibt Konten mit dem gleichen Betrag, aber nicht alle Beträge kommen vor:
nicht injektiv und nicht surjektiv
- b) Es gibt Studenten, die keine Vorlesung besuchen: **partiell**
Ein Student kann mehrere Vorlesungen gleichzeitig besuchen:
nicht rechtseindeutig => keine Funktion
- c) Nicht jede Person besitzt einen Kfz-Brief: **partiell**
Es gibt Personen, die mehrere Autos besitzen:
nicht rechtseindeutig => keine Funktion
- d) Eine natürliche Zahl hat mindestens einen Teiler: **total**
Sie kann aber auch mehr als einen Teiler haben:
nicht rechtseindeutig => keine Funktion
- e) Für jedes reelle Argument kann man den Sinus berechnen: **total**
Mehrere Argumente werden auf das gleiche Bild abgebildet und nicht alle Werte werden getroffen (z.B. 2):
nicht injektiv und nicht surjektiv

Aufgabe 2.2

Behauptung:

Wenn $f : A \rightarrow W$ und $g : W \rightarrow V$ Funktionen sind, dann ist $g \circ f : A \rightarrow V$ eine Funktion

Beweis mittels Kontraposition:

Wenn $(g \circ f : A \rightarrow V \text{ keine Funktion}) \Rightarrow (f : A \rightarrow W \text{ oder } g : W \rightarrow V \text{ keine Funktion})$

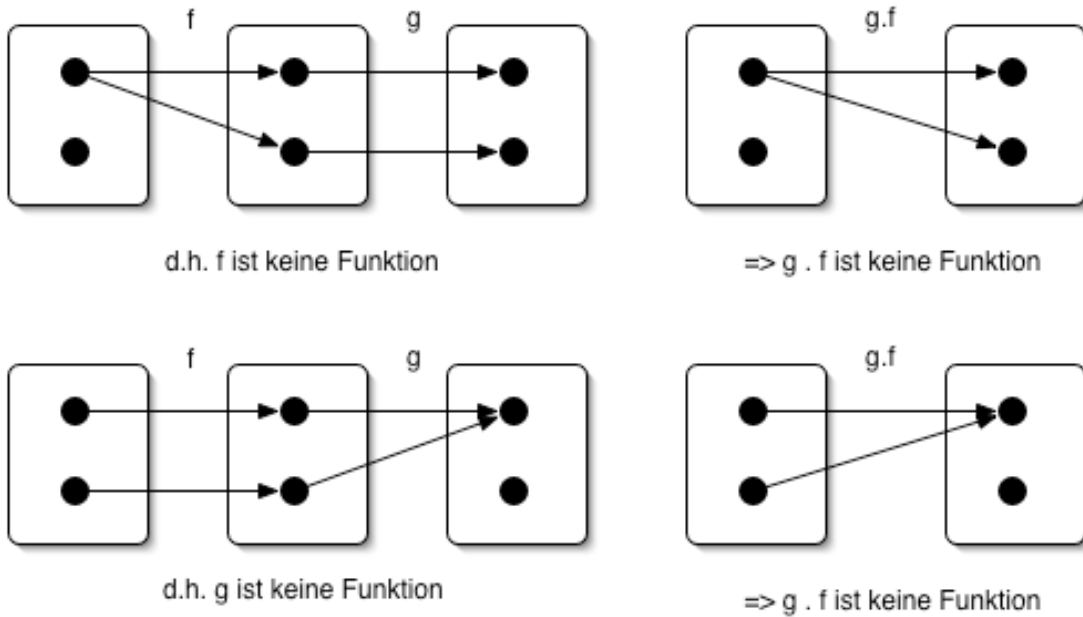
Damit die Komposition $g \circ f$ keine Funktion ist, muss gelten:

$\exists a \in A, \exists v, v' \in V$ mit $g(f(a)) = v_1$ und $g(f(a)) = v_2$

$\Leftrightarrow g \circ f$ ist nicht Rechtseindeutig

Wenn $g \circ f$ nicht Rechtseindeutig ist, gibt es genau 2 Fälle:

- f ist keine Funktion
- g ist keine Funktion



In beiden Fällen ist die resultierende Funktion $g \cdot f$ keine Funktion.

q.e.d.

a) $g \cdot f$ ist dann **total, surjektiv**

$$\begin{aligned} \text{Bsp.: } f &= \{ (a_1, w_1), (a_2, w_1) \} \\ g &= \{ (w_1, v_1) \} \\ g \cdot f &= \{ (a_1, v_1), (a_2, v_1) \} \end{aligned}$$

c) $g \cdot f$ ist dann **partiell, injektiv**.

$$\begin{aligned} \text{Bsp: } f &= \{ (a_1, w_1), (a_2, w_2) \} \\ g &= \{ (w_2, v_2), (w_3, v_3) \} \\ g \cdot f &= \{ (a_2, w_2) \} \end{aligned}$$

Aufgabe 2.3

- Der Punktoperator verbindet 2 einstellige Funktionen, ist rechtsassoziativ und bindet schwächer als Funktionen.
 $f x$ und $g y$ müssen einstellige Funktionen sein. Dann wird der Ausdruck als $(f x) \cdot (g y)$ ausgewertet.
- Funktionen binden stärker als Operatoren. Funktionen können n-stellig sein. Operatoren sind 2 stellig. Operatoren stehen in infix-Schreibweise und Funktion stehen in präfix-Schreibweise. Funktionen sind linksassoziativ und Operatoren haben Prioritätsregeln. Ein Operator in Präfixschreibweise wird als Funktion interpretiert.
- sqrt ist eine partielle Funktion, da die negativen Zahlen in \mathbb{R} nicht abgebildet werden und ein Argument nicht auf zwei Bilder abgebildet werden kann.

- d) Ein Prädikat ist eine n-stellige Funktion, die einen Wahrheitswert (True oder False) liefert.
- e) Soll die Umkehrfunktion f^{-1} total sein, so muss die Funktion f total und bijektiv sein.
Kann die Umkehrfunktion f^{-1} partiell sein, so muss die Funktion f injektiv oder bijektiv sein.
- f) Die Funktion ist **total**, da alle Argumente abgebildet werden können.
Es werden nicht alle Werte des Bildbereichs getroffen ($\sqrt{2}$) und einige Argumente werden auf das gleiche Bild abgebildet ($2^4 = (-2)^4$):
nicht injektiv und nicht surjektiv

Aufgabe 2.4

a) **Zu zeigen ist:**

$t \mid x$ und $t \mid y \Leftrightarrow t \mid (x \bmod y)$ und $t \mid y$ $t \mid x$ bedeutet t teilt x
d.h. wenn t Teiler von x und y ist, ist t auch Teiler von $x \bmod y$ und y und umgekehrt.

Vorüberlegungen

$t \mid y \Leftrightarrow y = a \cdot t$ $y, a \in \mathbb{N}$ y ist Vielfaches von t

$t \mid x \Leftrightarrow x = b \cdot t$ $x, a \in \mathbb{N}$ x ist Vielfaches von t

$x \bmod y = x - c \cdot y$ $x, y, c \in \mathbb{N}$ c ist die größte Zahl mit $c \cdot y \leq x$

Beweis:

\Rightarrow : zu zeigen ist:

$t \mid x$ und $t \mid y \Rightarrow t \mid (x \bmod y)$ und $t \mid y$

d.h. wenn t Teiler von x und y ist, ist t auch Teiler von $x \bmod y$ und y .

Gegeben:

$t \mid x$ und $t \mid y \Leftrightarrow y = a \cdot t$ und $x = b \cdot t$

Der Algorithmus wendet nun `mod` auf x und y an:

$\Rightarrow x \bmod y \Leftrightarrow a \cdot t \bmod b \cdot t$ nach Vorüberlegungen

$\Leftrightarrow a \cdot t - c \cdot b \cdot t$ nach Vorüberlegungen für hinreichend großes c

$\Leftrightarrow (a - c \cdot b) \cdot t$ mit $a - c \cdot b \geq 0$

$\Leftrightarrow x \bmod y$ ist Vielfaches von t

q.e.d.

\Leftarrow : zu zeigen ist:

$t \mid (x \bmod y)$ und $t \mid y \Rightarrow t \mid x$ und $t \mid y$

d.h. wenn t Teiler von $x \bmod y$ und y ist, ist t auch Teiler von x und y .

$$\begin{aligned} t \mid (x \bmod y) &\Leftrightarrow x \bmod y = a \cdot t && \text{nach Vorüberlegungen} \\ &\Leftrightarrow x - b \cdot y = a \cdot t && \text{nach Vorüberlegungen} \\ &\Leftrightarrow x = a \cdot t + b \cdot y && \text{durch Umformungen} \end{aligned}$$

Wir wissen zusätzlich aus der Prämisse, dass $t \mid y$

$$\begin{aligned} &\Leftrightarrow x = a \cdot t + b \cdot c \cdot t && \text{dank Prämisse} \\ &\Leftrightarrow x = t \cdot (b \cdot c + a) \\ &\Rightarrow x \text{ ist Vielfaches von } t \\ &\Rightarrow t \text{ teilt } x \Leftrightarrow t \mid x \end{aligned}$$

q.e.d.

Da \Rightarrow und \Leftarrow bewiesen wurden, gilt $t \mid x$ und $t \mid y \Leftrightarrow t \mid (x \bmod y)$ und $t \mid y$

q.e.d.

b) ggt-Algorithmus nach Euklid

ggt :: (Integral a) => a -> a -> a

-- y > 0

ggt x 0 = x

ggt x y = ggt y (x `mod` y)

```
*Programme> ggt 7888 32988
```

```
4
```

```
*Programme> ggt 9988 4444
```

```
44
```

Aufgabe 2.5

a) Maximum von 3 Zahlen

max3 :: (Ord a) => a -> a -> a -> a

max3 x y z = max x (max y z)

```
*Programme> :t max3
```

```
max3 :: forall a. (Ord a) => a -> a -> a -> a
```

```
*Programme> max3 1 2 3
```

```
3
```

b) Test, ob alle Parameter gleich sind

alleGleich :: (Eq a) => a -> a -> a -> Bool

alleGleich x y z = (x==y) && (y==z)

```

*Programme> :t alleGleich
alleGleich :: forall a. (Eq a) => a -> a -> a -> Bool
*Programme> alleGleich 1 1 1
True

```

Aufgabe 2.6

a) $(==0) . (\text{mod } 2)$

Die Funktion ist **korrekt**. Zuerst wird ``mod`2` auf ein Argument angewandt und anschließend wird das Ergebnis mit 0 verglichen. Es wird getestet, ob eine Zahl gerade ist und gibt dann `true` zurück.

```

*Programme> ((==0).( `mod ` 2)) 6
True
*Programme> ((==0).( `mod ` 2)) 3
False

```

b) $(\text{mod } 2 == 0)$

Die Funktion ist **falsch** definiert. ``mod`2` ist eine Section, die so stark wie eine Funktion und damit stärker als `==` bindet. Als nächstes Argument steht aber ein Operator. Somit kommt es zu einem Fehler.

c) $\text{flop } f \ b \ a = f \ a \ b$

Die Funktion ist **korrekt**. `flop` ist eine 3-stellige Funktion, die als Parameter eine zweistellige Funktion `f` und zwei weitere Parameter `b a` erhält. Die Parameter von `f` werden vertauscht.

```

*Programme> flop (^) 2 3
9

```

d) $(* \ \text{gcd } 8 \ 12)$

Die Funktion ist **korrekt**. Der Ausdruck ergibt eine anonyme Funktion, die auf einen Parameter angewandt werden kann. Der Parameter wird mit 4 multipliziert.

```

*Programme> (* gcd 8 12) 3
12

```

e) -

f) $f \ x \ y = (* \ \text{gcd } x \ y)$

Die Funktion ist **korrekt** definiert. Die Funktion ist 3-stellig. Es handelt sich um die verkürzte Schreibweise für $f \ x \ y \ z = (* \ \text{gcd } x \ y) \ z$

```

*Programme> f 2 4 5
10

```

g) $g \ a \ b \ c = f \ a \ b \ c + f \ a \ (b-1) \ c$

Die Funktion ist korrekt. Es werden $f(a, b, c)$ und $f(a, (b-1), c)$ addiert. Funktionsweise siehe Aufgabe f)

```
*Programme> g 20 30 10  
110
```