Funktionale Programmierung
1. Übungsblatt

28.10.17

Bearbeitet von: Nils Kerk, William Djalal

1.

a) 7^500 Das Ergebnis ist zu lang, da die Potenz zu groß ist.

b) div 5 2 = 2

c) 5 /= 5 false

d) '1' < 'a' true Da die Zahl 1 in der Haskell Reihenfolge vor dem Buchstaben a kommt.

e) -3 `mod` 5 = -3

f) 8^0 = 1

g) div 5 (-2) = -3

h) 0.9 == 3*(0.3) false Da 3*(0.3) nicht genau 0.9 beträgt.

i) mod 5 (-2) = -1

j) (-3) `mod` 5 = 2

k) 2**1023 ≈ 8.988

l) quot 5 2 = 2

m) 2^^8 = 256.0

n) rem 5 2 = 1

o) sqrt (-1) = NaN Man kann keine Wurzel aus einer negativen Zahl ziehen.

p) 2**1024 = Infinity 2**1023 ist das maximale Limit von Haskell

q) abs (-7) = 7

r) 'a' < 'b' true

s) rem 5 (-2) = 1

t) exp 1 ≈ 2.718

2.

a) (-) ((+)((+) 1 2) 3) (-2)
= (-) ((+) (3 3) (-2)
= (-) 6 (-2)
= 8

b) (-4 `mod` 5) == (-4 `rem` 5)
= -4 == -4

c) (4 `mod` (-5)) == (4 `rem` (-5))
= -1 == 4

d) 4 == (div 4 (-3)) * (-3) + (mod 4 (-3))
= 4 == (-2) * (-3) + (-2)
= 4 == 6 + (-2)
= 4 == 4

e) succ 4 * 8 == succ (4 * 8)
= 40 == succ 32
= 40 == 33

f) (10 ** 17) * ((0.1) * 3 - (0.1) * 2 - (0.1))
= (1.0e17) * ((0.3) - (0.2) - (0.1))
= 3.0e16 - (0.1)
≈ 3.0e16

g) log 0
= Infinity

3.

a) "-2" muss in Klammern stehen!

b) Man darf nicht durch die Zahl "0" dividieren!

c) 0.3 / 3 ist nie genau 0.1. Das Ergebnis ist meistens gerundet!

d) *int ist nicht der passende Datentyp für die Zahl "1.0"!