HTWK Leipzig, Fakultät IM

Prof. Dr. Sibylle Schwarz `sibylle.schwarz@htwk-leipzig.de`

10.Übung zur Vorlesung „Fortgeschrittene Programmierung“
Sommersemester 2020 zu lösen bis 24. Juni 2020

Aufgabe 10.1 (innermost / outermost Redexe)

Geben Sie für jeden der folgenden Ausdrücke `ti` zu jedem Redex darin an

- den Redex,
- seine Position,
- ob er innermost ist,
- ob er outermost ist,
- den Ausdruck `ti'`, der bei Reduktion des innermost Redex aus `ti` entsteht, und die Positionen aller Redexe in `ti'`,
- den Ausdruck `ti''`, der bei Reduktion des outermost Redex aus `ti` entsteht, und die Positionen aller Redexe in `ti''`

```
t1 = 1 + (3 * 5)
t2 = (1 + 4) * (3 + 5)
t3 = \ x -> ( 2 * 3) + (x * 5)
t4 = fst (1 + 2 , 3 + x)
t5 = (\ x -> 1 + x) (2 * 3)
```

Aufgabe 10.2 (Lazy Evaluation, schriftlich)

Berechnen Sie `sum $ map ( \ n -> n * n ) $ take 3 nats`
durch schrittweise Reduktion in Reihenfolge der Bedarfsauswertung.

Aufgabe 10.3 (Binärzahlen)

Binärdarstellungen der natürlichen Zahlen lassen sich durch den folgenden Datentyp repräsentieren:
`data B = Zero | Even B | Odd B`
Dabei entsprechen
`Zero, Even Zero, Odd Zero,  Even (Even Zero), Even (Odd Zero), Odd (Even Zero), ...`
den Binärdarstellungen $\varepsilon, 0, 1, 00, 10, 01, \ldots$ und damit den natürlichen Zahlen $0, 0, 1, 0, 2, 1, \ldots$
(Achtung: Reihenfolge der Bits, hilfreich zum Verständnis: Evaluierungsfunktion im Autotool).

a. Welche natürlichen Zahlen repräsentieren:
```
Odd (Even (Even (Even (Odd Zero))))
Even (Odd (Even (Odd (Even (Odd Zero)))))
Odd ( Even ( Odd  ( Even ( Odd ( Odd ( Odd ( Even ( Even (Zero ))))))))
```
Begründen Sie jeweils durch Angabe des Berechnung.

b. Definieren Sie eine Funktion `succB :: B -> B` , welche den Nachfolger berechnet. (Autotool)

c. Definieren Sie eine Funktion `plusB :: B -> B -> B` zur Addition zweier Binärzahlen.

d. Definieren Sie eine Funktion `mal7B :: B -> B` zur Berechnung des Siebenfachen.

e. Definieren Sie eine geeignete Version von `fold` für diesen Datentyp.

f. Definieren Sie mit Hilfe dieser Funktion `fold` die Transformation `toInt :: B -> Int`, welche jeder Binärzahl ihren Wert zuordnet.