Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Lars Noschinski, Dr. Jasmin Blanchette, Dmitriy Traytel

WS 2012/13
16. 10. 2012
Abgabe: 23. 10. 2012, 15:30

Einführung in die Informatik 2
1. Übung

Aufgabe G1.1 Auswertung

- Definieren Sie eine Funktion

  ```
  threeDifferent :: Integer -> Integer -> Integer -> Bool
  ```

  die genau dann `True` zurückgibt, wenn alle Parameter unterschiedlich sind.
- Definieren Sie eine Funktion

  ```
  fourEqual :: Integer -> Integer -> Integer -> Integer -> Bool
  ```

  die genau dann `True` zurückgibt, wenn alle Parameter gleich sind.
- Geben Sie zu den beiden folgenden Ausdrücken je eine zeilenweise Auswertung an:

  ```
  threeDifferent (2+3) 5 (11 `div` 2)
  fourEqual (2+3) 5 (11 `div` 2) (21 `mod` 11)
  ```

Aufgabe G1.2 Einfache Rekursion

- Schreiben Sie eine rekursive Funktion `fac :: Integer -> Integer`, die die Fakultät berechnet.
- Schreiben Sie eine Funktion `sum_ten :: Integer -> Integer`, die für den Parameter n die Summe $\sum_{i=n}^{n+9} i$ berechnet.

  Tipp: Gegebenenfalls ist eine Hilfsfunktion nützlich.

Aufgabe G1.3 Maximum

Gegeben sei eine Funktion `g` mit

```
g :: Integer -> Integer
g n = if n < 10 then n*n else n
```

- Definieren Sie eine rekursive Funktion `max_g :: Integer -> Integer`, so dass `max_g n` die Zahl $0 \leq$ `i` $\leq$ `n` ausgibt, für die `g i` am größten ist. Schreiben Sie diese Funktion so, also ob Sie nichts über `g` wüssten.

  Die Funktion `max` aus der Standardbibliothek könnte hilfreich sein.
- Schauen Sie sich die Funktion `g` an und überlegen Sie, wann `max_g n` $\neq$ `n` gilt. Verwenden Sie dies um einen Quickcheck-Test für `max_g` zu schreiben.

Aufgabe H1.1 Summe der Quadrate (6 Punkte, Wettbewerbsaufgabe)

Definieren Sie eine Funktion `sum_max_sq :: Integer -> Integer -> Integer -> Integer`, die aus den drei Eingabeparametern die beiden Größten bestimmt und die Summe ihrer Quadrate zurückliefert. Zum Beispiel:

$$\begin{array}{lcll} \texttt{sum\_max\_sq}\ 1\ 2\ 3 & = & 2^2 + 3^2 & = & 13 \\ \texttt{sum\_max\_sq}\ 1\ (-2)\ 5 & = & 1^2 + 5^2 & = & 26 \\ \texttt{sum\_max\_sq}\ 2\ 2\ 2 & = & 2^2 + 2^2 & = & 8 \end{array}$$

Die kürzeste Lösung bzgl. Anzahl der Tokens gewinnt! Tokens sind Identifier, Operatoren, Klammern, Zahlen und so weiter. Die Länge von Identifiern ist egal. Die vollständige Lösung (außer Funktionen aus der Standardbibliothek) muss innerhalb von Kommentaren `{-WETT-}` und `{-TTEW-}` abgegeben sein, um zu zählen. Zum Beispiel:

```
{-WETT-}
sum_max_sq :: Integer -> Integer -> Integer -> Integer
sum_max_sq x y z = ...
{-TTEW-}
```

Aufgabe H1.2 Rekursion (8 Punkte)

Sei `f :: Integer -> Integer` wie folgt definiert:

$$\texttt{f}\ n = \begin{cases} n - 10 & \text{if } n > 100 \\ \texttt{f}\ (\texttt{f}\ (n + 11)) & \text{sonst} \end{cases}$$

1. Übertragen Sie die Funktionsdefinition nach Haskell.
2. Evaluieren Sie Ihre in 1 definierte Funktion mit den Parametern $0, 1, 2, 3, 42, 101, 1001, -10$.
3. Geben Sie, basierend auf Ihren Beobachtungen in 2, eine alternative Definition von `f` an und überprüfen Sie Ihre Vermutung mit Quickcheck.

Aufgabe H1.3 Quadrate (6 Punkte)

Gesucht ist eine Funktion `is_square :: Integer -> Bool`, die für eine nicht-negative Zahl n genau dann `True` berechnet, wenn n eine Quadratzahl ist (d.h. es existiert ein $m \in \mathbb{N}_0$ mit $n = m^2$).

1. Definieren Sie eine rekursive Funktion `is_square' :: Integer -> Integer -> Bool` mit `is_square' n m = True` genau dann, wenn $n = i^2$ für ein $0 \leq i \leq m$ gilt.
2. Benutzen Sie `is_square'` um `is_square` zu definieren.