Universität Bremen

Praktische Informatik 3 WS 20/21

Christoph Lüth
Thomas Barkowsky
Tobias Brandt
Alexander Krug
Tarek Soliman
Robert Sachtleben

# 1. Übungsblatt

**Ausgabe:** 09.11.20
**Abgabe:** 16.11.20 12:00

## 1.1 *Mein Hut, der hat drei Ecken.* *5 Punkte*

1. Implementieren Sie eine Funktion, die für ein Argument $n$ eine Zeichenkette zurückgibt, die ein "Dreieck" aus Sternchen beschreibt:

   ```
   triangle :: Int→ String

   triangle 5 ⇝ "*\n**\n***\n****\n*****\n"
   ```

   Beachten Sie die Zeilenvorschübe. Wenn Sie die zurückgegebene Zeichenkette auf der Kommandozeile mit putStr ausgeben, sieht das so aus:

   ```
   putStrLn (triangle 5) ⇝
   *
   **
   ***
   ****
   *****
   ```

2. Warum sollen die Dreiecke immer linksbündig sein? Implementieren Sie eine Funktion

   ```
   triangle2 :: Int→ String
   ```

   die ein Dreieck gleicher Größe rechtsbündig ausgibt:

   ```
   putStrLn (triangle2 5) ⇝
       *
      **
     ***
    ****
   *****
   ```

   *Hinweis:* Sie werden hierfür eine Hilfsunktion benötigen, welche alle Zeilen bis zur $m$-ten Zeile erzeugt; diese besteht dann aus $n-m$ Leerzeichen, gefolgt von $m$ Sternchen. Die Hilfsfunktion ist rekursiv, die Funktion `triangle2 n` ruft dann nur die Hilfsfunktoin mit dem Argument `n` auf.

Für $n \leq 0$ sollte die leere Zeichenkette zurückgegeben werden.

## 1.2 *Fermats Testament* *5 Punkte*

Primzahlen sind immer nützlich, mit Rucola und Parmesan als leichter Salat oder zur Schlüsselgenerierung für kryptographische Verfahren. Aber wann ist eine Zahl eine Primzahl? Dazu wollen wir in dieser Übung Verfahren implementieren.

1. Schreiben Sie eine Funktion

   ```
   slowprime :: Integer→ Bool
   ```

die prüft, ob das Argumnt n eine Primzahl ist, indem es für alle Zahlen von 2 bis $\lfloor\sqrt{n}\rfloor$ prüft, ob $n$ durch diese Zahl teilbar ist. ($\lfloor\sqrt{n}\rfloor$ ist die *ganzzahlige Wurzel* aus $n$, und ist mit dem Namen `isqrt` in der Vorlage zum Aufgabenblatt vorgegeben). Hierbei ist $a$ durch $b$ teilbar, wenn der Rest der ganzzahligen Division 0 ist, d.h. $a \bmod b = 0$.

*Hinweis:* Beachten Sie bitte: 1 ist *keine* Primzahl.

2. Ihr Nachbar gibt an, dass sein Urururgroßvater der berühmte Mathematiker Pierre de Fermat war. Er (der Mathematiker, nicht ihr Nachbar) wurde berühmt, weil er der Nachwelt mit einer fadenscheinigen Begründung ("Kein Platz mehr es aufzuschreiben…") ein Theorem ohne Beweis hinterließ. Zum Glück hat Monsieur de Fermat einige seiner Theoreme tatsächlich bewiesen, darunter das kleine Fermatsche Theorem: wenn $p$ eine Primzahl ist, dann gilt für alle mit $p$ teilerfremden $a$

$$a^{p-1} \bmod p = 1 \tag{1}$$

Ihr Nachbar sagt jetzt, man müsse doch einfach nur prüfen, ob die Gleichung (1) für $a = 2$ gilt, das sei doch viel schneller als dieses ganze Gezähle. Ist dies nicht der Fall, kann $p$ keine Primzahl sein, aber stimmt das auch andersherum — d.h. wenn die Gleichung gilt, ist $p$ dann auch eine Primzahl?

Schreiben Sie eine Funktion

```
fastprime :: Integer→ Bool
```

die nach dem Fermatschen Verfahren prüft, ob n eine Primzahl ist, d.h. ob $2^{(n-1)} \bmod n = 1$ gilt.

3. Jetzt wird sich zeigen, ob Ihr Nachbar recht hat. Schreiben Sie eine Funktion

```
count_false :: Integer→ Integer
```

die zählt, für wieviele Zahlen von $1 \ldots n$ der Fermatsche Test ein falsches Ergebnis liefert.

*Hinweis:* Ihr Nachbar hat *nicht* recht (er ist auch sonst ein ziemlicher Angeber, hat zu allem eine Meinung aber von nichts viel Ahnung und einen HSV-Sticker an seinem SUV): von $1 \ldots 100$ gibt es eine, und für $1 \ldots 1000$ vier Abweichungen (welche?).[1]

Sie können annehmen, dass die Argumente Ihrer Primzahltests nicht negativ sind.

[1] Zur Ehrenrettung von Fermat sei gesagt, dass es auf diesem Test aufbauende Verfahren wie den Miller-Rabin-Test gibt, die wesentlich robuster sind.