

Übungen zu Einführung in die Informatik I

Aufgabe 36 Nichtstrikte Funktionen

Betrachten Sie folgende boolesche Ausdrücke und die zugehörigen Resultate:

```
# true || (1=1/0) ;;  
- : bool = true  
# false || (1=1/0) ;;  
Exception: Division_by_zero .  
# false && (1/0=2/0) ;;  
- : bool = false
```

Der bedingte Ausdruck **if ... then ... else ...** kann als dreistellige Funktion betrachtet werden. Betrachten Sie unter diesem Gesichtspunkt folgende Ausdrücke:

```
# if true then 1 else 2/0 ;;  
- : int = 1  
# if false then 1/0 else 2 ;;  
- : int = 2
```

Was folgt aus diesen Ausdrücken für die Auswertung von Konjunktion, Disjunktion und bedingten Ausdrücken in Ocaml?

Aufgabe 37 Datenströme in Ocaml

Ziel dieser Aufgabe ist es, sequenzartige Strukturen zu implementieren, deren Länge unbegrenzt ist. Beispiele derartiger Datenstrukturen sind Ein- und Ausgabeströme, die Liste der natürlichen Zahlen, etc.. Wesentliches Hilfsmittel werden dabei die Techniken zur Umsetzung von Lazy-Evaluation (siehe Blatt 8) sein.

- a) Aus Blatt 8 ist bekannt, dass durch Verwendung von Funktionen als Argument, eine Lazy-Evaluation Strategie umgesetzt werden kann. Geben Sie die Definition eines sequenzartigen Datentyps namens `'a stream` mit folgenden Eigenschaften an:

- Das erste Element der Sequenz soll gemäß „Eager-Evaluation“ berechnet werden;
- Der Rest der Sequenz soll bei Bedarf, d.h. gemäß „Lazy-Evaluation“, berechnet werden.

Warum erfüllt eine derartige Sequenz, die Anforderungen, die an Ein- und Ausgabeströme gestellt werden.

Hinweis: Eine Typausdruck kann in OCaml nicht nur Datentypen enthalten, sondern auch Funktionssignaturen. Beispielsweise wird ein Datentyp, dessen Elemente Funktionen über ganzen Zahlen sind, folgendermaßen definiert:

```
type intFun = int -> int ;;
```

- b) Implementieren Sie für den Datentyp `'a stream` die Funktionen `head` und `tail` !

- c) Geben Sie eine Funktion an, die die ersten n Elemente eines Datenstromes in eine Liste umwandelt!
- d) Erzeugen Sie einen Datenstrom für die natürlichen Zahlen (natürliche Zahlen beginnen in Informatik mit 0!)
- e) Geben Sie eine Funktion an, die zwei Ströme, ganzer Zahlen addiert. Sobald einer der Ströme endet, soll auch der „Resultatsstrom“ enden.

Aufgabe 38 Alkoholkonzentration im Blut

Die Alkoholkonzentration im Blut nimmt ohne weitere Alkoholaufnahme pro Stunde um ca. 20% ab. Die Alkoholkonzentration wird dabei nach folgender Formel (Widmark-Formel) berechnet:

$$\text{Alkohol (in Promille)} = \frac{0.8 \cdot \text{Flüssigkeit in Deziliter (100ml)} \cdot \text{Alkoholgehalt in \%}}{\text{Körpergewicht in kg} \cdot \text{Reduktionsfaktor}}$$

Der Reduktionsfaktor ist dabei 0.6 für Frauen und 0.7 für Männer. Schreiben Sie ein Java Programm das ermittelt, nach welcher Zeit die Alkoholkonzentration unter 0,3 Promille (Fahrtüchtigkeit erreicht) gefallen ist! (Hinweis: Zur Speicherung der Werte für die Berechnung des Alkoholgehalts können Sie globalen Variablen verwenden)

- a) Schreiben Sie das Java Programm Rekursiv, d. h. der für die Berechnung der Stunden relevante Code steht hinter dem `return` Statement.
- b) Schreiben Sie das Java Programm Iterativ, d. h. die Berechnung erfolgt innerhalb einer `while` Schleife.

Aufgabe 39 Primfaktorzerlegung

Jede natürliche Zahl $n \geq 2$ ist entweder selbst eine Primzahl oder lässt sich als Produkt von Primzahlen schreiben; d. h. in Primfaktoren zerlegen, z. B. $120 = 2 * 2 * 2 * 3 * 5$. In dieser Aufgabe ist ein Java-Programm zu entwickeln, das eine natürliche Zahl $n \geq 2$ in Primfaktoren zerlegt und das Ergebnis auf dem Bildschirm ausgibt; z. B. `primfak(120)` soll die Ausgabe `2 2 2 3 5` liefern.

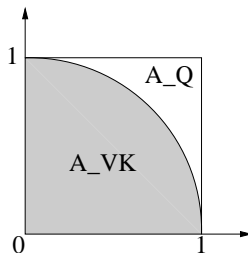
- a) Implementieren Sie eine Java-Methode `kleinsterTeiler`, die für eine natürliche Zahl n die kleinste natürliche Zahl $d > 1$ ermittelt, die n ohne Rest teilt.
- b) Implementieren Sie eine Java-Methode `primfak`, die die einzelnen Primfaktoren am Bildschirm ausgibt.

Aufgabe 40 (H) Berechnung von π nach der Monte-Carlo-Methode (5 + 5 = 10 Punkte)

Mit der Monte-Carlo-Methode kann der Wert von π angenähert werden. Dieses Verfahren setzt zufällige Punkte $P(x,y)$ in das Quadrat und stellt anschließend fest, ob der gesetzte Punkt im Viertelkreis (Treffer) liegt. $P(x,y)$ ist ein Treffer, falls $d^2 = x^2 + y^2 \leq r^2 = 1$.

Das Verfahren benutzt folgende Annahme: Die Flächen von Quadrat und Viertelkreis verhalten sich wie die jeweilige Anzahl der Zufallspunkte darin:

$$\frac{\text{treffer}}{\text{anzahl}} = \frac{A_{VK}}{A_Q} = \frac{\frac{1}{4} \cdot \pi}{1} = \frac{\pi}{4}$$



Implementieren Sie das Verfahren in Java und geben Sie den Wert von π nach 1000 Versuchen aus. (Hinweis: Zum Erstellen von Zufallszahlen können Sie die Methode `nextDouble()` aus der Klasse `java.util.Random` verwenden).

- Schreiben Sie das Java Programm Rekursiv, d. h. der für die Berechnung relevante Code steht hinter dem `return` Statement.
- Schreiben Sie das Java Programm Iterativ, d. h. die Berechnung erfolgt innerhalb einer `for` oder `while` Schleife.

Aufgabe 41 (H) Potenz und allgemeine Wurzel

(10 Punkte)

- Schreiben Sie eine iterative Java-Methode, die zu einer Fließkommazahl x und einer ganzen Zahl m , den Wert x^m berechnet.
- Schreiben Sie eine iterative Java-Methode, die für eine positive Fließkommazahl x die n -te Wurzel $\sqrt[n]{x}$, $n \in \mathbb{N}$ mittels Intervallschachtelung (Halbierungsverfahren, siehe Aufgabe 7d/e) berechnet. Die Genauigkeit soll dabei angegeben werden. (Die n -te Wurzel $y = \sqrt[n]{x}$ ergibt sich dabei als Lösung der Gleichung $y^n = x$. Bei der Lösung sollen keinerlei Funktionen aus der Bibliothek mathematischer Funktionen verwendet werden!)