Prof. Martin Hofmann, PhD
Dr. Steffen Jost

Ludwig-Maximilians-Universität München
Institut für Informatik
8. April 2013

1. Übung zur Vorlesung
Programmierung und Modellierung

A1-1 ***Kartesisches Produkt*** Berechnen Sie mit Papier und Bleistift die Menge des jeweiligen kartesischen Produktes. Wenn Sie möchten, können Sie ihre Antworten mit GHCI überprüfen.

Hinweis: im Gegensatz zu Listen ist die Reihenfolge der Auflisting ist bei Mengen natürlich unerheblich.

a) $\{11, 7\} \times \{\clubsuit, \heartsuit\}$

b) $(\{1, 2\} \times \{3, 4\}) \times \{5, 6\}$

c) $\{1, 2\} \times (\{3, 4\} \times \{5, 6\})$

d) $\{1, 2\} \times \{\}$

e) Sei A eine Menge mit n Elementen. Wie viele Elemente gibt es in $A \times \{27, 69\}$?

A1-2 ***Grundlegende Operatoren***

Die Funktionen für die logischen Operatoren UND, ODER und Negation wurden nicht in der Vorlesung erklärt. Finden Sie selbst heraus, wie diese Operatoren in Haskell heißen und testen Sie deren Funktionsweise kurz mit GHCI.

Hinweise: Die Vorlesungshomepage listet Links zur Lösung dieser Aufgabe. Zum Beispiel können Sie in der Dokumentation der Standardbibliotheken nachschlagen; grundlegende Funktionen findet man oft im Modul `Prelude` oder auch in den `Data`-Modulen, z.B. `Data.Bool`.

Die Haskell Suchmaschine Hoogle kann Bibliotheksfunktionen mithilfe des Typs schnell aufspüren; tippen Sie also alternativ den Typ der gesuchten Funktion in Hoogle ein.

Wenn Sie keinen Internetzugriff haben: Bei der Installation der Haskell Plattform sollte die Dokumentation der Standardbibliotheken auch lokal installiert worden sein. Eine weitere offline Möglichkeit zum Durchblättern der Dokumentation noch der GHCI Befehl `:browse`

A1-3 ***Pattern-Matching*** Schreiben Sie eine Funktion `myAnd`,[1] welche den Typ `(Bool,Bool) -> Bool` hat und die logische Operation "und" implementiert. Natürlich dürfen Sie dabei *nicht* die Operatoren der Standardbibliothek aus der vorherigen Aufgabe verwenden!

a) Verwenden Sie bei der Definition ausschließlich Pattern-Matching.

b) Verwenden Sie bei der Definition ausschließlich Wächter (keine konstanten Patterns).

c) Verwenden Sie weder Pattern-Matching noch Wächter.

[1] Zur Vermeindung von Namenkonflikten mit der Standardbibliothek nicht "and" verwenden

A1-4 ***List Comprehension*** Mia die modische Mathematikerin hat folgende Klamotten:

	Schwarz	Weiß	Pink	Flieder
Tops	3	3	3	4
Hosen	4	3	2	1
Schuhe	3	2	1	1

a) Mia möchte wissen, wie viele verschiedene Kombinationen Ihr damit zum Anziehen zur Verfügung stehen. Allerdings zieht Mia keine einfarbigen Outfits an; und die Farbe Ihrer Schuhe müssen zur Farbe Ihres Tops oder zu Ihrer Hose passen (gleiche Farbe). Als Mathematikerin kann Mia die Menge aller akzeptablen Kombinationen hinschreiben:

$$\Big\{(t,h,s) \;\Big|\; t \in M_\mathrm{t}, h \in M_\mathrm{h}, s \in M_\mathrm{s}, \text{ mit } \mathtt{f}(t) \neq \mathtt{f}(h) \text{ und } \big(\mathtt{f}(t) = \mathtt{f}(s) \text{ oder } \mathtt{f}(h) = \mathtt{f}(s)\big)\Big\}$$

wobei $M_\mathrm{t}, M_\mathrm{h}$ und M_s jeweils die Menge ihrer Tops, Hosen und Schuhe bezeichnen, und die Funktion `f` ein Kleidungsstück auf seine Farbe abbildet.

Typisch für Mathematiker ist diese Antwort zweifellos richtig, aber irgendwie nutzlos. Rechnen Sie die Anzahl der Kombinationen mit einer List-Comprehension fix aus!

Hinweis: Zur Vereinfachung modellieren wir Klamottenmengen durch Listen von ganzen Zahlen, wobei jeder Zahlwert für eine der möglichen Farben steht. Zum Beispiel kodieren wir M_s durch die Liste `[0,0,0,1,1,2,3]`. Listen können Elemente mehrfach enthalten; Schuhe gleicher Farbe brauchen wir ja nicht zu unterscheiden. Die Länge einer Liste berechnet man mit der Bibliotheksfunktionen `length :: [a] -> Int`

b) Mia's Freundin, Ina die Informatikerin, hat jeweils 4 schwarze Klamotten von jedem Typ. Weiterhin hat sie 3 weiße und 3 fliederfarbene Tops, eine weiße Hose und ein paar fliederfarbene Schuhe. Pinke Klamotten hat sie keine. Wie viele Kombinationen stehen Ina zur Verfügung, wenn Sie jeden Tag mindestens ein schwarzes Kleidungsstück tragen will?

c) Wie groß ist die Wahrscheinlichkeit, dass an einem Tag Mia und Ina Kleidungsstücke in der gleichen Farbkombination tragen, wenn Sie sich jeden Tag zufällig gemäß der geschilderten Regeln anziehen?

Hinweis: Verwenden Sie zum Ausrechnen der Wahrscheinlichkeit folgende Funktion; in der Vorlesung wird die Konvertierung zwischen den Zahlentypen noch behandelt:

```
intDiv :: Int -> Int -> Double
intDiv z n = (fromIntegral z) / (fromIntegral n)
```

A1-5 ***Auswertung*** Berechnen Sie möglichst mit Papier und Bleistift den Wert, zu dem der gegebene Ausdruck jeweils auswertet. Überprüfen Sie Ihr Ergebnis mit GHCI.

a) `(4 == 5.0):[]`

b) `'c':'o':'o':'l':"!"`

c) `let mond="käse" in if mond=="käse" && 1==2 then False else 1+1==2`

d)
```
[ z |c <- "grotesk", c/='k', c/='r', c/='e' && c/='s',
                     let z = if c=='o' then 'u' else c]
```

e) `[(a,b) | a<-[1..10], b<-[10..1], a/=b]`

f) `(\x->"nope!") [c | c <- "yes!"]`

H1-1 ***List Comprehension II*** **(4 Punkte)** (Geben Sie eine `.hs`-Datei als Lösung ab)
Robert der Rollenspieler möchte die Wahrscheinlichkeit berechnen, dass seine Spielfigur eine Fertigkeitsprüfung besteht, um die nächste Aufstufung seiner Spielfigur zu planen. In seinem Spiel wird eine Fertigkeitsprüfung wie folgt durchgeführt: Gegeben sind drei Attribute der Spielfigur und eine Schwierigkeit. Robert muss jeden der drei Attributswerte mit einem 20-seitigen Würfel (Zahlen von 1–20) unterwürfeln. Für jeden der drei Würfelwürfe wird die Differenz notiert, falls der Würfel über dem Attribut liegt und 0 sonst. Die Prüfung gelingt, wenn die Summe dieser abgeschnittenen Differenzen kleiner oder gleich dem Schwierigkeitswert ist. Zusätzlich gilt, dass die Prüfung immer gelingt, wenn mindestens zwei der drei Würfelwürfe eine 1 ergeben haben; aber die Prüfung misslingt auch schon, falls 2×20 fällt.

Robert hat leider fast alles vergessen, was er in der Schule über Wahrscheinlichkeitsrechnung gelernt hat. Seine mathematisch begabte Freundin Mia ist über das Wochenende leider zum Shopping verreist. Er weiß allerdings noch, dass er für die Berechnung der Wahrscheinlichkeit lediglich die Anzahl der günstigen Möglichkeiten durch die Anzahl aller Möglichkeiten für den Ausgang der 3 Würfelwürfe teilen muss. Da sich Robert mit Haskell's List Comprehensions auskennt, kann er sein Problem schnell lösen! "Schnell" bedeutet für Robert hier primär, dass er schnell ein korrektes Programm hingeschrieben hat. Ob die Berechnung dann 0.25 oder 2 Sekunden dauert, ist in seinem Anwendungsfall völlig egal.

a) Definieren Sie eine Funktion
```
erfolg :: (Int,Int,Int) -> Int -> (Int,Int,Int) -> Bool
```
welche als erstes Argument ein Tripel von Attributen, als zweites Argument einen Schwierigkeitswert und als drittes Argument das Ergebnis von drei Würfelwürfen bekommt, und entscheidet, ob eine Fertigkeitsprüfung gemäß der angegebenen Regeln erfolgreich war oder nicht. *Beispiel:*

```
> erfolg (16,13,8) 5 (1,17,10)
False
> erfolg (16,13,8) 5 (1,17,9)
True
```

b) Definieren Sie eine Funktion `chance :: (Int,Int,Int) -> Int -> Double` welche die Wahrscheinlichkeit berechnet, dass eine Fertigkeitsprüfung bestanden wird. Sie dürfen die Funktion `intDiv` aus Aufgabe A1-4 wiederverwenden. *Beispiel:*

```
> chance (16,13,8) 5
0.518625
```

Ein Fertigkeitsprüfung mit den Attributen 16, 13 und 8 und Schwierigkeit 5 wird also mit einer Chance von ungefähr 52% bestanden.

H1-2 *Typen* (0 Punkte) (Keine Abgabe)
Welchen Typ haben folgenden Ausdrücke?

a) `['a','b','c']`

b) `('a','b','c')`

c) `[(False,[1]),(True,[(2.0)])]`

d) `([True,True],('z','o','o'))`

e) `(\x -> ('a':x,False,([x])))`

f) `[(\x y-> (x*2,y-1)) m n | m <- [1..5], even m, n <- [6..10]]`

Hinweis: Kontrollieren Sie Ihre Antworten anschließend selbst mit GHCI! Auch wenn wir Typinferenz in der Vorlesung noch nicht behandelt haben, sollten Sie in der Lage sein, die meisten dieser einfachen Aufgaben bereits alleine mit Papier und Bleistift lösen zu können. Dies Aufgabe ist eigentlich auch einfacher als Aufgabe A1-5.

H1-3 *Rekursion* (2 Punkte) (Geben Sie eine `.hs`-Datei als Lösung ab)
In der Standardbibliothek gibt es eine Funktion `replicate :: Int -> a -> [a]`, welche einen gegebenen Wert wiederholt:

```
> replicate 3 "Ha!"
["Ha!","Ha!","Ha!"]

> replicate 5 5
[5,5,5,5,5]
```

a) Implementieren Sie diese Funktion ohne Rekursion unter Verwendung von List-Comprehension!

b) Implementieren Sie diese Funktion mit Hilfe von Rekursion, ohne Verwendung von List-Comprehension!

In beiden Fällen dürfen Sie natürlich keine Funktionen der Standardbibliothek verwenden; abgesehen von den Grundoperationen `(==),(\=),(&&),(||),not,(+),(-),(*),(/),(:)`

Abgabe: Lösungen zu den Hausaufgaben können bis Dienstag, den 29.04.2013, 11:00 Uhr mit UniworX abgegeben werden.

Aufgrund des Klausurbonus müssen die Hausaufgaben von Ihnen alleine gelöst werden. Abschreiben bei den Hausaufgaben gilt als Betrug und kann zum Ausschluss von der Klausur zur Vorlesung führen.

Studenten, welche zugunsten eine Gruppenabgabe freiwillig auf den Klausurbonus aus einem Übungsblatt verzichten wollen, können die Einrichtung einer Gruppenabgabe per eMail beantragen.