STARKER START ins Studium

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Fachbereich Informatik und Mathematik
Institut für Informatik

# Vorsemesterkurs Informatik

Sommersemester 2020

## Aufgabenblatt Nr. 4a

## Aufgabe 1 (Haskell Interpreter: GHCi)

a) Starten Sie den Haskell Interpreter GHCi aus Ihrem Homeverzeichnis, indem Sie `ghci` in einer Konsole eingeben. und lassen Sie sich die Hilfe anzeigen, indem Sie im GHCi das Kommando `:help` eingeben.

b) Üben Sie das Starten und Verlassen des GHCi: Der GHCi kann mit `:quit` beendet werden.

c) Lassen Sie für die folgenden Ausdrücke, jeweils den Wert vom GHCi berechnen:

- `4+5+6`
- `3^3*2`
- `(3^3)*2`
- `3^(3*2)`
- `10 / 3`
- `10 ‘mod‘ 3`
- `10 ‘div‘ 3`
- `10 > (2*2-10)`
- `6*8+10 <= 3245/65`
- `sqrt 2`
- `123 + -81`
- `123 + (-81)`
- `2*9 > 8*3 || 4*6 >= 3*6`
- `6*10 == 5*12 && not (True /= False)`

Benutzen Sie die Pfeiltasten, um nicht jedes mal alles neu zu tippen!

d) Lassen Sie den Wert von fünf weiteren Ausdrücken berechnen, die Sie sich selbst ausgedacht haben.

e) `even` testet, ob eine Zahl gerade ist, und `odd` testet, ob eine Zahl ungerade ist. Überprüfen Sie das, indem Sie für die Zahlen $X \in \{1, 2, 3, 4, 5\}$ jeweils `odd` $X$ und `even` $X$ im GHCi berechnen lassen.

f) Geben Sie jeweils Zahlen für $X$ und $Y$ an, sodass der entsprechende Ausdruck im GHCi zu `True` auswertet:

- `even` $X$ `&& odd` $Y$
- `even` $X$ `==` `odd` $Y$
- `not (even (2 *` $X$`)) || not (odd (`$X$ `*` $Y$`))`

**Lösung**

Z.B.

```
Prelude> even 2 && odd 1
Prelude> even 1 == odd 2
True
Prelude> even 2 == odd 1
True
Prelude> not (even (2*2)) || not (odd (2*2))
```

## Aufgabe 2 (Die Frage nach dem Pfefferdieb)

Das Rätsel vom Pefferdieb ist:

> *Es gibt drei Verdächtige: Den Hutmacher, den Schnapphasen und die Haselmaus. Folgendes ist bekannt:*
>
> - *Genau einer von ihnen ist der Pfefferdieb.*
> - *Unschuldige sagen immer die Wahrheit.*
> - *Der Schnapphase sagt: „Der Hutmacher ist unschuldig.“*
> - *Der Hutmacher sagt: „Die Hasel-Maus ist unschuldig.“*
>
> *Wer ist der Pefferdieb?*

Das folgende Haskell-Programm formalisiert das bekannte Wissen mit Aussagenlogik und Wahrheitswerten.

```
hutmacher   = undefined
schnapphase = undefined
haselmaus   = undefined

genau_einer =
    (hutmacher && not schnapphase && not haselmaus)
 || (not hutmacher &&  schnapphase && not haselmaus)
 || (not hutmacher && not schnapphase &&  haselmaus)

aussage1 = schnapphase || (not hutmacher)
aussage2 = hutmacher   || (not haselmaus)

raetsel = genau_einer && aussage1 && aussage2
```

Die Werte von `hutmacher`, `schnapphase` und `haselmaus` sollen genau dann wahr sein (ihr Wert ist `True`), wenn der jeweilige der Pfefferdieb ist. Der Wert von `genau_einer` ist genau dann `True`, wenn ein einziger der Pfefferdieb ist,

Der Wert von `aussage1` ist `True`, wenn der `Schnapphase` schuldig ist, oder wenn er unschuldig ist, ebenso der Hutmacher unschuldig ist (da der Schnapphase dann die Wahrheit sagt). Analog ergibt sich der Wert von `aussage2` für die Aussage des Hutmachers über die Haselmaus.

a) Legen Sie eine Datei namens `Pfefferdieb.hs` in einem Texteditor an, die obigen Programmcode enthält.

b) Starten Sie den GHCi in einer Shell und laden Sie das Programm, indem Sie `:load Pfefferdieb.hs` im GHCi eingeben.

c) Finden Sie durch Ausprobieren heraus, wer der Pfefferdieb ist:

1. Setzen Sie anstelle von `undefined` die Wahrheitswerte `True` bzw. `False` für `hutmacher`, `schnapphase` und `haselmaus` im Programmtext ein.
2. Speichern Sie den geänderten Quelltext ab.
3. Laden Sie die den Quelltext im GHCi erneut durch Eingabe von `:reload`.
4. Lassen Sie im GHCi den Wert von `raetsel` berechnen.
5. Wenn `raetsel` zu `True` auswertet, wissen Sie, wer der Pfefferdieb ist. Wenn `raetsel` zu `False` auswertet, ändern Sie die Wahrheitswerte für `hutmacher`, `schnapphase` und `haselmaus` ab und gehen zu Schritt 2.

**Lösung**

```
> ghci
GHCi, version 7.6.3: http://www.haskell.org/ghc/  :? for help
Loading package ghc-prim ... linking ... done.
Loading package integer-gmp ... linking ... done.
Loading package base ... linking ... done.
Prelude> :load Pfefferdieb.hs
[1 of 1] Compiling Main             ( Pfefferdieb.hs, interpreted )
Ok, modules loaded: Main.
*Main> raetsel
*** Exception: Prelude.undefined
*Main> :reload
[1 of 1] Compiling Main             ( Pfefferdieb.hs, interpreted )
Ok, modules loaded: Main.
*Main> raetsel
False
*Main> :reload
[1 of 1] Compiling Main             ( Pfefferdieb.hs, interpreted )
Ok, modules loaded: Main.
*Main> raetsel
False
*Main> :reload
[1 of 1] Compiling Main             ( Pfefferdieb.hs, interpreted )
Ok, modules loaded: Main.
*Main> raetsel
False
*Main> :reload
[1 of 1] Compiling Main             ( Pfefferdieb.hs, interpreted )
Ok, modules loaded: Main.
*Main> raetsel
True
```

Der Schnapphase ist der Dieb, d.h. `raetsel` wird zu `True` auswerten, für

```
hutmacher   = False
schnapphase = True
haselmaus   = False
```

## Aufgabe 3 (Programme testen)

Auf der Webseite zum Vorkurs (`http://vorkurs.informatik.uni-frankfurt.de/`) finden Sie eine Datei `magie.hs`. Laden Sie diese herunter und laden Sie sie anschließend in den GHCi. Die Datei stellt die Funktionen `magie1`, `magie2`, `magie3`, `magie4` und `magie5` bereit. Diese erwarten eine Zeichenkette als Eingabe und liefern eine veränderte Zeichenkette. Ein Test ist z.B. `magie1 "Hallo"`.

a) Finden Sie heraus, welche der fünf Funktionen aus allen Kleinbuchstaben Großbuchstaben macht, indem Sie jede der Funktionen mit einer Zeichenkette als Argument aufrufen, die Kleinbuchstaben enthält. Z.B. `magie1 "Der Hund geht mit der Katze spazieren"`

**Lösung**

`magie4`, z.B.:

```
*Main> magie1 "Der Hund geht mit der Katze spazieren"
"der hund geht mit der katze spazieren"
*Main> magie2 "Der Hund geht mit der Katze spazieren"
"Magische Zahl: 37 Der Hund geht mit der Katze spazieren"
*Main> magie3 "Der Hund geht mit der Katze spazieren"
"Der Hund geht mit der Katze spazieren"
*Main> magie4 "Der Hund geht mit der Katze spazieren"
"DER HUND GEHT MIT DER KATZE SPAZIEREN"
*Main> magie5 "Der Hund geht mit der Katze spazieren"
"Der Hund geht mit Katzze der spazzieren"
```

b) Welche der Funktionen verdoppeln alle im String vorkommenden Zahlen? Testen Sie alle fünf Funktionen mit geeigneten Eingaben.

**Lösung**

`magie5`, z.B.:

```
*Main Data.List> magie1 "100 200 1 2 3"
"000 100 0 1 2"
*Main Data.List> magie2 "100 200 1 2 3"
"I00 II00 I II III"
*Main Data.List> magie3 "100 200 1 2 3"
"000 100 0 1 2"
*Main Data.List> magie4 "100 200 1 2 3"
"100 200 1 2 3"
*Main Data.List> magie5 "100 200 1 2 3"
"200 400 2 4 6"
```

c) Welche der Funktionen entfernen alle runden Klammern?

**Lösung**

`magie3` und `magie4`, z.B.:
```
 "200 400 2 4 6"
*Main Data.List> magie1 "((1))"
"((0))"
*Main Data.List> magie2 "((1))"
"((I))"
*Main Data.List> magie3 "((1))"
"0"
*Main Data.List> magie4 "((1))"
"1"
*Main Data.List> magie5 "((1))"
"((2))"
```

d) Welche der Funktionen ersetzen alle Fragezeichen durch Ausrufezeichen?

**Lösung**

`magie1` und `magie3`
```
*Main Data.List> magie1 "?? ?? ??"
"!! !! !!"
*Main Data.List> magie2 "?? ?? ??"
"?? ?? ??"
*Main Data.List> magie3 "?? ?? ??"
"!! !! !!"
*Main Data.List> magie4 "?? ?? ??"
"?:-)??:-)?"
*Main Data.List> magie5 "?? ?? ??"
"?? ?? ??"
```

e) Die Funktion `magie2` gibt u.a. eine „Magische Zahl“ aus. Welche Zahl ist das, d.h. wie hängt sie mit der Eingabe zusammen?

**Lösung**

Die Länge des eingegebenen Strings.

f) Eine der Funktionen `magie4` und `magie5` gibt bei Eingabe eines Strings, der nur aus einer bestimmten Primzahl kleiner als 20 besteht, ein Gedicht von Goethe aus. Welche der beiden Funktionen ist es, und welche Primzahl ist gesucht?

**Lösung**

magie5 und 17