Prof. Bernd Finkbeiner, Ph.D.
Jana Hofmann, M.Sc.
Reactive Systems Group

Programmierung 1 (WS 2020/21)
Übungsblatt H

Lesen Sie im Buch Kapitel 9.6 - 10.2.3.

Hinweis: Über Aufgaben, die mit 🤔 markiert sind, müssen Sie eventuell etwas länger nachdenken. Falls Ihnen keine Lösung einfällt - kein Grund zur Sorge. Kommen Sie in die Office Hour, unsere Tutor:innen helfen gerne.

Mathematische Prozeduren

Aufgabe H.1
Beweisen Sie die Terminierung der folgenden Relationen, indem Sie natürliche Terminierungsfunktionen angeben.

(a) $\{((x,y),(x',y')) \in (\mathbb{N} \times \mathbb{R})^2 \mid x > x' \wedge y < y'\}$

(b) $\{((x,y),(x',y')) \in (\mathbb{Z} \times \mathbb{Z})^2 \mid x + y > x' + y' \geq -150\}$

Aufgabe H.2
Sei die folgende Prozedur gegegeben:

$\mathit{forall} : \mathbb{Z} \times \mathbb{Z} \times (\mathbb{Z} \to \mathbb{B}) \to \mathbb{B}$
$\mathit{forall}\ (m, n, p) = \text{if } m > n \text{ then } 1 \text{ else if } p\ m = 0 \text{ then } 0 \text{ else } \mathit{forall}\ (m + 1, n, p)$

(a) Ist *forall* linear- oder baumrekursiv?

(b) Geben Sie die Rekursionsfunktion von *forall* an.

(c) Geben Sie eine natürliche Terminierungsfunktion für *forall* an.

Aufgabe H.3
Zeigen Sie, dass die Prozedur

$p : \mathbb{N} \to \mathbb{N}$
$p\ n = \text{if } n < 1 \text{ then } 1 \text{ else } p\ (n - 1) + 2n + 1$

die Funktion $\lambda n \in \mathbb{N}.\ (n + 1)^2$ berechnet.

Aufgabe H.4
Geben Sie eine Prozedur mit der Rekursionsfunktion $\lambda x \in \mathbb{Z}.\ \text{if } x < 1 \text{ then } \langle\rangle \text{ else } \langle x - 1, x - 1\rangle$ an. 🤔

Aufgabe H.5
Zeigen Sie, dass die Prozeduren

$p : \mathbb{Z} \to \mathbb{Z}$
$p\ x = \text{if } x < 1 \text{ then } 0 \text{ else } p\ (x - 1) + x$

und

$q : \mathbb{Z} \to \mathbb{Z}$
$q\ x = \text{if } x < 1 \text{ then } 0 \text{ else } \frac{x}{2}(x + 1)$

semantisch äquivalent sind. Gehen Sie dabei wie folgt vor:

(a) Geben Sie natürliche Terminierungsfunktionen für p und q an.

(b) Geben Sie die Ergebnisfunktion von q an.

(c) Zeigen Sie, dass die Ergebnisfunktion von q die definierende Gleichung von p für alle $s \in \mathbb{Z}$ erfüllt.

(d) Argumentieren Sie, warum die obigen Schritte zeigen, dass p und q semantisch äquivalent sind.

Aufgabe H.6
Sei die folgende Prozedur gegeben:

$p : \mathbb{Z}^2 \to \mathbb{Z}$
$p\ (x, y) = \text{if } x < y \text{ then } p\ (x, y-1) \text{ else if } x > y \text{ then } p\ (x-1, y) \text{ else } x$

(a) Geben Sie die Rekursionsfolge und die Rekursionstiefe für den Aufruf $p\ (-2, 1)$ an.

(b) Geben Sie die Rekursionsfunktion und die Rekursionsrelation von p an.

(c) Geben Sie eine natürliche Terminierungsfunktion für p an.

(d) Beschreiben Sie die Ergebnisfunktion von p ohne Rekursion.

Aufgabe H.7
Beweisen Sie, dass die Prozedur aus Aufgabe H.6 die Minimumsfunktion $\lambda(x, y) \in \mathbb{Z}^2.\ \min\{x, y\}$ berechnet.

Aufgabe H.8
Betrachten Sie die bekannte Prozedur

$\mathit{euclid} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$
$\mathit{euclid}\ (x, y) = \text{if } y = 0 \text{ then } x \text{ else } \mathit{euclid}\ (y, x \bmod y)$

Zeigen Sie, dass *euclid* und die folgende Prozedur *gcd′* semantisch äquivalent sind:

$\mathit{gcd}' : \mathbb{N} \times \mathbb{N} \to \mathbb{Z}$
$\mathit{gcd}'\ (x, y) = \text{if } x = 0 \text{ then } y \text{ else if } y = 0 \text{ then } x \text{ else } \mathit{gcd}\ (x, y)$

Dabei ist *gcd* die Ergebnisfunktion der bekannten Prozedur selben Namens aus Abbildung 9.1 des Buches, für die in Abschnitt 9.8 gezeigt wird, dass sie den größten gemeinsamen Teiler zweier positiver ganzer Zahlen berechnet.

Aufgabe H.9
Machen Sie sich mit Hilfe der folgenden Beispiele klar, dass aus der Ergebnisfunktion einer Prozedur nicht ermittelt werden kann, welchen Argument- und Ergebnisbereich die Prozedur hat und ob sie rekursiv ist.

(a) Geben Sie eine Prozedur $\mathbb{N} \to \mathbb{N}$ an, die die Ergebnisfunktion $\emptyset$ hat.

(b) Geben Sie eine linear- und eine baumrekursive Prozedur $\mathbb{N} \to \mathbb{N}$ an, die die Funktion $\lambda n \in \mathbb{N}.\ 0$ berechnet.

Aufgabe H.10
Die Quersumme einer Zahl können wir wie folgt definieren:

$$\begin{aligned} &\mathit{cross} : \mathbb{N} \to \mathbb{N} && \\ &\mathit{cross}\ n = n && \text{für } n < 10 \\ &\mathit{cross}\ n = n \bmod 10 + \mathit{cross}\ (n \operatorname{div} 10) && \text{für } n \geq 10 \end{aligned}$$

Beachten Sie, dass $n \bmod 10$ die letzte Ziffer der Zahl n liefert.

(a) Konstruieren Sie eine endrekursive Prozedur *crossi*, die die Funktion

$\mathit{crossi} \in \mathbb{N} \times \mathbb{N} \to \mathbb{N}$
$\mathit{crossi}\ (a, n) = a + \mathit{cross}\ n$

berechnet und beweisen Sie die Korrektheit Ihrer Prozedur.

Aufgabe H.11
Machen Sie sich die folgenden Sachverhalte klar:

(a) In der Relation *Ter* aus Kapitel 8.7 gilt für jeden Knoten n, dass die von ihm ausgehenden Pfade höchstens die Länge n haben.

(b) In der Relation Ter_2 aus Kapitel 9.10 gilt für jeden Knoten (x, y) mit $x > 0$, dass die Länge der von ihm ausgehenden Pfade nach oben unbeschränkt ist.

Aufgabe H.12
Zeigen oder widerlegen Sie folgende Aussage:

Für eine Relation R gibt es genau dann eine strukturelle Terminierungsfunktion wenn es eine natürliche Terminierungsfunktion gibt.

Induktive Korrektheitsbeweise

Aufgabe H.13
Beweisen Sie folgende Aussagen durch Induktion:

(a) $\forall n \in \mathbb{N} : 3n \leq 3^n$

(b) $\forall n \in \mathbb{N} : \exists k \in \mathbb{N} : n^3 - n = 3k$

Aufgabe H.14
Man kann Proposition 10.2 aus dem Buch auch ohne die Benutzung der Vertauschungseigenschaft beweisen. Dazu zeigt man die allgemeinere Aussage

$$\forall x \in \mathbb{Z} : \forall n \in \mathbb{N} : \forall s \in \mathbb{Z} : \mathit{iter}\ (n, s, \lambda a.\ a \cdot x) = s \cdot x^n.$$

Beweisen Sie diese Aussage durch Induktion.

Aufgabe H.15
Für jede Menge X seien die folgenden Prozeduren gegeben:

$$\begin{aligned}
&\mathit{fac} : \mathbb{N} \to \mathbb{N} \\
&\mathit{fac}\ 0 = 1 \\
&\mathit{fac}\ n = n \cdot \mathit{fac}\ (n-1) && \text{für } n > 0 \\
\\
&\mathit{iter} : \mathbb{N} \times X \times (X \to X) \to X \\
&\mathit{iter}\ (0, x, f) = x \\
&\mathit{iter}\ (n, x, f) = \mathit{iter}\ (n-1, f\ x, f) && \text{für } n > 0
\end{aligned}$$

Beweisen Sie, dass für die Schrittfunktion $f = \lambda(k, x).\ (k+1, k \cdot x)$ gilt:

(a) $\forall n \in \mathbb{N} : f\ (n+1, \mathit{fac}\ n) = (n+2, \mathit{fac}\ (n+1))$

(b) $\forall n \in \mathbb{N} : (n+1, \mathit{fac}\ n) = \mathit{iter}\ (n, (1, 1), f)$

Für den Beweis von (a) benötigen Sie nur die definierenden Gleichungen von f und *fac*. Der Beweis von (b) gelingt mit Induktion über n sowie Teil (a) und Proposition 10.1 aus dem Buch. Orientieren Sie sich am Beweis von Proposition 10.2 im Buch.

kNobelpreis

Sie sind an den Lösungen für kNobelaufgabe E interessiert? Der/die Aufgabensteller:in bietet ein kNobeltutorium für alle Interessierte an:

Donnerstag, 7.1. um 16:15
Zoom-Link:
https://cs-uni-saarland-de.zoom.us/j/98947936655?pwd=TXdBOXUxSmI2SzRDeWNsellnUEhGQT09

Hinweis:

(a) Der Inhalt dieser kNobelaufgabe geht über die Vorlesung und damit auch SOMSL hinaus. Zur Anfertingung Ihrer Lösung empfehlen wir daher SML Interpreter, wie Moscow ML, SML of New Jersey oder Poly/ML

(b) Dieter hat sich diese Woche wieder an Ihrem Interpreter zu schaffen gemacht.[1] Jedes mal, wenn Sie eine Ausnahme werfen, bestellt Ihr Interpreter für Dieter eine Pizza. Verzichten Sie also in der folgenden Aufgabe bitte auf die Verwendung von Ausnahmen. Achten Sie bitte darauf, weder direkt: (`raise Empty`) noch indirekt (`hd nil`) Ausnahmen zu werfen. Auch Nicht-Terminierung kann der Interpreter nun magisch erkennen. In diesem Fall benachrichtigt er umgehend den Geist von Alan Turing höchst persönlich, der Sie für den Rest Ihres Lebens verfolgen wird. Sie dürfen also zusätzlich keine Prozeduren oder Abstraktionen erzeugen, *für die es Argumente gibt*, auf denen sie divergieren.

(c) Zum Ausgleich hat Dieter sich extra die Mühe gemacht, eine SML-Vorlage für Sie zu erstellen. Sie finden diese unter Materialien. Diese Vorlage enthält bereits Typannotationen und einige Deklarationen.

Logik in SML

Es ist Donnerstagabend, 23 Uhr. Sie befinden sich gerade auf dem Weg nach Hause, natürlich in Begleitung Ihres besten Freundes Dieter Schlau, denn Sie beide möchten bei Ihnen zu Hause auf die Ankunft eines weiteren Geistes warten. Weder Sie noch Dieter wollen glauben, dass, nachdem bereits in der letzten Woche kein Geist zu Besuch kam, der Spuk ein für alle mal vorbei ist.

Auf dem Weg sind Sie in ein Gespräch über Konstruktoren und Ausnahmen vertieft und Dieter spekuliert über die Unmengen von Aufgaben, die der Assistent auf das nächste Übungsblatt packen wird. Sie biegen gerade um eine Hausecke, als Dieter sagt „Und dann, dann wird er uns einen Konstruktor geben, aus dem wir…“, nur um dann plötzlich zu verstummen. Als Sie schauen, was Ihren Freund zum Verstummen gebracht hat (Sie wissen natürlich, dass das nicht oft passiert), sind Sie mindestens ebenso erstaunt wie er. Statt in der Querstraße stehen sie plötzlich in einem Hörsaal, und nachdem eben noch weit und breit keine Menschenseele zu sehen war, ist der Hörsaal nun doch mit hunderten Studenten gefüllt. „Das sind Geister!“, konstatiert Dieter sofort mit einem geübten Blick auf deren Durchsichtigkeit.

„Ich bitte um Ruhe!“, tönt es da laut. Sie und Dieter suchen sich schnell einen Platz im beinahe vollen Hörsaal um die unheimlich geisterliche Vorlesung nicht sogleich zu stören. Sie hatten zwar eher mit einem Geist zu Hause gerechnet, aber das hier ist sicher auch nicht das Schlechteste.

„Heute möchten wir uns mit dem Curry-Howard-Isomorphismus beschäftigen, einer höchst interessanten Korrespondenz zwischen Typen und logischen Ausdrücken“. – „Psssst! Psssst! Du da! Hey, du! Ja, du! Wer ist der Typ da vorne?“, wispert Dieter seinem Nachbarn zu. „Das ist Haskell B. Curry höchstpersönlich. Sei jetzt also *bitte* still“. – „Psssst! Pssst! Du da! Hey, du! Ja, du!“, es scheint ganz so als würde Dieter Sie meinen, „ist das der Curry, von dem die kaskadierten Prozeduren kommen?“, fragt er Sie dann. „Keine Ahnung, hör ihm doch einfach zu“, antworten Sie ihm.

Aufgabe H.16 *(Bekanntes Terrain)*

(a) „Ich möchte Sie zum Beginn der Vorlesung einmal bitten, sich die folgenden logischen Ausdrücke anzuschauen und zu entscheiden, welche von ihnen beweisbar sind.“, spricht Curry vorne, um dann mit einem Schnipsen eine riesige Projektion von Schriftzeichen in die Luft zu zaubern. „Betrachten Sie Großbuchstaben als Platzhalter für logische Ausdrücke und $\to$ als Implikationen“.

(i) $\forall A, B, C.\ A \to B \to C$

(ii) $\forall A.\ A \to A$

[1] Sie sollten wohl Ihrem besten Freund Dieter nicht mehr Ihren Interpreter anvertrauen. Auch `fortytwo` hat nicht überlebt.

(iii) $\forall A, B.\ A \to B \to A$

(iv) $\forall A, B.\ A \to B \to B$

(v) $\forall A, B.\ A \to B$

(vi) $\forall A, B.\ (A \to B) \to A \to B$

(vii) $\forall A, B, C.\ (A \to B \to C) \to (A \to B) \to A \to C$

(b) Nachdem Sie und Dieter bei allen bis auf der letzten Aufgabe die selben Ergebnisse hatten und Sie sich durchgesetzt haben, ergreift der Geist vorne wieder das Wort.

„Betrachten Sie die Ausdrücke nun als Typen von SML-Prozeduren, wobei Sie die Großbuchstaben $A, B, C, \ldots$ als Ersatz für die bekannten griechischen Buchstaben $\alpha, \beta, \gamma, \ldots$ in Typangaben betrachten.

Für welche Terme können Sie geschlossene Abstraktionen angeben, die den jeweiligen Typ haben? Geben Sie diese an oder begründen Sie kurz, warum dies nicht möglich ist."

(c) Nachdem Sie auch diese Aufgabe zügig bearbeitet haben lacht Dieter laut auf. „Wie lustig, genau die Ausdrücke, die beweisbar sind, lassen sich auch in SML angeben. Der Geist hat sich ganz schön Mühe gegeben beim Aussuchen! Oder, oder… Meinst du da gibt es einen Zusammenhang?". Besteht tatsächlich ein Zusammenhang? Überlegen Sie sich eine beweisbare Aussage und geben sie dann einen SML-Ausdruck an, der genau diesen Typ hat.

Aufgabe H.17

„Sie haben nun, meine Damen und Herren, am eigenen Leib erfahren, was der Curry-Howard-Isomorphismus ist. Ihre Professoren hatten also von Anfang an Recht: Programmieren und Mathematik sind quasi das selbe! Wir wollen nun auch das logische $\wedge$ und das logische $\vee$ hinzufügen.

Wir deklarieren nun den Datentypen `Or`, so dass Sie *genau dann* einen Wert des Typs `(`α`, `β`) Or` angeben können, wenn Sie einen Wert des Typs α angeben können oder einen Wert des Typs β angeben können:

```
datatype (α, β) Or = L of α | R of β
```

Den Datentypen `And` deklarieren wir nun so, dass Sie *genau dann* einen Wert des Typs `(`α`, `β`) And` angeben können, wenn Sie sowohl einen Wert des Typs α als auch einen Wert des Typs β angeben können:

`datatype (α, β) And =` [redacted][2]

Für bessere Übersicht definieren wir außerdem Äquivalenz als

`type (α, β) equiv =` [redacted][3]

Geben Sie nun Werte an, die die Typen der folgenden Aussagen besitzen:"

Dieses Mal erscheinen die Aufgaben direkt vor Ihnen schwebend.

(a) $\forall \alpha, \beta.\ (\alpha \to \beta) \vee (\alpha \to \alpha)$

(b) $\forall \alpha, \beta, \gamma.\ (\alpha \wedge (\beta \wedge \gamma)) \leftrightarrow ((\alpha \wedge \beta) \wedge \gamma)$

(c) $\forall \alpha, \beta, \gamma.\ (\alpha \wedge \beta \to \gamma) \leftrightarrow (\alpha \to \beta \to \gamma)$

(d) $\forall \alpha, \beta, \gamma.\ (\alpha \vee (\beta \wedge \gamma)) \leftrightarrow ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$

Unter ihnen steht klein: Verwenden Sie explizite Typangaben, falls nötig.

Aufgabe H.18

„Wir können in unserer Logik auch die wahre Aussage $\top$ und die falsche Aussage $\perp$ darstellen. Auch dazu verwenden wir einfach eigene Datentypen."

(a) „Deklarieren Sie einen (möglichst einfachen) Datentyp `True` und geben Sie einen Ausdruck des Typs `True` an."

[2] Leider musste Dieter kurz husten, und Sie konnten den Geist nicht verstehen. Da Sie aber die vorangehenden Aufgabenteile schon bearbeitet haben, wissen Sie bestimmt, was er meint.

[3] Dieter hat heute ganz schlimmen Husten.

(b) „Mit der Falschheit selbst sieht die Sache schon etwas komplizierter aus. Leider weist jeder Interpreter (wieder erscheint etwas in der Luft) `datatype False = ;` ab. Deklarieren Sie trotzdem einen Datentyp `False`, so dass sie keinen Ausdruck des Typs `False` angeben können. Begründen Sie Ihre Entscheidung gegenüber Ihrem Nachbarn kurz.“ – „Uff, ich geb auf. Mach Du mal und erklär mir dann was da ab geht.“, meint Dieter und erwartet, wie üblich, dass Sie nun anfangen zu arbeiten.

Hinweis: Die zugehörige SML-Datei enthält bereits eine Definiton, die aber Mittel nutzt die in der Vorlesung bisher noch nicht eingeführt wurden. Falls Ihnen keine eigene einfällt, können Sie diese verwenden um die weiteren Aufgaben zu lösen. Es ist sehr viel schöner möglich, mit bereits in der Vorlesung behandelten Mitteln Falschheit zu deklarieren. Verwenden Sie dementsprechend nur in der Vorlesung behandelte Methoden.

(c) „Mh, das ist intelligent!“, stellt Dieter begeistert fest, als Sie ihm Ihren Datentyp zeigen. „Aber kannst du damit einen Ausdruck des Typs

$$\forall\alpha.\ \bot \to \alpha$$

angeben? Schließlich lässt sich aus Falschem alles folgern.“

Deklarieren Sie eine *Prozedur* `exfalso` mit obigem Typschema. Beachten Sie bitte weiterhin den Inhalt des aller ersten Hinweises.

(d) „Wir definieren Negation wie folgt:“

$$\texttt{type}\ \alpha\ \texttt{neg} = \alpha \to \texttt{False}.$$

ruft Ihnen von vorne der Geist zu, „Zeit zu arbeiten! Deklarieren Sie Werte mit folgenden Typen:“

(i) $\forall\alpha,\beta.\ (\neg\alpha \land \neg\beta) \leftrightarrow \neg(\alpha \lor \beta)$

(ii) $\forall\alpha,\beta.\ (\neg\alpha \lor \neg\beta) \to \neg(\alpha \land \beta)$

Aufgabe H.19 *(“to be or not to be”)*
„Wie Sie vielleicht bemerkt haben, handelt es sich bei der letzten Aussage nur um eine Implikation. Sie fragen sich nun vermutlich, warum nicht auch die Rückrichtung zeigen? Versuchen Sie sich ruhig daran, dann werden Sie schnell merken warum.“

Nach wenigen Minuten schaut Dieter Sie entsetzt an. Die Schweißtropfen glänzen schon an seiner Schläfe. “Es klappt nicht. Die einfachsten Dinge klappen nicht mehr.” Plötzlich tönt es von vorne: „Wie ich sehe hat noch niemand eine Lösung gefunden. Das liegt daran, dass keine Lösung existiert. Wir wollen eine etwas simplere Aussage betrachten, die sich auch nicht beweisen lässt:“

Plötzlich, erscheint vor Ihren Augen folgende Aussage:

$$\forall\alpha.\ \alpha \lor \neg\alpha$$

(a) Betrachten Sie $\forall\alpha.\ \alpha \lor \neg\alpha$. Der Geist behauptet, für diese Aussage gibt es keinen entsprechenden Ausdruck in SML. Begründen Sie, warum er Recht hat! Eine Intuition ist hier völlig ausreichend. Der Geist erwartet keinen Beweis oder eine ausführliche Erklärung.

(b) „Aber es ist noch nicht alles verloren, zumindest diese wahre Aussage lässt sich noch zeigen. Vorschläge?“ Beweisen Sie

$$\forall\alpha.\ (\alpha \lor \neg\alpha) \to (\neg\neg\alpha \to \alpha)$$

Hinweis: Ab hier können nur noch Bonusdieter verdient werden. Die korrekte Bearbeitung aller vorherigen Aufgaben bringt Ihnen bereits 10 Punkte ein.

Aufgabe H.20 *(a silver lining)*
Dieter meldet sich und trägt Ihre Lösung vor. Der Geist ist ganz begeistert: „Sehr gut! Ich sehe in Ihnen steckt ein wahrer Logiker und zugleich ein SML-Profi. Wenn Sie etwas weiter grübeln, werden Sie feststellen, dass sich diese Aussagen auch nicht beweisen lassen:“

An der Tafel erscheinen plötzlich, folgende Aussagen:

- $\forall\alpha.\ \alpha \lor \neg\alpha$

- $\forall \alpha. \neg\neg\alpha \to \alpha$
- $\forall \alpha, \beta. (\neg\alpha \to \neg\beta) \to (\beta \to \alpha)$
- $\forall \alpha, \beta. ((\alpha \to \beta) \to \alpha) \to \alpha$

„Dies mag Ihnen sehr schade vorkommen. Doch erinnern wir uns an unsere Aussage von eben zurück:

$$\forall \alpha. (\alpha \vee \neg\alpha) \to (\neg\neg\alpha \to \alpha)$$

Unter Annahme von $\forall \alpha. \alpha \vee \neg\alpha$ reicht es plötzlich die Doppelnegierung zu zeigen. Versuchen Sie sich also an folgenden Doppelnegierungen:“

Langsam, Stück für Stück erscheinen die folgenden Aussagen nur Zentimeter vor Ihrem Gesicht. Beweisen Sie sie:

(a) $\forall \alpha. \neg\neg(\alpha \vee \neg\alpha)$

(b) $\forall \alpha. \neg\neg(\neg\neg\alpha \to \alpha)$

(c) $\forall \alpha, \beta. \neg\neg((\neg\alpha \to \neg\beta) \to (\beta \to \alpha))$

(d) $\forall \alpha, \beta. \neg\neg(((\alpha \to \beta) \to \alpha) \to \alpha)$

Hinweis: Versuchen Sie ruhig, die ursprünglichen Aussagen zu beweisen. Ruhm und Ehre winken, falls Sie es in diesem Kalkül schaffen sollten.

„Sehr beeindruckend, sehr beeindruckend. Ich sehe ich habe würdige Nachfolger gefunden. Viel Erfolg in Ihrem weiteren Studium.“

Der Geist klatscht einmal in die Hände und ganz plötzlich sitzen Sie statt auf einem Stuhl in einem Hörsaal auf der Bank einer Bushaltestelle. Nur Dieter sitzt immer noch neben Ihnen und an den Falten auf seiner Stirn können Sie erkennen, dass er so heftig über die Aufgabe nachdenkt, dass er noch nicht einmal gemerkt hat, dass Sie wieder zurück sind. Sie lehnen sich also entspannt zurück und denken ebenfalls darüber nach.

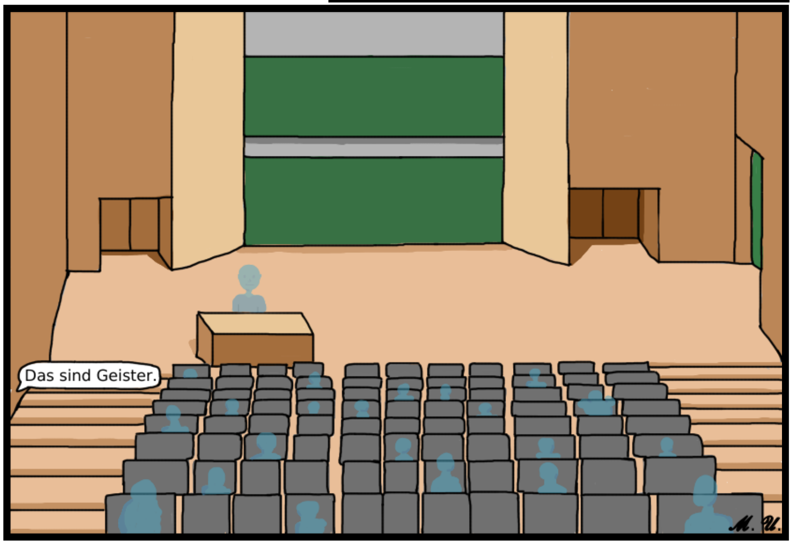
Hinweis: Wir können nun auch Aussagen zeigen, die oberflächlich keinen Sinn machen, zum Beispiel: $int \to int$. Das liegt daran, dass wir Typen wie $int, real, bool, \ldots$ nicht direkt mit Aussagen assoziieren. In einer Programmiersprache mit einem mächtigeren Typsystem als SML könnten dort durchaus Aussagen stehen, wie $4 \leq 5$.

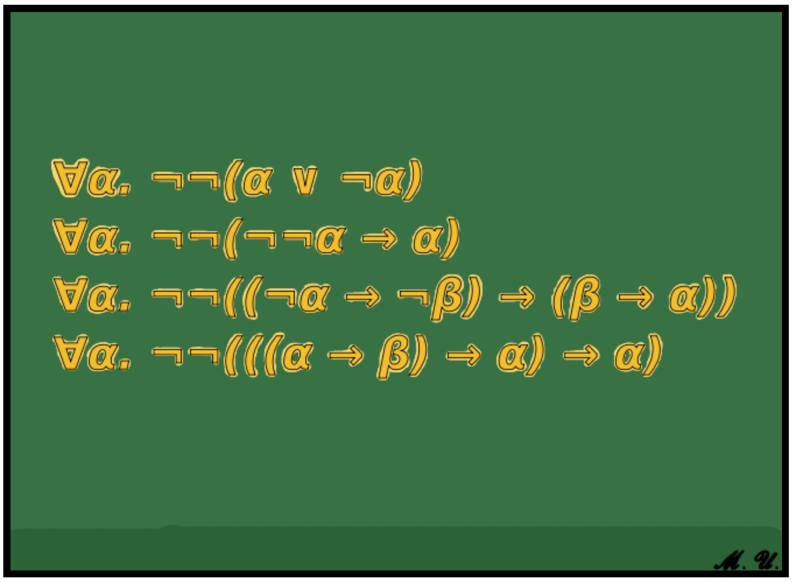
Es ist Donnerstagabend, 23 Uhr. Sie befinden sich gerade auf dem Weg nach Hause, natürlich in Begleitung Ihres besten Freundes Dieter Schlau, denn Sie beide möchten bei Ihnen zu Hause auf die Ankunft eines weiteren Geists warten. Weder Sie noch Dieter wollen glauben dass, nachdem bereits in der letzten Woche kein Geist zu Besuch kam, der Spuk ein für alle mal vorbei sei.

Auf dem Weg sind Sie in ein Gespräch über Konstruktoren und Ausnahmen vertieft und Dieter spekuliert über die Unmengen von Aufgaben, die der Assistent auf das nächste Übungsblatt packen wird. Sie biegen gerade um eine Hausecke, als Dieter sagt „Und dann, dann wird er uns einen Konstruktor geben, aus dem wir…“, nur um dann plötzlich zu verstummen. Als Sie schauen, was Ihren Freund zum Verstummen gebracht hat (Sie wissen natürlich, dass das nicht oft passiert), sind Sie mindestens ebenso erstaunt wie er.

Statt in der Querstraße stehen sie plötzlich in einem Hörsaal, und nachdem eben noch weit und breit keine Menschenseele zu sehen war, ist der Hörsaal nun doch mit hunderten Studenten gefüllt. „Das sind Geister!“, konstatiert Dieter sofort mit einem geübten Blick auf deren Durchsichtigkeit.
„Ich bitte um Ruhe!“, tönt es da laut. Sie und Dieter suchen sich schnell einen Platz im beinahe vollen Hörsaal um die unheimlich geisterliche Vorlesung nicht sogleich zu stören. Sie hatten zwar eher mit einem Geist zu Hause gerechnet, aber das hier ist sicher auch nicht das Schlechteste.

„Heute möchten wir uns mit dem Curry-Howard-Isomorphismus beschäftigen, einer höchst interessanten Korrespondenz zwischen Typen und logischen Ausdrücken“.





„Sehr gut! Ich sehe in Ihnen steckt ein wahrer Logiker und zugleich ein SML-Profi. Wenn Sie etwas weiter grübeln, werden Sie feststellen, dass sich diese Aussagen auch nicht beweisen lassen:“
„Dies mag Ihnen sehr schade vorkommen. Doch erinnern wir uns an unsere Aussage von eben zurück:

$$\forall\alpha.\,(\alpha \vee \neg\alpha) \to (\neg\neg\alpha \to \alpha)$$

Unter Annahme von $\forall\alpha.\,\alpha \vee \neg\alpha$ reicht es plötzlich die Doppelnegierung zu zeigen. Versuchen Sie sich also an folgenden Doppelnegierungen:“