
Aufgabe: *Eine Logelei*

Die folgende Aufgabe ist dem Buch

99 Logeleien von Zweistein

entnommen, das 1968 von *Thomas von Randow* unter dem Pseudonym "Zweistein" im Verlag *Christian Wegner* veröffentlicht worden ist.

Die Herren Amann, Bemann, Cemann und Demann heißen — nicht unbedingt in derselben Reihenfolge — mit Vornamen Erich, Fritz, Gustav und Heiner. Sie sind alle mit genau einer Frau verheiratet. Außerdem wissen wir über sie und ihre Ehefrauen noch dies:

1. Entweder ist Amanns Vorname Heiner, oder Bemanns Frau heisst Inge.
2. Wenn Cemann mit Josefa verheiratet ist, dann — und nur in diesem Falle — heisst Klaras Mann nicht Fritz.
3. Wenn Josefass Mann nicht Erich heisst, dann ist Inge mit Fritz verheiratet.
4. Wenn Luises Mann Fritz heisst, dann ist der Vorname von Klaras Mann nicht Gustav.
5. Wenn die Frau von Fritz Inge heisst, dann ist Erich nicht mit Josefa verheiratet.
6. Wenn Fritz nicht mit Luise verheiratet ist, dann heisst Gustavs Frau Klara.
7. Entweder ist Demann mit Luise verheiratet, oder Cemann heisst Gustav.

Wie heissen die Herren mit vollem Namen, wie ihre Ehefrauen mit Vornamen?

Verwenden Sie zur Lösung des Problems die Vorlage, die Sie im Netz unter der Adresse

github.com/karlstroetmann/Logik/blob/master/Aufgaben/Blatt-08/logelei-frame.stlx

finden. Dieser Rahmen enthält (über geeignete load-Befehle) die Implementierung der Funktion

`davisPutnam(Clauses, Literals)`,

die für eine gegebene Menge von Klauseln eine Lösung berechnet. Eine Lösung ist dabei eine aussagenlogische Belegung, die allerdings als Menge sogenannter Unit-Literale dargestellt wird. Beispielsweise wird die Belegung

$\{\langle p, \text{true} \rangle, \langle q, \text{false} \rangle, \langle r, \text{true} \rangle\}$

als die Menge

$\{\{p\}, \{\neg q\}, \{r\}\}$

dargestellt. Beim Aufruf der Funktion `davisPutnam` gilt für den Parameter *Literals* immer *Literals* = `{}`. Der Wert des Parameter *Literals* ist nur bei rekursiven Aufrufen von `{}` verschieden. Außerdem steht eine Funktion

`normalize(F)`

zur Verfügung, die eine aussagenlogische Formel *F* in konjunktive Normalform überführt.

Bei der Lösung dieser Aufgabe sollten Sie die folgenden Terme als aussagenlogische Variablen verwenden:

1. $@Name(x, y)$ ist genau dann wahr, wenn x der Vorname und y der Nachname einer der Herren ist. Beispielsweise ist

$@Name("Heiner", "Amann")$

genau dann wahr, wenn Herr Amann mit Vornamen *Heiner* heisst.

2. $@Ehe(x, y)$ ist genau dann wahr, wenn der Mann, der mit Vornamen x heisst, mit der Frau, die mit Vornamen y heisst, verheiratet ist. Beispielsweise wäre

$@Ehe("Heiner", "Luise")$

genau dann wahr, wenn Heiner mit Luise verheiratet ist.

Zur Lösung des Problems sollten Sie die folgenden Teilaufgaben bearbeiten:

- (a) Implementieren Sie eine Funktion `atMostOne`. Für eine Menge S von aussagenlogischen Variablen soll der Aufruf `atMostOne(S)` eine Menge von Klauseln berechnen, die ausdrückt, dass höchstens eine der Variablen in S wahr ist.
- (b) Implementieren Sie eine Funktion `atLeastOne`. Für eine Menge S von aussagenlogischen Variablen soll der Aufruf `atLeastOne(S)` eine Menge von Klauseln berechnen, die ausdrückt, dass mindestens eine der Variablen in S wahr ist.
- (c) Implementieren Sie eine Funktion `exactlyOne`. Für eine Menge S von aussagenlogischen Variablen soll der Aufruf `exactlyOne(S)` eine Menge von Klauseln berechnen, die ausdrückt, dass genau eine der Variablen in S wahr ist.
- (d) Implementieren Sie eine Funktion `bijective`. Die Funktion `bijective` wird in der Form

`bijective(A, B, fct)`

aufgerufen. Hierbei sind A und B Mengen von Strings. Beispielsweise könnte A die Menge der männlichen Vornamen sein und B könnte die Menge der weiblichen Vornamen sein. f ist der Name eines Funktors. Beispielsweise könnte f den Wert "Ehe" sein. Die Funktion `bijective` berechnet eine Menge von Klauseln, die zu der folgenden Formel äquivalent ist:

$$(\forall x \in A : \exists! y \in B : f(x, y)) \wedge (\forall y \in B : \exists! x \in A : f(x, y))$$

- (e) Beachten Sie, dass bereits eine Funktion `isWifeOf` vorhanden ist. Der Aufruf

`isWifeOf(y, z)`

berechnet eine Formel, die ausdrückt, dass y die Frau von z ist. Hierbei ist z der Nachname eines Mannes. Diese Formel wird als String dargestellt.

- (f) Vervollständigen Sie nun die Implementierung der Funktion `computeClauses`. Wenn Sie alles richtig gemacht haben, berechnet der Aufruf der Funktion `solve` die Lösung des Problems. Diese Lösung wird dann mit Hilfe der vordefinierten Funktion `displaySolution` in lesbarer Form ausgegeben.