Fakten, Regeln, und Anfragen

- Was sind Wissensbasen?
- Wie sehen Prolog-Wissensbasen aus?
- Und was für Anfragen können wir stellen?
- Die Prologsyntax.

Eine Wissensbasis (auf deutsch)

```
toto ist ein frosch.

toto ist grün.

bodo ist ein storch.
```

Diese Wissensbasis besteht aus drei Fakten.

Der Prolog-Interpreter

Der Prologinterpreter “liest” eine Wissensbasis und beantwortet dann Anfragen zu dieser Wissensbasis.

Zum Beispiel:

```
toto ist ein frosch.
toto ist grün.
bodo ist ein storch.
```

ist toto ein frosch? $\Rightarrow$ Prolog: Ja

ist toto ein storch? $\Rightarrow$ Prolog: Nein

wer ist grün? $\Rightarrow$ Prolog: toto

Der Prolog-Interpreter

Der Prologinterpreter “liest” eine Wissensbasis und beantwortet dann Anfragen zu dieser Wissensbasis.

Zum Beispiel:

```
toto ist ein frosch.
toto ist grün.
bodo ist ein storch.
```

ist toto ein frosch? $\Rightarrow$ Prolog: Ja

ist toto ein storch? $\Rightarrow$ Prolog: Nein

wer ist grün? $\Rightarrow$ Prolog: toto

Achtung! Der Prologinterpreter weiß nur das, was in der Wissensbasis steht. Einzelne Wörter haben keine Bedeutung für Prolog.

Deswegen:

ist toto ein tier? $\Rightarrow$ Prolog: Nein

Noch eine Wissensbasis (auf deutsch)

toto ist ein frosch.

toto ist grün.

bodo ist ein storch.

alle frösche sind tiere.

alle störche sind tiere.

Diese Wissensbasis besteht aus drei Fakten und zwei Regeln.

Wie antwortet der Prologinterpreter auf die Anfrage:

ist bodo ein tier?

Die 1. Wissensbasis “auf Prolog”

toto ist ein frosch.	`ist_ein_frosch(toto).`
toto ist grün.	`ist_grün(toto).`
bodo ist ein storch.	`ist_ein_storch(bodo).`

Die 1. Wissensbasis “auf Prolog”

toto ist ein frosch.	`ist_ein_frosch(toto).`
toto ist grün.	`ist_grün(toto).`
bodo ist ein storch.	`ist_ein_storch(bodo).`

Andere Möglichkeiten:

```
frosch(toto).        f(toto).
grün(toto).          g(toto).
storch(bodo).        s(bodo).
```

Die Wissensbasis ‘kb1’

```
woman(mia).
woman(jody).
woman(yolanda).
playsAirGuitar(jody).
```

Diese Wissensbasis besteht aus vier Fakten. Sie definiert zwei Prädikate, nämlich `woman` und `playsAirGuitar`.

Anfragen “auf Prolog”:

```
woman(mia).
playsAirGuitar(mia).
woman(vincent).
man(vincent).
```

Die Wissensbasis ’kb2’

```
listensToMusic(mia).
happy(yolanda).
playsAirGuitar(mia) :- listensToMusic(mia).
playsAirGuitar(yolanda) :- listensToMusic(yolanda).
listensToMusic(yolanda) :- happy(yolanda).
```

Diese Wissensbasis besteht aus zwei Fakten und drei Regeln, bzw. fünf Klauseln.

Regeln

`:-` heißt *wenn dann*.

Die Regel `playsAirGuitar(mia) :- listensToMusic(mia).` kann so gelesen werden: *Wenn* `listensToMusic(mia)` *wahr ist, dann ist auch* `playsAirGuitar(mia)` *wahr.*

Die linke Seite einer Regel heißt Kopf (head), und die rechte Seite heißt Körper oder Rumpf (body).

Die Wissensbasis ‘kb3’

```
happy(vincent).
listensToMusic(butch).
playsAirGuitar(vincent) :- listensToMusic(vincent),
                           happy(vincent).
playsAirGuitar(butch) :- happy(butch).
playsAirGuitar(butch) :- listensToMusic(butch).
```

Die Wissensbasis ‘kb4’

```
woman(mia).
woman(jody).
woman(yolanda).

loves(vincent,mia).
loves(marcellus,mia).
loves(pumpkin,honey_bunny).
loves(honey_bunny,pumpkin).
```

Wer ist eine Frau?:

`woman(X).`

Prolog: `X = mia`

Gibt es noch andere?: `;`

Prolog: `X = jody`

Und noch andere?: `;`

Prolog: `X = yolanda`

Und noch andere?: `;`

Prolog: no

Wissensbasis ‘kb5’

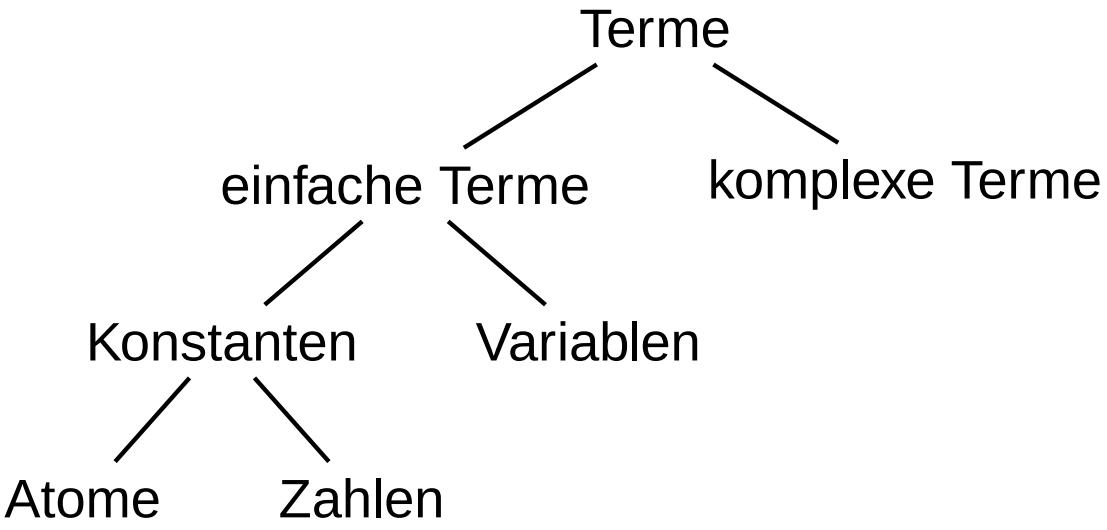
```
loves(vincent,mia).
loves(marcellus,mia).
loves(pumpkin,honey_bunny).
loves(honey_bunny,pumpkin).

jealous(X,Y) :- loves(X,Z),
                loves(Y,Z).
```

Aufgaben

1. Wie könnte man unsere 2. ‘deutsche’ Wissensbasis als Prolog-Wissensbasis hinschreiben?
2. Denkt euch eine kleine Wissensbasis aus und schreibt sie sowohl auf deutsch also auch in Prolognotation auf.
3. Erfindet 4 Anfragen, die man an diese Wissensbasis stellen könnte. (Es sollen welche dabei sein, die Prolog mit ’ja’ beantworten würde und solche die Prolog mit ’nein’ beantworten würde.)
4. Tauscht die Wissensbasis und die Anfragen (ohne die Antworten!) mit einer anderen Gruppe und überlegt euch wie Prolog auf diese Anfragen antworten würde.

Terme

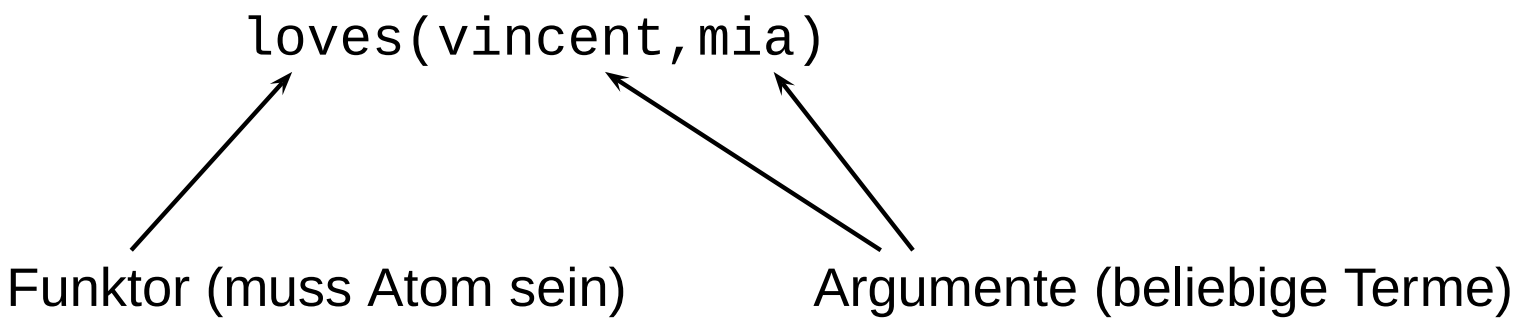


Atome `butch`, `m_monroe2`, `'Vincent'`, `'The Gimp'`, `' '`, `'@&3$'`, `;`, `:-`, `===>`

Zahlen `23`, `1001`, `0`, `-345`

Variablen `X`, `Variable`, `_variable`, `X_234`

Komplexe Terme




```
hide(X,father(father(father(butch))))
```

Komplexe Terme – Arität

Anzahl an Argument = Arität

`loves(vincent,mia)` $\longrightarrow$ Arität: 2

`hide(X,father(father(father(butch))))` $\longrightarrow$ Arität: ?

Prädikate – Arität

Welche Prädikate werden von der folgenden Datenbasis definiert?

```
loves(vincent,mia).
loves(marcellus,mia).
jealous(marcellus,vincent).
jealous(marcellus,vincent,mia).
```

Prädikate – Arität

Welche Prädikate werden von der folgenden Datenbasis definiert?

```
loves(vincent,mia).
loves(marcellus,mia).
jealous(marcellus,vincent).
jealous(marcellus,vincent,mia).
```

```
loves
jealous
jealous
```

Prädikate – Arität

Welche Prädikate werden von der folgenden Datenbasis definiert?

```
loves(vincent,mia).
loves(marcellus,mia).
jealous(marcellus,vincent).
jealous(marcellus,vincent,mia).
```

```
loves/2
jealous/2
jealous/3
```

Zusammenfassung

Wir haben gesehen wie

- Prologwissensbasen aussehen und wie
- man Anfragen stellt.

Wichtige Begriffe: Terme, Atome, Variablen, komplexe Terme (Funktor, Argumente, Arität), Fakten, Regeln (Kopf, Rumpf oder Körper), Klauseln, Prädikate.

Morgen: Wie findet Prolog die Antworten?