KI

A) Suchen in Entscheidungsbäumen

B) Spiele: Dame/Schach

C) Beweisen (Prolog)

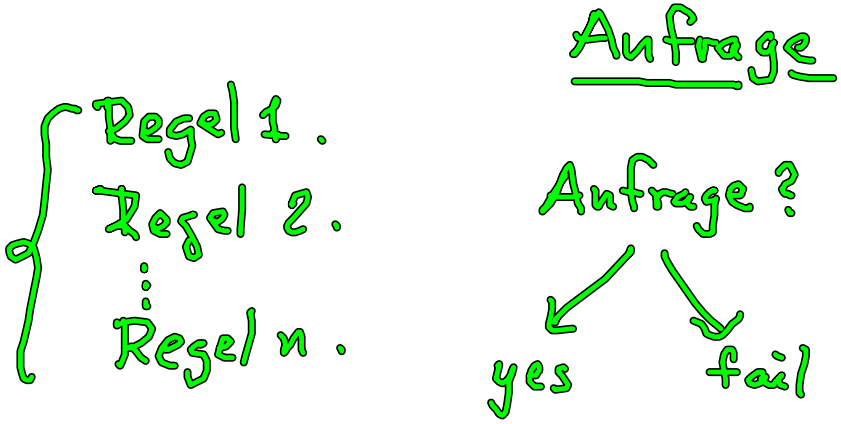
D) Terra incognita

⋮

PROLOG

Programming in Logic

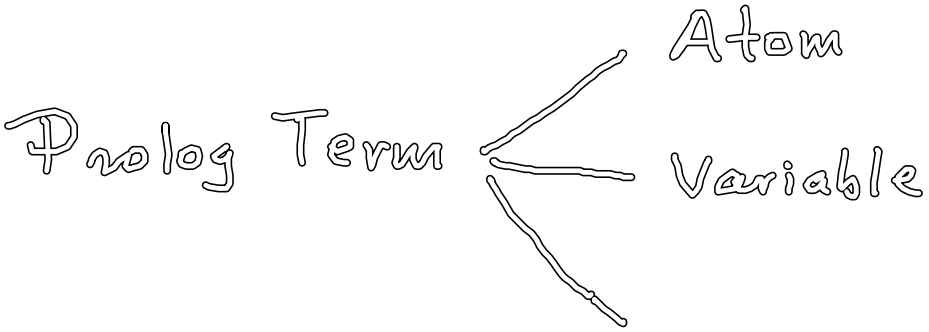
Programm in Prolog } Regelsatz



Zeichen: a ... z A ... Z

Sonderzeichen: > = == ....

Syntax



Beisp.

Atom: adam adam2 eva_one

↑

{ Kleine Buchstabe

Konstanten: 123 5 6 70

Variablen: Adam X Y X1

↑
grop

_a

↑
Variable

Strukturen:

functor(term)

functor (term, term)

:

functor (term, ..., term)

wie Atome (keine Zahlen)

~~1ab~~ (a)

a1b (a)

Bibel

Atome Strukturen

adam
eva
judas
cain
abel

vater(adam, abel)
mutter(eva, cain)

vater(X, abel)
mutter(X, Y)

name(wolfgang, mozart)

buch(name(wolfgang, mozart),
titel(don_giovanni, 17??))

Achtung:

buch((wolf, mozart), (,))

Programme

Regel ≡ Prolog Term

reine Regel { Schlussfolgerung ← Voraussetzungen

Fakten { Schlussfolgerung ← true

vater (adam, abel) ← true.
vater (adam, cain) ← true.
mutter (eva, abel) ← true.
mutter (eva, cain) ← true.
vater (cain, jakob) ← true.
vater (jakob, isaac) ← true.

? vater (adam, abel).
yes

? vater (X, abel).
!!!
vater (adam, abel)

Computer Intelligenz

arity 2

X = adam

X = adam
yes (noch was...)

? vater (X, Y).

vater (adam, abel)

X = adam
Y = abel

Choice Point

CP → Regel 1
Regel 2
Regel 3
⋮

X = adam
Y = abel

yes ;

vater (adam, cain)

X = adam
Y = cain

X = adam Y = cain

yes →

backtracking

X = cain Y = jako6

yes →

Unifikation { X = jako6 Y = isaac

yes

syntactic sugar

?- X = 1.

X = 1

yes

?- '='(X,1).

In Deiner Regel, stimmt das?

?- X = 1, X = 2.

UND

fail

$\exists x, x \in \{1\} \wedge x \in \{2\}$

?- ';'('='(X,1), '='(X,2)).

?- `` = name(Y, m ller).

X = name(Y, mueller)

yes

?- name(heinz, X) = name(Y, mueller).

X = mueller

Y = heinz

yes

Bsp:

```
vater(adam, abel).
vater(adam, cain).
.....
vater(
```

```
opa(X, Y) :- vater(X, Z), vater(Z, Y).
```

```
opa(X,Y) :- ':'(vater(X,Z), vater(Z,Y)).
':-'(opa(X,Y), .... )
```

backward chaining

```
? opa(X, jakob).
```

Klausel `opa(X, Y) :- vater(X, Z), vater(Z, Y)`

(head) Kopf — Körper (body)

$X = x \quad Y = jakob$

(*) ? `vater(X, Z), vater(Z, jakob).`

vater (adam, abel)

X = adam, Z = abel (Verzweigung bei Regel 2)

? vater (abel, jakob).

fail

(Backtracking)

vater (adam, cain)

X = adam, Z = cain

[? vater (cain, jakob).

yes

(*) yes X = adam, Z = cain

? opa (X, jakob)

X = adam

yes

?