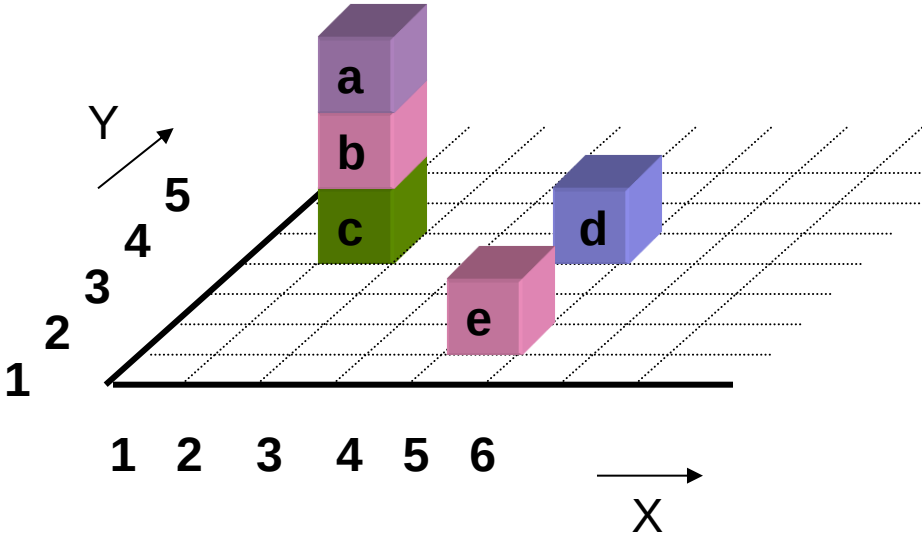
PROLOG

Ivan Bratko
Fakulteta za računalništvo in informatiko
Univerza v Ljubljani

PRIMER PROGRAMA

- Robot, ki premika kocke na mizi
- Robot vidi kocke s kamero na stropu
- Robota zanimajo koordinate kocke, ali je kocko možno zgrabiti, itd.

ROBOT'S WORLD



% see(Block, X, Y)

see(a, 2, 5).

see(d, 5, 5).

see(e, 5, 2).

% on(Block, BlockOrTable)

on(a, b).

on(b, c).

on(c, table).

on(d, table).

on(e, table).

INTERACTION WITH ROBOT PROGRAM

```
Start Prolog interpreter
?- [robot].                    % Load file robot.pl
File robot consulted
?- see( a, X, Y).              % Where do you see block a
X = 2
Y = 5
?- see( Block, _, _).          % Which block(s) do you see?
Block = a;                     % More answers?
Block = d;
Block = e;
no
```

INTERACTION, CTD.

```
?-  see( B1, _, Y),   see( B2, _, Y).     % Blocks at same Y?

% Prolog's answers may surprise!

% Perhaps this was intended:
?- see( B1, _, Y),   see( B2, _, Y),   B1 \== B2.
```

X, Y koordinate

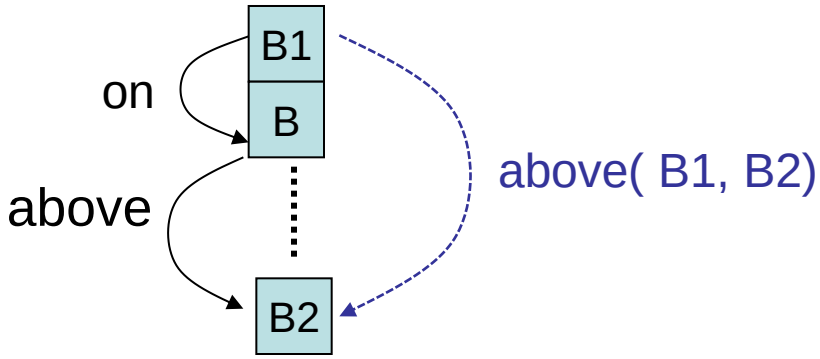
```
% xy( Block, X, Y): X, Y coord. of Block

xy( B, X, Y)  :-
  see( B, X, Y).

xy( B, X, Y)  :-
  on( B0, B),
  xy( B0, X, Y).      % Blocks in stack have the same xy-coord.
```

Relacija above(Block1, Block2)

% above(Block1, Block2): Block1 is above Block2 in same stack

```
above( B1, B2) :-
   on( B1, B2).
```


```
above( B1, B2) :-
  on( B1, B),
  above( B, B2).
```



DEKLARATIVNI POMEN TEGA POGRAMA

DANI AKSIOMI

```
on( a, b).
on( b, c).
on( c, table).
...

above( B1, B2)  :-
   on( B1, B2).

above( B1, B2)  :-
  on( B1, B),
  above( B, B2).
```

KAJ LAHKO IZPELJEMO IZ AKSIOMOV?

```
on( a, b).
on( b, c).
...

above( a, b).
above( b, c).
above( a, c).
above( a, table).
...
```

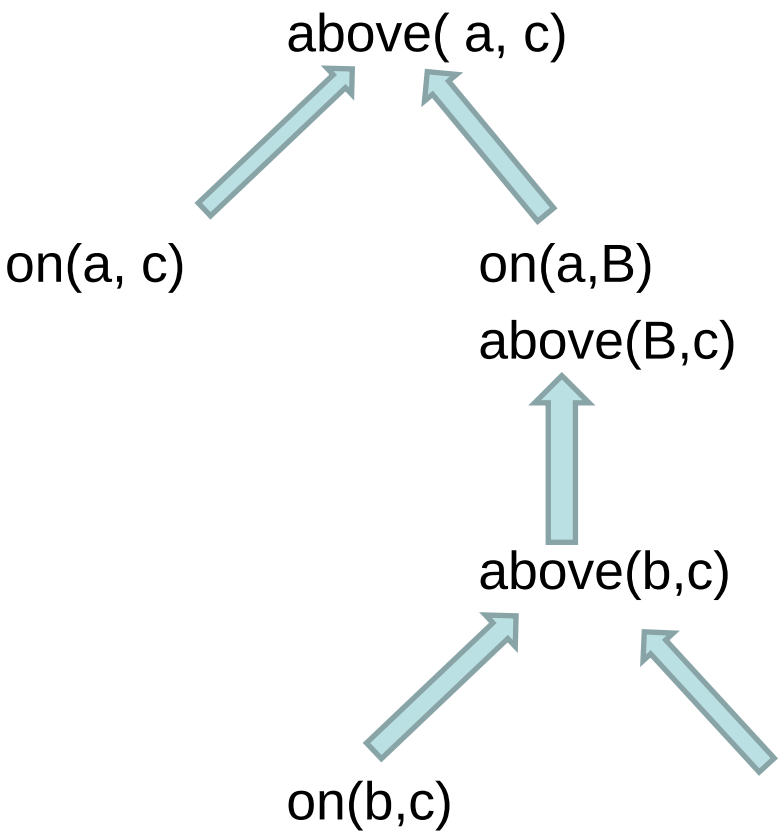
Vse to tvori ***deklarativni pomen***

RELACIJA „ABOVE“

```
% above( Block1, Block2): Block1 above Block2 in the same stack

above( B1, B2)  :-
  on( B1, B2).

above( B1, B2)  :-
  on( B1, B),
  above( B, B2).

?-  above( a, c).     % Trace proof tree for this
```

PROLOG IŠČE DOKAZ

- ?- above(a, c).



INTERPRETER ZA PROLOG = DOKAZOVALNIK IZREKOV

- Preiskuje drevo možnih dokaznih poti
- Avtomatsko vračanje (backtracking)
- Strategija preiskovanja: v globino

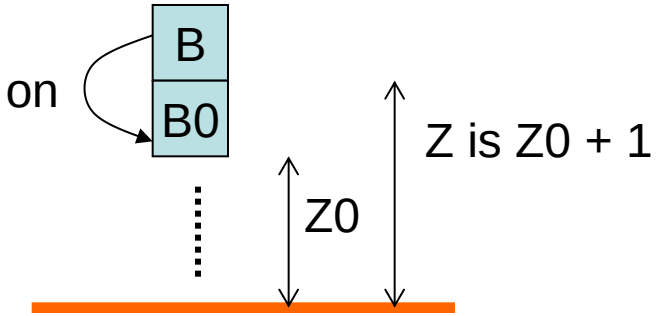
z-koordinate kocke

```
% z( Block, Z):  Z is z-coord. of Block

z( B, 0)  :-                    % Z=0 if block is on table
  on( B, table).
```

Poišči Z koordinato kocke

```
% z( Block, Z):  z-coord. of Block

z( B, 0)  :-
  on( B, table).
```


```
z( B, Z)  :-
  on( B, B0),
  z( B0, Z0),
  Z is Z0 + 1.
```



Prolog najraje pusti rezultate v simbolični obliki

```
% An attempt at shortening this

z( B, 0)  :-
  on( B, table).

z( B, Z0 + 1)  :-
  on( B, B0),
  z( B0, Z0).

?-  z( a, Za).
Za  =  0+1+1          % Prolog constructs a formula!
```

Prolog z lahkoto sestavi splošno formulo

```
z( B, Z0 + height( B0))  :-        % Add hight of block B
  on( B, B0),
  z( B0, Z0).
```

```
?-  z( a, Za).
```

Za = 0 + height(c) + height(b)

Sledenje izvajanja

?- z(c, Z).

Z = 0

?- z(a, Z).

Z = 2

% Trace proof tree of this execution

?- trace.

...

?- z(a, Z).

...

KRATKI PRIMERI

- Poišči skrajne desne kocke
- Poišči vse vidne kocke
- Poišči vse kocke
- Poišči vse kocke na višini 2

TERMINOLOGIJA

- *Stavek* = *dejstvo* ali *pravilo* ali *vprašanje*

- Primer pravila:

above(B1, B2) :- on(B1, B), above(B, B2).

glava *telo*, *“cilji”* v telesu

- *Spremenljivka* (X, Z, B1, X12, ...)
- *Konstanta* = *število* (4, -1, ...) ali *atom* (a1, x12, ...)

DEKLARATIVNI IN POSTOPKOVNI POMEN PROLOGA

- A & B je logično ekvivalentno B & A

- Deklarativni pomen prologa = logični pomen
- Vrstni red ciljev in stavkov **ne vpliva** na deklarativni pomen

- Postopkovni pomen prologa = algoritem, ki poišče dokaz
- Vrstni red ciljev in stavkov **vpliva** na postopkovni pomen

VARIANTE „ABOVE“

Z zamenjavo vrstnega reda stavkov ali ciljev v telesu stavka dobimo nove variante procedure above, npr.:

```
above2( B1, B2)  :-
  above2( B, B2),
  on( B1, B).

above2( B1, B2)  :-
  on( B1, B2).

?-  above2( a, c).    % Trace to see what happens
```

POSKUSI NAJPREJ ENOSTAVNE STVARI!

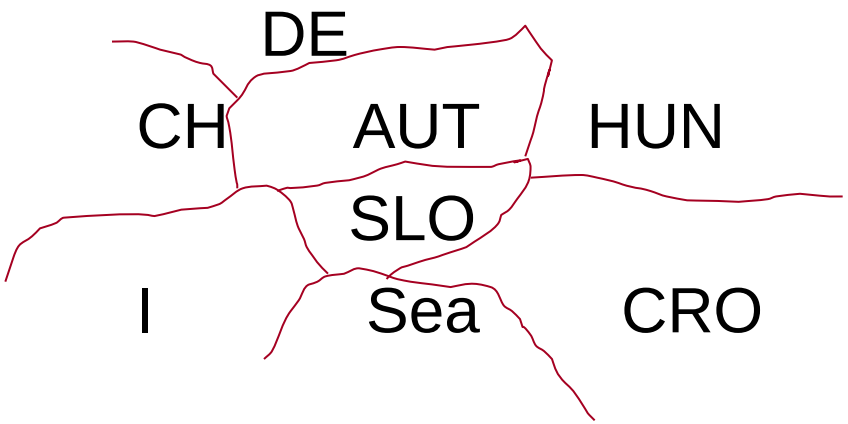
- Zakaj **above2** postopkovno ne uspe?
- **above2** vedno poskusi najprej zapletene stvari - slaba ideja!

- Princip, ki pogosto deluje: Poskusi najprej enostavne stvari

- Ali moramo vedno preveriti postopkovne detalje? Ne, navadno ne!
- Prav nasprotno: Naj ne bi!
- Programer naj bi razmišljal deklarativno – to je bistvo deklarativnih jezikov („Kaj?“ in ne „Kako?“)

BARVANJE ZEMLJEVIDA

- Problem: Pobarvaj države v danem zemljevidu
- Izrek: Štiri barve zadoščajo za katerikoli zemljevid
- Primer zemljevida:



- Sosednje države:

  n(I, SLO), n(I, Sea), n(Sea, SLO), …

BARVANJE ZEMLJEVIDA

% Possible pairs of colors of neighbour countries

n(red, green). n(red, blue). n(red, yellow).
n(green, red). n(green, blue). n(green, yellow).
n(blue, red). n(blue, green). n(blue, yellow).
n(yellow, red). n(yellow, green). n(yellow, blue).

BARVANJE ZEMLJEVIDA

```
% Part of Europe

colors( IT, SI, CR, CH, AT, HU, DE, SV, CZ, PO, SEA)  :-
   SEA = blue,
   n( IT, CH), n( IT, AT), n( IT, SI), n( IT, SEA),
   n( SI, AT), n( SI, CR), n( SI, HU), n( SI, SEA),
   n( CR, HU), n( CR, SEA),
   n( AT, CH), n( AT, DE), n( AT, HU), n( AT, SV), n( AT, CZ),
   n( CH, DE),
   n( HU, SV),
   n( DE, SV), n( DE, CZ), n( DE, PO),
   n( SV, CZ), n( SV, PO),
   n( CZ, PO).
```

KRIŽANKA

```
% Primer križanke
%
%        L1   L2   L3   L4   L5
%        L6        L7        L8
%        L9   L10  L11  L12  L13  L14
 %       L15                 L16

 % Vpiši črke L1, L2, … tako, da tvorijo veljavne besede
 %  iz danega slovarja
```

KRIŽANKA

```
% Possible words - Dictionary

word( d,o,g).      word( r,u,n).      word( t,o,p).      word( f,i,v,e).
word( f,o,u,r).    word( l,o,s,t).    word( m,e,s,s).
word( u,n,i,t).    word( b,a,k,e,r).  word( f,o,r,u,m).
word( g,r,e,e,n).  word( s,u,p,e,r).  word( p,r,o,l,o,g).
word( v,a,n,i,s,h). word( w,o,n,d,e,r). word( y,e,l,l,o,w).
```

KRIŽANKA

```
% Problem statement

solution( L1,L2,L3,L4,L5,L6,L7,L8,L9,L10,L11,L12,L13,L14,L15,L16)  :-
    word( L1,L2,L3,L4,L5),                  % Top horizontal word
    word( L9,L10,L11,L12,L13,L14),          %  Second horizontal word
    word( L1,L6,L9,L15),                    %  First vertical word
    word( L3,L7,L11),                       %  Second vertical word
    word( L5,L8,L13,L16).                   %  Third vertical word
```

PLANIRANJE PROJEKTNEGA SESTANKA

% Organising a project meeting according to these specifications

% The meeting is organised in 3 sessions: artificial intelligence, bioinformatics, and databases

% Each session takes half a day, morning or afternoon

% Session on bioinformatics takes place before session on databases

% Each session concerns a topic, and at least two participants
% of a session have to be experts in the session's topic

% Problem is to assign times and experts to sessions

```
% Possible times of sessions

time( morning).
time( afternoon).

before( morning, afternoon).          % morning is before afternoon

% Experts for topics

expert( bioinformatics, barbara).
expert( bioinformatics, ben).
expert( artificial_intelligence, adam).
expert( artificial_intelligence, ann).
expert( artificial_intelligence, barbara).
expert( databases, adam).
expert( databases, danny).
```

```
% session( Time, Topic, P1, P2):
%    session at Time on Topic attended by experts P1, P2

session( Time, Topic, P1, P2)  :-
   time( Time),
   expert( Topic, P1),
   expert( Topic, P2),
   P1  \=  P2.                    % P1, P2 different persons
```

```
% no_conflict( Time1, P1, P2, Time2, Q1, Q2):
%   There is no time conflict between two sessions at Time1 and Time2
%   and experts P1, P2, and Q1, Q2, respectively

no_conflict( Time1, _, _, Time2, _, _)  :-
   Time1  \=  Time2.                    % OK, sessions at differet times

no_conflict( Time, P1, P2, Time, Q1, Q2)  :-        % Parallel sessions
   P1  \=  Q1,  P1  \=  Q2,                         % No overlap between experts
   P2  \=  Q1,  P2  \=  Q2.
```

```
% schedule( TimeA, A1, A2, TimeB, B1, B2, TimeD, D1, D2):
%   TimeA and expertsA1, A2 assigned to session on Artificial Intelligence,
%   TimeB, B1, B2 assigned to session on bioinformatics, etc.

schedule( Ta, A1, A2, Tb, B1, B2, Td, D1, D2)  :-
   session( Ta, artificial_intelligence, A1, A2),
   session( Tb, bioinformatics, B1, B2),
   session( Td, databases, D1, D2),
   before( Tb, Td),                          % Bioinformatics happens before Databases
   no_conflict( Ta, A1, A2, Tb, B1, B2),     % No conflict between AI and Bioinfo
   no_conflict( Ta, A1, A2, Td, D1, D2),     % No conflict between Databases and AI
   no_conflict( Tb, B1, B2, Td, D1, D2).     % No conflict between Bioinfo and Data.
```

?- schedule(Ta, A1, A2, Tb, B1, B2, Td, D1, D2).

A1 = adam,

A2 = ann,

B1 = barbara,

B2 = ben,

D1 = adam,

D2 = donald,

Ta = morning,

Tb = morning,

Td = afternoon ;

...

VPRAŠANJE

- Koliko je možnih urnikov

```
?- findall( 1,
      schedule( Ta, A1, A2, Tb, B1, B2, Td, D1, D2),L),
   length(L,N).

L = [1,1,1,1,1,1,1,1,1,1|...],
N = 16 ?
```