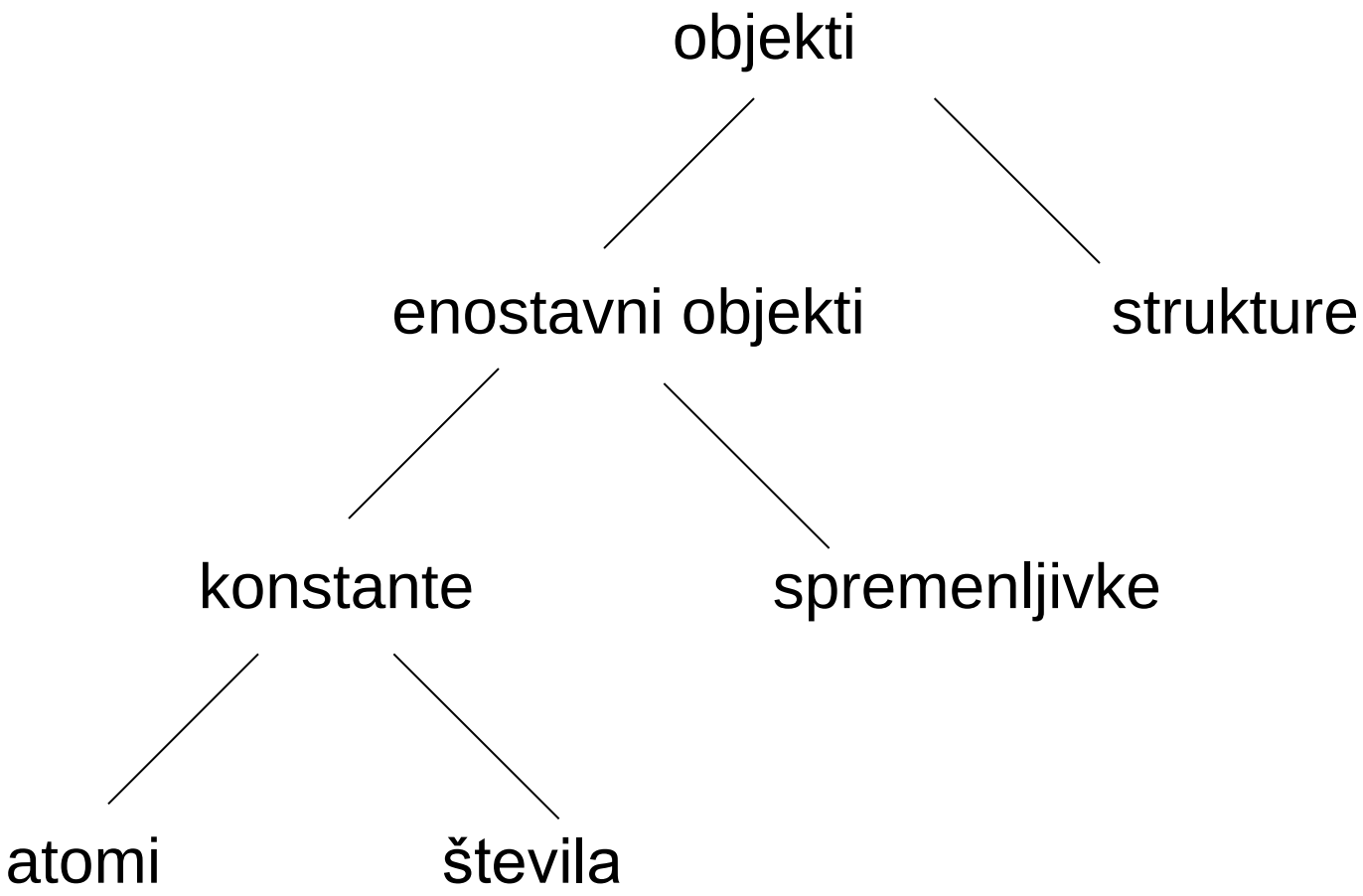
PROLOG
SINTAKSA IN SEMANTIKA

Ivan Bratko
Univerza v Ljubljani

PODATKOVNI OBJEKTI



SINTAKSA OBJEKTOV

- Tip objekta je vedno razpoznaven iz sintaktične oblike

TRI SINTAKTIČNE OBLIKE ZA ATOME

(1) Zaporedja črk, cifer ali “_”, začenši z malo črko

x x15 x_15 aBC_CBa7

alpha_beta_algorithm taxi_35

peter missJones miss_Jones2

ATOMI, NAD.

(2) Zaporedja posebnih znakov

---> <==> <<

. < > + ++ ! .. .::. ::= []

ATOMI, NAD.

(3) Zaporedja znakov v narekovajih

‘X_35’ ‘Peter’ ‘Britney Spears’

ŠTEVILA

- Cela števila

  1 1313 0 -55

- Realna števila (plavajoča vejica)

  3.14 -0.0045 1.34E-21 1.34e-21

SPREMENLJIVKE

- Zaporedja črk, cifer ali podčrtajev, začenši z veliko črko

  X Results Object2B Participant_list

  _x35 _335

- Leksikalni domet imen spremenljivk je en stavek

- Podčrtaj stoji za *anonimno* spremenljivko
- Vsaka pojavitev podčrtaja je nova anonimna spremenljivka

ANONIMNE SPREMENLJIVKE

```
visible_block( B)  :-
  see( B, _, _).
```

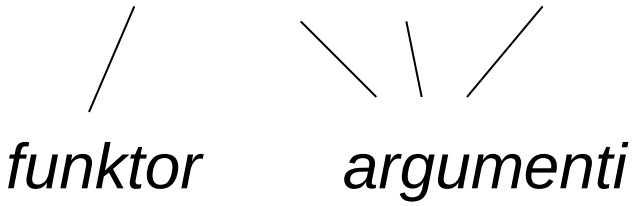
Ekvivalenten zapis:

```
visible_block( B)  :-
  see( B, X, Y).
```

STRUKTURE

- Strukture so objekti z več komponentami
- Npr.: datum je struktura s tremi komponentami
- Datum 5. marec 2017:

date(5, march, 2017)



- Argument je lahko katerikoli objekt, tudi struktura

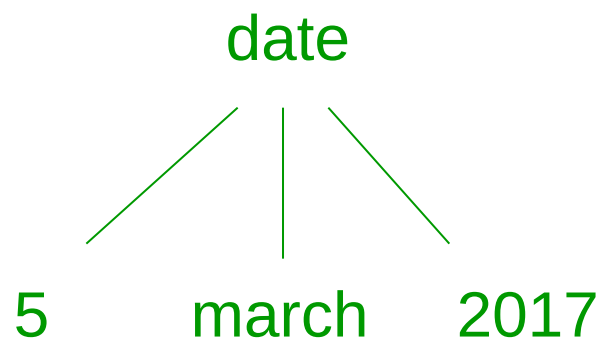
FUNKTORJI

- Ime funktorja, izbere programer
- Sintaksa funktorjev: atomi
- Funktor je definiran z imenom in mestnostjo („arity“)

DREVESNA PREDSTAVITEV STRUKTUR

Strukture včasih ponazorimo kot drevesa

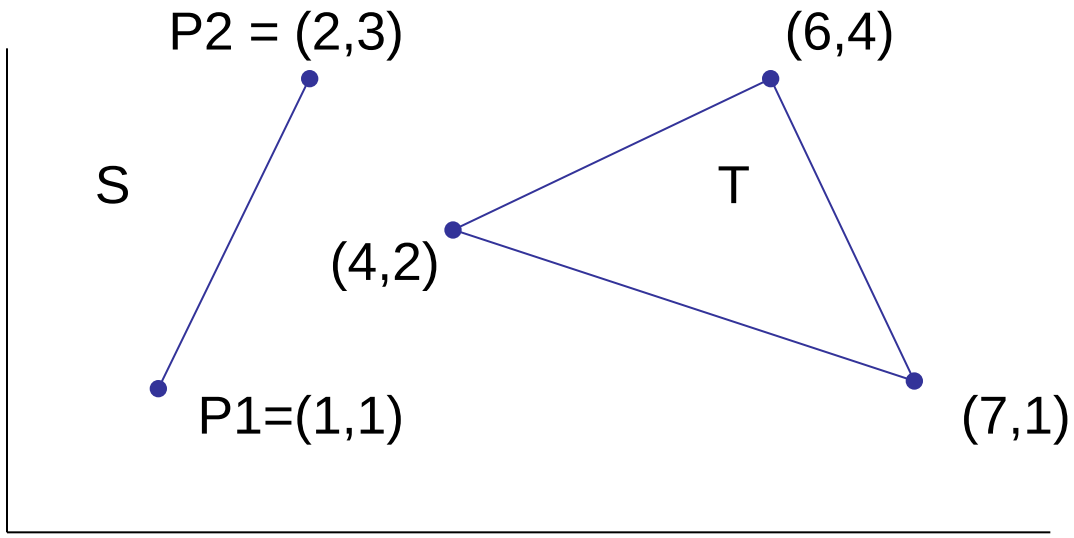
date(5, march, 2017)



- Vse strukturirane objekte v prologu lahko ponazorimo z drevesi

- To je *edini* način gradnje struktur v prologu

NEKAJ GEOMETRIČNIH OBJEKTOV



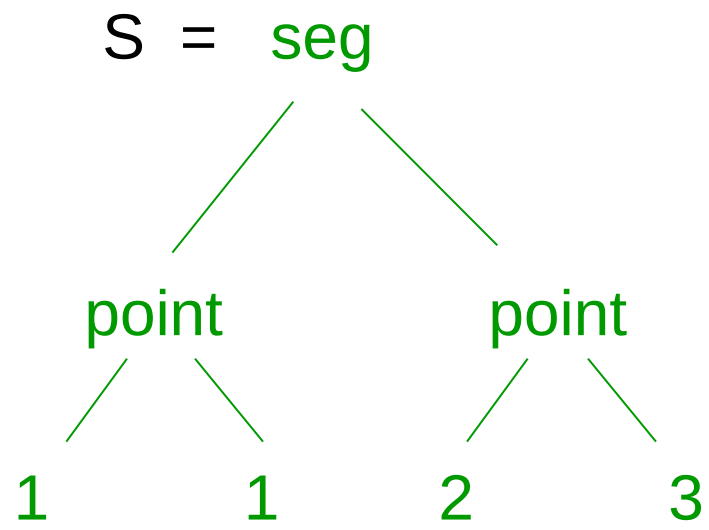
P1 = point(1, 1) P2 = point(2, 3)

S = seg(P1, P2) = seg(point(1,1), point(2,3))

T = triangle(point(4,2), point(6,4), point(7,1))

DALJICE

S = seg(point(1,1), point(2,3))

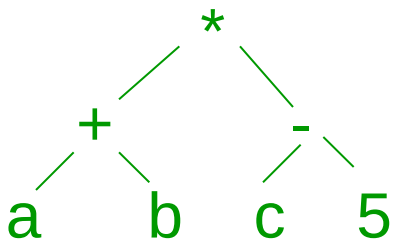


TUDI ARITMETIČNI IZRAZI SO DREVESA

- Na primer: (a + b) * (c - 5)

- Zapisano kot izraz s funktorji:

*(+(a, b), -(c, 5))



PRILAGAJANJE

- Prilagajenje je operacija na strukturah

- Dve strukturi s lahko prilagodita, če:

  (1) Sta identični, ali
  (2) Ju lahko naredimo identični s primerno opredelitvijo spremenljivk

- *opredelitev spremenljivke* =
  spremenljivka dobi vrednost =
  instantiation of variable

PRIMER PRILAGAJANJA

- Prilagajanje datumov:

  date(D1, M1, 2006) = date(D2, june, Y2)

- Spremenljivke se opredelijo takole:

  D1 = D2

  M1 = june

  Y2 = 2006

- To je *najbolj splošna* prilagoditev (*most general instantiation*)
- Manj splošna prilagoditev: D1=D2=17, ...

NAJBOLJ SPLOŠNA PRILAGODITEV

- V prologu prilagajanje vrne vedno najbolj splošno prilagoditev

- S tem so spremenljivke opredeljene do najmanjše mere, imamo čim večjo fleksibilnost za nadaljnja prilagajanja

- Npr:

```
?- date( D1, M1, 2006)  =  date( D2, june, Y2),
   date( D1, M1, 2006)  =  date( 17, M3, Y3).
```

D1 = 17, D2 = 17, M1 = june, M3 = june,

Y2 = 2006, Y3 = 2006

PRILAGAJANJE

- Prilagajanje uspe ali ne uspe

- Izraza S in T se prilagodita:

  (1) Če sta S inT konstanti, potem se prilagodita le,
  če sta identična

  (2) Če je S premeljivka, prilagajanje uspe,
  S postane enak T

  (3) Če sta S in T strukturi, potem se prilagodita le če:

  (a) imata isti glavni funktor in

  (b) vsi njuni ustrezajoči argumenti se med seboj prilagodijo

PRILAGAJANJE ≈ UNIFIKACIJA

- Unifikacija je pojem iz predikatne logika
- Unifikacija = Prilagajanje + Preverba pojavitve („Occurs check“)
- Kaj se zgodi, če prolog vprašamo:

```
?-  X = f(X).
X = f(f(f(f(f(f(f(f(f(f( ... ))))))))))
```

Prilagajanje uspe, unifikacija ne uspe

RAČUNANJE S PRILAGAJANJEM

```
% Definition of vertical and horizontal segments
vertical( seg( point( X1,Y1), point( X1, Y2))).
horizontal( seg( point( X1,Y1), point( X2, Y1))).

?- vertical( seg( point( 1,1), point( 1, 3))).
yes
?- vertical( seg( point( 1,1), point( 2, Y))).
no
?- vertical( seg( point( 2,3), P)).
P = point( 2, _173).
```

ZANIMIVA DALJICA

- Ali obstaja daljica, ki je horizontalna in vertikalna

?- vertical(S), horizontal(S).

S = seg(point(X,Y), point(X,Y))

- Prolog lahko prikaže rezultate s preimenovanimi spremenljivkami:

S = seg(point(_13,_14), point(_13, _14))

DEKLARATIVNI POMEN

- Naj bo dan program P in cilj G
- G je resničen (to je logično sledi iz P), če in samo če:

  (1) Obstaja stavek C v P, da velja

  (2) Obstaja primerek I stavka C, da velja

  (a) glava od I je enaka G, in

  (b) vsi cilji v telesu stavka I so resnični

- *Primerek* stavka C dobimo s preimenovanjem spremenljivk v C in po možnosti zamenjavo spremenljivke z izrazom. Npr. primer stavka

  p(X,Y) :- q(Y,Z)

  je

  p(U,a) :- q(a,V).

DEKLARATIVNI IN POSTOPKOVNI POMEN PROGRAMA

- Vzemimo stavek:

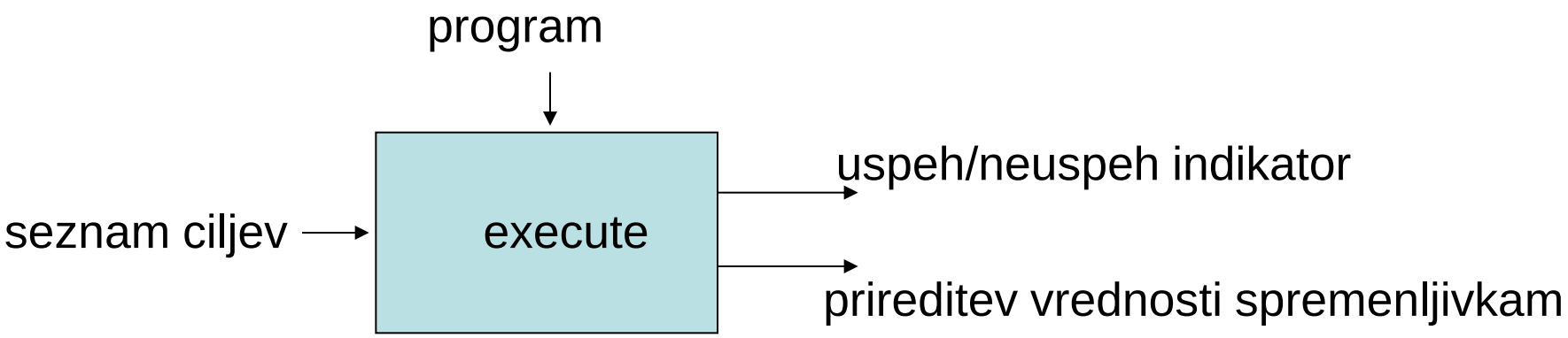
  P :- Q, R.

- Deklarativno branje stavka:
  - P is true if Q *and* R are true.
  - From Q *and* R follows P.

- Postopkovno branje:
  - Da rešiš problem P, *naprej* reši podproblem Q in *zatem* R.
  - Da uresničiš P, *najprej* uresniči Q in *zatem* R.

POSTOPKOVNI POMEN

- Pove, kako prolog dokazuje cilje
- Postopkovni pomen je algoritem za izvršitev seznama ciljev glede na dani program



procedura execute(Program, GoalList, Success)

- execute = deklarativni pomen + postopkovni elementi

Preišči program od zgoraj navzdol

G je resničen (to je logično sledi iz P), če in samo če:

(1) Obstaja stavek C v P, da velja

(2) Obstaja primerek I stavka C, da velja

(a) glava od I je enaka G, in

(b) vsi cilji v telesu stavka I so resnični

Prilagajanje:

G = glava od I

Izvedi podcilje v vrstnem redu, kot so zapisani v programu