

## Primeri Prolog programa

### 1. Definirati predikat kojim se određuje

a) maksimum

b) minimum

za dva broja A i B.

maksimum(A,B,A):-A>=B.

maksimum(A,B,B):-A<B.

min(A,B,A):-A<= B.

min(A,B,B):-A>B.

### 2. Definirati predikat kojim se za dati prirodan broj N određuje suma

$1+2+\dots+N$ .

suma(0,0).

suma(N,S):-N>0,N1 is N-1, suma(N1,S1),  
S is S1+N.

### 3. Definirati predikat kojim se za dati ceo broj X određuje apsolutna vrednost.

abs(X,X):- X>=0.

abs(X,Y):- X<0, Y is -X.

**PROVERA, poziv abs(a,X). i u X se upisuje apsolutna vrednost od a.**

**Napomena: Ispitivanje u drugoj klauzuli je neophodno kada se predikat koristi u provrenom smislu, tj. zbog mogućih upita abs(5,-5). koji predstavljaju korišćenje u provrenom smislu.**

```
1 ?- abs(-5,5) .  
true.
```

```
2 ?- abs(5,-5) .  
false.
```

```
3 ?- abs(5,X) .  
X = 5 ;  
false.
```

```
4 ?- abs(5,X) .  
X = 5 .
```

```
5 ?- abs(-5,X) .  
X = 5 .
```

Provera za program

abs(X,X):- X>=0.

abs(X,Y):- Y is -X.

IZLAZ

```
1 ?- abs(5,X).
```

```
X = 5 .
```

```
2 ?- abs(5,-5).
```

```
true.
```

3 ?- abs(-5,X).

X = 5.

#### 4. Definisati predikat kojim se za dati prirodan broj N odredjuje suma

a) parnih brojeva od 2 do N

b) neparnih brojeva od 1 do N

a)

glavni(N,S):-N1 is N - N mod 2,sumaPar(N1,S).

sumaPar(0,0).

sumaPar(N,S):-N>0, N1 is N-2,

sumaPar(N1,S1),S is S1+N.

#### 5. Definisati predikat kojim određuje NZD i NZS dva prirodna broja.

Za nalaženje NZD i NZS dva prirodna broja koristićemo Euklidov algoritam. U navedenom primeru klauzula nzd poziva samu sebe, tj. primenjuje se rekurzivni metod.

Rekurzija predstavlja glavni metod u PROLOG-u i primenjuje se u rešavanju velikog broja problema. Prilikom primene rekurzije mora postojati i klauzula koja omogućava završetak rekurzije. To je klauzula sa istim imenom i brojem argumenata, ali u kojoj se ne javlja rekurzivan poziv.

NZS se dobija jednostavno množenjem oba data prirodna broja i deljenjem sa NZD.

nzd(N,0,N).

nzd(N,M,K):- P is N mod M, nzd(M,P,K).

nzs(N,M,K):- nzd(N,M,P), K is N\*M//P.

IZLAZ

1 ?- nzd(15,9,K).

K = 3 ;

ERROR: mod/2: Arithmetic: evaluation error: `zero\_divisor'

Napomene:

1. Uočite **K is N\*M//P**

// je operator celobrojnog deljenja, ako u Vasem prologu postoji operator realnog deljenja.

Ako se ne koristi skup realnih brojeva, operator za celobrojno deljenje se jednostavnije oznacava sa /

2. U navedenom rešenju se mogu pojaviti problemi ako se nakon nađenog resenja zahteva da se nadje sledece resenje (pritiskom na taster ; u interakciji sa PROLOG sistemom). Šta se dešava u gornjem primeru izlaza?

PROLOG pokušava da nađe rešenje, ali zalazi u skup negativnih brojeva (ili pokušaj deljenja nulom) i nastaje beskonačan ciklus. Da bi se to izbeglo, treba izmeniti 1. klauzulu u nzd(N,0,N) :-!.

nzd(N,0,N) :-!.

nzd(N,M,K):- P is N mod M, nzd(M,P,K).

nzs(N,M,K):- nzd(N,M,P), K is N\*M//P.

IZLAZ

3 ?- nzd(15,9,K).

K = 3.

Ovo je tipičan primer u kom se primenjuje predikat odsecanja !

IZLAZ

5 ?- nzd(12,8,K).

K = 4.

6 ?- nzd(12,8,NZS).

NZS = 4.

7 ?- nzs(13,17,NZS).

NZS = 221

Zaključak: Predikat odsecanja sprečava pretraživanje po celom stablu pretraživanja i na taj način ubrzava postupak pretraživanja u PROLOG-u, a takođe pravi i uštede u zauzimanju memorijskog prostora.

Za razmišljanje: Da li sledeći Prolog program za izračunavanje NZD dva broja korektno radi i u generativnom i u provernom smislu? Savet: postavite upite gcd(18,12,X), gcd(12,18,X), gcd(18,12,6), gcd(12,18,9).

gcd(X, Y, G) :- X = Y, G = X.

gcd(X, Y, G) :- X < Y, Y1 is Y - X, gcd(X, Y1, G).

gcd(X, Y, G) :- X > Y, gcd(Y, X, G).

**6. Napisati predikat koji za dati ceo broj N ispisuje N-ti član Fibonacijevog niza na poziciji N.**

Resenje 01

```
/* fib(1,1):-!.
fib(2,1):-!.
*/
```

```
fib(1,1).
```

```
fib(2,1).
```

```
fib(1,1).
```

```
fib(2,1).
```

```
fib(N, X):- M is N-1,
```

```
    fib(M, Y),
```

```
    K is N-2,
```

```
    fib(K, Z),
```

```
    X is Y+Z.
```

IZLAZ

12 ?- fib(5,F).

F = 5 .

13 ?- fib(8,F).

F = 21 .

14 ?- fib(1,F).

F = 1 .

Resenje 02 (sa medjuispisom)

```
fib(1,1):-!.
fib(2,1):-!.

```

```


```

```
%fib(1,1).
```

```
%fib(2,1).
```

```
fib(N, X):- M is N-1,
```

```
    write(M), nl,
```

```
    fib(M, Y),
```

```
    K is N-2,
```

```
    fib(K, Z),
```

```
    X is Y+Z.
```

**7. Napisati predikat koji za dati ceo broj izračunava njegov faktorijel.**

fakt(0,1):-!.

fakt(N, F):- N1 is N-1, fakt(N1, X), F is N\*X.

Dakle, računamo faktorijal na uobičajen način:

$1*2*...*(n-1)*n$

**IZLAZ**

6 ?- fakt(5,X).

X = 120.

7 ?- fakt(6,X).

X = 720.

8 ?- fakt(0,X).

X = 1.

Zadaci za samostalan rad

1. Definisati predikat kojim se za dati prirodan broj N odredjuje

a) broj cifara

b) suma cifara

c) maksimalna cifra.

2. Definisati predikat kojim se određuje N-ti stepen (N prirodan broj) celog broja A.

3. Definisati predikat kojim se određuje suma delioca broja N, ne uključujući broj N.

4. Definisati predikat kojim se proverava da li je dati prirodan broj savrsen jednak sumi svojih delioca (ne uključujući broj N).

5. Definisati predikat kojim se proverava da li je dati prirodan broj prost.

6. Definisati predikat kojim se za dati prirodan broj N određuju suma parnih i suma neparnih prirodnih brojeva od 1 do N.

**PRIMER: Lista1.pl**

lista([]).

lista([a,b,c]).

lista([mara, pera, zika, 1, 2.67, sima]).

lista([prvi, [jabuka, kruska], drugi, datum(5,6,2001)]).

lista([2.45, 0, mika, <>, 'Marko Kraljevic', ime(ilic, pera), [a, b, 1],4]).

**IZLAZ**

1 ?- lista(X).

X = [] .

2 ?- lista(X).

X = [] ;

X = [a, b, c] ;

X = [mara, pera, zika, 1, 2.67, sima] ;

X = [prvi, [jabuka, kruska], drugi, datum(5, 6, 2001)] ;

X = [2.45, 0, mika, <>, 'Marko Kraljevic', ime(ilic, pera), [a, bl...], 4].

**3 ?- lista([G|R]).**

G = a,

R = [b, c] ;

G = mara,

R = [pera, zika, 1, 2.67, sima] ;

G = prvi,  
R = [[jabuka, kruska], drugi, datum(5, 6, 2001)] ;  
G = 2.45,  
R = [0, mika, <>, 'Marko Kraljevic', ime(ilic, pera), [a, b, 1], 4].

### **Pripada.pl**

pripada( X, [Y|\_]) :- X=Y.  
pripada(X, [\_|Y]) :- pripada(X,Y).

#### IZLAZ

5 ?- pripada(a,[a,b,c]).  
true .

6 ?- pripada(a,lista).  
false.

7 ?- pripada(X,[a,b,c]).  
X = a ;  
X = b ;  
X = c ;  
false.

### **Razgovor.pl**

zameni(ti, ja).  
zameni(si, nisam).  
zameni(umes, umem).  
zameni(dali, ne).  
zameni(mozes, mogu).  
zameni(X,X).  
tra([],[]).  
tra([G|R], [X|Y]):-zameni(G,X),tra(R,Y), !.

### **HirurgC.pl**

/\* Ovde nalazi samo jednog hirurga! \*/

lekar(marko).  
lekar(petar).  
lekar(milan).  
operise(petar).  
operise(milan).  
hirurg(X):- lekar(X), operise(X),!.

### **HirurgCU.pl**

/\* odlican primer za crveni Cut-predikat  
Ovde nema hirurga u generativnom smislu \*/

lekar(marko).  
lekar(petar).  
lekar(milan).  
operise(petar).  
operise(milan).  
hirurg(X):- lekar(X), !, operise(X).

## **biblioteka.pl**

clan(markovic).

clan(pesic).

clan(acic).

clan(zivic).

nije\_vratio(pesic).

nije\_vratio(zivic).

usluge(Clan, Vid\_usluge) :- nije\_vratio(Clan), !, osnovne(Vid\_usluge).

usluge(Clan, Vid\_usluge) :- opste(Vid\_usluge).

osnovne(koriscenje\_kataloga).

osnovne(citanje\_novina).

dopunske(uzimanje\_knjiga).

dopunske(medjubibliotecka\_razmena).

dopunske(koriscenje\_racunara).

opste(X) :- osnovne(X).

opste(X) :- dopunske(X).

## **razgovorCut.pl**

zameni(ti, ja).

zameni(si, nisam).

zameni(umes, umem).

zameni(dali, ne).

zameni(mozes, mogu).

zameni(X,X).

tra([],[]):-!.

tra([G|R], [X|Y]):-zameni(G,X),tra(R,Y).

## **kolicnik.pl**

kolicnik(N1,N2,Rez):- ceo\_broj(Rez),

    P1 is Rez\*N2,

    P2 is (Rez+1)\*N2,

    P1=<N1, P2>N1, !.

ceo\_broj(0).

ceo\_broj(X) :- ceo\_broj(Y), X is Y+1.

## **inostrani.pl**

inostrani(dzon).

inostrani(martin).

dohodak(dzon, 5000).

dohodak(martin, 10000).

dohodak(dragan, 10000).

dohodak(pera, 20000).

placa\_porez(X) :- inostrani(X),!,fail.

placa\_porez(X) :- dohodak(X,D), D>1000.

## **zaposlen.pl**

/\* Program se upotrebljava bez i sa Cut-predikatom.

Kada se koristi sa Cut-predikatom, daje kontradiktorne

odgovore. Postaviti upite:

?- zaposlen(X). i

?- zaposlen(marko).

\*/

zaposlen(X) :- zanatlija(X).

zaposlen(X) :- sluzbenik(X).

zanatlija(X) :- ima\_radnju(X),!,ima\_diplomu(X).

zanatlija(X) :- X=aca\_pekar, ima\_diplomu(X).

ima\_radnju(milan).

ima\_radnju(marko).

ima\_radnju(janko).

ima\_diplomu(X).

sluzbenik(darko).

### **PRIMER izlaza za program bez !**

1 ?- zaposlen(X).

X = milan ;

X = marko ;

X = janko ;

X = aca\_pekar ;

X = darko.

2 ?- zaposlen(marko).

true .

### **PRIMER izlaza za program sa !**

1 ?- zaposlen(X).

X = milan ;

X = darko.

2 ?- zaposlen(marko).

true .

### **Spajanje dve liste bez i sa Cut predikatom**

/\* Inace, program, osim spajanja, omogućava i razdvajanje listi (deluje u inverznom smeru).

Problem u izlaznom rezultatu za inverzni smer tj. razdvajanje listi je razlika u izlazu u resenju koje koristi Cut predikat \*/

spoj([],X,X).

spoj([Xg|Xr],Y,[Xg|Z]) :- spoj(Xr,Y,Z).

### **IZLAZ**

1 ?- spoj([a,b,c],[d,e],X).

X = [a, b, c, d, e].

2 ?- spoj([a,b,c],X,Y).

Y = [a, b, c|X].

/\* Program, osim spajanja, omogućava i razdvajanje listi (deluje u inverznom smeru).

Problem u ovom primeru je razlika u izlazu u resenju koje koristi Cut predikat \*/

3 ?- spoj(X,Y,[a,b,c]).

X = [],

Y = [a, b, c] ;

**X = [a],**  
**Y = [b, c] ;**  
**X = [a, b],**  
**Y = [c] ;**  
**X = [a, b, c],**  
**Y = [] ;**  
**false.**

**spoj([],X,X) :- !.**  
**spoj([G|R],X,[G|Y]) :- spoj(R,X,Y).**

### **IZLAZ**

1 ?- spoj([a,b,c],[d,e],X).  
X = [a, b, c, d, e].

2 ?- spoj([a,b,c],X,Y).

Y = [a, b, c|X].

**3 ?- spoj(X,Y,[a,b,c]).**

**X = [],**  
**Y = [a, b, c].**

## **Podsećanje na primere sa prošlog časa**

### **PRIMER 01 - Primer01.pl**

```
/* Rodbinske relacije */  
muskarac(marko).  
muskarac(milan).  
muskarac(petar).  
muskarac(dusan).  
zena(jelena).  
zena(draga).  
zena(marija).  
roditelji(milan, marija, marko).  
roditelji(jelena, marija, marko).  
roditelji(petar, marija, marko).  
roditelji(draga,jelena, dusan).  
sestra(S,X) :- zena(S), roditelji(S,M,O),  
               roditelji(X,M,O), not(S=X).  
brat(B,X):-muskarac(B), roditelji(B,M,O),  
           roditelji(X,M,O), not(B=X).  
majka(M,D):-roditelji(D,M,O).  
otac(O,D):-roditelji(D,M,O).
```

### Izvršavanje u SWI-prolog

1. Dupli klik na .pl file (default folder Prolog)
2. upit  
muskarac(X). (klik na enter)



```
SWI-Prolog -- c:/Users/ilija/Documents/Prolog/primer01.pl
File Edit Settings Run Debug Help
$ library(win_menu) compiled into win_menu 0.02 sec, 11,452 bytes
$ library(swi_hooks) compiled into pce_swi_hooks 0.00 sec, 2,232 bytes
Warning: c:/users/ilija/documents/prolog/primer01.pl:17:
Singleton variables: [O]
Warning: c:/users/ilija/documents/prolog/primer01.pl:18:
Singleton variables: [M]
$ c:/Users/ilija/Documents/Prolog/primer01.pl compiled 0.00 sec, 2,796 bytes
Welcome to SWI-Prolog (Multi-threaded, 32 bits, Version 5.10.5)
Copyright (c) 1990-2011 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic) . or ?- apropos(Word) .

1 ?- muskarac(X) .
X = marko .

2 ?-
```

### 3. upit

muskarac(X). (klik na enter, potom vise puta pritisnuti ;)

```
5 ?- muskarac(X) .
X = marko ;
X = milan ;
X = petar ;
X = dusan.
```

Da li uočavate razliku?

### 4. upit

otac(X,petar).

```
4 ?- otac(X,petar) .
X = marko.
```

## PRIMER 02

/\* Program za ispitivanje odnosa u suncevom sistemu \*/

planeta(merkur).

planeta(venera).

planeta(zemlja).

planeta(mars).

planeta(venera).

planeta(saturn).

planeta(jupiter).

planeta(uran).

planeta(neptun).

veca(venera, merkur).

veca(zemlja, venera).

veca(uran,zemlja).

veca(saturn, uran).

manja(saturn, jupiter).

veca(X,Y) :- zvezda(X), planeta(Y).

zvezda(sunce).

veca(X,Y) :- planeta(Z), veca(X,Z), veca(Z,Y).

veca(X,Y) :- manja(Y,X).

manja(venera, saturn).

manja(zemlja, jupiter).

For help, use ?- help(Topic) . or ?- apropos(Word) .

```
1 ?- planeta(X) .
X = merkur ;
X = venera ;
X = zemlja ;
X = mars ;
X = venera ;
X = saturn ;
X = jupiter ;
X = uran ;
X = neptun.
```

```
2 ?- ■
```

### PRIMER 03

/\* Baza podataka sadrzi cinjenice

```
    vojnik(Ime, Visina, Tezina).
*/
```

```
vojnik('Aca Peric', 183, 78).
vojnik('Milan Ilic', 192, 93).
vojnik('Stanoje Sobic', 173, 81).
vojnik('Sasa Minic', 162, 58).
vojnik('Dragan Sadzakov', 180, 103).
vojnik('Pera Peric', 200, 80).
vojnik('Rade Dokic', 160, 56).
zadovoljava(Ime) :- vojnik(Ime, Visina, Tezina),
    Visina>170, Visina<190, Tezina =< 95,
    Tezina>60.
otpada(Ime) :- vojnik(Ime, _, Tezina), Tezina =< 60.
otpada(Ime) :- vojnik(Ime, Visina, _), Visina>200.
```

### Primer04

/\* Baze cinjenica:

```
    stan(Porodica, Kvadratura_stana).
    clan(Porodica, Broj_Clanova).
*/
```

```
stan(ciric, 76).
stan(pasic, 93).
stan(djokic, 55).
stan(music, 123).
stan(petrovic, 67).
```

```
clan(ciric,3).
clan(pasic,5).
clan(djokic,2).
clan(music,3).
clan(petrovic, 4).
```

```
po_clanu(Porodica, Prosek):- stan(Porodica, X), clan(Porodica, Y),
    Prosek is X/Y.
```

### Primer05

/\* Program za odredjivane broja dana u mesecu

```
    Poziva se sa:
    dana_u_mesecu(Dan, Mesec, Godina).
*/
```

dana\_u\_mesecu(31, januar, \_).

dana\_u\_mesecu(29, februar, G):- prestupna(G).

dana\_u\_mesecu(28, februar, G):- not(prestupna(G)).

dana\_u\_mesecu(31, mart, \_).

dana\_u\_mesecu(30, april, \_).

dana\_u\_mesecu(31, maj, \_).

dana\_u\_mesecu(30, juni, \_).

dana\_u\_mesecu(31, juli, \_).

dana\_u\_mesecu(31, avgust, \_).

dana\_u\_mesecu(30, septembar, \_).

dana\_u\_mesecu(31, oktobar, \_).

dana\_u\_mesecu(30, novembar, \_).

dana\_u\_mesecu(31, decembar, \_).

prestupna(G) :- je\_deljivo(G,400).

prestupna(G) :- not(je\_deljivo(G,100)), je\_deljivo(G,4).

je\_deljivo(X,Y) :- 0 is X mod Y.

---

## Zadaci za samostalan rad

### 1. Pogledaj primer primer01.pl

Definisati pravila: majka, baba, deda, dete, predak, sestra, brat, sric, ujak, imati bar dvoje dece.

### 2. Ako su date cinjenice oblika:

ucenik( Sifra\_ucenika, Ime\_ucenika, Odeljenje)

ocene( Sifra\_ucenika, Sifra\_predmeta, Ocena)

predmet( Sifra\_predmeta, Naziv\_predmeta, Broj\_cas)

Napisati pravila

a) sif\_5(S) u znacenju ucenik cija je sifra S ima bar dve petice.

b) bar\_dve\_petice(X) ucenik cije je ime X ima bar dve petice.

c) odeljenje\_5(X,Y) u odeljenju X postoje bar dve petice iz predmeta Y.

### 3. Neka je data baza znanja u Prologu cinjenicama:

film( Naziv\_filma, Zanr\_filma, Ime\_reditelja, Sifra\_glumca)

glumac( Sifra\_glumca, Ime\_glumca, God\_rodj, Mesto\_rodj)

a) Napisati pravilo filmski\_umetnik(X)

X je filmski\_umetnik ako je X reditelj nekog filma i X igra u nekom filmu.

b) Napisati pravilo glumac\_2(X)

X igra u bar dva razlicita filma

c) Napisati pravilo opsti\_glumac(X)

X igra u bar dva filma razlicitog zanra

d) Napisati pravilo zanrovski\_glumac(X,Y)

glumac cije je ime X igra u filmu zanra Y.

### 4. Date su cinjenice roditelj(X,Y) i godina\_rodjenja(X,G).

- a) Napisati pravilo naslednik(X,Y)  
osoba X je naslednik osobe Y
- b) Napisati pravilo bar\_dva(X,Y,Z)  
osoba X ima dva naslednika Y i Z rodjena iste godine.
- c) Napisati pravilo predak\_c(X,Y,G)  
osoba Y je predak osobe X rođen godine G.

5. Učenici nekog odeljenja nalaze se u koloni po jedan po visinama, u rastućem poretku.

Date su činjenice

pored\_d(X,Y) - desno pored osobe X u koloni je osoba Y

godina(X,Y) - osoba X rođena je godine Y

Napisati pravilo

- a) pa(X,Y) - X je osoba koja je niza od osobe Y
- b) pb(X,Y) - X je osoba koja je niza od osobe Y a rođene su iste godine
- c) pc(X,Y,Z) - osobe Y i Z su dve različite osobe koje su nize od osobe X

6. Date su činjenice

brzi(SX,SY) - automobil sifre SX brzi je od automobila SY

auto(Naziv\_automobila, Sifra\_automobila)

vlasnik(Ime\_vlasnika, Naziv\_automobila)

Napisati pravilo

- a) brziauto(X,Y)  
automobil naziva X je brzi od automobila naziva Y
- b) imaauto(X)  
lice čije je ime X ima automobil
- c) imabrzi(X,Y)  
X je vlasnik brzeg automobila nego što je Y

7. Date su činjenice, koje nam govore za svaka dva susedna čina u vojsci koji je viši.

visi\_cin(zastavnik,vodnik).

visi\_cin(major,zastavnik).

visi\_cin(pukovnik,major).

visi\_cin(generalmajor,pukovnik).

visi\_cin(generalpukovnik,generalmajor).

visi\_cin(general,generalpukovnik).

Napisati pravilo blizi(X,Y) u značenju čin X je bliži generalskom činu od čina Y.

Zadaci za razmišljanje

1. Definisati predikat kojim se za dati prirodan broj N određuje  $N! = 1 * 2 * \dots * N$ .
2. Definisati predikat kojim se za dati prirodan broj N određuje
  - a) broj cifara
  - b) suma cifara
  - c) maksimalna cifra.
3. Definisati predikat kojim se određuje N-ti stepen (N prirodan broj) celog broja A.
4. Definisati predikat kojim se određuje suma delioca broja N, ne uključujući broj N.
5. Definisati predikat kojim se proverava da li je dati prirodan broj savršen jednak sumi svojih delioca (ne uključujući broj N).
6. Definisati predikat kojim se proverava da li je dati prirodan broj prost.
7. Odredi broj neparnih cifara u prirodnom broju N.
8. Definiši predikat kojim se određuje suma parnih i suma neparnih delioca datog prirodnog broja N.

suma parnih delioca broja 12 je 24 ( $2+4+6+12$ ), a suma neparnih delioca je 4 ( $1+3$ )