

Deklaratív Programozás  
3. Prolog gyakorlat  
Prolog meta-logikai eljárások, ZH mintafeladatok

Hasznos beépített eljárások

"univ" azaz =.. /2 -- kifejezések szétszedése és összerakása

Hívási módok:

+Kif =.. ?Lista

-Kif =.. +Lista

Jelentése: igaz, ha

o Kif = Str(A1,...,An) és Lista = [Str,A1,...,An],

ahol Str egy névkonstans és A1,...,An tetszőleges kifejezések; vagy

o Kif = C, és Lista = [C], ahol C egy (név- vagy szám-)konstans.

Mintapélda: formula behelyettesítési értékének kiszámítása (350. fólia)

atom\_codes/2 -- atomok szétszedése és összerakása

Hívási módok:

atom\_codes(+Atom, ?Codes)

Atom - tetszőleges névkonstans

atom\_codes(-Atom, +Codes)

Codes - karakterkódok listája.

Jelentése, az Atom névkonstanst alkotó karakterek listája Codes.

Példák:

```
| ?- atom_codes(a0b, L).          ----> L = [97,48,98] ? ; no
| ?- atom_codes(A, [98,48,97]).  ----> A = b0a ? ; no
```

Meta-logikai beépített eljárásokkal kapcsolatos feladatok

1. Atomok szeletelése

Egy A atom prefixumának nevezünk egy P atomot, ha P az A első valahány karakterét tartalmazza, az A-beli sorrend megtartásával.

% atom\_prefix(+Atom, ?Prefix, +N): Atom-nak Prefix N hosszú prefixuma.

% Másszóval: az Atom első N karakteréből képzett névkonstans a Prefix atom.

```
| ?- atom_prefix(abcde, Prefix, 0).  ----> Prefix = '' ? ; no
| ?- atom_prefix(abcde, Prefix, 3).  ----> Prefix = abc ? ; no
| ?- atom_prefix(abcde, Prefix, 5).  ----> Prefix = abcde ? ; no
| ?- atom_prefix(abcde, Prefix, 6).  ----> no
```

Nem használhatja a sub\_atom/5 beépített eljárást!

Ötlet: használja az atom\_codes/2, length/2, append/3 beépített eljárásokat.

2. Általános Prolog kifejezés bizonyos részkifejezéseinek felsorolása

% reszatom(+K, ?A): A a K általános Prolog kifejezésben előforduló atom.

```
| ?- reszatom(a, X).                ----> X = a ? ; no
| ?- reszatom(f(X,[1,3,b],g(2,1,a0)), A).  ----> A = b ? ; A = [] ? ;
                                           A = a0 ? ; no
```

Megjegyzés: a struktúranevet nem tekintjük a struktúrakifejezés reszatomjának.

Segítségként magyar nyelven megfogalmazunk egy kijelentést, amely Prologba átírható:

Az "A egy a K kifejezésben előforduló atom" állítás két esetben állhat fenn

1. K maga egy atom -- ilyenkor A = K

2. K összetett, argumentumlistája KL, és KL-nek van olyan K1 eleme, hogy "A1 egy a K1 kifejezésben előforduló atom" teljesül -- ilyenkor A = A1

## 3. Általános Prolog kifejezés bizonyos részkifejezéseinek akkumulálása

```
% osszege(+K, ?Ossz): Ossz a K kifejezésben előforduló egész számok
% összege.
```

```
| ?- osszege(a, S).          ----> S = 0 ? ; no
| ?- osszege(1, S).         ----> S = 1 ? ; no
| ?- osszege(f(X,[1,3,b],g(2,1,a0)), S). ----> S = 7 ? ; no
```

-----

A következő feladatsor példa arra, hogy a Prolog NZH-ban milyen "egyklózos-programozás" jellegű feladatokat kell megoldani.

-----

"Egyklózos-programozás" jellegű NZH mintafeladatok

=====

Az alábbi feladatok megoldásaként egyetlen \*nem rekurzív\* klózból álló predikátumot kell leírni.

Ha a felsorolás sorrendjét a feladatban nem említjük, akkor ez a sorrend tetszőleges lehet.

Az alábbi beépített ill. könyvtári eljárások használhatók:

```
append/3, append/2, member/2, select/3, reverse/2, findall/3, bagof/3,
setof/3, sumlist/2, valamint az aritmetikai beépített eljárások.
```

A Prolog vezérlési szerkezetei (diszjunkció, negáció, feltételes kifejezés) szabadon használhatók.

Törekedjék minél tömörebb megoldásra!

## 4. Adott bemeneti számlista esetén jobbról balra haladva sorolja fel a lista 7-nél határozottan nagyobb elemeit!

```
% f4(L, X): X az L lista olyan eleme, amely határozottan nagyobb 7-nél.
% A megoldásokat jobbról balra sorolja fel!
```

```
| ?- f4([1,9,2,8,3,7,12], X).
X = 12 ? ;
X = 8 ? ;
X = 9 ? ;
no
```

## 5. Adott bemeneti számlista esetén balról jobbra haladva sorolja fel a lista szélsőértékeit, azaz azokat az elemeket, amelyeknek mindkét oldalán van szomszédja, és amelyek vagy mindkét közvetlen szomszédnál határozottan nagyobbak, vagy mindkettőnél határozottan kisebbek!

```
% f5(L, X): X az L lista szélsőértéke.
% A megoldásokat balról jobbra sorolja fel!
```

```
| ?- f5([1,9,2,8,3,7,12], X).
X = 9 ? ;
X = 2 ? ;
X = 8 ? ;
X = 3 ? ;
no
```

6. Adott bemeneti számlista esetén sorolja fel az összes olyan A-B párt, amelyre A és B az adott listában közvetlenül szomszédos elemekként (ebben a sorrendben) előfordulnak és amelyre az A+B összeg értéke 5!

% f6(L, A-B): A és B két olyan szomszédos eleme az L számlistának,  
% amelyek összege 5.

```
| ?- f6([1,2,3,2,4], X).
X = 2-3 ? ;
X = 3-2 ? ;
no
```

7. Adott bemeneti számlista esetén sorolja fel az összes olyan h(A,S,B) struktúrakifejezést, amelyre A, S és B a bemeneti listában (nem feltétlenül szomszédos) elemekként ebben a sorrendben előfordulnak és amelyre az A+B összeg értéke S!

% f7(L, h(A,S,B)): Az A, S, B számok ebben a sorrendben (de nem  
% feltétlenül szomszédosan) előfordulnak az L listában, és A+B=S.

```
| ?- f7([1,5,4,7,3,7,-1], X).
X = h(1,5,4) ? ;
X = h(1,4,3) ? ;
X = h(5,4,-1) ? ;
X = h(4,7,3) ? ;
X = h(4,3,-1) ? ;
no
```

8. Adott L bemeneti egészlista esetén állítsa elő a listában előforduló páros számok összegét!

% f8(L, S): S az L egészlista páros elemeinek összege.

```
| ?- f8([1,9,2,8,3,7,12], S).
S = 22 ? ;
no
```

---

A következő feladatok bemutatják, hogy a Prolog NZH-ban milyen jellegű programozási feladatokat kell megoldani, milyen szerkezetben.

Két, egymásra épülő feladatot kell kidolgozni: először egy segédeljárást kell elkészíteni, majd ezt felhasználva egy, a teljes feladat megoldását előállító predikátumot kell megírni.

---

Prolog programozási NZH M1 és M2 mintafeladatok  
 =====

9. M1 mintafeladat segédeljárása:

Írjon Prolog nyelven egy olyan eljárást, amely előállítja egy konstans értékét egy helyettesítési lista alapján. A helyettesítési lista minden eleme Név-Szám alakú, ahol Szám a Név névkonstans helyettesítési értéke. Egy számkonstans helyettesítési értéke önmaga, egy a helyettesítési listában nem szereplő atom helyettesítési értéke pedig 0. Ha egy névkonstans többször szerepel a helyettesítési listában, akkor az első előfordulást szabad csak figyelembe venni.

% helyettesitese(+K, +HL, ?E): A K konstansnak a HL behelyettesítési  
 % lista szerinti értéke E.

Példák:

```
?- helyettesitese(y, [x-1,y-2,z-3], H).      ---->  H = 2 ? ; no
?- helyettesitese(u, [x-1,y-2,z-3], H).      ---->  H = 0 ? ; no
?- helyettesitese(x, [x-1,z-3,x-2], H).      ---->  H = 1 ? ; no
?- helyettesitese(4, [x-1,y-2,z-3], H).      ---->  H = 4 ? ; no
```

10. M1 teljes feladat:

A helyettesitese/3 eljárás segítségével írjon olyan Prolog eljárást, amely egy többváltozós kifejezés adott lista szerinti behelyettesítési értékét számítja ki! A kifejezést egy olyan Prolog adatstruktúrával adjuk meg, amely atomokból és számokból az 'is' beépített eljárás által megengedett egy- ill. kétargumentumú műveletekkel épül fel.

% erteke(+Kif, +Hely, ?Ert): A Kif kifejezés értéke a Hely behelyettesítési  
 % lista által adott helyen Ert.

Példák:

```
?- erteke(x, [x-22], E).      ---->  E = 22 ? ; no
?- erteke(-x, [x-5], E).      ---->  E = -5 ? ; no
?- erteke(33, [x-22], E).      ---->  E = 33 ? ; no
?- erteke((x+y)*(x+y)+1, [x-1,y-2], E).      ---->  E = 10 ? ; no
?- erteke(x*abs(z)+x, [x-1,z-(-2)], E).      ---->  E = 3 ? ; no
```

-----  
 Az M2 mintafeladathoz felidézünk két Prolog szintaktikus édesítőszert  
 -----

Egy 'k' karakter kódja a 0'k jelöléssel írható le Prologban. Például a 0'a karaktersorozat a kis a betű ASCII kódját jelöli, ennek megfelelően beolvasáskor a 97 értékű egész számmá alakítódik (tehát ez a jelölés csak egy "szintaktikus édesítőszert").

A jelölés bemutatására adunk egy példát, amely az M2 feladat megoldásában is felhasználható.

```
% kisbetu(K) : K egy ASCII kisbetű kódja.
kisbetu(K) :-
    K >= 0'a, K <= 0'z.
```

Egy további szintaktikus édesítőszert a fűzér: a " jelek közé tett szöveget a Prolog rendszer beolvasáskor egy olyan listává alakítja amely a szöveg karaktereinek kódjait tartalmazza, "ab1" ugyanaz mint [0'a,0'b,0'1], ami ugyanaz mint [97,98,49].

(SWI Prologban a " jelek közé tett szöveg egy újfajta konstans, string lesz.)

A 11. feladat futási példái által kiadott behelyettesítéseket -- a könnyebb olvashatóság érdekében -- fűzér alakban is megadjuk.

## 11. M2 mintafeladat segédeljára:

Írjon olyan Prolog eljárást, amely egy karakterkódokból álló listát két részre vág! Az első rész legyen a lista olyan maximális hosszú kezdőszelete, amelyben minden elem egy ASCII kisbetű kódja, feltéve, hogy ez a kezdőszelet legalább kételemű. A másik rész legyen a lista fennmaradó része. Ha a lista nem kisbetű-kóddal kezdődik, vagy csak egyetlen kisbetű-kód áll az elején, akkor az eljárás hiúsuljon meg!

```
% kezdő_szava(+L, ?Kezdet, ?Maradék): Kezdet az L karakterkód-lista maximális
% hosszú csak kisbetű-kódokat tartalmazó kezdőszelete, amely legalább
% kételemű. Maradék az L-ben Kezdet után álló elemek listája.
```

Példák:

(A `_Cs` változó behelyettesítését a Prolog rendszer nem írja ki, mert a változónév aláhúzásjellel kezdődik)

```
| ?- atom_codes(ab1, _Cs), kezdő_szava(_Cs, K, M).
----> K = [97,98], M = [49] ? ; no
      % K = "ab", M = "1" ? ; no
| ?- atom_codes(abrak_adabra, _Cs), kezdő_szava(_Cs, K, M).
----> K = [97,98,114,97,107], M = [95,97,100,97,98,114,97] ? ; no
      % K = "abrak", M = "_adabra" ? ; no
| ?- atom_codes('a-1nn', _Cs), kezdő_szava(_Cs, K, M). ----> no
| ?- atom_codes('aBCde', _Cs), kezdő_szava(_Cs, K, M). ----> no
| ?- atom_codes('', _Cs), kezdő_szava(_Cs, K, M). ----> no
```

## 12. M2 teljes feladat:

A `kezdő_szava/3` predikátum segítségével írjon olyan Prolog eljárást, amely egy adott atomban keres egy abban előforduló legalább kétkarakteres olyan folytonos részatomot, amely csupa kisbetűből áll, és maximális, azaz egyik irányban sem terjeszthető ki kisbetűvel! Az eljárás adja ki a megtalált részatom kezdő indexpozícióját is (1-től számozva)! Visszalépéskor legyen hajlandó az összes ilyen részatomot felsorolni! A szavakat az előfordulásuk sorrendjében sorolja fel!

```
% szava(Atom, Szó, Index): Az Atom atomban az Index kezdőpozíción a Szó
% áll, amely csupa kisbetűből álló maximális, legalább kétbetűs atom.
```

Példák:

```
| ?- szava(szia, Sz, I).
I = 1, Sz = szia ? ; no

| ?- szava('Szia világ!', S, I).
I = 2, S = zia ? ;
I = 6, S = világ ? ; no

| ?- szava('Az eros gyógy* 6ott', S, I).
I = 4, S = eros ? ;
I = 9, S = gyógy ? ;
I = 17, S = ott ? ;
no
```