

Deklaratív Programozás gyakorlat
Prolog programozás: meta-logikai eljárások
2015.10.19 és 22

Az alábbi feladatok megoldásában, ha másként nem mondjuk, használhat segédeljárásokat, de ezekhez mindig adjon meg fejkommentet. Megoldásában mindig felhasználhatja az előző feladatokhoz megírt eljárásokat.

Meta-logikai beépített eljárásokkal kapcsolatos feladatok
=====

Emlékeztető - egyes beépített eljárások leírása

=.. /2 azaz az univ eljárás

+Kif =.. ?Lista
-Kif =.. +Lista

Kif - Az argumentum egy tetszőleges kifejezés.
Lista - Az argumentum egy lista, az első eleme egy név vagy egy szám, a többi eleme tetszőleges kifejezés. A lista első eleme csak akkor lehet szám, ha több eleme már nincsen.

Igaz, ha Kif = StrNev(A₁, ..., A_n) és Lista = [StrNev, A₁, ..., A_n].

var(X): X (behelyettesíthető) változó
nonvar(X): X nem változó
compound(X): X struktúra (összetett kifejezés)
atom(X): X névkonstans
integer(X): X egész szám

atom_codes/2: atomok szétszedése és összerakása

atom_codes(+Atom, ?Codes)
atom_codes(-Atom, +Codes)
Atom - tetszőleges névkonstans
Codes - karakterkódok listája.

Igaz, ha az Atom névkonstans alkotó karakterek listája Codes.

Példák:

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

1. Atomok szeletelése

Egy A atom szuffixumának nevezünk egy S atomot, ha S az A utolsó valahány karakterét tartalmazza, az A-beli sorrend megtartásával.

% atom_suffix(+Atom, ?Suffix, +Before): Az Atom névkonstansból a Suffix % atom úgy áll elő, hogy A elejéről elhagyunk Before számú karaktert.

```
| ?- atom_suffix(abcde, Suffix, 3).  
Suffix = de ? ;  
no  
| ?- atom_suffix(abcde, Suffix, 5).  
Suffix = '' ? ;  
no  
| ?- atom_suffix(abcde, Suffix, 6).  
no
```

(Nem használhatja a sub_atom/5 beépített eljárást.)

2. Szimbolikus aritmetikai kifejezések kiértékelése

Egy Prolog kifejezést nevezünk szimbolikus aritmetikai kifejezésnek, ha az az 'x' névkonstansból és számokból az is/2 beépített eljárás által megengedett nulla-, egy- és kétargumentumú struktúrakifejezésekkel épül fel.

Példák szimbolikus aritmetikai kifejezésekre:

```
-x+3*(x/2+1/x)  
max(x/exp(pi*x/2), 10)
```

A fenti példákban a pi egy 0-argumentumú, a - és az exp 1-argumentumú, a +, /, max stb. 2-argumentumú olyan függvénykifejezés, amelyet az is/2 ill. az aritmetikai összehasonlító eljárások hajlandók kiérkélni (természetesen csak akkor, ha a kifejezés számokból épül fel, ami a szimbolikus aritmetikai kifejezésekre nem teljesül, hiszen azok az 'x' névkonstans is tartalmazhatják).

Az alábbi feladat megoldásában feltételezheti (azaz nem kell ellenőriznie), hogy a szimbolikus kifejezésben szereplő minden (x-től különböző) atom, valamint 1- és 2-argumentumú struktúra olyan, amit az is/2 beépített eljárás feldolgozni képes.

% erteke(+Kif, +X, ?E): A Kif szimbolikus aritmetikai kifejezés x=X % helyen vett értéke E.

```
| ?- erteke((x+1)*x+x+2*(x+x+3), 2, E).  
E = 22 ? ;  
no  
| ?- erteke(100**min(x,1/x), 2, E).  
E = 10.0 ? ;  
no  
| ?- erteke(-x+3*(x/2+1/x), 1, E).  
E = 3.5 ? ;  
no  
| ?- erteke(max(x*exp(pi*x/2), 10), 2, E).  
E = 46.281385265558534 ? ;  
no  
| ?- erteke(max(x*exp(pi*x/2), 10), 1, E).  
E = 10.0 ? ;  
no
```

3. Általános Prolog kifejezés részkifejezéseinek vizsgálata

% mern(+K, +N): A K általános Prolog kifejezésben előforduló összes egész % szám határozottan nagyobb mint N (mern = minden egész részkifejezése % nagyobb mint)

```
| ?- mern(1, 1).  
no  
| ?- mern(1, 0).  
yes  
| ?- mern(0.0, 1).  
yes  
| ?- mern(f(X,[1,3,b],g(2,1,3)), 0).  
yes  
| ?- mern(f(X,[1,3,b],g(2,1,3)), 1).  
no
```

Megjegyzések:

- A "K1 Prolog kifejezésben előfordul a K2 kifejezés" relációt reflexívnek tekintjük, azaz egy K kifejezésben önmaga mindenképpen előfordul. Ez a megjegyzés vonatkozik az ezután következő feladatokra is.
- Vigyázzon arra, hogy a kifejezésben változók is előfordulhatnak.

4. 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 részének.

5. Általános Prolog kifejezés bizonyos részkifejezéseinek akumulálása

```
% 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
```

6. Általános Prolog kifejezés bizonyos részkifejezéseinek átalakítása

```
% rovidített(+Kif0, ?Kif): Kif a Kif0 általános Prolog kifejezésből úgy  
% áll elő, hogy minden, benne argumentumként előforduló nem-0-hosszú  
% atom első karakterét elhagyjuk. (Az "argumentumként előforduló"  
% kifejezés arra utal, hogy a struktúraneveket változatlanul kell hagyni).
```

```
| ?- rovidített(abc, Kif).  
Kif = bc ? ;  
no  
| ?- rovidített(f(ab, X, b, gh(uv)), ''), Kif).  
Kif = f(b,X,'',gh(v),'').  
no  
| ?- _Kif0=[abc], rovidített(_Kif0, Kif),  
write_canonical(_Kif0), nl, write_canonical(Kif).  
'.'(abc,[])  
'.'(bc,'')  
Kif = [bc|''] ? ;  
no
```