

Deklaratív Programozás gyakorlat  
Prolog programozás: listakezelés, az univ eljárás  
2013.10.24.

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

Listakezeléssel kapcsolatos feladatok  
=====

1. Beszúrás rendezett listába

```
% insert_ord(+RL0, +Elem, ?RL): Az RL monoton növő számlista úgy áll
% elő, hogy az RL0 szigorúan növő számlistába beszúrjuk az Elem számot,
% feltéve hogy Elem nem eleme az RL0 listának; egyébként RL = RL0.
```

```
| ?- insert_ord([1,3,5,8], 6, L).
L = [1,3,5,6,8] ? ; no
| ?- insert_ord([1,3,5,8], 3, L).
L = [1,3,5,8] ? ; no
```

Használjon feltételes szerkezetet!

Szorgalmi feladatként adhat olyan megoldást is, amelyben vágót használ. Tartsa be a vágás alapszabályát, próbáljon meg minél kevesebb klózból álló megoldást adni.

-----

A 'graph' adatstruktúrát a következő Mercury-szerű típusdefiníciókkal definiáljuk:

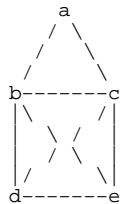
```
% :- type graph == list(edge).
% :- type edge ---> node-node.
% :- type node == atom.
```

Eszerint egy Prolog kifejezés a 'graph' típusba tartozik, ha X-Y alakú struktúrák listája, ahol X és Y névkonstansok (atomok).

Az [a1-b1,a2-b2,...,an-bn] 'graph' típusú kifejezés azt az irányítatlan gráfot írja le, amelynek csomópontjai a1,..,an,b1,..,bn, és egy (irányítatlan) él vezet ai és bi között, minden i=1,..,n esetén. (Megjegyzés: az így megadott gráfoknak nyilván nem lehet izolált pontja.)

Például az [a-b,a-c], [a-c,b-a], [b-a,a-c], [c-a,a-b] stb. mind ugyanazt a (matematikai értelemben vett) irányítatlan gráfot írják le.

Az [a-b,a-c,b-c,b-d,b-e,c-d,c-e,d-e] kifejezés az alábbi gráfot írja le:



Gyerekkorukban találkozhattak azzal a feladattal, hogy ezt a gráfot egy folytonos vonallal rajzolják meg.

Egy 'graph' típusú [a1-b1,a2-b2,...,an-bn] Prolog listát folytonos vonalnak hívunk, ha b1=a2, b2=a3, ..., b(n-1) = an.

2. Írja meg az alábbi fejkommentnek megfelelő draw/2 Prolog eljárást

```
% draw(+G, -L): Az L folytonos vonal "megrajzolja" a G gráfot, azaz az
% L folytonos vonal ugyanazt a matematikai értelemben vett gráfot írja
% le, mint a G Prolog kifejezés.
```

Tehát a "| ?- draw(G, L)." hívás, ahol G adott és L egy változó, felsorolja L-ben az összes olyan folytonos vonalat, amely "megrajzolja" G-t.

Törekedjék minél egyszerűbb megoldásra, nem kell a hatékonysággal foglalkoznia. Használhatja a lists könyvtár eljárásait.

```
| ?- draw([a-b,a-c], L).
L = [b-a,a-c] ? ;
L = [c-a,a-b] ? ;
no
| ?- draw([a-b,a-c,b-c,b-d,b-e,c-d,c-e,d-e], L), L = [d-e|_].
L = [d-e,e-b,b-a,a-c,c-b,b-d,d-c,c-e] ? ;
L = [d-e,e-b,b-a,a-c,c-d,d-b,b-c,c-e] ? ;
L = [d-e,e-b,b-c,c-a,a-b,b-d,d-c,c-e] ? ;
L = [d-e,e-b,b-c,c-d,d-b,b-a,a-c,c-e] ? ;
L = [d-e,e-b,b-d,d-c,c-a,a-b,b-c,c-e] ? ;
L = [d-e,e-b,b-d,d-c,c-b,b-a,a-c,c-e] ? ;
L = [d-e,e-c,c-a,a-b,b-c,c-d,d-b,b-e] ? ;
L = [d-e,e-c,c-a,a-b,b-d,d-c,c-b,b-e] ? ;
L = [d-e,e-c,c-b,b-a,a-c,c-d,d-b,b-e] ? ;
L = [d-e,e-c,c-b,b-d,d-c,c-a,a-b,b-e] ? ;
L = [d-e,e-c,c-d,d-b,b-a,a-c,c-b,b-e] ? ;
L = [d-e,e-c,c-d,d-b,b-c,c-a,a-b,b-e] ? ;
no
```

3. Írjon Prolog eljárást egy (irányítatlan) gráf fokszámlistájának előállítására. A fokszámlista típusa:

```
% :- type degrees == list(node_degree).
% :- type node_degree --> node - degree.
% :- type degree == int.
```

A fokszámlista tehát egy olyan lista, amelynek elemei Cs-N alakú párok, ahol Cs a gráf egy csomópontja, és N a Cs csomópont fokszáma. A csomópontok tetszőleges sorrendben szerepelhetnek a fokszámlistában.

% degree\_list(G, Ds): A G gráf fokszámlistája Ds.

```
| ?- degree_list([b-a,a-c], Ds).
Ds = [b-1,a-2,c-1] ? ; no
```

4. (szorgalmi feladat)

Írjon idraw/2 néven egy Prolog eljárást, amelynek jelentése azonos a 2. feladatban szereplő draw/2 eljárással! Törekedjék minél hatékonyabb megoldásra! Használhatja a ugraphs könyvtárat.

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(A<sub>1</sub>, ..., A<sub>n</sub>) és Lista = [StrNev,A<sub>1</sub>,... A<sub>n</sub>].

-----

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.

-----

5. 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

(Nem használhatja a sub\_atom/5 beépített eljárást.)

6. 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(f(X,[1,3,b],g(2,1,0.0)), 0).

true ? ;

no

| ?- mern(f(X,[1,3,b],g(2,1,a0)), 1).

no

Megjegyzések:

a. 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.

b. Vigyázzon arra, hogy a kifejezésben változók is előfordulhatnak.

7. 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.

8. 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

9. Általános Prolog kifejezés bizonyos részkifejezéseinek átalakítása

% roviditett(+Kif0, ?Kif): Kif a Kif0 általános Prolog kifejezésből úgy áll % elő, hogy minden, benne argumentumként előforduló nem-üres atom első % karakterét elhagyjuk.

| ?- roviditett(abc, Kif).

Kif = bc ? ;

no

| ?- roviditett(f(ab, X, b, gh(uv)), ''), Kif).

Kif = f(b,X,'',gh(v),'').

no

| ?- \_Kif0=[abc], roviditett(\_Kif0, Kif),

write\_canonical(\_Kif0), nl, write\_canonical(Kif).

..'(abc,[])

..'(bc,[])

Kif = [bc|'] ? ;

no