

## *Logikai programozás*

2.

## BEMELEGÍTŐ

Definiáljuk a testvér (minimum féltestvér) fogalmát!

testvere(A,B) :- szuloje(C,A), szuloje(C,B), A \= B.

szuloje(Idos, Fialal) :- anyja(Idos, Fialal) ; apja(Idos, Fialal).

vagy:

szuloje(Idos, Fialal) :- anyja(Idos, Fialal).

szuloje(Idos, Fialal) :- apja(Idos, Fialal).

## MÉG EGYSZER A KOLLÉGISTÁK

Valaki kollégista, ha vidéki és jó tanuló.

Mit jelent a vidékiség?

Megoldandó: peldak/koleszos.pl

## MÉG EGYSZER A KOLLÉGISTÁK

videki(Diak,Kozpont) :- lakik(Diak, Varos), Varos \= Kozpont.

jotanulo(Diak,Hatar) :- atlag(Diak, Atl), Atl >= Hatar.

kolligista(Diak,Kozpont,Hatar) :- videki(Diak,Kozpont),  
jotanulo(Diak,Hatar).

kolligista(Diak,Kozpont,Hatar) :- lakik(Diak, Varos),  
Varos \= Kozpont,  
atlag(Diak, Atl),  
Atl >= Hatar.

## KOLLÉGISTÁK I/O-VAL

Beolvasás/kiíratással :

```
start :- write('Kérem a központot: '), read(Kozpont),
        write('Kérem az átlaghatárt: '), read(Hatar),
        kollegista(Diak, Kozpont, Hatar),
        writef("%w kollégista \n", [Diak]).
```

Hátrány: csak egyetlen megoldást ír ki.

Megoldás: A Prolog háttéralgoritmus + némi „kényszer”

## ELŐBB: A PROLOG MEGKÖZELÍTÉSI MÓDJAI

A Prolog logikai (deklaratív nyelv), szabályaira mégis tekinthetünk deklaratív és procedurális módon is:

### Deklaratív szemlélet:

Egy szabály egy **logikai következtetés**: a szabályfej pontosan akkor lesz igaz, ha bizonyos feltételek ÉS kapcsolata (szabálytörzs) igaz.

Az azonos fejű szabályok VAGY kapcsolatban állnak.

## ELŐBB: A PROLOG MEGKÖZELÍTÉSI MÓDJAI

A

B :- A<sub>11</sub>, A<sub>12</sub>, ..., A<sub>1n</sub>.

B :- A<sub>21</sub>, A<sub>22</sub>, ..., A<sub>2m</sub>.

...

B :- A<sub>r1</sub>, A<sub>r2</sub>, ..., A<sub>rk</sub>.

szabály-együttes a következő logikai állítással egyenértékű:

A B állítás igaz, ha

vagy az A<sub>11</sub>, A<sub>12</sub>, ..., A<sub>1n</sub> állítások mindegyike,

vagy az A<sub>21</sub>, A<sub>22</sub>, ..., A<sub>2m</sub> állítások mindegyike, ...

vagy az A<sub>r1</sub>, A<sub>r2</sub>, ..., A<sub>rk</sub> állítások mindegyike igaz.

## ELŐBB: A PROLOG MEGKÖZELÍTÉSI MÓDJAI

**Procedurális szemlélet:** Egy szabály **eljáráshívások sorozatának** is tekinthető.

Pontosabban: egy szabály egy eljárásnak felel meg. A szabályfejben szereplő predikátumnév az eljárás neve, argumentumai az eljárás paraméterei. Az eljárásban további eljárásokat hívunk meg, mégpedig a szabály-törzsben szereplő predikátumnevekkel azonosítható eljárásokat.

Az **azonos szabályfejű szabályok együttese** ugyanilyen módon **eljárásnak** tekinthető, csak ez az eljárás elágazást is tartalmaz, melyben az elágazás egyes ágai az egyes szabálytörzsek.

Belépési pont: A lekérdezés.

## A PROLOG (egyik) HÁTTÉRALGORITMUSA

Visszalépéses algoritmus (back track):

A kiértékelés elindul egy útvonalon, és addig tart, ameddig csak lehet:

- a/ sikeresen véget ért az útvonal;
- b/ hibás eredményhez jutunk.

A b/ esetben az algoritmus egyetlen lépést visszalép, és ha az előző hívásnak van alternatívája, akkor végrehajtja azt.

## A PROLOG (egyik) HÁTTÉRALGORITMUSA

Visszalépéses algoritmus:

Pl.:

$B :- (A_{11}, A_{12}, \dots, A_{1k-1}, A_{1k}, A_{1k+1}, \dots, A_{1n})$

$B :- A_{21}, A_{22}, \dots, A_{2m}$

...

## A KIÍRATÁSI PROBLÉMA MEGOLDÁSA

Visszalépésre kényszerítés:

```
start :- write('Kérem a központot: '), read(Kozpont),
         write('Kérem az átlaghatárt: '), read(Hatar),
         kollegista(Diak, Kozpont, Hatar),
         writef('%w kollégista \n', [Diak]), fail.
```

start.

Más írásmóddal:

```
start :- write('Kérem a központot: '), read(Kozpont),
         write('Kérem az átlaghatárt: '), read(Hatar),
         kollegista(Diak, Kozpont, Hatar),
         writef('%w kollégista \n', [Diak]), fail; start.
```

## CSALÁDFA

Tegyük fel, hogy csak apák és fiúk vannak. Ismerjük az apa-fiú kapcsolatokat, írjuk fel:

nagyapja(Idos, Fialtal) :- apja(Idos, Z), apja(Z, Fialtal).

dedapja(Idos, Fialtal) :- apja(Idos, Z), nagyapja(Z, Fialtal).

ukapja(Idos, Fialtal) :- apja(Idos, Z), dedapja(Z, Fialtal).

szepapja(Idos, Fialtal) :- ...

ose(Idos, Fialtal) :- apja(Idos, Z), ose(Z, Fialtal).

## CSALÁDFA

Rekurzió: A szabálytörzs a szabályfejen lévő predikátumra hivatkozik.

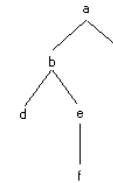
A rekurzió **mindig két részből** áll:

- rekurzív szabály
- megállító feltétel.

$ose(Iidos, Fiatal) :- apja(Iidos, Fiatal).$

$ose(Iidos, Fiatal) :- apja(Iidos, Z), ose(Z, Fiatal).$

## CSALÁDFA - PÉLDA



Írjuk fel a mellékelt családfát!

$apja(a,b).$   
 $apja(a,c).$   
 $apja(b,d).$   
 $apja(b,e).$   
 $apja(e,f).$   
 $ose(X,Y) :- apja(X,Y).$   
 $ose(X,Y) :- apja(X,Z), ose(Z,Y).$

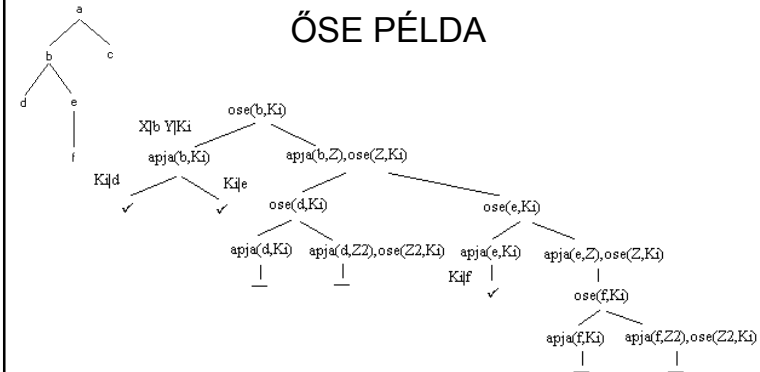
peldak/os.pl kipróbálása (nyomkövetéssel is)

## KIÉRTÉKELÉS: ILLESZTÉSSEL

Az illesztés feltételei:

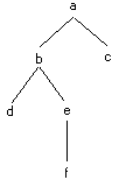
- Csak azonos nevű predikátumok illeszthetők.
- Csak azonos argumentumszámú predikátumok illeszthetők.
- Konstans saját magával illeszthető.
- Változó konstanssal vagy másik változóval illeszthető.

## ŐSE PÉLDA



$apja(a,b).$   
 $apja(a,c).$   
 $apja(b,d).$   
 $apja(b,e).$   
 $apja(e,f).$   
 $ose(X,Y) :- apja(X,Y).$   
 $ose(X,Y) :- apja(X,Z), ose(Z,Y).$

## CSALÁDFA - MÓDOSÍTÁS

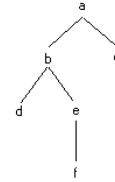


apja(a,b).  
apja(a,c).  
apja(b,d).  
apja(b,e).  
apja(e,f).  
ose(X,Y) :- apja(X,Y).  
ose(X,Y) :- apja(X,Z), ose(Z,Y).

Jó-e így a második szabály?:

ose(X,Y) :- ose(X,Z), apja(Z,Y).

## CSALÁDFA - MÓDOSÍTÁS



apja(a,b).  
apja(a,c).  
apja(b,d).  
apja(b,e).  
apja(e,f).  
ose(X,Y) :- apja(X,Y).  
ose(X,Y) :- apja(X,Z), ose(Z,Y).

És mi történik, ha a szabályok sorrendjét is felcseréljük?:

ose(X,Y) :- ose(X,Z), apja(Z,Y).  
ose(X,Y) :- apja(X,Y).