Домашна задача 3

Тијана Атанасовска (196014)

Задача 1

Првин се поставуваат фактите за основните информации. Кој е личност, која храна се користи, кое хоби и каква боја на маица. Како и во било која база на податоци, мора да се креира еден затворен свет кој ќе се обработува.

```
licnost(mira).
licnost(teo).
licnost(bruno).
licnost(igor).

hrana(piza).
hrana(pita).
hrana(hamburger).
hrana(sendvic).

hobi(krstozbor).
hobi(pisuva).
hobi(cita).
hobi(fotografija).

boja(bela).
boja(crvena).
boja(sina).
boja(zolta).
```

Потоа се поставуваат посложените факти кои се релација меѓу основните (кој која храна ја јаде или кое хоби го има). Овие факти се директно изложени во речениците кои се зададени во задачата, а не изведени! Дури и предикатот за машко враќа нешто само ако тоа не е женско.

***Овде може да се зададе и услов дека предикатот се повикува само на личности, но во овој случај нема да се прави злоупотреба на предикатот.**

```
21 e_zensko(mira).
22 e_masko(X):-not(e_zensko(X)).
23
24 %ima_maica(Ime,Boja)
25 ima_maica(Ime,bela):-e_zensko(Ime).
26 ima_maica(bruno,zolta).
27
28 %ima_hobi(Ime,Hobi)
29 ima_hobi(mira,krstozbor).
30 ima_hobi(igor,cita).
31
32 %jade(Ime,Hrana)
33 jade(mira,pita).
34 jade(teo,sendvic).
35
```

Позициите за седење се претставени од 0 до 3, така што 0 = најлево.

Фактите: Teo седи најлево и Личноста која јаде пита седи покрај Teo, се претставени со следниот код. Teo е соодветно на позиција 0 (најлево). За да се дознае кој седи на позицијата до Teo се користат следните два предикати, така што се опфатени случаите личноста која јади пита да седи лево или десно од Teo, бидејќи не знаеме точно. Но, притоа се прави проверка дали позицијата која ќе и се додели на личноста која јаде пита е во рангот на возможни вредности за седење, односно [0,3]. На овој начин, вториот предикат никогаш нема да е точен, односно личноста која јаде пита не може да седе лево од Teo бидејќи тоа би било позиција -1.

```
36 sedi(teo,0).
37 sedi(Ime,BrPozicija):-jade(Ime,pita), sedi(teo,PozTeo),
38     BrPozicija is (   PozTeo+1), BrPozicija=<3.
39 sedi(Ime,BrPozicija):-jade(Ime,pita), sedi(teo,PozTeo),
40     BrPozicija is (   PozTeo-1), BrPozicija>=0.
```

Факт: Сина маица има личноста која седи десно од девојката.

Десно од девојката е позицијата на девојката - 1.

```
42 ima_maica(Ime,sina):-sedi(Ime,BrPozicija),e_zensko(Y),sedi(Y,PozicijaY),
43     BrPozicija is (PozicijaY-1).
44
```

Останатите возможни комбинации за кои немаме точни факти да ги кодираме, се пронаоѓаат со следните предикати.

Пример, за првиот предикат за седење, се изминуваат сите возможни позиции за седење и сите личности од базата. Доколку не постои запис дека некој седи на дадена позиција или дека лицето има место за седење, тогаш се генерира таква комбинација. Важно е да опфатат и двата случаи!

```
46 sedi_other(Ime,BrPozicija):-findall(L,licnost(L),LLicnost),
47                             member(BrPozicija,[0,1,2,3]), member(Ime,LLicnost),
48                             not(sedi(Ime,_)), not(sedi(_,BrPozicija)).
49
50 jade_other(Ime,Hrana):-findall(L,licnost(L),LLicnost),findall(H,hrana(H),LHrana),
51                         member(Ime,LLicnost),member(Hrana,LHrana),
52                         not(jade(Ime,_)), not(jade(_,Hrana)).
53
54
55 ima_maica_other(Ime,Boja):-findall(L,licnost(L),LLicnost),findall(B,boja(B),LBoja),
56                            member(Ime,LLicnost),member(Boja,LBoja),
57                            not(ima_maica(Ime,_)),not(ima_maica(_,Boja)).
58
59 hobi_other(Ime,Hobi):-findall(L,licnost(L),LLicnost),findall(H,hobi(H),LHobi),
60                       member(Ime,LLicnost),member(Hobi,LHobi),
61                       not(ima_hobi(Ime,_)),not(ima_hobi(_,Hobi)).
62
```

Со повик на првиот предикат се враќа одговор:

```
Ime = bruno,
Pozicija = 2
Ime = igor,
Pozicija = 2
Ime = bruno,
Pozicija = 3
Ime = igor,
Pozicija = 3
```

```
?- sedi_other(Ime,Pozicija)
```

- Бидејќи нема точна информација за нив двајцата па можно е било кој да седи на позиција 2 или позиција 3. Понатаму во задачата со други услови ќе се пронајде што е точното решение.

До овој момент фактите за седење и податоците за седење кои ги пресметавме како возможни комбинации се во различни предикати. За да се соберат во еден предикат се

користи sedi_all. Истото се прави и за кој што јаде, кој какво хоби има, или боја на маица.

```
63 sedi_all(Ime,Pozicija):-sedi(Ime,Pozicija);sedi_other(Ime,Pozicija).
64
65 jade_all(Ime,Hrana):-jade(Ime,Hrana);jade_other(Ime,Hrana).
66
67 hobi_all(Ime,Hobi):-ima_hobi(Ime,Hobi);hobi_other(Ime,Hobi).
68
69 maica_all(Ime,Boja):-ima_maica(Ime,Boja);ima_maica_other(Ime,Boja).
70
```

Кога се повикува за седење, се добива следниот одговор:

```
Ime = teo,
Pozicija = 0
Ime = mira,
Pozicija = 1
Ime = bruno,
Pozicija = 2
Ime = igor,
Pozicija = 2
Ime = bruno,
Pozicija = 3
Ime = igor,
Pozicija = 3
```

```
?- sedi_all(Ime,Pozicija)
```

- За позиција 0 и 1 имаме точно по 1 вредност (бидејќи се факти), но за 2 и 3 има повеќе возможни.

До овде го претставивме целото знаење кое го имаме со фактите, без да вметне имплицитно знаење!

Задача 2

За да се најде решение на задачата потребно е да ги кодираме останатите услови да важат.

За да се најде комбинација од Име, Храна, Хоби и Боја со моменталните податоци се користи reshenie_eden. Со предикатот po_pozicija се групираат во листа сите комбинации за една позиција.

```
reshenie_eden([BrPozicija,Ime,Hrana,Hobi,Boja]):-member(BrPozicija,[0,1,2,3]),
    sedi_all(Ime,BrPozicija),jade_all(Ime,Hrana),
    hobi_all(Ime,Hobi),maica_all(Ime,Boja).


po_pozicija(L,BrPozicija):-findall((BrPozicija,Ime,Hrana,Hobi,Boja),
                                   reshenie_eden([BrPozicija,Ime,Hrana,Hobi,Boja]),L).
```

Излез:

```
L = [ (0,teo,sendvic,pisuva,sina),
      (0,teo,sendvic,fotografija,sina),
      (1,mira,pita,krstozbor,bela),
      (2,bruno,piza,pisuva,zolta),
      (2,bruno,piza,fotografija,zolta),
      (2,bruno,hamburger,pisuva,zolta),
      (2,bruno,hamburger,fotografija,zolta),
      (2,igor,piza,cita,crvena),
      (2,igor,hamburger,cita,crvena),
      (3,bruno,piza,pisuva,zolta),
      (3,bruno,piza,fotografija,zolta),
      (3,bruno,hamburger,pisuva,zolta),
      (3,bruno,hamburger,fotografija,zolta),
      (3,igor,piza,cita,crvena),
      (3,igor,hamburger,cita,crvena)
    ]

?- po_pozicija(L,BrPozicija)
```

Потребно е да се генерираат комбинации од 4те позиции и да се филтрираат тие кои нарушуваат дадени услови.

За генерирање на комбинации се користи предикатот kombinacii, кој враќа листа со 4 елементи – по 1 елемент за секоја позиција.

```
kombinacii([M0,M1,M2,M3]):-po_pozicija(L0,0),po_pozicija(L1,1),po_pozicija(L2,2),po_pozicija(L3,3),
    member(M0,L0),member(M1,L1),member(M2,L2),member(M3,L3).
```

Следните ограничувања се дефинирани во pred1, pred2, pred3.

За личноста која пишува и јаде хамбургер мора да важат и двете заедно, па затоа за другите два случаи (кога е исполнето само едното) враќа !,fail што е всушност негација во Пролог.

```
%Личноста која седи покрај онаа во бела маица сака пица.
pred1(L):-pozicija_bela(L,PozBela),pozicija_piza(L,PozPiza),
    (   (   PozPiza is (PozBela+1)) ; (   PozPiza is (PozBela-1))).

%Бруно седи покрај оној што јаде пица.
pred2(L):-pozicija_bruno(L,PozBruno),pozicija_piza(L,PozPiza),
    (   (   PozPiza is (PozBruno+1)) ; (   PozPiza is (PozBruno-1))).

%Оној што сака да пишува јаде хамбургер
pred3([H|T]):-( _,_,hamburger,pisuva,_) = H,!.
pred3([H|T]):-( _,_,hamburger,_,_) = H,!,fail.
pred3([H|T]):-( _,_,_,pisuva,_) = H,!,fail.
pred3([H|T]):-pred3(T).
```

За да се проверат дали важат тие, потребно е од комбинациите да се извлечат информации за позицијата која и е доделена на личноста во бела маица, на Бруно и да се најде кој јаде хамбургер.

```
pozicija_bruno([H|T],PozBruno):-( Poz,Ime,_,_,_) = H,Ime==bruno,PozBruno is Poz.
pozicija_bruno([H|T],PozBruno):-( Poz,Ime,_,_,_) = H,Ime\=bruno,pozicija_bruno(T,PozBruno).

pozicija_piza([H|T],PozPiza):-( PozPiza,_,piza,_,_)=H,!.
pozicija_piza([H|T],PozPiza):-pozicija_piza(T,PozPiza).

pozicija_bela([H|T],PozBela):-( PozBela,_,_,_,bela)=H,!.
pozicija_bela([H|T],PozBela):-pozicija_bela(T,PozBela).
```

На крај се изминуваат можните комбинации и се проверува за која важат наведените услови.

```
108 resenie(K):-kombinacii(K),
109   pred1(K),
110     pred2(K),
111     pred3(K).
```

КРАЈНО РЕШЕНИЕ: Сите услови се задоволени! :)

```
L = [ (0,teo,sendvic,fotografija,sina),
      (1,mira,pita,krstozbor,bela),
      (2,igor,piza,cita,crvena),
      (3,bruno,hamburger,pisuva,zolta)
    ]
false

?- resenie(L)
```