CEng 242 Homework 6

Due: 28^{th} May 2001

In this homework you will write a matchmaker (çöpçatan) application in Prolog for matching couples in a person database. `person` entries are Prolog structures in the form:
`person(`*name*`,`*sex*`,[`*feat1, feat2, ..., featn*`]).`
Where *name* is an atom describing the name of the person, *sex* is either `male` or `female` and the following list is any list of Prolog terms with arbitrary structure.
The aim is to find a matching among couples with different sex with maximum features are the same (unifiable in Prolog terms).
You should implement the following Prolog clauses:

`match(`*NameA*`,`*NameB*`,`*num*`)`
: This predicate checks for *NameA* is male, `NameB` is female and there are at least *num* features unifiable among them. Note that the features have free order so this predicate should find existence of any and the only first mapping with *num* elements among the list of features.

`maxmatch(`*NameA*`,`*NameB*`,`*Num*`)`
: This predicate finds the maximum number of matched features for *NameA* and *NameB* (`male` and `female` respectively) in *Num*.

`nevertry(`*NameA*`,`*NameB*`)`
: This predicate checks for again two names (`male` and `female` respectively) if no feature is same among two people. Those two names should not have any unifiable feature.

`bestcouple(`*NameA*`,`*NameB*`)`
: This predicate finds the couples with the maximum match value among all pairs of couples. If more than one couples have the same maximum match value, all couples are given as the solution.

Example people database:

```
person(ayse,female,[like(music),works,like(sports),hate(painting)]).
person(ayten,female,[like(music),hate(sports),student,hate(painting)]).
person(fatma,female,[like(sculpture),like(sports),like(painting),student]).
person(hatice,female,[works,like(swimming),hate(painting),like(sports),sings]).
person(hulya,female,[like(music),works,like(sports),hate(painting)]).
person(mualla,female,[like(music),like(painting),works]).
person(nihal,female,[like(swimming),works,like(sports),hate(painting)]).
person(sema,female,[student,like(music),sings,hate(sports),hate(painting),tall]).
person(zeynep,female,[hate(music),like(sports),like(painting),student,tall]).

person(ahmet,male,[like(music),like(sports),hate(painting),works]).
person(ali,male,[like(music),like(sports),like(painting),works,sings]).
person(hasan,male,[hate(music),like(sports),like(painting),works]).
person(huseyin,male,[like(music),like(sports),hate(painting),works]).
person(ibrahim,male,[like(sculpture),like(sports),like(painting),student]).
person(ismail,male,[like(swimming),sings,like(sports),hate(painting),works]).
person(mehmet,male,[like(music),hate(sports),hate(painting),student]).
person(osman,male,[hate(music),hate(sports),hate(painting),student]).
person(zeki,male,[like(music),hate(sports),hate(painting),student,tall]).
```

Example queries:

```
| ?- match(A,B,4).
A = ahmet, B = ayse ? ;
A = ahmet, B = hulya ? ;
A = huseyin, B = ayse ? ;
A = huseyin, B = hulya ? ;
A = ibrahim, B = fatma ? ;
A = ismail, B = hatice ? ;
A = ismail, B = nihal ? ;
A = mehmet, B = ayten ? ;
A = mehmet, B = sema ? ;
A = zeki, B = ayten ? ;
A = zeki, B = sema ? ;
no

| ?- maxmatch(A,B,4).
A = ahmet, B = ayse ? ;
A = ahmet, B = hulya ? ;
A = huseyin, B = ayse ? ;
A = huseyin, B = hulya ? ;
A = ibrahim, B = fatma ? ;
A = ismail, B = nihal ? ;
A = mehmet, B = ayten ? ;
A = mehmet, B = sema ? ;
A = zeki, B = ayten ? ;
no
```

```
| ?- maxmatch(zeki,A,N).
A = ayse, N = 2 ? ;
A = ayten, N = 4 ? ;
A = fatma, N = 1 ? ;
A = hatice, N = 1 ? ;
A = hulya, N = 2 ? ;
A = mualla, N = 1 ? ;
A = nihal, N = 1 ? ;
A = sema, N = 5 ? ;
A = zeynep, N = 2 ? ;
no

| ?- nevertry(A,B).
A = hasan, B = ayten ? ;
A = hasan, B = sema ? ;
A = osman, B = mualla ? ;
no

| ?- bestcouple(A,B).
A = ismail, B = hatice ? ;
A = zeki, B = sema ? ;
no
```

You should only supply your prolog code in the homework. Do not supply any `person(...)` entry. You are free to declare and use any other helper predicates. Your predicates should not generate extra repetitions of the solutions except the repetitions from the domain (like repetition of a name in person database). Generating all solutions and then removing the duplicates is not acceptable.