# Fall 2005 22C:111

## Homework 7 Solution

**Problem 1:**

```
(a)birth(_,_,X,_,(M,D,_)), birth(_,_,Y,_,(M,D,_)), X\=Y.

(b)birth(F,M,X1,_,D), birth(F,M,X2,_,D), X1\=X2.

(c)parentOf(Child,Parent), birth(_,_,Child,_,(Mon,_,_)),
birth(_,_,Parent,_,(Mon,_,_)).



(a)

?- birth(_,_,X,_,(M,D,_)), birth(_,_,Y,_,(M,D,_)), X\=Y.

X = kathleen_Kennedy

M = 2

D = 20

Y = jean_Kennedy ;

X = jean_Kennedy

M = 2

D = 20

Y = kathleen_Kennedy ;

X = kathleen_Kennedy2

M = 7

D = 4

Y = christopher_Kennedy ;

X = michael_Kennedy

M = 2
```

```
D = 27

Y = kara_Kennedy ;

X = christopher_Kennedy

M = 7

D = 4

Y = kathleen_Kennedy2 ;

X = matthew_Kennedy

M = 1

D = 11

Y = mariah_Cuomo ;

X = matthew_Kennedy

M = 1

D = 11

Y = cara_Cuomo ;

X = kara_Kennedy

M = 2

D = 27

Y = michael_Kennedy ;

X = tatiana_Schlossberg

M = 5

D = 5

Y = katherine_Schwarzenegger ;

X = katherine_Schwarzenegger
```

```
M = 5

D = 5

Y = tatiana_Schlossberg ;

X = mariah_Cuomo

M = 1

D = 11

Y = matthew_Kennedy ;

X = mariah_Cuomo

M = 1

D = 11

Y = cara_Cuomo ;

X = cara_Cuomo

M = 1

D = 11

Y = matthew_Kennedy ;

X = cara_Cuomo

M = 1

D = 11

Y = mariah_Cuomo ;

No
```

(b)

```
?-birth(F,M,X1,_,(Mon,D,Y)), birth(F,M,X2,_,(Mon,D,Y)), X1\=X2.

F = andrew_Cuomo
```

```
M = kerry_Kennedy

X1 = mariah_Cuomo

Mon = 1

D = 11

Y = 1995

X2 = cara_Cuomo ;

F = andrew_Cuomo

M = kerry_Kennedy

X1 = cara_Cuomo

Mon = 1

D = 11

Y = 1995

X2 = mariah_Cuomo ;

No

(c)

?-parentOf(Child,Parent), birth(_,_,Child,_,(Mon,_,_)),
birth(_,_,Parent,_,(Mon,_,_)).

Child = kara_Kennedy

Parent = edward_Kennedy

Mon = 2 ;

No
```

**Problem 2:**

The definitions of the two predicates added into file 'familial' is:

```
%solution to problem 2(a)
%motherInLawOf(Person, MotherInLaw)- the dictionary definition of
%mother-in-law is the mother of a spouse

motherInLawOf(Per, M) :-

    (marriage(Per, SP, _); marriage(SP, Per, _)), motherOf(SP, M).

%solution to problem 2(b)
%ancestorOf(Person, Ancestor)- - succeeds when one of Person's
%ancestors (i.e., a person from whom he/she is
%descended, a forebear) is Ancestor.

ancestorOf(Per, A) :- parentOf(Per,A).

ancestorOf(Per, A) :- parentOf(Per,B), ancestorOf(B, A).
```

**Problem 3:**

Two alternative ways to define the predicate are provided in the following.

DEFINITION(A)

```
%solution to 22C:111 Fall 2005 Homework7, problem 3

% add a counter to memorize number of ( not matched so far.
balanced(X) :- balanced2(X,0).

%if (, increment the counter, check the rest of the ascii code list
balanced2([40|XS], N) :- N2 is N+1,
                         balanced2(XS, N2).

%if ), decrement the counter, check the rest of the ascii code list
%if there is no ( unmatched, then the string is not matched.
balanced2([41|XS], N) :- N>0,
                         N2 is N-1,
                         balanced2(XS, N2).

%if the first char is not ( or ), just skip it.
balanced2([X|XS], N) :- X=\=40,
                        X=\=41,
                        balanced2(XS, N).

%empty string is balanced.
balanced2([], 0).
```

DEFINITION(B)

```
% Test a string for balanced parens
balanced("").
balanced([Z|Zs]) :- Z\==40, Z\==41, balanced(Zs).   % ignore leading non-(&)

% otherwise a balanced string Zs must have the structure
% |------- Zs -------|
% |(--------)--------|
%  |-- Xs --|-- Ys --|
% i.e., Zs = [(] ++ Xs ++[)] ++ Ys (in Haskell notation)
% where Xs and Ys are each balanced (and shorter than Zs).

% A string Zs can be split in this fashion by an application
% of append as follows:

balanced([40|Zs]) :- append(Xs, [41|Ys], Zs),
             balanced(Xs),
             balanced(Ys).
```