CSc 8710, Fall 2005
Exam1, October 5, 2005 Name:

1. Consider the HW3 notation for representing relations in Prolog memory.

   a. Define the predicate:
   `cartesianProductSchema(Rname1,Attr1,Rname2,Attr2,Attr)`

   where there are four input parameters: `Rname1` and `Rname2`, the names of relations, and `Attr1` and `Attr2`, their corresponding attribute name lists. The output parameter `Attr` is the attribute name list for the cartesian product of the two input relations. For example the following goals should succeed:

```
?- cartesianProductSchema(suppliers,[sno,sname,city],
                          spj,[sno,pno,qty],A).
A = [suppliers.sno,sname,city,spj.sno,pno,qty]
Yes

?- cartesianProductSchema(r,[a,b,c,d],s,[b,c,d,e],A).
A = [a, r.b, r.c, r.d, s.b, s.c, s.d, e]
Yes
```

   You may use the built-in predicate `concat(A,B,C)` which concatenates two atoms `A` and `B` and produces the atom `C`.

   b. Define a predicate `cartesianProduct(Tuples1,Tuples2,Tuples)` where `Tuples1` and `Tuples2` are input sets of tuples and `Tuples` is the cartesian product of these sets of tuples, produced as output. For example, the following goal should succeed:

```
?-  cartesianProduct([['S1','Jones','Atlanta'],
                      ['S2','Smith','Duluth']],
                     [['S1','P1', 100],
                      ['S1','P2', 100]],
                     R).

R = [['S1', 'Jones', 'Atlanta', 'S1', 'P1', 100],
     ['S1', 'Jones', 'Atlanta', 'S1', 'P2', 100],
     ['S2', 'Smith', 'Duluth', 'S1', 'P1', 100],
     ['S2', 'Smith', 'Duluth', 'S1', 'P2', 100]]
Yes
?-
```

2. Consider the following relational database:

   ```
   drinker(dname)
   bar(bname)
   beer(rname)
   frequents(dname,bname)
   serves(bname,rname)
   likes(dname,rname)
   ```

   The first three relations record the names of drinkers, bars and beers of interest. The `frequents` relation indicates the bars a drinker visits; the `serves` relation tells what beer each bar sells, and the `likes` relation indicates which beer each drinker likes to drink.

   a. Write expressions in DRC and RA to answer the query: *Get names of drinkers that frequent only bars that serve some beer that they like*.
   b. Write expressions in DRC and either in Datalog or RA to answer the query: *Get names of drinkers that frequent no bar that serves a beer that they like (i.e. every bar a drinker frequents does not serve a single beer he/she likes)*.
   c. Write expressions in DRC and Datalog to answer the query: *Get pairs of drinker names such that the two drinkers frequent EXACTLY the same bars and like EXACTLY the same beers*.

3. Consider the following definite logic program.

```
widget(w1).
widget(w2).
widget(w3).
widget(w4).

supplies(acme,p1).
supplies(acme,p4).
supplies(aaa,w3).
supplies(aaa,w4).
supplies(foobar,X) :- widget(X).
supplies(X,Y) :- subPart(Y,Z), supplies(X,Z).

partOf(p2,p1).
partOf(p3,p2).
partOf(p5,p4).
subPart(X,Y) :- partOf(X,Y).
subPart(X,Y) :- partOf(X,Z), subPart(Z,Y).
```

Construct the minimal model for the program using the Tdb operator technique. Show the work after each application of the Tdb operator.

4. For each of the following DRC formulas state whether they are domain-independent or not. Justify your answer.

   a. `(Exists X)(P(X,Y) and not Q(X,Z))`

   b. `(Exists X)(P(X,Y) and not (Q(X) or R(Y)))`

5. Consider the following Prolog program:

```
del(X,[X|L],L).
del(X,[Y|L],[Y|M]) :- del(X,L,M).
```

Construct the entire SLD Refutation tree for the goal:

```
?- del(A,[a,b,c],N).
```