Checking Data Integrity with Datalog (DLV)

General Instructions

1. To get started, install the DLV system on your computer.
2. Read the DLV tutorial. There are some notions that will not be easy to understand at first. In particular, try and understand the following most important subparts:

   – The Family Tree Example: Predicates, Variables, and Recursion

   – DLV as a Deductive Database System: Comparison Operators

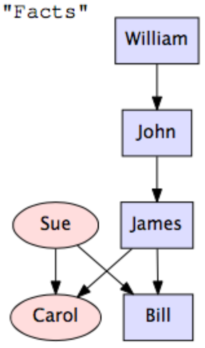
3. Download and extract the datalog_assignment.zip file. In the extracted zip file, you will find 2 dlv files: family.dlv and publication.dlv.
4. You must answer the following problems in the dlv files by using facts that have been given. Make sure you answer them using the correct rule name.
5. You are not able to add more facts other than the ones that have been given. However, you can add additional rules to support your answer.
6. You can test your answer using the Python script dlvExecutor.py; execute it using Python2.7. For example: python2.7 dlvExecutor.py
7. After your answers are ready (family.dlv and publication.dlv), you can submit them all at once using submit.py; execute the script using Python 2.7 as well.
8. After submission, you can check the grading results in the Coursera web submission page.

Part 1: Family (family.dlv)

```
% EDB (Extensional Database), "Facts"
parent(william, john).
parent(john, james).
parent(james, bill).
parent(sue, bill).
parent(james, carol).
parent(sue, carol).

male(john).
male(james).
male(bill).

female(sue).
female(carol).
```



```
% IDB (Intensional Database), "Rules"
grandparent(X,Y) :-
    parent(X, Z), parent(Z, Y).
father(X,Y)   :-
    parent(X, Y), male(X).
mother(X,Y)   :-
    parent(X, Y), female(X).
brother(X,Y)  :-
    parent(P, X), parent(P, Y),
    male(X), X != Y.
sister(X,Y)   :-
    parent(P, X), parent(P, Y),
    female(X), X != Y.
```

1. descendant: descendant(X,Y) holds if X is a descendant of Y
2. sibling: sibling(X ,Y) holds if X and Y are siblings. Hint: X and Y share a parent P.
3. Write the following integrity constraints (ICs) as "soft constraints" in *denial form*, i.e., write rules that yield variable bindings that serve as "constraint violation witnesses":
   a. icv_person_has_parent: *Every person must have a parent*. Hint: Look at the IC-rules in the course handouts.

b. icv_person_has_father_mother: *Every person has a father and a mother.* Hint: Same idea as in the Warm-Up. First, write a rule that yields the people for which the IC is satisfied. Then write another rule that reports as an IC-violation those people who are *not* in the answer to that first query.

Part 2: Publication (publication.dlv)

Consider "dirty" dataset store publication data in this following relation. The IC-checking capabilities of a database provide a powerful way to detect inconsistencies.

PUBLICATION

Pid	Authors	Year	Title	Journal	Vol	No	Fp	Lp	Publisher
6755	Hyatt, A.	1872	Fossil Cephalopods of the Museum of Comparative Zoology	Bull. MCZ	5	5	91	9	
2580	Rolfe, W.D.I.	1962	A new phyllocarid crustacean from the Upper Devonian of Ohio	Breviora	151	151	4	6	MCZ
2044	Francis Arthur Bather	1934	Chelonechinus N.G., A Neogene Urechinid	GSA Bull.	45	4	808	832	
4407	Kummel, B.	1969	Ammonoids of the Late Scythian	Bull. MCZ	137	3	476		
⋮									⋮

This relational dataset has been transformed into Datalog format, which you can find in publication.dlv.

1. Define the following ICs in *denial form* in Datalog syntax. You can assume that the table is available as a Datalog predicate of the form publication(I, A, Y, T, J, V, N, F, L, P). Recall that in Datalog, arbitrary (capitalized) names can be chosen as variables since it is the argument *position* that determines which attribute/column is meant.
   a. icv_pid_key: The publication identifier Pid is a *key*, i.e., if a row agrees with another row on the key attribute Pid, then it also agrees on all other attributes (i.e., the "two" rows are in fact one and the same). As usual, your rule should return the IC-violations.

   b. icv_journal_publisher: Every journal has a single publisher. Like (FD-1), this is a *functional dependency*. It is sometimes written as Journal → Publisher.
   c. ncv_firstpage_lastpage: The last page Lp cannot be smaller than the first page Fp. Note: This numerical constraint can be evaluated independently on each row.
2. Now consider that an additional table cites(P_1, P_2) is given, which records pairs of publication P_1, P_2, where P_1 is citing P_2. Define the following IC in denial form:

CITES

Pid1	Pid2
4711	2020
4711	3799
3799	2580
⋮	⋮

   a. icv_cited_publication: Every cited publication in CITES also occurs in PUBLICATION. This is an *inclusion dependency* and is usually written in the form: CITES[Pid$_2$] ⊆ PUBLICATION[Pid].

b. ncv_p1_greater_p2: If P1 cites P2, then P2’s year of publication cannot be greater than P1’s year of publication.