CS-4337
Prolog Programming Assignment
Due: November 26, 2018 11:59PM

Description

Define and test the Prolog predicates described below. Each of your predicates ***must*** have the same name and signature as the examples below. In Prolog, predicate profiles are indicated with the number of parameters that they take, e.g. `suspect/2` is a predicate named "`suspect`" that takes 2 parameters.

Your predicates must behave properly on all instances of valid input types.

Submission

Your submission should consist of a single source code text file that includes all facts, predicate definitions, propositions, and rules. Your file must be named ***`your_net_id`*****`.prolog`**.

You may find additional Prolog language help at the following links:

- SWI-Prolog manual
- SWI-Prolog documentation
- Learn Prolog Now!
- http://www.csupomona.edu/~jrfisher/www/prolog_tutorial/contents.html

Clue

Four guests (Colonel Mustard, Professor Plum, Miss Scarlett, Ms. Green) attend a dinner party at the home of Mr. Boddy. Suddenly, the lights go out! When they come back, Mr Boddy lies dead in the middle of the table. Everyone is a suspect.

Upon further examination, the following facts come to light:

- Mr Boddy was having an affair with Ms. Green.
- Professor Plum is married to Ms. Green.
- Mr. Boddy was very rich.
- Colonel Mustard is very greedy.
- Miss Scarlett was also having an affair with Mr. Boddy.

There are two possible motives for the murder:

- Hatred: Someone hates someone else if that other person is having an affair with his/her spouse.
- Greed: Someone is willing to commit murder if they are greedy and not rich, *and* the victim is rich.

Part A: Write the above facts and rules in your Prolog program. Use the following names for the people: **`colMustard`**, **`profPlum`**, **`missScarlet`**, **`msGreen`**, **`mrBoddy`**. Be careful about how you encode (or don't encode) symmetric relationships like marriage - you don't want infinite loops! **`married(X,Y) :- married(Y,X) % INFINITE LOOP`**

Part B: Write a predicate, **`suspect/2`**, that determines who the suspects may be, i.e. who had a motive, given a victim.

```
?- suspect(Killer,mrBoddy)
Killer = suspect_name_1
Killer = suspect_name_2
etc.
```

Part C: Add a *single fact* to your database that will result in there being a unique suspect. Clearly indicate this line in your source comments so that it can be removed/added for grading.

```
?- suspect(Killer,mrBoddy)
Killer = unique_suspect.
```