PROLOG HW 2 SPRING 2020

1. Write PROLOG code to take a prolog tree constant and replace each

atom 'sword' by the the atom 'plowshare'. You may assume the tree

has only atomic data. There are various possibilities for doing this,

but you can think in terms of traversals of trees. Or not.

e.g. substit(a(sword,joe,b(moe,sword,sword,toe)),X).

X = a(plowshare,joe,b(moe,plowshare,plowshare,toe))

NOTE: ignore 'functors'; only replace leaves

--------

In solving Caliban problems you should translate loyally each 'clue'

from English to Prolog. Use rules like

clue1(L):-and translate the first clue here

Your goal will be, more or less, of the form

soln(L):-setup(L), clue1(L), clue2(L),etc

since things will then be easier to debug.

2. Brown, Clark, Jones and Smith are 4 substantial citizens who serve their

community as achitect, banker, doctor and lawyer, though not necessarily

respectively.

Brown, who is more conservative than Jones but more liberal than Smith,

is a better golfer than the men who are younger than he is and has a

larger income than the men who are older than Clark.

The banker, who earns more than the architect, is neither the youngest

nor the oldest.

The doctor, who is a poorer golfer than the lawyer, is less conservative

than the architect.

As might be expected, the oldest man is the most conservative and has the

largest income, and the youngest man is the best golfer.

What is each man's profession?

hint: to rank people for weath, ability, relative age, etc

use the numbers 1,2,3,4 Be careful to state whether 1 represents,

e.g., youngest or oldest. Doing this makes comparisons easy to code.

Nota Bene: you might find that your answers are not 'unique' for

something like golfing prowess, but all your answers should agree as

to whom does which job. Display all 'answers', including things like

golfing prowess. This will give us more confidence in your code.

3. Prolog uses general trees, not binary trees. An example is

a(b,c,d(e,f,g)) where root a has 3 kids, as does kid d.

It is possible to define both preorder and postorder for general trees,

although inorder of course makes no sense.

For this assignment we are interested in postorder, which is defined as

follows:

to 'visit' a tree in postorder,

you visit the subtrees of the root, in left to right order,

in postorder, and then you visit the root

Thus the example above would yield the following postorder traversal:

b c e f g d a

Write Prolog code which will perform a postorder traversal of a Prolog

tree constant. Hint: you might use 'univ', or its cousins.

Sample dialog:

?- postorder(a(b,c,d(e,f,g))).

b c e f g d a true

4. Write code to 'reverse' a prolog list. So if the input is

[d,o,g] the output will be [g,o,d].

due date: lets make this march 26, since that seems to be a face

to face class meeting.