Question 1: Datalog Syntax

Consider the language that consists of just the predicate symbols $\{a, b, c, d\}$, all of which are 0-ary (and thus do not take any arguments).

List all of the different facts that can be written in this language. (A fact is a definite clause that just has a left-hand side).

How many different bodies can be written? (Do not include different re-orderings of the atoms.) List several of them. A body is a conjunction of atoms.

How many different rules can be written. A rule is a definite clause that has a head and a non-empty body. Again, do not include different re-orderings of the atoms in the body. Explain your answer. List several of them.

Do not worry about whether the rules convey any useful information or not, just whether they are legal rules that can be written.

Question 2: Datalog Semantics

This question was removed from the homework.

Question 3

Write the following in datalog. You can use the 0-ary predicates son and child, where son means that you have a son and child means that you have a child.

If you have a son then you have a child.

Question 4: Tcl Lists

Lists are an important data structure in Tcl, and we will use them to represent atoms, composed of a predicate and its arguments. We will use Tcl’s list command to construct the list and use tc’s set command to assign it to the Tcl variable atom.

set atom [list pred foyer X foyer parlour Y X]

Note that Tcl likes to in fact view everything as a list, especially strings. So if you do the following tcl command:

set atom "pred foyer X foyer parlour Y X"

it treats this exactly as the previous tcl command, as a list with 7 items.

To confuse things further, you could also write:

set atom \{pred foyer X foyer parlour Y X\}

This will also do the same as the lines above. Now, enter the following into a Tcl command shell.

set sub \{b c\}
set atom1 \{pred $sub\}
set atom2 "pred $sub"
set atom3 [list pred $sub]
Check the results of each variable using the **puts** command. You should see that the curly brackets prevent tcl from interpreting anything inside of the brackets, while this is not the case for quotes or the tcl list command. When in doubt, use the **list command**.

Now try out each of the following and report what you get. Also, give a short answer (1 sentence or so) of what the command did.

```tcl
set list1 {1 2 c 4 e}
set list2 {6 g 7 i j}
concat $list1 $list2
list $list1 $list2
list $list1 list2
list a $list2
concat a $list2
lindex $list1 3
lindex 3 $list1
lrange $list1 1 end
llength $list1
lappend list1 5
lappend list1 6 7
lappend list1 $list2
```

**Question 5: Programming**

Please see the web page at [http://cslu.ohsu.edu/~heeman/cs560/tcl.html](http://cslu.ohsu.edu/~heeman/cs560/tcl.html) to get started using Tcl.

Consider an atom such as **pred(foyer,X,foyer,parlour,Y,X)**. Write a routine that when given an atom, it will list all of the terms and classify them as either a constant or a variable. The output for the above should be:

```
constant foyer
variable X
constant foyer
constant parlour
variable X
variable Y
```

Rather than use the Datalog syntax for atoms, we will use Tcl’s lists, which will make programming a lot simpler. The predicate name will be the first item in the list, and the arguments will be the subsequent items.

Rather than have you start from scratch, start with the following code.
proc parse {atom} {
    set predicate [lindex $atom 0]
    set arguments [lrange $atom 1 end]
    puts "Predicate is $predicate"
    puts "Arguments are $arguments"
}

set myatom [list pred foyer X foyer parlour Y X]
parse $myatom

Change this code so that it analyzes the arguments as either variables or constants (it should have the output given above). The tcl commands you will need to use are ‘if - else’; ‘foreach’; ‘string index’; and ‘string is upper’.

Hand in a copy of your code. Anytime you hand in your code, make sure that it is nicely formatted. If a command is too long for one line, use a ‘\’ to split it into to multiple lines. Also your code should be indented to reflect the level of nesting. For each nesting, indent at least 1/2 an inch so that I can easily read your code. Also, do not make any one procedure more than one page in length. If you need more space, break it into subroutines.

Also hand in a copy of its output. You can do this in TclPro by right clicking on the console header, selecting edit, and then selecting select all, and then selecting copy.

**Question 6**

Now we want to change the previous program so that it only lists each constant or variable once. The new output should be:

constant foyer
variable X
constant parlour
variable Y

To this, use an associative array to keep track of which constants and variables have been seen ("http://www.lib.uchicago.edu/k"
Call the associate array seen. Also, use ‘info exists’ to see if you have already seen constant or variable already.

Hand in a copy of your code.