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

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 3: Python Lists

Lists are an important data structure in Python, and we will use them to represent atoms, composed of a predicate and its arguments. To assign a list to a variable atom, do the following:

atom = ["pred", "foyer", "X", "foyer", "parlour", "Y", "X"]

Consider the following.

list1 = [1, 2, "c", 4, "e"]
list2 = [6, "g", 7, "i", "j"]

Make sure you understand what the following does by running the code. No need to turn anything in.

list3 = list1+list2
print(list3)

list4 = [list1,list2]
print(list4)

list1.append("x")
print(list1)

list1.extend(list2)
print(list1)

Sometimes you want to access an element in a list. Make sure you understand how the following code works, including the use of negative indexes.
list1 = [1, 2, "c", 4, "e"]
print(list1[0])
print(list1[3])
print(list1[-1])

Sometimes you want a range from a list. Make sure you understand how the following code works. Note that the second argument indicates all elements up to that index, but not including it. Also understand the use of negative indexes.

print(list1[1:3])
print(list1[:])
print(list1[2:])
print(list1[:2])
print(list1[-3:-1])

Explain the difference between the following commands. Turn in your answer.

print(list1[2:3])
print(list1[2])

Lists are mutable objects in Python. Run the following code. Explain what the assignment in the second line actually doing. Explain why list2 has its final value. Feel free to use any web resources to explain this.

list1 = [1, 2, "c", 4, "e"]
list2 = list1
list1.append("x")
print(list2)

Run the following code and explain why the value of list2 is as it is. In your explanation, explain how the assignment is working. Make sure you contrast your answer to your answer for the previous question with append.

list1 = [1, 2, "c", 4, "e"]
list2 = list1
list1 = list1 + [6]
print(list2)

Sometimes, you will want a separate copy of a list. Run the following code. Explain why list1 and list2 have different values.

list1 = [1, 2, "c", 4, "e"]
list2 = list1[:]
list1.append("x")
print(list1)
print(list2)

**Question 4: Programming**

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:

costant foyer
variable X
constant foyer
classic parlour
variable X
variable Y

Rather than use the Datalog syntax for atoms, we will use Python'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.

```python
def parse(atom):
    predicate = atom[0]
    arguments = atom[1:]
    print("Predicate is %s" % predicate)
    print("Arguments are %s" % arguments)

myatom = ["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 python commands you will need to use are 'if - else'; 'for'; and the method 'is.upper'. You will also need to know how to index into a string, which is similar to how you index into a list.

Hand in a copy of your code. Make sure that when you hand in your python code, that it capture its natural indentation. 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.

**Question 5**

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

```python
classic foyer
variable X
classic parlour
variable Y
```

To this, use a python dictionary to keep track of which constants and variables have been seen. Call it dictionary `seen`. Also use 'x in seen' to see if you have already seen x already.

Hand in a copy of your code.