PRACTICE Some questions from old exams suitable for Take Home (open book)

1. Define a Definite Clause Grammar (DCG) for strings $a^n b^m c^k$ over the alphabet \{a, b, c\} where $0 \leq n, m, k$ and $n + m \leq k$. For example,

   \[\text{?- s([a,b,b,c,c,c,c],L).} \]
   \[\text{L = [a, b, b, c, c, c, c];} \]
   \[\text{L = [c];} \]
   \[\text{L = [];} \]
   \[\text{false} \]

2. Recall that the non-negative integers can be encoded as numerals given by 0 and its successors (with, for example, the numeral $s(s(s(0)))$ encoding 3).

   \[\text{numeral(0).} \]
   \[\text{numeral(s(X)) :- numeral(X).} \]
   \[\text{add(0,X,X).} \]
   \[\text{add(s(X),Y,s(Z)) :- add(X,Y,Z).} \]

(a) Define \text{mult(X,Y,Z)} to hold when Z encodes the product $X$ times $Y$. If your definition is not tail recursive, give a modification \text{mu(X,Y,Z)} that is.

(b) Define \text{less(X,Y)} to hold when $X$ and $Y$ encodie non-negative integers $x$ and $y$ such that $x < y$. For example,

   \[\text{?- less(0,s(0)).} \]
   \[\text{yes.} \]
   \[\text{?- less(s(s(0)),s(s(0))).} \]
   \[\text{no.} \]

(c) Define the predicate \text{decrList/1(L)} that holds when $L$ is a list of numerals in strictly decreasing order. For example,

   \[\text{?- decrList([0,0]).} \]
   \[\text{no.} \]
   \[\text{?- decrList([s(s(0)),X]).} \]
   \[\text{X = s(0);} \]
   \[\text{X=0;} \]
   \[\text{no} \]

(d) Define the predicate \text{subset/2(X,Set)} that holds when $X$ is a numeral and $Set$ is a list of numerals less than or equal to $X$ such that \text{decrList(Set)}. For example,

   \[\text{?- subset(s(0),Set).} \]
   \[\text{Set = [];} \]
   \[\text{Set = [0];} \]
   \[\text{Set = [s(0)];} \]
   \[\text{Set = [s(0),0];} \]
   \[\text{no} \]
(e) Define the predicate \( \text{interval}/2(X,I) \) that holds when \( \text{subset}(X,I) \) and furthermore, \( I \) is an interval in that whenever two numerals \( A \) and \( B \) are members of \( I \) then so is every numeral between \( A \) and \( B \). For example,

\[
\begin{align*}
?- \text{interval}(s(s(0)), [s(0), 0]).
& \text{yes} \\
?- \text{interval}(s(s(0)), [s(s(0)), 0]).
& \text{no}
\end{align*}
\]

(f) Negative integers can be encoded with a predecessor (\(-1\)) function \( p \) such that \(-3\) is \( p(p(p(0))) \). Define the predicate \( \text{neg}/2(\text{Numeral}, \text{Neg}) \) that holds when \( \text{Neg} \) encodes the negative of the integer encoded by \( \text{Numeral} \). For example,

\[
\begin{align*}
?- \text{neg}(s(s(0)), X).
& X = p(p(0)) ; \\
& \text{no}.
\end{align*}
\]

(g) Define the predicate \( \text{pOm}/1(X) \) (for “plus-or-minus” \( X \)) that holds of both numerals and their negatives, and can be used to enumerate them as in

\[
\begin{align*}
?- \text{pOm}(X).
& X = 0 ; \\
& X = s(0) ; \\
& X = p(0) ; \\
& X = s(s(0)) ; \\
& X = p(p(0)) ; \\
& X = s(s(s(0))) ; \\
& X = p(p(p(0))) ; \\
& \ldots
\end{align*}
\]

For full marks, make sure that every numeral and its negative is enumerated exactly once, avoiding unfair listings that leave out negative numerals, as in

\[
\begin{align*}
?- \text{pOm}(X).
& X = 0 ; \\
& X = s(0) ; \\
& X = s(s(0)) ; \\
& \ldots
\end{align*}
\]

3. Is there a list \( L \) such that \( \text{append}([a],L,L) \)? Justify your answer.

4. A binary predicate \( \text{edge}/2 \) is symmetric if whenever \( \text{edge} \) holds of a pair \( (a,b) \), it also holds of \( (b,a) \). An obvious way to make \( \text{edge} \) symmetric is by adding the rule

\[
\text{edge}(A,B) :- \text{edge}(B,A).
\]

Why is this problematic, and how can we get around this problem whilst turning \( \text{edge} \) into a symmetric predicate \( \text{symEdge} \)?

5. How is \text{cut} ! used to define \text{if-then-else}/3?
6. How can we use assert and retract to implement assignment statements such as $x := x + x$?

7. The $n$th **derangement number** $D_n$ is the integer defined inductively as follows

$$
D_1 := 0 \\
D_2 := 1 \\
D_n := (n - 1)(D_{n-1} + D_{n-2}) \quad \text{for } n > 2
$$

so that, for example,

$$
D_3 = 2(1 + 0) = 2 \\
D_4 = 3(2 + 1) = 9.
$$

This question is about defining a binary predicate $d$ computing derangement numbers, leading the Prolog interpreter to respond, for example, as follows.

?- d(3,D).
D=2 ;
no

?- d(4,D).
D=9 ;
no.

(a) A simple attempt to define $d$ is given below, with question marks


that must be replaced by suitable Prolog variables.

\[
\begin{align*}
\text{d}(1,0). \\
\text{d}(2,1). \\
\text{d}(\text{N},\text{D}) & : - \text{N} > 2, \\
& \quad ?1 \text{ is } \text{N}-1, \\
& \quad ?2 \text{ is } \text{N}-2, \\
& \quad \text{d}(\text{N}1,?3), \\
& \quad \text{d}(\text{N}2,?4), \\
& \quad \text{D \ is \ } ?5*\text{(D1+D2)}. \\
\end{align*}
\]

Replace the question marks ?1,?2,?3,?4,?5 so that the code above computes derangement numbers.

(b) Revise the definition of $d/2$ above for a tail-recursive program computing the $n$th derangement number.

(c) Use memoization to compute the $n$th derangement number, revising the definition of $d/2$ in part (a) above.

8. (a) A simple DCG for generating bit strings (i.e., lists of 0 and 1) is
s --> [] .
s --> s , b .

b --> [0] .

For example,
?- s(L, []).  
L = [] ;  
L = [0] ;  
L = [1] ;  
L = [0,0] ;  
L = [0,1] ;  
L = [1,0] ;  
L = [1,1] ;  
L = [0,0,0] ;
...

But what is problematic about the query
?- s([a], []).  

and how can we fix the DCG above so that the Prolog interpreter answers
no to this query.

(b) Write out the DCG given in part (a) as ordinary Prolog clauses, making the
difference lists explicit.

(c) Write a DCG for a predicate s that outputs lists of the form

[ b_1, b_2, ..., b_n, k ]

for every integer n ≥ 0 such that each b_i is either 0 or 1 for 1 ≤ i ≤ n, and
k is the integer that the bitstring b_1b_2⋯b_n encodes.

For example,
?- s([1,1,0,X],[]).
X = 6 ;
no

?- s([1,B,0,4],[]).
B = 0 ;
no

since 110 is the binary encoding of 6, and 100 is the binary encoding of 4.

(d) Three students, no two of the same age, play three different sports, and come
from three different counties.

In alphabetic order (which may or may not correspond with age),

(i) the students are: chris, pat, sandy
(ii) the sports are: boxing, football, tennis

(iii) the counties are: cork, dublin, mayo.

Write a DCG that outputs strings of the form

[Stu1,Spo1,Cou1, Stu2,Spo2,Cou2, Stu3,Spo3,Cou3]

(of length 9) representing all the possibilities such that

(i) Stui plays Spo for 1 ≤ i ≤ 3, and
(ii) Stui is older than Stuj for 1 ≤ i < j ≤ 3.

For example,

?- s([chris,football,mayo, pat,PatSport,cork, Stu,boxing,Cou],[]).
  PatSport = tennis,
  Stu = sandy,
  Cou = dublin ;
  no.