Prolog Exercises

Alessia Carotenuto

13th February, 2021

1 Exercise 1

Define a predicate that, given an integer I and a list L returns the value of the element of the list L in the position I.

```
get_elem([A|_REST],1,A).
get_elem([_A|REST],K,OUT):-
      K > 1, X is K-1, get_elem(REST,X,OUT).
```

2 Exam 01/2021

```
isLL3([]).
isLL3([A,B|[REST]]):-
     atom(A),atom(B),
     isLL3(REST).
%isLL3([a,b,[]).
%isLL3([a,b,[c,d,[e,f,[]]]]).

makeLL3([],[],[]).
makeLL3([A|REST1],[B|REST2],[A,B,OUT]):-
     atom(A),atom(B),makeLL3(REST1,REST2,OUT).
%makeLL3([a,c,e],[b,d,f],LL3).

makeLL3_2([],[]).
makeLL3_2([A,B|REST],[A,B,OUT]):-
     makeLL3_2(REST,OUT).
makeLL3_2([A|[]],[A,A,[]]).
```

3 Exam 02/2018

Write a PROLOG program that, given two lists of integers INDXS and VALUES, returns a list whose first element is the value stored in the position corresponding to the first element of INDXS of the list VALUES. For example, given INDXS=[2,1,4,3] and VALUES=[2,4,6,8], the output is [4,2,8,6]. You may start by defining a predicate that, given an integer I and a list L

returns the value of the element of the list L in the position I. Explain what happens if the position I does not correspond to an element of L.

```
get_elem([A|_REST],1,A).
get_elem([_A|REST],K,OUT):-
    K > 1, X is K - 1, get_elem(REST,X,OUT).

ex_prolog([],[_B|_REST],[]).
ex_prolog([A|REST1],[B|REST2],[OUT1|OUT2]):-
    get_elem([B|REST2],A,OUT1),
    ex_prolog(REST1,[B|REST2],OUT2).
```

4 Midterm 12/2018

An associative list is a list all of whose elements are 2-element lists; the first element of each element of the associative list is an atom, the second one is an arbitrary term. Example AL=[[a,f(0)],[b,f(1)],[c,f(2)]] is an associative list. Use atom(X) to test whether the argument is an atom.

(a) Write a PROLOG program that checks whether its argument is an associative list.

(b) Write a PROLOG program FINDT that, given in input an associative list AL, and atom A returns a term T, corresponding to the second element of the element of AL whose first element is A. Assume that every atom in the AL appears only once. Example: the result of FINDT([[a,f(0)],[b,f(1)],[c,f(2)]],b,T) is T=f(1).

(c) [EXTRA QUESTION tbd only after having answered everything else] Write a PROLOG program SUBS that, given in input an associative list AL, and a list of atoms L, returns the result of replacing the occurrences of the atoms that are in AL with the terms associated to them. Assume that every atom in AL appears only once. Example: the result of SUBS(AL,[a,b,a,d],L) is L=[f(0),f(1),f(0),d].

```
ass_list([[A,_B]]):- atom(A).
ass_list([[A,_B]|REST]):-
    atom(A), ass_list(REST).

find_t([[A,B]|_REST],A,B).
find_t([[A,_B]|REST],X,T):-
    A\=X, find_t(REST,X,T).

subs(_AL,[],[]).
subs(AL,[A|REST],[T|OUT]):-
    find_t(AL,A,T),
    subs(AL,REST,OUT).
subs(AL,[A|REST],[A|OUT]):-
    not(find_t(AL,A,_D)),
    subs(AL,REST,OUT).
```

5 Exam 01/2019

(a) Write a prolog program that, given a number, returns a list of numbers from 0 to the given number. Let us call this operation num2list. For example, when the input is 5, the output is [0,1,2,3,4,5].

(b) Write a prolog program that given a list of numbers, returns a list such that the numbers in the input are replaced by the result of num2list, when they are followed by a bigger number, and otherwise skipped. When the list has only one number, this should also be replaced with the result of num2list. For example, if the input is [2,4,1,3,4,3,2,1], the result is [[0,1,2],[0,1],[0,1,2,3],[0,1]].

```
num2list(0,[0]).
num2list(N,L):-
    X is N-1,X >= 0,num2list(X,OUT), append(OUT,[N],L).

num2list_2([N],[OUT]):-
    num2list(N,OUT).
num2list_2([A,B|C],[OUT|OUT2]):-
    A < B, num2list(A,OUT),
    num2list_2([B|C],OUT2).
num2list_2([A,B|C],OUT2):-
    A >= B,
    num2list_2([B|C],OUT2).
```

6 Midterm 12/2017

Write a PROLOG program that given a list of lists of integers and an integer N, and returns a list containing only the lists of the input structure, whose elements are all greater than N. For example, given the input [[1,2,3],[1,5,2],[4,6,8]] N=2, the answer is [[4,6,8]].

```
func([A],N):- A > N.
func([A|REST],N):-
    A > N,   func(REST,N).

ex_prolog([],_N,[]).
ex_prolog([L|REST],N,[L|OUT]):-
    func(L,N),
        ex_prolog(REST,N,OUT).
ex_prolog([L|REST],N,OUT):-
    not(func(L,N)),
    ex_prolog(REST,N,OUT).
```

Write a PROLOG program that given a list of lists of integers and an integer N returns a list containing only the lists of the input structure that have less than N elements. The check on the length of the list must be implemented. For example, given the input [[1,2,3],[1,5],[4,6]] and N is 3 the answer is [[1,5],[4,6]].

```
len_ex([], 0).
len_ex([_A | REST], D):-
    len_ex(REST, DIM), D is DIM + 1.
ex_prolog([], _N, []).
ex_prolog([A|REST], N, [A|OUT]):-
    len_ex(A, DIM), DIM < N,
    ex_prolog(REST, N, OUT).
ex_prolog([A|REST], N, OUT):-
   len_ex(A, DIM), DIM >= N,
    ex_prolog(REST, N, OUT).
```

7 Midterm 12/2019

A matrix that stores natural numbers can be represented in PROLOG as a list of lists of numbers.

(a)WriteaPROLOGprogramGETDthat,givenininputamatrixMandanindexI,returnsthevalue V of the element in position [I,I].Rows and columns are counted starting from 1. If the index is out of bounds the program should fail. For example, GETD([[11,12,13],[21,22,23],[31,32,33]],2,V) returns V=22.

(b) Write a PROLOG program GETDIAG that, given in input a matrix M, returns a list L containing all the elements in the main diagonal. For example, GETDIAG([[11,12,13],[21,22,23],[31,32,33]],L) returns L=[11,22,33].

```
getd(M,I,OUT):-
    nth1(I,M,V),
    nth1(I,V,OUT).

cont(M,N,[]):- not(getd(M,N,_OUT)).
cont(M,N,[OUT1|OUT2]):-
    getd(M,N,OUT1),
    X is N + 1,
    cont(M,X,OUT2).

getdiag(M,OUT):-
   cont(M,1,OUT).
```

8 Midterm 12/2020

(a) Write a PROLOG program allE(X) that, given a list, returns true if all of its elements are the same atom. With the empty list the program must return false.

(b) Write a PROLOG program allButOne(X) that, given a list, returns true if all of its elements but one are the same atom. Example input: L=[a,b,a,a,a] returns true, while L=[a,b,a,c] returns false.

```
allE([_A]).
allE([A,A|REST]):-
    allE([A|REST]).

allButOne([A,B]):- A\=B.
allButOne([A,B|REST]):-
    allE([A|REST]), A\=B.
allButOne([A,B|REST]):-
    allE([B|REST]), A\=B.
allButOne([A,A|REST]):-
     allButOne([A|REST]).
```

9 Exam 01/2018

Write a PROLOG program that given a binary tree of with integer numbers stored in the nodes. Write a program that returns the maximum value stored in the tree. For example, given the input [4,[1,[],[]],[7,[],[]]] the algorithm should return 7. Write a modified version of the program so that it also counts the number of occurrences of the maximum value.

```
max(A,B,A):-
    A >= B.
max(A,B,B):-
    A < B.
ex_prolog([R,[],[],R).
ex_prolog([R,LT,[],MAX1):-
        ex_prolog(LT,VL),
    max(R,VL,MAX1).
ex_prolog([R,[],RT],MAX2):-
        ex_prolog(RT,VR),
    max(R,VR,MAX2).
ex_prolog([R,LT,RT],MAX2):-
        ex_prolog(RT,VR),
    ex_prolog(LT,VL),
    max(VR,VL,MAX1),
    max(MAX1,R,MAX2).
```

10 Exam 01/20

Consider a data structure list, whose elements are lists of two atoms, such that the two atoms in each list are not equal (e.g. [[a,b],[c,d],[e,f]]), and call it LA.

(a) Write a Prolog program isLA(X), that checks whether its argument is an LA.

(b) Write a Prolog program makeLA(X,Y), that, given an input list X, returns an LA Y, obtained by removing all elements of X, that do not match the structure of the LA (i.e. lists of two equal atoms, lists where one element is a list instead of an atom, lists that are longer or shorter than 2). The predicate atom(X) checks whether its argument is an atom.

```
isLA([[X,Y]]):-
    X\=Y,atom(X),atom(Y).
isLA([[X, Y]|REST]):-
    X\=Y,atom(X),atom(Y),
    isLA(REST).
makeLA([A,B],[[A,B]]):- isLA([[A,B]]).
makeLA([A,B|REST], [[A,B]|OUT]):-
    isLA([[A,B]]),
    makeLA(REST, OUT).
makeLA([A,B|REST], OUT):-
    not(isLA([[A,B]])),
    makeLA(REST, OUT).
```

11 Exam 02/2020

Consider a data ALBMAX, which is a binary tree storing natural numbers on its nodes, such that the number stored in a node is the max of the numbers of the nodes in its children.

(a) Write a Prolog program isALBMAX(X), that checks whether its argument is an ALBMAX.

(b) Write a Prolog program filterALBMAX(X,Y), that, given an input list X of binary trees, returns a list of all the binary trees contained in X that are ALBMAX.

```
max(A,B,A):-
    A >= B.
max(A,B,B):-
    A < B.
isALBMAX([_R,[],[]]).
isALBMAX([R,[R, LT, RT],[]]):-
        isALBMAX([R,LT,RT]).
isALBMAX([R,[],[R, LT, RT]]):-
        isALBMAX([R,LT,RT]).
isALBMAX([R,[A,B,C],[D,E,F]]):-
        isALBMAX([A,B,C]),
        isALBMAX([D,E,F]),
        max(A,D,R).

%isALBMAX([4,[2,[],[]],[4,[],[]]]).

filterALBMAX([],[]).
filterALBMAX([T|REST],[T|OUT]):-
    isALBMAX(T),
    filterALBMAX(REST,OUT).
filterALBMAX([T|REST],OUT):-
    not(isALBMAX(T)),
    filterALBMAX(REST,OUT).

%filterALBMAX([[7,[7, [], []], [3, [], []]],[7,[4, [], []], [3, [], []]],
```

```
%[7,[1,  [],  []],  [[]],  OUT).
```