COS4851 Assignment 2 (2019) [90]

ASSIGNMENT 02

DATE:	15 July 2019
SUBMISSION PROCEDURE:	Written
WEIGHT:	20
UNIQUE NUMBER:	684985
STUDY MATERIAL:	*Bratko*, Chapters 4 to 6
	Chapter 4: ignore section 4.6
ADDITIONAL RESOURCES:	Notes on *Bratko* chapters 4, 5 and 6
	COS1501 study guide for principles of set theory and operations on sets.

Important: The main purpose of this assignment is to develop your skills regarding the use of built-in predicates in Prolog, the control of backtracking and the way in which Prolog can be used for simple arithmetic procedures.

Please note that you need to submit screen shots (using *fn+prt sc* for Windows in most instances) of results for questions where a program/procedure or query is required. This will assist us to see whether you obtained the correct results and if not, try to point out where you went wrong.

Your programs should also be *robust*. This means that it should check whether all the input arguments for a specific procedure are legal. For example, if you know you are working with integers, an input of an argument that is not an integer is not acceptable.

Important note: It is not advisable that you search for Prolog solutions to the questions in this assignment on the internet. You may find a solution to a specific problem but that will not assist you in acquiring the necessary skills for mastering this programming paradigm.

Question 1 [12]

(a) Write a Prolog procedure `split(L,N,L1,L2)` to split a given list `L` into two separate lists, `L1` and `L2`, where `L1` contains integers >= `N` and `L2` contains integers

< N. All other items in the given list L should be ignored. Use the cut facility where appropriate. (8)

(b) Briefly discuss the difference between green cuts and red cuts. (4)

Question 2 [10]

(a) Write a procedure filter(List,PredName,R) that returns the list R containing the list of Ys for all the numbers X in List for which PredName(X,Y) succeeds. The predicate PredName/2 should be defined when calling the procedure apply.

Let fun be defined as

```
fun(X,Y):-
      number(X),
      Y is X^3.

?- apply([1,2,3,4],fun,L).
L = [1,8,27,64]
```

NB: Use the Prolog built-in predicate =.. (6)

(b) It is quite simple to simulate arrays in Prolog. Instead of formally defining an array as a lookup table, a facility not available in Prolog, we can use a functor with many arguments and access the individual arguments using the built-in predicate arg, for example primes(2,3,5,7,11,13,17,19,23,29) can be used to represent the first 10 prime numbers. (Note that primes is the *functor* in this instance.)

Use the predicate arg to find the 6th prime number in primes(2,3,5,7,11,13,17,19,23,29). (4)

Question 3 [31]

3.1 Use the database you defined for question 1 of assignment 1 to write the following clauses using setof, bagof or findall:

(a) The clause breeds(L) returns a list L of all the different breeds of dog listed in the database. (4)

(b) The clause count_hunt(L,N) returns a list L of all the dog breeds used for hunting with N giving the total number of hunting breeds. (5)

(c) The clause sizes(N) returns the number of different sizes of dog are defined in the database. (4)

3.2 Lists can be used to represent sets in Prolog. Write the following Prolog procedures using the Prolog built-in predicates `setof`, `bagof` or `findall`.

(a) The procedure `symmetric_difference(A,B,L)` returns the symmetric difference `L` between `A` and `B`, i.e. the set `L` contains all elements that belong to `A` or to `B` but not to both. (9)

(b) The procedure `complement(A,U,C)` returns the complement `C` of set `A` relative to the universal set `U` (`A` is a subset of `U`). The complement of set `A` consists of all those elements that belong to `U` but not to `A`. (9)

Question 4 [8]

Consider the following set of Prolog clauses concerning race horses with exceptional colouring:

```
roan(blue_lightning).
paint(shadows).
clay_bank(colorado).
roan(X):-  not paint(X),
           not clay_bank(X).
```

Sometimes Prolog returns unexpected results, e.g. if you enter the query `roan(flame)` Prolog returns `yes`.

(a) Create the database and test this statement. (2)

(b) Explain why this happens. (5)

(c) What assumption causes the phenomenon mentioned above? (1)

Question 5 [5]

Construct a table listing the symbols that can be used to find the relationship between two terms X and Y and state the meaning of each. (For your own benefit, make sure you understand the meaning of each.)

Question 6 [10]

The Russian Multiplication problem can be defined as follows: Say you want to multiply *x* with *y* giving *z*. The problem is solved using the following iterative loop:

With each iteration, *x* gets the value *x*/2 and *y* gets the value *y**2. If *x* is even, the *y*-entry is ignored. If *x* is odd, *y* is added to a running total. The loop terminates when *x* = 0.

For example:Calculate *z* = 24 * 52.

x	*y*	**Calculation of total (T)**	**Total**
24	52		0
12	104		0
6	208		0
3	416	T = T + 416	416
1	832	T = T + 832	1248
0			**1248**

Write a Prolog program to implement the Russian Multiplication Problem.

Question 7 [14]

Assume that a Prolog database contains the following facts:

```
m(a).      n(a,1).      q(1,1).      q(3,5).
m(b).      n(a,2).      q(1,2).      q(3,6).
           n(b,3).      q(2,3).      q(4,7).
           n(b,4).      q(2,4).      q(4,8).
```

What are the effects of the following queries? List all alternatives.

(a) `?- m(X),n(X,Y),!,q(Y,Z).` (2)

(b) `?- m(X),once(n(X,Y)),q(Y,Z).` (4)

(c) `?- m(X),!,n(X,Y).` (3)

(d) `?- bagof(X,(m(X),X=a,n(X,Y)),L).` (2)

(e) `?- setof(X,Y^q(X,Y),L).` (3)