CSc 372

Comparative Programming Languages

26 : Prolog — Exercises

Department of Computer Science
University of Arizona

collberg@gmail.com

Christian Collberg

Problem I

Write a procedure `islist` which succeeds if its argument is a list, and fails otherwise.

Problem II

Write a procedure `alter` which changes English sentences according to rules given in the database.
Example:

```
change(you, i).
change(are, [am, not]).
change(french, german).
change(do, no).
?- alter([do,you,know,french],X).
   X = [no,i,know,german]
?- alter([you,are,a,computer],X).
   X = [i,[am,not],a,computer]
```

Problem III

Write a list subtraction procedure.
Example:

```
?- sub([1,2,4,6,8], [2,6], L).
   L=[1,4,8].
```

Problem IV

Write a procedure `pick` which returns the first `N` elements of a given list.
Example:

```
?- pick([1,2,4,6,8], 3, L).
   L=[1,2,4].
```

Problem V

Write a procedure `alt` which produces every other element in a list.
Example:

```
?- alt([1,2,3,4,5,6], A).
   A = [1,3,5]
```

Problem VI

Write a procedure `del` which removes duplicate elements from a list.
Example:

```
?- del([a,c,x,a,g,c,d,a], A).
   A = [a,c,x,g,d]
```

Problem VII

Write a procedure `tolower` which converts an atom containing upper case characters to the corresponding atom with only lower case characters.
Example:

```
?- tolower('hEj_HoPp3', A).
   A = hej_hopp3
```

Problem VIII

Write a procedure `max3` which produces the largest of three integers.
Example:

```
?- max3(3,5,1,X).
   X = 5
```

Problem IX

Write a procedure `double` which multiplies each element in a list of numbers by 2.
Example:

```
?- double([1,5,3,9,2], A).
   A = [2,10,6,18,4]
```

Problem X

Write a procedure `ave` which computes the average of a list of numbers.
Example:

```
?- ave([1,5,3,9,2], A).
   A = 4
```

Problem XI

Write a procedure `sum` which produces the sum of the integers up to and including its first argument.
Example:

```
?- sum(5, S).
   S = 15
```

Problem XII

Suppose our database contains facts of the form

```
person_age(Name, Age).
person_sex(Name, Sex).
```

where `Sex` is either `male` or `female`. Write a procedure `combine` which extends the database with additional facts of the form

```
person_full(Name, Age, Sex).
```

The procedure should produce one such fact for each person who has both an age record and a sex record.

Problem XII...

Example: Given the following database

```
person_age(chris, 25).  % Yeah, right...
person_sex(chris, male).
person_age(louise, 8).
person_sex(louise, female).
```

`combine` should produce these additional facts:

```
person_full(chris, 25, male).
person_full(louise, 8, female).
```

Problem XIII

Write a Prolog procedure which reverses the order of Johns children in the database. For example, given the following database

```
child(mary, john).
child(jane, john).
child(bill, john).
```

the goal `?- reversefacts.` should change it to

```
child(bill, john).
child(jane, john).
child(mary, john).
```

Problem XIV

Write a Prolog procedure to assemble a list of someone's children from the facts in the database. The database should remain unchanged.
Example:

```
child(mary, john).
child(jane, john).
child(bill, john).

?- assemble(john, L).
   L = [mary, jane, bill]
```

Problem XV

Write down the *all* results (including variable bindings) of the following query:

```
?- append([], [1, 2|B], C),
   append([3,4], [5], B).
```

Problem XVI

Write down the *all* results (including variable bindings) of the following query:

```
?- bagof(X, Y^append(X, Y, [1,2,3,4]), Xs).
```

Problem XVII

Write down the *all* results (including variable bindings) of the following query:

```
?- L=[1,2], member(X, L), delete(X, Y, L).
```

Problem XVIII

Write down the *all* results (including variable bindings) of the following query:

```
?- member(X, [a,b,c]), member(Y, [a,b,c]), !, X \= Y.
```

Problem XIX

Given the following Prolog database

```
balance(john, 100).
balance(sue, 200).
balance(mary, 100).
balance(paul, 500).
```

list *all* the results of these Prolog queries:

1. `?- bagof(Name, balance(Name, Amount), Names).`
2. `?- bagof(Name, Amount^balance(Name, Amount), Names).`
3. `?- bagof(Name, Name^balance(Name, Amount), Names).`

Problem XX

Describe (in English) what the following predicate does:

```
% Both arguments to bbb are lists.
bbb([], []).
bbb(A, [X|F]) :- append(F, [X], A).
```

Problem XXI

Given the following program

```
a(1,2).
a(3,5).
a(R, S) :- b(R, S), b(S, R).

b(1,3).
b(2,3).
b(3, T) :- b(2, T), b(1, T).
```

list the first answer to this query:

```
?- a(X, Y), b(X, Y)
```

Will there be more than one answer?

Problem XXII

Given the following definitions:

```
f(1, one).
f(s(1), two).
f(s(s(1)), three).
f(s(s(s(X))), N) :- f(X, N).
```

what are the results of these queries? If there is more than one possible answer, give at least two.

1. `?- f(s(1), A).`
2. `?- f(s(s(1), two).`
3. `?- f(s(s(s(s(s(s(1)))))), C).`
4. `?- f(D, three).`

Problem XXIII

Write a Prolog predicate `sum_abs_diffs(List1, List2, Diffs)` which sums the absolute differences between two integer lists of the same length.
Example:

```
?- sum_abs_diffs([1,2,3], [5,4,2], X).
      X = 7 % abs(1-5) + abs(2-4) + abs(3-2)
```

Problem XXIV

Write a Prolog predicate `transpose(A, AT)` which transposes a rectangular matrix given in row-major order.
Example:

```
?- transpose([[1, 2], [3, 4]], AT).
      AT = [[1, 3], [2, 4]]
```

Problem XXV

Write Prolog predicates that given a database of countries and cities

```
% country(name, population (in thousands),
% capital).
country(sweden, 8823, stockholm).
country(usa, 221000, washington).
country(france, 56000, paris).
% city(name, in_country, population).
city(lund, sweden, 88).
city(paris, usa, 1).  % Paris, Texas.
```

Problem XXV...

Answer the following queries:

1. Which countries have cities with the same name as capitals of other countries?
2. In how many countries do more than $\frac{1}{3}$ of the population live in the capital?
3. Which capitals have a population more than 3 times larger than that of the secondmost populous city?

Problem XXV...

```
%country(name, population (in thousands), capital).
country(sweden, 8823, stockholm).
country(usa, 221000, washington).
country(france, 56000, paris).
country(denmark, 3400, copenhagen).
% city(name, in_country, population).
city(lund, sweden, 88).
city(new_york, usa, 5000).  % Paris, Texas.
city(paris, usa, 1).  % Paris, Texas.
city(copenhagen, denmark, 1200).
city(aarhus, denmark, 330).
city(odense, denmark, 120).
city(stockholm, sweden, 1300).
city(gothenburg, sweden, 350).
city(washington, usa, 3400).
city(paris, france, 2000).
city(marseilles, france, 1000).
```

Problem XXVI

Write a Prolog predicate that extracts all words immediately following "the" in a given list of words.
Example:

```
?- find([the, man, closed, the, door,
         of, the, house], X).
   X = [man, door, house]
```

Problem XXVII (Midterm Exam 372/04)

Write a Prolog predicate dup that duplicates each element of a list. Example:

```
?- dup([2,5,x], A).
   A = [2,2,5,5,x,x]
```

Problem XXVIII (Midterm Exam 372/04)

The following Prolog program evaluates constant expressions:

```
eval(A+B, V) :- eval(A, V1), eval(B, V2),
         V is V1 + V2.

eval(A*B, V) :- eval(A, V1), eval(B, V2),
         V is V1 * V2.

eval(X, X) :- integer(X).

?- eval(3*4+5, V).
   V = 17
```

Problem XXVIII... (Midterm Exam 372/04)

Modify the program so that it allows the expression to contain variables. Variable values should be taken from an environment (a list of variable/value pairs), like this:

```
?- eval([x=3,y=4], x*y+5, V).
   V = 17
?- eval([x=3], x*y+5, V).
   no
```

Problem XXIX (Midterm Exam 372/04)

Write a predicate `mult` which, for all pairs of numbers between 0 and 9, adds their product to the Prolog database. I.e., the following facts should be asserted:

```
times(0, 0, 0).   % 0*0=0
times(0, 1, 0).   % 0*1=0
...
times(9, 7, 63).  % 9*7=63
times(9, 8, 72).  % 9*8=72
times(9, 9, 81).  % 9*9=81
```

The interaction should be as follows:

```
?- times(5,5,X).
no
?- mult.
yes
?- times(5,5,X).
X=25
?- times(2,9,18).
yes
```

Problem XXX (Midterm Exam 372/04)

Use a *2nd-order-predicate* to write a predicate `alltimes(L)` which, given the `times(X,Y,Z)` database above produces a list of all the multiplication facts:

```
?- alltimes(L).
L = [1*1=2,1*2=2,1*3=3,...,9*9=81].
```

Problem XXXI (Midterm Exam 372/04)

Show the results (`yes`/`no`) and resulting variable bindings for the following queries:

a) `?- f(g(X,X), h(Y,Y)) = f(g(Z), Z).`

b) `?- f(g(X,X), h(Y,Y)) = f(g(h(W,a),Z), Z).`

c) `?- f(g(X,X), h(_,_)) = f(g(h(W,a),Z), Z).`

d) `?- f(x(A,B),C ) = f( C,x(B,A)).`

Problem XXXII (Final Exam 372/04)

Given this Prolog predicate definition

```
mystery(L, B) :-
   member(X, L),
   append(A,[X],L),
   append(B,C,A),
   length(B,BL),
   length(C,CL),
   BL > CL.
```

what does the query

```
| ?- mystery([1,2,3,4,5],C), write(C), nl, fail.
```

print?