CSc 372, Fall 2001
Prolog Examination Solutions

Problem 1: (5 points)

Show an example of each of the following:

A fact: `p.`

A rule: `p :- q.`

A query: `p.`

A clause: `p.`

An atom: `p`

Problem 2: (2 points)

What is the relationship between facts, rules, and clauses?

Facts and rules are clauses.

Problem 3: (5 points)

True or false: The following is a working implementation of the `member` predicate, as studied in class:

```
member(X, [X]).
member(X, [_|T]) :- member(X, T).
```

False—it only succeeds if a value is the last element of the list.

Problem 4: (5 points)

True or false: The following is a working implementation of the `length` predicate, as studied in class:

```
length([], 0).
length([_|T], Sum) :- length(T, Sum), Sum is Sum + 1.
```

False—`Sum is Sum + 1` fails.

Problem 5: (12 points)

Write a predicate `sumval(+List, +Value, -Sum)` that produces the sum of all occurrences of `Value` in the list `List`. Assume that `Value` and all elements in `List` are integers.

```
sumval([ ], _, 0).
sumval([X|T], X, Sum) :- sumval(T, X, TSum), Sum is X + TSum, !.
sumval([_|T], X, Sum) :- sumval(T, X, Sum).
```

Problem 6: (4 points)

Write a predicate `sumvals(+List, +ListOfValues, -Sum)` that produces the sum of all occurrences of members of the list `ListOfValues` in the list `List`. Assume that `Value` and all elements in `List` are integers.

The order of values in `ListOfValues` is inconsequential. Note that a given value may appear multiple times in `ListOfValues` but that does not affect the result.

```
sumvals([], _, 0).
sumvals([H|T], L, Sum) :- member(H, L), sumvals(T, L, TSum),
                          Sum is H + TSum, !.
sumvals([_|T], L, Sum) :- sumvals(T, L, Sum).
```

Problem 7: (12 points)

Write a predicate `listeq(+L1, +L2)` that succeeds if the lists `L1` and `L2` are identical and fails otherwise. `L1` and `L2` may be arbitrarily complicated lists but all values will be either integers or lists.

```
listeq(L,L).
```

Problem 8: (15 points)

Write a predicate `consec(+Value, +N, +List)` that succeeds if and only if `List` contains `N` consecutive occurrences of `Value`. Assume that `Value` and all elements of `List` are atoms or integers. Assume `N > 0`.

```
consec(Val, N, List) :- repl(Val, N, Vals),
                        append(Vals, _, List), !.

consec(Val, N, [_|T]) :- consec(Val, N, T).
```

Problem 9: (15 points)

Write a predicate `order3(+L1, -L2)` that assumes that `L1` contains three integers and instantiates `L2` to be a list of those integers in ascending order:

```
order3(L,[A,B,C]) :-
     getone(A,L,R), getone(B,R,[C]), A =< B, B=< C, !.
```

Problem 10: (20 points)

In this problem you are to write a predicate `inventory/0` that does an inventory calculation for a fruit stand. [...]

```
inventory :- fruit(F), get_qty(F,Q), cost(F,C), TC is Q*C/100,
             format('~p: ~p at ~p = $~p~n', [F,Q,C,TC]), fail.
inventory.

get_qty(F,Q) :- qty(F,Q), !.
get_qty(_,0).
```

Problem 11: (5 points)

For each of the two following queries, write in the values computed for each variable. If a query fails, indicate it.

```
| ?- X = [1,2,3], Y = [4|X], [A,B|C] = Y.

     A = 4
     B = 1
     C = [2,3]

| ?- A = [], B = 1, C = [A,B], B = 2, [D,E] = C.
```

This query fails because `B` can't be unified with both `1` and `2`.

EXTRA CREDIT SECTION (one point each)

(a) What is the Prolog 1000?

A list of significant applications written in Prolog and related languages.

(b) In what country was Prolog developed?

France

(c) What country made a big investment in Prolog?

Japan, with its Fifth Generation project.

(d) What language was used for the first implementation of Prolog?

FORTRAN

(e) What is the sound of a combinatorial explosion?

Silence

(f) What is inaccurate about this specification: `append(+L1,+L2,-L3)`?

The correct specification is `append(?L1, ?L2, ?L3)`. All, some, or none of the arguments might be specified.

(g) Why is a warning about a singleton variable significant?

The variable is question is used only once. That might indicate a misspelled name or other error.

(h) In a Prolog library you see these two predicates: `get_chr(+Number, -Char)` and `get_ord(+Char, -Number)`. What's odd about that?

With Prolog, one predicate can perform both calculations.