1: Append in Prolog II

In Prolog, one list (written as [a,b,c] or [] for the empty list) is appended to another by the following code:

```
acc_append([], Ys, Ys).
acc_append([X|Xs], Ys, [X|Zs]) :- acc_append(Xs, Ys, Zs).
```

What happens if you call acc_append „in reverse" – giving variables for the two input lists and a result list for the accumulator/result variable? Why? Trace the Prolog query and show the search tree for acc_append(X, Y, [1,2]).

2: List-reverse in Prolog

Here is the LISP code for list reverse using an accumulator. Re-write it in Prolog.

```
(define list-reverse-aux (lambda (L A)
             (if (null? L) A
                 (list-reverse-aux (cdr L) (cons (car L) A)))))

(define list-reverse (lambda (L) (list-reverse-aux L '())))
```

Hint 1: There is no cons predicate; use the [Head|Rest] notation to create a new list.

Hint 2: If you want to see the result of the list reversal, your query will have to contain an unbound variable that takes on the value of A eventually (in addition to the actual accumulator).

3: Optional: Logic puzzle (from Learn Prolog Now! with thanks)

There is a street with three neighbouring houses that all have a different colour, namely red, blue, and green. People of different nationalities live in the different houses and they all have a different pet. Here are some more facts about them:

- The Englishman lives in the red house.
- The jaguar is the pet of the Spanish family.
- The Japanese lives in the blue house.
- The snail keeper lives to the left of the blue house.

Who keeps the zebra? Define a predicate zebra/1 that tells you the nationality of the owner of the zebra!

Hint 1: Think of a representation for the houses and the street. Code the constraints in Prolog.

Hint 2: member/2 checks whether a term ist he member of a given list.

Hint 3: is/2 does arithmetic computations – e.g., is(X,7+2).