CM20019 – Prolog Lab –Part 3

This is an important exercise, whose purpose is to give you practice in writing recursive list-processing programs. Test each one with representative queries.

Preliminary

Enter `:- use_module(library(lists))` at the start of your Prolog script to enable access to the Sicstus library of list-processing predicates.

1. Write a program for the relation ones(X) which holds when X is a list each of whose members, if any, is 1.

Refer only to this relation.

2. Write a program for the relation ones_zeros(X) which holds when X is a list each of whose members, if any, is either 1 or 0.

Refer only to this relation.

3. Write a program for the relation alt_ones_zeros(X) which holds when X is a non-empty list of even length containing alternate 1s and 0s, starting with 1.

Refer only to this relation.

4. Write a program for the relation hasdups(X) which holds when X contains at least two occurrences of some member.

Refer only to this relation and the primitive member(U, X).

5. Write a program for the relation scrub(U, X, Y) which holds when Y is the list obtained by deleting all occurrences, if any, of the member U from the list X.

Refer only to this relation and the primitive \==.

6. Write a program for the relation dups(X, Y, D) which holds when D is a duplicate-free list whose members, if any, are those occurring at least twice in X. Use the middle argument Y to accumulate the distinct members that D will eventually contain, making D equal to Y only at the end of the computation.
The initial query will have [] as the middle argument.

Refer only to this relation and (optionally) scrub
and the primitives member(U, X) and non_member(U, X).