Some Prolog Practice Questions

Define the following predicates in Prolog using any auxiliary predicates you wish.

1. **subList(L1, L2)** to mean every element in list L1 is also in list L2.

You can assume both arguments are grounded in the call.

E.g. sublist([1,2,3], [1,1,3,2, 3,4]) should succeed.

2. **intersect (L1, L2, I)** to mean I is the intersection of lists L1 and L2.

You can assume both arguments L1 and L2 are grounded in the call. Intersection should have no repeated elements.

a) The element in output I should be in ascending order.

E.g. intersect([1,4,2,5], [1,1,7,2, 3,4, 4, 8], I) should give the answer I=[1,2,4].

Make sure you deal correctly with the case where L1 and L2 have no elements in common.

E.g. intersect([1, 2], [4, 8], I) should give the answer I=[], not the answer no.

b) The elements in output I should be in ascending order of the number of their occurrences in L2. If two elements in the intersection occur the same number of times in L2 then they should be in I in ascending order.

E.g. intersect([1,4,2,5], [1,1,7,2, 2, 2, 3,4, 8], I) should give the answer I=[4, 1, 2].

(b) is not as easy as (a), but you can do it!

Hint: First use (a) to find the elements in the intersection, then order these elements and the number of times they occur in L2 in ascending order of the latter (using setof), and then process the resulting list by dropping the counts and retaining the elements in their right order.

3. **disjoint(L1, L2)** to mean L1 and L2 share no elements, and at least one of them is not empty.

You can assume both arguments are grounded in the call.

disjoint([1,2,3], [a,b,c]) should succeed, and disjoint([1,b,3], [a,b,c]) should fail.

4. **difference(L1, L2, L)** to mean L consists of all the elements in L1 that are not in L2.

You can assume both arguments L1 and L2 are grounded in the call.

E.g. difference([1,1,2,3, 5, 5], [1,3,2, 3,4], L) should give L=([5, 5]. The repetition in the list L is fine and you may get the same answer more than once. That too is fine.

difference([a, c], [], L) should give L=([a,c].

5. **sift(L, N, Result)** to mean Result is list L but with all occurrences of elements greater than N removed. You can assume both arguments L and N are grounded in the call.

E.g. sift([1,4,3,6,8], 3, X) should give X=[1,3].

6. **process(L1, L2, Consistent, Inconsitent)** where L1 is a given list of items of the form (Name, Number), and L2 is a given list of items of the form (Name, Number, MoreInfo). Then the output Consistent should be those items (Name, Number, MoreInfo) in L2 that agree on (Name, Number) with list L1, and Inconsitent should be whatever is left over from list L2.

E.g. Suppose L1 has (Name , Age) items and L2 has (Name, Age, Marital_status) items.

Then Consistent should be those items (Name, Age, Marital_status) where for the same Name L1 provides the same Age.

E.g. process([(mary, 20), (john, 30), (pete, 40)], [(mary, 20, single), (pete, 40, single), (joe, 35, widowed), (john, 35, married)], C, I)

should give

C= ([(mary, 20, single), (pete, 40, single)]

I= ([(john, 35, married), (joe, 35, widowed].

The order of the elements in C and I is not important.

The Bratko book has many exercises.