Some Prolog Practice Questions

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

Try to avoid using the cut.

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]) and **subList([1,1,4,3], [5,1,3,2,3,4])** should both succeed.

2. **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].

3. **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].

4) **common(L1, L2, I)**
to mean **I** is the list of the common elements of lists **L1** and **L2**.

You can assume both arguments **L1** and **L2** are grounded in the call. The resulting list **I** should have no repeated elements. The order of the elements in list **I** is not important. If **L1** and **L2** have no common elements then the output **I** should be the empty list [].

E.g.

common([1,1,4,2,5], [1,1,7,2,3,4,4,8], I) should give the answer **I**=[1,2,4], but the order of the elements in **I** does not matter.

common([1,2], [4,8], I) should give the answer **I**=[].

5. **process(L1, L2, Consistent, Inconsistent)**

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 Inconsistent 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 the answer

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.

6. split(L, N, L1, L2)
Split a list L into two parts L1 and L2 such that the length of the first part is N.

Example:
?- split([a,b,c,d,e,f,g,h,i,k],3,L1,L2).
L1 = [a,b,c]
L2 = [d,e,f,g,h,i,k]

7. drop(L, N, Result) Drop every N'th element from a list L.

Example:
?- drop([a,b,c,d,e,f,g,h,i,k],3,X).
X = [a,b,d,e,g,h,k]

8. enrolment(L, Student, Degree)
Given a list L of enrolments, and a student's name, Student, the program finds the degree of the student.

L is a list of students and their degree. Each element of L is of the form
(Degree, List of students).

For example given
[(msc, [john, mary, pete]), (meng, [bob, rob, tod]), (msc, [dave, mave])]
as L, and
rob as Student, the program should return meng as Degree.

9. student_list(L, Meng, MSc)
Separate a list L of students into the Meng students and the MSc students.
L is a list of students and their degree. Each element of L is of the form
(Degree, List of students).

For example given
[(msc, [john, mary, pete]), (meng, [bob, rob, tod]), (msc, [dave, mave])]
as L
Meng should be
[bob, rob, tod]
and MSc should be
[john, mary, pete, dave, mave],
although the order of the names in these lists is not important.

The Bratko book recommended in the slides has many exercises.