

A list is either empty or it is composed of a first element (head) and a tail, which is a list itself. In Prolog we represent the empty list by the atom [] and a non-empty list by a term [H|T] where H denotes the head and T denotes the tail.

1.01 Find the last element of a list.

Example:

```
?- my_last(X,[a,b,c,d]).
```

```
X = d
```

1.02 Find the last but one element of a list.

1.03 Find the K'th element of a list.

The first element in the list is number 1.

Example:

```
?- element_at(X,[a,b,c,d,e],3).
```

```
X = c
```

1.04 Find the number of elements of a list.

1.05 Reverse a list.

1.06 Find out whether a list is a palindrome.

A palindrome can be read forward or backward; e.g. [x,a,m,a,x].

1.07 Flatten a nested list structure.

Transform a list, possibly holding lists as elements into a 'flat' list by replacing each list with its elements (recursively).

Example:

```
?- my_flatten([a, [b, [c, d], e]], X).
```

```
X = [a, b, c, d, e]
```

Hint: Use the predefined predicates is_list/1 and append/3

1.08 Eliminate consecutive duplicates of list elements.

If a list contains repeated elements they should be replaced with a single copy of the element. The order of the elements should not be changed.

Example:

```
?- compress([a,a,a,a,b,c,c,a,a,d,e,e,e,e],X).
```

```
X = [a,b,c,a,d,e]
```

1.09 Pack consecutive duplicates of list elements into sublists.

If a list contains repeated elements they should be placed in separate sublists.

Example:

```
?- pack([a,a,a,a,b,c,c,a,a,d,e,e,e,e],X).
```

```
X = [[a,a,a,a],[b],[c,c],[a,a],[d],[e,e,e,e]]
```

1.10 Run-length encoding of a list.

Use the result of problem 1.09 to implement the so-called run-length encoding data compression method. Consecutive duplicates of elements are encoded as terms [N,E] where N is the number of duplicates of the element E.

Example:

```
?- encode([a,a,a,a,b,c,c,a,a,d,e,e,e,e],X).
```

```
X = [[4,a],[1,b],[2,c],[2,a],[1,d],[4,e]]
```

1.11 Modified run-length encoding.

Modify the result of problem 1.10 in such a way that if an element has no duplicates it is simply copied into the result list. Only elements with duplicates are transferred as [N,E] terms.

Example:

```
?- encode_modified([a,a,a,b,c,c,a,a,d,e,e,e,e],X).
```

```
X = [[4,a],b,[2,c],[2,a],d,[4,e]]
```

1.12 Decode a run-length encoded list.

Given a run-length code list generated as specified in problem 1.11. Construct its uncompressed version.

1.13 Run-length encoding of a list (direct solution).

Implement the so-called run-length encoding data compression method directly. I.e. don't explicitly create the sublists containing the duplicates, as in problem 1.09, but only count them. As in problem 1.11, simplify the result list by replacing the singleton terms [1,X] by X.

Example:

```
?- encode_direct([a,a,a,b,c,c,a,a,d,e,e,e,e],X).
```

```
X = [[4,a],b,[2,c],[2,a],d,[4,e]]
```

1.14 Duplicate the elements of a list.

Example:

```
?- dupli([a,b,c,c,d],X).
```

```
X = [a,a,b,b,c,c,c,c,d,d]
```

1.15 Duplicate the elements of a list a given number of times.

Example:

```
?- dupli([a,b,c],3,X).
```

```
X = [a,a,a,b,b,b,c,c,c]
```

What are the results of the goal:

```
?- dupli(X,3,Y).
```

1.16 Drop every N'th element from a list.

Example:

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

```
X = [a,b,d,e,g,h,k]
```

1.17 Split a list into two parts; the length of the first part is given.

Do not use any predefined predicates.

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]
```

1.18 Extract a slice from a list.

Given two indices, I and K, the slice is the list containing the elements between the I'th and K'th element of the original list (both limits included). Start counting the elements with 1.

Example:

```
?- slice([a,b,c,d,e,f,g,h,i,k],3,7,L).
```

```
L = [c,d,e,f,g]
```

1.19 Rotate a list N places to the left.

Examples:

```
?- rotate([a,b,c,d,e,f,g,h],3,X).
```

```
X = [d,e,f,g,h,a,b,c]
```

```
?- rotate([a,b,c,d,e,f,g,h],-2,X).
```

```
X = [g,h,a,b,c,d,e,f]
```

Hint: Use the predefined predicates `length/2` and `append/3`, as well as the result of problem 1.17.

1.20 Remove the K'th element from a list.

Example:

```
?- remove_at(X,[a,b,c,d],2,R).
```

```
X = b
```

```
R = [a,c,d]
```

1.21 Insert an element at a given position into a list.

Example:

```
?- insert_at(alfa,[a,b,c,d],2,L).
```

```
L = [a,alfa,b,c,d]
```

1.22 Create a list containing all integers within a given range.

Example:

```
?- range(4,9,L).
```

```
L = [4,5,6,7,8,9]
```