Q 1

Define an iterative function UNIQ that takes a list and removes all the repeated elements in the list keeping only the first occurrence. This is the expected behavior:

```
* (uniq '(a b r a c a d a b r a))
(A B R C D)
```

Q 2

The mean of n numbers is computed by dividing their sum by n. A running mean is a mean that gets updated as we encounter more numbers. Observe the following input-output sequences:

```
* (run-mean '(3 5 7 9))
(3 4 5 6)
```

The first element 3 is the mean of the list (3), the second element 4 is the mean of (3 5), and so on. Implement RUN-MEAN by using DOTIMES and NTH.

Q 3

See the PAIRLISTS in lecture notes. Define a function that "pairs" an arbitrary number of lists. Here is a sample interaction:

```
* (pairlists '((a b) (= =) (1 2) (+ -) (3 9)))
((A = 1 + 3) (B = 2 - 9))
```

Q 4

Define a function SEARCH-POS that takes a list as search item, another list as a search list and returns the list of positions that the search item is found in the search list. As usual, positioning starts with 0. Use DOTIMES. A sample interaction:

```
* (search-pos '(a b) '(a b c d a b a b))
(6 4 0)
* (search-pos '(a a) '(a a a a b a b))
(2 1 0)
```

Q 5

In one of the previous assignments, you were asked to define a custom REVERSE function that reversed a list using DOLIST. Now slightly alter that definition in such a way that your new function REVERSE0 not only reverses the order of the top-level elements in the list but also reverses any members which are themselves lists.

```
* (reverse0 '(a (b c (x d)) k))
(K ((D X) C B) A)
```

Q 6

Now define a list reversing function REVERSE1 which reverses the order of the top-level elements of a list. This time do NOT use iteration with DOLIST, do everything with recursion.

```
* (reverse1 '(a (b c (x d)) k))
(K (B C (X D)) A)
```

Q 7

Now alter REVERSE1 to REVERSE2 so that it reverses any embedded lists as well. It should behave exactly like REVERSE0 above, but it should do this without using iteration.

Q 8

Define a recursive function `HOW-MANY?` that counts the top-level occurrences of an obj in a list.

```
(how-many? 'a '(a b r a c a d a b r a))
5
```

Q 9

Define a recursive function `R-REMOVE` which behaves exactly like the built-in `REMOVE`.