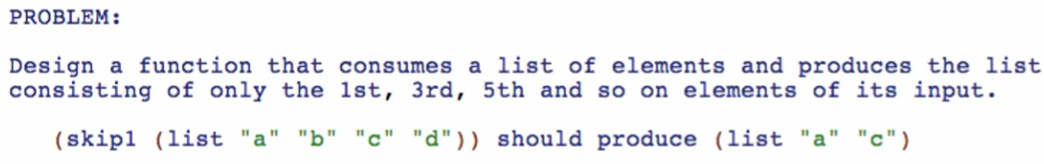
Context Preserving Accumulator

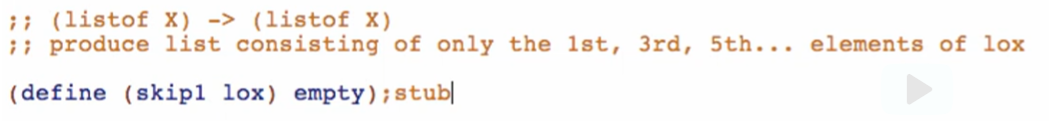
* Preserves context that otherwise gets lost in structural recursion

Not immediately knowing that we need an accumulator

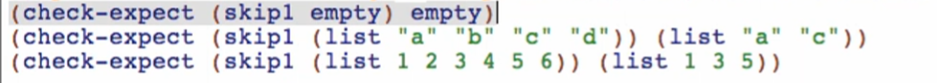


***Function Definition***

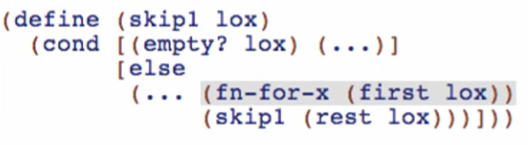
**Signature, purpose, and stub**

****

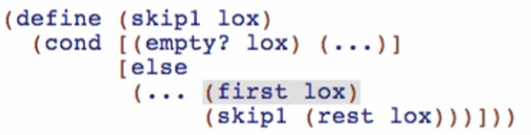
**Examples**

****

**Template**

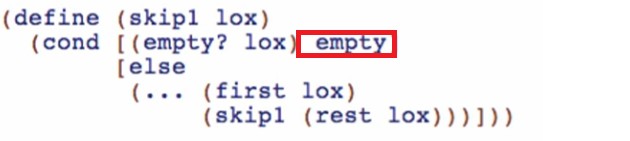
****

But since this function operates on any X, it cannot contain a function which operates on a specific type. So we will remove fn-for-x:

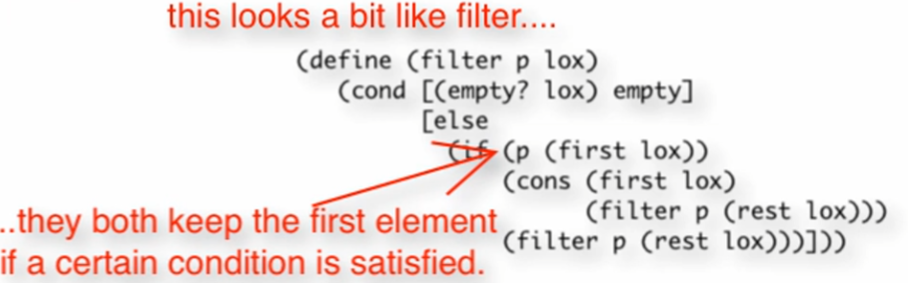


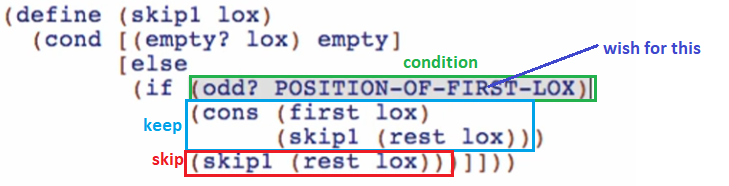
**Code Body**

Base case empty



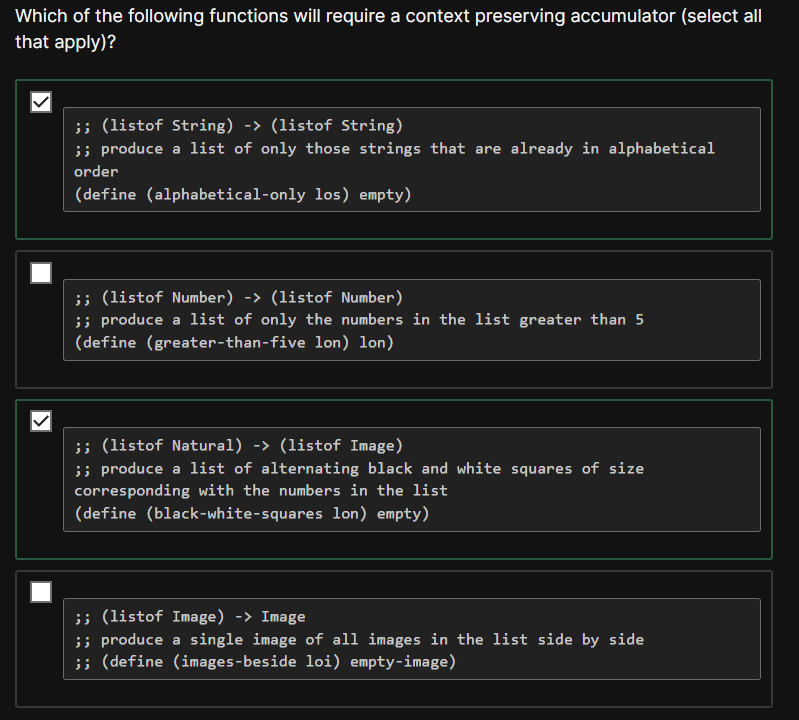
Other cases

* This is like a filtering thing
* 



* Context information about how far we are (position of our current index) is lost in the structural recursion template
  + We need a way to preserve this context of how far we’ve gone (position of our current index inside the list) like a counter
  + This is one kind of problem that we are solving when using accumulators

Questions



Explanation

In **alphabetical-only** we must make sure that each subsequent element of the list is alphabetically after the previous element. To do this, we must keep track of the previous element of the list using a context-preserving accumulator.

In **black-white-squares** we need a context-preserving accumulator to decide whether the current square will be black or white.