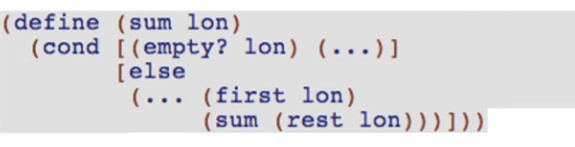
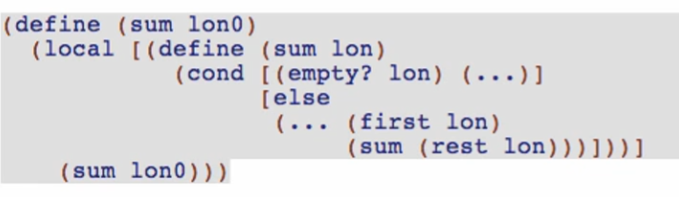
Redo the template using the recipe for accumulator template

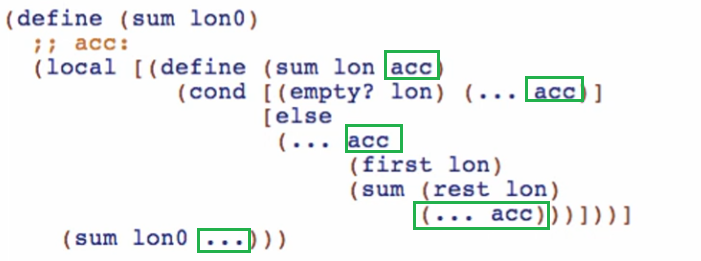
1. Structural recursion



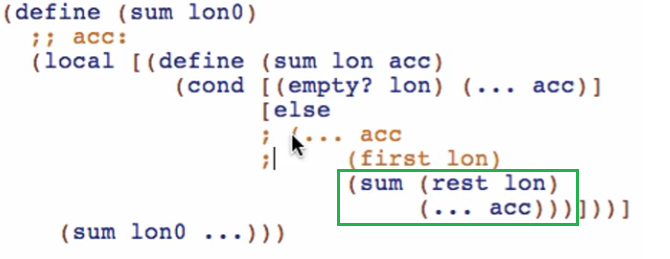
1. Outer function, local, and trampoline



1. Add the accumulator parameter

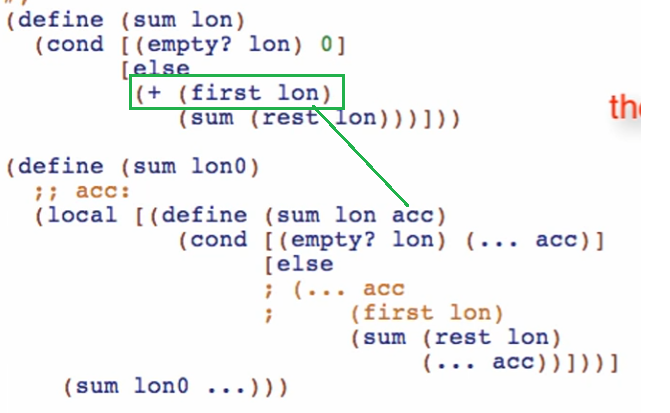


Our recursive call should be the tail position for the else statement!



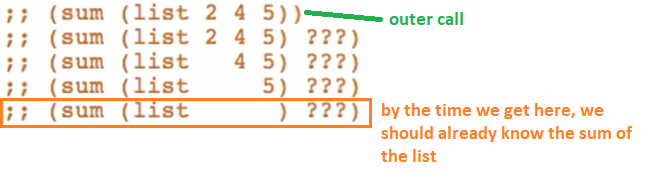
* Comment out the other expressions of the else body and retain only the recursive call

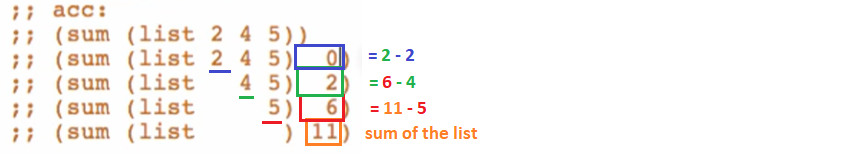
Somehow we need the (+ (first lon) to be in the **accumulator argument**



Working through the **accumulator** type, invariant and examples

*Examples:*



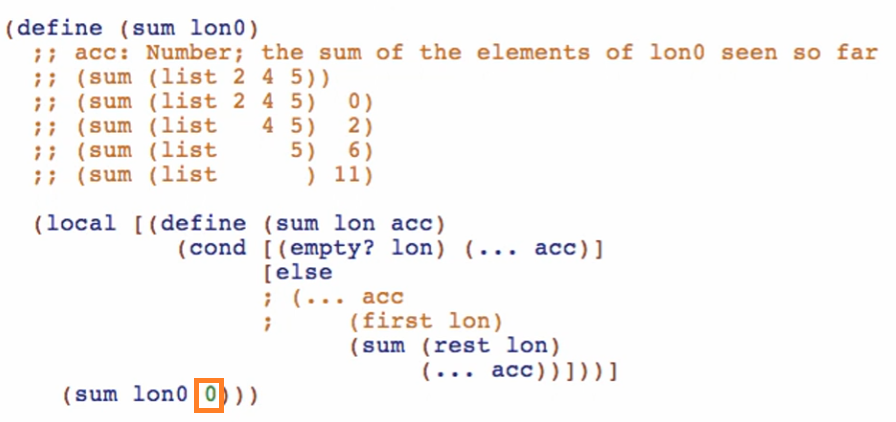


*Type and invariant:*

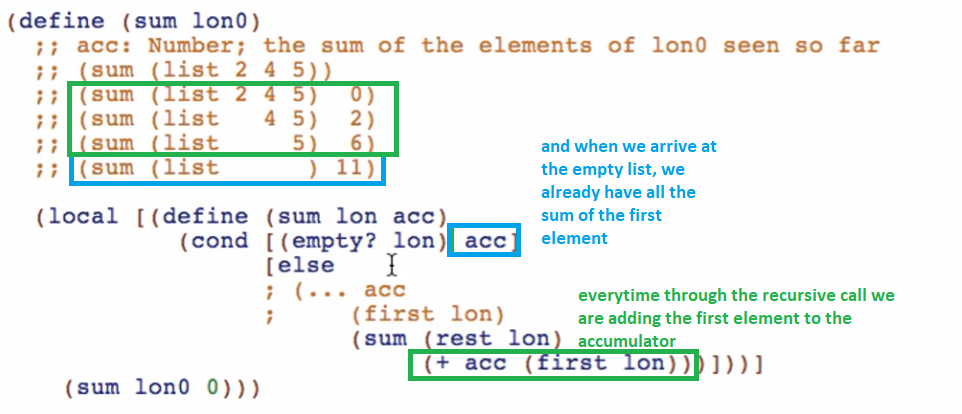


**Code body**

1. Initialize the accumulator

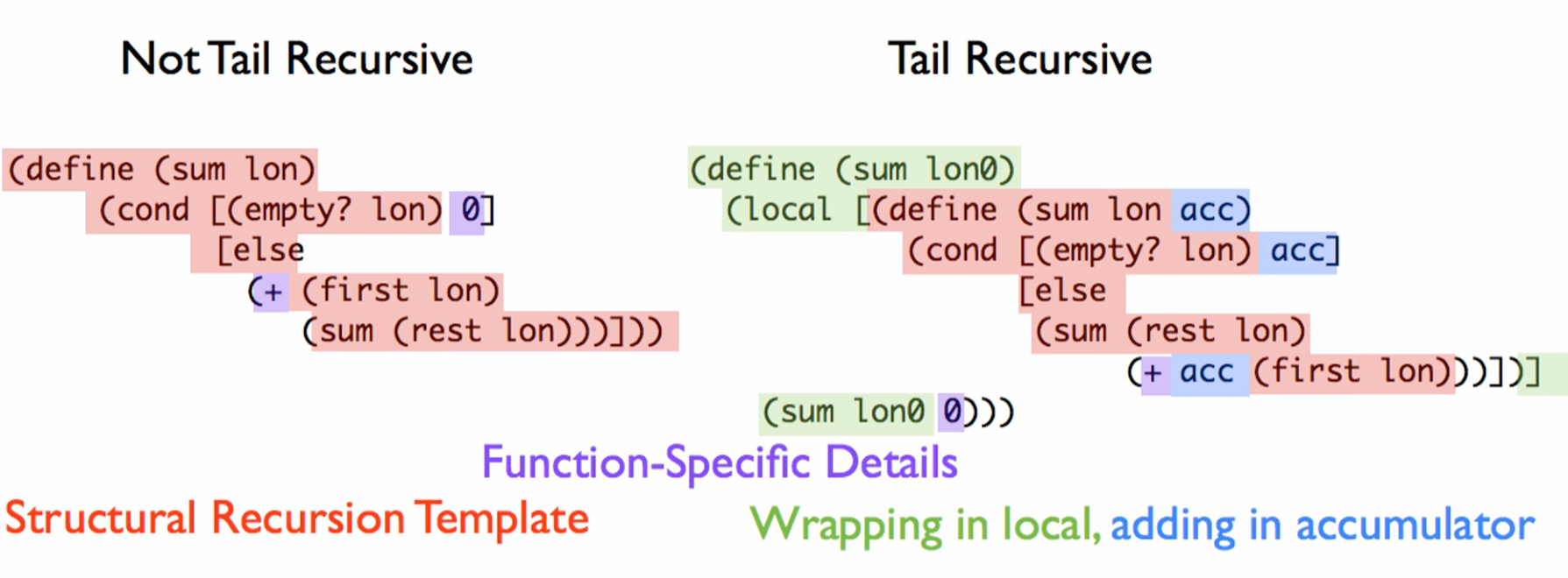


1. Update the accumulator



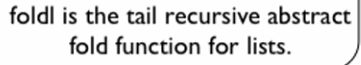
Run and debug

Comparison Overview



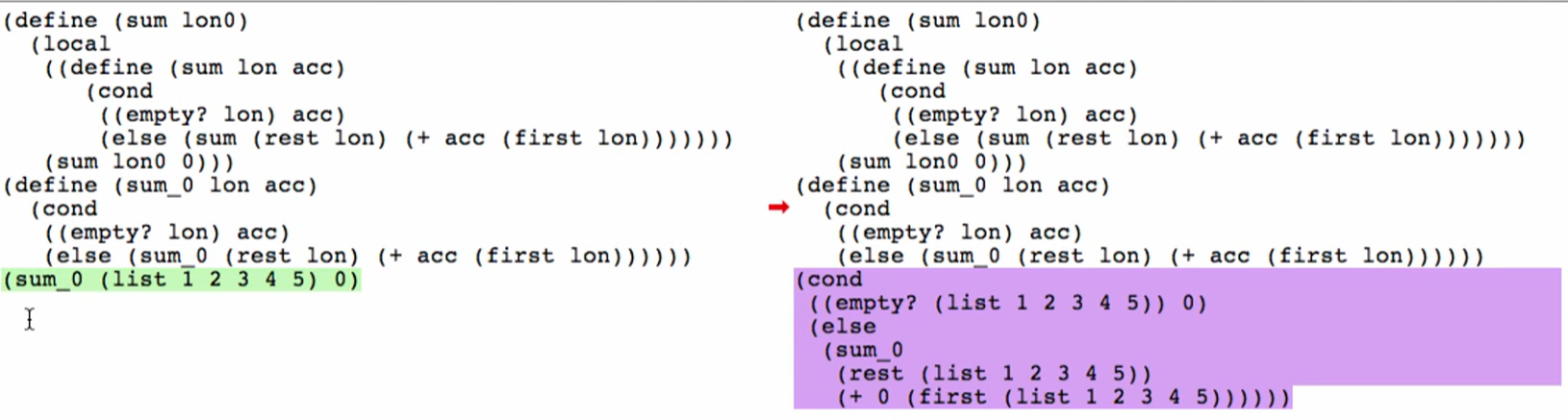
Function specific details are in different positions!

Fun fact:

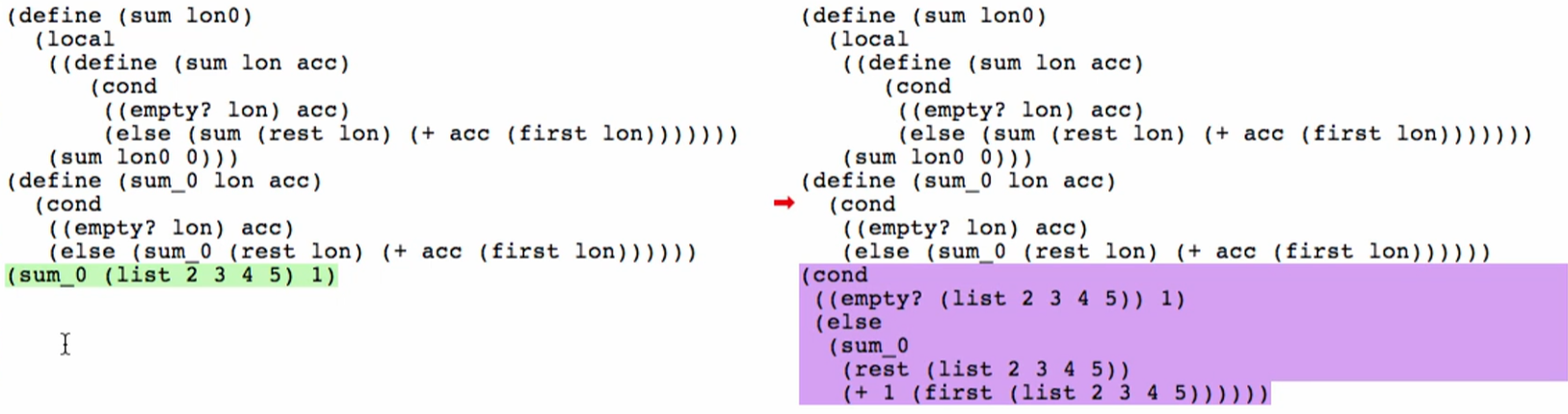


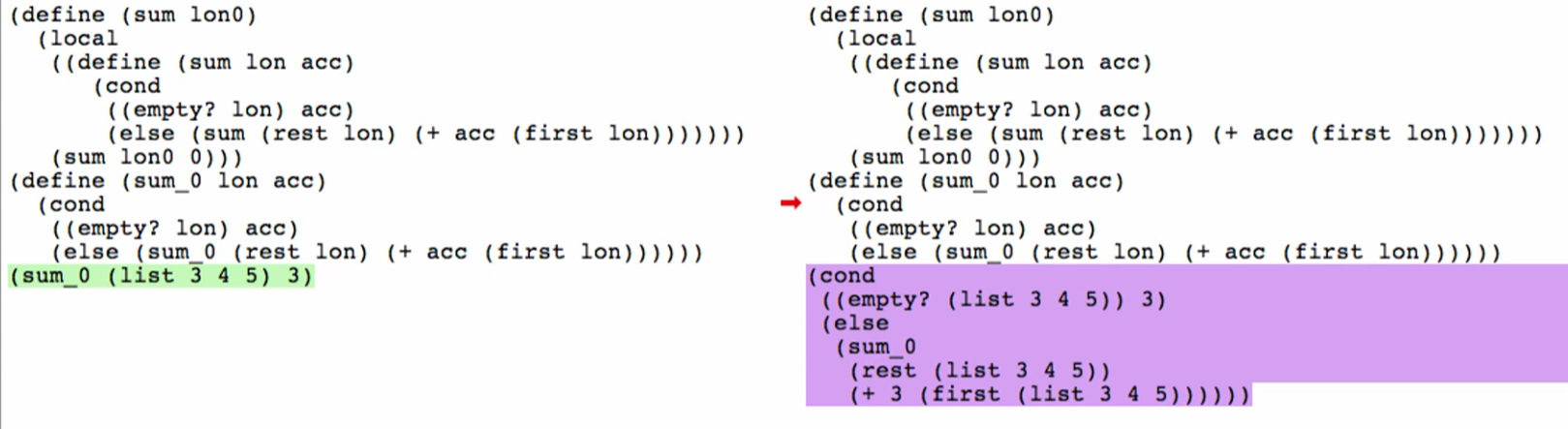
Checking if it really makes a difference. Use stepper

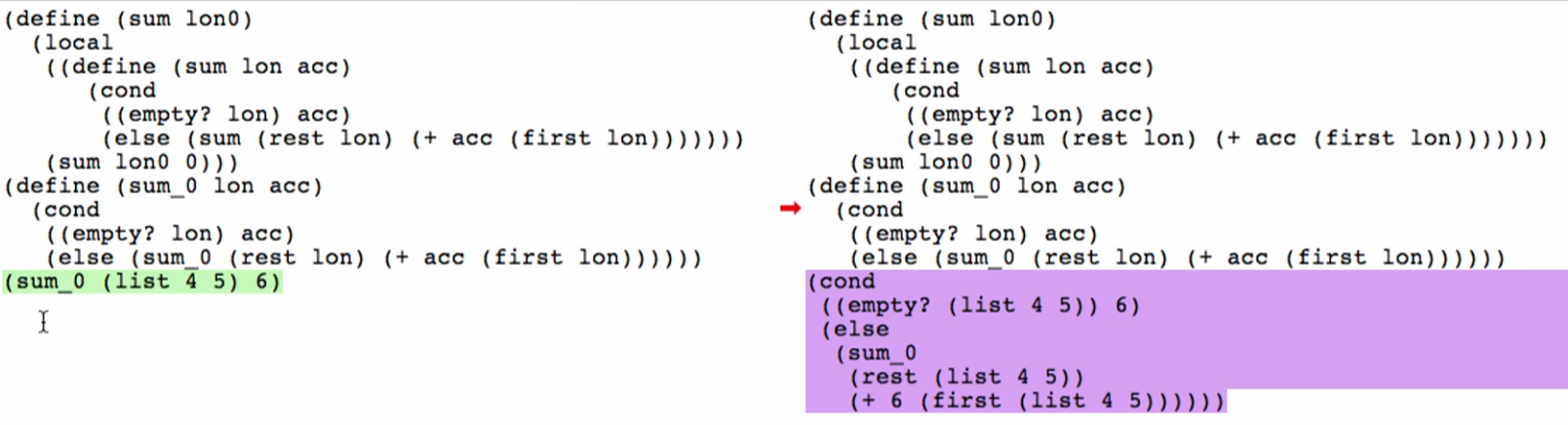
1st call



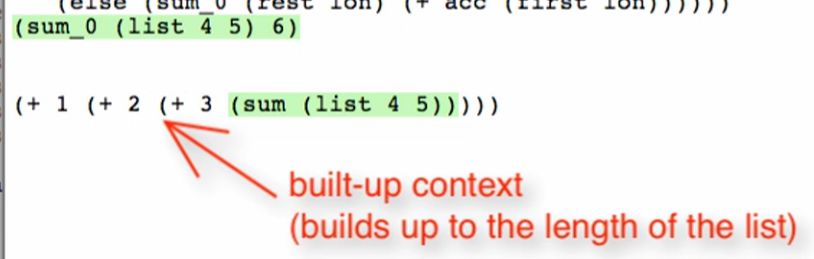
2nd call (1st recursive call)







This is not happening anymore:



Nice! 😊

Overview:

1. Template according to the accumulator recipe
   1. Structural recursion
   2. Wrap in outer function, local, and trampoline
   3. Add accumulator parameter
2. Delete part of the template wrapping around the recursive call (make the recursive call tail position of the else case)
3. Computation of the deleted part will be in the **accumulator argument** position of the recursive call (now in the tail position)

Sample Questions

