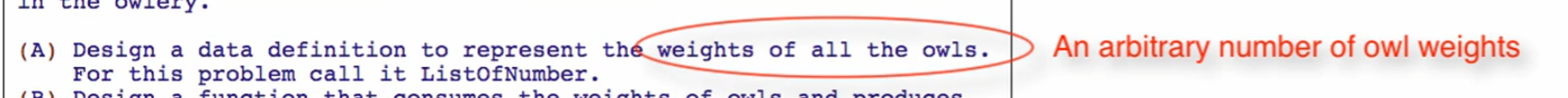
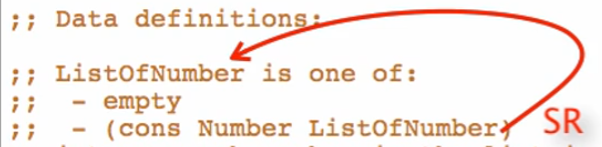
Data definitions:



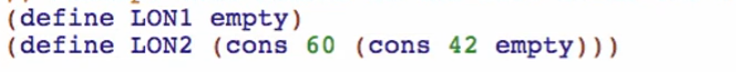
1. Type comment



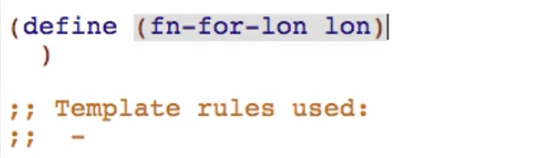
1. Interpretation



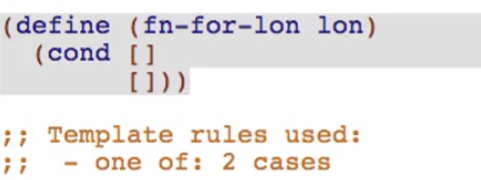
1. Examples
   1. At least one base case
   2. At least one example of self-reference case



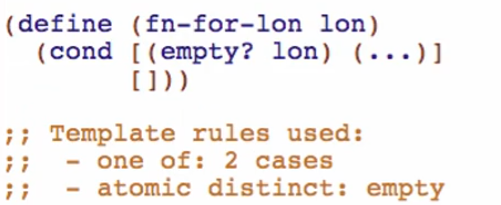
1. Template



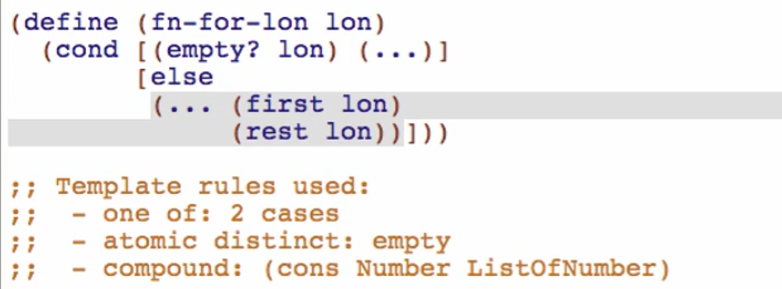
One of: 2 cases



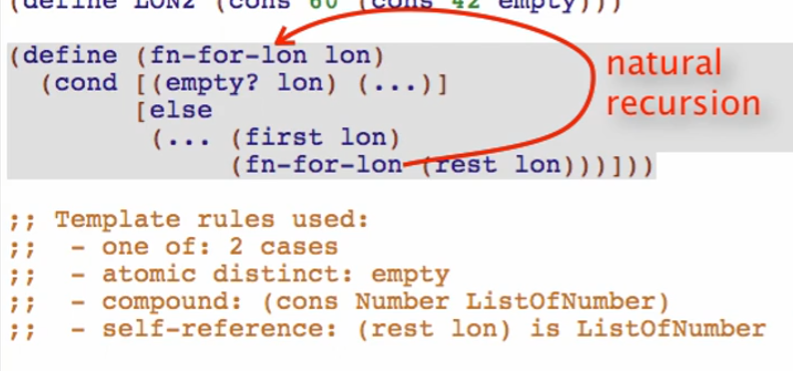
Atomic distinct empty



Compound: (cons Number ListOfNumber)



Self-reference: (rest lon) is ListOfNumber

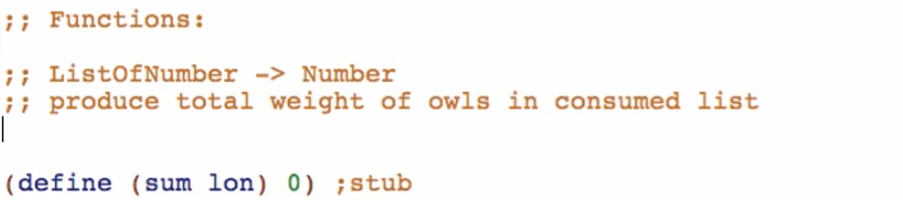


Run to check if well-formed, then comment out template

Function Definition:



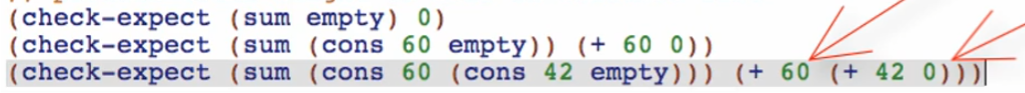
1. Signature, purpose, and stub



1. Examples

* Base case example first



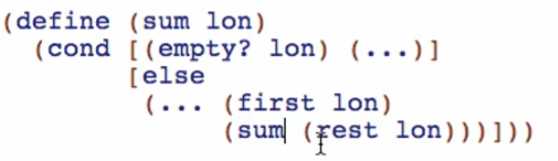


* We write it like this so we can see in detail how the sum function does in the background

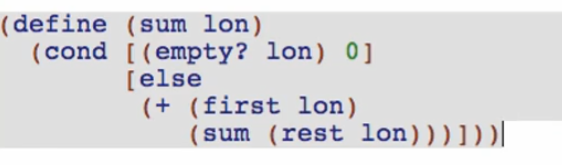
Run to see if well-formed

1. Template

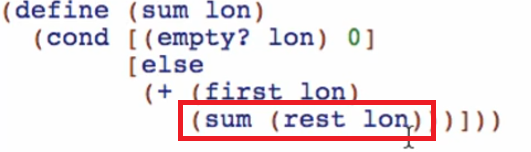
Copy template from data definition, rename the functions and all natural recursions



1. Code body

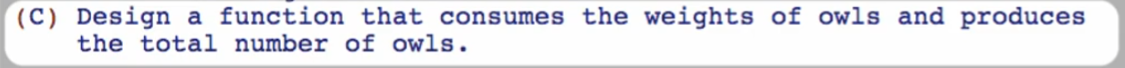


Trust the natural recursion:

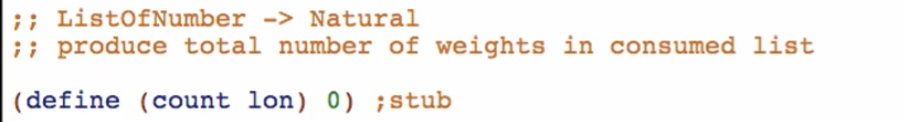


1. Test and debug

Function Definition:



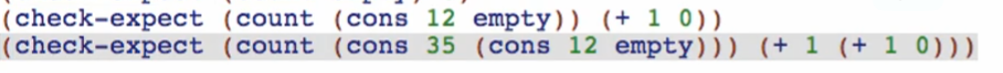
1. Signature, purpose, and stub



1. Examples

* Base case example first



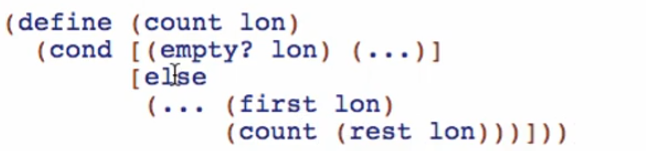


* We write it like this so we can see in detail how the sum function does in the background
* Testing for at least 2 elements long list will catch mistakes in not using recursions properly

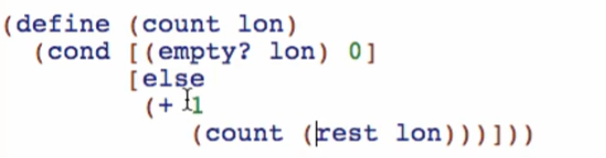
Run to see if well-formed

1. Template

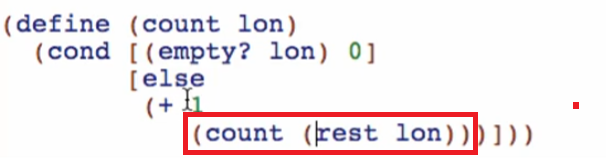
Copy template from data definition, rename the functions and all natural recursions



1. Code body



Trust the natural recursion:



1. Test and debug