

**Function Definition**

* Again with the new rule regarding mutual recursion, do ALL definitions right away for every step

**Signature**

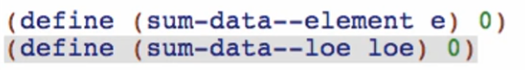


* ??? means we’re not sure of the output of fn-for-loe
* We’ll come back to it later

**Purpose**



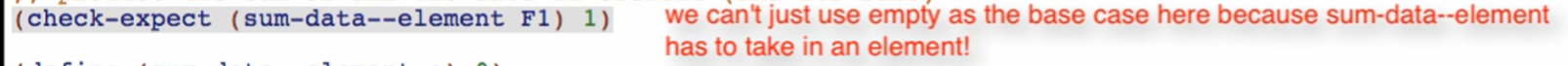
**Stub**



* --element & --loe is added so we can distinguish the 2 functions

**Examples**

Base case for **sum-data--element**

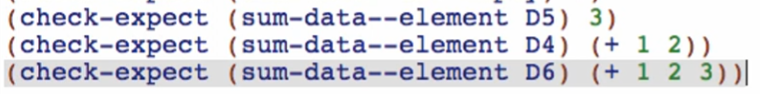


Base case for **sum-data--loe**

****

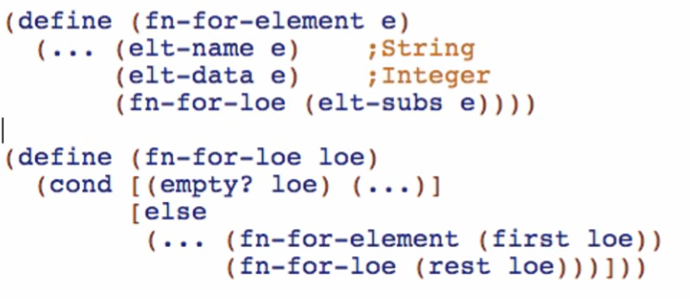
Other cases:

**For sum-data--element:**

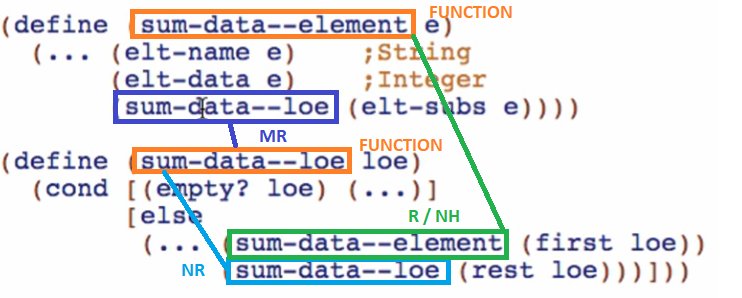


**Templates**

Copy both from data definition



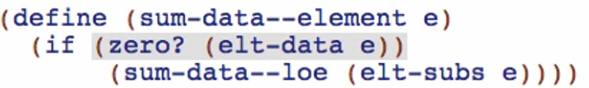
Rename functions, mutual recursions, and natural recursions



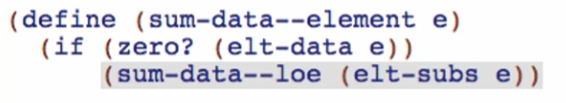
**Code body**

sum-data--element:

Start with condition:

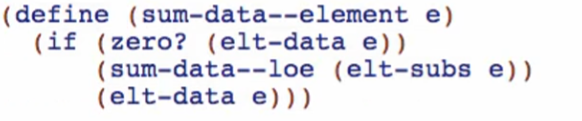


True answer:



* Data = 0 means that it is a directory
  + We need to have recursive function that sum the data of its sub-elements since it is a directory

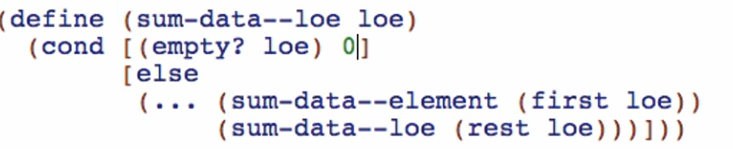
False answer:



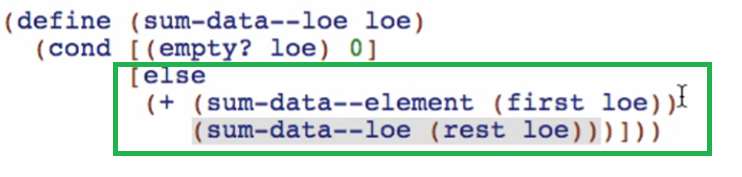
* Data != 0 means that it is a file already
  + We need to produce the size of the file itself

sum-data--loe:

Empty/base case:



Other case:



* Just the sum of the first element of the list, and the sum of the rest of the elements of the list
  + TRUST THE NATURAL RECURSION!
  + TRUST THE REFERENCES/NATURAL HELPERS!
  + TRUST THE NATURAL MUTUAL RECURSIONS!

We are now sure that the return type of sum-data—loe is Integer! 