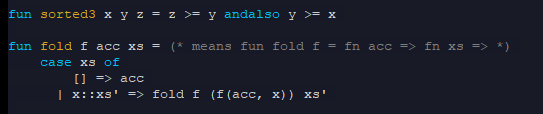


Our previous code:



If a curried function is applied to “too few” arguments, that returns, which is often useful:



This returns: fn z = z >= 0 andalso 0 >= 0



Another example:

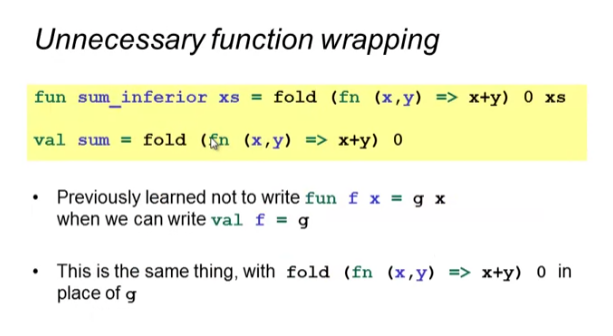


This returns: fn xs => case xs of [] => acc | x::xs’ => fold f (f(acc, x)) xs’



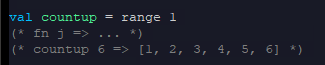
Inferior versions





Other examples:





* Countup is waiting for the 2nd argument
* The first argument is already defined!

Inferior version

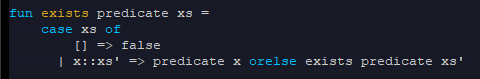


Common style:

* Curry higher-order functions with function arguments
* First to enable convenient partial application

Example:

Higher order function:



Full application: (no waiting args)



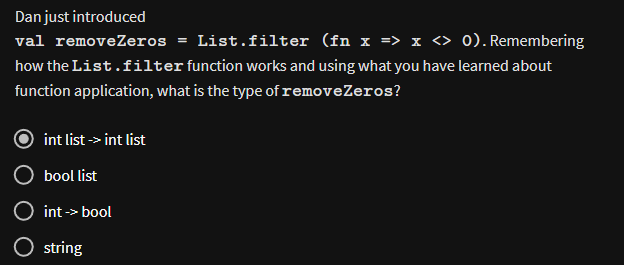
Partial applications: (waiting for the list)

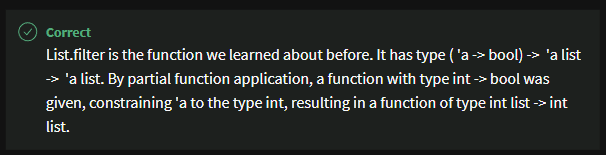


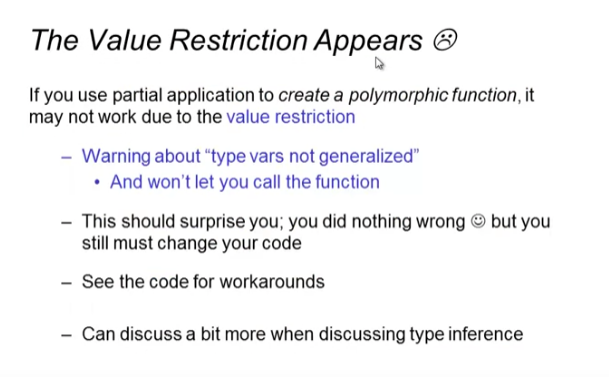
Library functions are also curried in ML

Partial applications: (waiting for the list)









Example of value restriction ONLY in **polymorphic** functions:



Workarounds:

* Give up partial application, use function wrapping



* Putting in an explicit type



This works because the result is not polymorphic:

