Functional Programming - Lab Class Exercises 3

Frank C Langbein
frank@langbein.org

Version 1.4.0

1. productUntilOne

- Write a function `productUntilOne :: Int -> IO Int` that reads integers from IO until 0 is entered and then returns a command that prints the product of the integers.
- Hint: `read s :: Integer` converts the value of `s` to an `Integer`

2. whileIO

- Generalise the function from 1. to a function `whileIO :: Read a => (a -> a -> a) -> (a -> Bool) -> a -> IO a` that takes as argument an operator `a->a->a` (product in 1.), a termination predicate `a->Bool` (stop when entering 1 in 1.), and a termination value as argument. It combines the entered values with the operator, until the termination predicate is `True` and uses the termination value as default value.
- The function from 1. is then `whileIO (*) (\x ->x == 1) 1`
- Note `Read a` means that for type a there exists a `read` implementation to convert a string to type a. That means `read s` where `s` is a string of a value of type a will return the value of `s`.