```haskell
-- Model solution for Lab04
--
-- Copyright [2000..2004] Manuel M T Chakravarty

module Lab04
where

-- Enumerate a list of numbers in a given range with given step
--
-- Example: enumFromToBy 5 50 7 = [5, 12, 19, 26, 33, 40, 47]
--
enumFromToBy                    :: Int -> Int -> Int -> [Int]
enumFromToBy m n k | m > n       = []
                   | otherwise   = m : enumFromToBy (m + k) n k

-- Remove all empty strings from a list of strings
--
-- Example: removeEmpty ["", "Hello", "", "", "World!"] = ["Hello", "World!"]
--
removeEmpty             :: [String] -> [String]
removeEmpty []           = []
removeEmpty ([] :strs)   = removeEmpty strs
removeEmpty (str:strs)   = str : removeEmpty strs

-- Fibonacci Numbers
--
-- Examples: fibonacci 4  = 2
--           fibonacci 20 = 4181
--
fibonacci   :: Int -> Int
fibonacci 1  = 0
fibonacci 2  = 1
fibonacci n  = fibonacci (n - 2) + fibonacci (n - 1)

-- Count the number of 'True' values in a given list
--
-- Example: countTrue [False, True, True, False, True] = 3
--
countTrue                    :: [Bool] -> Int
countTrue []                  = 0
countTrue (b:bs) | b          = 1 + countTrue bs
                 | otherwise  = countTrue bs

-- Make numerals positive (optional)
--
-- Examples: makePositive [-1, 0, 5, -10, -20] = [1, 0, 5, 10, 20]
--
makePositive                    :: [Int] -> [Int]
makePositive []                  = []
makePositive (x:xs) | x < 0      = -x : makePositive xs
                    | otherwise  = x : makePositive xs
```