Function Haskell Exercises

Young W. Lim

2018-09-18 Tue

Outline

1. Based on

2. Function
   - Using FCT.hs

Based on

"The Haskell Road to Logic, Maths, and Programming",
K. Doets and J. V. Eijck

Using FCT.hs

```
module FCT

where

import List
```

```
:load FCT
```

Relation Composition

```
GHCi, version 7.10.3: http://www.haskell.org/ghc/  :? for help
Prelude> :load FCT
[1 of 1] Compiling FCT              ( FCT.hs, interpreted )
Ok, modules loaded: FCT.
*FCT> image (*2) [1,2,3]
[2,4,6]
*FCT> injective (*2) [1, 2, 3]
True
*FCT> injective (^2) [1, 2, 3]
True
*FCT> injective (^2) [-1, -2, -3]
True
*FCT> injective (*2) [-1, -2, -3]
True
*FCT> surjective (*2) [1, 2, 3] [0, 1, 2, 3]
False
*FCT> surjective (*2) [1, 2, 3] [1, 2, 3]
False
*FCT> surjective (*2) [1, 2, 3] [2, 4, 6]
True
```

Basic Notations

- $f(x) = x^2 + 1$

```
f x = x^2 + 1
```

- $abs(x) = |x|$

```
absReal x | x >= 0    = x
          | otherwise = -x |
```

- the identity function

```
id :: a -> a
ix x = x
```

From a function to a list

- converting a function to a list

```
list2fct :: Eq a => [(a,b)] -> a -> b
list2fct [] _ = error "function not total"
list2fct ((u,v):uvs) x | x == u    = v
                       | otherwise = list2fct uvs x
```

- examples

```
*Main> fct2list f [1, 2, 3]
[(1,2),(2,5),(3,10)]
*Main> lst = fct2list f [1, 2, 3]
*Main> lst
[(1,2),(2,5),(3,10)]
```

From a list to a function

- converting a list to a function

```
fct2list :: (a -> b) -> [a] -> [(a,b)]
fct2list f xs = [ (x, f x) | x <- xs ]
```

- examples

```
*Main> lst
[(1,2),(2,5),(3,10)]
*Main> list2fct lst 1
2
*Main> list2fct lst 2
5
*Main> list2fct lst 3
10
```