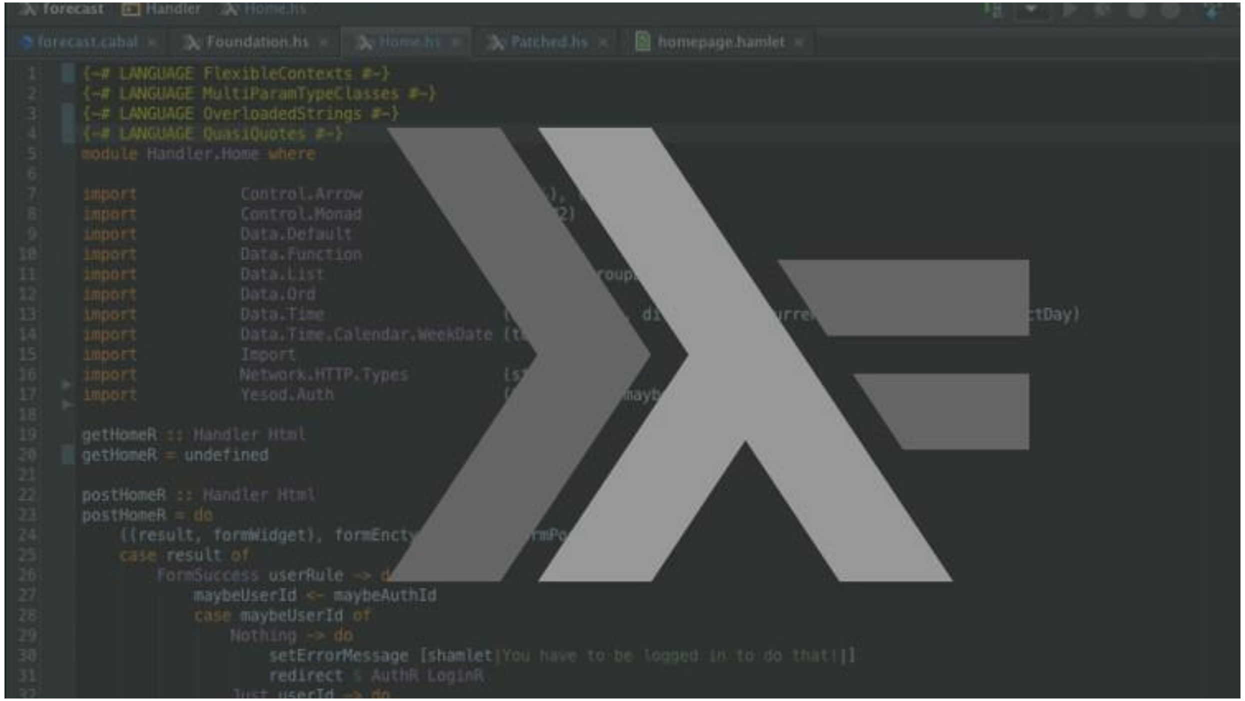
Infinite Lists Problems



```
{-# LANGUAGE FlexibleContexts #-}
{-# LANGUAGE MultiParamTypeClasses #-}
{-# LANGUAGE OverloadedStrings #-}
{-# LANGUAGE QuasiQuotes #-}
module Handler.Home where

import           Control.Arrow
import           Control.Monad
import           Data.Default
import           Data.Function
import           Data.List
import           Data.Ord
import           Data.Time
import           Data.Time.Calendar.WeekDate
import           Import
import           Network.HTTP.Types
import           Yesod.Auth

getHomeR :: Handler Html
getHomeR = undefined

postHomeR :: Handler Html
postHomeR = do
    ((result, formWidget), formEncty
    case result of
        FormSuccess userRule -> d
            maybeUserId <- maybeAuthId
            case maybeUserId of
                Nothing -> do
                    setErrorMessage [shamlet|You have to be logged in to do that!|]
                    redirect $ AuthR LoginR
                Just userId -> do
```

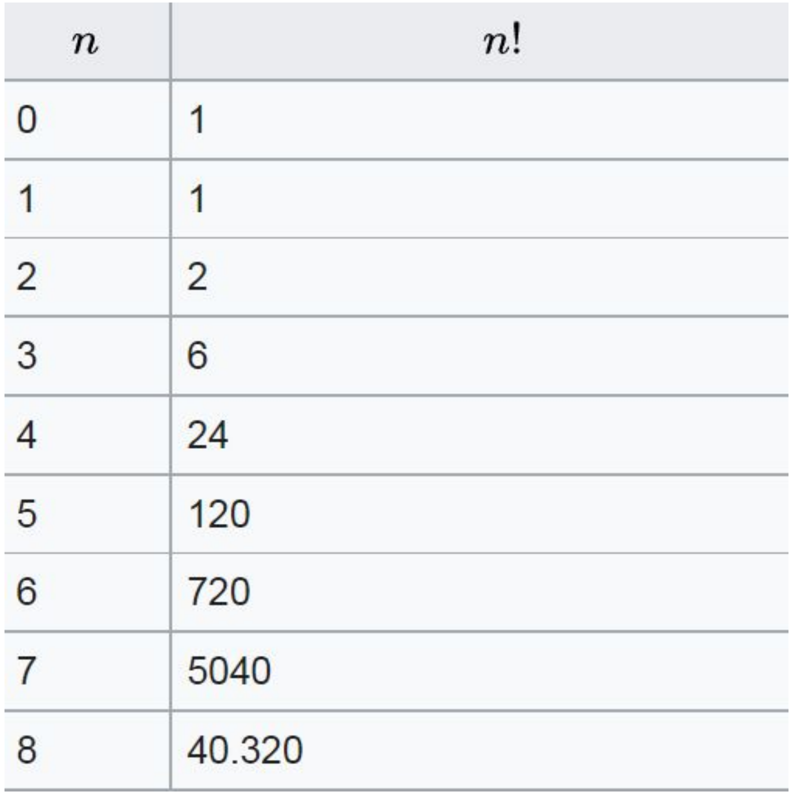
Lucas Bazilio - Udemy

Problem 5

The goal of this problem is to work the definition of infinite lists. In particular, you are required to define the function that generates the sequence of the factorial numbers [1,1,2,6,24,120,720,5040…]. Use the function *factorials :: [Integer]*

Input	Output
`take 6 factorials`	`-> [1,1,2,6,24,120]`
`take 4 factorials`	`-> [1,1,2,6]`

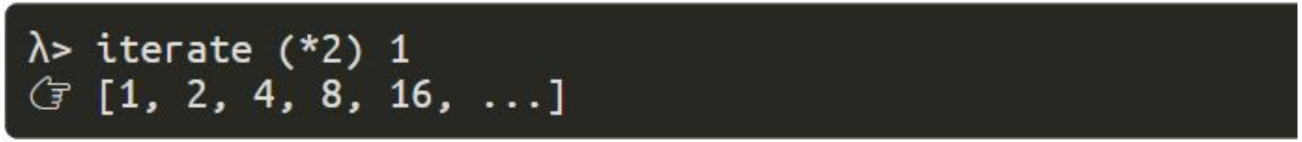
Factorial Numbers



n	$n!$
0	1
1	1
2	2
3	6
4	24
5	120
6	720
7	5040
8	40.320

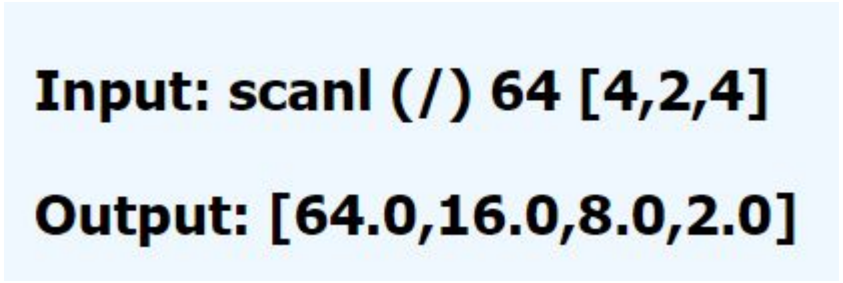
Lucas Bazilio - Udemy

iterate



```
λ> iterate (*2) 1
☞ [1, 2, 4, 8, 16, ...]
```

scanl



```
Input: scanl (/) 64 [4,2,4]

Output: [64.0,16.0,8.0,2.0]
```

Instructor Youtube Channel: Lucas Science

Lucas Science

SUBSCRIBE