1 Problem

A permutation is an ordered arrangement of objects. For example, 3124 is one possible permutation of the digits 1, 2, 3 and 4. If all of the permutations are listed numerically or alphabetically, we call it lexicographic order. The lexicographic permutations of 0, 1 and 2 are:

012 021 102 120 201 210

What is the millionth lexicographic permutation of the digits 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9?

2 Solution

```
import Data.List
import qualified Data.Map as Map
import Data.Maybe
import System.Environment
import Debug.Trace
digitList = [0..9]
```

The *permutation counts* indicate the cycling period for each digit. For example the leading digit changes every 362880 permutations.

```
permutationCounts =
  [362880, 40320, 5040, 720, 120, 24, 6, 2, 1, 1]
getPermutationNumber n = getDigit [] [0..9] (n - 1)
getDigit digs [] n = digs
getDigit digs choices n =
  let ind     = length digs
      thisDig = choices !! ((fromIntegral n) `div` (permutationCounts !! ind))
      digs'   = concat [digs, [thisDig]]
      choices' = filter (/= thisDig) choices
  in getDigit digs' choices' (n - (permutationCounts !! ind) * ((fromIntegral n) `div` (permutationCounts
main = do
  args <- getArgs
  let cmdLineArg = read (args !! 0) :: Int
  let permString = filter (/= ',') $ (reverse . tail . reverse . tail . show) $ getPermutationNumber cmdL
  putStrLn $ "The " ++ show cmdLineArg ++ "th lexicographic permutation of the digits 0-9 i
```

This naive method was not efficient enough.

```
-- lexSortedPerms = sort $ permutations digitList
--
-- badMain = do
```

```
-- args ¡- getArgs
-- let chosenIndex = read (args !! 0) :: Int
-- chosenItem = lexSortedPerms !! chosenIndex
```

3 Result

```
runhaskell problem24.lhs
The 1000000th lexicographic permutation of the digits 0-9 is 2783915460.
```