99 questions/Solutions/25

From HaskellWiki

< 99 questions | Solutions

Generate a random permutation of the elements of a list.

```
rnd_permu :: [a] -> IO [a]
rnd_permu xs = rnd_select xs (length xs)
```

Uses the solution of problem 23 (rnd_select). Choosing N distinct elements from a list of length N will yield a permutation.

Or we can generate the permutation recursively:

```
import System.Random (randomRIO)

rnd_permu :: [a] -> IO [a]
rnd_permu []     = return []
rnd_permu (x:xs) = do
    rand <- randomRIO (0, (length xs))
    rest <- rnd_permu xs
    return $ let (ys,zs) = splitAt rand rest
             in ys++(x:zs)

rnd_permu' [] = return []
rnd_permu' xs = do
    rand <- randomRIO (0, (length xs)-1)
    rest <- let (ys,(_:zs)) = splitAt rand xs
            in rnd_permu' $ ys ++ zs
    return $ (xs!!rand):rest
```

Or we can use the
`permutations`
function from
`Data.List`
:

```
import System.Random (getStdGen, randomRIO)
import Data.List (permutations)

rndElem :: [a] -> IO a
rndElem xs = do
  index <- randomRIO (0, length xs - 1)
  return $ xs !! index

rndPermutation :: [a] -> IO [a]
rndPermutation xs = rndElem . permutations $ xs
```

WARNING: this may choke long lists

Retrieved from "https://wiki.haskell.org/index.php?title=99_questions/Solutions/25&oldid=58053"

Category:

- Programming exercise spoilers
