99 questions/Solutions/24

From HaskellWiki

< 99 questions | Solutions

Lotto: Draw N different random numbers from the set 1..M.

```
import System.Random
diff_select :: Int -> Int -> IO [Int]
diff_select n to = diff_select' n [1..to]

diff_select' 0 _  = return []
diff_select' _ [] = error "too few elements to choose from"
diff_select' n xs = do r <- randomRIO (0,(length xs)-1)
                       let remaining = take r xs ++ drop (r+1) xs
                       rest <- diff_select' (n-1) remaining
                       return ((xs!!r) : rest)
```

The random numbers have to be distinct!

In order to use randomRIO here, we need import module System.Random.

As can be seen, having implemented problem 23, rnd_select, the solution is trivial.

```
diff_select n to = rnd_select [1..to] n
```

Alternative solution:

```
diffSelect :: Int -> Int -> IO [Int]
diffSelect n m = do
  gen <- getStdGen
  return . take n $ randomRs (1, m) gen
```

(Note that this doesn't really solve the problem, since it doesn't generate *distinct* numbers).

Using nub from Data.List:

```
diff_select :: Int -> Int -> StdGen -> [Int]
diff_select n m = take n . nub . randomRs (1, m)
```

Or without giving StdGen as argument and returning IO [Int]:

```
import System.Random
import Data.List
import Control.Applicative

diff_select :: Int -> Int -> IO [Int]
diff_select n m = take n . nub . randomRs (1, m) <$> getStdGen
```

/24&oldid=58051"
Category:

- Programming exercise spoilers
