99 questions/Solutions/49

From HaskellWiki

< 99 questions | Solutions

(**) Gray codes.

An n-bit Gray code is a sequence of n-bit strings constructed according to certain rules. Find out the construction rules and write a predicate with the following specification:

```
% gray(N,C) :- C is the N-bit Gray code
```

Solution:

```
gray :: Int -> [String]
gray 0 = [""]
gray n = let xs = gray (n-1) in map ('0':) xs ++ map ('1':) (reverse xs)
```

A similar solution using list comprehension to build the sub-lists:

```
gray :: Int -> [String]
gray 0 = [""]
gray n = [ '0' : x | x <- prev ] ++ [ '1' : x | x <- reverse prev ]
  where prev = gray (n-1)
```

The Gray code can be recursively defined in the way that for determining the gray code of n we take the Gray code of n-1, prepend a 0 to each word, take the Gray code for n-1 again, reverse it and prepend a 1 to each word. At last we have to append these two lists. For more see the Wikipedia article (http://en.wikipedia.org/wiki/Gray_code) .

Another completely different solution (using folds) that is way more efficient, because it needs just the space which is occupied by the list itself:

```
gray :: Integral a => a -> [String]
gray 0 = [""]
gray n = foldr (\s acc -> ("0" ++ s):("1" ++ s):acc) [] $ gray (n-1)
```

The key is that the algorithm just crawls one time over the input-list and uses the (++) operator in a way that it just has a running time of O(1).

Retrieved from "https://wiki.haskell.org/index.php?title=99_questions/Solutions/49&oldid=57453"
Category:

- Programming exercise spoilers
