99 questions/Solutions/22

From HaskellWiki

< 99 questions | Solutions

Create a list containing all integers within a given range.

```
range x y = [x..y]
```

or

```
range = enumFromTo
```

or

```
range x y = take (y-x+1) $ iterate (+1) x
```

or

```
range start stop
    | start > stop  = reverse (range stop start)
    | start == stop = [stop]
    | start < stop  = start:range (start+1) stop
```

The following does the same but without using a reverse function

```
range :: Int -> Int -> [Int]
range n m
    | n == m = [n]
    | n < m = n:(range (n+1) m)
    | n > m = n:(range (n-1) m)
```

or, a generic and shorter version of the above

```
range :: (Ord a, Enum a) => a -> a -> [a]
range a b | (a == b) = [a]
range a b = a:range ((if a < b then succ else pred) a) b
```

or with scanl

```
range l r = scanl (+) l (replicate (l - r) 1)
```

with support for both directions

```
range l r = scanl op l $ replicate diff 1
  where
  op = if l < r then (+) else (-)
  diff = abs $ l - r
```

Since there's already syntactic sugar for ranges, there's usually no reason to

define a function like 'range' in Haskell. In fact, the syntactic sugar is implemented using the enumFromTo function, which is exactly what 'range' should be.

Category:

- Programming exercise spoilers
