Exercises
Exercises on *foldr*

Exercise 1

Using the higher-order function *foldr*, define a function *sumsq* which takes an integer n as its argument and returns the sum of the squares of the first n integers. That is to say, $sumsqn = 1^2 + 2^2 + 3^2 + ... + n^2$.
(answer given to start you off)

```
sumsq :: Integral a => a -> a
sumsq n = foldr op 0 [1..n]
              where op x y = x*x + y
```

Exercise 2

Define *lengthr*, which returns the number of elements in a list, using *foldr*.

****** Exercise 3

Define *minlist*, which returns the smallest integer in a non-empty list of integers, using *foldr*1 . (*foldr*1 is a Prelude function - look it up yourself or continue and come back to this)

**** Exercise 4

Define *myreverse*, which reverses a list, using *foldr*.

**** Exercise 5

Using *foldr* , define a function remove which takes two strings as its arguments and removes every letter from the second list that occurs in the first list. For example,

```
remove "first" "second" = "econd".
```

Hint: Use a helper function in your lambda

***** Exercise 6

The function *remdups* removes adjacent duplicates from a list. For example,

```
remdups [1, 2, 2, 3, 3, 3, 1, 1] = [1, 2, 3, 1]
```

Define *remdups* using *foldr* (and using a helper method).