Exercises
Higher-order functions

Exercise 1

Show how the list comprehension

```
[f x | x <- xs, p x]
```

can be re-expressed using the higher-order functions ***map*** and ***filter***.

Exercise 2

Without looking at the definitions from the standard prelude, define the following higher-order library functions on lists.

1. Decide if all elements of a list satisfy a predicate:

   ```
   all :: (a -> Bool) -> [Bool] -> Bool
   ```

2. Decide if all elements of a list satisfy a predicate:

   ```
   any :: (a -> Bool) -> [Bool] -> Bool
   ```

3. Select elements from a list while they satisfy a predicate:

   ```
   takeWhile :: (a -> Bool) -> [a] -> Bool
   ```

4. Remove elements from a list while they satisfy a predicate

   ```
   dropWhile :: (a -> Bool) -> [a] -> Bool
   ```

Exercise 3

Redefine the functions

```
map f
```

and

```
filter p
```

using

```
foldr
```