Exercises Higher Order Functions

This exercise sheet is all about higher order functions (functions that take functions as a parameter), the most important ones that you will get to understand are `map, filter, foldr, foldl`.

1 It begins

`map` is often used in conjunction with a function that makes some transformation on an element. Use `map` to solve the following exercises, possibly using a helper function.

- Get a list of square roots, from a list of numbers.
- Get a list of lengths, from a list of strings.
- Add your favorite number to a list of numbers.
- Double the numbers in a list unless they exceed some threshold.
- Given a list, make a list of lists (each element is its own list).
- Using `fizzbuzz` from the second exercise sheet (if you made it), make a list of the first few (e.g. 20) "numbers" in FizzBuzz.

Remember that `filter` keeps the elements that match the predicate, rather than exclude them. Use `filter` to solve the following exercises.

- Get a list of numbers below some threshold, from a list of numbers.
- Come up with your own function that returns a bool and use it with `filter`.
- Recreate quicksort.

To better understand what `foldr` and `foldl` does, try expanding the following expressions by hand, according to the pseudo definitions:

```
foldr f z [x1, x2, ..., xn] == x1 ‘f‘ (x2 ‘f‘ ...  (xn ‘f‘ z)...)
foldl f z [x1, x2, ..., xn] == (...((z ‘f‘ x1) ‘f‘ x2) ‘f‘...)  ‘f‘ xn
```

- `foldr (||) False [False, True, False]`

- `foldl mod 1337 [1166, 86, 43]`

Now try to use `foldr` or `foldl` to solve the following exercises.

- Make your own `sum` function.
- Make your own `maximum` function, you may use `max`. Hint: assume numbers are >= 0.
- Make your own `reverse` function. Hint: use `flip`.