Map

This bad boi:

```
> map :: (a -> b) -> [a] -> [b]
> map f []       =  []
> map f (x:xs)   =  f x : (map f xs)
```

Do you have a cool function that works on single items and that you wanna apply to a whole list? Well MAP is your answer!!!!!!!!! Map is a function that takes a function that works on a single `a` (e.g. a conversion), and a list, then applies the function to every item on that list, putting the results into a new list.

For example, say you had `[1,2,3]` but you REALLY wanted `['a','b','c']` and you had a function that could turn a single `Int` to a `Char` (`toAlpha`), you could `map` that single function onto `[1,2,3]`.

```
map toAlpha [1,2,3] = ['a','b','c']
```

`map` goes through the list recursively applying the function f to each item until it hits the base case of the empty list.