Exercises
Function Application and Function Composition

Exercise 1:
Write **bigCubes** that takes a list and returns a list of cubes that are > 500

```
bigCubes :: [Int] -> [Int]
```

Exercise 2:
Write **lottaBiggest** that takes a list and replicates the largest element 4 times.

```
lottaBiggest :: [Int] -> [Int]
e.g.
lottaBiggest [2,5,3,1] = [5,5,5,5]
```

Hint: You can use the *maximum* function that returns the maximum value in a numeric list.

Exercise 3:
Write **powers** that takes a number and creates a list of that number squared, cubed, and quadrupled.

```
e.g. powers 2 = [4,8,16]
```

Exercise 4:
Given **process** that takes a list of numbers, filters out the negative numbers, doubles the remaining numbers, and returns the sum of the doubled numbers:

```
process x = sum (map double (filter (>0) x))
```

Rewrite **process** using function composition and eliminating the x parameter.

Exercise 5:
Given

```
compute x y = length (take x (filter even y))
```

which takes a number and a list of numbers, filters out the even numbers, and returns the length of the first x even numbers. Rewrite **compute** using function composition and eliminating the y parameter.

Exercise 6:
Assume people are dining. We have a list of tip percents (assume people tip at different rates):

```
e.g. pcts = [0.15, 0.2, 0.21]
```

We have a list of bills (what people owe, minus tip)

```
e.g. amts = [20.5, 30, 25]
```

Write **calcBill** that takes amts and pcts and calculates what each person will pay, based on their amt and pct. Then apply a 4% tax rate.

```
calcBillamtspcts :: [Float] -> [Float] -> Float
calcBillamtspcts [20.5, 30, 25] [0.15, 0.2, 0.21] = [24.518,37.44,31.46]
```