3.5: Functions

A *function* is an operator that takes some input and gives an output based on the input. For example, let's say that have a number and we want to determine its absolute value. R has a function called `abs()` that takes in a number and outputs its absolute value:

```
> x <- -3
> abs(x)
[1] 3
```

Most functions take an input like the `abs()` function (which we call an *argument*), but some also have special keywords that can be used to change how the function works. For example, the `rnorm()` function generates random numbers from a normal distribution (which we will learn more about later). Have a look at the help page for this function by typing `help(rnorm)` in the console, which will cause a help page to appear below. The section of the help page for the `rnorm()` function shows the following:

```
rnorm(n, mean = 0, sd = 1)

Arguments

n     number of observations.

mean    vector of means.

sd   vector of standard deviations.
```

You can also obtain some examples of how the function is used by typing `example(rnorm)` in the console.

We can see that the rnorm function has two arguments, *mean* and *sd*, that are shown to be equal to specific values. This means that those values are the *default* settings, so that if you don't do anything, then the function will return random numbers with a mean of 0 and a standard deviation of 1. The other argument, *n*, does not have a default value. Try typing in the function `rnorm()` with no arguments and see what happens — it will return an error telling you that the argument "n" is missing and does not have a default value.

If we wanted to create random numbers with a different mean and standard deviation (say mean == 100 and standard deviation == 15), then we could simply set those values in the function call. Let's say that we would like 5 random numbers from this distribution:

```
> my_random_numbers <- rnorm(5, mean=100, sd=15)
> my_random_numbers
[1] 104 115 101  97 115
```

You will see that I set the variable to the name `my_random_numbers`. In general, it's always good to be as descriptive as possible when creating variables; rather than calling them *x* or *y*, use names that describe the actual contents. This will make it much easier to understand what's going on once things get more complicated.