: Getting fancy widdit

Functions, packages, and all that jazz

What is a function?

A **function** in R is an object containing multiple interrelated statements that are run together in a predefined order every time the function is called. What this means, is that within every function, there are set of instructions to be followed in their respective order to complete a desired task.

For example, let's say we want to find the mean value of a desired set of numbers.

```
(1+2+3+4+5+6) / 6
```

```
[1] 3.5
```

This is an effective method for acquiring the average of a small set of numbers that are not saved in R. But, what if they were saved?

In this example, we create a vector named 'Mean_Example' with the previous six numbers. Rather than adding them manually, we use the **sum** function to automatically add the values. We then set this sum to be divided by six, which is the total number of values.

```
Mean_Example <- c(1,2,3,4,5,6)

sum(Mean_Example)/6
```

```
[1] 3.5
```

Now, let's crank this up a notch. What do we do if we have a large data sheet and want to calculate the mean of a column? First, I created a data set named *'bugs'* with three columns: *spiders, beetles, and wasps*. Then, using the **rnorm** function, I set the number of values per column with a **default mean of 0** and standard deviation of 1.

Once this data set is created, we can test out the **mean** function on one of the columns. Within the mean function, I tell R to take the mean of the spider column FROM the 'bugs' data set we created. The '$' symbol tells R where to look within an existing data set.

```
bugs <- data.frame(
  spiders = rnorm(200),
  beetles = rnorm(250),
  wasps = rnorm(1000)
)

mean(bugs$spiders)
```

```
[1] 0.01940774
```

```
#Remember, our default mean was 0
```

What is a package?

While R has many built in functions (e.g., **mean()**), some of the most useful functions do not come pre-installed. When this is the case, they are provided to us in well made, neatly packed downloadable objects called *packages*. In essence, the creator of the package has nestled a bunch of things to make your prgramming life easiser into a little folder you can download, and use, at your leisure. An R package can bundle together useful function, help files, and data sets. Typically, a package will have a list of functions all related to the same task or set of tasks.

Let's take a look at the **ggplot2** package. The *purpose* of this package is on the **grammar of grpahics**; the idea that you can build every graph from the same components: a **data** set, a **coordinate system**, and **geoms**-visual marks that represent data points. Functions, such as **ggplot**, that reside within this package are all designed for the ease of figure development.

Let's take a look at how to download a package, starting with **ggplot2**.

To download a package, we must use the **install.packages** function, and place the desired package name in *"quotation marks"* within the function parenthesis. Once downloaded, we must then 'call' the function into our system. Using the **library** function, we tell R to load this package into our current project. We only need to install the package once, but we must 'call' it in every time we restart Rstudio. (To run this code, you must remove the # symbol from install.packages())

```r
#install.packages("ggplot2")

library(ggplot2)
```

Packages with data

Now that we can investigated installing and downloading a package for the use of functions, we will now explore available data sets on R. There are many available data sets within R that we can download and practice programming, but for this tutorial we will work with **Palmer Penguins**. It is wise to download this now, because we will revisit this data set in future sections.

```r
#install.packages(palmerpenguins)

library(palmerpenguins)
```