Daily Primer: Session 6 Wrap-Up
(doc in progress)

At a glance:
Lecture: Statistics, Functions, Knitr
Lab: Statistics Lab
Applications: RStudio (or RStudio Cloud), Zoom, web browser (Chrome, Safari, Firefox)
R Packages: jhur, tidyverse, lubridate, scales, broom, tidylog, RColorBrewer
Datasets: mortality, cars, and your own!

Course Learning Objectives:
1. Read multiple data formats into R
2. Recode and manipulate data
3. Write custom R functions and use add-on packages
4. Make exploratory and customizable plots
5. Perform, interpret, and visualize basic statistical tests
6. Use basic "programming" syntax

Feedback:
1. Daily session surveys (one for each of the five sessions)
2. JHSPH course evaluation

Additional Materials:
1. Cheatsheets: data wrangling, ggplot2, lubridate
   a. Note, I’ve placed a bunch of cheat sheets in the CoursePlus Online Library under Data Science Resources > Cheatsheets.
2. Looking for more information?
   a. Data visualization
      ■ Data visualization with ggplot2
      ■ Deep dive on data visualization
      ■ R Graph Gallery: resources on charts/colors plus a cool tool to find the best way to visualize all kinds of data
      ■ Colors cheatsheet
Quick Reference Guide

**Common data checking functions:**

- `is.na()` is TRUE if the data is FALSE otherwise
- `!` means negation (NOT)
  - if `is.na(x)` is TRUE, then `!is.na(x)` is FALSE
- `all()` takes in a logical and will be TRUE if ALL are TRUE
  - `all(!is.na(x))` are all values of x are NOT NA
- `any()` will be TRUE if ANY are true
  - `any(is.na(x))` asks if we have any NA's in x
- `complete.cases()` - returns TRUE if EVERY value of a row is NOT NA
  - very stringent condition
  - FALSE even if missing one value (even if not important)
  - `tidyr::drop_na()` will drop rows with any missing values
- `unique()` gives you the unique values of a variable
- `table(x)` will give a one-way table of x
  - `table(x, useNA = "ifany")` will have row NA
- `table(x, y)` will give a cross-tab of x and y
- `df %>% count(x, y)` is like `df %>% group_by(x, y) %>% tally`

**Troubleshooting:**

1. When I try to load a package with `library()`, I get an error saying there is no package with that name.
   a. Have you installed the package? You must install a package before you can load it.
      i. You can see what packages are installed with the function `installed.packages()`
      ii. If you need to install the package:
         1. `install.packages("package_name")` for packages on CRAN
         2. `remotes::install_github("package_name")` for packages on GitHub
         3. `BiocManager::install("package_name")` for packages on Bioconductor

2. I loaded the tidyverse package with `library(tidyverse)`, but I’m getting errors when trying to use the function `read_excel()`.
   a. Have you also loaded the `readxl` package using `library(readxl)`?
      i. While `readxl` is technically part of the tidyverse suite of packages, it is not loaded when you load `tidyverse` (likely due to inconvenient function naming schemes). Usually, you can just load the `tidyverse` and use any of the packages contained within it, but `readxl` is an exception to the rule.

3. The dataset is not found when I run the script and/or knit the RMarkdown document.
   a. Did you import the dataset into RStudio? If you use a download link to obtain a dataset, you will need to import the data into RStudio before you can work with it.
   b. Have you set and checked your working directory? Is the path to the dataset location correct?

4. When I try to perform the function, I get an error that says...
   a. the argument is not ______ class and then something about NAs.
   b. the object cannot be coerced to type ‘double.’
      i. These errors mean that your data are not of the correct class for the function you are trying to use. Perhaps you are trying to perform a mathematical operation on a character vector, for example. Check the function requirements using `?function_name` or `??function_name` or help(`function_name`).