Homework 1

This goal of this homework is as follows:

- Give you an opportunity to practice some R programming basics
- Make sure you are comfortable creating and knitting an R Markdown document

Grading

- Each question is weighted evenly

Setup

Make sure you have access to a computer with R and RStudio installed (with the ability to install new packages). If you do not, let me know as soon as possible!

If you have not done so already, install R and RStudio (step-by-step guides provided in week 1 course content).

I strongly encourage you to stay organized. I recommend that you directory on your computer where you can save all of your work for this class (e.g., "cis635" would make for a good name). Within that directory, I recommend keeping each of your homework assignments and projects separated into their own directories. For example, I might organize things as follows:

```
cis635/
  homeworks/
    hw01
    hw02
    ...
  project/
    ...
```

Create a new R Markdown file with the title "Homework 1" and with you as the author (hint: this information will go into your R Markdown frontmatter at the top of the file). In your R markdown file, create a section heading for each of the following parts of your homework:

- Part A. Use R as a calculator
- Part B. Built-in Functions
- Part C. Creating vectors
- Part D. Subsetting vectors
- Part E. Types of data

Part A. Use R as a calculator

Under your Part A heading, write one code chunk for each of the following calculations:

1. $$1 + 2(3 + 4)$$
2. $$log_2(4^3 + 3^{2+1})$$
3. $$\sqrt{(4 + 3)(2 + 1)}$$

For example, if the calculation is `2 + 2`, your code chunk should look something like this when your R Markdown document is compiled:

```
2+2
```

```
## [1] 4
```

I want to see each equation *translated directly into code.* Do not simplify any of the calculations (e.g., `2+1` should be `2+1` in your code, not `3`).

Part B. Built-in Functions

A built-in function is one that comes pre-loaded in R (you don't need to install and load a package to use). To learn how to use a built-in loaded function that you don't know how to use appropriately, use the `help()` function. `help()` takes one parameter, the name of the function that you want information about (e.g., `help(abs)`). Instead of the help function, you could also use enter `?` and then the name of the function in your R console (e.g., `?abs`).

Familiarize yourself with the built-in functions `abs()`, `round()`, `sqrt()`, `tolower()`, and `toupper()`.

Under your Part B heading, use these built-in functions to write code that prints the following items (put each of these into a different code chunk in your R Markdown document):

1. The absolute value of -15.5.
2. 4.483847 rounded to one decimal place. The function `round()` takes two arguments, the number to be rounded and the number of decimal places.
3. "species" in all capital letters.
4. "SPECIES" in all lower case letters.

Part C. Creating vectors

Under your Part C heading, Create the following vectors using just `seq()` and/or `rep()`. (don't use `c()`)

1. Positive integers from 1 to 99
2. Odd integers between 1 and 99
3. The numbers `1,1,1,2,2,2,3,3,3` (hint: read the help pages for `seq` and `rep`!)

Part D. Subsetting vectors

Under your Part D heading, use subsetting syntax (square brackets) to write code that completes the following using the vector `y`.

```
y <- c(3,2,15,-1,22,1,9,17,5)
```

1. Display only the first value of `y`.
2. Display the last value of `y`, in a way that would work if `y` were any length. (hint: `?length`)
3. Display only the values in `y` that are greater than the mean of `y`.

Part E. Types of data

Under your part E heading, using the vector `y` write code that completes the following tasks:

```
y <- c(3,2,15,-1,22,1,9,17,5)
```

1. Make a logical (TRUE/FALSE) vector describing which values in y are positive.

2. Make a logical vector describing whether any of the values of y are equal to the mean of y.

3. Coerce the vector you just made (in #2 above) from a logical vector to a character vector.

4. Make a logical vector describing whether any of the values of y are equal to the median of y.

5. Coerce the vector you just made (in #4 above) into a categorial vector (using `factor()`).

6. Make a matrix with 4 rows and 3 columns that looks like this:

   ```
   ##      [,1] [,2] [,3]
   ## [1,]    1    2    3
   ## [2,]    4    5    6
   ## [3,]    7    8    9
   ## [4,]   10   11   12
   ```

7. Coerce that matrix into a dataframe.