# Homework 3

## Overview

This assignment may be completed with your group. Each group will submit one copy of their homework responses to blackboard.

Unlike previous in your previous homework assignments, in this assignment, you will use R markdown to complete your homework. You will turn in both the .Rmd file and the knitted .pdf file to blackboard. I have provided a template R markdown file for you to use (on blackboard).

### Objectives

This goal of this homework is as follows:

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

### Grading

- Uploaded requested files, 5%
- File is properly/clearly formatted, 5%
  - Proper section headers for each part of your homework.
  - You clearly indicate which question each of your responses are associated with.
- All questions: 90%
  - All questions are weighted evenly.

### Deliverables

- .Rmd file used to knit your .pdf file
- .pdf file knitted from your R markdown file

I strongly recommend that you read over your knitted .pdf file before submitting your homework to make sure that it is formatted as you expect.

## Setup

I strongly encourage you to stay organized for this course. I recommend that you directory on your computer where you can manage all of your work for this class (e.g., “cis331” 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:

```
cis331/
  homeworks/
    hw01
    hw02
    hw03
    ...
```

Download the R markdown template provided to you (on blackboard) and answer all of the questions below. The template is meant to help you organize/format your responses. You will need to add explanatory text and code chunks as necessary (be sure to update your names!).

**Recommended reference material**

You will need to refer to R documentation / R programming reference material in order to answer all of the questions.

- Introduction to dplyr
- Introduction to tidying data with tidyr
- R for Data Science

## Part A. Use R as a calculator

Write one code chunk for each of the calculations below. 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
```

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

1.

$$1 + 2(3 + 4)$$

2.

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

3.

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

## 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()`.

Use these built-in functions to write code that prints the following items:

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

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

Use subsetting syntax (square brackets) to write code that completes the following using the vector `y`. If you're not sure where to start on these questions, I suggest reading through the following Software Carpentry lesson on subsetting data: https://swcarpentry.github.io/r-novice-gapminder/06-data-subsetting/index.html

```
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

Using the vector `y` write code that completes the following tasks. If you're not sure how to coerce one data type into another, I suggest the following Software Carpentry guide: https://lms.gvsu.edu/ultra/courses/_397858_1/cl/outline

```
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 the matrix you made in the previous question into a dataframe.

## Part F. Control flow

Learning to use R (or any data analysis tool) will require you to be comfortable learning by readint documentation / online tutorials. Read through the following Software Carpentry guide on control flow in R: https://swcarpentry.github.io/r-novice-gapminder/07-control-flow/index.html. After reading the guide, you should be familiar with how to use loops and if statements in R. For extra help with looping in R, see Chapter 21 in R for Data Science.

1. Given the vector of 6 WNBA teams and the vector of teams that qualified for the playoffs, write a loop that iterates over each team in `teams` and prints out the team name and whether or not that team qualified for the playoffs. If a team qualified for the playoff, print the team's name and "Qualified". Otherwise, print the team's name and "Not qualified". For example, "Atlanta Dream: Qualified" or "Indiana Fever: Not qualified". **For credit, your solution *must* must a loop and an if-else statement.** Hint: the `%in%` operator may be useful for checking if a team is qualified for the playoffs.

```r
teams <- c(
  "Atlanta Dream",
  "Chicago Sky",
  "Connecticut Sun",
  "Indiana Fever",
  "New York Liberty",
  "Washington Mystics"
)

qualified_playoffs <- c(
  "Las Vegas Aces",
  "Chicago Sky",
  "Connecticut Sun",
  "Seattle Storm",
  "Washington Mystics",
  "Dallas Wings",
  "New York Liberty",
  "Phoenix Mercury"
)
```

## Part G. Tidyverse

1. Select one of the packages in the tidyverse (see: https://www.tidyverse.org/packages/).
2. Find the function reference for that package (i.e., the webpage that lists the functions and links to the documentation for each function).
3. Pick one of the functions you think looks useful and run the example code in that function's documentation (your answer to this question should be working example code).