Week 2 R & Tidy assignment

Add in code chunks and code as necessary and descriptions to the template document below. Add your name and knit to pdf. Submit the PDF to the canvas assignment in the schedule.

Part I: Reinforce concepts of data organization

OVERVIEW

MATERIALS NEEDED

Using HW_Data_File.xlsx, which is in the data folder of this project, do the following (part 1 can be done outside of RStudio):

1. Organize/create a spreadsheet following 12 basic principles. The end result should be a spreadsheet that is 13 rows by 5 columns for future data analysis.
2. Create a readme file (data dictionary) in plain text format that contains relevant information corresponding to the dataset. Include variable names, definitions, units, and original data source.

put your data dictionary here

3. Validate the data using R code
    a. Read the data into R (upload your tidy dataset from 1, see the end of the intro document for `read_xlsx`, `read_csv`, and other `read_` functions, or use "Import Dataset…" from the File pane).
    b. Check whether all "FFAs reduction %" values range between 0 and 100. If not, flag the variable.
    c. Check whether all "cdw" values are numbers.
    d. Follow the 12 basic principles for consistency, numbers, validation, names.

Part II: R & Tidy

OVERVIEW

Below, you'll find the 7 exercises that are this weeks homework. You'll want to modify the top of this document to include your name, date, title reflecting that this homework is for week 2, etc.

Once you complete each exercise, you'll knit the document, save as a pdf, and upload the document. Ensure that the code and output show up. I'll post an example for the first exercise.

Exercise 1

Perform the following calculations. Ensure that the results are shown. Here, the point is to understand how the programming language interprets order of operations.

Calculations:

1. 100 * 4 (Note the *, allows the asterix to be shown in markdown as opposed to making italics. This is called an escape character).
2. 5 - 7
3. 9 - 42
4. (9-4)2
5. 800/2^2

Exercise 2

Engineering results frequently require unit conversions to share information appropriately with an audience. In the course Introduction to BSE, you learned about mass balances. In our field, we frequently measure the volume of a substance (e.g., water, chemical) through a place (e.g. pipe, river) over a given time. We call this a volumetric flux, with units of L^3/T (L = length, T = time).

Imagine you are on an international consulting team that is working on the design of a large food processing facility. While here in the US, we often use english units for length (e.g. feet) whereas most of the rest of the world uses the SI system (e.g., meters).

Create an R-script in the code chunk below that converts a flowrate of ft^3/second to m^3/s. Include the following: - header - flowrate variable in ft^3/second that is equal to 1,500 - an equation that creates a new flowrate variable in m^3/s - the printed value of the new variable in m^3/s

After you create the script, run the script. In the grey box below on the upper right corner, the green arrow will run the contents within the grey box (in R-markdown, the grey box = chunk)

Exercise 3

Here are some example functions:

- `abs()` returns the absolute value of a number (e.g., `abs(-2)`)
- `round()`, rounds a number (the first argument) to a given number of decimal places (the second argument) (e.g., `round(12.1123, 2)`)
- `sqrt()`, takes the square root of a number (e.g., `sqrt(4)`)
- `tolower()`, makes a string all lower case (e.g., `tolower("HELLO")`)
- `toupper()`, makes a string all upper case (e.g., `toupper("hello")`)

Use these built-in functions to print the following items:

1. The absolute value of -15.5.
2. 4.483847 rounded to one decimal place.
3. 3.8 rounded to the nearest integer. You don't have to specify the number of decimal places in this case if you don't want to, because round() will default to using 0 if the second argument is not provided. Look at help(round) or ?round to see how this is indicated. 1
4. "species" in all capital letters.
5. "SPECIES" in all lower case letters.
6. Assign the value of the square root of 2.6 to a variable. Then round the variable you've created to 2 decimal places and assign it to another variable. Print out the rounded value.
7. Do the same thing as task 6 (immediately above), but instead of creating the intermediate variable, perform both the square root and the round on a single line by putting the sqrt() call inside the round() call.
8. Assign a variable x to the value of 10.5. Look up both the ceiling and floor functions within R-base package. Using the appropriate function, assign a new variable y to 10 using only the appropriate function and x. In the output, show the value of y.

Exercise 4

Create the following vector object `numbers`. Then use code to print the requested values related to the vector. You'll need to use na.rm = TRUE to ignore the null values.

```
numbers <- c(7, 6, 22, 5, NA, 42)
```

1. The smallest number in the numbers vector

```
# put your answer here
```

2. The largest number in the numbers vector
3. The average of the numbers in the numbers
4. The sum of the values in the numbers vector

Exercise 5

Create the following vector, `numbers` and then use code to print the requested values related to the vector.

```
numbers <- c(5, 2, 26, 8, 16)
```

1. The number of items in the numbers vector (using the `length` function)
2. The third item in the numbers vector (using `[]`)
3. The smallest number in the numbers vector (using the `min` function)
4. The largest number in the numbers vector (using the `max` function)
5. The average of the numbers in the numbers vector (using the `mean` function)
6. The first, second and third numbers in the numbers vector (using `[]`)
7. The sum of the values in the numbers vector (using the `sum` function)

Exercise 6

You have data on the length, width, and height of 10 individuals of the yew Taxus baccata stored in the following vectors:

```
length <- c(2.2, 2.1, 2.7, 3.0, 3.1, 2.5, 1.9, 1.1, 3.5, 2.9)
width <- c(1.3, 2.2, 1.5, 4.5, 3.1, NA, 1.8, 0.5, 2.0, 2.7)
height <- c(9.6, 7.6, 2.2, 1.5, 4.0, 3.0, 4.5, 2.3, 7.5, 3.2)
```

Create the above vector objects and then determine the following:

1. The volume of each shrub (length $\times$ width $\times$ height). Storing this in a variable will make some of the next problems easier.
2. The sum of the volume of all of the shrubs (using the `sum` function).
3. A vector of the height of shrubs with lengths > 2.5.
4. A vector of the height of shrubs with heights > 5.
5. A vector of the heights of the first 5 shrubs (using `[]`).
6. A vector of the volumes of the first 3 shrubs (using `[]`).
7. A vector of the volumes of the last 5 shrubs with the code written so that it will return the last 5 values regardless of the length of the vector (i.e., it will give the last 5 values if their are 10, 20, or 50 individuals).

Exercise 7

Create the following variables using the described approach:

1. A vector named x from 1 to 10 with a step size of 1
2. A vector names x from 10 to 1 with a step size of 1
3. A vector named x from 1 to 0 with a step size of 0.5
4. Create a vector named y using the following command: `y <- 1:5`. Now, using the length command, create a new variable z which contains only the last value in y.
5. Create a new vector z that multiplies each element of y by 10.
6. Create a new vector z that contains only the last 2 values in the y-vector.