# STAT 412/612 HW 2: Pipes, Functions, Logicals

Richard Ressler

2020-09-02

## Instructions

- After completing the questions, upload both the .RMD and PDF files to Canvas.
- Learning Outcomes:
- Create functions
- Create and source scripts
- Employ piping
- Employ logicals
- Employ conditionals.

## Grading Rubric

| Question.Part: | Points | Topic |
|---|---|---|
| 1.a | 2.00 | Correct logic Flow in Text Chunk |
| 1.b | 3.00 | Working code with results for 3, 5, 15, 16 |
| 1.c | 1.00 | Working function with results for 3, 5, 15, 2 |
| 1.e | 1.00 | Error checking for single value and numeric |
| 1.f | 2.00 | Roxygen Documentation: title, description, usage, arguments and return |
| 1.g1 | 1.00 | Proper .R file |
| 1.g2 | 2.00 | Proper Sourcing and Results in Rmd File. |
| 2.1 | 2.00 | Working function for new cut |
| 2.b | 1.00 | Adjusted for < |
| 2.c | .50 | Cite proper advantage |
| 3.1 | 1.50 | Correct sequence with pipe and results |
| 4.1 | 1.00 | Correct functions and proportion |
| 5.1 | 2.00 | Correct Results |
| Total | 20 | |

# 1 Implement a `fizzbuzz()` function

1. Create a new function which takes a single number as input. Use logicals and conditionals as well as the operator `%>%` as required.

- Requirements
  - If the input number is divisible by three, return `"fizz"`.
  - If it's divisible by five, return `"buzz"`.
  - If it's divisible by three *and* five, return `"fizzbuzz"`.
  - Otherwise, return the input number.

a. Design your function. Write out in words in a text chunk the steps your function will need to accomplish in order to convert the input number into the required text or numeric output.

1.

2.

3.

4. etc.*As many steps as you need*

b. Write R code in a code chunk to implement your steps using a variable `x`. Test it with x having values of of 3, 5, 15, and 16.

c. Once the code is working, then copy it to a new code chunk and turn it into function with input argument of `x`.

d. **Show your output for the following inputs**: `3`, `5`, `15`, `2`.

e. Update your function to include error checking

- Ensure the input is both numeric and a single value - not a vector.
- Test it on `cat`, and `c(1,5)`.
- Remember, in the code chunk where you run the function, **set your code chunk parameter for error to be TRUE**,
  - e.g. `{r, error=TRUE}`, so it will knit with the error.

f. Complete your function by inserting and completing Roxygen comments in the code chunk, above the function, to document the function.

- Include the following elements: title, description, usage or syntax, arguments ( the params), and return value.

g. Create a script out of your `fizzbuzz()` function

- Copy and paste the code from your working function into a new .R file and save in the R directory with the file name fizzbuzz_s.R
- Rename the function to fizzbuzz_s
- Use the following code in a code chunk to show your code
  - `cat(readr::read_file("./R/fizzbuzz_s.R"))`
  - Adjust the relative path as necessary
- Write code in a new code chunk in your original homework file to source the `fizzbuzz_s()` function
- Run the function in your homework .Rmd file to show the results with the values 35, 18, 45, and -1

## 2 Create a new `cut()` function

1. Write a function that uses the function `cut()` to simplify this set of nested if-else statements?

- Consider using `-Inf` and `Inf`.
- Note, this will also output the levels of the factors.

   a. Show the output for inputs: `31`, `30`, `10`, `-10`.

```
if (temp <= 0) {
  "freezing"
} else if (temp <= 10) {
  "cold"
} else if (temp <= 20) {
  "cool"
} else if (temp <= 30) {
  "warm"
} else {
  "hot"
}
```

   b. Look at help for `cut()`. Change the call to `cut()` to handle `<` instead of `<=` in the comparisons.

   c. What is the other chief advantage of the `cut()` method for this problem? (Hint: what happens if you have many values in `temp`?)

## 3 Using the Forward Pipe

1. Using the forward pipe `%>%`,

- Sample from the vector `1:10` 1000 times **with replacement**,
- Calculate the resulting sampled vector's mean, then
- Exponentiate that mean.

## 4 Calculate a proportion

- Select a random sample of 100 normally distributed values with mean 10 and variance of 3.
- Calculate the proportion greater than 12.

## 5 Logical Comparisons and Subsetting

- Create the values:
  - x <- c(TRUE, FALSE, TRUE, TRUE)
  - y <- c(FALSE, FALSE, TRUE, FALSE)
  - z <- NA
- What are the results of the following:
  - x & y
  - x & z
  - !(x | y)
  - x | y
  - y | z
  - x[y]
  - y[x]
  - x[x|y]