HOMEWORK on FUNCTIONS

p. 273 ( 19.2.1) # 2 & 5:

2. (2 pts) In the second variant of rescale01(), infinite values are left unchanged. Rewrite rescale01() so that -Inf is mapped to 0, and Inf is mapped to 1. Add error checking to return the message “Error: input must be numeric” if the input is not all numeric.

5. (1 pt) Write both\_na(), a function that takes two vectors of the same length and returns the number of positions that have an NA in both vectors.

p. 276 (19.3.1) #1:

1. (1.5 pts) Read the source code for each of the following three functions, puzzle out what they do, and then brainstorm better names.

f1 <- function(string, prefix) {

**substr**(string, 1, **nchar**(prefix)) == prefix

}

f2 <- function(x) {

if (**length**(x) <= 1) **return**(NULL)

x[-**length**(x)]

}

f3 <- function(x, y) {

**rep**(y, length.out = **length**(x))

}

p. 279-280 (19.4.4) # 2, 3, & 4

1. (1.5 pts) Write a greeting function that says “good morning”, “good afternoon”, or “good evening”, depending on the time of day. (Hint: use a time argument that defaults to lubridate::now(). That will make it easier to test your function.)
2. (2 pts) Implement a fizzbuzz function. It takes a single number as input. If the number is divisible by three, it returns “fizz”. If it’s divisible by five it returns “buzz”. If it’s divisible by three and five, it returns “fizzbuzz”. Otherwise, it returns the number. Make sure you first write working code before you create the function.
3. (2 pts) How could you use cut() to simplify this set of nested if-else statements?

if (temp <= 0) {

"freezing"

} else if (temp <= 10) {

"cold"

} else if (temp <= 20) {

"cool"

} else if (temp <= 30) {

"warm"

} else {

"hot"

}

How would you change the call to cut() if I’d used < instead of <=? What is the other chief advantage of cut() for this problem? (Hint: what happens if you have many values in temp?)