1. Errors and where they come from

   (a)
   ```
   mtcars %>%
     summarize(mean(mpg))
   ## Error in mtcars %>% summarize(mean(mpg)):  could not find function "%>%"
   ```

   Problem: haven't imported package/library
   Solution: run `library(package)` to load the missing package. (In this case, `library(dplyr)`)

   (b)
   ```
   library(naniar)
   ## Error in library(naniar):  there is no package called 'naniar'
   ```

   Problem: haven't installed package/library
   Solution: install package using `install.packages()` (In this case, `install.packages("naniar")`)

   (c) argument missing

   ```
   cor(y = mtcars$mpg)

   ## Error in is.data.frame(x):  argument "x" is missing, with no default
   ```

   Problem: argument missing
   Solutions: provide argument or add a default to the function definition

   (d)
   ```
   read_csv("amcnamara/Documents/projects/cooldata.csv")
   ## Error:  'amcnamara/Documents/projects/cooldata.csv' does not exist in current working
   directory ('/Users/amcnamara/STAT360/quizzes').
   ```

   Problem: filepath wrong
   Solution: use `getwd()` or `here()` to debug, make an absolute path that works on your computer or a relative path that works on anyone's computer

2. Environments, search paths, getting the code from functions

   (a) how to get the code from a function?
   Solution: by writing the name of the function and hitting enter
   If the function is a generic, might need a dot (e.g. `plot.ecdf`) or might need three colons (`:::`) for unexported functions (e.g. `stats:::plot.acf`)

   (b) What's first in my search path?
   Solution: `.Globalenv`

   (c) What's second in my search path?
   Solution: the most recently loaded package

   (d) How to specify a particular version of a function?
   Solution: double colon (e.g., to specify the `dplyr` version of `filter`, we use `dplyr::filter`)

3. Accessing parts of objects

   (a) How to access a variable in a dataset
   Solution: use dollar sign,(`df$variablename`, e.g. `mtcars$mpg`)

   (b) How to access a variable in an s4 objects
   Solution: use the at operator, `@`, to get at slots. If I want a variable from the data in the data slot, `df@data$variablename`. E.g. `counties_rgdal@data$Scale`

4. Programming functions– I might ask you to debug a function, or finish it for me. You should know
   (a) what the parts of a function are
      i. body
      ii. formal arguments
   (b) how to use `stop()` function to throw errors
   (c) basic control structures (`for` loops– generate from memory, e.g. `for (i in 1:10){}`)