Exercise 02

Contents

For this homework, you will save the `R Markdown` (e.g., `.Rmd`) homework document and save it to your `homework` project directory on your computer. Save a new copy of this file so that its name has the prefix `2023_Lastname_FirstInitial`. Full name example: `2023_cookg_EX_02.Rmd`. Feel free to work with a partner but understand that the work submitted has to be your own.

This homework is due on Tuesday by 2:30pm.

Enter your name here

Problem 1: Checking your working directory using `{here}`

You should have the `{here}` library installed. The following code block contains a function from the `{here}` library that will return the name of the your working directory. If the function does not run, you will need to install `{here}`. In addition, if the working directory is not put in a place where your homework goes, you will need to either move your `.Rmd` homework file or set up a project directory for homework.

```
here::here()
```

```
## [1] "C:/Users/gcook/Sync/git/dataviz23"
```

Problem 2: Loading libraries

Inside the following code block, use `library()` to load the `{readr}` and `{dplyr}` libraries. You won't use them for this exercise but you will for the next one.

Problem 3: Creating objects

Create a code chunk in R Markdown by either a) typing on a Windows system `CONTROL+ALT+i` or on a Mac `COMMAND+OPTION+i` all at the same time or by 2) clicking the green icon containing a + and a c that appears in RStudio near the file tab for your opened `.Rmd` file. You can also go here to see the icon image.

In that code chunk:

1) assign the character string "hello" to an object named x,
2) assign any numeric object to y

Problem 4: Adding code snippets to RStudio

Code snippets are useful for creating shortcuts for task you repeat. Copy the snippet below and then got to `Tools -> Edit Code Snippets` and paste the snippet. Save.

```
snippet proj_dir
    suppressWarnings(lapply(c("data", "r", "figs", "report", "refs", "notes", "notes/more_notes"), FUN =
```

Problem 5: Running code snippets

For this snippet, you only have to do it once, so the recommendation is to type the snippet hotstring `proj_dir` at the `R` console prompt. When you start typing, RStudio will likely start to auto-populate options and one that will appear is `proj_dir {snippet}`. Select it an run that code in the console.

The snippet is using the `lapply()` function to apply a function to either a `list` or a `vector` object. The `vector` is a character vector containing several elements which will become names of directories (e.g., `"/r"`) or sub-directories (e.g., `"/notes/more notes"`. That `vector` is then passed to `dir.create()`, which will create (iteratively, element by element) a directory (or sub-directory) at the specified path location. That location is the path returned by `here::here()`.

Run the code snippet at the console to create the directories. Must be done for next step to work.

Problem 6: Creating a `.R` Script file

`.R` files are different from `.Rmd` files. They contain only `R` code. In `RStudio`, create a new file that is an `R Script` file. `Save As` `my_functions` to your newly created `.../homework/r` directory (previous problem). The file extension will be `.R` so your file will now be `my_functions.R`. In that file, paste the code below, which is a function I wrote for viewing data frames and tibbles in `html` form. I prefer this function over the built-in `View()` function. Maybe you won't prefer it but this is homework anyway.

```
view_html <- function(object, rows = F, show = 100,...) {

    if (!require(DT)) {

        stop("DT library not installed")

    } else {

        if (tibble::is_tibble(object)) {
            object = as.data.frame(object)
            #message("converted tibble to dataframe for viewing")
        }

        if (is.null(dim(object)) & class(object) == "list") {
            message("Object is a list. Viewer displays last list element. Consider passing each element

            lapply(object, function(x) {
                DT::datatable(x, rownames = rows, options = list(pageLength = show))
            })
        } else {
            DT::datatable(object, rownames = rows, options = list(pageLength = show))
        }

    }
}
```

Close the `my_functions.R` file.

Problem 7: Sourcing `.R` Script files

Often times, you will have code that you do not want to add to your `RMarkdown` file because that file is likely more for reports than for running raw code. You don't want that file getting too busy. Within the `RMarkdown` file, you can use `source()` to call and execute code saved in a different file (e.g., `my_functions.R`)

Sourcing will require you to use `here::here()` because finding the path location will be easy (if it exists). The function in following code block will return a character object of the `...path to project working dir/r/my_functions.R"`

```
here::here("r", "myfunctions.R")
```

```
## [1] "C:/Users/gcook/Sync/git/dataviz23/r/myfunctions.R"
```

You can test whether the full path exists using `file.exists()`.

```
file.exists(here::here("r", "my_functions.R"))
```

```
## [1] TRUE
```

In this final code block, `source()` that file path (hint: you want the path only, you do not want to know whether the path exists). The first argument in `source()` is `file`, so the `file` is that path. Type ?source in the R console to see the help file if you want to read more.

Having sourced the file, the `view_html()` function is now defined and running in R.

Problem 8: Viewing data frames/tibbles

To test out the difference between `View()` and `view_html()`, you will use them on the `mtcars` data frame built into `R`. Make note of the approaches to viewing the data frame. To shorted things up, we will wrap the function with `head()` so that only the first few rows of data appear.

```
head(mtcars)
```

```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
View(head(mtcars))
```

In this code block, pass `mtcars` to `view_html()` to view with that function.

```
source(here::here("r", "my_functions.R"))
view_html(head(mtcars))
```

```
## Loading required package: DT
```

```
## Google Chrome was not found. Try setting the ‘CHROMOTE_CHROME‘ environment variable to the executable
```

mpg cyl disp hp drat wt qsec vs am gear carb

Problem 9: RMarkdown headings

Practice using `RMarkdown` to create headers for sections. Create 3 headers (e.g., level 1, level 2, and level 3). You can find tips here. Once you render you `html` file, take note of what these headers do.

Problem 10: RMarkdown lists

Practice using some `RMarkdown` code by creating: **a)** bulleted list with three items that describe you, **b)** a numbered list with your top 3 band names, and **c)** a sentence that has one word in bold and one word in italics. You can find some tips here.

Rendering RMarkdown files for submission

You will want to produce two rendered documents. You can do this using code or more simply using the toolbar icon. You can find tips for both approaches here.

However, I have discovered that if there are `html` objects present in your `RMarkdown` file, they will not easily render to `pdf` format. A solution is to install a library named `webshot` (so install it) and then install PhantomJS using `webshot::install_phantomjs()` and the console.

In order to render `pdf` files from `RMarkdown`, you will need a `LaTeX` installation. If you use `LaTeX`, you likely already have this set up. If not, when you `knit` from the toolbar and select `knit to pdf`, `R` will throw an error and suggest a simple installation approach which is to execute `tinytex::install_tinytex()`. So, run that code in your `R` console and then `knit` your homework as a `pdf` so that you can see what it looks like and so that you can compare that output to the `html` version.

Please knit an `html` file and a `pdf` file and upload both here.