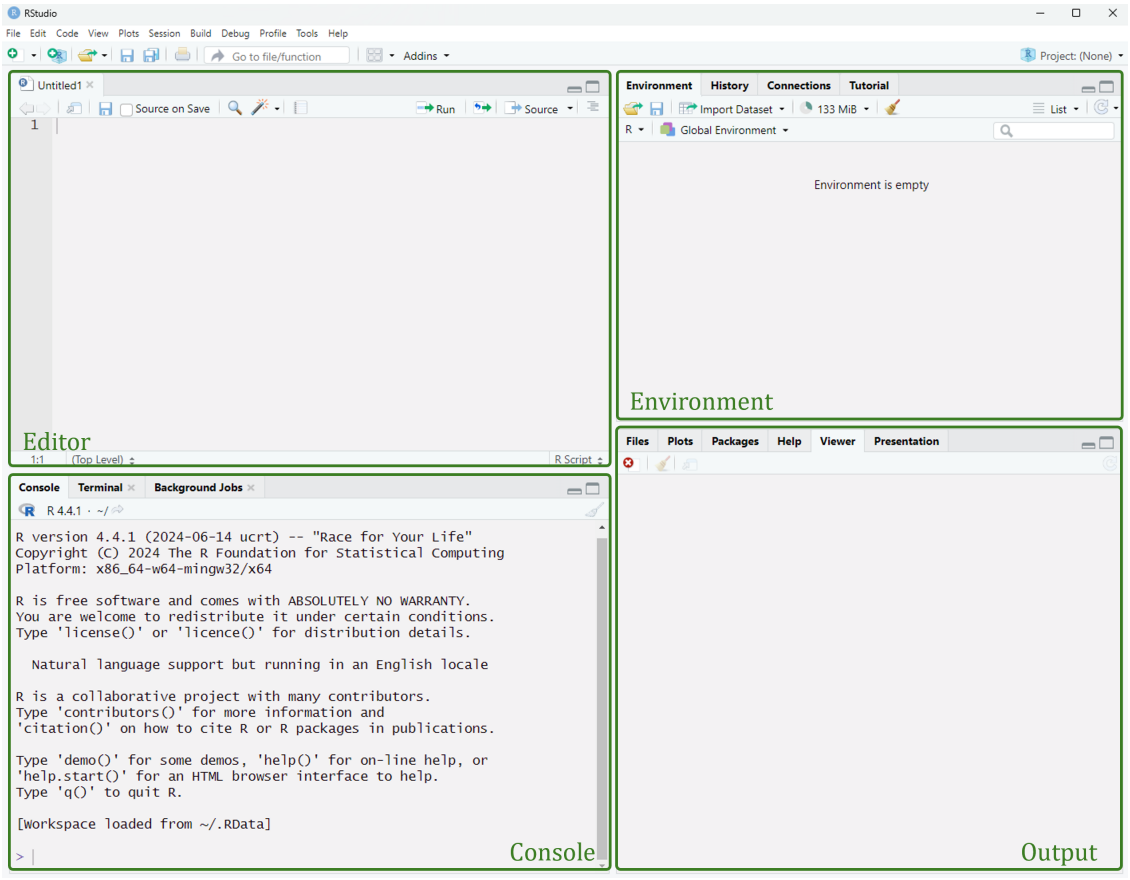
Day 2 - Intro to R and RStudio

Introduction

The purpose of this activity is to install R and RStudio and develop some familiarization with them as tools for conducting statistical analyses. In your groups, work through the following questions, adding text and code to this document as needed.

Installing R and Rstudio

1. Install R and RStudio. R and RStudio may be installed for free at https://cran.r-project.org/ and https://posit.co/downloads/, respectively.



The default layout of RStudio.

Running code, packages, and script files

2. To run code in RStudio, you can type code directly into the **console** panel in the bottom left. For example, type `runif(10)` in the console and hit `enter/return` to generate 10 random numbers between 0 and 1.

3. One of the main advantages of RStudio is that it allows us to keep track of multiple lines of code and save them as a script (.R) file. Click **File → New File → R Script** to open a new script file. Then type or copy-paste the following lines of code

```
print("Print is not a very useful function unless you are writing R packages!")

rbinom(1, size = 10, prob = .5) # flip a coin 10 times

round(sum(sapply(1:10, function(x) x)) + prod(3, 4)) # R can do math!

2 + 2 # it can also do math like this

sample(c("a", "b", "c"), size = 2, replace = F) # very useful function for stats
```

4. You can run code from script files in the following ways:

   - place your cursor on a line of code and press `ctrl + enter` or `cmd + return`
   - highlight multiple lines of code and press `ctrl + enter` or `cmd + return`
   - the same processes, but clicking the `Run` button at the top of the Editor panel (not recommended)

5. Functions (like `round`, `sum`, `sample`, etc.) are organized into collections called **R Packages**. Many functions are contained within packages that are automatically loaded when you open R; all the previously mentioned functions come with the `base` package, which is automatically loaded. To add new functions contained in other packages, we must first install the package (which only needs to be done once). Run the following code **in the console** to install the `tidyverse` package, which is actually a collection of useful packages.

```
# REMEMBER: only run install.packages() once in the console!
install.packages("tidyverse")
```

6. Once a package is installed, its function are not available until you load them, using the `library` function.

```
# this must go in the script file before you try to use any of the functions
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.2.1     v readr     2.2.0
v forcats   1.0.1     v stringr   1.6.0
v ggplot2   4.0.3     v tibble    3.3.1
v lubridate 1.9.5     v tidyr     1.3.2
v purrr     1.2.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to
```

```
# tibble lives in one of the packages in the tidyverse
tibble(row_id = 1:10, x = rnorm(10))
```

```
# A tibble: 10 x 2
   row_id      x
    <int>  <dbl>
 1      1 -1.30
 2      2  0.559
 3      3  0.386
 4      4 -0.203
 5      5  0.359
 6      6 -0.606
 7      7 -0.583
 8      8 -0.731
 9      9 -0.695
10     10 -1.21
```

7. **Create a STAT0116 folder in your documents or on your desktop right now!** Then create a new folder (perhaps `Day 2`), and save this script file in that folder with `ctrl + s` or `cmd + s`.

Working directories

8. One of the most important things to understand about R is: **FILES ARE SPECIFIED RELATIVE TO YOUR CURRENT WORKING DIRECTORY.** You can check your current working directory by running the following code:

```
getwd()
```

9. To read data into R, we must specify the path to the data file relative to the current working directory. For example, run the following code and notice the error:

```
frogs <- read.csv("data-Jeb.csv")
```

`No such file or directory` signals that R cannot find the file relative to your current working directory. The safest fix to this problem is to **always store your data files alongside your R scripts and .qmd files**. Then, **Session → Set Working Directory → Source File Location** will run code that sets your working directory to wherever the file you are editing lives.

10. Download the `data-Jeb.csv` file from the course website, place it in the same folder as your **saved** R script, set the working directory to the source file location, and run the provided code to load in a data set about frogs.

11. Click on the `frogs` item in the Environment tab to view the data set like an excel table.

Quarto documents

While R script files are useful for storing code, the real power of RStudio is in combining code, output, and written commentary using Quarto documents (.qmd files).

12. Click **File → New File → Quarto Document...**. Title the document "Practice with Quarto", select HTML, and click create. Set the editor type to `Source` using the buttons at the top of the Editor panel. Paste the following into the document (you may need to manually add the line breaks), below the second set of three dashes:

    ````
    ## This is a heading

    You can write text like a word document. You must include code by
    surrounding it in a **code chunk**.

    ```{r, echo = T, eval = T}
    sample(1:10, size = 4)
    ```
    ````

    Then click `Render` at the top of the editor pane. Doing so creates an HTML file that includes your text, code, and output!

13. HTML files can be useful, but nothing looks better than a crispy .pdf file! In order to render to .pdf, we need a LaTeX distribution. Modern versions of RStudio come with one (Typst), but I will prefer that we all use `tinytex` for this course, as it is made specifically for use with R. To install `tinytex`, run the following code:

    ```
    install.packages("tinytex")
    tinytex::install_tinytex()
    ```

14. Finally, verify that you are able to render to .pdf by creating a new .qmd file (**File → New File → Quarto Document...**) and selecting **PDF (LaTeX)**. Paste in the code provided in question 10, and click render. Doing so should produce a .pdf file in the same folder as the .qmd file.

15. Finally, download the `stat116_day2.zip` folder from the course website, open the .qmd file in the folder, and click render. After a couple of minutes, it should produce the very .pdf file that you are reading now. For this course, I will always provide a .pdf file and a .zip file for you.

If this feels uncomfortable, do not worry! We will spend plenty more time in class developing familiarity with R and RStudio. Look forward to the next week of class! Additionally, your practice problems this week focus on building familiarity with R.