Lab 1: Getting Started with R and RStudio

1 Part 1: Setup

1. Install R

- Go to the CRAN R download page: https://cran.r-project.org/.
- Choose your operating system (Windows, macOS, Linux).
- Download and install the latest version.

2. Install RStudio

- Go to the RStudio download page: https://posit.co/download/rstudio-desktop/
- Download the free RStudio Desktop installer.
- Install and open RStudio.

3. Test Your Setup

- Open RStudio.
- In the Console, type a simple calculation such as `2 + 2`.
- If you get back `4`, you're ready!

2 Part 2: Working with R Scripts

1. Open RStudio.
2. Go to **File** → **New File** → **R Script**.
3. Copy the following code into your new script:

```
x <- rnorm(10)
y <- rnorm(10)
lm0 <- lm(y ~ x)
lm0_summary <- summary(lm0)
```

4. Save your file as `lab1.R`.
5. Run the code (highlight and press **Ctrl+Enter** or **Cmd+Enter**).

3 Part 3: Working with Rmd Documents

R Markdown (`.Rmd`) files allow you to combine text, code, and output in a single document. They are especially useful for writing reproducible reports.

Creating an R Markdown file:

1. In RStudio, go to **File** → **New File** → **R Markdown...**.
2. Give your document a title and author name.
3. Choose **HTML** as the default output format.
4. Click **OK** to create the file.

An R Markdown file has three main parts:

- **YAML header** (at the top, between `---`): contains metadata such as title, author, and output format.
- **Text sections**: written in plain Markdown. You can format text using headers (#), lists, bold/italic, etc.
- **Code chunks**: regions where you include R code. Code chunks start with three backticks and `{r}`, and end with three backticks.

Running code in R Markdown:

- You can run individual chunks by clicking the green "play" button at the top-right of the chunk.
- You can run the whole document by clicking **Knit** (the blue yarn ball button at the top of RStudio).
- Try out formatting the code above in an Rmd document.

4 Part 4: R Objects

Answer the following questions by typing code into your script:

1. Use the `typeof` function to find the types of the objects `x`, `y`, `lm0`, and `lm0_summary`.
2. Use the `str` function to look at the elements of `lm0_summary`. Describe a few of the elements you see (there are 11 total, no need to describe them all).
3. What is the type of `lm0_summary[4]`? What is the type of `lm0_summary[[4]]`? Explain the difference between single brackets `[]` and double brackets `[[]]`.
4. Suppose you wanted a vector containing just the standard errors of the coefficient estimates in this model. Notice that the standard errors, along with some other information, are contained in the fourth element of `lm0_summary`.

   What is wrong with the following strategy?

   ```
   estimates_and_errors <- lm0_summary[4]
   std_errors <- estimates_and_errors[,2]
   ```

5. How would you modify the code above so that `std_errors` is a vector with the standard errors?