Objectives:

* Understand the R environment and RStudio.
* Use R in interactive sessions.

What is R?

R is a free software environment that includes a set of base packages for graphics, math, and statistics. You can also make use of specialized packages contributed by R users or write your own new functions. R was developed as a part of the [GNU project](http://www.gnu.org/) from the S language.

\*\*Advantages\*\*

* Offers direct programmability and extensibility, serving as a complete programming language.
* Benefits from extensive community support, with numerous robust advanced packages readily available.
* Provides comprehensive built-in functionalities for data manipulation, analysis, and scientific visualization.
* - Free of cost!

However, becoming proficient in R and mastering its intricacies may entail a steep learning curve (which we are about to embark upon). Specifically, R can be daunting at first glance: with its customizable and extensible programming framework, users must navigate through a mix of core language features and a plethora of built-in or extended functions offered by base and add-on packages, respectively.

R is unforgiving; it diligently attempts (sometimes overly so!) to execute given commands, often resorting to complex and unexpected data transformations to achieve results, even if they do not align with the user's expectations.

Effectively utilizing R necessitates learning diagnostic techniques and cultivating the habit of testing and validating intermediate steps. While R lacks professional or commercial support, a plethora of resources including books, tutorials, manuals, and mailing lists are available both online and offline.

What is RStudio?

RStudio is an integrated development environment (IDE) specifically designed for working with the R programming language. It provides a user-friendly interface for writing code, running scripts, visualizing data, and managing projects. RStudio offers several features that enhance the R programming experience, including:

1. \*\*Script Editor\*\*: RStudio includes a script editor with features such as syntax highlighting, code completion, and code folding, which make writing and editing R code more efficient.

2. \*\*Console\*\*: It includes a console where R code can be executed interactively. Users can run individual commands, view output, and debug code directly within the console.

3. \*\*Data Viewer\*\*: RStudio's data viewer allows users to easily explore and interact with data frames and other R objects. It provides a spreadsheet-like interface for viewing data, making it easier to understand its structure and contents.

4. \*\*Plots and Visualization\*\*: RStudio provides tools for creating and customizing plots and visualizations directly within the IDE. Users can generate a wide variety of charts and graphs using R's powerful plotting capabilities.

5. \*\*Package Management\*\*: RStudio includes features for managing R packages, such as installing, updating, and loading packages. It also provides tools for browsing and searching packages from CRAN (Comprehensive R Archive Network) and other repositories.

6. \*\*Project Management\*\*: RStudio supports project-based workflows, allowing users to organize their R code, data, and related files into projects. Projects help maintain a clean directory structure and make it easier to collaborate with others.

7. \*\*Version Control\*\*: RStudio integrates with version control systems such as Git and SVN, enabling users to track changes to their code and collaborate with others more effectively.

Overall, RStudio is a powerful and user-friendly environment for working with R, suitable for both beginners and experienced R users alike.

Based on the provided R Markdown document, we can define the following objectives:

1. \*\*Understanding Basic R Operations\*\*:

- Learn to perform basic arithmetic operations such as addition, subtraction, multiplication, division, and exponentiation in R.

- Understand how to assign values to variables.

- Familiarize with different methods of subsetting vectors, including indexing, logical conditions, and functions like `subset()`.

2. \*\*Working with Basic Data Structures in R\*\*:

- Gain knowledge about fundamental data structures in R such as vectors, matrices, arrays, lists, and data frames.

- Understand how to create and manipulate these data structures.

- Learn about subsetting data frames based on specific conditions using logical operators (`&`, `|`, `!`).

3. \*\*Logical Operators in Data Subsetting\*\*:

- Understand the usage of logical operators (`&` for "AND", `|` for "OR", `!` for "NOT") in subsetting data frames.

- Learn to combine logical operators to create complex conditions for data subsetting.

4. \*\*Practical Application\*\*:

- Apply the learned concepts in practical scenarios such as subsetting data frames based on multiple conditions.

- Execute R code snippets provided in the document to practice and reinforce understanding.

5. \*\*Documentation and Note-taking\*\*:

- Understand the importance of documentation and inline comments in R code for better understanding and reproducibility.

- Learn to document R code using markdown syntax for clear presentation and explanation.

These objectives aim to provide a foundational understanding of R programming language, basic operations, data structures, and practical applications in data manipulation and analysis.