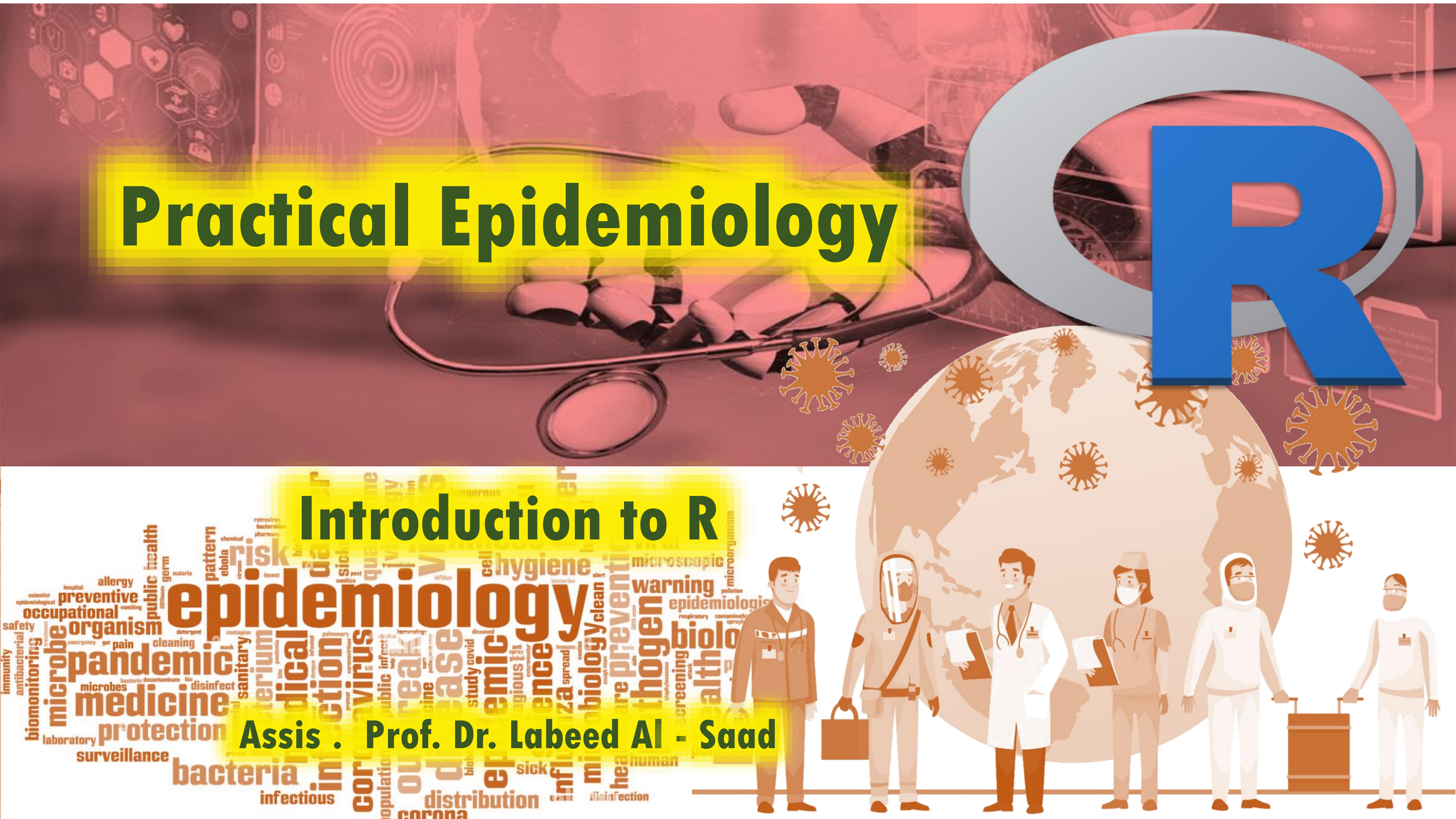


Practical Epidemiology

Introduction to R

Assis . Prof. Dr. Labeed Al - Saad

The Objectives

- What is R.
- Why use R.
- How to install R.
- What is Rstudio and how to install it.
- Rstudio interface description.
- What is working directory.
- R Print Output.
- R Comments.

What is R?

R software is a powerful and versatile programming language and environment specifically designed for statistical computing, data analysis, and graphical representation. In the context of bioinformatics, R is widely used for analyzing and interpreting complex biological data, such as genomic sequences, gene expression data, and proteomic data.

Why use R?

- Statistical Analysis: R provides a comprehensive suite of statistical techniques, including linear and nonlinear modeling, classical statistical tests, time-series analysis, classification, and clustering. These capabilities are essential for analyzing high-throughput biological data.
- Data Manipulation: R offers extensive tools for data manipulation, including data cleaning, transformation, and aggregation. This is crucial for handling the large and often messy datasets encountered in bioinformatics.
- Graphical Capabilities: R has advanced graphical capabilities for creating high-quality plots and visualizations. This is important for visualizing complex biological data, such as heatmaps, phylogenetic trees, and network graphs.
- Packages and Libraries: R has a rich ecosystem of packages and libraries specifically tailored for bioinformatics. Some popular bioinformatics packages include Bioconductor, which provides tools for the analysis of genomic data, and ggplot2 for advanced data visualization.

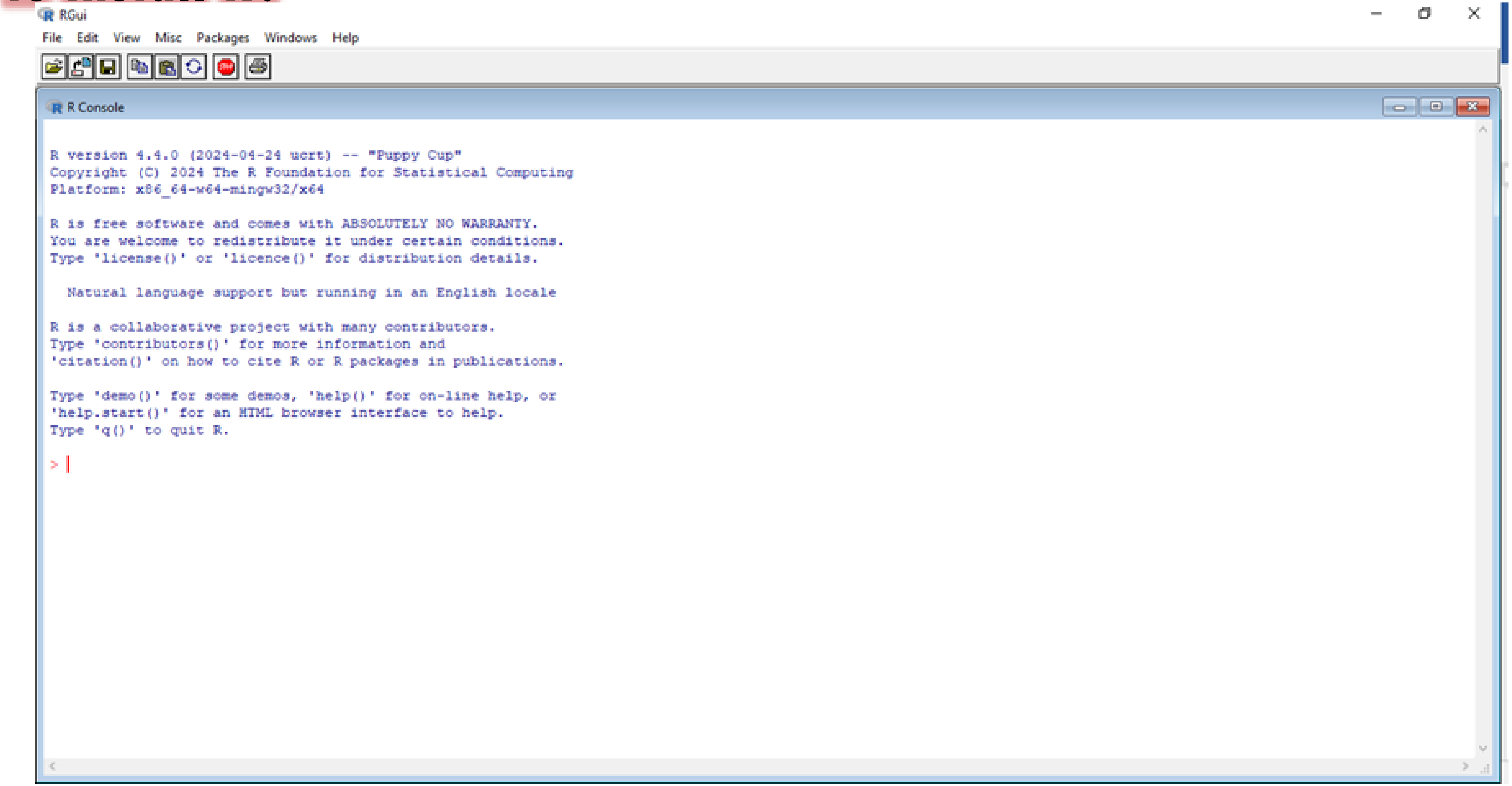
Why use R?

- Reproducibility: R supports reproducible research through the use of scripts and markdown documents (e.g., R Markdown). This allows bioinformaticians to document their analysis workflows and share them with others, ensuring transparency and reproducibility.
- Integration with Other Tools: R can be integrated with other programming languages and tools commonly used in bioinformatics, such as Python, Perl, and SQL databases. This allows for seamless workflows that leverage the strengths of multiple tools.
- Community and Support: R has a large and active community of users and developers, which means that there is a wealth of resources, tutorials, and forums available for learning and troubleshooting. This is particularly valuable for bioinformaticians who may need to quickly adapt to new challenges and technologies.

How to install R?

- To install R, go to https://cloud.r-project.org/ and download the latest version of R for Windows, Mac or Linux.
- When you have downloaded and installed R, you can run R on your computer.
- The screenshot below shows how it may look like when you run R on a Windows PC:

How to install R?



```
R version 4.4.0 (2024-04-24 ucrt) -- "Puppy Cup"
Copyright (C) 2024 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> 
```

What is Rstudio ?

- RStudio is a popular integrated development environment (IDE) for the R programming language.
- It provides a user-friendly interface for writing, executing, and debugging R code, as well as managing data, visualizations, and projects.
- RStudio is available in both desktop and server versions and supports multiple operating systems, including Windows, macOS, and Linux.

How to install Rstudio ?

1. Download RStudio

- Go to the official RStudio website: https://www.rstudio.com/products/rstudio/download/.
- Choose the free version of RStudio Desktop (unless you need the paid Pro version).
- Select the installer for your operating system:
    - Windows: Download the .exe file.
    - macOS: Download the .dmg file.
    - Linux: Download the .deb (for Debian/Ubuntu) or .rpm (for Fedora/Red Hat) file.

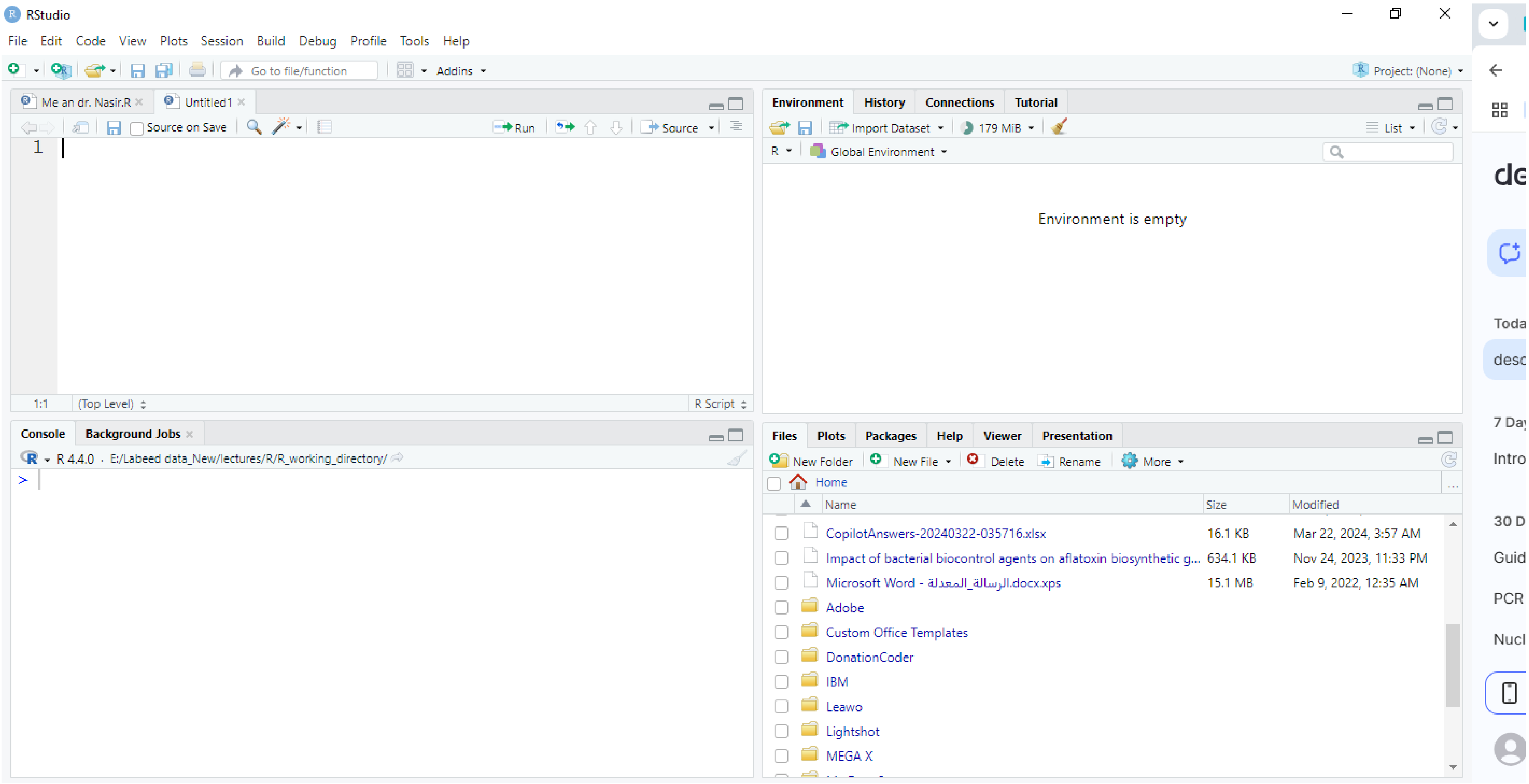
How to install Rstudio ?

2. Install Rstudio on Windows:

- Locate the downloaded .exe file and double-click it.
- Follow the installation wizard:
  - Accept the license agreement.
  - Choose the installation location (default is fine for most users).
  - Select any additional tasks (e.g., creating a desktop shortcut).
- Click Install and wait for the installation to complete.
- Launch RStudio from the Start menu or desktop shortcut.

How to install Rstudio ?

When you open RStudio on a Windows system, the interface is divided into several panes or windows, each serving a specific purpose.



Environment is empty

R 4.4.0 · E:/Labeed data_New/lectures/R/R_working_directory/

```
>
```

Name	Size	Modified
CopilotAnswers-20240322-035716.xlsx	16.1 KB	Mar 22, 2024, 3:57 AM
Impact of bacterial biocontrol agents on aflatoxin biosynthetic g...	634.1 KB	Nov 24, 2023, 11:33 PM
Microsoft Word - الرسالة_المعدلة.docx.xps	15.1 MB	Feb 9, 2022, 12:35 AM
Adobe		
Custom Office Templates		
DonationCoder		
IBM		
Leawo		
Lightshot		
MEGA X		

Rstudio Interface Description?

1. Source Editor (Top-Left Pane)

- This is where you write and edit your R scripts.
- You can open multiple script files as tabs within this pane.
- Features include syntax highlighting, code completion, and line numbers.
- You can run selected lines of code or the entire script directly from this pane.

Rstudio Interface Description?

2. Console (Bottom-Left Pane)

- The console is where R code is executed interactively.
- You can type commands directly into the console and see the results immediately.
- Output from scripts or commands run in the source editor also appears here.
- The console also displays error messages, warnings, and other diagnostic information

Rstudio Interface Description?

3. Environment/History (Top-Right Pane)

- Environment Tab: Displays all the objects (variables, data frames, functions, etc.) currently loaded in your R session. You can view and inspect these objects.
- History Tab: Shows a list of all the commands you've executed in the current session. You can search, rerun, or save these commands.

Rstudio Interface Description?

4. Files/Plots/Packages/Help/Viewer (Bottom-Right Pane)

- Files Tab: Allows you to navigate and manage files and folders in your working directory.
- Plots Tab: Displays graphs and visualizations generated by your R code. You can zoom, export, or navigate through multiple plots.
- Packages Tab: Shows a list of installed R packages. You can load, update, or install new packages from here.
- Help Tab: Provides access to R documentation and help files. You can search for functions and view their usage and examples.
- Viewer Tab: Used to display local web content, such as HTML widgets or Shiny app previews

Additional Features in RStudio for Windows:

- Toolbar: Located at the top of the window, it provides quick access to common actions like opening files, saving scripts, running code, and managing projects.
- Workspace Management: RStudio allows you to save and load workspaces, making it easy to resume your work later.
- Project Management: You can create and manage R projects, which help organize your scripts, data, and outputs in a structured way.
- Customization: You can rearrange panes, change themes, and adjust font sizes to suit your preferences.
- Integration with Version Control: RStudio supports Git and SVN for version control, allowing you to track changes and collaborate on projects.

Keyboard Shortcuts (Windows-Specific):

- Run selected code: Ctrl + Enter
- Run entire script: Ctrl + Shift + Enter
- Comment/Uncomment lines: Ctrl + Shift + C
- Insert assignment operator (<-): Alt + -
- Open a new script: Ctrl + Shift + N
- Save current script: Ctrl + S
- Open the console: Ctrl + 2
- Clear console: Ctrl + L

What is Working Directory?

- In R, the working directory is the folder or directory on your computer where R looks for files (e.g., data files, scripts) and where it saves outputs (e.g., plots, exported data) by default. It serves as the reference point for file paths in your R session.

Key Points About the Working Directory:

- In R, the working directory is the folder or directory on your computer where R looks for files (e.g., data files, scripts) and where it saves outputs (e.g., plots, exported data) by default. It serves as the reference point for file paths in your R session.

Key Points About the Working Directory:

1. Check the current working directory:

```
getwd()
```

Output might be: "C:/Users/YourName/Documents"

2. Change the working directory:

```
setwd("C:/Projects/R")
```

3. Verify the change:

```
getwd()
```

Output: "C:/Projects/R"

R Print Output:

Print:

Unlike many other programming languages, you can output code in R without using a print function:

Example:

```
> "Hello world"
[1] "Hello world"
```

However, R does have a print() function available if you want to use it. This might be useful if you are familiar with other programming languages, such as Python, which often uses the print() function to output code.

Example:

```
> print("Hello world")
[1] "Hello world"
```

R Print Output:

And there are times you must use the print() function to output code, for example when working with for loops (which you will learn more about in a later chapter):

Example:

```
> for (x in 1:10) {print(x)}
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
[1] 6
[1] 7
[1] 8
[1] 9
[1] 10
```

R Comments:

Comments

Comments can be used to explain R code, and to make it more readable. It can also be used to prevent execution when testing alternative code.

Comments starts with a #. When executing code, R will ignore anything that starts with #.

This example uses a comment before a line of code:

Example:

```
> # This is a comment
> "Hello World!"
[1] "Hello World!“
```

This example uses a comment at the end of a line of code:

```
> "Hello World!" # This is a comment
[1] "Hello World!"
```

R Comments:

Comments does not have to be text to explain the code, it can also be used to prevent R from executing the code:

Example (The first code line will not executed):

```
> # "Good morning!"
> "Good night!"
[1] "Good night!“
```

Multiline Comments:

Unlike other programming languages, such as Java, there are no syntax in R for multiline comments. However, we can just insert a # for each line to create multiline comments:

Example:

```
> # This is a comment
> # written in
> # more than just one line
> "Hello World!"
[1] "Hello World!"
```


THANK YOU!