



UNIT-2

Introduction to R

A large blue 'R' logo is positioned on the left side of the slide. The background is dark with glowing green and blue lines, resembling a network or data visualization. There are also some faint binary digits (0s and 1s) scattered around.

History of R

S.no	Year	Author's/ Creator's	Contribution
1.	1991	Ross Ihaka and Robert Gentleman	Created R
2.	1993	Ross Ihaka and Robert Gentleman	Announcement of R was made to the public
3.	1995	Martin Machler	Gave idea to the authors to use the GNU to make R free software.
4.	1996	Ross Ihaka and Robert Gentleman	Published R Documentation for data analysis and graphics in the “ <i>Journal of Computational and Graphical Statistics</i> ”
5.	1997	Ross Ihaka and Robert Gentleman	R Core Group was formed and controls the source code for R.
6.	2000	Ross Ihaka and Robert Gentleman	R version 1.0.0 was released to the public.



What is R

- R is a programming/scripting language for statistical data manipulation and analysis, visualization and reporting.
- Because R is open source software, it's easy to get help from the user community.
- So you can use it to automate analyses and create new functions that extend the existing language features.
- A lot of new functions are contributed by users, many of whom are prominent statisticians.
- R is used by statisticians with advanced machine learning training and by programmers familiar with other languages.



What is R used for?

- Statistical Inference
- Data Analysis
- Data Visualization
- Machine Learning Algorithms





R Vs R Studio

- **R Studio** is a software that provides an interface to R.
- It's sometimes referred to as an Integrated Development Environment (IDE).

R Studio comes in two flavors:

- A desktop application that installs directly on your computer
- A server application that is accessible via a web browser.

Both platforms offer nearly identical experiences. The former runs on top of R installed on your computers, the latter runs off of an instance of R running on a remote server.



Basic Features of R

- R runs on any standard computing platform and operating system.
- Its open-source nature means that anyone is free to adapt the software.
- R shares with many popular open-source projects with frequent releases and bugs will be addressed in a timely manner which is one of the nice features.
- R has many other statistical packages and many more visualization packages.



Basic Features of R

- Other graphics packages like lattice and ggplot2 allow for complex and sophisticated visualizations of high-dimensional data.
- R provides a language that is both useful for interactive work and contains a powerful programming language for developing tools.
- R is a platform and thousands of people around the world, researchers, scientists come together to make contributions to R, and to develop packages for all kinds of applications.




Getting Started with R

- R is a wonderful tool for statistical analysis, visualization and reporting.
- R compiles and runs on a wide variety platforms like Windows, Mac OS X, and Linux systems.
- R can be downloaded and installed from CRAN website. CRAN stands for Comprehensive R Archive Network.

Step by Step procedure for Installing R on Windows:

Step1: First we have to download R setup from <https://cran.r-project.org/>





CRAN

[Mirrors](#)

[What's new?](#)

[Search](#)

[CRAN Team](#)

About R

[R Homepage](#)

[The R Journal](#)

Software

[R Sources](#)

[R Binaries](#)

[Packages](#)

[Task Views](#)

[Other](#)

Documentation

[Manuals](#)

[FAQs](#)

[Contributed](#)

Donations

[Donate](#)

R-4.4.1 for Windows

[Download R-4.4.1 for Windows](#) (82 megabytes, 64 bit)

[README on the Windows binary distribution](#)

[New features in this version](#)

This build requires UCRT, which is part of Windows since Windows 10 and Windows Server 2016. On older systems, UCRT has to be installed manually from [here](#).

If you want to double-check that the package you have downloaded matches the package distributed by CRAN, you can compare the [md5sum](#) of the .exe to the [fingerprint](#) on the master server.

Frequently asked questions

- [Does R run under my version of Windows?](#)
- [How do I update packages in my previous version of R?](#)

Please see the [R FAQ](#) for general information about R and the [R Windows FAQ](#) for Windows-specific information.

Other builds

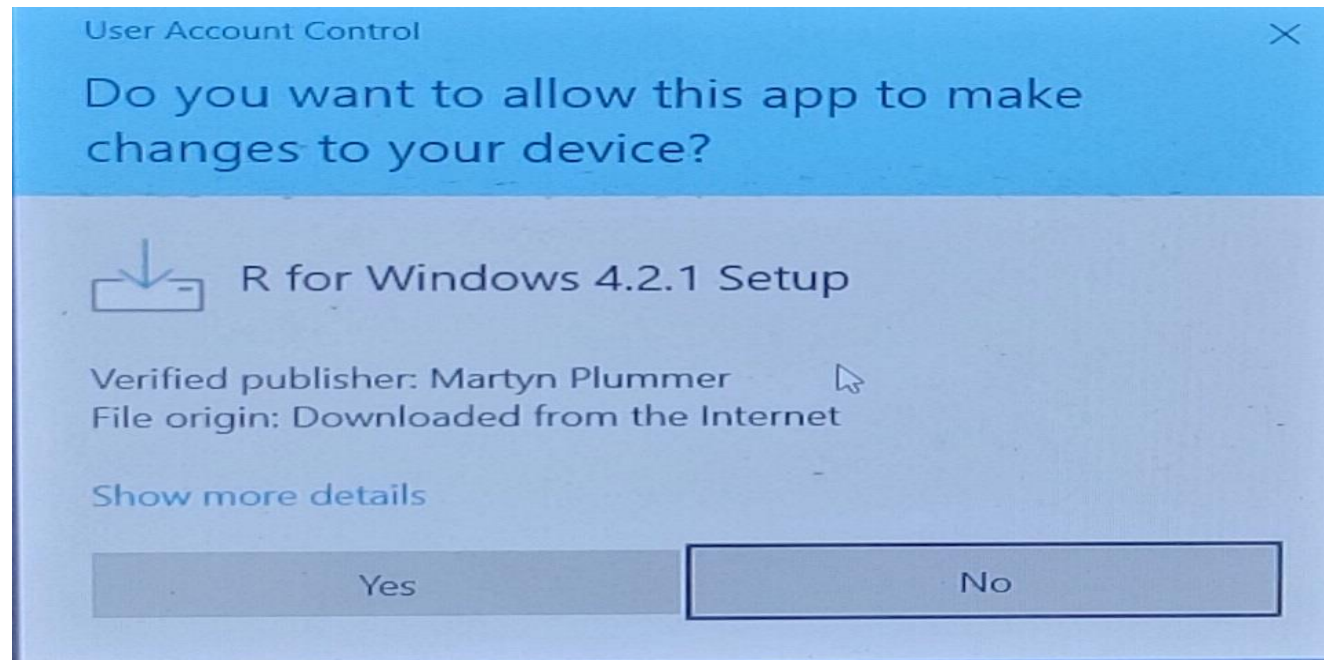
- Patches to this release are incorporated in the [r-patched snapshot build](#).
- A build of the development version (which will eventually become the next major release of R) is available in the [r-devel snapshot build](#).
- [Previous releases](#)

Note to webmasters: A stable link which will redirect to the current Windows binary release is [<CRAN MIRROR>/bin/windows/base/release.html](https://cran.r-project.org/bin/windows/base/release.html).

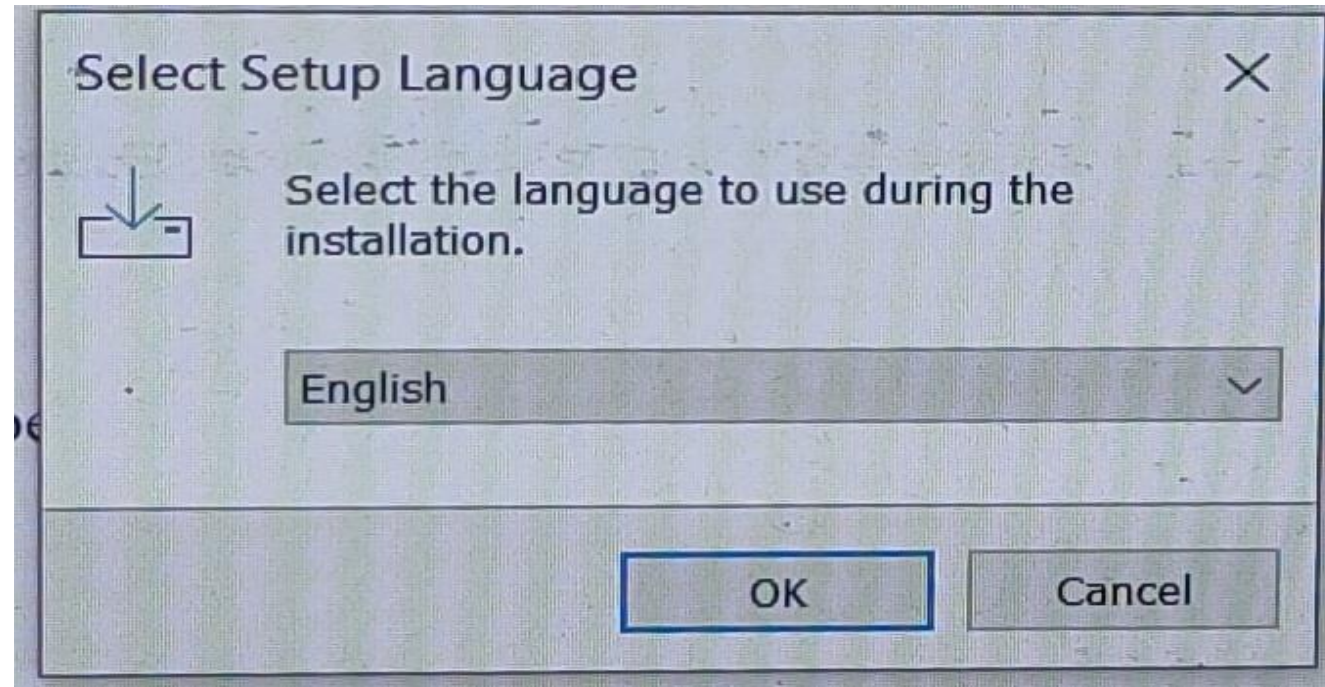
Last change: 2024-06-15

Step 2: When we click on **Download R 4.4.1 for windows**, our downloading will be the start of R setup. Once the downloading is finished, we have to run the setup of R in the following way:

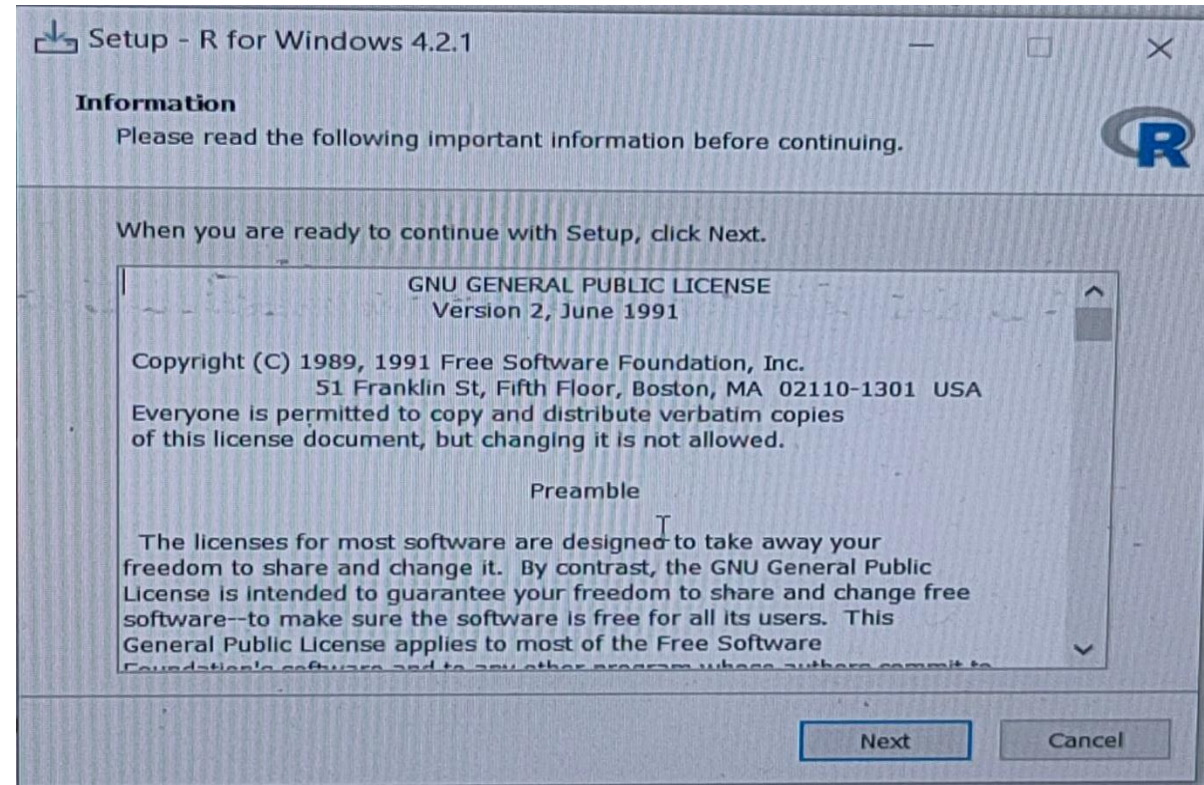
1) Click on the downloaded R setup. You can see this pop-up message whether to allow this or not. Now click on “Yes”.



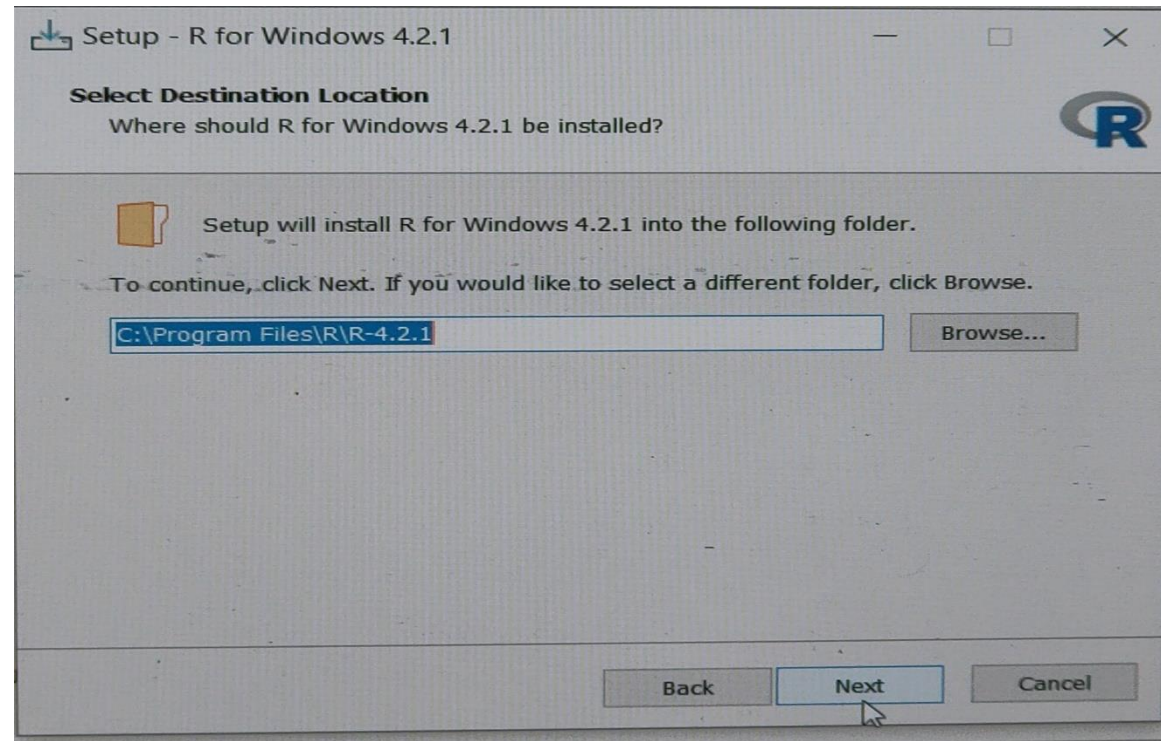
- 2) Now it shows you a dialog box and ask to select the language you prefer. Select English and then click on OK.



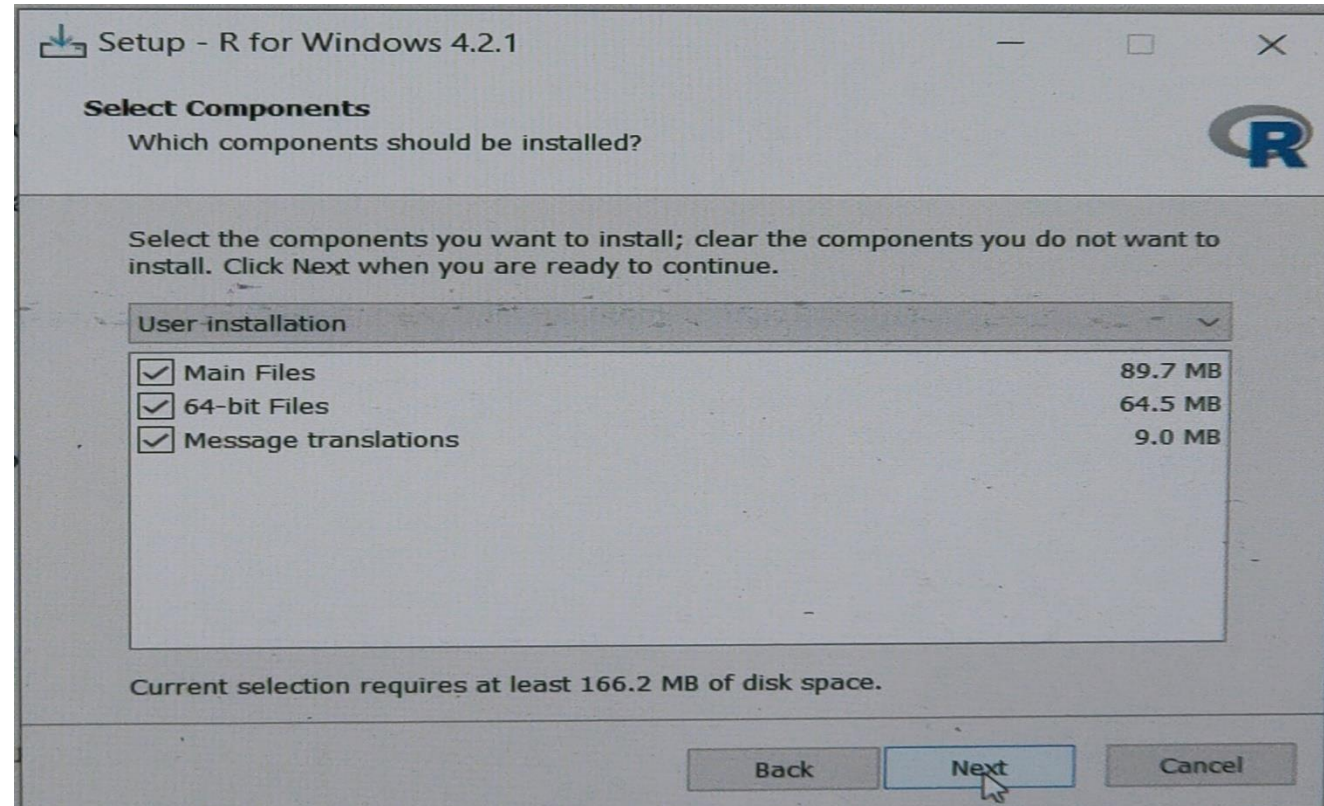
- 3) Now read the information regarding the R and then click on NEXT.



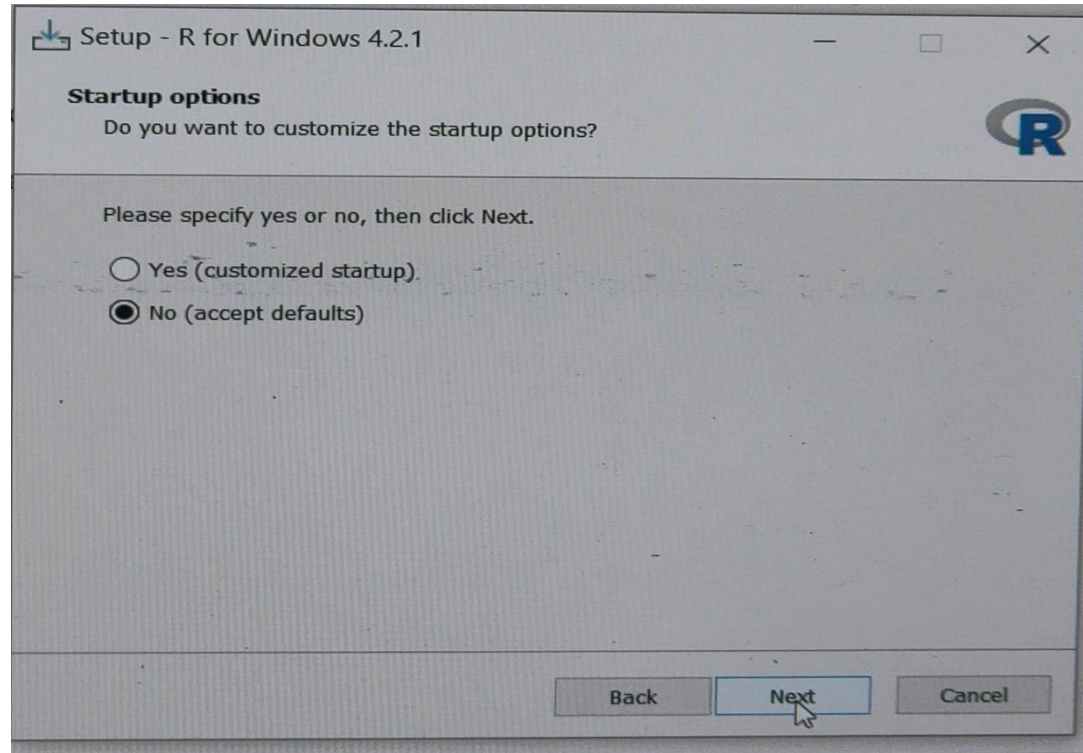
4) Select the path where we want to download the R and proceed to Next.



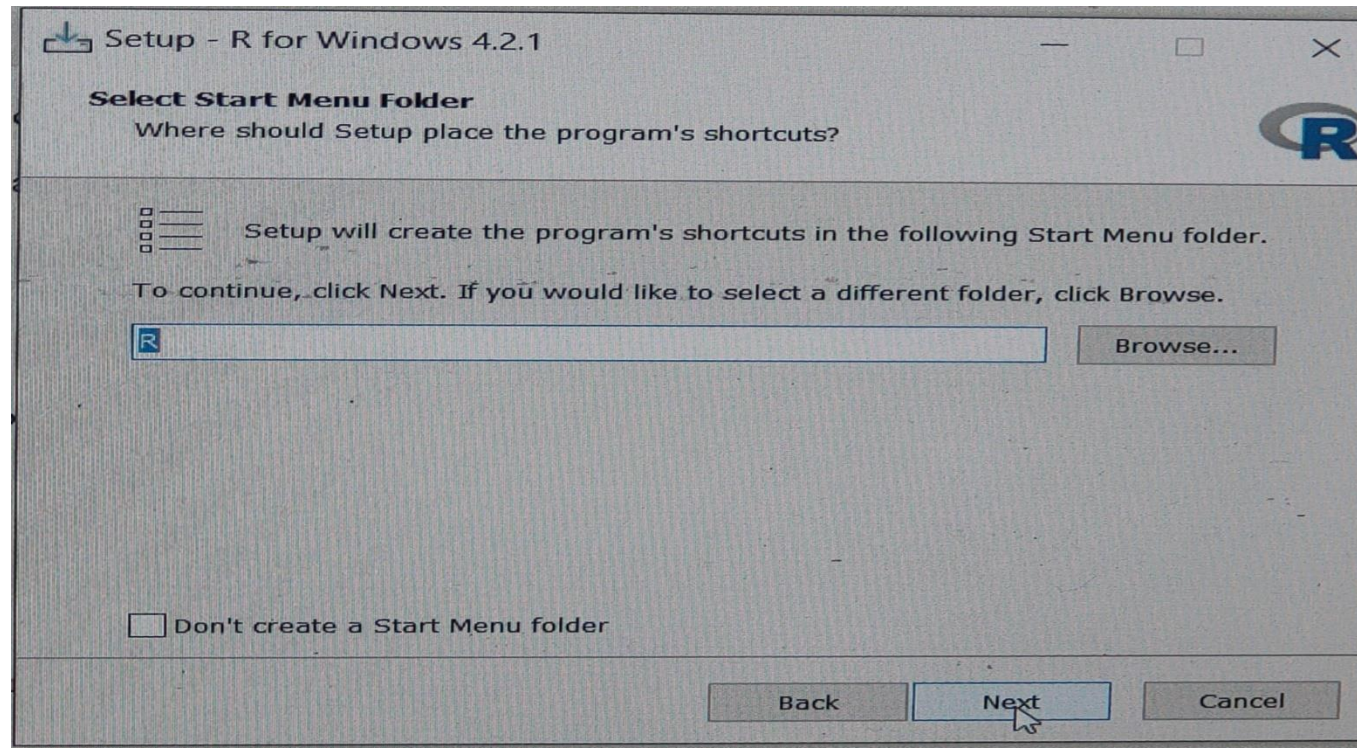
- 5) Select all components which we want to install, and then we will proceed to **Next**.



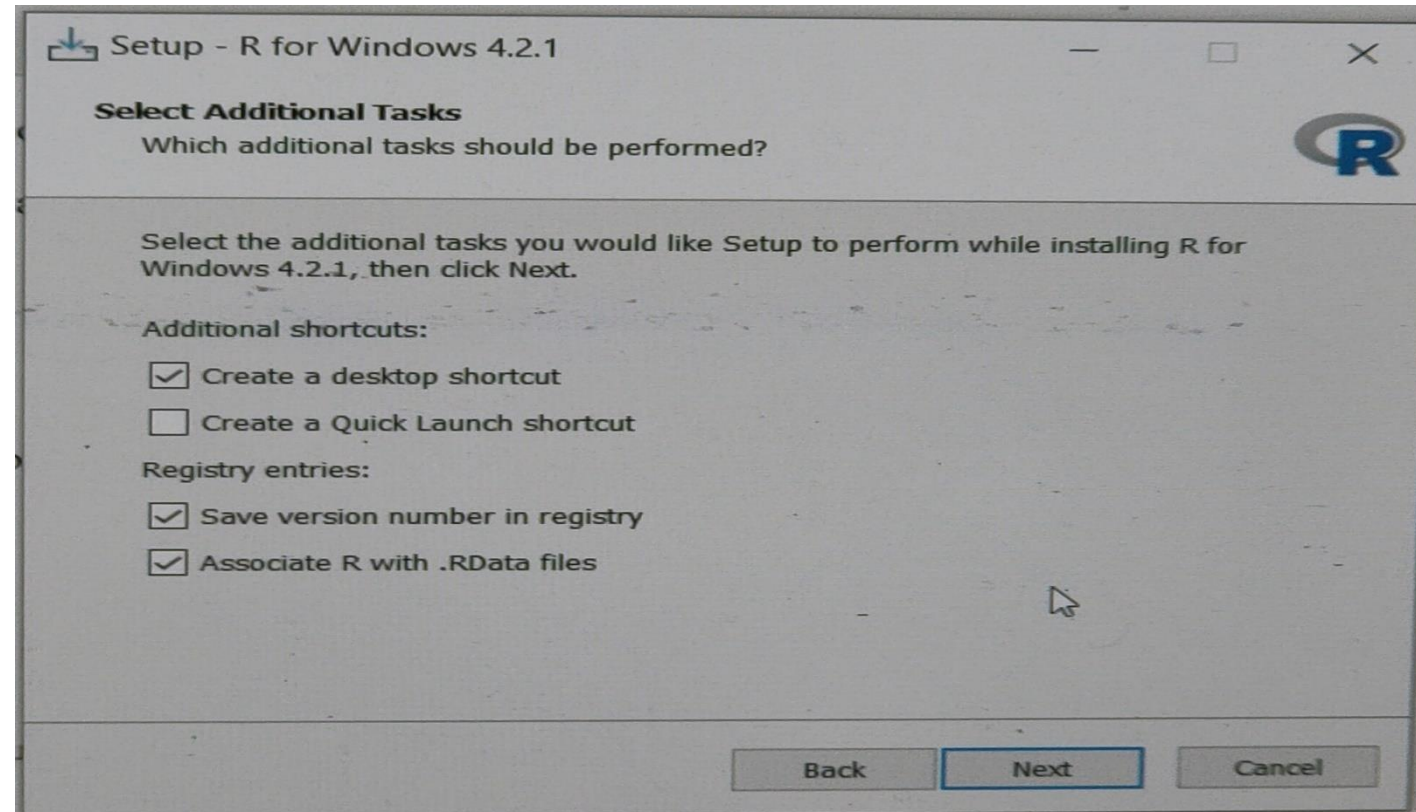
- 6) In the next step, we have to select either customized startup or accept the default, and then we proceed to **Next**.



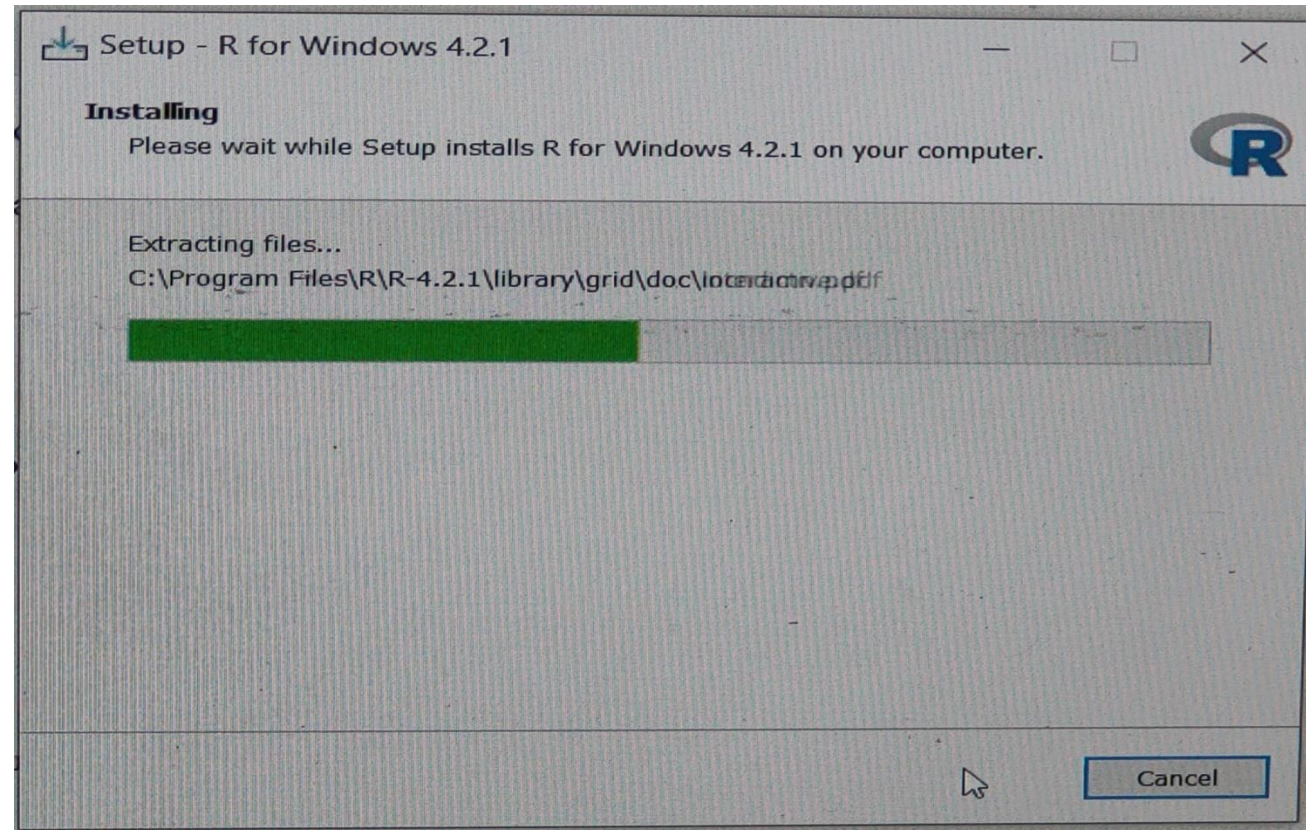
7) Select the path where we want to download R folder and proceed to Next.



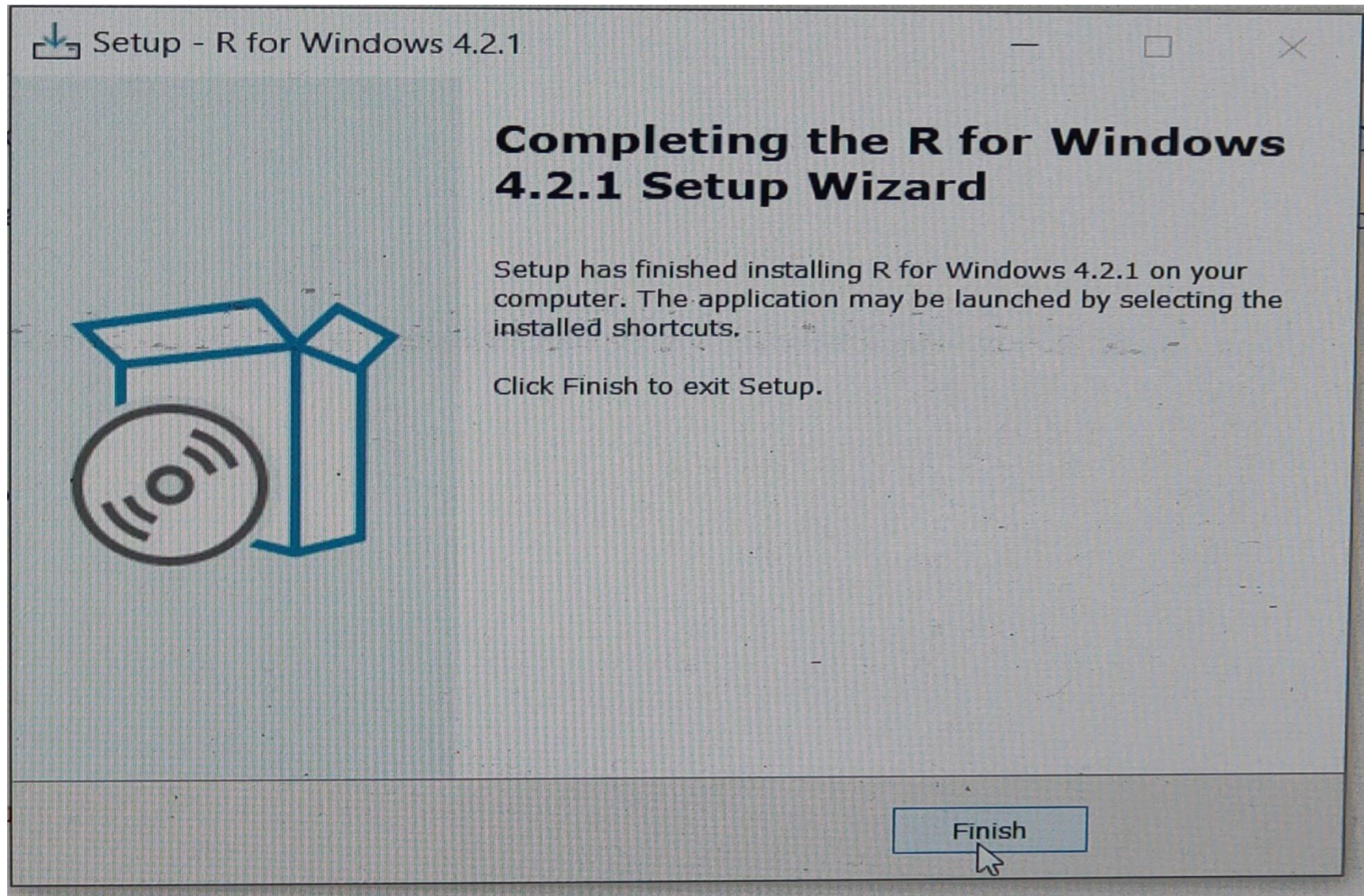
8) We have to select additional tasks and then proceed to Next.



9) When we proceed to next, our installation of R in our system will get started:

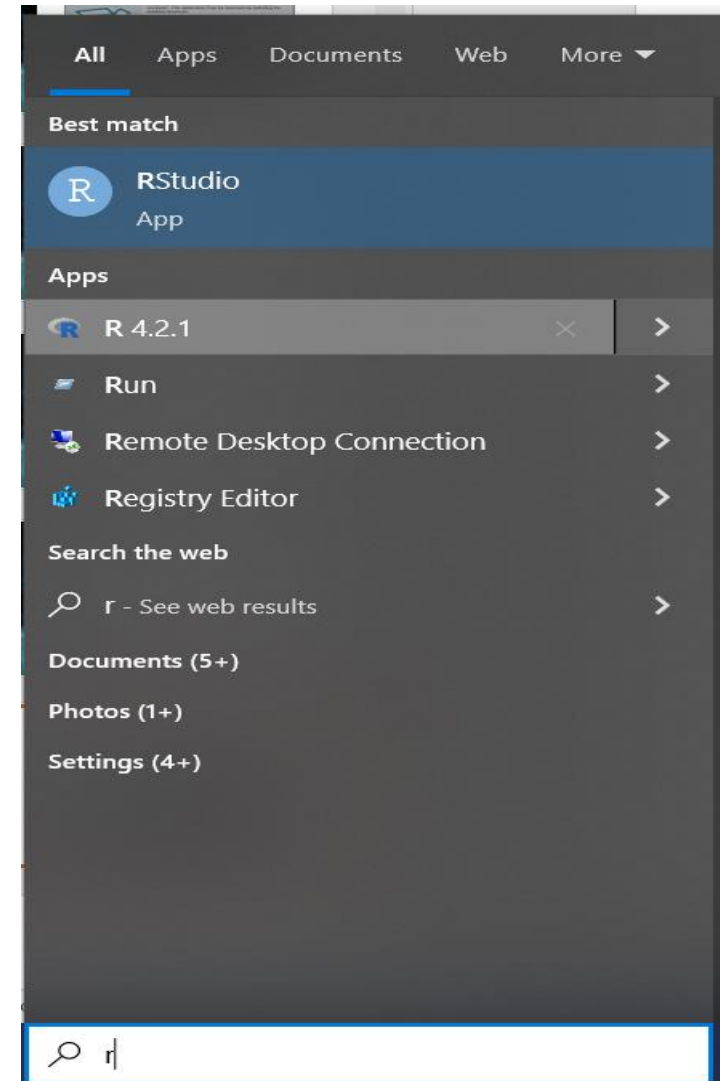


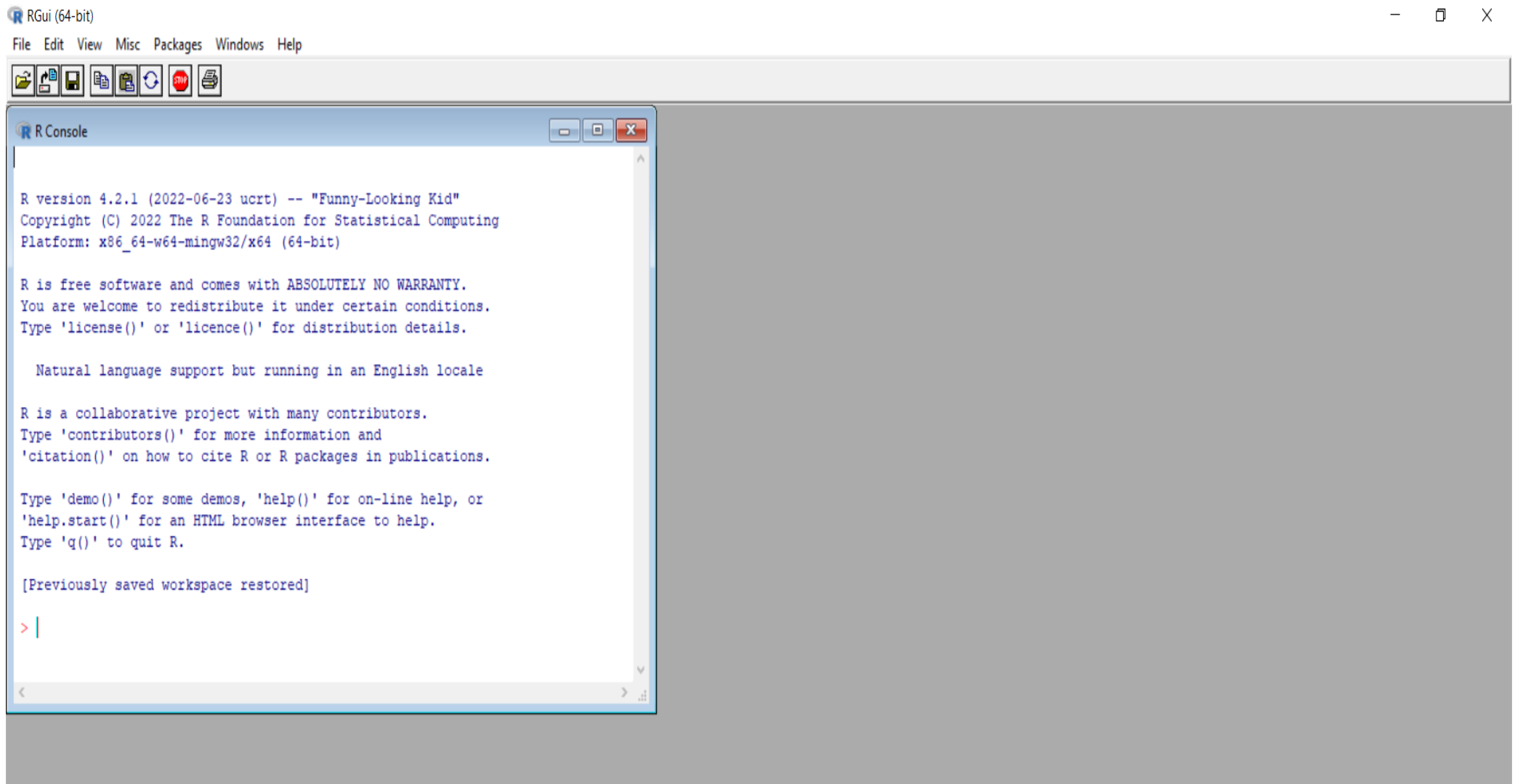
10) In the last, we will click on finish to successfully install R in our system.





- 11) Now go to windows search and type R you will be able to see the R 4.4.1 version. Now click on it.
- 12) The standard R interface in Windows can be seen.



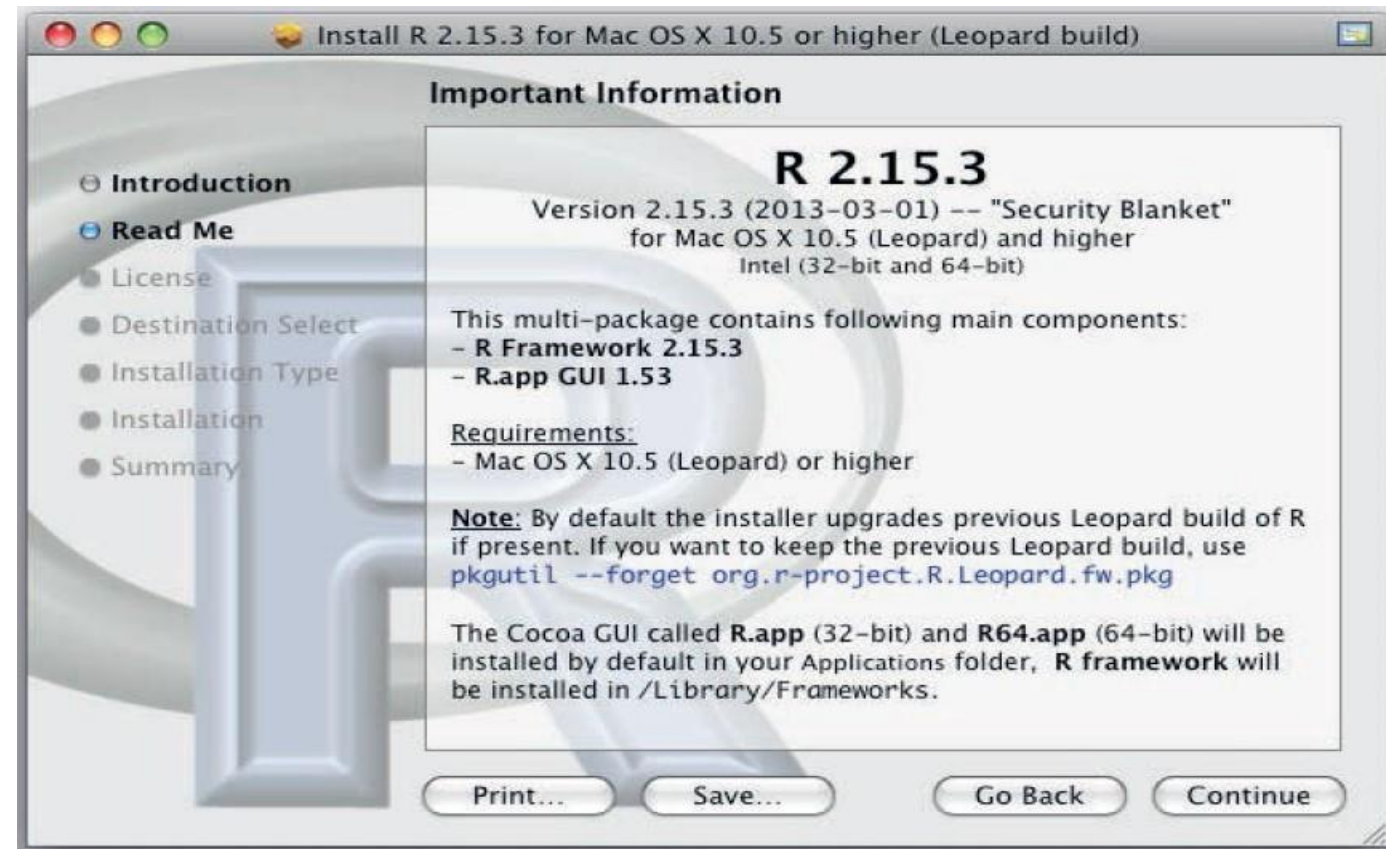


Step by Step procedure for Installing R on Mac OS

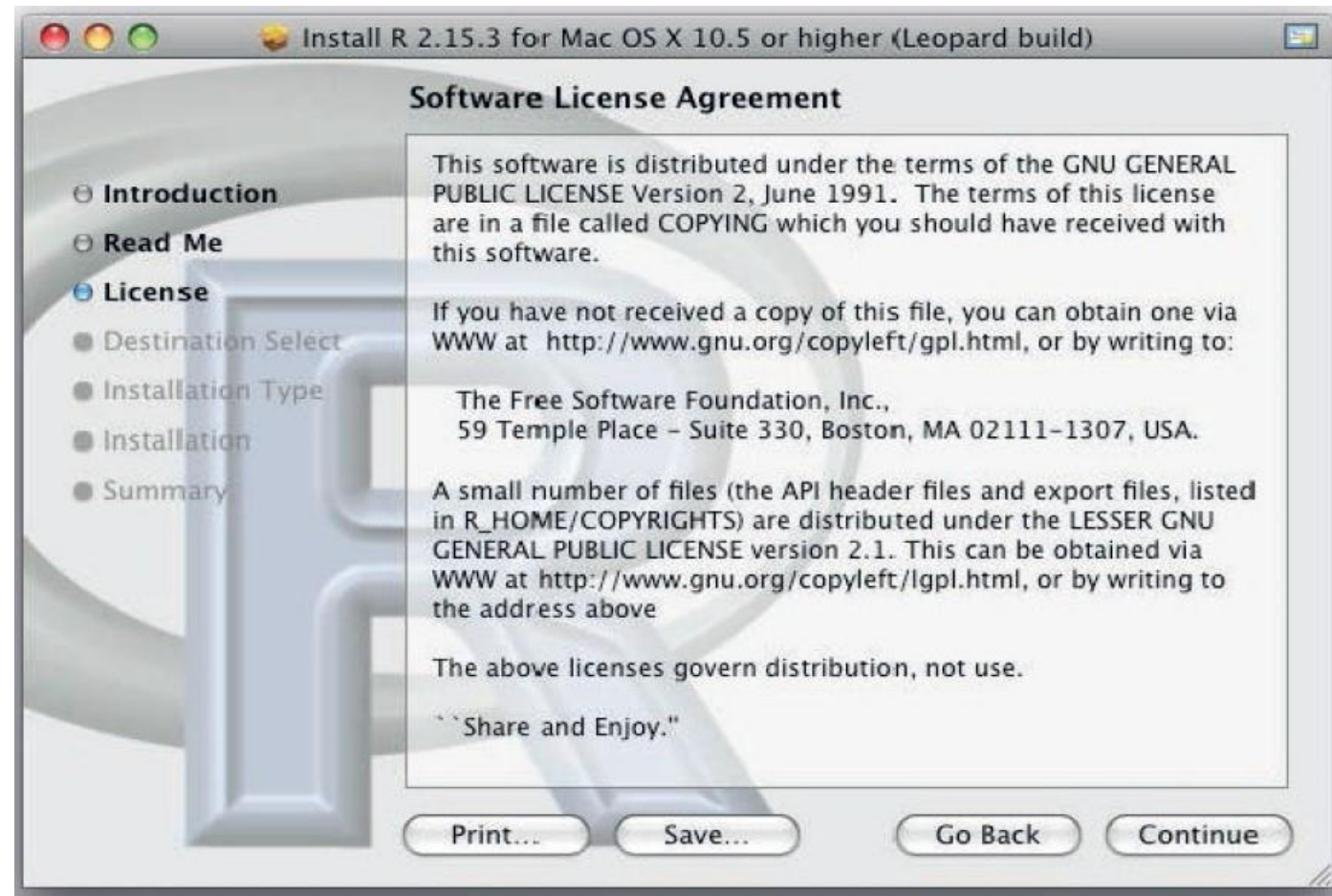
- ❑ Find the appropriate installer, which ends in .pkg, and launch it by double-clicking. This brings up the introduction, click Continue to begin the installation process.



- ❑ This brings up some information about the version of R being installed. There is nothing to do except click Continue.



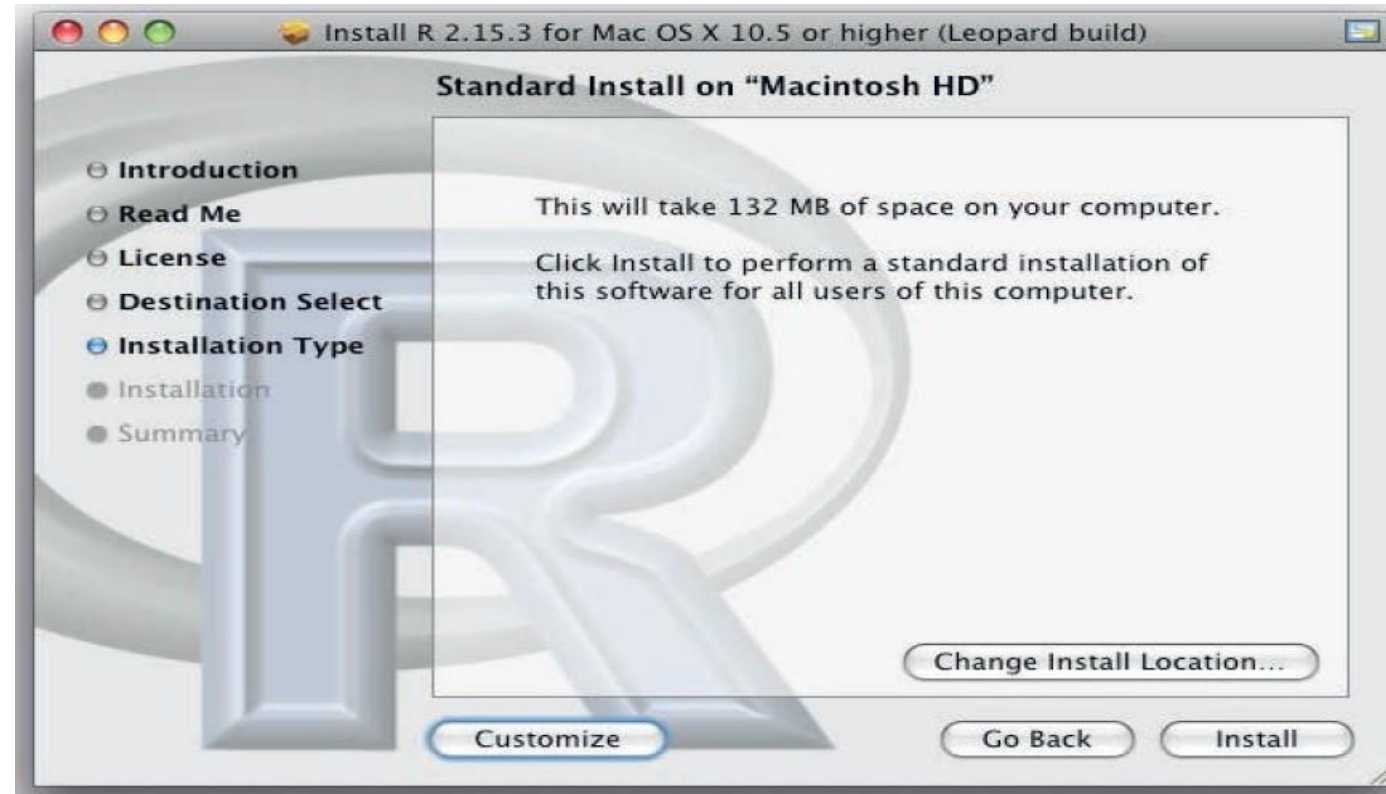
- ❑ Then the license information is displayed. Click Continue to proceed.



- ☐ Click Agree to confirm that the license is agreed to, which is mandatory to use R.



- ☐ To install R for all users, click Install; otherwise, click Change Install Location to pick a different location.



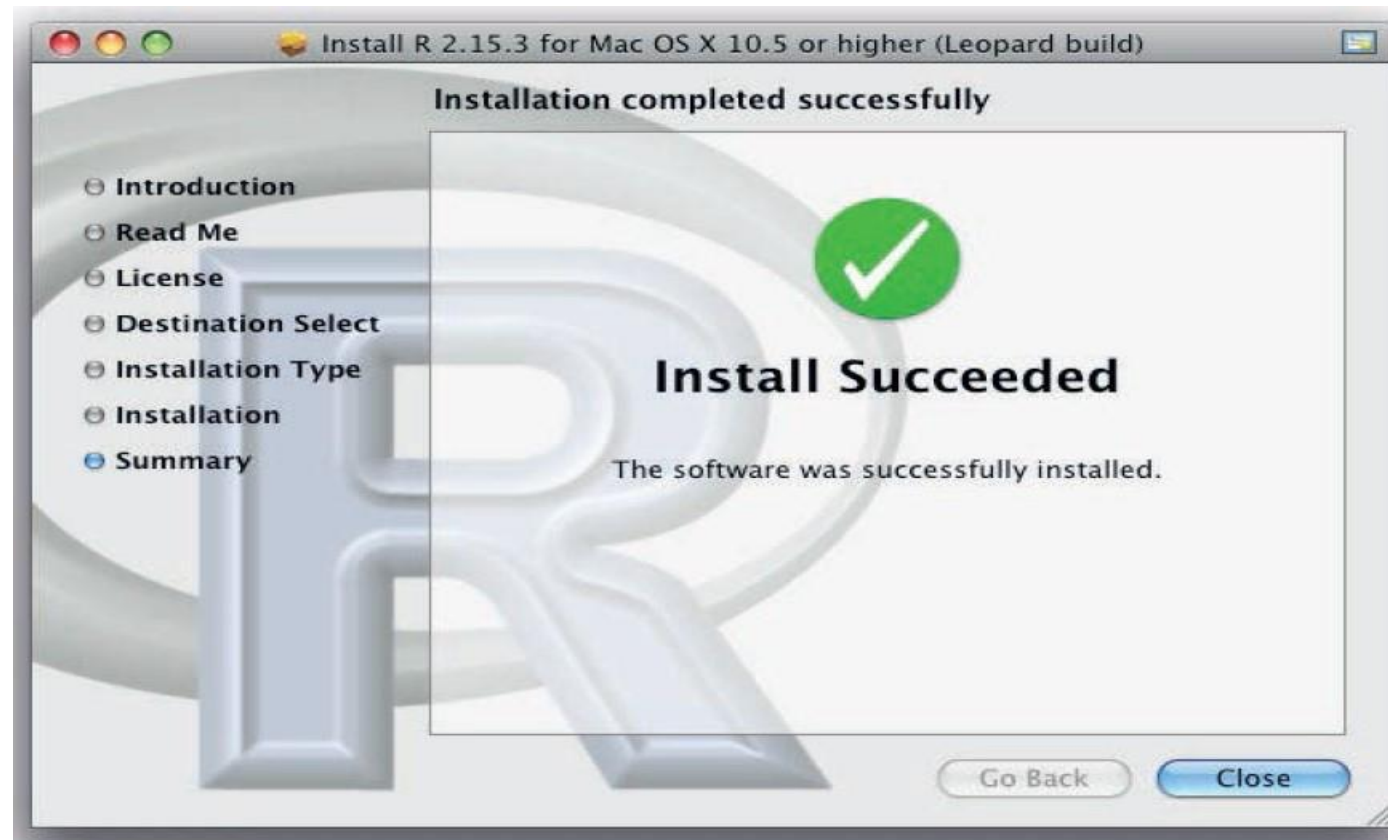
- ☐ If prompted, enter the necessary details like name & password and then click OK.



❑ This starts the installation process, which displays a progress bar.

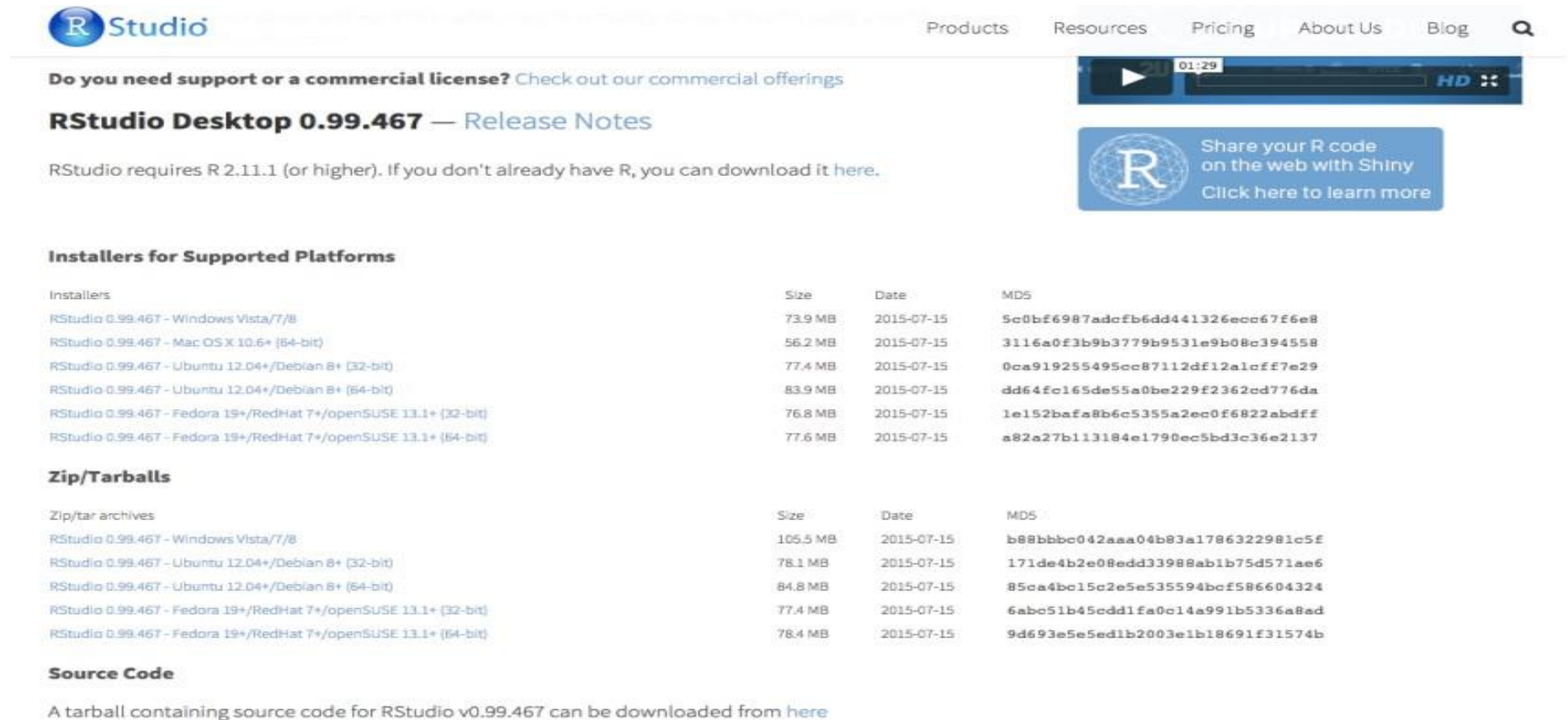


❑ When done, the installer signals show that the installation is completed successfully. Now click “Close” to finish the installation.



Step by Step procedure for Installing R Studio

- Step1: We visit the R Studio official site and click on **Download R Studio**.
<https://www.rstudio.com/products/rstudio/download/>



RStudio Products Resources Pricing About Us Blog

Do you need support or a commercial license? Check out our commercial offerings

RStudio Desktop 0.99.467 — Release Notes

RStudio requires R 2.11.1 (or higher). If you don't already have R, you can download it here.

Share your R code on the web with Shiny. Click here to learn more

Installers for Supported Platforms

Installers	Size	Date	MDS
RStudio 0.99.467 - Windows Vista/7/8	73.9 MB	2015-07-15	5c0bf6987adcfb6dd441326ecc67f6e8
RStudio 0.99.467 - Mac OS X 10.6+ (64-bit)	56.2 MB	2015-07-15	3116a0f3b9b3779b9531e9b08c394558
RStudio 0.99.467 - Ubuntu 12.04+/Debian 8+ (32-bit)	77.4 MB	2015-07-15	0ca919255495cc87112df12a1cffe7e29
RStudio 0.99.467 - Ubuntu 12.04+/Debian 8+ (64-bit)	83.9 MB	2015-07-15	dd64fc165de55a0be229f2362cd776da
RStudio 0.99.467 - Fedora 19+/RedHat 7+/openSUSE 13.1+ (32-bit)	76.8 MB	2015-07-15	1e152bafa8b6c5355a2ec0f6822abdff
RStudio 0.99.467 - Fedora 19+/RedHat 7+/openSUSE 13.1+ (64-bit)	77.6 MB	2015-07-15	a82a27b113184e1790ec5bd3c36e2137

Zip/Tarballs

Zip/tar archives	Size	Date	MDS
RStudio 0.99.467 - Windows Vista/7/8	105.5 MB	2015-07-15	b88bbbc042aaa04b83a1786322981c5f
RStudio 0.99.467 - Ubuntu 12.04+/Debian 8+ (32-bit)	78.1 MB	2015-07-15	171de4b2e08edd33988ab1b75d571ae6
RStudio 0.99.467 - Ubuntu 12.04+/Debian 8+ (64-bit)	84.8 MB	2015-07-15	85ca4bc15c2e5e535594bcf586604324
RStudio 0.99.467 - Fedora 19+/RedHat 7+/openSUSE 13.1+ (32-bit)	77.4 MB	2015-07-15	6abc51b45cdd1fa0c14a991b5336a8ad
RStudio 0.99.467 - Fedora 19+/RedHat 7+/openSUSE 13.1+ (64-bit)	78.4 MB	2015-07-15	9d693e5e5ed1b2003e1b18691f31574b

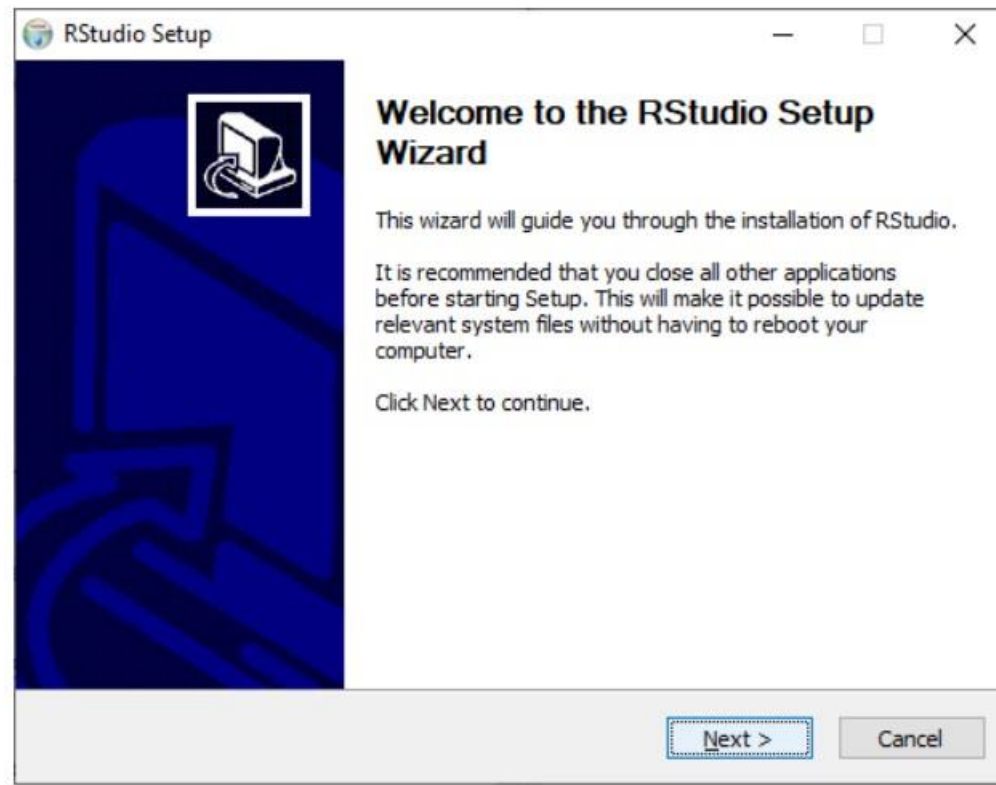
Source Code

A tarball containing source code for RStudio v0.99.467 can be downloaded from [here](#)

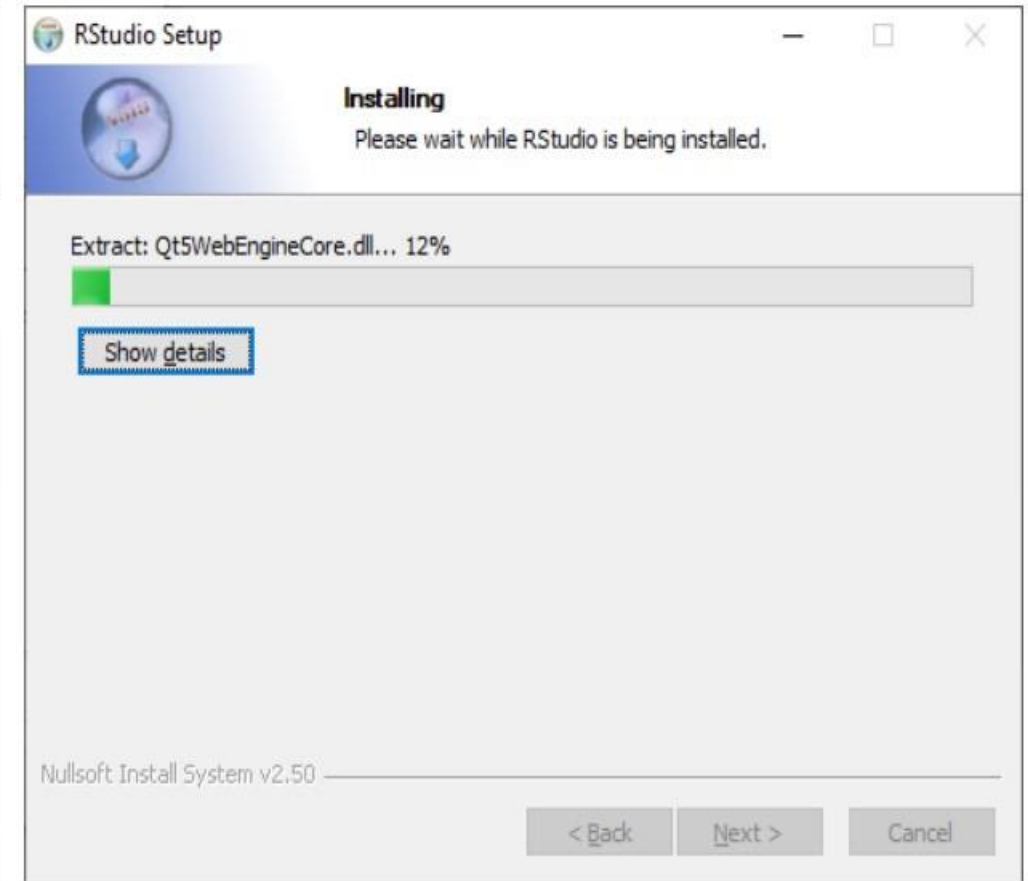
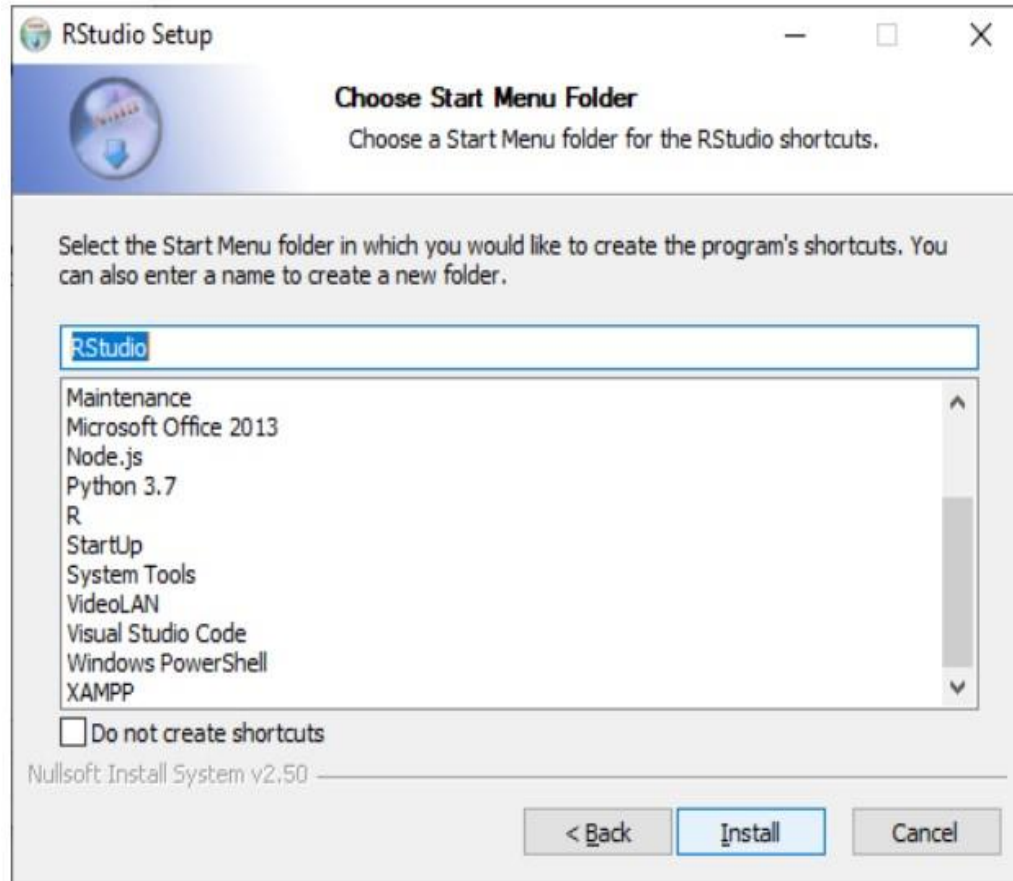
Step2: After downloading the R Studio setup, click on the downloaded R Studio setup. You can see a pop-up message whether to allow this or not. Now click on “Yes”.

Step3: In the next step, we will run our setup in the following way:

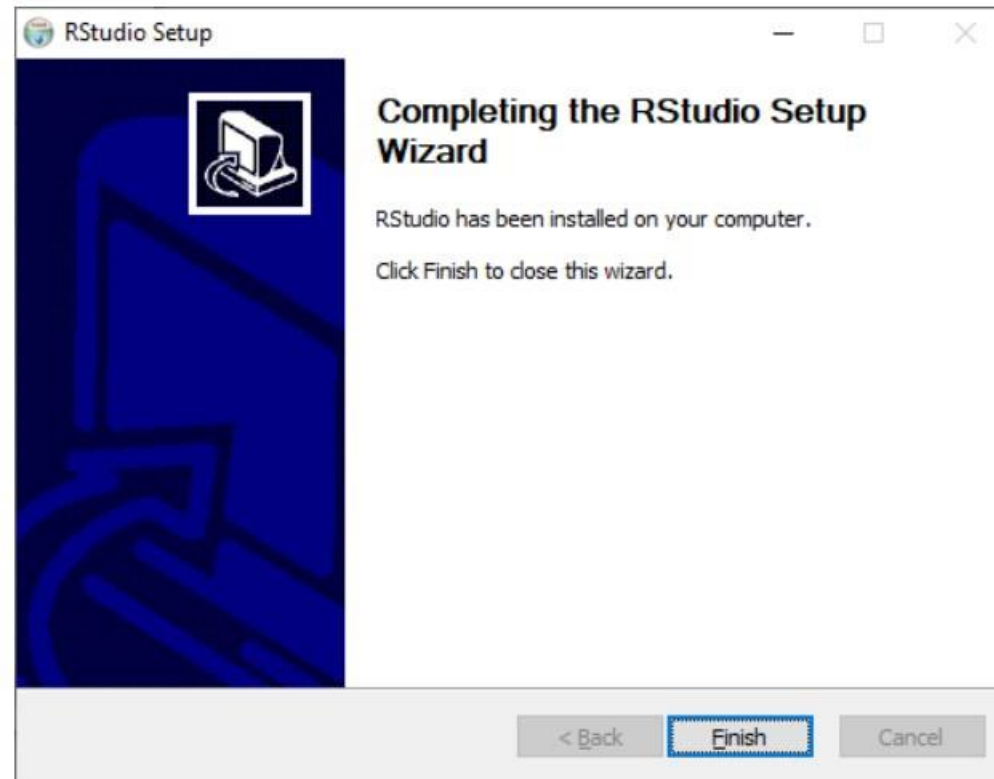
1) Click on Next.



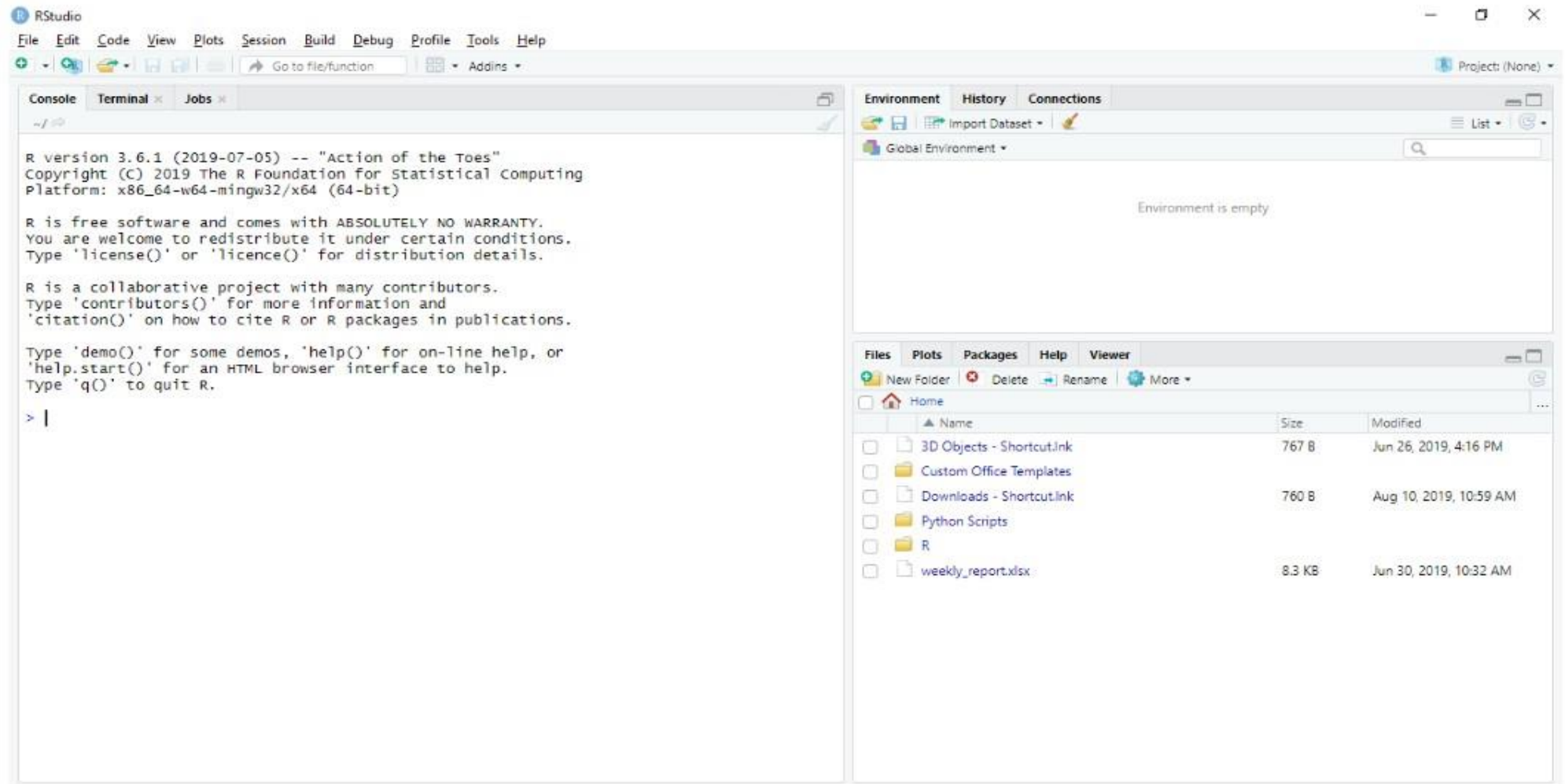
2) Click on Install.



3) Click on finish.



4) R Studio is ready to work.



The left side of the slide features a vertical decorative bar. It contains a large, stylized blue 'R' logo for the R programming language. The background of this bar is dark with glowing blue and green lines, suggesting data connections or a network. There are also some faint, glowing numbers and symbols scattered throughout.

Basic Concepts

A. Variables: The `<-` symbol is the assignment operator. Variables are an integral part of any programming language. A variable can take on any available data type.

❖ **Variable Assignment:** There are several ways to assign a value to a variable, and again, this does not depend on the type of value being assigned.

Eg. `X <- 10`

X

Basic Concepts

❖ **Removing Variables:** For various reasons, a variable may need to be removed. This is quickly done using **remove** or its shortcut **rm**.

E.g.,

```
a <- 1
```

```
b <- 2
```

```
c <- 3
```

```
print(a) # Output: 1
```

```
print(b) # Output: 2
```

```
print(c) # Output: 3
```

```
rm(a, b)
```

Try to print the variables again

```
print(a) # This will cause an error because 'a' is no longer defined
```

```
print(b) # This will cause an error because 'b' is no longer defined
```

```
print(c) # Output: 3 (still defined)
```




B. Basic Math: R is a statistical programming language that uses basic mathematical calculations. These follow the basic order of the operation: i.e., PEMDAS

P- Parenthesis


E- Exponent

M- Multiplication

D- Division

A- Addition

S- Subtraction


The background of the slide features a large, stylized blue 'R' logo on the left side. The background is dark with glowing blue and green lines and patterns, resembling a network or data visualization. There are also some faint, glowing numbers and symbols scattered throughout.

C. R Objects/ Data types: There are numerous data types in R that store various kinds of data. The five main types of data used are:

- numeric
- integer
- character (string)
- Date(time-based) and
- logical (TRUE/FALSE).

D. Data Structures: A number of data structures exist in R. The most important data structures are:

- **vector:** the most important data structure that is essentially a collection of elements.
- **matrix:** A table-like structure with rows and columns
- **data frame:** A tabular data structure to perform statistical operations
- **lists:** A collection that can hold a combination of data types.
- **factors:** to represent categorical data

The left side of the slide features a vertical decorative strip. It contains a large, stylized blue 'R' logo for the R programming language. The background of this strip is dark with glowing green and blue lines, suggesting data connections or network paths. There are also some faint, glowing numbers and symbols scattered throughout.

E. Attributes: R objects can have attributes, which are like metadata for the object. These metadata can be very useful in that they help to describe the object. For example, column names on a data frame help to tell us what data are contained in each of the columns.

- names, dimnames
- dimensions (e.g. matrices, arrays)
- class (e.g. integer, numeric)
- length
- other user-defined attributes/metadata.

The image features a large, stylized blue 'R' logo on the left side, set against a dark background with glowing blue lines and numbers. The 'R' is partially obscured by a green circular graphic element. The background also contains faint, glowing numbers and lines, suggesting a data or network theme.

F. Creating Vectors: The `c()` function can be used to create vectors of objects by concatenating things together.

- `x<-c(0.25, 0.6) #numeric`
- `x<-c(TRUE, FALSE) #logical`
- `x<-c(T, F) #logical`
- `x<-c("A", "B", "C") #character`
- `x<-9:29 #integer`
- `x<-c(1+0i, 2+4i) # complex`

G. Mixing Objects: There are occasions when different classes of R objects get mixed together.

- `x<-c(1.7, "a") #character`
- `x<-c(TRUE, 2)`
- `x<-c("a", TRUE)`




R Operators

❖ **Operators** are the symbols directing the compiler to perform various operations between the operands. R supports four kinds of binary operators between a set of operands—the different types of **operators in R Programming language** and their usage.

Types of the operator in R language

- Arithmetic Operators
- Logical Operators
- Relational Operators
- Assignment Operators
- Miscellaneous Operator

- 
- The image features a large, stylized blue 'R' logo on the left side, set against a dark background with glowing blue and green lines and numbers, suggesting a technical or data-driven theme.
- **Arithmetic Operators:** Arithmetic operations simulate various math operations, like addition, subtraction, multiplication, division, and modulo, using the specified operator between operands, which may be either scalar values, complex numbers, or vectors.

Addition (+):

Input: `x<-c(2, 6.1)`

`y<-c(2.33, 8)`

`print(x+y)`

Output: 4.33, 14.1

Subtraction (-): The second operand values are subtracted from the first.

Input: $x < -6.1$

 $y < -8.5$

```
print(x-y)
```

Output:

Multiplication (*): The multiplication of corresponding elements of vectors and Integers are multiplied with the use of '*' operator.

Input: a<-12

b<-12

```
print(a*b)
```

Output:144


Division (/): The first operand is divided by the second operand with the use of the ‘/’ operator.


Input: a<-12


$$b < -4$$

```
print(a/b)
```

Output: 3

- 
- **Logical Operators:** Logical operations are element-wise decision operations, based on the specified operator between the operands, which are then evaluated to either a True or False Boolean value.
 - 1. **Element-wise Logical AND operator (&):** Returns True if both the operands are True.
 - 2. **Element-wise Logical OR operator (|):** Returns True if either of the operands is True.
 - 3. **NOT operator (!):** A unary operator that negates the status of the elements of the operand.
 - 4. **Logical AND operator (&&):** Returns True if both the first elements of the operands are True.
 - 5. **Logical OR operator (||):** Returns True if either of the first elements of the operands are True.

- 
- **Relational Operators:** The relational operators carry out comparison operations between the corresponding elements of the operands. Returns a Boolean TRUE value if the first operand satisfies the relation compared to the second.
 - 1. **Less than (<):** Returns TRUE if the corresponding element of the first operand is less than that of the second operand. Else returns FALSE.
 - 2. **Less than equal to(<=):** Returns TRUE if the corresponding element of the first operand is less than or equal to that of the second operand. Else returns FALSE.
 - 3. **Greater than (>):** Returns TRUE if the corresponding element of the first operand is greater than that of the second operand. Else returns FALSE.
 - 4. **Greater than equal to(>=):** Returns TRUE if the corresponding element of the first operand is greater than or equal to that of the second operand. Else returns FALSE.
 - 5. **Not equal to (!=):** Returns TRUE if the corresponding element of the first operand is not equal to the second operand. Else returns FALSE.

- 
- The image features a large, stylized blue 'R' logo on the left side, set against a dark background with glowing blue lines and numbers, suggesting a data or programming theme.
- **Assignment Operators:** Assignment operators are used to assigning values to various data objects in R. The objects may be integers, vectors, or functions. These values are then stored by the assigned variable names. There are two kinds of assignment operators:

1. **Left Assignment (<- or <<- or =):** Assigns a value to a vector.

Input: `vec1 = c("ab", TRUE) / vec1<- c("ab", TRUE)`

`print(vec1)`


Output: "ab" "TRUE"

2. **Right Assignment (-> or ->> or =):** Assigns a value to a vector.

Input: `c("ab", TRUE) = vec1 / c("ab", TRUE) ->vec1`

`print(vec1)`

Output: "ab" "TRUE"

- 
- The image features a large, stylized blue 'R' logo on the left side, set against a dark background with glowing blue and green lines and patterns, resembling a network or data visualization. The 'R' is partially obscured by a green circular shape.
- **Miscellaneous Operators:** These are the mixed operators that simulate the printing of sequences and assignment of vectors, either left or right-handed.

1. **%in% Operator:** Checks if an element belongs to a list and returns a Boolean value TRUE if the value is present else FALSE.

Example:

```
vector1 <- c(1, 2, 3, 4, 5)
vector2 <- c(3, 4, 5, 6, 7)
# Use %in% operator to check if elements of vector1 are in vector2
result <- vector1 %in% vector2
# Print the result print(result)
```

Output: [FALSE, FALSE, TRUE, TRUE, TRUE]

2. **Colon Operator(:):** Prints a list of elements starting with the element before the colon to the element after it.

Example: print (1:5)

Output: 1 2 3 4 5

3. %*% Operator: This operator is used to multiply a matrix with its transpose. Transpose of the matrix is obtained by interchanging the rows to columns and columns to rows. The number of columns of the first matrix must be equal to the number of rows of the second matrix. Multiplication of the matrix A with its transpose, B, produces a square matrix.

Example: `mat=matrix(c(1,2,3,4,5,6),nrow=2,ncol=3)` # Creating matrix
`print(mat)` # Matrix of order 2x3
`print(t(mat))` # Transpose matrix of order 3x2
`product=mat %*% t(mat)` # Applying Miscellaneous operator
`print(product)` # Product matrix of order 2x2

Output:

```
> print(mat)
      [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> print(t(mat))
      [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
> pro = mat %*% t(mat)
> print(pro)
      [,1] [,2]
[1,]   35   44
[2,]   44   56
```