
INTRODUCCION A R CON RSTUDIO

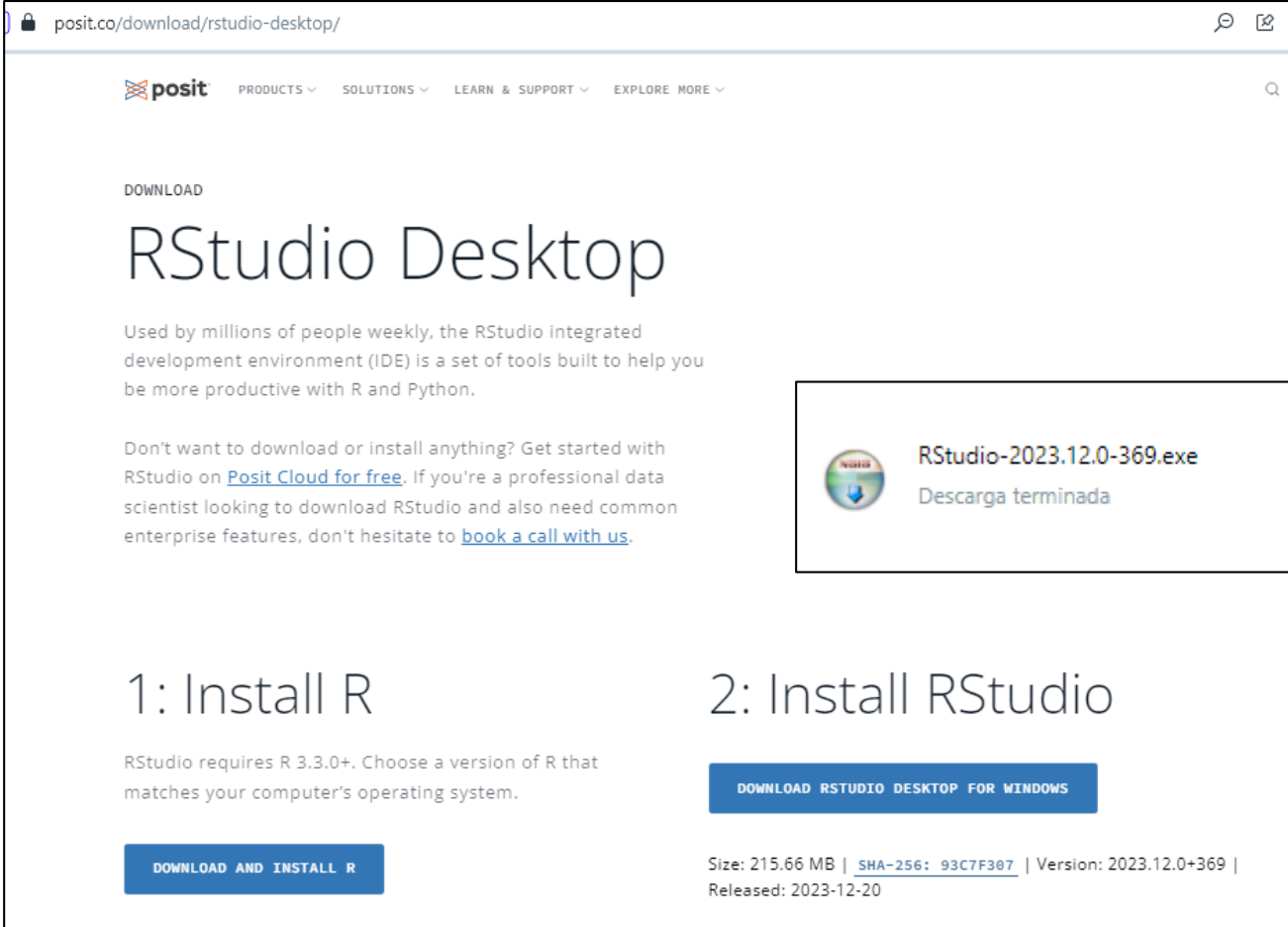
EDUARD LARA

1. INDICE

1. Instalación de R y RStudio
2. Introducción a Rstudio
3. Operaciones aritméticas
4. Variables
5. Tipos de datos
6. Vectores
7. Operadores de comparación

1. INSTALACION DE RSTUDIO

Paso 1. Vamos a la pagina <https://posit.co/downloads/>
Descargamos RStudio IDE



posit PRODUCTS SOLUTIONS LEARN & SUPPORT EXPLORE MORE

DOWNLOAD

RStudio Desktop

Used by millions of people weekly, the RStudio integrated development environment (IDE) is a set of tools built to help you be more productive with R and Python.

Don't want to download or install anything? Get started with RStudio on [Posit Cloud for free](#). If you're a professional data scientist looking to download RStudio and also need common enterprise features, don't hesitate to [book a call with us](#).

1: Install R

RStudio requires R 3.3.0+. Choose a version of R that matches your computer's operating system.

[DOWNLOAD AND INSTALL R](#)

2: Install RStudio

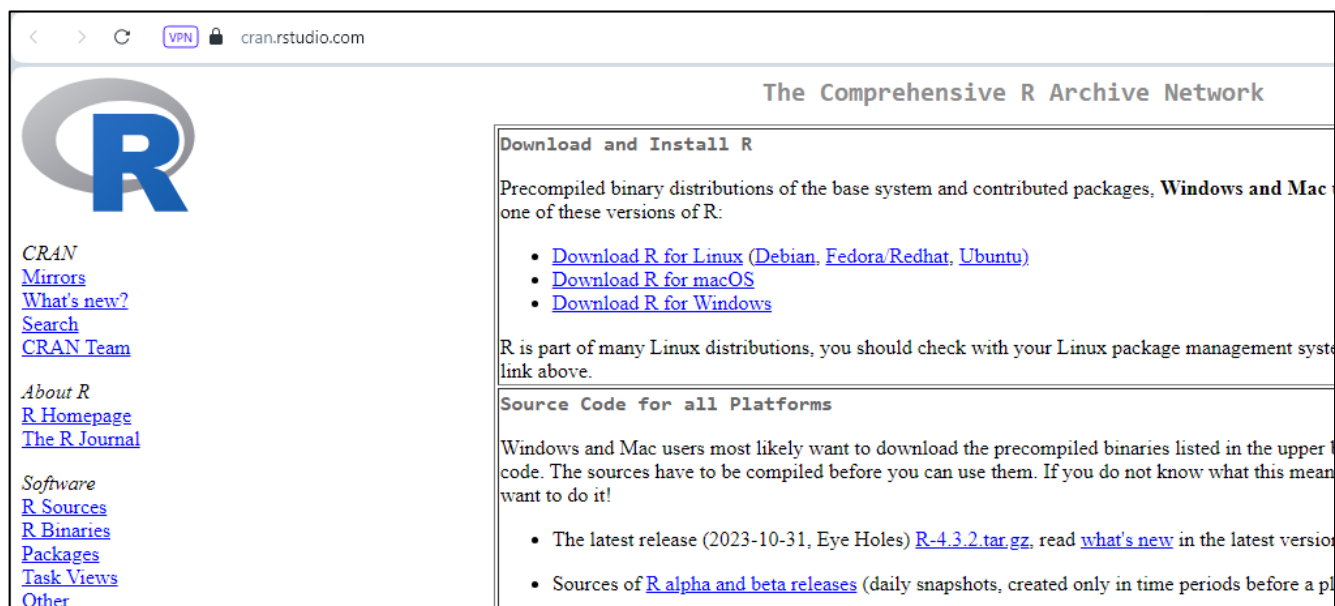
[DOWNLOAD RSTUDIO DESKTOP FOR WINDOWS](#)

Size: 215.66 MB | [SHA-256: 93C7F307](#) | Version: 2023.12.0+369 | Released: 2023-12-20

RStudio-2023.12.0-369.exe
Descarga terminada

1. INSTALACION DE RSTUDIO

Paso 2. Si hacemos click para descargar R, el lenguaje de programación que va a utilizar Rstudio, nos lleva a a pagina web <https://cran.rstudio.com>



1. INSTALACION DE RSTUDIO

Paso 3. Aquí descargamos R, en nuestro caso R para windows

R-4.3.2 for Windows

[Download R-4.3.2 for Windows](#) (79 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 from [here](#).

If you want to double-check that the package you have downloaded matches the package distributed by CRAN, you can compare the [md5](#) the [fingerprint](#) on the master server.

Frequently asked questions

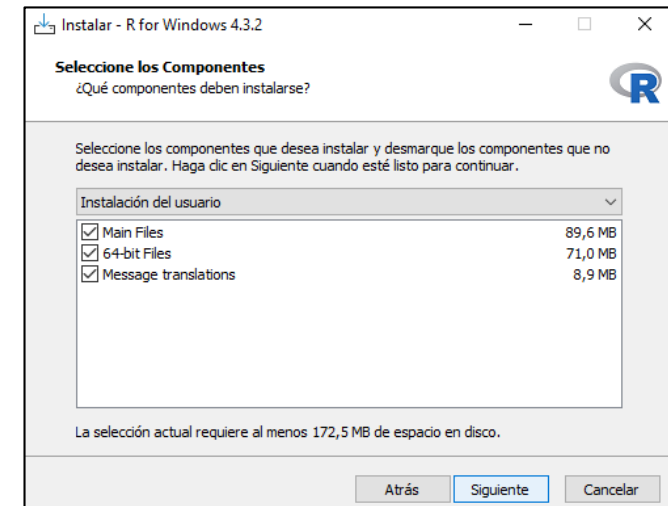
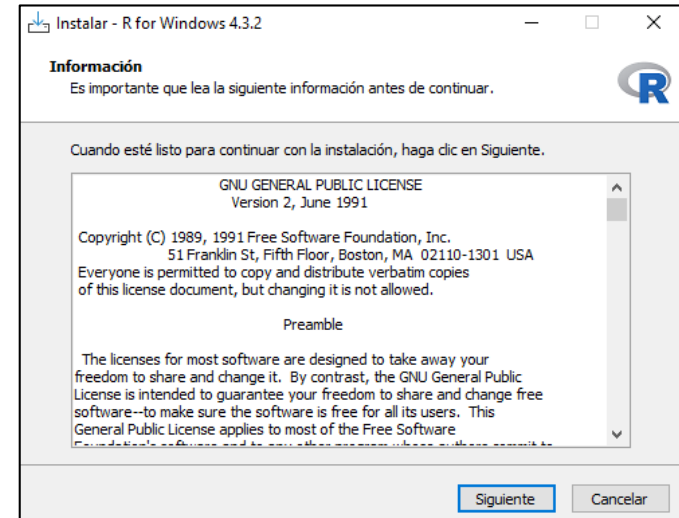
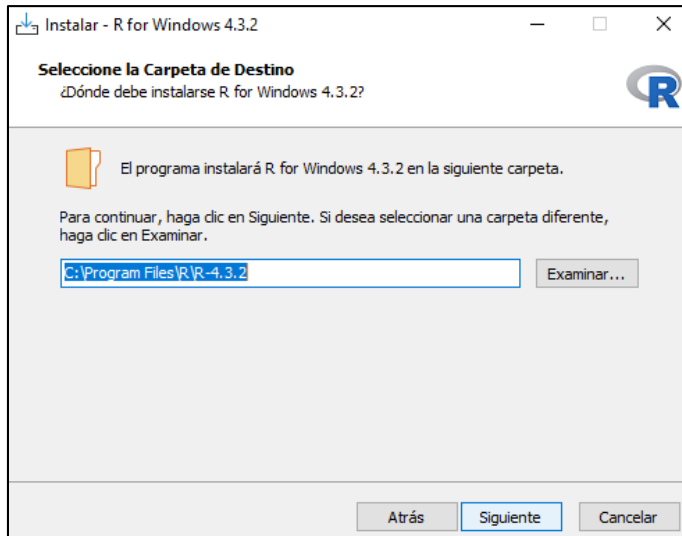
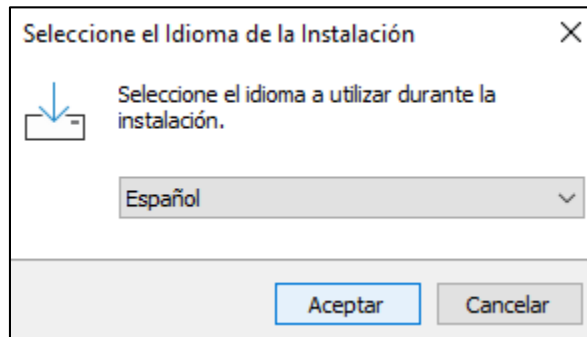
- [Does R run under my version of Windows?](#)
- [How do I update packages in my previous version of R?](#)



R-4.3.2-win.exe
Descarga terminada

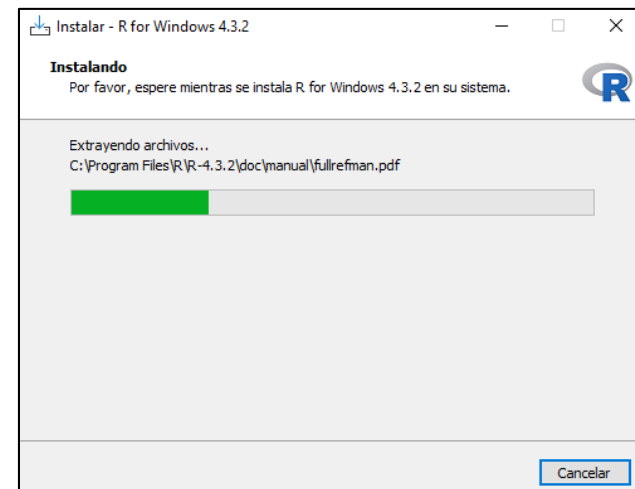
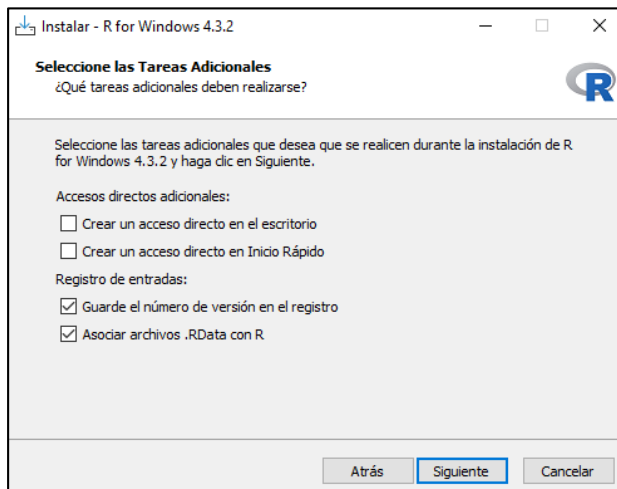
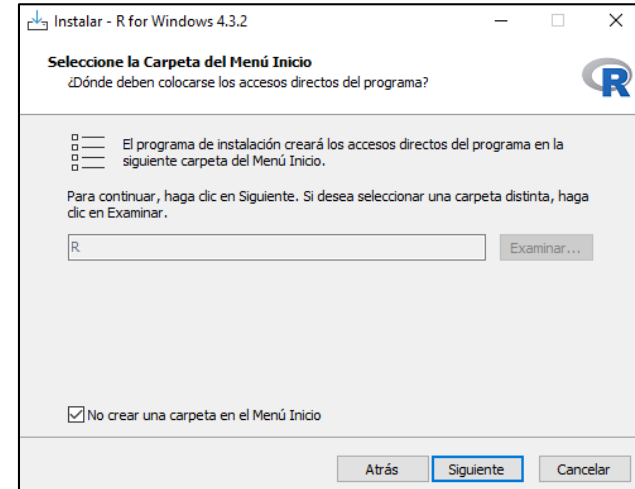
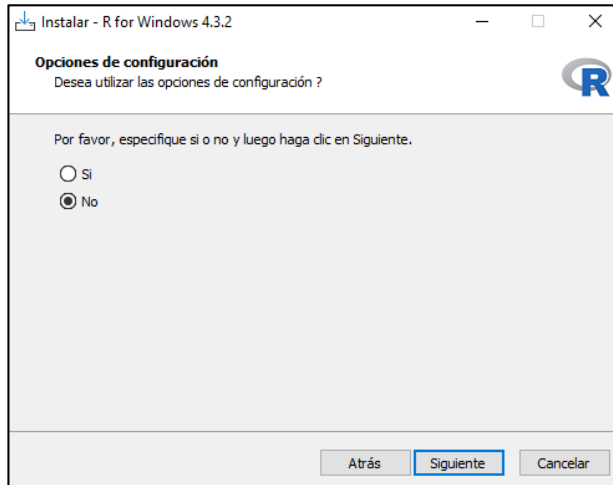
1. INSTALACION DE RSTUDIO

Paso 4. Empezamos la instalación de R



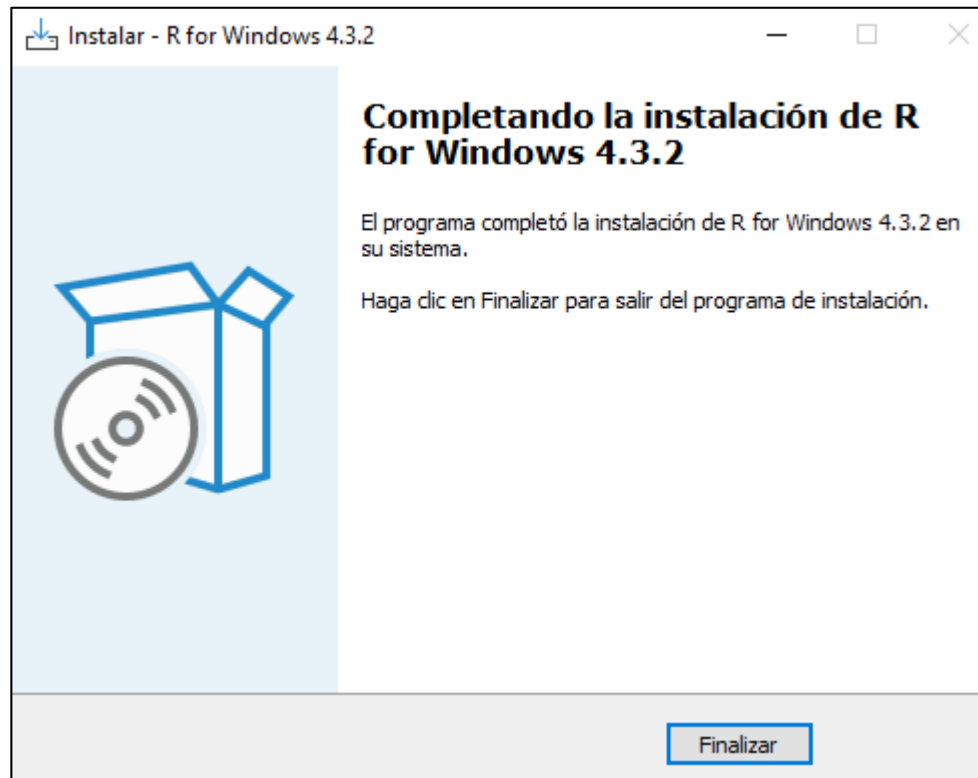
1. INSTALACION DE RSTUDIO

Paso 5. Seguimos con la instalación de R



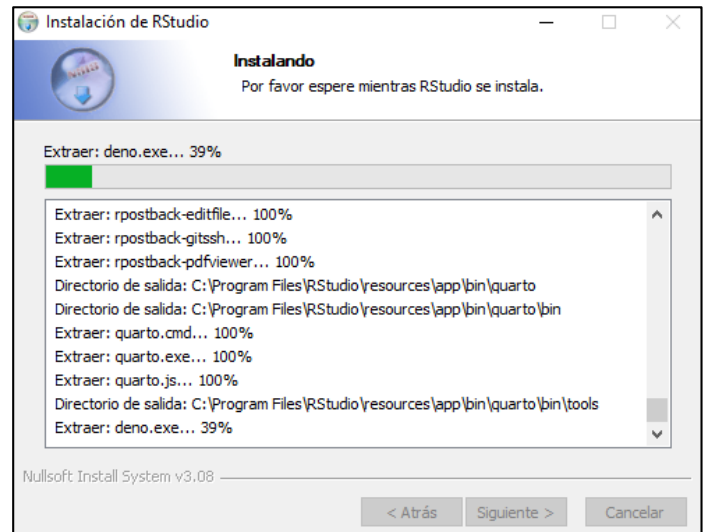
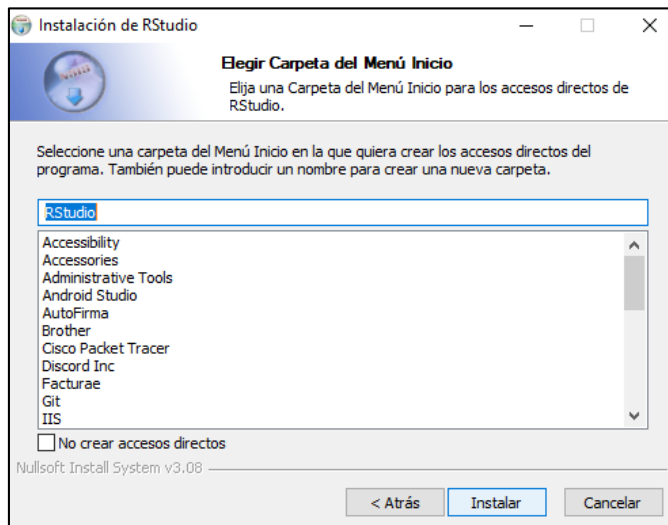
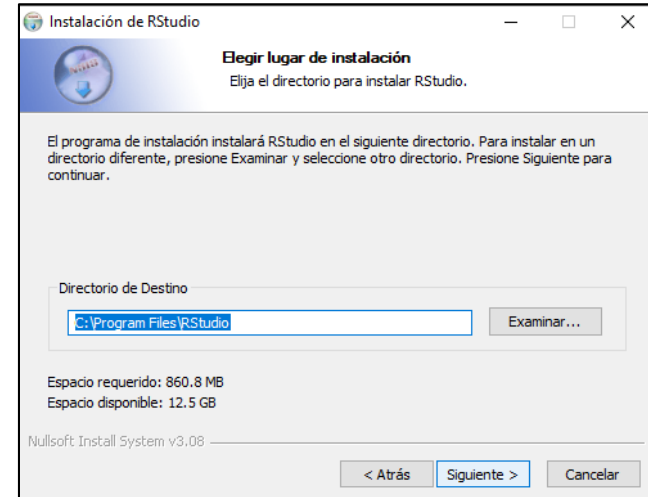
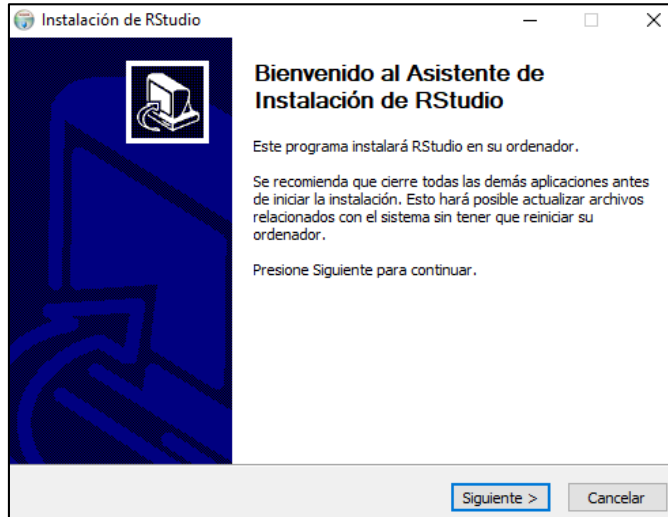
1. INSTALACION DE RSTUDIO

Paso 6. Finaliza la instalación de R, que nos instalará el lenguaje de programación y las librerías necesarias para ejecutar programas en R nuestro ordenador



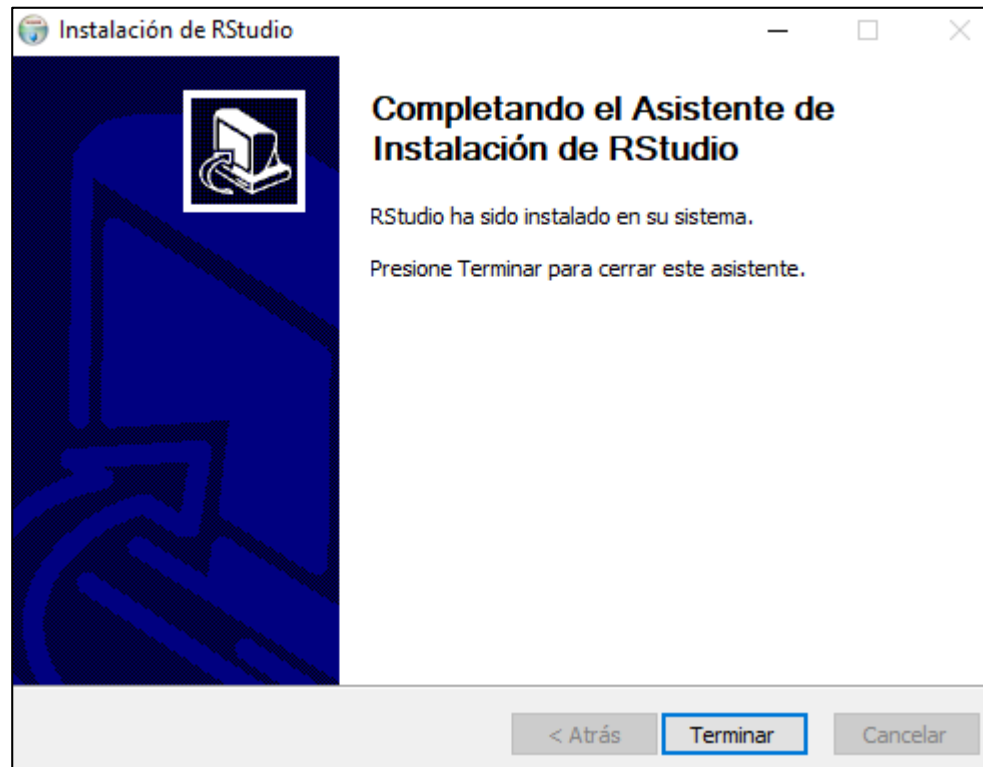
1. INSTALACION DE RSTUDIO

Paso 7. A continuación instalamos RStudio



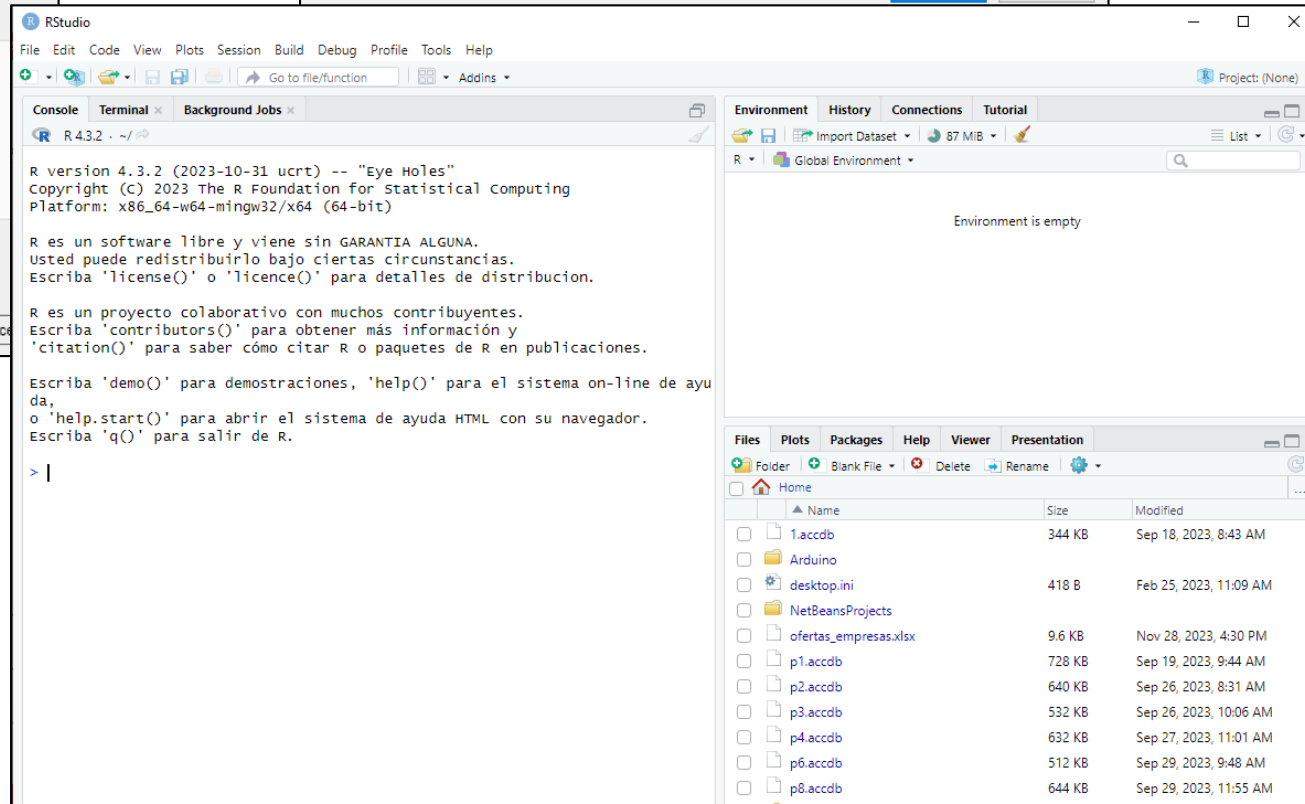
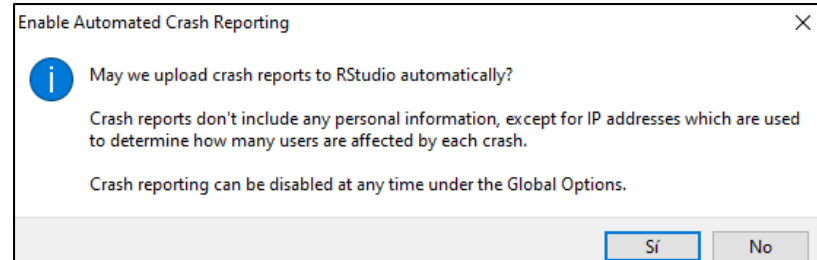
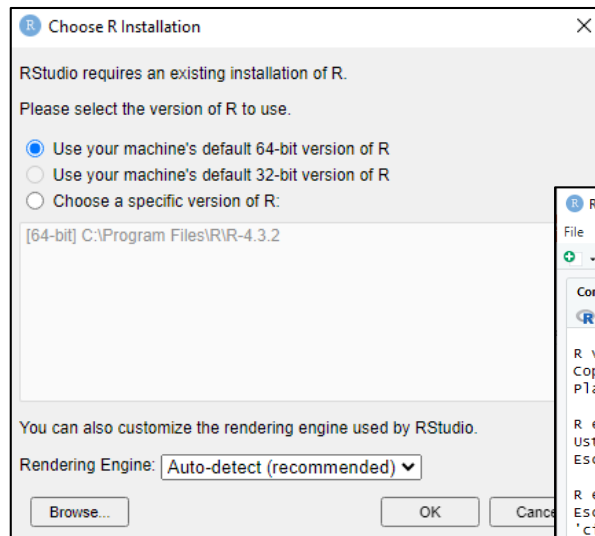
1. INSTALACION DE RSTUDIO

Paso 8. Finalizamos la instalación de RStudio Desktop para utilizar el lenguaje de programación R y hacer programas en R.



1. INSTALACION DE RSTUDIO

Paso 9. Una vez que ya tenemos instalado R y RStudio, iniciamos RStudio.



2. INTRODUCCION A RSTUDIO

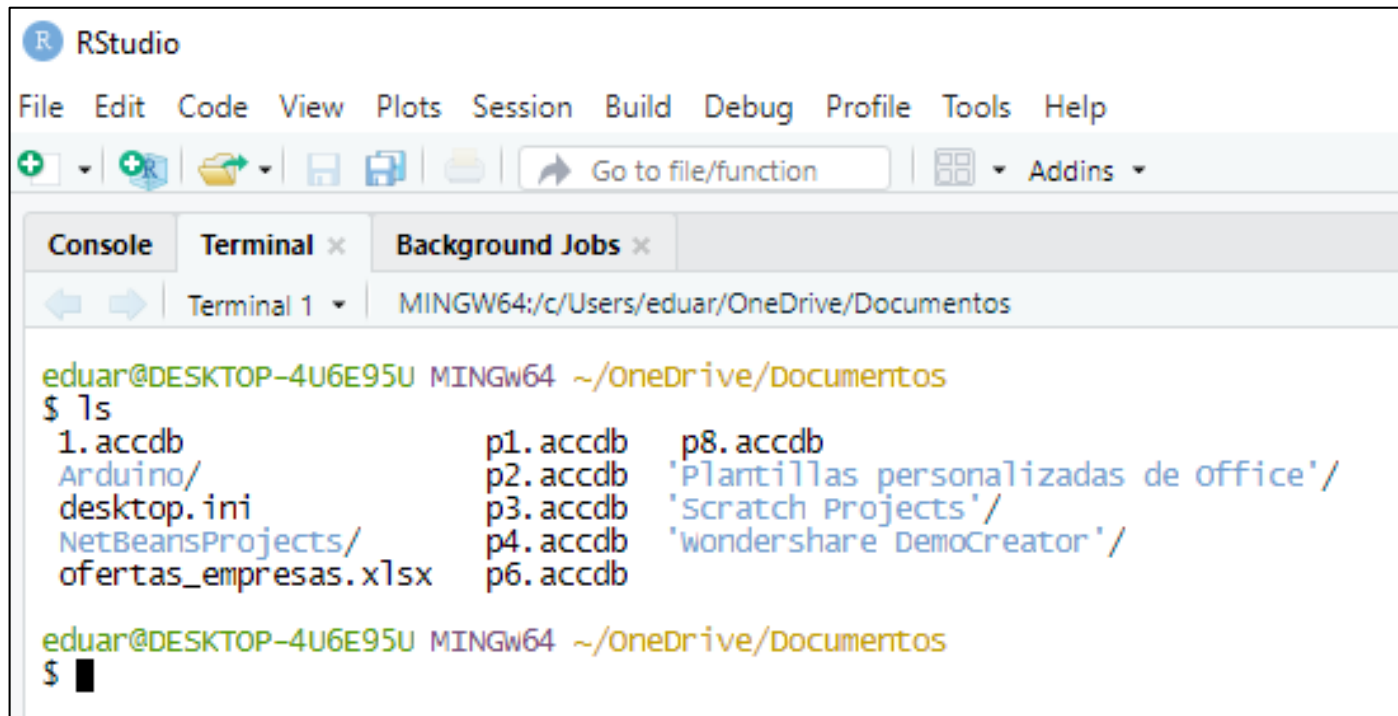
Paso 1. Al abrir RStudio, vemos la pestaña de consola donde podemos introducir directamente expresiones en R. Por ejemplo dentro de console podemos imprimir una línea mediante la función print:

```
> print(hola que tal estas)
Error: unexpected symbol en "print(hola que"
> print("hola que tal estas")
[1] "hola que tal estas"
>
```

Cuando le damos a Enter ejecuta la línea, es decir imprime muestra por pantalla la ejecución del comando R, en este caso el texto que hemos puesto.

2. INTRODUCCION A RSTUDIO

Paso 2. En la pestaña terminal, tenemos la línea de comandos de nuestro ordenador en forma de Linux, que nos permite tener acceso a todos nuestros ficheros.



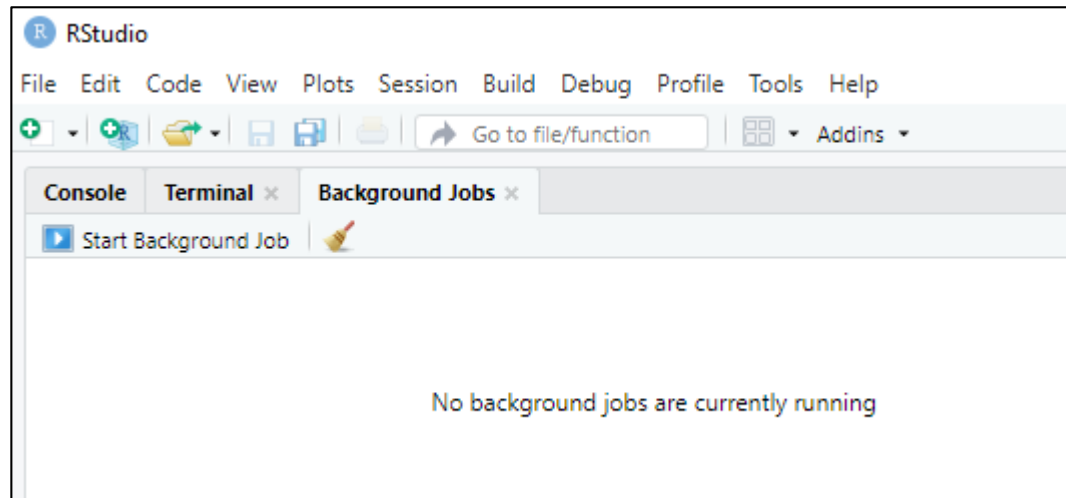
The screenshot shows the RStudio interface with the Terminal pane active. The terminal window title is 'Terminal 1' and the path is 'MINGW64:/c/Users/eduar/OneDrive/Documentos'. The user 'eduar@DESKTOP-4U6E95U' has executed the command 'ls', resulting in a list of files and directories. The output is as follows:

```
eduar@DESKTOP-4U6E95U MINGW64 ~/OneDrive/Documentos
$ ls
1. accdb          p1. accdb      p8. accdb
Arduino/          p2. accdb      'Plantillas personalizadas de office'/
desktop.ini       p3. accdb      'Scratch Projects'/
NetBeansProjects/ p4. accdb      'wondershare DemoCreator'/
ofertas_empresas.xlsx p6. accdb
```

The terminal prompt '\$' is visible at the bottom, indicating the command is ready to be entered.

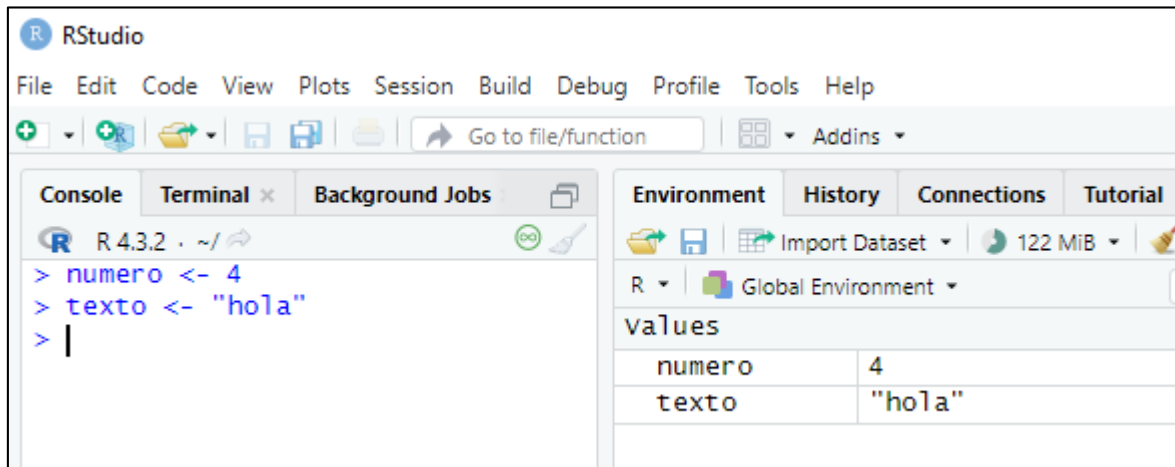
2. INTRODUCCION A RSTUDIO

Paso 3. En la pestaña Background Jobs permite ver los programas que se están ejecutando en background



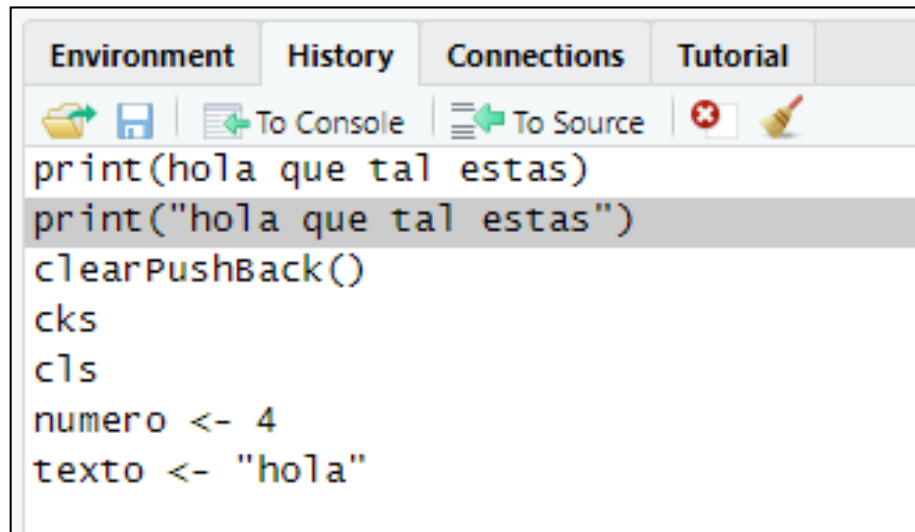
2. INTRODUCCION A RSTUDIO

Paso 4. En Environment se muestra todos los objetos o todas las variables que vayamos creando en la consola o en la ejecución de nuestros programas. Va almacenando los objetos con sus valores. Si en Console creamos dos variables (una numérica y otra texto) se pueden ver en la ventana environment



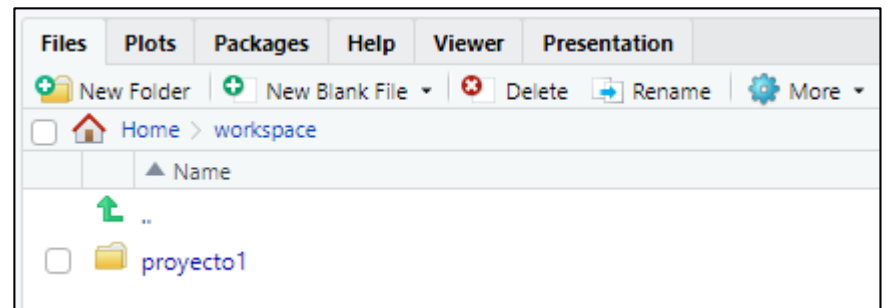
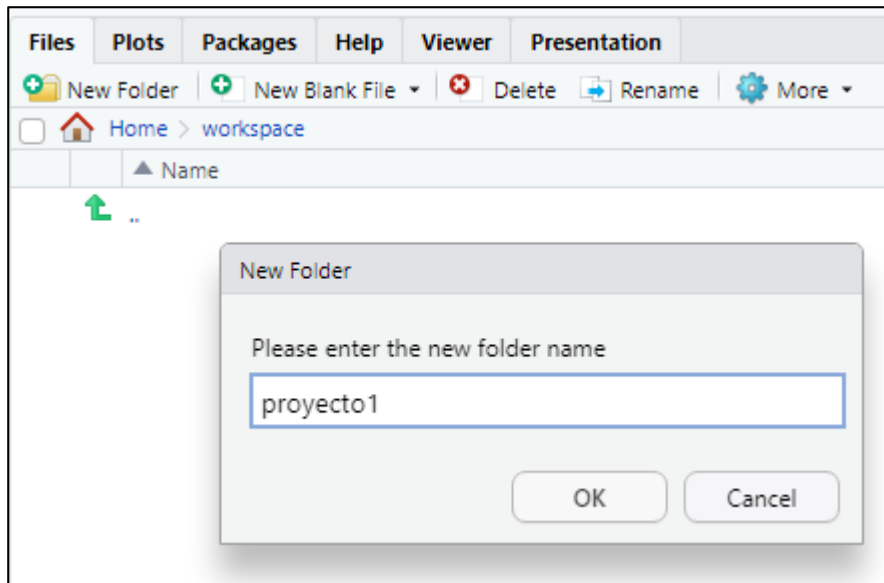
2. INTRODUCCION A RSTUDIO

Paso 5. En la pestaña History muestra todas las ejecuciones realizadas, por si queremos repetir alguna. Basta con solo posicionarse encima de esa instrucción y hacer dobleclick



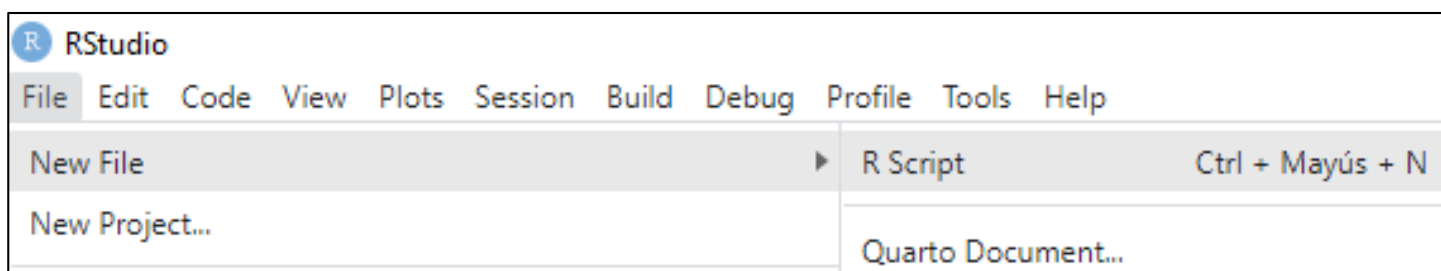
2. INTRODUCCION A RSTUDIO

Paso 6. En la pestaña Files aparecen los ficheros de nuestro directorio de trabajo. Crearemos una carpeta workspace donde iremos almacenamos todos nuestros proyectos. Dentro crearemos otra carpeta llamada proyecto1.

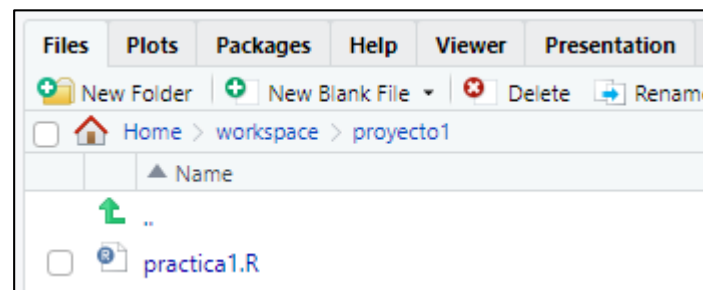
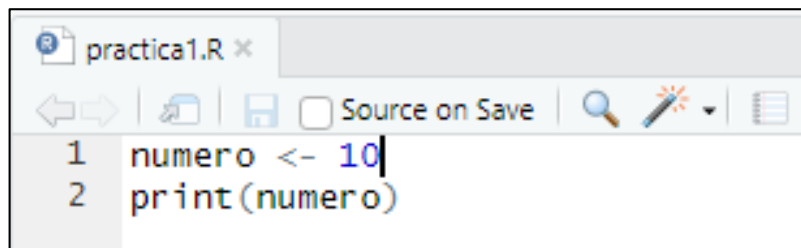


2. INTRODUCCION A RSTUDIO

Paso 7. En la carpeta proyecto1 crearemos un fichero. Vamos a File y creamos un nuevo script R

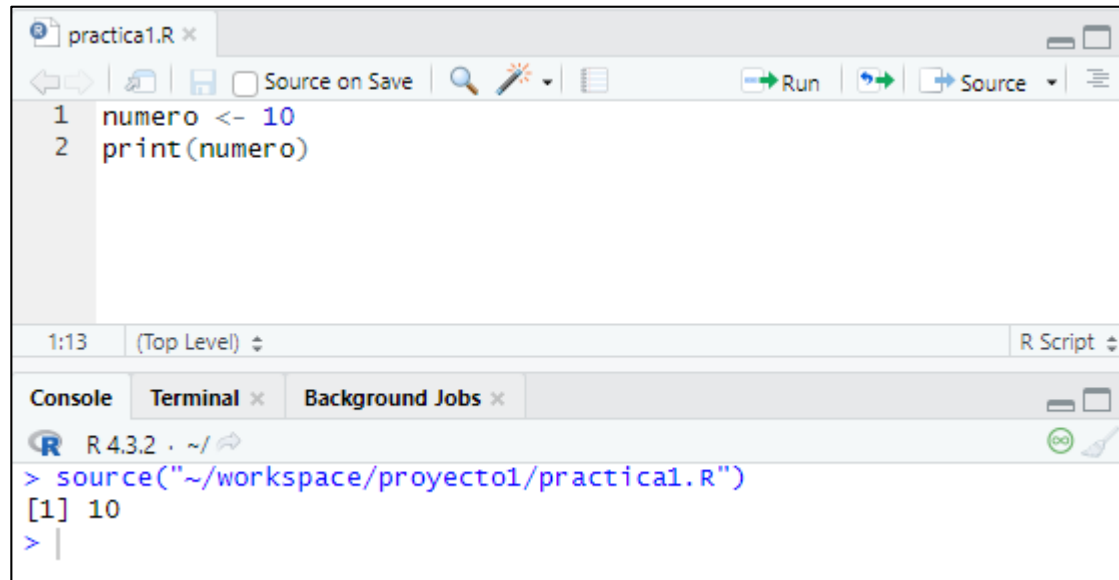


En el fichero, a la variable numero le asignamos un 10, y después la imprimimos. En File/Save As... guardamos el fichero en la carpeta proyecto1 con el nombre practica1.R



2. INTRODUCCION A RSTUDIO

Paso 8. Para ejecutar el fichero practica1.R vamos al botón Source del script y se ejecuta su contenido en la pantalla de console



The screenshot shows the RStudio interface. The top pane displays a script file named 'practica1.R' with two lines of code: `1 numero <- 10` and `2 print(numero)`. The toolbar includes buttons for 'Run' and 'Source'. The bottom pane shows the 'Console' tab with the following output: `> source("~/workspace/proyecto1/practica1.R")`, `[1] 10`, and a prompt `> |`.

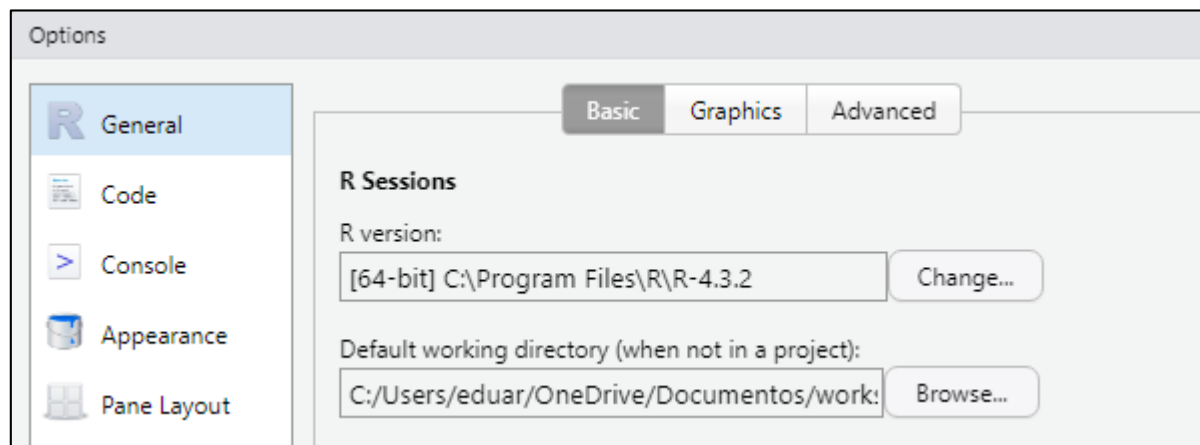
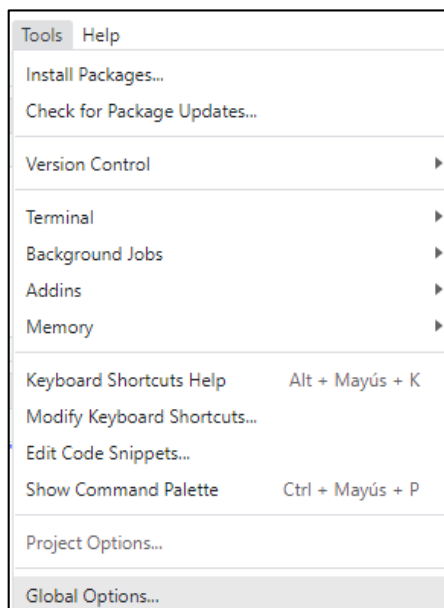
```
1 numero <- 10
2 print(numero)
```

```
> source("~/workspace/proyecto1/practica1.R")
[1] 10
> |
```

Imprime el valor 10, que es el valor asignado a la variable `numero`, que es la que se imprime

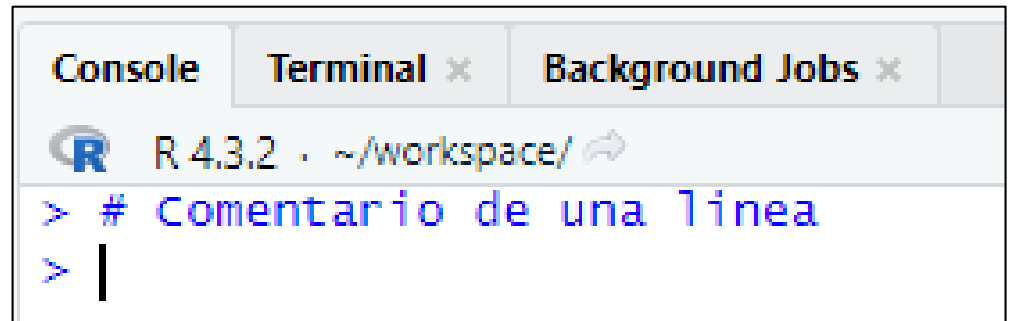
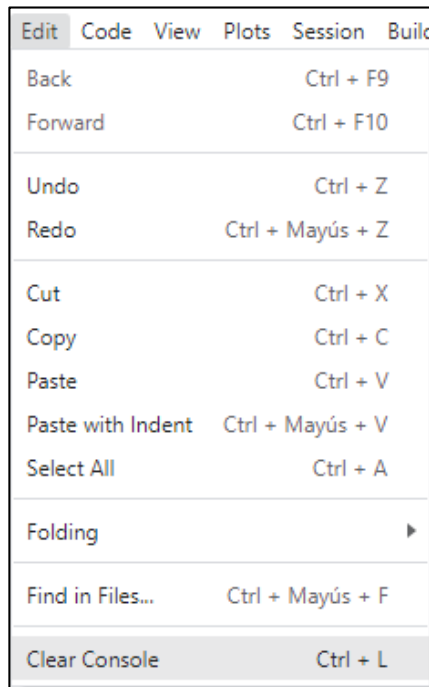
2. INTRODUCCION A RSTUDIO

Paso 9. Si queremos que cuando entremos en RStudio, nos aparezca por defecto nuestro workspace (donde vamos a ir creando nuestros proyectos), vamos Tools/Global Options/ Pestaña General. Aquí configuraremos la carpeta de trabajo que queremos que nos abra en Files siempre que entremos en RStudio



2. INTRODUCCION A RSTUDIO

Paso 10. Si queremos borrar la consola lo podemos hacer yendo a la opción de menú Edit/Clear console o mediante la combinación de letras Ctrt+L



Paso 11. Para poner comentarios en R se usa la almohadilla

3. OPERACIONES ARITMETICAS

Paso 1. Vamos a ver las operaciones aritméticas, realizar una suma, una resta, una multiplicación o división.

```
Console Terminal x Background Jobs x
R 4.3.2 · ~/workspace/
> 5+2
[1] 7
> 3-4
[1] -1
> 100-200+500
[1] 400
> (200-300)-(400-300)
[1] -200
> |
```

Sumas y restas con paréntesis

```
Console Terminal x Background Jobs x
R 4.3.2 · ~/workspace/
> 5*4
[1] 20
> 3*2*4
[1] 24
> 2*3/4
[1] 1.5
> 3*4*5
[1] 60
>
```

Multiplicaciones y divisiones

3. OPERACIONES ARITMETICAS

Paso 2. Para operaciones con potencias se utiliza el gorrito ^ mediante shift+tecla símbolo.

```
Console Terminal x Background Jobs x
R 4.3.2 · ~/workspace/ ↗
> 2^2
[1] 4
> 10^3
[1] 1000
> 10^(2*3)
[1] 1e+06
> |
```

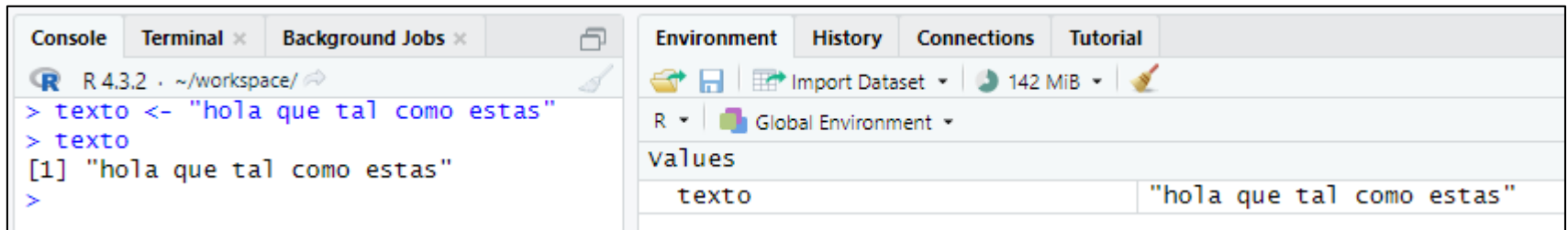
Potencia de un numero
mediante el símbolo ^

```
Console Terminal x Background Jobs x
R 4.3.2 · ~/workspace/ ↗
> 10/3
[1] 3.333333
> 10%%3
[1] 1
> 10/5
[1] 2
> 10%%5
[1] 0
> |
```

El resto de una división entera
se hace mediante %%

4. VARIABLES EN R

Paso 1. Para crear una variable tipo cadena, basta con asignar a una variable un texto entrecomillado



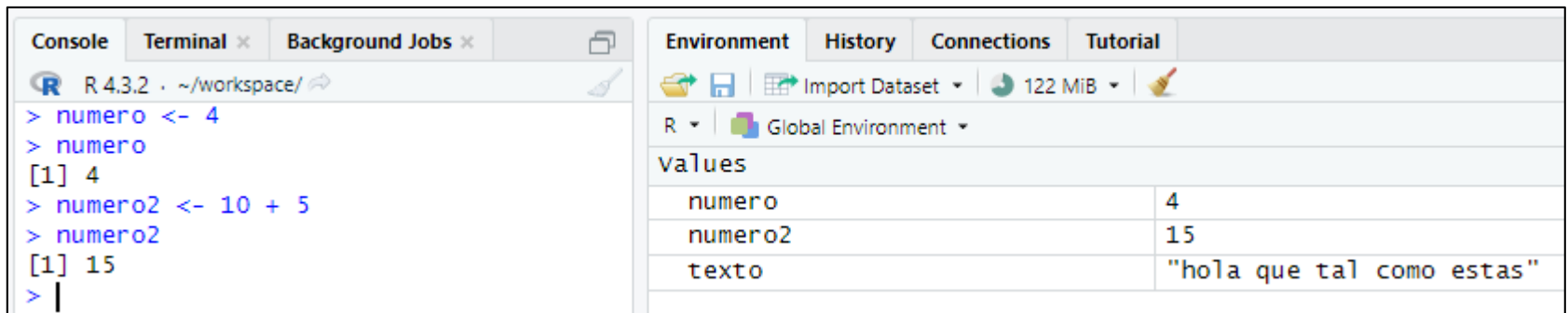
The screenshot shows the R Studio interface. The Console on the left contains the following commands and output:

```
R 4.3.2 . ~/workspace/
> texto <- "hola que tal como estas"
> texto
[1] "hola que tal como estas"
>
```

The Environment pane on the right shows the 'Global Environment' with a table of values:

values	
texto	"hola que tal como estas"

Paso 2. Para crear una variable entera, le asignamos un valor numérico, o mediante una suma de dos números



The screenshot shows the R Studio interface. The Console on the left contains the following commands and output:

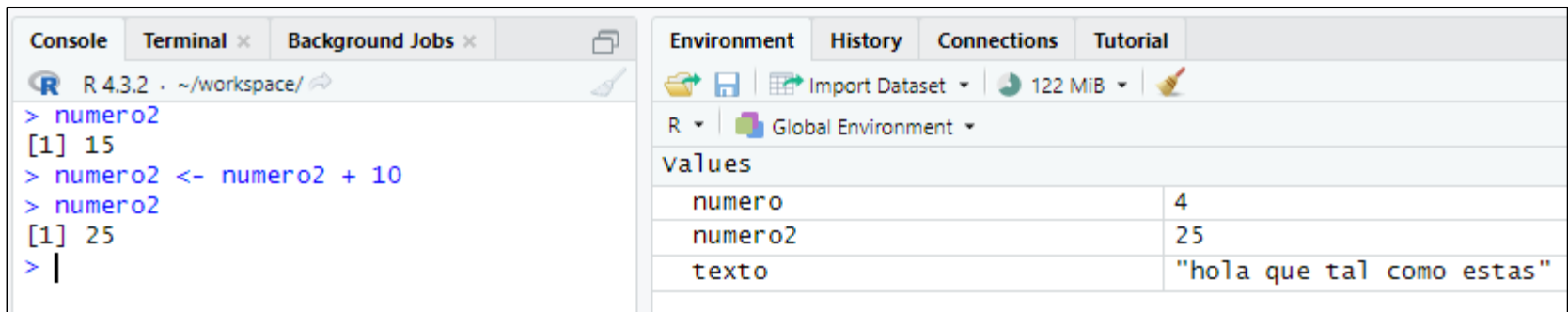
```
R 4.3.2 . ~/workspace/
> numero <- 4
> numero
[1] 4
> numero2 <- 10 + 5
> numero2
[1] 15
> |
```

The Environment pane on the right shows the 'Global Environment' with a table of values:

values	
numero	4
numero2	15
texto	"hola que tal como estas"

4. VARIABLES EN R

Paso 3. También podemos sumar el valor que ya tiene una variable más 10.



The screenshot shows the RStudio interface. The console on the left displays the following R code and output:

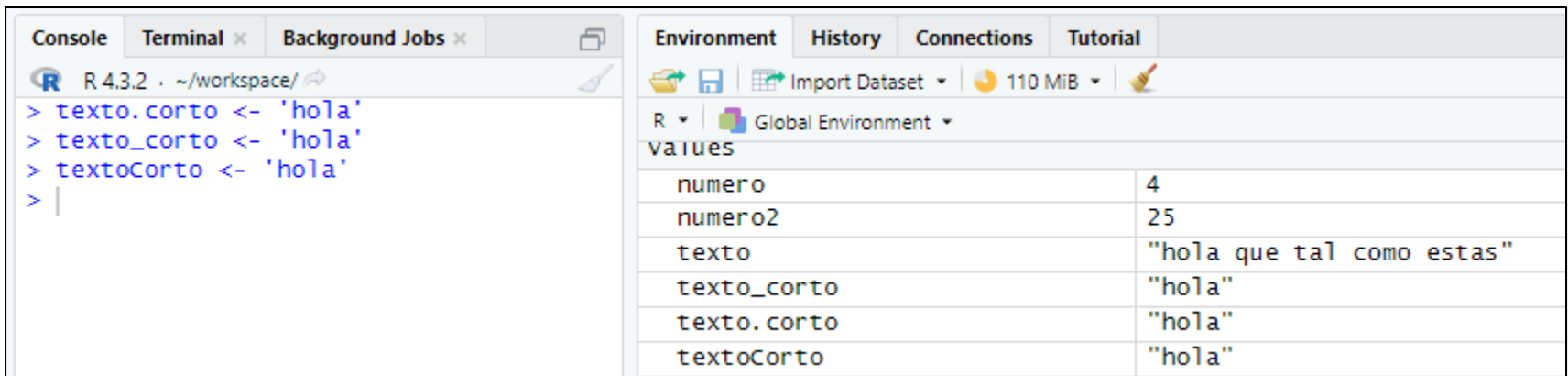
```
> numero2  
[1] 15  
> numero2 <- numero2 + 10  
> numero2  
[1] 25  
> |
```

The environment pane on the right shows the 'Global Environment' with the following variables:

values	
numero	4
numero2	25
texto	"hola que tal como estas"

4. VARIABLES EN R

Paso 4. El nombre de variables con nombres compuesto se puede hacer de las siguientes maneras:



The screenshot shows the R Studio interface. The console on the left displays the following commands:

```
R 4.3.2 . ~/workspace/  
> texto.corto <- 'hola'  
> texto_corto <- 'hola'  
> textoCorto <- 'hola'  
> |
```

The environment pane on the right shows the 'Global Environment' with the following variables:

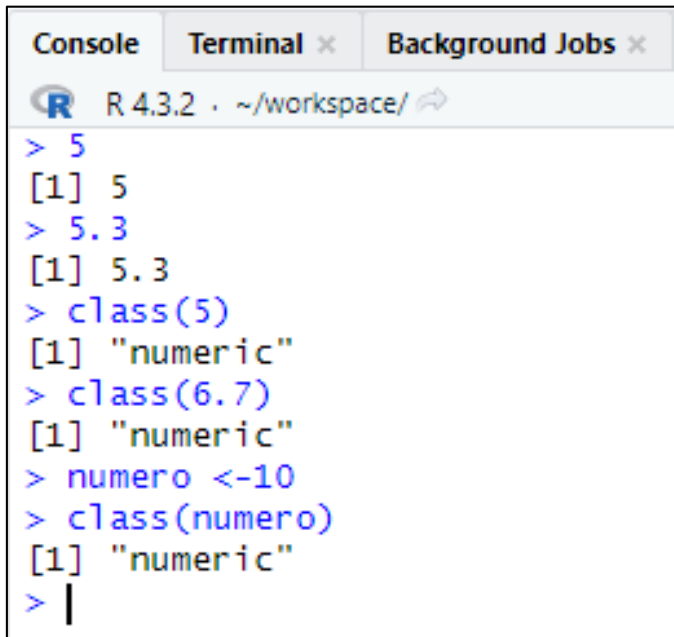
values	
numero	4
numero2	25
texto	"hola que tal como estas"
texto_corto	"hola"
texto.corto	"hola"
textoCorto	"hola"

Cuando el nombre de la variable es compuesto (más de una palabra) es aconsejable utilizar el punto para conectar las distintas palabras en minúsculas.

5. TIPOS DE DATOS

Paso 1. Los tipos de datos van a ser de tipo:

- Numérico
- Lógico
- Cadena de caracteres



```
Console Terminal x Background Jobs x
R 4.3.2 · ~/workspace/
> 5
[1] 5
> 5.3
[1] 5.3
> class(5)
[1] "numeric"
> class(6.7)
[1] "numeric"
> numero <- 10
> class(numero)
[1] "numeric"
> |
```

La función `class` nos indica el tipo a que pertenece la variable

5. TIPOS DE DATOS

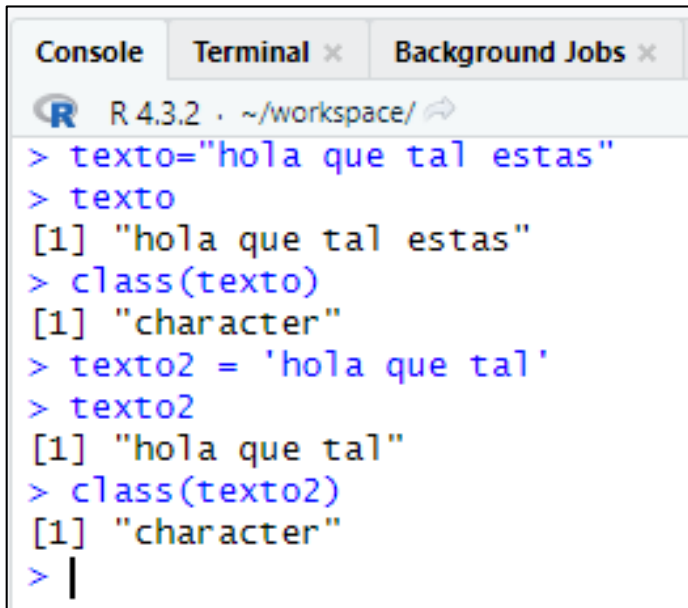
Paso 2. Las variables de tipos lógicos o booleanas, pueden tener los valores TRUE o FALSE, o T o F. Nos servirá luego para hacer condiciones: si se cumple la condición o si es true, haremos unas acciones, en caso contrario, si es falsa y haremos otra cosa.

```
Console Terminal x Background Jobs x
R 4.3.2 . ~/workspace/ ↗
> TRUE
[1] TRUE
> FALSE
[1] FALSE
> T
[1] TRUE
> F
[1] FALSE
> class(TRUE)
[1] "logical"
> class(FALSE)
[1] "logical"
> |
```

```
Console Terminal x Background Jobs x
R 4.3.2 . ~/workspace/ ↗
> resultado=TRUE
> resultado
[1] TRUE
> class(resultado)
[1] "logical"
> resultado2=T
> resultado2
[1] TRUE
> class(resultado2)
[1] "logical"
>
```

5. TIPOS DE DATOS

Paso 3. Las variables de tipo cadena de caracteres pueden estar delimitadas tanto por comillas simples como por comillas dobles

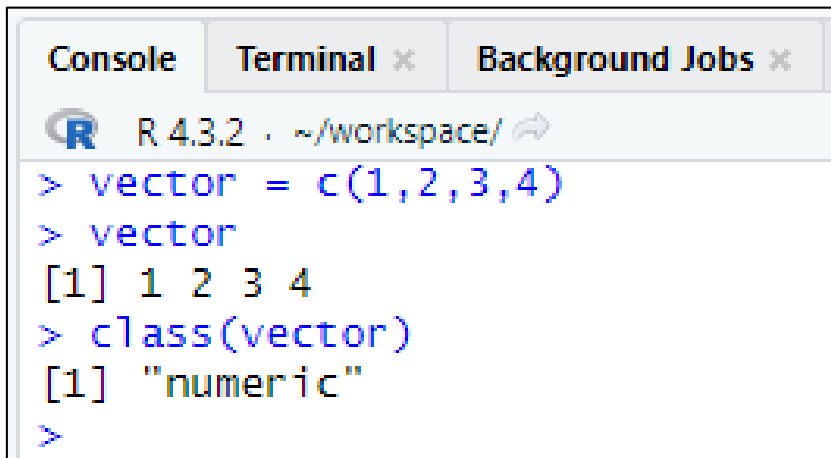


```
Console Terminal x Background Jobs x
R 4.3.2 · ~/workspace/ ↗
> texto="hola que tal estas"
> texto
[1] "hola que tal estas"
> class(texto)
[1] "character"
> texto2 = 'hola que tal'
> texto2
[1] "hola que tal"
> class(texto2)
[1] "character"
> |
```

6. VECTORES EN R

Paso 1. Un vector es un array de una única dimensión que puede tener valores numéricos, lógicos o cadenas de caracteres. Pero todos tienen que ser del mismo tipo, es decir, todos tienen que ser o numéricos, o de tipo carácter o de tipo lógico.

Para crear vectores usaremos la función **combine()** o **c()** para componer una lista de valores que asignaremos al vector.

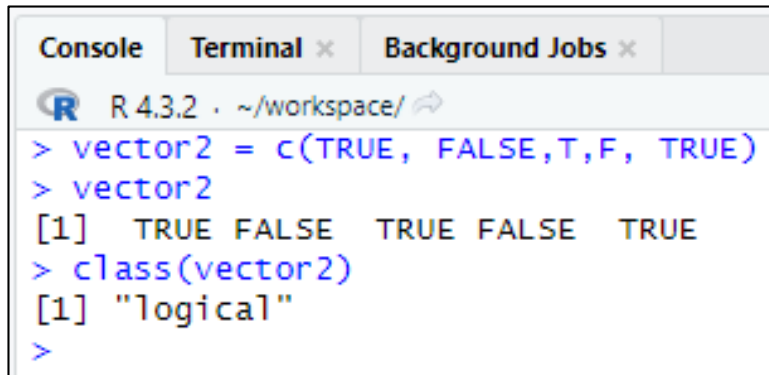


```
Console Terminal x Background Jobs x
R 4.3.2 . ~/workspace/
> vector = c(1,2,3,4)
> vector
[1] 1 2 3 4
> class(vector)
[1] "numeric"
>
```

La clase de la variable vector nos indica que es de tipo numérico porque sus cuatro elementos son números.

6. VECTORES EN R

Paso 2. Vamos a hacer ahora un vector de valores lógicos

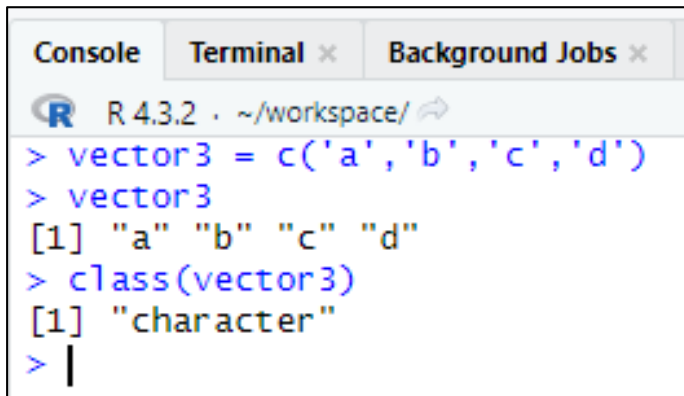


```
R 4.3.2 . ~/workspace/
> vector2 = c(TRUE, FALSE, T, F, TRUE)
> vector2
[1] TRUE FALSE TRUE FALSE TRUE
> class(vector2)
[1] "logical"
>
```

La T y la F son sustitutos de TRUE y FALSE.
Al tratarse de una lista de valores lógicos, el tipo del vector en este caso será lógico

6. VECTORES EN R

Paso 3. Vamos a hacer ahora un vector de tipo cadenas de caracteres



```
R 4.3.2 · ~/workspace/
> vector3 = c('a','b','c','d')
> vector3
[1] "a" "b" "c" "d"
> class(vector3)
[1] "character"
> |
```

Las letras o cadenas se pueden poner entre comillas simples o comillas dobles

Es de tipo character, porque todos los elementos son de tipo carácter

6. VECTORES EN R

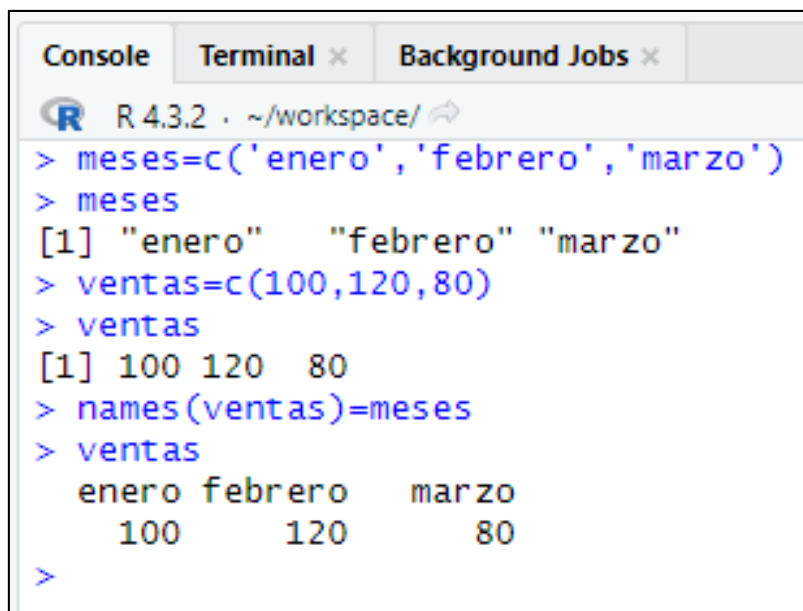
Paso 4. ¿Que pasa si tenemos varios elementos diferentes en el mismo vector?

```
Console Terminal x Background Jobs x
R 4.3.2 · ~/workspace/ ↗
> vector4 = c(TRUE, 10, 30)
> vector4
[1] 1 10 30
> class(vector4)
[1] "numeric"
> vector5 = c(10,15,30,"hola")
> vector5
[1] "10" "15" "30" "hola"
> class(vector5)
[1] "character"
> vector6 = c(TRUE,10,20,"adios")
> vector6
[1] "TRUE" "10" "20" "adios"
> class(vector6)
[1] "character"
>
```

En los vectores heterogéneos domina el tipo cadena y después el numérico. En el vector4 domina el tipo numérico y convierte el booleano a un valor numérico (TRUE→1, FALSE→0). En el vector5 domina el tipo carácter y convierte los valores numéricos en cadenas de caracteres. El vector6 combina números, booleanos y cadenas de caracteres, y el que domina es este último. Todo se convierte en cadenas de caracteres.

6. VECTORES EN R

Paso 5. Vamos a ver cómo podemos ponerle nombre a las columnas de un vector. Por ejemplo tenemos las ventas de tres meses enero, febrero y marzo y queremos hacer un vector que tenga los valores de las ventas y además indique cada valor de qué meses.



```
Console Terminal x Background Jobs x
R 4.3.2 · ~/workspace/
> meses=c('enero', 'febrero', 'marzo')
> meses
[1] "enero"  "febrero" "marzo"
> ventas=c(100,120,80)
> ventas
[1] 100 120 80
> names(ventas)=meses
> ventas
  enero febrero  marzo
    100     120     80
>
```

Mediante la función `names` le decimos que los nombres de las columnas de los valores numéricos del vector `ventas` van a ser los meses.

Esta sería la forma de ponerle un nombre a cada uno de los valores que tenemos dentro de un vector.