



BT1013

Análisis de la

Biología Computacional

Puntos

- 0.3 por participación.
- Leer alguna diapositiva, no cuenta.

Discusión

¿Cómo podemos aportar en la pandemia como computólogos?

Herramientas informáticas para detectar, diagnosticar, tratar y prevenir la propagación del virus.

¿Cómo podemos aportar en la pandemia como computólogos?

1. Desarrollo de aplicaciones: Los ingenieros en software pueden desarrollar aplicaciones para el seguimiento y monitoreo de la propagación del COVID-19, así como para el seguimiento de casos y contactos. También pueden desarrollar aplicaciones de telemedicina para permitir la atención médica remota, lo que reduce la necesidad de visitas presenciales a hospitales y consultorios médicos.
2. Análisis de datos: Los ingenieros en software pueden ayudar en el análisis de grandes cantidades de datos relacionados con la propagación del COVID-19. Pueden desarrollar algoritmos y modelos de aprendizaje automático para predecir la propagación del virus y ayudar a tomar decisiones informadas sobre políticas de salud pública.
3. Desarrollo de sistemas de monitoreo y detección: Los ingenieros en software pueden desarrollar sistemas de monitoreo en tiempo real para detectar la propagación del COVID-19. Estos sistemas pueden incluir sensores, cámaras y sistemas de reconocimiento facial para detectar personas con síntomas de COVID-19.
4. Desarrollo de tecnologías para el trabajo remoto: Los ingenieros en software pueden desarrollar tecnologías para facilitar el trabajo remoto y la educación en línea. Estas tecnologías incluyen plataformas de videoconferencia, herramientas de colaboración en línea y sistemas de gestión de proyectos.
5. Desarrollo de tecnologías de desinfección: Los ingenieros en software pueden desarrollar tecnologías para la desinfección de superficies y aire, utilizando la inteligencia artificial, el aprendizaje automático, y la robótica. Esto puede incluir el desarrollo de sistemas de robots de limpieza y desinfección, y sistemas de ventilación inteligente.

Introducción a R

Para que se utiliza R principalmente?



El lenguaje R

- R es un lenguaje de programación y un entorno de software libre para computación estadística y gráficos soportados por la R Foundation for Statistical Computing
- Compila y se ejecuta en una amplia variedad de plataformas UNIX, Windows y MacOS
- R es ampliamente utilizado entre los estadísticos, los dataminers y los bioinformáticos



El lenguaje R

- R fue creado por **Ross Ihaka** y **Robert Gentleman** en la Universidad de Auckland, Nueva Zelanda
- El proyecto fue concebido en 1992, con una versión inicial lanzada en 1995 y una versión beta estable (v1.0) el 29 de febrero de 2000.



Robert Gentleman y Ross Ihaka

El lenguaje



- R y sus bibliotecas implementan una amplia variedad de técnicas estadísticas y gráficas, que incluyen modelos lineales y no lineales, pruebas estadísticas clásicas, análisis de series temporales, clasificación, agrupamiento y otros.
- R produce gráficos con calidad de publicación.
- R se puede vincular con C, C++, Fortran, Python y Java.
- Aunque R tiene una interfaz de línea de comando, hay varias interfaces gráficas de usuario de terceros, como **RStudio**, un entorno de desarrollo integrado (IDE), y **jupyter**, una notebook interface (texto dinámico).



El lenguaje R

- Es una **herramienta muy poderosa** para todo tipo de procesamiento y manipulación de datos.
- Algunas **técnicas avanzadas y robustas** solo pueden realizarse con este software.
- Permite crear gráficos de alta calidad exportables en diversos formatos: PostScript, pdf, bitmap, pictex, png, jpeg, etc.
- **Consume pocos recursos** informáticos.
- Está **disponible para todos los sistemas operativos** (Windows, Macintosh y sistemas Unix como Linux).
- R **trabaja con otros lenguajes** y permite leer datos de otros softwares como SPSS, SAS, Excel, etc.
- Puedes **crear aplicaciones web interactivas** (apps) con la herramienta Shiny.
- Puedes **crear un flujo de trabajo** para escribir informes reproducibles y dinámicos y hacerlo en varios formatos (pdf, word, html).

Utilizar R u otros lenguajes?

- La elección de un lenguaje de programación dependerá de muchos factores, como el tipo de proyecto, el dominio del problema, la experiencia previa del programador, la disponibilidad de herramientas y bibliotecas, entre otros.

R es un lenguaje relativamente joven, pero que ha experimentado un crecimiento acelerado en su adopción durante los últimos 10 años.

¿Quién usa R?

Language Ranking: IEEE Spectrum

Rank	Language	Type	Score
1	Python	  	100.0
2	Java	  	96.3
3	C	  	94.4
4	C++	  	87.5
5	R		81.5
6	JavaScript		79.4
7	C#	   	74.5
8	Matlab		70.6
9	Swift	 	69.1
10	Go	 	68.0

Applications of



Instalación de R

Liga de RStudio: <https://rstudio.com/products/rstudio/download/>

R en línea?

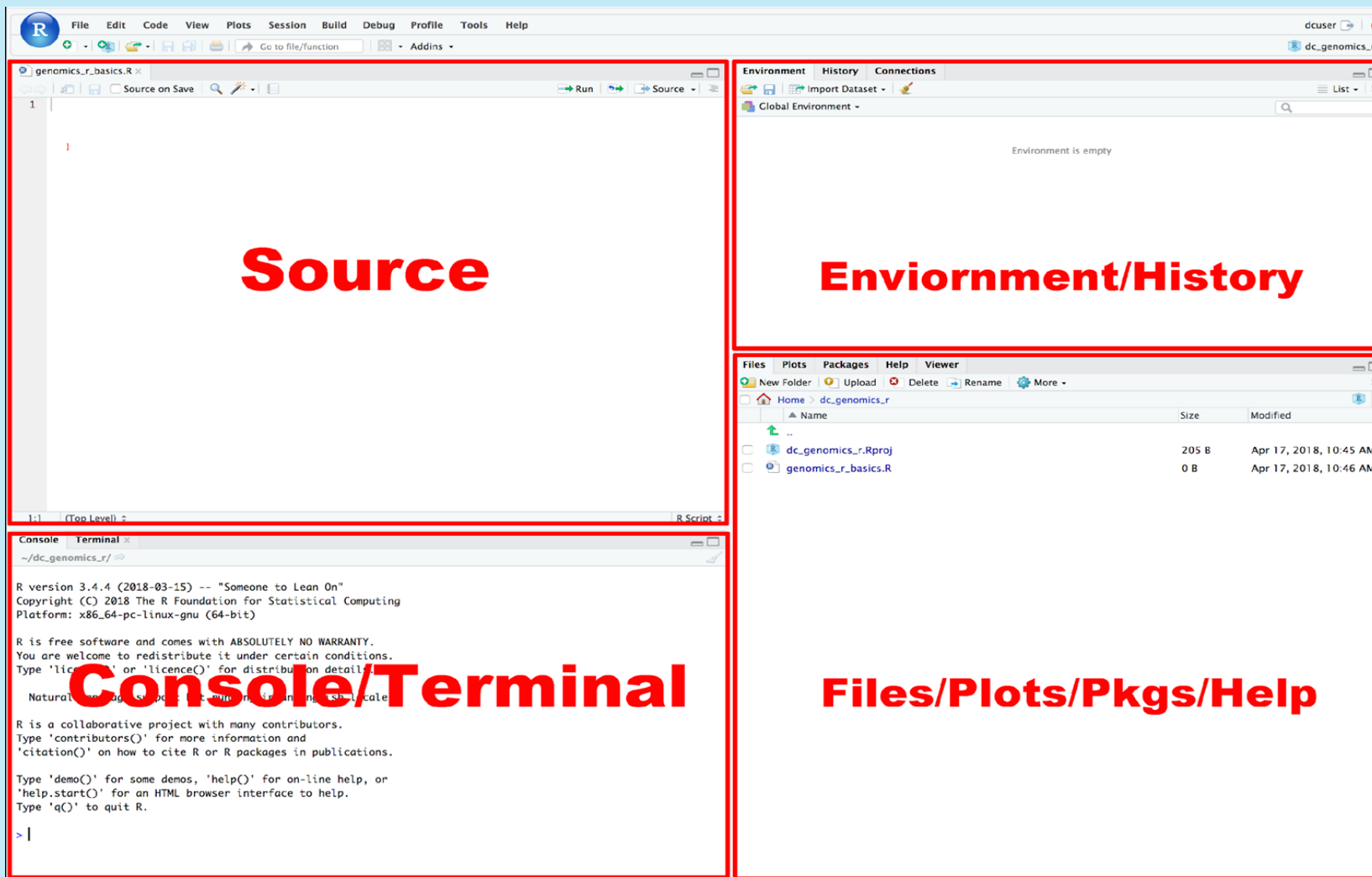
1. RStudio Cloud: Es una plataforma en línea que permite trabajar con RStudio y ejecutar código R en la nube. Ofrece un entorno de desarrollo integrado (IDE) para R, y permite colaborar con otros usuarios en proyectos.
2. Google Colaboratory: Es una plataforma en línea gratuita de Google que permite ejecutar código R en un entorno en la nube. Permite trabajar con conjuntos de datos grandes y ofrece acceso a recursos de hardware de alta potencia.
3. Azure Notebooks: Es una plataforma en línea de Microsoft que ofrece soporte para R, Python y F#. Permite crear y compartir cuadernos de código, y ofrece integración con servicios en la nube de Azure.
4. DataCamp: Es una plataforma de aprendizaje en línea que ofrece cursos y proyectos de R, Python y SQL. También proporciona un entorno en la nube para practicar y ejecutar código en R. ¹⁴

**“HELP! I need
somebody”**



Tip:
Ctrl + L

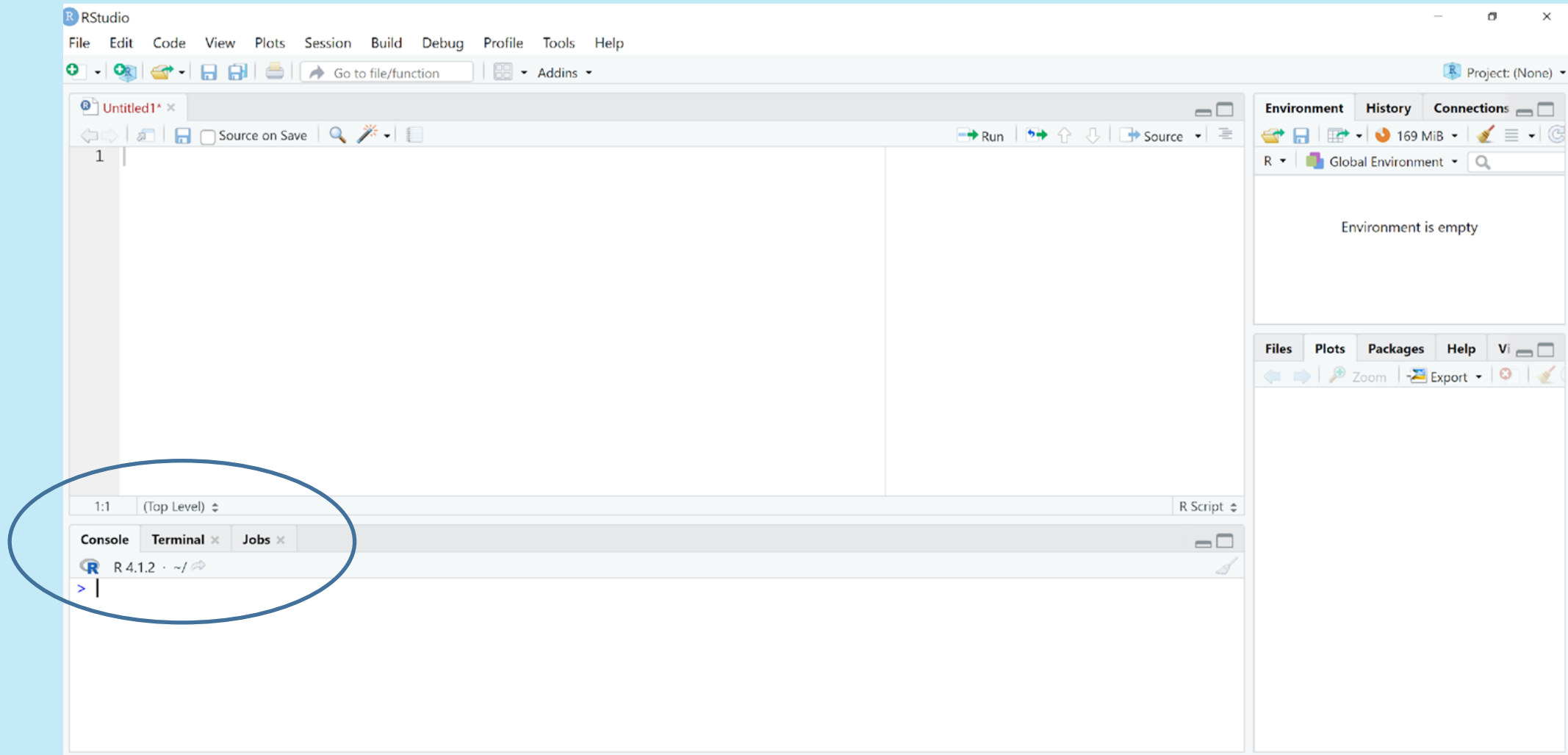
Los archivos que contienen scripts R terminan con la extensión .R



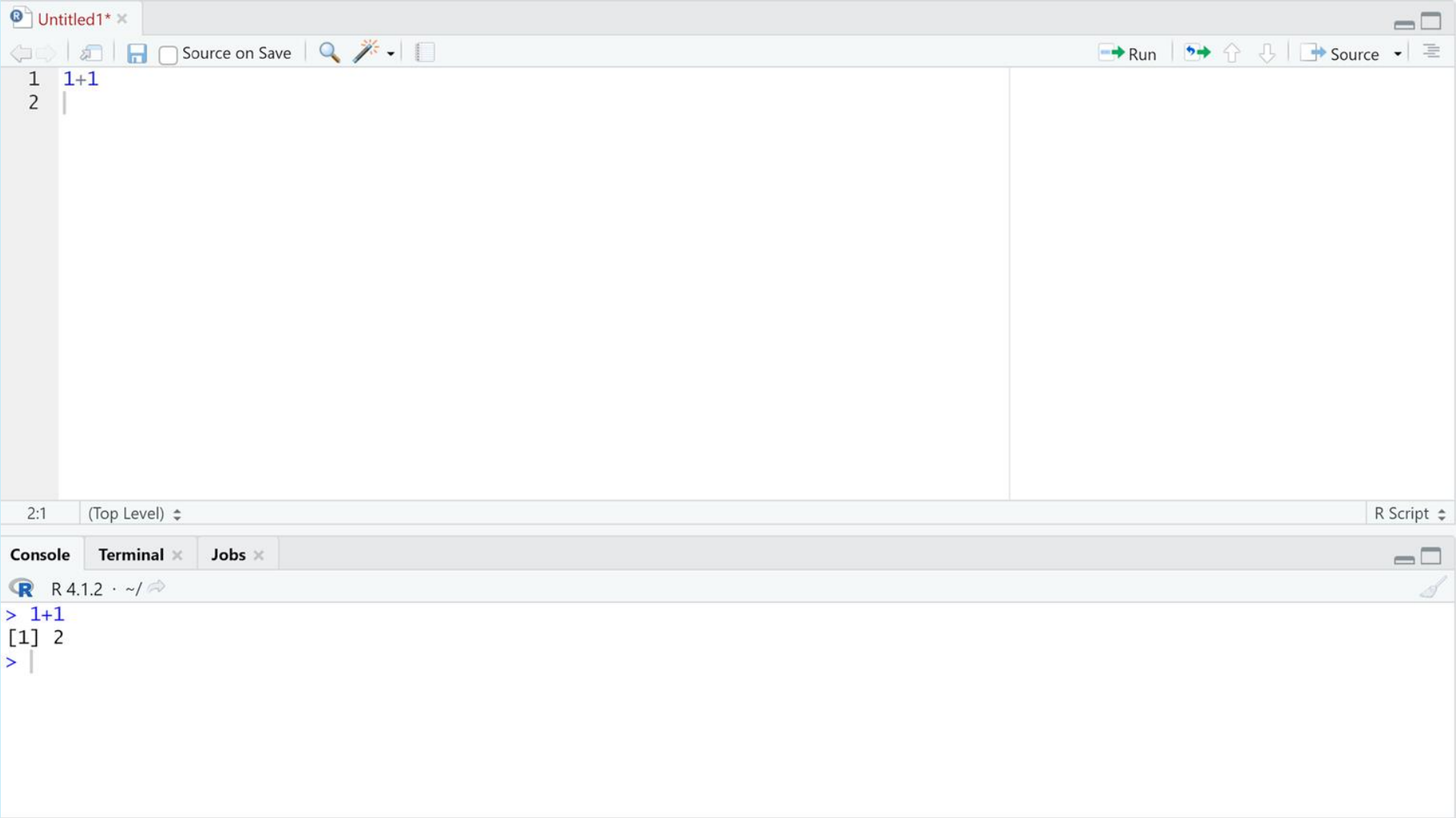
layout

- **Source:** panel para escribir/visualizar/guardar los scripts, incluyendo funciones, comandos, objetos, valores y paquetes.
- **Console:** panel donde se ejecutan los comandos. Esta es la misma pantalla que vería si estuviera usando R en la línea de comando sin RStudio. Se puede trabajar de forma interactiva (es decir, ingresando los comandos de R), pero regularmente se ejecuta un script (o líneas en un script) en el source y se observa su ejecución y salida aquí.
- **Environment/History:** muestra qué conjuntos de datos y objetos (variables) se han creado y cuáles están definidos en la memoria. También lista algunas propiedades de objetos/conjuntos de datos, como su tipo y dimensiones. La pestaña "Historial" contiene un historial de los comandos que se han ejecutado.
- **Files/Plots/Packages/Help:** panel multipropósito que muestra el contenido de los directorios en la computadora. La pestaña "Files" sirve para navegar y configurar el directorio de trabajo. La pestaña "Plots" muestra el resultado de los gráficos generados. En "Packages" se instalan o activan paquetes. "Help" muestra archivos de ayuda para funciones y paquetes de R.

La Consola de R



Ejecutar, llamar, correr y devolver



The screenshot displays the RStudio environment. The top pane is the script editor, titled 'Untitled1*', containing two lines of R code: `1 1+1` and `2`. The bottom pane is the console, showing the execution of the first line of code: `> 1+1` followed by the output `[1] 2`. The console title bar indicates 'R 4.1.2 · ~/'. The interface includes a toolbar with icons for running, saving, and other functions, and a status bar at the bottom showing the current line and column (2:1) and the file type (R Script).

```
1 1+1
2
```

```
> 1+1
[1] 2
>
```

?functionName

?setwd

```
?mean()
```

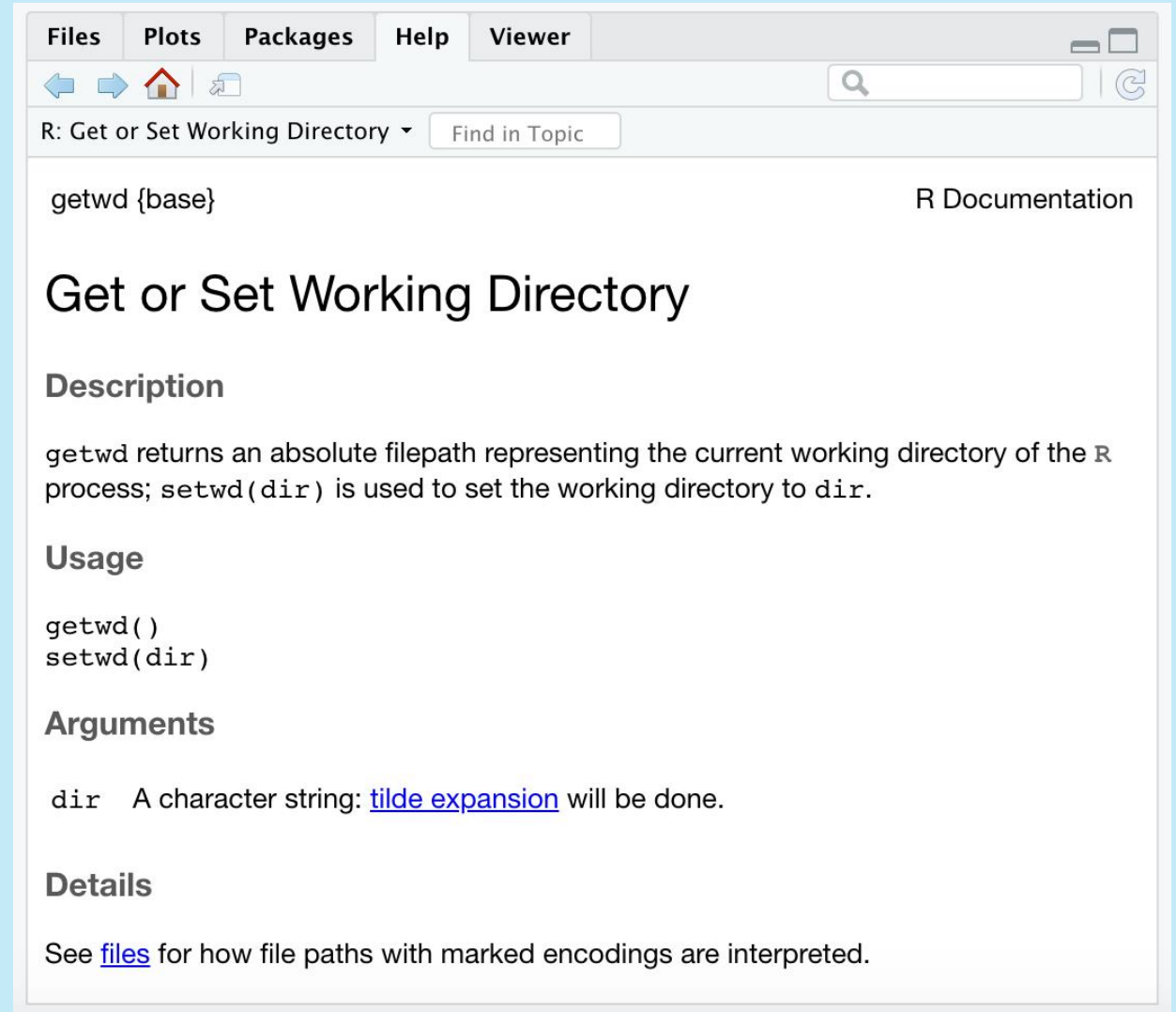
```
help("mean")
```

```
# Ver archivos
```

```
list.files()
```

```
# Ver directorios
```

```
list.dirs()
```



The screenshot shows the R help viewer window. The title bar includes tabs for Files, Plots, Packages, Help, and Viewer. Below the tabs is a navigation bar with back, forward, and home icons, a search bar, and a 'Find in Topic' button. The main content area displays the documentation for 'getwd {base}' and 'setwd {base}'. The title 'Get or Set Working Directory' is prominently displayed. The 'Description' section explains that 'getwd' returns the current working directory and 'setwd' sets it. The 'Usage' section lists the functions 'getwd()' and 'setwd(dir)'. The 'Arguments' section describes the 'dir' argument as a character string with tilde expansion. The 'Details' section refers to the 'files' help page for file path encodings.

R Documentation

Get or Set Working Directory

Description

`getwd` returns an absolute filepath representing the current working directory of the R process; `setwd(dir)` is used to set the working directory to `dir`.

Usage

```
getwd()  
setwd(dir)
```

Arguments

`dir` A character string: [tilde expansion](#) will be done.

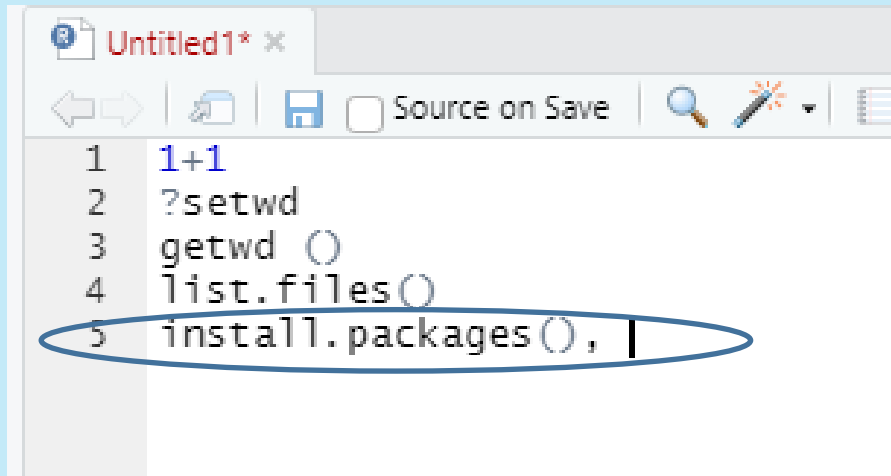
Details

See [files](#) for how file paths with marked encodings are interpreted.

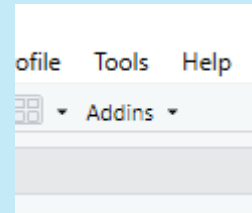
Paquetes

R puede ser expandido con **paquetes**. Cada paquete es una colección de funciones diseñadas para atender una tarea específica. Por ejemplo, hay paquetes para trabajo visualización geoespacial, análisis psicométricos, minería de datos, interacción con servicios de internet y muchas otras cosas más.

Instalarlo



```
1 1+1
2 ?setwd
3 getwd ()
4 list.files()
5 install.packages(), |
```



```
install.packages("readr")
```

Lllamarlo

```
library(readr)
```

Paquetes

Topics

Bayesian	Bayesian Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trial Design, Monitoring, and Analysis
Cluster	Cluster Analysis & Finite Mixture Models
Databases	Databases with R
DifferentialEquations	Differential Equations
Distributions	Probability Distributions
Econometrics	Econometrics
Environmetrics	Analysis of Ecological and Environmental Data
ExperimentalDesign	Design of Experiments (DoE) & Analysis of Experimental Data
ExtremeValue	Extreme Value Analysis
Finance	Empirical Finance
FunctionalData	Functional Data Analysis
Genetics	Statistical Genetics
Graphics	Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization
HighPerformanceComputing	High-Performance and Parallel Computing with R
Hydrology	Hydrological Data and Modeling
MachineLearning	Machine Learning & Statistical Learning
MedicalImaging	Medical Image Analysis
MetaAnalysis	Meta-Analysis
MissingData	Missing Data

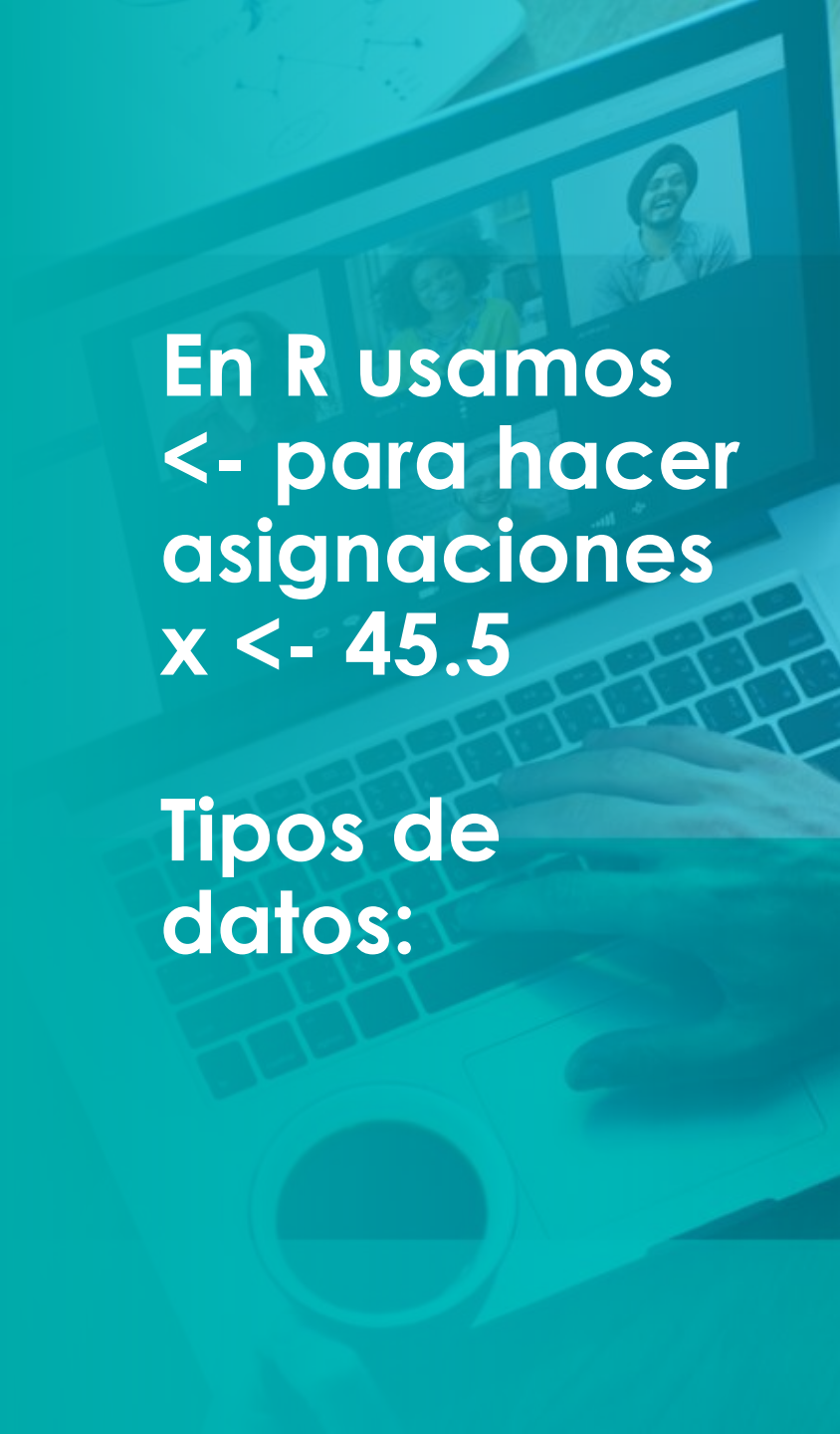
ModelDeployment	Model Deployment with R
Multivariate	Multivariate Statistics
NaturalLanguageProcessing	Natural Language Processing
NumericalMathematics	Numerical Mathematics
OfficialStatistics	Official Statistics & Survey Methodology
Optimization	Optimization and Mathematical Programming
Pharmacokinetics	Analysis of Pharmacokinetic Data
Phylogenetics	Phylogenetics, Especially Comparative Methods
Psychometrics	Psychometric Models and Methods
ReproducibleResearch	Reproducible Research
Robust	Robust Statistical Methods
SocialSciences	Statistics for the Social Sciences
Spatial	Analysis of Spatial Data
SpatioTemporal	Handling and Analyzing Spatio-Temporal Data
Survival	Survival Analysis
TeachingStatistics	Teaching Statistics
TimeSeries	Time Series Analysis
Tracking	Processing and Analysis of Tracking Data
WebTechnologies	Web Technologies and Services
gR	gRaphical Models in R

Repositorios

- [CRAN](#): el repositorio oficial, es una red de servidores mantenidos por la comunidad R de todo el mundo. La fundación R lo coordina, y para que un paquete se publique aquí, debe pasar varias pruebas que garanticen que el paquete siga las políticas de CRAN
- [Bioconductor](#): este es un repositorio destinado a software de código abierto para bioinformática. Como CRAN, tiene sus propios procesos de presentación y revisión, y su comunidad es muy activa teniendo varias conferencias y reuniones por año
- [Github](#): no es específico de R, Github es probablemente el repositorio más popular para proyectos de código abierto. Su popularidad proviene del espacio ilimitado para el código abierto, la integración con git, un software de control de versiones y su facilidad para compartir y colaborar con otros. No tiene un proceso de revisión asociado

Tipos de datos





En R usamos
`<-` para hacer
asignaciones
`x <- 45.5`

Tipos de
datos:

Tipo	Ejemplo	Nombre en inglés
Entero	1	integer
Numérico	1.3	numeric
Cadena de texto	"uno"	character
Factor	uno	Factor
Lógico	TRUE	logical
Perdido	NA	NA
Vacio	NULL	null

Numéricos: estos son los números reales, como 3.14 o 2.71828. En R, se almacenan como objetos de clase "numeric".

Coerción

La coerción de tipos se realiza de los tipos de datos más restrictivos a los más flexibles.

La coerción es muy importante. Cuando pedimos a R ejecutar una operación, intentará coercionar de manera **implícita**, sin avisarnos, los datos de su tipo original al tipo correcto que permita realizarla.

Las coerciones ocurren en el siguiente orden.

```
lógico -> entero -> numérico -> cadena de texto
```

Coerción

Función	Tipo al que hace coerción
<code>as.integer()</code>	Entero
<code>as.numeric()</code>	Numerico
<code>as.character()</code>	Cadena de texto
<code>as.factor()</code>	Factor
<code>as.logical()</code>	Lógico
<code>as.null()</code>	NULL

```
as.character(5)
```

```
as.numeric("cinco")
```

```
> as.numeric("cinco")  
[1] NA  
Warning message:  
NAs introduced by coercion
```

Comprobemos el comportamiento especial de los factores. Podemos coercionar al número 5 y la palabra “cinco” en un factor.

```
as.factor(5)
```

```
## [1] 5
```

```
## Levels: 5
```

```
as.factor("cinco")
```

```
## [1] cinco
```

```
## Levels: cinco
```


Si coercionamos un dato de tipo lógico a numérico, TRUE siempre devolverá 1 y FALSE dará como resultado 0.

```
as.numeric(TRUE)
```

```
## [1] 1
```

```
as.numeric(FALSE)
```

```
## [1] 0
```

TRUE se trata como equivalente a 1 y FALSE se trata como equivalente a 0.

Por último, la función `as.null()` siempre devuelve NULL, sin importar el tipo de dato que demos como argumento.

```
# Lógico  
as.null(FALSE)
```

```
## NULL
```

```
# Numérico  
as.null(457)
```

```
## NULL
```

```
# Cadena de texto  
as.null("palabra")
```

```
## NULL
```



Operadores

15 * 3

[1] 45

Operadores aritméticos

Operador	Operación	Ejemplo	Resultado
+	Suma	$5 + 3$	8
-	Resta	$5 - 3$	2
*	Multiplicación	$5 * 3$	18
/	División	$5 / 3$	1.666667
\wedge	Potencia	$5 \wedge 3$	125
%%	División entera	$5 \% \% 3$	2

Operadores relacionales R vs Python

Operador	Comparación	Ejemplo	Resultado
<	Menor que	5 < 3 (Coinciden)	FALSE
<=	Menor o igual que	5 <= 3 (Coinciden)	FALSE
>	Mayor que	5 > 3 (Coinciden)	TRUE
>=	Mayor o igual que	5 >= 3 (Coinciden))	TRUE
==	Exactamente igual que	5 == 3 (Coinciden)	FALSE
!=	No es igual que	5 != 3 (Coinciden)	TRUE

Operador	Descripción	Resultado
`==`	Igualdad	`True` si los valores son iguales, `False` en caso contrario
`!=`	Desigualdad	`True` si los valores son diferentes, `False` en caso contrario
`>`	Mayor que	`True` si el primer valor es mayor que el segundo, `False` en caso contrario
`<`	Menor que	`True` si el primer valor es menor que el segundo, `False` en caso contrario
`>=`	Mayor o igual que	`True` si el primer valor es mayor o igual que el segundo, `False` en caso contrario
`<=`	Menor o igual que	`True` si el primer valor es menor o igual que el segundo, `False` en caso contrario

Pasa al pizarrón a contestar los ejercicios

Ejemplos	Respuestas
$5 < 3$	
$5 \leq 3$	
$5 > 3$	
$5 \geq 3$	
$5 == 3$	
$5 != 3$	

Operadores relacionales

Por ejemplo, `"casa" > "barco"` nos devuelve `TRUE`

```
"casa" > "barco"
```

```
## [1] TRUE
```

```
"casa"> "barco"  
"altura">"television"
```


Operadores lógicos

Operador	Comparación	Ejemplo	Resultado
x y	x Ó y es verdadero	TRUE FALSE	TRUE
x & y	x Y y son verdaderos	TRUE & FALSE	FALSE
!x	x no es verdadero (negación)	!TRUE	FALSE
is TRUE(x)	x es verdadero (afirmación)	is TRUE (TRUE)	TRUE

Operador "y" lógico: & Este operador devuelve TRUE si ambas expresiones son verdaderas.

- Ejemplo:
- $x <- 3$
- $y <- 4$
- $x > 2 \ \& \ y < 5 \ \# \text{ devuelve TRUE}$

Operador "y" lógico: & Este operador devuelve TRUE si ambas expresiones son verdaderas.
Ejemplo:

```
x <- 3  
y <- 4  
x > 2 & y < 5 # devuelve TRUE
```

Operador "o" lógico: | Este operador devuelve TRUE si al menos una de las expresiones es verdadera.
Ejemplo:

```
x <- 3  
y <- 4  
x > 2 | y < 5 # devuelve TRUE
```

Operador "no" lógico: !. Este operador invierte el valor de la expresión. Si la expresión es verdadera, devuelve FALSE; si es falsa, devuelve TRUE.
Ejemplo:

```
x <- 3  
y <- 4  
!(x > 2 & y < 5) # devuelve FALSE
```

Operadores de asignación

Operador	Operación
<code><-</code>	Asigna un valor a una variable

- En este ejemplo, asignamos valores a las variables estatura y peso.

```
estatura <- 1.73  
peso <- 83
```

Llamamos a sus valores asignados

```
estatura
```

```
## [1] 1.73
```

```
peso
```

```
## [1] 83
```

OJO: Si deseamos que una operación ocurra antes que otra, rompiendo este orden de evaluación, usamos paréntesis.

Orden de operaciones

En R, al igual que en matemáticas, las operaciones tienen un orden de evaluación definido.

Orden	Operadores
1	\wedge
2	* /
3	+ -
4	< > <= >= == !=
5	!
6	&
7	
8	<-

Orden de operaciones (Hacerlo en la calculadora del celular)

El orden de las operaciones matemáticas en R sigue las reglas estándar de la aritmética, que se conocen como el acrónimo PEMDAS (Paréntesis, Exponentes, Multiplicación y División, Suma y Resta). Esto significa que las operaciones dentro de los paréntesis se realizan primero, seguidas de las potencias, luego la multiplicación y la división (en el orden en que aparecen en la expresión) y finalmente la suma y la resta (en el orden en que aparecen en la expresión).

$$(4 + 5) * 3 / 2^2 - 1$$

Resultado

$$6.75 - 1 = 5.75$$



Estructuras de datos

Vectores

Un vector es la estructura de datos más sencilla en R.

Un vector es una colección de uno o más datos del mismo tipo.

```
miVector <- c(1, 2, 3, 5, 8, 13)  
# crea un vector
```

```
miVector <- c(miVector, 79)  
# agregar un dato
```

```
miVector <- 1:10  
# vector como una secuencia
```

```
miVector <- -43:-30  
# vector como una secuencia
```

Creación de vectores

- Creamos vectores usando la función `c()` (*combinar*).
- Llamamos esta función y le damos como argumento los elementos que deseamos combinar en un vector, separados por comas.

```
# Vector numérico
```

```
c(1, 2, 3, 5, 8, 13)
```

```
# Vector de cadena de texto
```

```
c("arbol", "casa", "persona")
```

```
# Vector lógico
```

```
c(TRUE, TRUE, FALSE, FALSE, TRUE)
```

Creación de vectores

- Si deseamos agregar un elemento a un vector ya existente, podemos hacerlo combinando nuestro vector original con los elementos nuevos y asignando el resultado a nuestro vector original.

```
mi_vector <- c(TRUE, FALSE, TRUE)
```

```
mi_vector <- c(mi_vector, FALSE)
```

```
mi_vector
```

```
## [1] TRUE FALSE TRUE FALSE
```

Creación de vectores

- Naturalmente, podemos crear vectores que son combinación de vectores.

```
mi_vector_1 <- c(1, 3, 5)
mi_vector_2 <- c(2, 4, 6)

mi_vector_3 <- c(mi_vector_1, mi_vector_2)

mi_vector_3

## [1] 1 3 5 2 4 6
```

Creación de vectores

- Podemos crear vectores de secuencias numéricas usando :
- De un lado de los dos puntos escribimos el número de inicio de la secuencia y del otro el final.
- Por ejemplo, creamos una secuencia del 1 al 10.

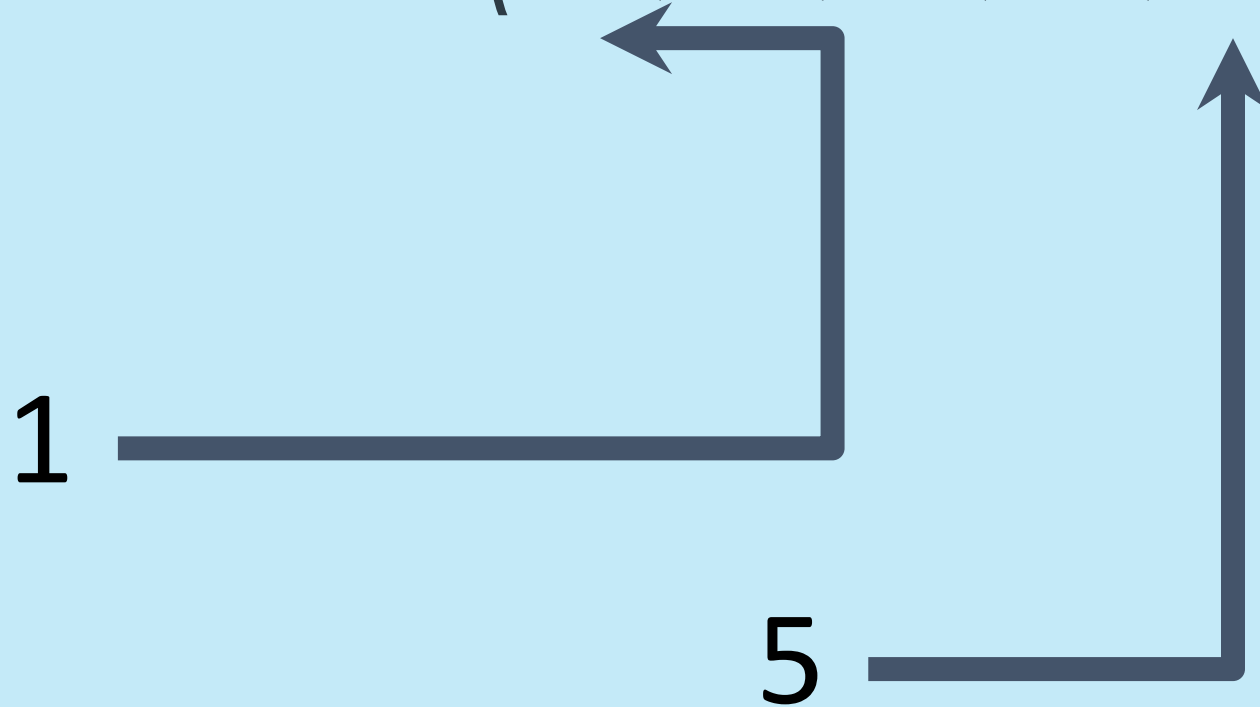
```
1:10
```

```
##      [1]  1  2  3  4  5  6  7  8  9 10
```

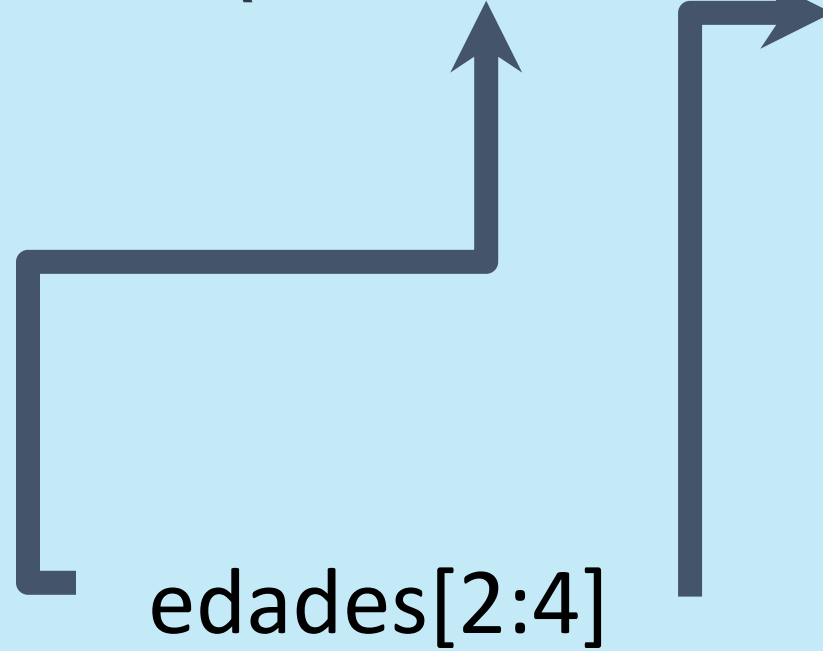


```
edades <- c(60,37,68,28,40)
```

```
edades <- c(60,37,68,28,40)
```



```
edades <- c(60,37,68,28,40)
```



```
edades <- c(60,37,68,28,40)  
edades[2:4]
```

```
> edades[2:4]  
[1] 37 68 28
```

Podemos asignar a los vectores numéricos, la información que corresponde a sus valores, como podemos observar, Jara tiene 60 años, Odile tiene 37, Karla 68 Esther 28, Y rebeca 40 años

```
edades <- c(jara=60,odile=37,karla=68,esther=28,rebeca=40)
```

```
> edades
  jara  odile  karla  esther  rebeca
   60    37    68    28    40
```

```
edades <- c(jara=60,odile=37,karla=68,esther=28,rebeca=40)
```

```
edades["jara"]
```

```
edades["rebeca"]
```

```
> edades["jara"]  
jara  
60
```

```
edades <- c(jara=60,odile=37,karla=68,esther=28,rebeca=40)
```

```
edades[c(2,4)]
```

```
edades[c(2:4)]
```

```
edades[c(4:2)]
```



```
edades <- c(jara=60,odile=37,karla=68,esther=28,rebeca=40)
```

```
edades>=60
```

```
> edades>=60
jara  odile  karla  esther  rebeca
TRUE  FALSE  TRUE  FALSE  FALSE
> |
```

```
edades[edades>=60]
```

```
> edades[edades>=60]
jara  karla
  60    68
```

Matrices y arrays Función matrix(), y acepta nrow y ncol.

Las matrices y arrays pueden ser descritas como **vectores multidimensionales**. Al igual que un vector, únicamente pueden contener datos de un sólo tipo.

```
# matrix() sin especificar renglones ni columnas
matrix(1:12)
```

##		[,1]
##	[1,]	1
##	[2,]	2
##	[3,]	3
##	[4,]	4
##	[5,]	5
##	[6,]	6
##	[7,]	7
##	[8,]	8
##	[9,]	9
##	[10,]	10
##	[11,]	11
##	[12,]	12

```
# Tres renglones y cuatro columnas
matrix(1:12, nrow = 3, ncol = 4)
```

##		[,1]	[,2]	[,3]	[,4]
##	[1,]	1	4	7	10
##	[2,]	2	5	8	11
##	[3,]	3	6	9	12

data.frame()

Data frames

Los data frames son estructuras de datos de dos dimensiones (rectangulares) que pueden contener datos de diferentes tipos, por lo tanto, son heterogéneas.

##	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
## 1	5.1	3.5	1.4	0.2	setosa
## 2	4.9	3.0	1.4	0.2	setosa
## 3	4.7	3.2	1.3	0.2	setosa
## 4	4.6	3.1	1.5	0.2	setosa
## 5	5.0	3.6	1.4	0.2	setosa

Listas

Las listas, al igual que los vectores, son estructuras de datos unidimensionales, solo tienen largo, pero a diferencia de los vectores cada uno de sus elementos puede ser de diferente tipo o incluso de diferente clase.

```
miVector <- 1:10  
miMatriz <- matrix(1:4, nrow = 2)  
miDf <- data.frame("num" = 1:3, "let"  
= c("a", "b", "c"))  
  
mi_lista <- list("un_vector" = miVector,  
"una_matriz" = miMatriz, "un_df" = miDf)
```

```
mi_vector <- 1:10
mi_matriz <- matrix(1:4, nrow = 2)
mi_df <- data.frame("num" = 1:3, "let" = c("a", "b", "c"))

mi_lista <- list("un_vector" = mi_vector, "una_matriz" = mi_matriz, "un_df" = mi_df)

mi_lista
```

```
$un_vector
[1] 1 2 3 4 5 6 7 8 9 10

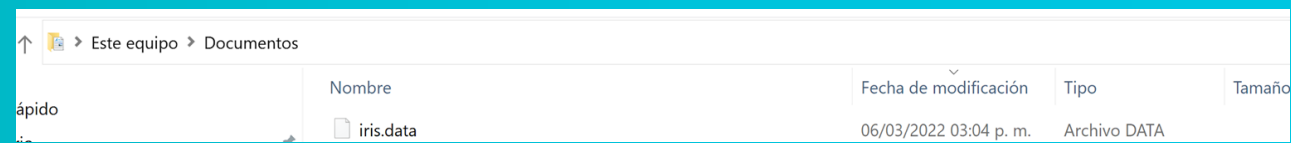
$una_matriz
      [,1] [,2]
[1,]    1    3
[2,]    2    4

$un_df
  num let
1   1   a
2   2   b
3   3   c
```

Importar y exportar datos

- R puede importar datos de una amplia variedad de tipos de archivo con las funciones en *base* además de que esta capacidad es ampliada con el uso de paquetes específicos.
- Cuando importamos un archivo, estamos guardando su contenido en nuestra sesión como un objeto. Dependiendo del procedimiento que usemos será el tipo de objeto creado.

Descargando datos



Este equipo > Documentos				
	Nombre	Fecha de modificación	Tipo	Tamaño
árido	iris.data	06/03/2022 03:04 p. m.	Archivo DATA	

- Función para descargar archivos de internet: `download.file()`
- Ejemplo: para descargar una copia del set *iris* disponible en el [UCI Machine Learning Repository](https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data) usamos la siguiente dirección como argumento url:
- <https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data>
- Y asignamos "iris.data" al argumento dest.

```
download.file(  
  url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data",  
  destfile = "iris.data"  
)
```

Fisher's *Iris* Data

Largo de sépalo ↕	Ancho de sépalo ↕	Largo de pétalo ↕	Ancho de pétalo ↕	Especies ↕
5.1	3.5	1.4	0.2	<i>I. setosa</i>
4.9	3.0	1.4	0.2	<i>I. setosa</i>
4.7	3.2	1.3	0.2	<i>I. setosa</i>
4.6	3.1	1.5	0.2	<i>I. setosa</i>
5.0	3.6	1.4	0.2	<i>I. setosa</i>
5.4	3.9	1.7	0.4	<i>I. setosa</i>
4.6	3.4	1.4	0.3	<i>I. setosa</i>
5.0	3.4	1.5	0.2	<i>I. setosa</i>
4.4	2.9	1.4	0.2	<i>I. setosa</i>
4.9	3.1	1.5	0.1	<i>I. setosa</i>
5.4	3.7	1.5	0.2	<i>I. setosa</i>
4.8	3.4	1.6	0.2	<i>I. setosa</i>
4.8	3.0	1.4	0.1	<i>I. setosa</i>
4.3	3.0	1.1	0.1	<i>I. setosa</i>
5.8	4.0	1.2	0.2	<i>I. setosa</i>
5.7	4.4	1.5	0.4	<i>I. setosa</i>
5.4	3.9	1.3	0.4	<i>I. setosa</i>
5.1	3.5	1.4	0.3	<i>I. setosa</i>
5.7	3.8	1.7	0.3	<i>I. setosa</i>



Iris setosa



Iris versicolor



Tablas (datos rectangulares)

read.table()

Usage

```
read.table(file, header = FALSE, sep = "", quote = "\"",  
           dec = ".", numerals = c("allow.loss", "warn.loss", "no.loss"),  
           row.names, col.names, as.is = !stringsAsFactors,  
           na.strings = "NA", colClasses = NA, nrows = -1,  
           skip = 0, check.names = TRUE, fill = !blank.lines.skip,  
           strip.white = FALSE, blank.lines.skip = TRUE,  
           comment.char = "#",  
           allowEscapes = FALSE, flush = FALSE,  
           stringsAsFactors = FALSE,  
           fileEncoding = "", encoding = "unknown", text, skipNul = FALSE)  
  
read.csv(file, header = TRUE, sep = ",", quote = "\"",  
         dec = ".", fill = TRUE, comment.char = "", ...)  
  
read.csv2(file, header = TRUE, sep = ";", quote = "\"",  
          dec = ",", fill = TRUE, comment.char = "", ...)  
  
read.delim(file, header = TRUE, sep = "\t", quote = "\"",  
           dec = ".", fill = TRUE, comment.char = "", ...)  
  
read.delim2(file, header = TRUE, sep = "\t", quote = "\"",  
            dec = ",", fill = TRUE, comment.char = "", ...)
```

Tablas (datos rectangulares)

read.table()

library(readr)

- file: La ruta del archivo que importaremos, como cadena de texto. Si el archivo se encuentra en nuestro [directorio de trabajo](#), es suficiente dar el nombre del archivo, sin la ruta completa.
- header: Si nuestro archivo tiene encabezados, para ser interpretados como nombres de columna, definimos este argumento como TRUE.
- sep: El caracter que es usado como separador de columnas. Por defecto es “;”.
- col.names: Un vector opcional, de tipo caracter, con los nombres de las columnas en la tabla.
- stringsAsFactors: Esta función convierte automáticamente los datos de texto a factores. Si este no es el comportamiento que deseamos, definimos este argumento como FALSE.

OJO: Es importante señalar que el objeto obtenido al usar esta función es siempre un data frame.

Ejemplo

```
download.file( url = "https://raw.githubusercontent.com/jboscomendoza/r-principiantes-bookdown/master/datos/breast-cancer-wis.data", dest = "breast-cancer-wis.data" )
```

```
bcancer <- read.table(file = "breast-cancer-wis.data")
```

```
head(bcancer)
```

##	V1
## 1	1000025,5,1,1,1,2,1,3,1,1,2
## 2	1002945,5,4,4,5,7,10,3,2,1,2
## 3	1015425,3,1,1,1,2,2,3,1,1,2
## 4	1016277,6,8,8,1,3,4,3,7,1,2
## 5	1017023,4,1,1,3,2,1,3,1,1,2
## 6	1017122,8,10,10,8,7,10,9,7,1,4

Ejemplo

```
bcancer <- read.table(file = "breast-cancer-wis.data", header = FALSE, sep = ",")
```

```
# Resultado
```

```
head(bcancer)
```

```
##           V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11
## 1 1000025   5  1  1  1  2  1  3  1  1  2
## 2 1002945   5  4  4  5  7 10  3  2  1  2
## 3 1015425   3  1  1  1  2  2  3  1  1  2
## 4 1016277   6  8  8  1  3  4  3  7  1  2
## 5 1017023   4  1  1  3  2  1  3  1  1  2
## 6 1017122   8 10 10  8  7 10  9  7  1  4
```

Ejemplo

- <https://raw.githubusercontent.com/jboscomendoza/r-principiantes-bookdown/master/datos/breast-cancer-wis.names>

```
nombres <- c("id", "clump_t", "u_csize", "u_cshape", "m_adh", "spcs", "b_nuc",  
            "b_chr", "n_nuc", "mit", "class")
```

```
bcancer <- read.table(file = "breast-cancer-wis.data", header = FALSE, sep = ",",  
                    col.names = nombres)
```

Resultado

```
head(bcancer)
```

##	id	clump_t	u_csize	u_cshape	m_adh	spcs	b_nuc	b_chr	n_nuc	mit	class
## 1	1000025	5	1	1	1	2	1	3	1	1	2
## 2	1002945	5	4	4	5	7	10	3	2	1	2
## 3	1015425	3	1	1	1	2	2	3	1	1	2
## 4	1016277	6	8	8	1	3	4	3	7	1	2
## 5	1017023	4	1	1	3	2	1	3	1	1	2
## 6	1017122	8	10	10	8	7	10	9	7	1	4

Hojas de cálculo de Excel

- R base no tiene una función para importar archivos almacenados en archivos con extensión **.xsl** y **.xlsx**, creados con *Excel*.
 - Usamos la función `installpackages()`
 - `install.packages("readxl")`
 - Ya instalado, cargamos el **readxl** a nuestra sesión de trabajo.
 - `library(readxl)`

Gráficas

En R, la función `plot()` es usada de manera general para crear gráficos.

x	y	Gráfico
Continuo	Continuo	Diagrama de dispersión (<i>Scatterplot</i>)
Continuo	Discreto	Diagrama de dispersión, <i>y</i> coercionada a numérica
Continuo	Ninguno	Diagrama de dispersión, por número de renglón
Discreto	Continuo	Diagrama de caja (<i>Box plot</i>)
Discreto	Discreto	Gráfico de mosaico (Diagrama de Kinneman)
Discreto	Ninguno	Gráfica de barras
Ninguno	Cualquiera	Error

En donde los tipos de dato son:

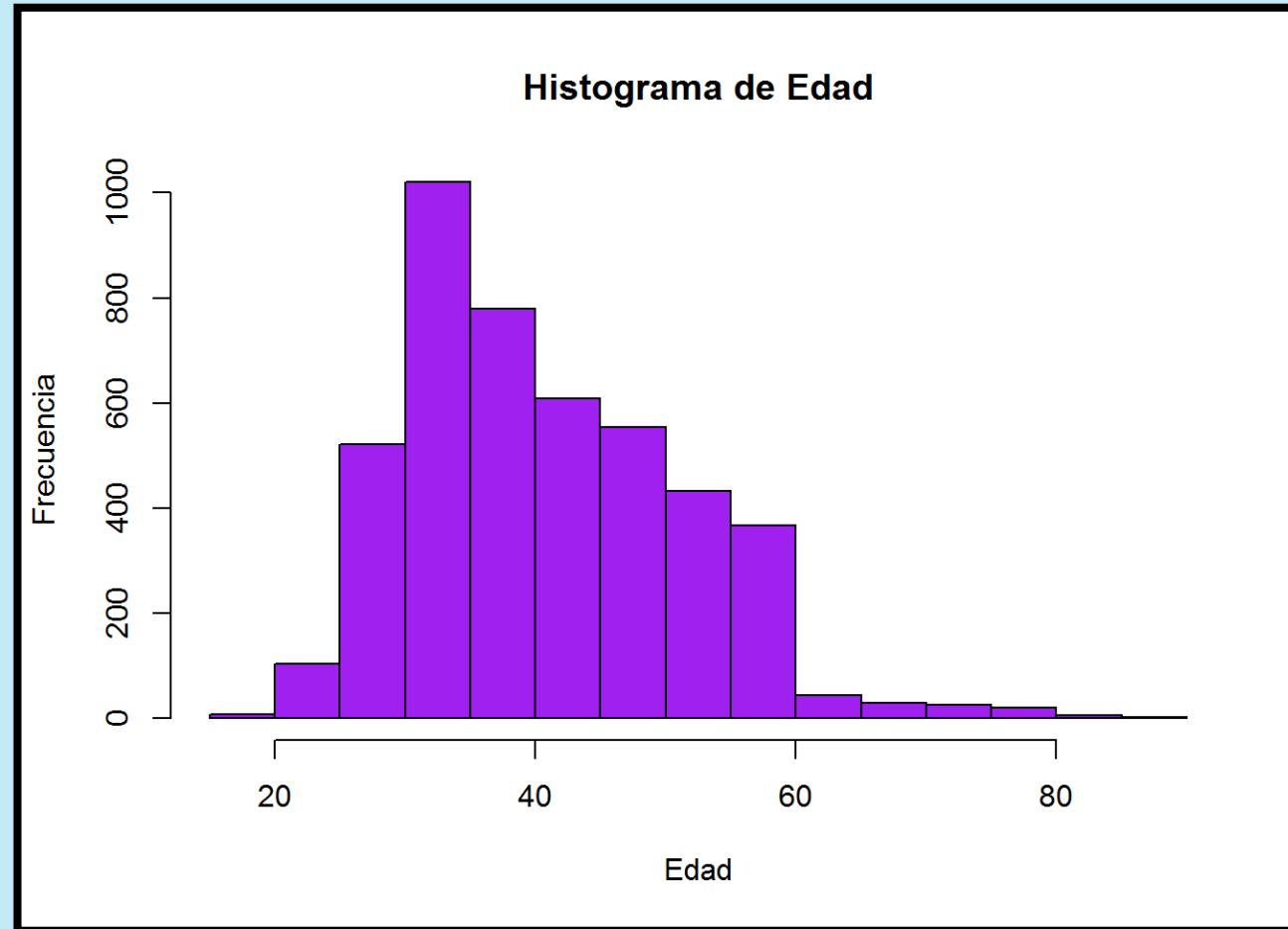
- **Continuo**: Un vector numérico, entero, lógico o complejo.
- **Discreto**: Un vector de factores o cadenas de texto.

Gráficas

- Gráfica de barras:
- `plot()`
- `barplot()`

- Histogramas:
- `hist()`

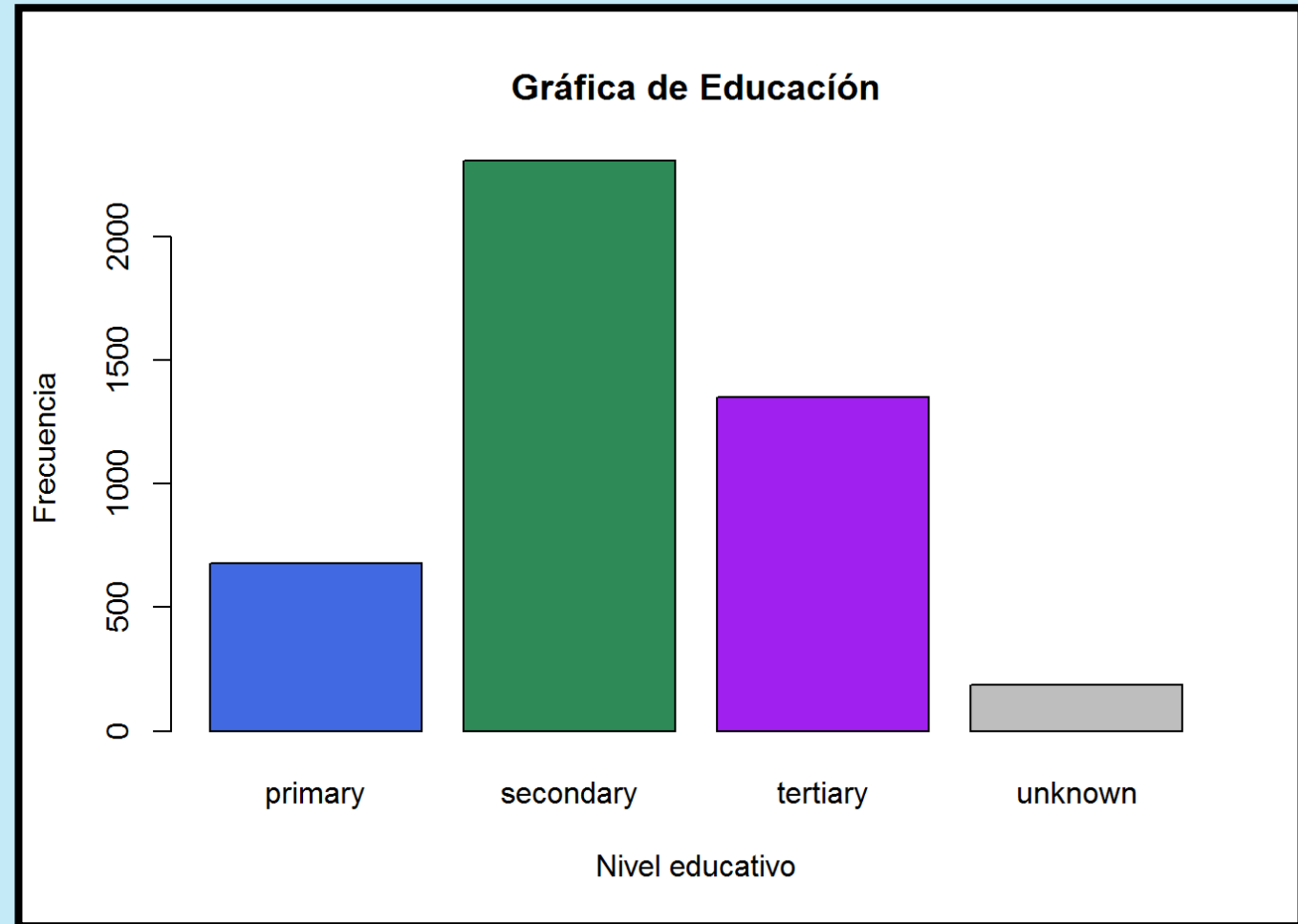
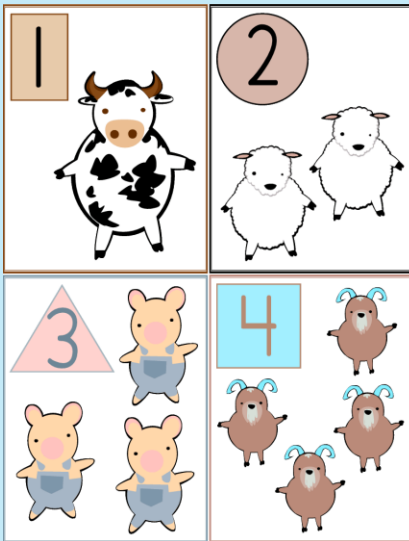
Histograma de Edad



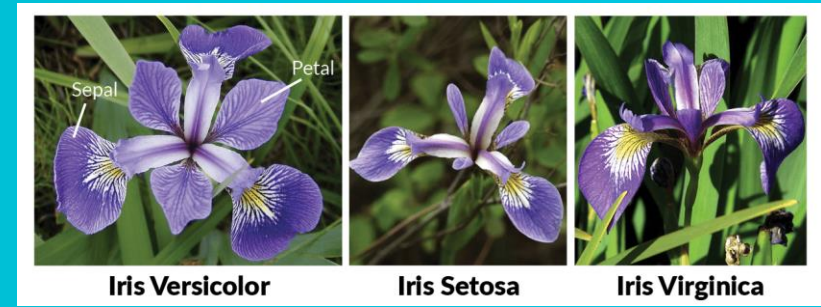
Gráficas de barras

- **Variable discreta:**
número de empleados
de una fábrica; número
de hijos.

El número de animales en una
granja (0, 1, 2, 3, 4, 5, 6, 7, ...)



Diagramas de dispersión



- La longitud de una pieza: 1.5, 2, 2.25, 3.15 cm.
- La altura de cinco amigos: 1.73, 1.82, 1.77, 1.69, 1.75 metros.
- El tiempo que demora un repartidor de comida en entregar un pedido: una hora; una hora y cuarto; una hora y media; media hora.

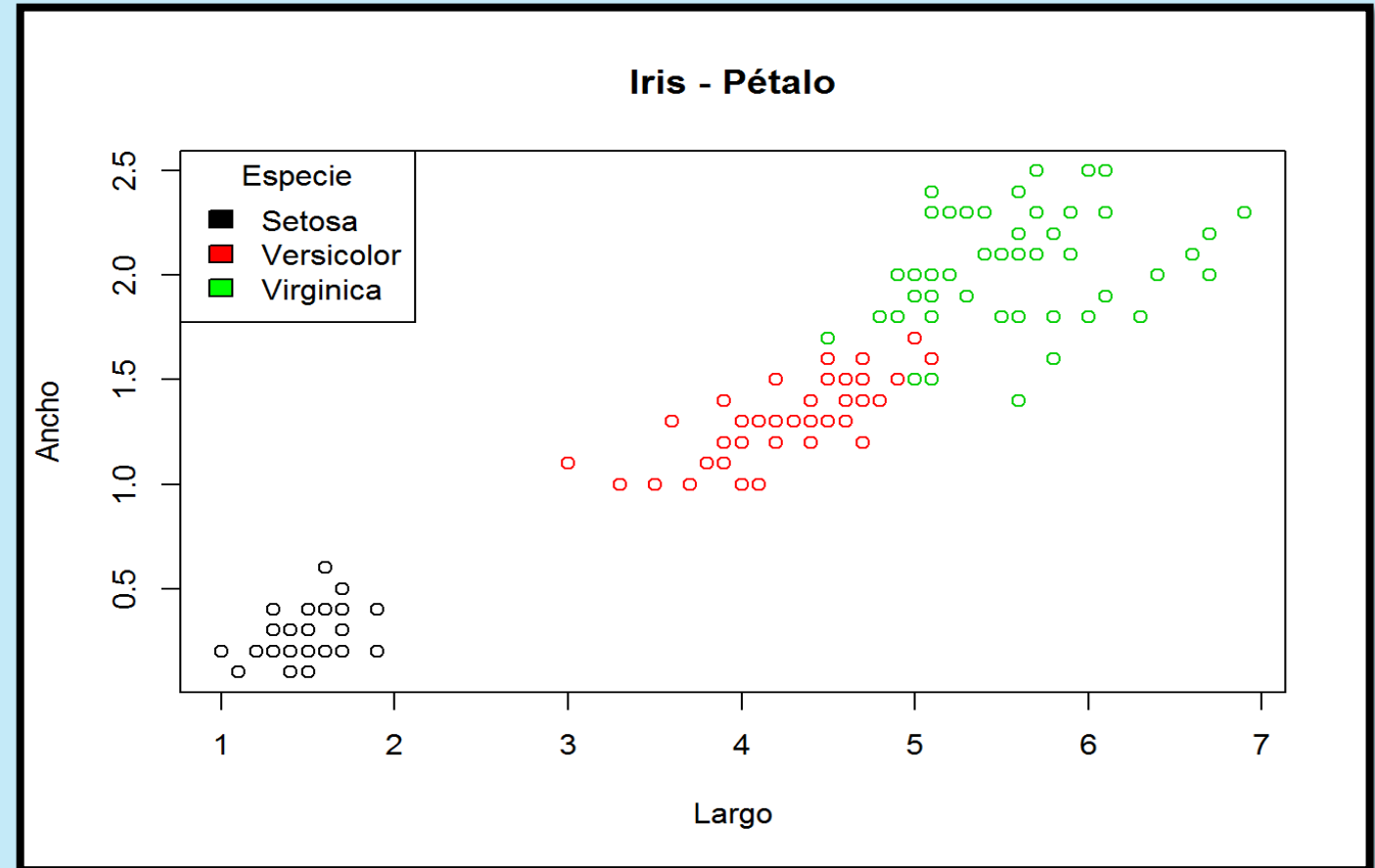
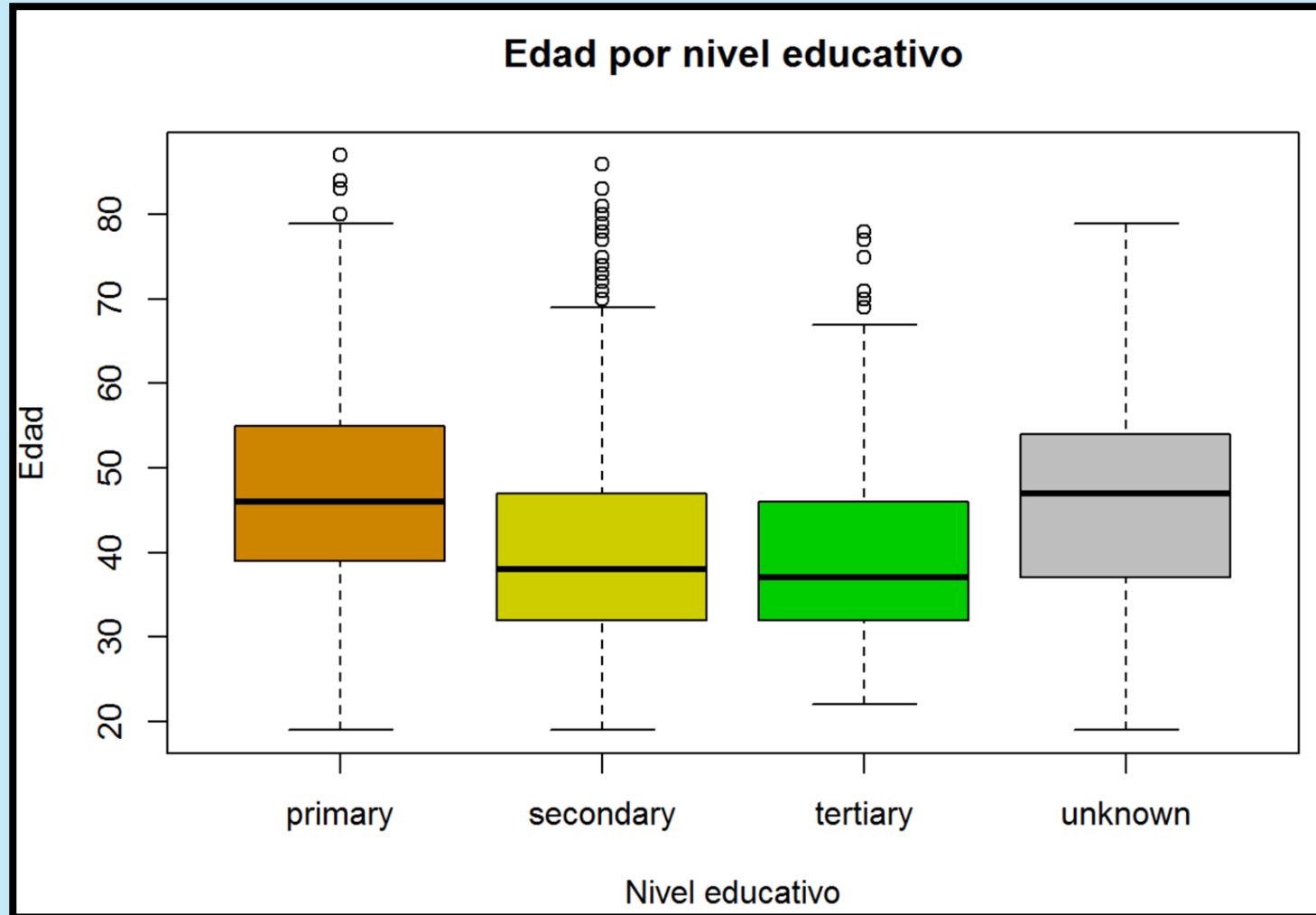
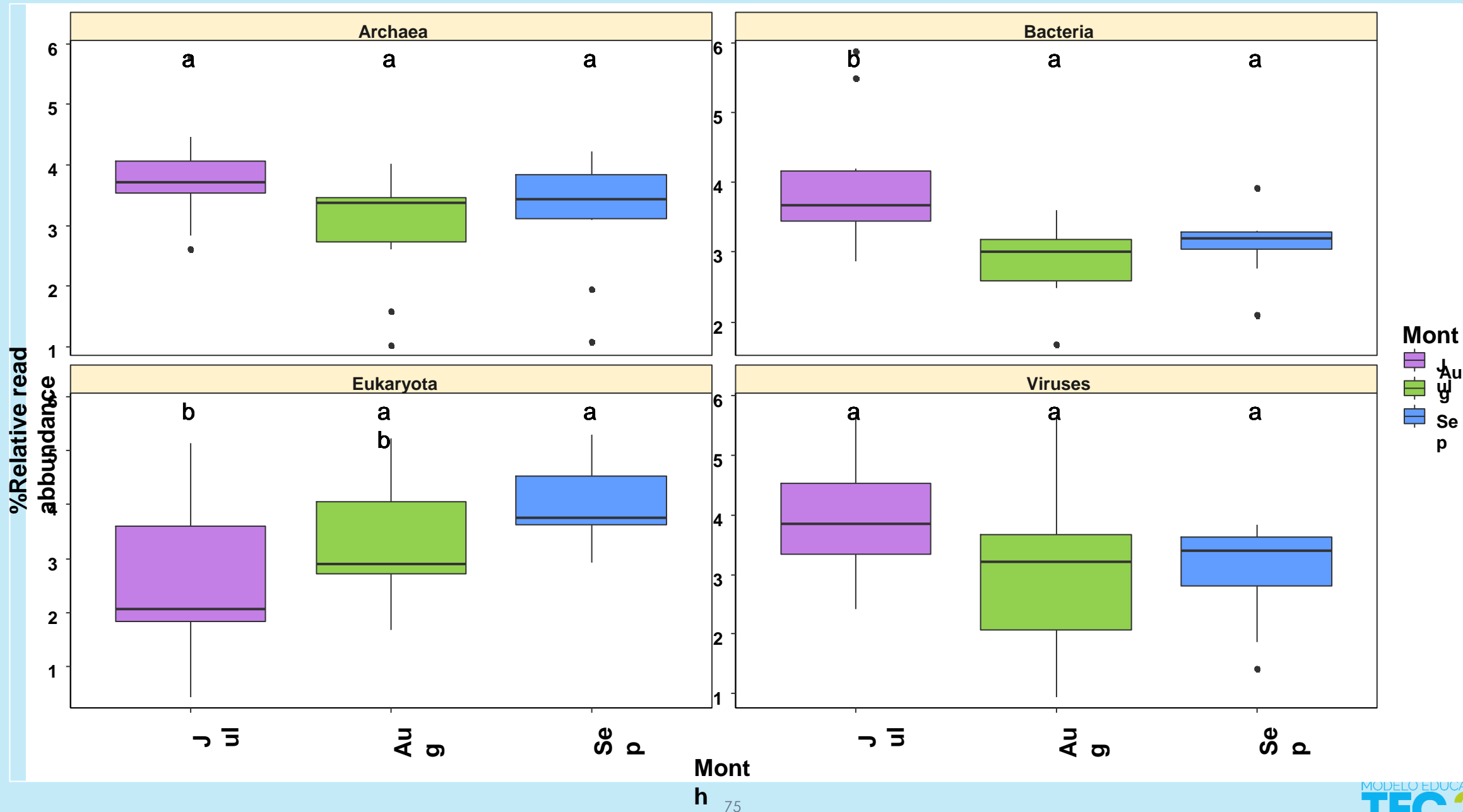


Diagrama de cajas





Tips en R

1. Control + Shift + F10: Ejecutar todo el código en el archivo actual.
2. Control + Shift + L: Limpiar tu consola
3. #: Poner un comentario
4. Ayuda sobre una función: Help(), ejemplo Help(max), saber sobre un paquete en específico ?max
5. Control + Enter: Ejecutar una línea de código seleccionada o la línea actual.
6. Control + Shift + M: Insertar un comentario en el código seleccionado o la línea actual.
7. Control + Shift + C: Comentar o descomentar una línea de código seleccionada o la línea actual.
8. Control + Shift + D: Insertar un signo de porcentaje para crear un chunk de código en un archivo de R Markdown.
9. Control + Shift + T: Insertar una sección de código de prueba en un archivo de R Markdown.
10. Control + Shift + R: Insertar una sección de resultados en un archivo de R Markdown.
11. Control + 1-9: Ir a la pestaña correspondiente en la parte superior de la ventana de RStudio (por ejemplo, Control + 1 para ir a la primera pestaña).
12. Control + Shift + F: Buscar y reemplazar texto en el archivo actual.
13. Control + Shift + N: Crear un nuevo script de R.
14. Control + Shift + W: Cerrar todos los archivos abiertos en RStudio.
15. Control + Shift + S: Guardar todos los archivos abiertos en RStudio.

Referencias

- Level (2017). How Big Companies Are Using R for Data Analysis. Recuperado en septiembre de 2017 de: <http://www.northeastern.edu/levelblog/2017/05/31/big-companies-using-r-data-analysis/>
- Microsoft (2014). Companies using R in 2014. Recuperado en septiembre de 2017 de: <http://blog.revolutionanalytics.com/2014/05/companies-using-r-in-2014.html>
- Bhalla, D. (2016) Companies using R. Recuperado en septiembre de 2017 de: <http://www.listendata.com/2016/12/companies-using-r.html>
- R FAQ. Recuperado en Septiembre de 2017 de: https://cran.r-project.org/doc/FAQ/R-FAQ.html#What-is-R_003f
- TIOBE Index for September 2017. Recuperado en Septiembre de 2017 de: <https://www.tiobe.com/tiobe-index/>
- Adesanya, T. (2017). A Gentler Introduction to Programming. Recuperado en Septiembre de 2017 de: <https://medium.freecodecamp.org/a-gentler-introduction-to-programming-707453a79ee8>

Tarea: 2 Quizzes

▾ Semana 1 (del 28 de marzo al 01 de abril)	Complete todos los ítems	+	⋮
Actividades iniciales de la modalidad presencial			⋮
Sesión presencial 1 28 de mar Ver			⋮
Conoce la situación problema Ver			⋮
Introducción al lenguaje R			⋮
Aprende sobre El lenguaje de programación R 31 de mar Ver			⋮
Sesión presencial 2 31 de mar Ver			⋮
Quiz en clase Introducción a R (Kahoot) 31 de mar 100 pts Entregar			⋮
Actividad en clase Ejercicios básicos de R 2 de abr 100 pts			⋮

Navegación: Límite de tiempo 30 min. 24 horas para resolverlo a partir de hoy

78

Código de ética: Límite de tiempo no hay. 2 intentos, 24 horas para resolverlo a partir de hoy

Quiz | Navegación en la plataforma



Especificaciones de entrega

- El quiz consta de 5 preguntas de opción múltiple y/o falso verdadero.
- Solo tendrás **1 oportunidad** para contestarlo.
- Cuentas con 30 minutos una vez iniciada la prueba. **Administra adecuadamente tu tiempo.**
- Cuando hayas terminado el examen haz clic en el botón **Enviar evaluación** que te aparecerá al final de la pantalla.
- Una vez que abras el quiz no podrás cancelarlo.
- Si ocurriese algún problema tecnológico fuera de tu control al momento de realizar la prueba debes reportarlo **DE INMEDIATO** de lo contrario no se te tomará en cuenta.



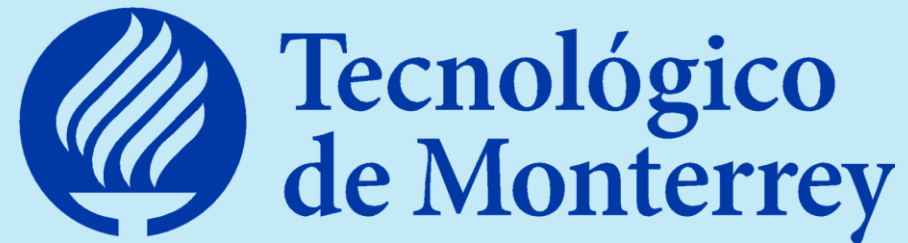
Evaluación y retroalimentación

Quiz | Código de ética



Especificaciones de entrega

1. Esta prueba es en línea.
2. El examen estará conformado por preguntas de opción múltiple.
3. En caso de que excedas el tiempo límite, la prueba **se bloquea automáticamente** y serás evaluado con lo que hayas contestado hasta ese momento.
4. Una vez que abras la prueba no podrás cancelarla. Tienes 2 oportunidades para realizar la prueba.
5. Tú puedes elegir el lugar en el que contestarás tu prueba, por lo que te recomendamos que busques un lugar adecuado donde tengas una buena conexión a Internet.
6. Cuando hayas terminado de responder a las preguntas haz clic en el botón **Enviar la evaluación**. Al finalizar esto te aparecerá la calificación obtenida.



Se prohíbe la reproducción total o parcial de esta obra por cualquier medio sin previo y expreso consentimiento por escrito del Instituto Tecnológico y de Estudios Superiores de Monterrey.

D.R.© Instituto Tecnológico y de Estudios Superiores de Monterrey, México. 2020
Ave. Eugenio Garza Sada 2501 Sur Col. Tecnológico C.P. 64849
Monterrey, Nuevo León | México

- <https://www.statmethods.net/>
- <http://www.cookbook-r.com/>
- <https://www.r-bloggers.com/>
- <https://stackoverflow.com/tags/r/info>
- <https://google.github.io/styleguide/Rguide.html>
- <https://bookdown.org/matiasandina/R-intro/>
- <https://bookdown.org/jboscomendoza/r-principiantes4/>