Resumen de comandos de R

Contents

Este notebook ha sido creado en RStudio, un entorno que incluye una consola para insertar código, historial, gráficos, paquetes y librerías.

Constantes y variables

En R, se usa <- para asignar valores a una variable. Los nombres de las variables pueden incluir letras, números, puntos y guiones bajos, sin embargo, siempre deben empezar con una letra.

```
# Asignandole 1 a la variable a
a<-1
a
```

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

Ayuda

Es posible obtener la documentación de una función digitando un signo de interrogación (?) al inicio de esta o escribiendo help(" ") con el nombre de la función dentro de las comillas. Al hacerlo, en la pestaña de Help se mostrará cómo utilizar dicha función, sus parámetros y ejemplos.

```
?sum()
# o help("sum")
```

La documentación de un paquete se puede obtener con:

```
help(package = "datasets")
```

Funciones

En R, una función posee la siguiente sintaxis: **nombre_funcion()**. Dentro de los paréntesis, van los argumentos de la función. Algunas funciones básicas definidas de R son:

- sum()
- mean()
- max()
- min()

Las funciones también pueden ser definidas por el usuario usando la siguiente sintaxis:

```
nombre_función <- function(argumentos) {
  # codigo
}
```

Paquetes

En R, un paquete es una colección con funciones que no están en R base. CRAN es el repositorio de paquetes oficial de R, los cuales se pueden instalar mediante `install.packages()`. Por ejemplo:

```r
install.packages("stats")
```

```
## Warning: package 'stats' is in use and will not be installed
```

Luego de instalar el paquete, las funciones de este se podrán utilizar después de ejecutar `library()` con el nombre del paquete dentro de la función.

```r
library(stats)
```

Cada vez que se inicia una nueva sesión y se requiera usar una función que pertenezca a un paquete, se debe ejecutar `library()`.

Para saber qué paquetes están instalados, se debe ejecutar `installed.packages()`, sin argumento.

Operadores

Aritméticos

Los operadores aritméticos de R base que se pueden utilizar con datos enteros y numéricos son:

- +: suma.
- -: resta.
- *: multiplicación.
- /: división.
- ^: exponencial.
- %%: módulo, devuelve el residuo.
- %*%: multiplicación entre matrices.

Ejemplo:

```r
3*5+2
```

```
## [1] 17
```

Relacionales

Los operadores relacionales se utilizan para comparar un valor con otro. Siempre devuelven TRUE o FALSE.

- : mayor que.
- =: mayor o igual que.
- <: menor que.
- <=: menor o igual que.
- ==: igual.
- !=: distinto.

Ejemplo:

```r
234 > 243
```

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

Lógicos

Los operadores lógicos se utilizan para crear condiciones. Devuelven TRUE o FALSE.

- &: y.
- |: o.
- !: not, negación lógica.

Ejemplo:

```r
# Al usar &, si uno de los valores es FALSE, devuelve FALSE
# Como 234 es menor que 243, devuelve FALSE
234 & 2000 > 243
```

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

```r
# Al usar |, si uno de los valores es TRUE, devuelve TRUE
# Como 2000 es mayor que 243, devuelve TRUE
234 | 2000 > 243
```

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

Tipos de datos

Los tipos de datos más comunes en R son:

1. integer: entero. Ejemplo: 1.
2. double: decimales. Ejemplo: 1.7
3. numeric: real. Ejemplo 4.5.
4. character: cadena de texto. Ejemplo: "Hola mundo"
5. factor: se utiliza para representar variables categóricas. Ejemplo: Categoría de productos como A, B, C.
6. logical: lógico, booleano. Ejemplo: FALSE.

Para que devuelva el tipo de dato de una variable, se usa `typeof`:

```r
typeof(1.5)
```

```
## [1] "double"
```

Se pueden convertir un tipo de dato en otro por medio de funciones que comienzan con `as.`.

1. `as.integer()`: convertir a entero.
2. `as.double()`: convertir a decimal.

3. `as.character()`: convertir a cadena de texto.
4. `as.logical()`: convertir a booleano.

Ejemplo:

```
# Convirtiendo la variable b en una cadena de texto
b<-456 # integer
b<-as.character(b)
typeof(b)
```

```
## [1] "character"
```

```
# Como b es un texto, devuelve el resultado entre comillas
b
```

```
## [1] "456"
```

Si al aplicar el `as.` no se puede convertir el dato al tipo de dato deseado, retornará un `NA`.

```
c <- "abuela"
c <- as.integer(c)
```

```
## Warning: NAs introducidos por coerción
```

Por otro lado, se puede determinar el tipo de dato de un dato con `is.`:

1. `is.integer()`: verifica si es entero.
2. `is.double()`: verifica si es decimal.
3. `is.character()`: verifica si es cadena de texto.
4. `is.logical()`: verifica si es booleano.

Ejemplo:

```
# Verificando si 432 es una cadena de texto.
is.character(432)
```

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

Condicionales

if

for

while

Estructuras de datos

Vector

Matriz

Array

Listas

Dataframe

Bibliografía

1. https://r-coder.com/inicio/
2. https://bookdown.org/jboscomendoza/r-principiantes4/