Data Science

¿TIPOS DE DATA

BASIC EN RSTUDIO

Data Science

BASIC DATA TYPES *

CHARACTER

El tipo de datos de caracteres representa texto y es fácil reconocerlo porque un dato siempre esta rodeado de comillas, simples o dobles. De manera convencional, nos referimos a este tipo de datos como cadenas de texto, es decir, secuencias de caracteres.
Este es el tipo de datos más flexible de R, pues una cadena de texto puede contener letras, números, espacios, signos de puntuación y símbolos especiales.

LOGICAL

Los datos de tipo lógico sólo tienen dos valores posibles: verdadero (TRUE) y falso (FALSE). Estos se utilizan para determinar si ciertas condiciones son verdaderas o no, debemos de tener en cuenta que también puede usar “T” para VERDADERO y F para FALSO. Además, este tipo de dato es, generalmente, el resultado de operaciones relacionales y lógicas, son esenciales para trabajar sobre todo en Algebra (#-operadores) y este tipo de dato sólo admite dos valores específicos, por tanto es considerado como el tipo de dato más restrictivo de R.

ENTEROS Y NÚMEROS

Enteros: Los datos enteros representan números enteros, sin una parte decimal o fraccionaria, que pueden ser usados en operaciones matemáticas.

Números: Por su parte los números se dividen en subtipos adicionales llamados dobles o float (flotantes) . Este nombre se debe a que, en realidad, son números de doble precisión, pues tienen una parte entera y una fraccionaria decimal, y son llamados float debido a que se usa un punto flotante para su representación computacional, entonces nos centraremos principalmente en el subtipo doble, que es el subtipo predeterminado.

```
print(1)
print(1.5)
print(-1)
```

THE CLASS() FUNCTION

class() es similar a print() en que ambas son funciones que imprimen cosas en la consola; sin embargo, class() imprimirá qué tipo de datos tiene entre paréntesis, mientras que print() simplemente imprime los datos en sí. Pruebe las siguientes instrucciones class() en RStudio:

```
class(1)
class("1")
class(TRUE)
```

Data Science

¿CÓMO CREAR VARIABLES EN R STUDIO

www.datascience.pe

Data Science

En esta sección veremos cómo crear las variables en R. Por un lado, algunos consejos y mejores prácticas a la hora de poner nombres a variables y utilizar el # para comentar y veremos cómo asignar variables en un script.
Y por otro lado, aprenderemos a realizar operaciones aritméticas en la consola de R. Por último, utilizaremos distintos tipos de datos en R.

Data Science

¿QUÉ ES UNA VARIABLE?

Las variables en R son similares a las variables en álgebra, donde una variable representa algún tipo de datos. En álgebra, las variables normalmente solo almacenan números. Sin embargo, en R, puede almacenar cualquier tipo de datos en una variable. Para hacerlo, creamos una variable como x seguida de los símbolos “<-” o “=“ (también conocido como el operador de asignación), y finalmente los datos que desea almacenar, como 2. Luego, para ver lo que está almacenado en la variable x, puede usar la función print().

Es un espacio en la computadora donde guardamos un objeto, algo así como una caja con un nombre para guardar cosas.
Cuando comience a asignar un valor a una variable, verá su conexión dentro de la pestaña Entorno de RStudio. Esto ayuda a realizar un seguimiento de todas las variables que está utilizando y los valores que se almacenan en ellas.


```
x <- 2
print(x)
```

Data Science

NOMBRES DE VARIABLES

Las variables se pueden etiquetar con un solo carácter, como x, o con varios caracteres, como “Perimetro”. También puede incluir una combinación de números, guiones bajos y puntos. Sin embargo, una variable no puede comenzar con un guion bajo (_) o un número. Si comienza con un punto (.), no puede continuar con un número. Estos son algunos ejemplos de nombres de variables aceptables e inaceptables:

```
# Legal variable names:
myvar <- "John"
my_var <- "John"
myVar <- "John"
MYVAR <- "John"
myvar2 <- "John"
.myvar <- John
# Illegal variable names:
2myvar <- "John"
my-var <- "John"
my var <- "John"
_my_var <- "John"
my_v@ar <- "John"
TRUE <- "John"
```

Por qué son útiles las variables

Las variables son buenos cuando vamos a reutilizar un valor particular varias veces. Por ejemplo, si quiero usar “2” para ayudarme a determinar la circunferencia de un círculo, puedo almacenarlo dentro de una variable como x. Entonces, puedo seguir usando x para ayudarme a calcular la circunferencia de varios círculos con diferentes diámetros.

Data Science

COMENTARIOS

Los comentarios son líneas de código que el programa ignora. Para comentar una línea de código para que el programa los ignore, coloque un símbolo # delante de él. Teniendo en cuenta que el símbolo # le dice al programa que comience a ignorar todo el código hasta la siguiente línea. Los comentarios son de mucha ayuda cuando se usan como recordatorios o instrucciones. Por ejemplo,
agrega tu código debajo de esta línea y
agrega tu código arriba de esta línea
se usan como comentarios para indicarnos las instrucciones para colocar código al espacio entre esas líneas comentadas. Agregue las siguientes líneas de código en RStudio y fuente del programa.

```
x <- "Today is " # assigns character to variable x
print(x)
print("Monday")
# print("Tuesday")
# print("Wednesday")
# print("Thursday")
# print("Friday")
```

Data Science

OPERACIONES ARITMÉTICAS *

R es una calculadora, puede realizar operaciones aritméticas usando ciertos símbolos en R. Como por ejemplo:

- + representa suma
- - representa la resta
- * representa la multiplicación
- / representa división
- ^ y ** representan exponente
- %% representa módulo
- %/% División entera

Tenga en cuenta que los operadores aritméticos no funcionarán con el tipo de datos de caracteres. Solo trabajan con números y lógicos. Además las operaciones aritméticas nos pueden ayudar a

- Crear nuevas variables para el análisis
- Realizar cálculos complejos
- Resolver un acertijo de redes sociales como este

¿Cuánto es el resultado de…?

1+1-1*(1+1-1)/1+1*(-1) = ?

Sólo el 5% puede resolverlo

Data Science

ADICIÓN

Podemos agregar ciertos tipos de datos, como números, usando el operador +. Por ejemplo,

```
print(1 + 2)
```

porque 1 + 2 = 3. Además, también se puede agregar lógicos juntos. Por defecto, VERDADERO es equivalente a un valor de 1 y FALSO es equivalente a 0. Por ejemplo,

```
print(TRUE + TRUE + FALSE)
```

porque 1 + 1 + 0 = 2. En una nota final, también puede sumar números y lógicos juntos. Por ejemplo,

```
print(3 + TRUE + FALSE + TRUE + 5)
```

Porque 3 + 1 + 0 + 1 + 5 = 10.

Data Science

SUSTRACCIÓN

Al igual que la suma, puede restar valores numéricos y lógicos utilizando el operador -.

```
print(5 - 2)
print(-6 - 3)
print(FALSE - TRUE)
print(TRUE - -3 - 7)
```

MULTIPLICACIÓN

A continuación se muestran ejemplos del operador de multiplicación *.

```
print(5 * 2)
print(-6 * 3)
print(FALSE * TRUE)
print(TRUE * -3 * 7)
```

Tenga en cuenta que Inf significa "infinito" o "infinito negativo" y también es el resultado de dividir por cero (0).

EXPONENT

A continuación se muestran ejemplos de los operadores exponentes ^ y **. Tengamos en cuenta que no hay diferencia entre los dos operadores. Puedes elegir el que prefieras.

```
print(2 ^ 2)
print(-2 ** 3)
print(FALSE ^ TRUE)
print(TRUE ** -3 ^ 9)
```

MÓDULO

Módulo (%%) es similar a la división porque ambos implican dividir datos. Sin embargo, el módulo devolverá el resto del cociente en lugar del cociente en sí. Por ejemplo, 5 dividido por 2 es 2 con un resto de 1. Cuando realiza el módulo usando 5 %% 2, obtiene el resto de 1 a cambio.

```
print(5 %% 2)
print(-6 %% 3)
print(-1 %% -3)
print(24 %% 5 %% 5)
```

Guía:

```
print(5 %% 2) = 5 divide 2 = 2 remainder 1 = 1
print(-6 %% 3) = -6 divide 3 = -2 remainder 0 = 0
print(-1 %% -3) = -1 divide -3 = 0 remainder -1 = -1
print(24 %% 5 %% 5) = 24 divide 5 = 4 remainder 4 = 4
4 divide 5 = 0 remainder 4 = 4
```

Data Science

DIVISIÓN ENTERA*

Cuando realiza una división de enteros con el operador %/%, el sistema redondeará efectivamente al número entero más pequeño, independientemente de los valores decimales que existan. Por ejemplo, 5 dividido por 2 es 2,5, pero con la división de enteros, el sistema devolverá solo 2, ya que 2 es menor que 3. Una vez más, la división de enteros no tiene en cuenta los valores decimales. Siempre se redondeará hacia abajo al número entero más pequeño.

```
print(5 %/% 2)
print(-6 %/% 4)
print(-1 %/% -3)
print(24 %/% 5 %/% 3)
```

Guía:

```
print(5 %/% 2) = 2.5 = 2
print(-6 %/% 4) = -1.5 = -2 (-2 is smaller than -1)
print(-1 %/% -3) = 0.3333333 = 0
print(24 %/% 5 %/% 3) = 24 / 5 = 4.8 = 4
                        4 / 3 = 1.3333333 = 1
```

Data Science

PENDAS*

Tenga en cuenta que las reglas de PEMDAS aún se aplican cuando se usan operadores aritméticos en R. PEMDAS especifica que las operaciones deben realizarse de izquierda a derecha en este orden de prioridad:

- paréntesis ()
- Exponentes ^ y **
- Multiplicación *, división /, módulo %% y división entera %/%
- Suma + y resta -

Tenga en cuenta que el módulo y la división de enteros tienen el mismo nivel de prioridad que la multiplicación y la división. Por ejemplo,

```
print(2 + 3 * 5 - 7^2 %% 4 + (5 / 2))
```

Guía:

```
print(2 + 3 * 5 - 7^2 %% 4 + (5 / 2))

parentheses: 5 / 2 = 2.5
exponent: 7^2 = 49
multiplication: 3 * 5 = 15
modulo: 49 %% 4 = 1
addition: 2 + 15 = 17
subtraction: 17 - 1 = 16
addition: 16 + 2.5 = 18.5
```

Data Science

LOGICAL OPERATORS *

Además de las operaciones aritméticas, también puede realizar operaciones lógicas usando ciertos símbolos en R. Aquí hay una lista:

- < representa menor que
- <= representa menor o igual que
- > representa mayor que
- >= representa mayor o igual que
- == representa exactamente igual a
- != representa no igual a
- ! representa “no” o negación
- | representa Disyunción "o"
- & representa conjunción “y”

Las expresiones lógicas se evalúan como VERDADERO o FALSO.

Data Science

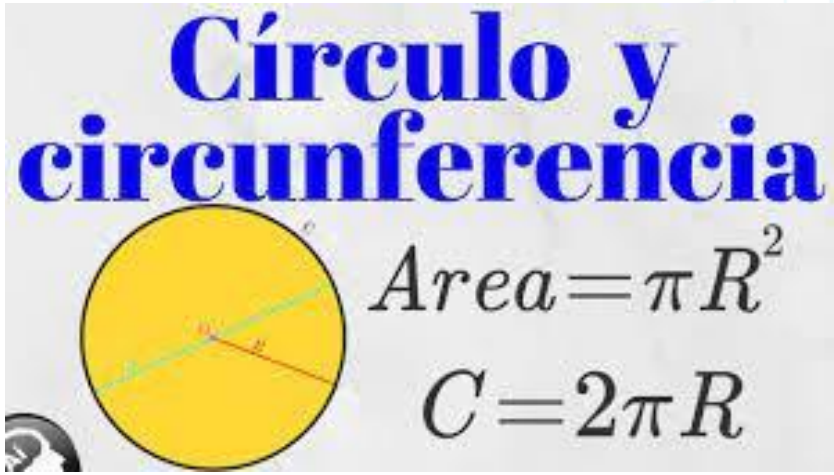
CREAR VARIABLES R *

Una variable es un espacio en la computadora donde guardamos un objeto.

- Las variables, son útiles cuando vamos a:

  - ✔ Utilizar los mismos datos varias veces
  - ✔ Repetir operaciones aritméticas
  - ✔ Estructurar el análisis

Usar variables nos ayuda a optimizar el tiempo



Data Science

GRACIAS

www.datascience.pe