R 2019

03/04/2019

REPASO DE LA CLASE ANTERIOR

'TYPES': TIPOS DE DATOS

- **'logical'**: TRUE o FALSE
- **'integer'**: enteros: ..., -1, 0, 1, ...
- **'double'**: irracionales: 3.1415926
- **'character'**: alfanuméricos: "pi"
- *'complex': 1+i10*
- *'raw': 48 65 6c 6c 6f*

OBJETOS R: VECTORES

Existen dos clases de vectores:

1. Vectores 'atómicos' (*atomic vectors*), todos los elementos del mismo tipo
    - puede haber de los 6 tipos: 'logical', 'integer', 'double', 'character', 'complex' y 'raw'
    - Integer y double son tratados como 'numeric'
    - No hay escalares en R, si no vectores de longitud 1
2. Vectores recursivos - *Listas* :
    - Pueden a su vez contener listas (por eso lo de recursivos)
    - *data frames* son caso especial, cuando los vectores que la componen son de igual longitud

La principal diferencia entre los vectores atómicos y las listas es que los primeros son homogéneos, o sea todos sus elementos son del mismo tipo, mientras que en las listas no es necesario.

PROPIEDADES DE LOS VECTORES

Las propiedades más importantes de los vectores son:

1. Que *tipo* de vector es. `typeof()`
2. Que *longitud* tiene. `length()`
3. Cuales *atributos* tiene asociados. `attributes()`

Los atributos son metadata arbitraria que se puede asociar a cualquier objeto R. Se determinan y consultan con *attr()* para alguno en particular y con *attributes()* se consultan todos los que el objeto tenga. Los tres atributos más importantes se obtienen con *names()*, *class()* y *dim()*.

LISTAS

Sus elementos pueden tener cualquier tipo, longitud (dimensión!) o atributos, incluyendo otras listas o funciones ¯_(ツ)_/¯

```
x <- list(1, 2, 1:100)
x
## [[1]]
## [1] 1

## [[2]]
## [1] 2

## [[3]]
##  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##  [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##  [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##  [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##  [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##  [91]  91  92  93  94  95  96  97  98  99 100

str(nasa)
## List of 2
##  $ mets:List of 7
##   ..$ cloudhigh  : num [1:24, 1:24, 1:12, 1:6] 26 20 16 13 7.5 8 14.5 19.5 22.5 21 .
##   ..$ cloudlow   : num [1:24, 1:24, 1:12, 1:6] 7.5 11.5 16.5 20.5 26 30 29.5 26.5 27
```

VECTORES "AUMENTADOS"

- Data frames (y tibbles) sobre 'lists'
- Factores, construidos sobre vectores 'integer'
- Dates y date-times, sobre vectores 'numeric'

DATA FRAMES

Un data frame es una lista de vectores de igual longitud. Su estructura es 2d, con lo cual tiene cosas en común con listas y con matrices. Se le pueden aplicar `names()`, `colnames()` y `rownames()`.

Con `length()` obtenemos la longitud del data frame, y como es la cantidad de elementos de la lista, es igual a `ncol()`.

```
df <- data.frame(x = 1:3, y = c("a", "b", "c"))
str(df)
identical(length(df), ncol(df))
```

FACTORES

Los factores son vectores **no-atómicos** 'aumentados', usados para representar variables categóricas (nominales). Estos datos pueden tomar sus valores de un conjunto fijo de elementos. Internamente son representados por enteros, y R les asigna un atributo 'levels'.

```
x <- factor(c("ab", "cd", "ab"), levels = c("ab", "cd", "ef"))
typeof(x)
#> [1] "integer"
attributes(x)
#> $levels
#> [1] "ab" "cd" "ef"
#> str(x)
#> Factor w/ 3 levels "ab","cd","ef": 1 2 1
#> $class
#> [1] "factor"
```

DATES Y DATE-TIMES

Lo veremos más adelante.

SUBSETTING

Subsetting se refiere a un conjunto de métodos para acceder a partes de objetos en R.

`[` es uno de los operadores más frecuentes para hacer *subsetting*. Si `x` es un vector atómico, `x[a]` es el elemento `a` del vector `x`. Hay distintas maneras de usar `[`.

USANDO VECTORES DE ENTEROS POSITIVOS O NEGATIVOS

Un vector numérico con enteros, todos positivos, todos negativos, o cero.

```
x <- c("one", "two", "three", "four", "five")
x[c(3, 2, 5)]
#> [1] "three" "two"   "five"

x[c(1, 1, 5, 5, 5, 2)]
#> [1] "one"  "one"  "five" "five" "five" "two"

x[c(-1, -3, -5)]
#> [1] "two"  "four"

# caso particular, el cero, devuelve un vector vacio
x[0]
## numeric(0)
```

USANDO VECTORES DE ELEMENTOS LÓGICOS

Subsetting con un vector lógico devuelve solo los valores de correspondientes a TRUE. Como *filter*, se usa frecuentemente en conjunto con *expresiones lógicas*.

```
x <- c(10, 3, NA, 5, 8, 1, NA)

# Todos los valores que no son NA de x
x[!is.na(x)]
#> [1] 10  3  5  8  1

# Todos los valores pares (o NAs!) de x
x[x %% 2 == 0]
#> [1] 10 NA  8 NA
```

USANDO VECTORES DE CARACTERES PARA DEVOLVER ELEMENTOS CON NOMBRES

Si tenemos un vector con sus elementos, o un data frame con sus columnas, con nombres, podemos acceder a los elementos así:

```
x <- c(abc = 1, def = 2, xyz = 5)
x[c("xyz", "def")]
#> xyz def
#>   5   2

mtcars[, c("mpg", "disp")]
##                    mpg  disp
## Mazda RX4         21.0 160.0
## Mazda RX4 Wag     21.0 160.0
## Datsun 710        22.8 108.0
## ...
```

NO USANDO NADA!

La manera más simple de hacer *subsetting* es no usando nada, `x[]`, que devuelve el objeto original. Si es 2d, podemos dejar vacía una de las dimensiones, por ej. `x[1, ]` para obtener una fila (con todas sus columnas), o x[, -1] que selecciona todas las filas columnas menos la indicada.

Para el caso de data frames y matrices, al hacer subsetting es posible usar `drop = FALSE` para preservar las dimensiones del objeto original.

OTROS OPERADORES DE SUBSETTING: `[[` Y `$`

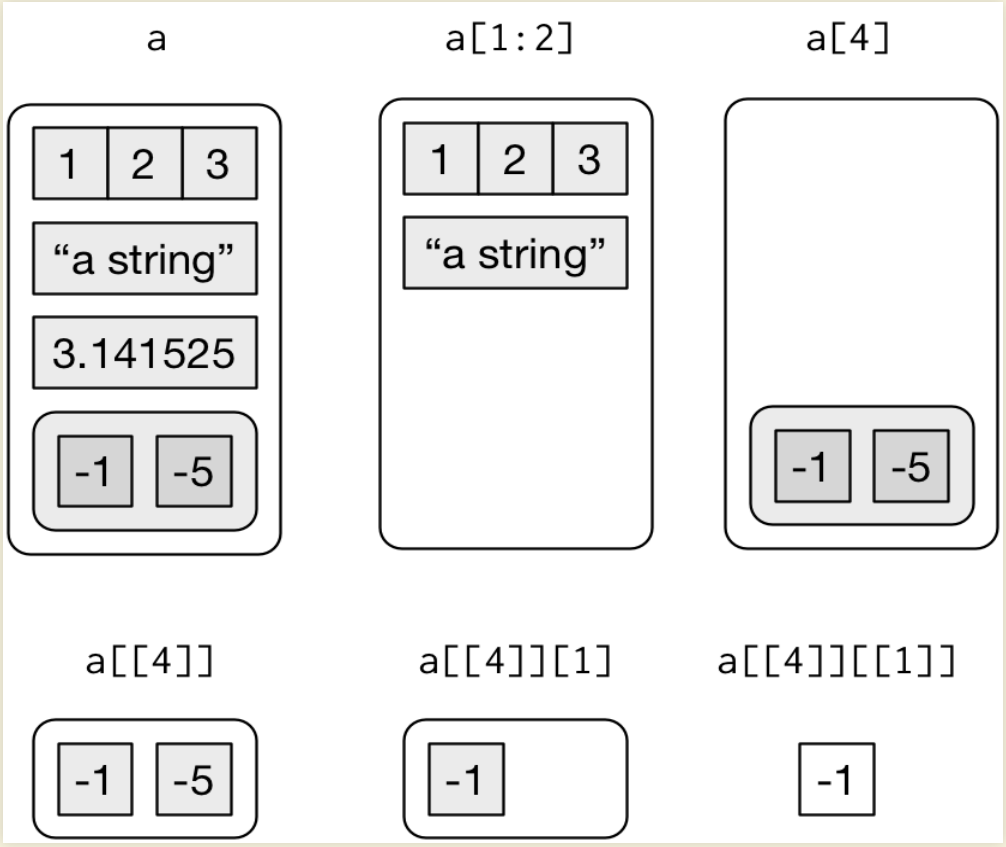
Para listas, necesitamos `[[` para acceder al contenido de los elementos de la lista, porque aplicando `[` a una lista siempre devuelve otra lista (más exactamente, una sub-lista).

En cambio, `[[` saca un nivel de la jerarquía de la lista y puede devolver cualquier tipo de objeto, dependiendo del elemento siendo accedido.

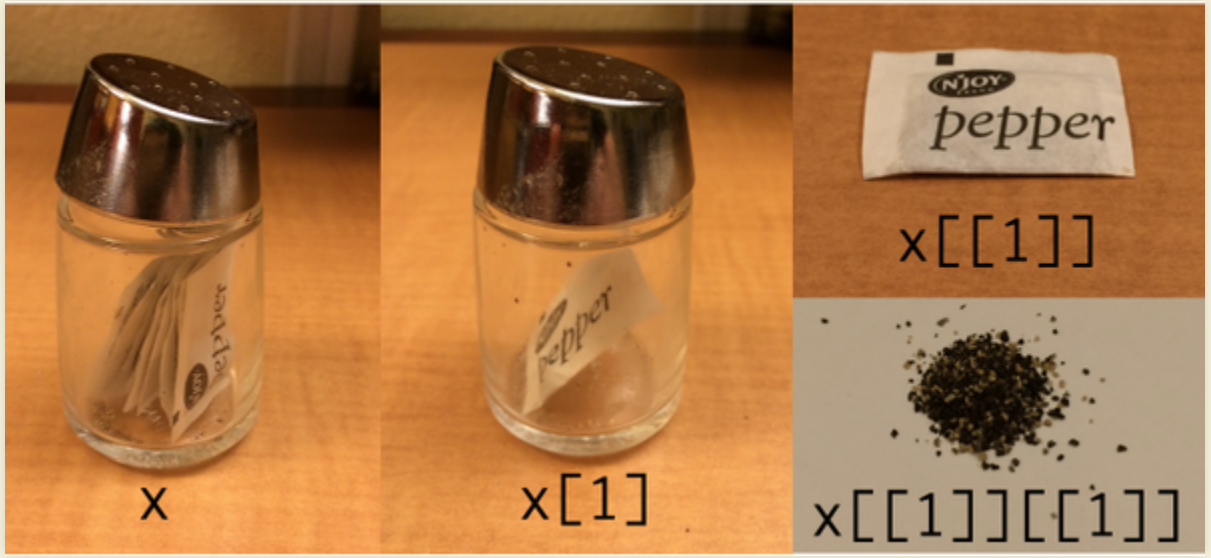
El operador `$` es una abreviación para obtener elementos *nombrados* de una lista. Se usa parecido a `[[` solo que no hace falta usar las comillas.

Como los data frames son listas de columnas de igual longitud, podemos acceder a sus columnas usando `mtcars[[1]]`, `mtcars[["cy"]]` o `mtcars$cy`. Estos dos últimos son equivalentes.

DIFERENCIA ENTRE [Y [[PARA LISTAS



INDEXADO DE LISTAS: '[[' VS. '['



crédito - Hadley Wickham: http://t.co/YQ6axb2w7t

PRÁCTICA 6

Descargar práctica 6.