Tipos de datos en R

Code

Variables

Usar el simbolo `<-` o `=` para asignar variables. El estilo recomendado en R es el primero.

Hide

```
x <- 0
x
```

Hide

```
x <- "hola"
x
```

La unidad mas basica de procesamiento en R son los vectores que se obtienen concatenando elementos con la funcion `c()` y separando con comas. Son como las lista de Python pero todos los elementos deben ser del mismo tipo.

Hide

```
x <- c(1,2,3,1,2,3)
x
```

Hide

```
y <- c("hola", "adios")
y
```

Todos los elementos de un vector deben ser del mismo tipo, pero R adivina el tipo si dan elementos de distintos tipos. Por ejemplon en el siguiente ejemplo los numeros se vuelven strings, llamados characters en R.

Hide

```
x <- c(1,2,"hola")
class(x)
```

Los vectores en R permiten hacer aritmetica puntual, es decir, pueden sumar, multiplicarse y dividirse elemento a elemento entre vectores del mismo tamano.

Hide

```
x <- 1:4 # forma rapida de c(1,2,3,4)
y <- c(-1,-2,-3,-4)
```

Hide

```
x + y
```

Hide

```
x * y
```

Hide

```
x / y
```

Los tipos de datos en R

Los tipos de datos en R estan basados en los tipos de datos “estadisticos”. La unidad basica en R son los vectores. Un dato solo es un vector de longitud “1”. Los posibles tipos de vectores son:

- character (strings/texto)
- numeric (datos continuos numericos)
- integer (enteros)
- logical (TRUE/FALSE)
- (el raro) factor (el uso es como variable categorica, se especifican las posibles categorias)
- (tambien raro) ordered (es un factor pero las categorias tienen un orden)

Ejemplos:

Hide

```
x <- 10:15
print(x)
```

```
[1] 10 11 12 13 14 15
```

Hide

```
print(class(x))
```

```
[1] "integer"
```

Hide

```
print(class(3.1))
```

```
[1] "numeric"
```

Hide

```
print(class(c(0.0, 0.1)))
```

```
[1] "numeric"
```

Hay recoversion automatica

Hide

```
x <- c(1,2) + 2.5
print(x)
```

```
[1] 3.5 4.5
```

Hide

```
print(class(x))
```

```
[1] "numeric"
```

Listas

Las listas, a diferencias los vectores, pueden tener elementos de distintos tipos. Se construyen con la funcion `list` , pueden contener vectores, o cualquier tipo vector en sus entradas.

Hide

```
mylist <- list(1:10, letters[1:5], c(TRUE, TRUE, FALSE))
print(mylist)
```

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

[[2]]
[1] "a" "b" "c" "d" "e"

[[3]]
[1]  TRUE  TRUE FALSE
```

Una diferencia importante con los vectores, es que mientras los elementos de un vectores se acceden con corchetes simples, los elementos de una lista con doble corchete.

Hide

```
mylist[[2]]
```

```
[1] "a" "b" "c" "d" "e"
```

Las listas pueden asignar nombres a cada uno de sus elementos.

Ahora puedo acceder a cada elemento de la lista por su nombre, ya sea usando dobles corchetes, o el signo de pesos.

Hide

```
mylist2$letras
```

```
[1] "a" "b" "c" "d" "e"
```

Hide

```
mylist2[["enteros"]]
```

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

DataFrames o Bases de Datos

Un caso particular de listas es un data frame. Pero es el mas importante y todo R esta disenado para funcionar con data frames. Son listas donde todos los elementos son vectores y todos son del mismo tanmano, es como un “array” con muchos tipos datos, y cada elemento de la lista se interpreta como COLUMNA.

Hide

```
mydata <- data.frame(
  peso = c(76, 66, 70), #kilogramos
  altura = c(173, 170, 155), #centrimetros
  sexo = c("H", "M", "M")
)
mydata
```

peso <dbl>	altura <dbl>	sexo <fctr>
76	173	H
66	170	M
70	155	M

3 rows

Hide

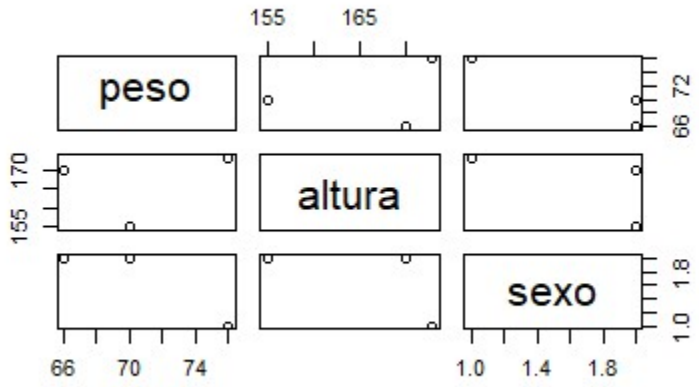
```
print(class(mydata))
```

```
[1] "data.frame"
```

Hay muchos metodos hechos ya para data.frames

Hide

```
pairs(mydata)
```



Acceden a las variables es igual que con listas

Hide

```
mydata$sexo
```

```
[1] H M M
Levels: H M
```

Por default los data.frames se crean los strings como factores, hay dos formas de cambiar ese comportamiento: 1) Usando la opcion `stringsAsFactors = FALSE` al crear el data.frame 2) O haciendo una conversion posterior con `as.character()`

Hide

```
print(class(mydata$sexo))
```

```
[1] "factor"
```

Hide

```
print(class(as.character(mydata$sexo)))
```

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

Los nombres de los data.frames (y de las listas) se accede (y cambian) con la funcion names

Hide

```
names(mydata)
```

```
[1] "peso"   "altura" "sexo"
```

Hide

```
names(mydata) <- c("Peso", "Altura", "Sexo")
```

Hide

```
mydata
```

Peso <dbl>	Altura <dbl>	Sexo <fctr>
76	173	H
66	170	M
70	155	M

3 rows

Los data.frames a diferencia de las listas, tambien tienen row.names,

Hide

```
row.names(mydata) <- paste("Sujeto", 1:3)
mydata
```

	Peso <dbl>	Altura <dbl>	Sexo <fctr>
Sujeto 1	76	173	H
Sujeto 2	66	170	M
Sujeto 3	70	155	M

3 rows

Y pueden acceder a un dato por fila y variable

Hide

```
mydata["Sujeto 1", "Peso"]
```