

1.2_intro

Argomenti

- **Oggetti**
 - tipologie di oggetti
 - creare/nominare un oggetto
 - tipologie di dato
- **Operatori**
 - matematici
 - relazionali
 - logici

Oggetti

Tutto quello che possiamo creare in R viene definito oggetto (e.g., numeri, vettori, matrici, funzioni).

```
1 numero = 4; numero
```

```
[1] 4
```

```
1 vettore = c(1,2,3,4); vettore
```

```
[1] 1 2 3 4
```

```
1 matrice = matrix(nrow = 2, ncol = 2, data = vettore); matrice
```

```
      [,1] [,2]  
[1,]     1     3  
[2,]     2     4
```

Funzioni

Tutto quello che facciamo in R è chiamare **funzioni** su oggetti.

Le funzioni ci permettono di **creare** e **modificare** oggetti.

```
1 matrice[1,2] = vettore[1]  
2 matrice
```

```
      [,1] [,2]  
[1,]    1    1  
[2,]    2    4
```

```
1 meanM = mean(matrice)  
2 meanM
```

```
[1] 2
```

Creare/nominare oggetti

Gli oggetti si possono creare e tramite il comando `<-` oppure `=`

```
1 x1 = 3 # nome = oggetto
2 x1
```

```
[1] 3
```

```
1 x2 <- 3 # nome <- oggetto
2 x2
```

```
[1] 3
```

```
1 x1 == x2 # i due oggetti sono identici?
```

```
[1] TRUE
```

Voi potete scegliere il comando che preferite, l'importante è essere consistenti.

Regole sulla denominazione di oggetti

- Deve iniziare con una lettera e può contenere lettere, numeri, underscore (_), o punti (.).
- Potrebbe anche iniziare con un punto (.) ma in tal caso non può essere seguito da un numero.

```
1 .3 = 3
```

```
Error in 0.3 = 3: invalid (do_set) left-hand side to assignment
```

```
1 .x = 3
```

- Non deve contenere caratteri speciali come #, &, \$, ?, etc.
- Non deve essere una parola riservata ovvero quelle parole che sono utilizzate da R con un significato speciale (? reserved).

```
1 function = 3
```

```
Error in parse(text = input): <text>:1:10: unexpected '='  
1: function =  
      ^
```

```
1 TRUE = 3
```

```
Error in TRUE = 3: invalid (do_set) left-hand side to assignment
```

```
1 if = 3
```

```
Error in parse(text = input): <text>:1:4: unexpected '='  
1: if =  
    ^
```

Convenzioni

Ci sono alcuni nomi che non sono proibiti ma sono sconsigliati

```
1 T
```

```
[1] TRUE
```

```
1 TRUE
```

```
[1] TRUE
```

```
1 T == TRUE
```

```
[1] TRUE
```

```
1 T = 1; T
```

```
[1] 1
```

```
1 T == TRUE
```

```
[1] TRUE
```

```
1 sum(2, 3)
```

```
[1] 5
```

```
1 sum = 4; sum
```

```
[1] 4
```


L'uso di “.” nei nomi degli oggetti (ad esempio, “my.data”) va bene in **R** ma non è consentito in **Python**, dove “.” fa parte della sintassi del linguaggio.

Tra i diversi linguaggi, le *convenzioni di denominazione* per i nomi di variabili più lunghi e composti da più parole privilegiano **snake_case** (ad esempio, “my_data”) o **camelCase** (ad esempio, “myData”), e **abbreviazioni** dove appropriato (ad esempio, “unipdData” meglio di “university_of_padova_dataset”).

R è case-sensitive

```
1 Nome = "Margherita"  
2 nome = "margherita"  
3  
4 Nome
```

```
[1] "Margherita"
```

```
1 nome
```

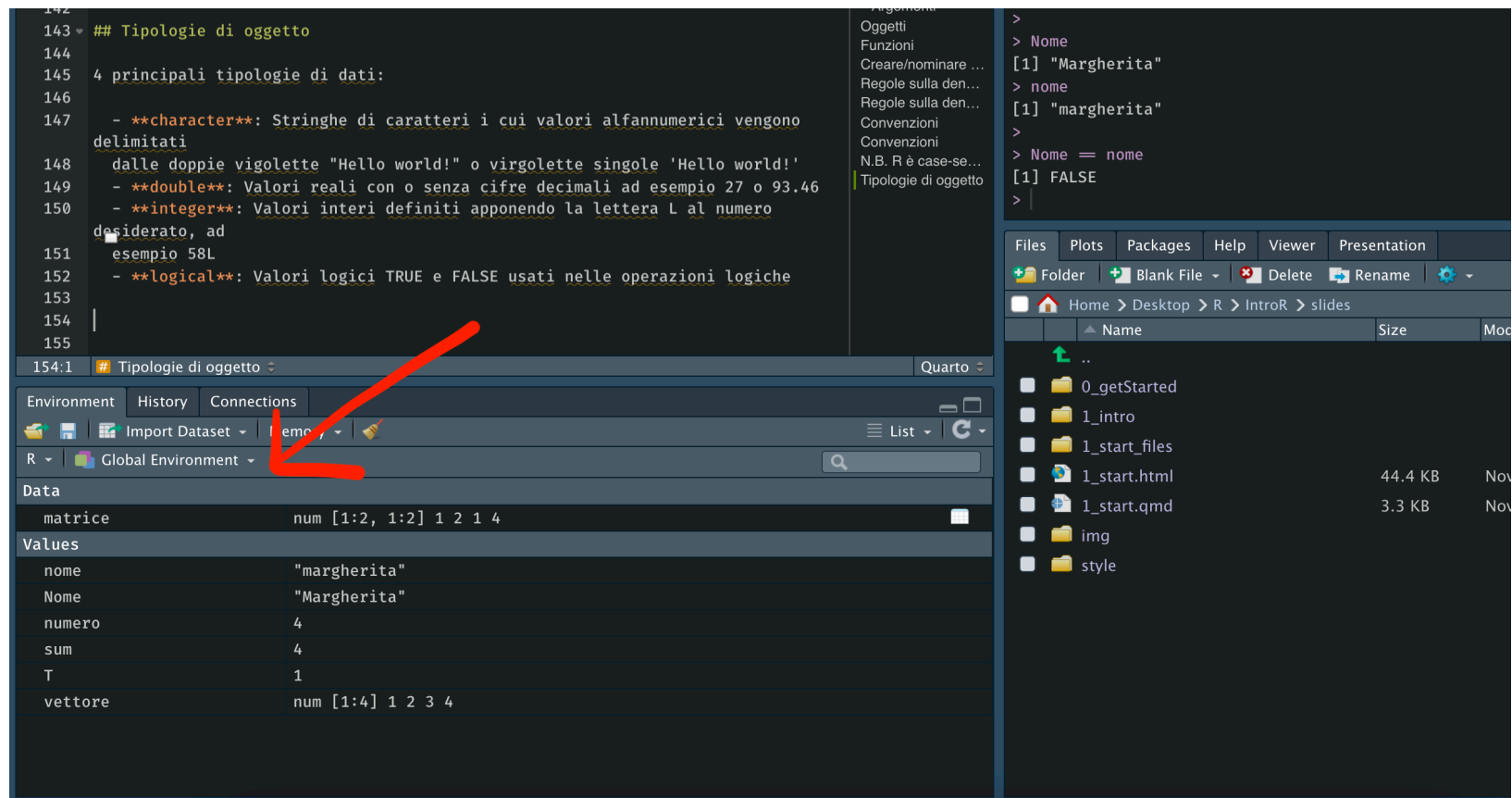
```
[1] "margherita"
```

```
1 Nome == nome
```

```
[1] FALSE
```

Dove vengono salvati gli oggetti?

Di default gli oggetti sono creati nel **global environment** accessibile con `ls()` o visibile in R Studio con anche alcune informazioni aggiuntive:



The screenshot displays the R Studio interface. The top-left pane shows R code with comments in Italian describing object types. The top-right pane shows the console output of the `ls()` function. The bottom-left pane shows the Environment window, which is currently displaying the Global Environment. A red arrow points to the 'Global Environment' dropdown menu. The bottom-right pane shows a file explorer view of the 'IntroR' directory.

```
142
143 ## Tipologie di oggetto
144
145 4 principali tipologie di dati:
146
147 - **character**: Stringhe di caratteri i cui valori alfanumerici vengono
delimitati
148 dalle doppie virgolette "Hello world!" o virgolette singole 'Hello world!'
149 - **double**: Valori reali con o senza cifre decimali ad esempio 27 o 93.46
150 - **integer**: Valori interi definiti apponendo la lettera L al numero
desiderato, ad
151 esempio 58L
152 - **logical**: Valori logici TRUE e FALSE usati nelle operazioni logiche
153
154
155
```

```
> ls()
[1] "Margherita"
> nome
[1] "margherita"
>
> Nome == nome
[1] FALSE
>
```

Environment | History | Connections

R - Global Environment

Data	
matrice	num [1:2, 1:2] 1 2 1 4

Values	
nome	"margherita"
Nome	"Margherita"
numero	4
sum	4
T	1
vettore	num [1:4] 1 2 3 4

Files | Plots | Packages | Help | Viewer | Presentation

Home > Desktop > R > IntroR > slides

Name	Size	Mod
0_getStarted		
1_intro		
1_start_files		
1_start.html	44.4 KB	Nov
1_start.qmd	3.3 KB	Nov
img		
style		

Eliminare oggetti

Possiamo eliminare un oggetto presente nel nostro environment attraverso il comando `rm("nomeoggetto")`.

E' possibile anche pulire completamente/svuotare il nostro environment attraverso il comando `rm(list = ls())`.

Tipologie di dato

- **character**: Stringhe di caratteri i cui valori alfanumerici vengono delimitati dalle doppie virgolette “Hello world!” o virgolette singole ‘Hello world!’
- **numeri**:
 - **double**: Valori reali con o senza cifre decimali ad esempio 27 o 93.46
 - **integer**: Valori interi definiti apponendo la lettera L al numero desiderato, ad esempio 58L

Operatori Matematici

Funzione	Cosa fa?	Esempio	Risultato
+	addizione	5.4 + 6.1	11.5
-	sottrazione	9 - 4.3	4.7
*	moltiplicazione	7 * 1.4	9.8
/	divisione	9/3	3
%%	resto	9%%2	1
^	potenza	15 ^ 2	225

Funzione	Cosa fa?	Esempio	Risultato
<code>abs</code>	valore assoluto	<code>abs(-8)</code>	8
<code>sqrt</code>	radice quadrata	<code>sqrt(225)</code>	15
<code>exp</code>	funzione esponenziale	<code>exp(0)</code>	1
<code>log</code>	logaritmo, base <i>e</i>	<code>log(1)</code>	0
<code>round</code>	arrotondamento, intero	<code>round(1.738)</code>	2
<code>round</code>	arrotondamento	<code>round(1.738, 2)</code>	1.74

Operazioni Matematiche

L'ordine delle operazioni in R segue le regole della matematica, a meno che non si specifichi un ordine diverso usando le parentesi ().

Esempi

```
1 # Senza parentesi
2 1 + 2 * 3
```

```
[1] 7
```

```
1 # Con le parentesi
2 (1 + 2) * 3
```

```
[1] 9
```


Operatori Relazionali

In R è possibile valutare se una data relazione è vera o falsa. R valuterà le proposizioni e ci restituirà il valore **TRUE** se la proposizione è vera oppure **FALSE** se la proposizione è falsa.

Funzione	Nome	Esempio	Risultato
<code>==</code>	uguale	<code>30 == 30</code>	TRUE
<code>!=</code>	diverso	<code>30 != 30</code>	FALSE
<code>>/>=</code>	maggiore/o uguale	<code>30 > 10</code>	TRUE
		<code>30 >= 10</code>	TRUE
<code></<=</code>	minore/o uguale	<code>30 < 10</code>	FALSE
		<code>10 <= 10</code>	TRUE
<code>%in%</code>	inclusione	<code>10%in%c(1,2,10)</code>	TRUE

Non vale solo per i numeri!

```
1 Nome = "Margherita"  
2  
3 nome = "margherita"  
4  
5 Nome == nome
```

```
[1] FALSE
```

PS. Ricordatevi che **=** è diverso da **==**

Operatori Logici

In R è possibile congiungere più relazioni per valutare una desiderata proposizione.

```
1 x = 30 #Assegnamo a x il valore 30.
```

Funzione	Nome	Esempio	Risultato
&	Congiunzione	$x > 25$ & $x < 60$	TRUE
	Disgiunzione Inclusiva	$x > 25$ $x > 60$	TRUE
!	Negazione	! ($x < 18$)	TRUE

TRUE equivale a 1 e FALSE a 0

```
1 TRUE == 1
```

```
[1] TRUE
```

```
1 TRUE == 2
```

```
[1] FALSE
```

```
1 FALSE == 0
```

```
[1] TRUE
```

```
1 FALSE == 1
```

```
[1] FALSE
```

```
1 esempio = c(rep("casa",4),rep("bottega",2))  
2 sum(esempio == "casa"); sum(esempio == "bottega")
```

```
[1] 4
```

```
[1] 2
```

Ordine Valutazione Relazioni

Nel valutare le veridicità delle proposizioni R esegue le operazioni nel seguente ordine:

1. Operatori matematici (e.g., \wedge , $*$, $/$, $+$, $-$, etc.)
2. Operatori relazionali (e.g., $<$, $>$, \leq , \geq , $==$, $!=$)
3. Operatori logici (e.g., $!$, $\&$, $|$)

In caso di dubbi riguardanti l'ordine di esecuzione delle operazioni, la cosa migliore è utilizzare le parentesi tonde $()$ per disambiguare ogni possibile fraintendimento.

Facciamo un po' di pratica!

Aprite e tenete aperto questo link:

<https://etherpad.wikimedia.org/p/arca-corsoR>



