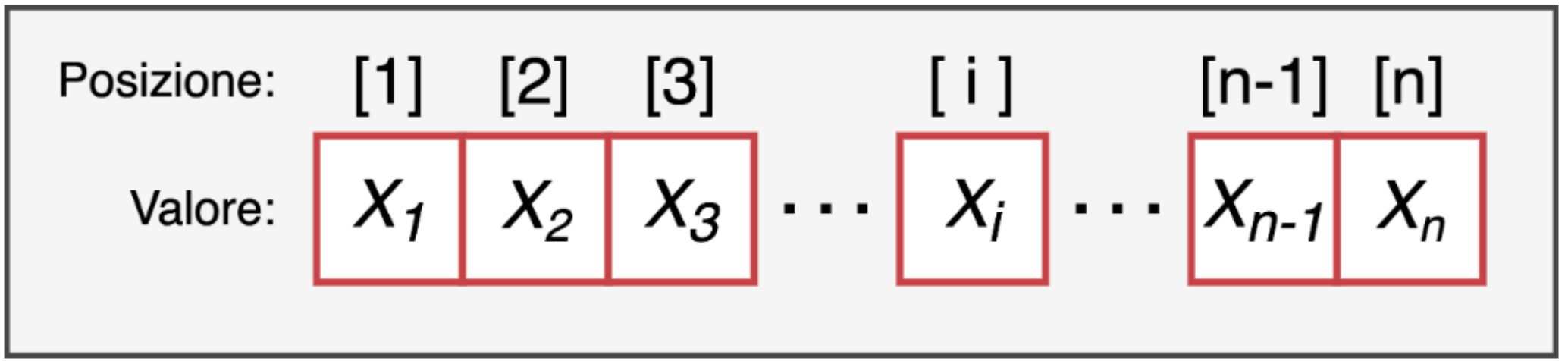
2.1_vettori

Strutture Dati

- ***vettori***
- fattori
- liste
- matrici
- array
- dataframe

Vettori

I vettori sono una struttura dati unidimensionale e sono la più semplice presente in R.

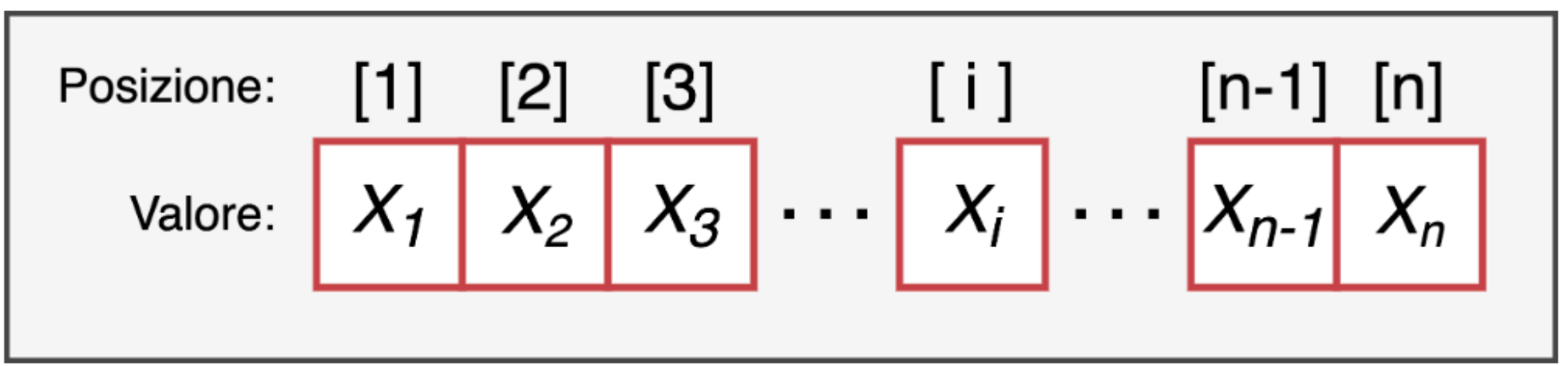


Caratteristiche di un vettore

- **la lunghezza**: il numero di elementi da cui è formato il vettore
- **la tipologia**: la tipologia di dati da cui è formato il vettore. Un vettore infatti deve esssere formato da **elementi tutti dello stesso tipo**!

Caratteristiche degli elementi di un vettore

- **un valore**: il valore dell'elemento che può essere di qualsiasi tipo ad esempio un numero o una serie di caratteri
- **un indice di posizione**: un numero intero positivo che identifica la sua posizione all'interno del vettore.



Creare un vettore

I vettori si possono creare attraverso il comando **`c()`**, indicando tra le parentesi i valori degli elementi nella sucessione desiderata e separati da una virgola.

```r
num_vect = c(1,2,3,4)
# è possibile anche utilizzare la funzione seq()
num_vect_seq = seq(from = 1,to = 4, by = 1)
num_vect;num_vect_seq
```

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

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

```r
char_vect = c("R","R","R","ok")
# è possibile anche utilizzare la funzione rep()
char_vect_rep = c(rep("R", 3), "ok")
char_vect;char_vect_rep
```

```
[1] "R"  "R"  "R"  "ok"
```

```
[1] "R"  "R"  "R"  "ok"
```

Tiplogia di vettore

La tipologia di dati da cui è formato il vettore.

```
1 class(num_vect)
```

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

```
1 class(char_vect)
```

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

Tiplogia di vettore

Un vettore deve esssere formato da **elementi tutti dello stesso tipo**!

```
1 wrong = c(1,2,3,"non so", 4)
2 class(wrong)
```

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

```
1 wrong
```

```
[1] "1"      "2"      "3"      "non so" "4"
```

Altrimenti si “rischia” che tutto venga trasformato a carattere.

```
1 correct = c(1,2,3,NA, 4)
2 class(correct)
```

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

```
1 correct
```

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

is.* & as.*

Possiamo testare o convertire (quando possibile) la tipologia del vettore attraverso queste funzioni **`is.`** & **`as.`**

```
is.character(char_vect)
```

```
[1] TRUE
```

```
as.numeric(char_vect) #!!
```

```
[1] NA NA NA NA
```

```
is.numeric(num_vect)
```

```
[1] TRUE
```

```
as.character(num_vect) #!!
```

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

```
logi_vect = c(TRUE,FALSE,TRUE)
is.logical(logi_vect)
```

```
[1] TRUE
```

```
as.numeric(logi_vect)
```

```
[1] 1 0 1
```

Attributi del vettore

Ogni elemento del vettore può essere associato ad un nome.

```
1 names(num_vect) #nessun nome associato
```

```
NULL
```

```
1 names(num_vect) = letters[1:4]
2
3 num_vect
```

```
a b c d
1 2 3 4
```

Ogni vettore è caraterizzato da una dimensione (**`dim()`**), in realtà essendo il vettore unidimensionale usiamo il comando **`length()`** per ottenere la lunghezza del vettore

```
1 dim(num_vect)
```

```
NULL
```

```
1 length(num_vect)
```

```
[1] 4
```

Indicizzazione

Possiamo selezionare, eliminare, estrarre elementi semplicemente usando l'indice di posizione tramite le parentesi quadre vettore[pos]

```r
# Creo un vettore formato da 20 numeri casuali pescati da 1 a 100
my_vect = round(runif(n = 20,min = 1, max = 100))
my_vect
```

```
 [1] 84 93 47 61 22 12 44 25 63 61 63 76 60 18 55 50 17 67 88 28
```

```r
my_vect[1] # estraggo il primo elemento
```

```
[1] 84
```

```r
my_vect[1:5] # estraggo i primi 5 elementi
```

```
[1] 84 93 47 61 22
```

```r
my_vect[c(1,4,2,9)] # estraggo elementi a scelta
```

```
[1] 84 61 93 63
```

```r
my_vect[length(my_vect)] #ultimo elemento (perchè?)
```

```
[1] 28
```

Inidicizzazione negativa

Allo stesso modo possiamo decidere di estrarre tutti gli elementi del vettore eccetto alcuni

```
my_vect[-c(1)] #tutti tranne il primo elemento
```

```
[1] 93 47 61 22 12 44 25 63 61 63 76 60 18 55 50 17 67 88 28
```

```
my_vect[-c(1:10)] #tutti tranne i primi 10
```

```
[1] 63 76 60 18 55 50 17 67 88 28
```

Se assegnamo nomi agli elementi del vettore, possiamo indicizzare questi “chiamandoli per nome” (meno comune)

```
1 names(my_vect) = letters[1:length(my_vect)]
2
3 names(my_vect)
```

```
 [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q" "r"
"s"
[20] "t"
```

```
1 my_vect
```

```
 a  b  c  d  e  f  g  h  i  j  k  l  m  n  o  p  q  r  s  t
84 93 47 61 22 12 44 25 63 61 63 76 60 18 55 50 17 67 88 28
```

```
1 my_vect["a"]
```

```
 a
84
```

```
1 my_vect[1]
```

```
 a
84
```

Indicizzazione Logica

Indicizzare con la posizione è l'aspetto più semplice e intuitivo. E' possibile anche selezionare tramite valori **TRUE** e **FALSE**: possiamo estrarre elementi dal vettore basandoci su specifiche condizioni logiche

```
1 numeri = 1:7
2 numeri
```

```
[1] 1 2 3 4 5 6 7
```

```
1 numeri[numeri<2]
```

```
[1] 1
```

L’idea è che se abbiamo un vettore di lunghezza **n** e un’altro vettore logico di lunghezza **n**, tutti gli elementi **TRUE** saranno selezionati:

```
numeri<2 # vettore logico
```

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

```
# verrà quindi estratto il primo elemento
# perchè è l'unico che risulta TRUE
numeri[numeri<2]
```

```
[1] 1
```

```
# non vale solo per elementi del vettore numeri ma per qualsiasi vettore
lettere = letters[1:6]
lettere
```

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

```
lettere[numeri<2]
```

```
[1] "a"
```

Indicizzazione Interna

Attraverso la funzione **`which()`** possiamo ottenere la posizione associata ad una selezione logica:

```
1 lettere
```

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

```
1 which(lettere == "a")
```

```
[1] 1
```

```
1 my_vect
```

```
 a  b  c  d  e  f  g  h  i  j  k  l  m  n  o  p  q  r  s  t
84 93 47 61 22 12 44 25 63 61 63 76 60 18 55 50 17 67 88 28
```

```
1 which(my_vect > 20)
```

```
 a  b  c  d  e  g  h  i  j  k  l  m  o  p  r  s  t
 1  2  3  4  5  7  8  9 10 11 12 13 15 16 18 19 20
```

```
1 my_vect[which(my_vect > 20)] # Cosa cambia tra questi 2 comandi?
```

```
 a  b  c  d  e  g  h  i  j  k  l  m  o  p  r  s  t
84 93 47 61 22 44 25 63 61 63 76 60 55 50 67 88 28
```

Indicizzazione e assegnazione

Indicizzare può essere utile anche per assegnare un valore ad un determinato elemento del vettore

```
1 lettere
```

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

```
1 lettere[1] = "z"
2 lettere
```

```
[1] "z" "b" "c" "d" "e" "f"
```

```
1 lettere[9] = "?"
2 lettere
```

```
[1] "z" "b" "c" "d" "e" "f" NA  NA  "?"
```

```
1 # Possiamo anche modificare l'elemento
2 lettere[2] = paste(lettere[2], sep = "", "o")
3 lettere
```

```
[1] "z"  "bo" "c"  "d"  "e"  "f"  NA   NA   "?"
```

Operazioni sui vettori

Possiamo eseguire operazioni sui vettori, ed applicare la stessa operazione a tutti gli elementi del vettore (element-wise)

```r
new_vect = rep(2:4, each = 2)
new_vect
```

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

```r
# potete svolgere qualsiasi operazione
new_vect/2
```

```
[1] 1.0 1.0 1.5 1.5 2.0 2.0
```

```r
log(new_vect)
```

```
[1] 0.6931472 0.6931472 1.0986123 1.0986123 1.3862944 1.3862944
```

```r
exp(new_vect)
```

```
[1]  7.389056  7.389056 20.085537 20.085537 54.598150 54.598150
```

Summary Statistics

Le operazioni più comuni sono per esempio la media **`mean()`**, la deviazione standard **`sd()`**, la mediana **`median()`**, il valore massimo **`max()`** e minimo **`min()`** .

```r
# creo un vettore campionando 1000 da una distribuzione normale
new_vect = rnorm(n = 1000, mean = 1, sd = 4)

mean(new_vect)
```

```
[1] 1.178953
```

```r
sd(new_vect)
```

```
[1] 4.025112
```

```r
median(new_vect)
```

```
[1] 1.31426
```

```r
summary(new_vect)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
-10.489  -1.585   1.314   1.179   4.073  12.961 
```

Valori NA

A volte (molto spesso) può capitare di avere dati mancanti

```
1 new_vect[1] = NA # Assegnamo NA al primo elemento del vettore
2
3 mean(new_vect) # è impossibile calcolare la media!
```

```
[1] NA
```

```
1 mean(new_vect, na.rm = T) # soluzione!
```

```
[1] 1.180797
```

Frequenze

Un altra operazione comune che viene svolta sui vettori (e sulle altre strutture dati), è il conteggio delle frequeze

```r
# riprendiamo il vettore creato all'inizio della lezione
char_vect
```

```
[1] "R"  "R"  "R"  "ok"
```

```r
# attraverso la funzione table possiamo calcolare la frequenze
table(char_vect)
```

```
char_vect
ok  R
 1  3
```

```r
# ricordatevi che R è case-sensitive!
char_vect[1] = "r";char_vect
```

```
[1] "r"  "R"  "R"  "ok"
```

```r
table(char_vect)
```

```
char_vect
ok  r  R
 1  1  2
```

Facciamo un po’ di pratica!

Aprite e tenete aperto questo link:

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