

ARRAY & SLICE

- "VETTORE"

$$\vec{x} = \underline{x} = \cancel{x}$$

lunghezza del vettore

$$\vec{x} = (x_0, x_1, x_2, x_3) \in \mathbb{R}^4$$

↑ ↑ ↑ ↑
componenti
(elementi) del vettore

vettore

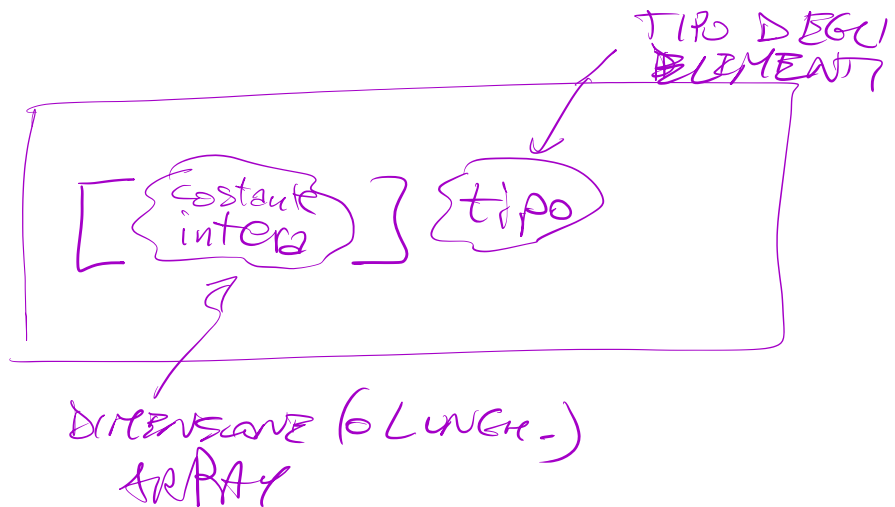
```

var x, s, n int
fut. Scan (&n)
for i := 0; i < n; i++ {
    fut. Scan (&x)
    s += x
}
media := float64(s) / float64(n)
fut. Println (media)

```

$$\begin{aligned}
 \sigma^2 = \text{var} &\stackrel{\Delta}{=} \frac{\sum_{i=1}^n (x_i - \mu)^2}{n} = \\
 &= \frac{(x_1 - \mu)^2 + (x_2 - \mu)^2 + \dots + (x_n - \mu)^2}{n}
 \end{aligned}$$

ARRAY



Var x [100] int

Var y [5] string

Var d [20] data

Var p [10] * int

↓
costante



y[3] = "Bolo"

Diagram illustrating the structure of a data structure, likely a hash table or array, with three main sections: $d[0]$, $d[1]$, and $d[11]$.

The first section, $d[0]$, contains three entries: g , m , and a , each with a value of 0.

The second section, $d[1]$, contains three entries: g , m , and a , each with a value of 0.

The third section, $d[11]$, contains three entries: g , m , and a , each with a value of 0.

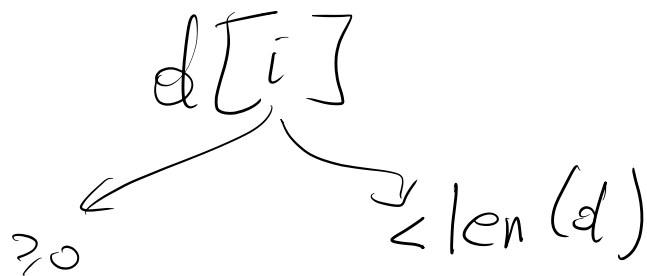
$$d[0] = \text{data } \{29, 11, 1968\}$$

opportunity

$$d[0].g = 29$$

$$d[0].m = 11$$

$$d[0].a = 1968$$



var a [100] int

if $n > \text{len}(a)$ {
 $n = \text{len}(a)$

func. Scan (&n) {

for $i := 0; i < n; i++$ {
 func. Scan (&a[i])

}

var s int

for $i := 0; i < n; i++$ {
 $s += a[i]$

}

media := float64(s) / float64(n)

var sq float64

for $i := 0; i < n; i++$ {

$sq += (\text{float64}(a[i]) - \text{media}) *$
 $(\text{float64}(a[i]) - \text{media})$

letura si ciclu

```

}
varianza := sq / Plost64(n)
sqm = wath. Sqrt(varianza)

```

ARRAY E CEL

I°

```

for i := 0; i < len(a); i++ {
    ... i ... a[i] ...
}

```

II°

```

for i := range a {
    ... i ... a[i] ...
}

```

III°

```

for i, x := range a {
    ... i ... x ...
    a[i]
}

```

serve solo per la lettura

"DIFETTI" DEGLI ARRAY

- DIMENSIONE STATICA
⇒ solo di dimensionamento
- ASSEGNAMENTO = COPIA

<u>var</u>	a	[100] int
<u>var</u>	b	[100] int

↖
b = a

- FUNZIONI CON ARGOMENTO
UN ARRAY ~~SONO~~ A
DIMENSIONE FISSA

func media (2 [100] int, n int)
float64 {

...

LETTERALI ARRAY

$$X ::= [8] \underline{\text{int}} \{1, 2, 3, 4, 5\}$$

x

1	2	3	4	5	0	0	0
---	---	---	---	---	---	---	---

$$X_{\bullet} = [\dots] \text{int } \{1, 2, 3, 4, \dots\}$$

X

1	2	3	4	5
---	---	---	---	---

$$x_{i,0} = [10] \text{ int } \{ \underbrace{0:3}, \underbrace{7:5} \}$$

x

0	1	2	3	4	5	6	7	8	9
3	0	2	0	0	0	0	5	0	0


```
func printData (d data) {  
    nome Mese := [...] string { "Gen",  
        "Feb", "Mar", ... }  
}
```

```
    printf("%d _ %s _ %d",  
        d.g, nome Mese[d.m-1],  
        d.a)
```

```
}
```

SLICE

- DIMENSIONE NON È
STATICA E PUÒ
VARIARE NEL TEMPO

[] {tipo}

var x [] int
var s [] string
var t [] data
var p [] *int

} nil

len(x) ~> lunghezza attuale
della slice

len(nil) = 0

x[0], x[1], ..., x[len(x) - 1]

CREATEZONE

- LITERAL SLICE

$x = [] \text{int } \{1, 2, 3, 4, 5\}$

x	0	1	2	3	4
	1	2	3	4	5

~~$x = [5] \text{int } \{1, 2, 3, 4, 5\}$~~

- RUNZONE make

$x = \text{make}([] \text{int}, 10)$

TYPE OF SLICE
OR CREATE

LENGTH

x	0	1	2	3	4	5	...	9
	0	0	0	0	0	0		0

```

var    n    int
func. Scan (&n)
var    a    []int
a = make([]int, n) } → a := make([]int, n)
for    i := 0; i < n; i++ {
    func. Scan (&a[i])

```

```

}
for    _, x := range a {
    s += x

```

```

}
media := float64(s) / float64(n)
for    _, x := range a {
    sq += (float64(x) - media) *
        (float64(x) - media)

```

```

}
varianza := sq / float(len(a))

```

ALLUNGAMENTO SLICE

$X = \text{append}(x, lo)$

SLICE DA
ALLARG.

NUOVO
ELEMENTO

var s []string
s = []string {"Paolo", "Boldi"}
s = append(s, "Anna")
s = append(s, "Marco")

var s []string
s = make([]string, 5)
s = append(s, "Paolo")

0	1	2	3	4	5
z	<	Pippo	<	<	Marco

s[2] = "Pippo"

s[5] = "Marco"

ESERCIZIO: leggere una

sequenza di nomi

risparmiarli al computer

e stampare il nome più
lungo