TIPI STRUTTURATI: ARRAY

Fondamenti di Programmazione 2021/2022

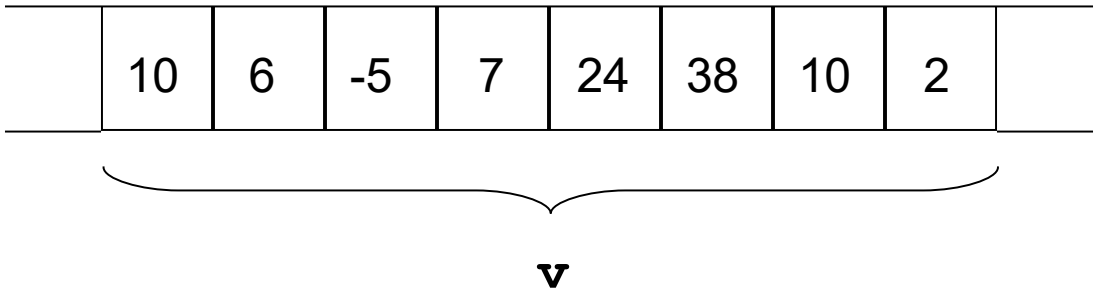
Francesco Tortorella

Gli array

- In alcuni casi, l'informazione che bisogna elaborare consiste di un'aggregazione di valori, piuttosto che di un valore solo.
- Questo significa che sarebbe conveniente indicare l'insieme di valori di interesse con una sola variabile piuttosto che con tante variabili quante sono i valori da considerare: una variabile di **tipo strutturato**.
- In C (come nella maggior parte dei linguaggi), questa possibilità è offerta dagli ***array***.

Gli array

- Un array è un insieme di variabili, tutte dello stesso tipo, identificato da un nome unico. Gli elementi dell'array sono disposti in memoria in posizioni consecutive



Definizione di un array

- Per definire una variabile array, è necessario specificare:
  - il nome della variabile array
  - il tipo degli elementi
  - il numero degli elementi presenti (cardinalità dell'array)

Esempio

- Definizione di una variabile array v contenente 20 interi:

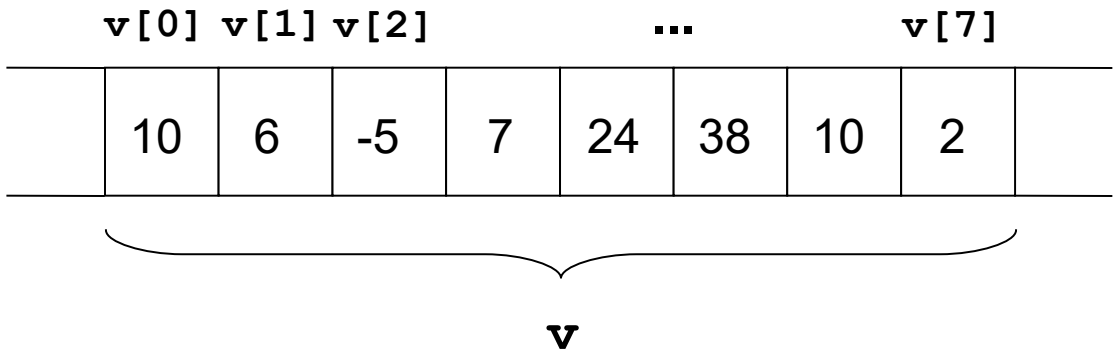
```
int v[20];
```

- Definizione di una variabile array **w** contenente 10 float:

```
float w[10];
```

Accesso agli elementi dell'array

- Per accedere ai singoli elementi di un array, è necessario specificare il nome della variabile array e la posizione dell'elemento di interesse tramite un valore intero (variabile o costante) che si definisce indice.



Accesso agli elementi dell'array

- Si noti che l'indice parte da 0; quindi `v[0]` sarà il primo valore dell'array, mentre l'N-mo sarà `v[N-1]`.
- Va quindi ricordato che, se si definisce un array con `N` elementi, l'indice dovrà essere limitato tra `0` ed `N-1`.
- Questo controllo è a cura dell'utente, in quanto non ci sono controlli automatici della correttezza dell'indice. Nel caso si consideri un indice errato (es. `v[N]`), sarà effettuato un accesso ad una zone della memoria che non appartiene all'array, con effetti imprevedibili a runtime.

Accesso agli elementi dell'array

Ogni elemento di un array è, a tutti gli effetti, una variabile del tipo costituente l'array e quindi può essere impiegato come tale:

```
int a,b,i;
int v[10];


v[2]=3;
v[7]=0;
printf("Valore: %d\n",v[7]);
i=2;
a=v[i]*4+6;
b=v[i+5];
```

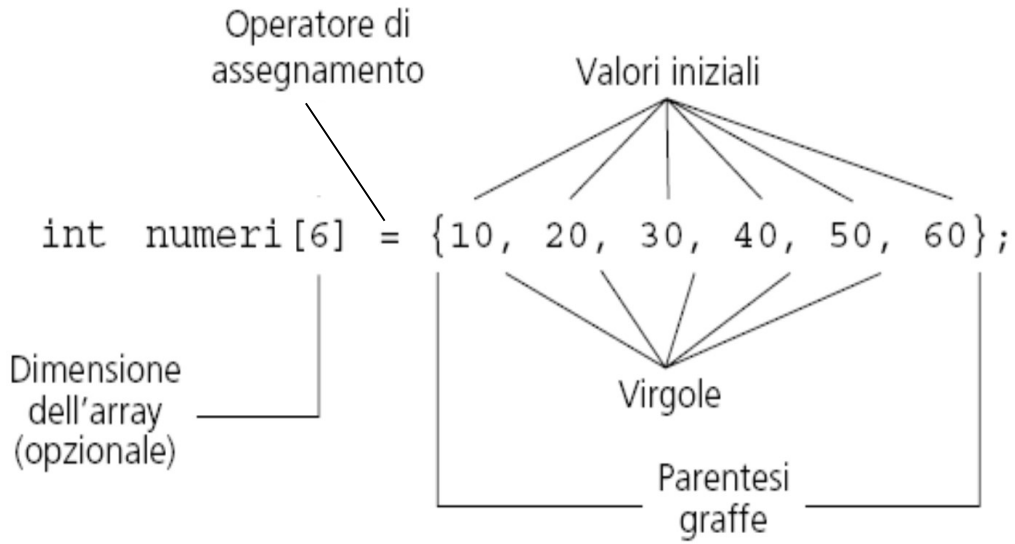
Inizializzazione di un array

- Un array può essere inizializzato in fase di definizione:

```
int numeri[6] = {10,20,30,40,50,60};
```

- La dimensione dell'array può essere anche implicita:

```
int numeri[] = {10,20,30,40,50,60};
```



Assegnazione tra array

- Diversamente dalle variabili di tipo atomico, non è possibile fare assegnazioni dirette tra array.
- L'unica possibilità per assegnare i valori degli elementi di un array agli elementi di un altro array è quindi fare una serie di assegnazioni tra elementi corrispondenti:

```
int a[]={7,9,6,3};
int b[4];
```

```
b=a;
```

errata

```
b[0]=a[0];
b[1]=a[1];
b[2]=a[2];
b[3]=a[3];
```

corretta

Dimensione dell'array

- La dimensione fornita all'atto della definizione dell'array **deve essere una costante**.
- Esempi di definizioni corrette:

```
/* dimensione costante numerica */
int vet[100];

/* dimensione costante definita */
#define SIZE 150
…
int vet[SIZE];
```

Dimensione dell'array

- Nel caso il numero degli elementi da memorizzare in un array fosse noto solo a tempo di esecuzione, comunque la dimensione dell'array da impiegare **deve essere una costante**
- In questo caso, per la definizione si sceglie una dimensione che si ritiene adeguata ad ospitare il numero massimo di elementi previsto e si affianca all'array una variabile intera che contiene il numero effettivo di elementi memorizzati nell'array (riempimento).

Lettura e stampa degli elementi di un array

- Per inizializzare da input una variabile array, è necessario realizzare un'operazione di input per ciascuno degli elementi
- Analogamente, per stampare il contenuto di un array, è necessario fare la stampa di ognuno degli elementi.
- Qual è il costrutto da utilizzare ?
- Problema:
  - leggere da input la dimensione e gli elementi di un array e stampare il risultato della lettura

```c
#include <stdio.h>
#define MAXSIZE 100
int main() {
        int vet[MAXSIZE];
        int i,num;

        printf("Quanti elementi?");
        scanf("%d",&num);
        while(num > MAXSIZE){
                        printf("Numero eccessivo!\n");
                        printf("Quanti elementi?");
                        scanf("%d", &num);
        }

        for(i=0; i < num; i++){
                        printf("Fornire il valore n.%d:",i)
                        scanf("%d",&vet[i]);
        }

        for(i=0; i < num; i++)
                        if(vet[i] < 0)
                                vet[i] = -vet[i];

        for(i=0; i < num; i++)
                        printf("Valore [%d]: %d\n",i, vet[i]);
}
```

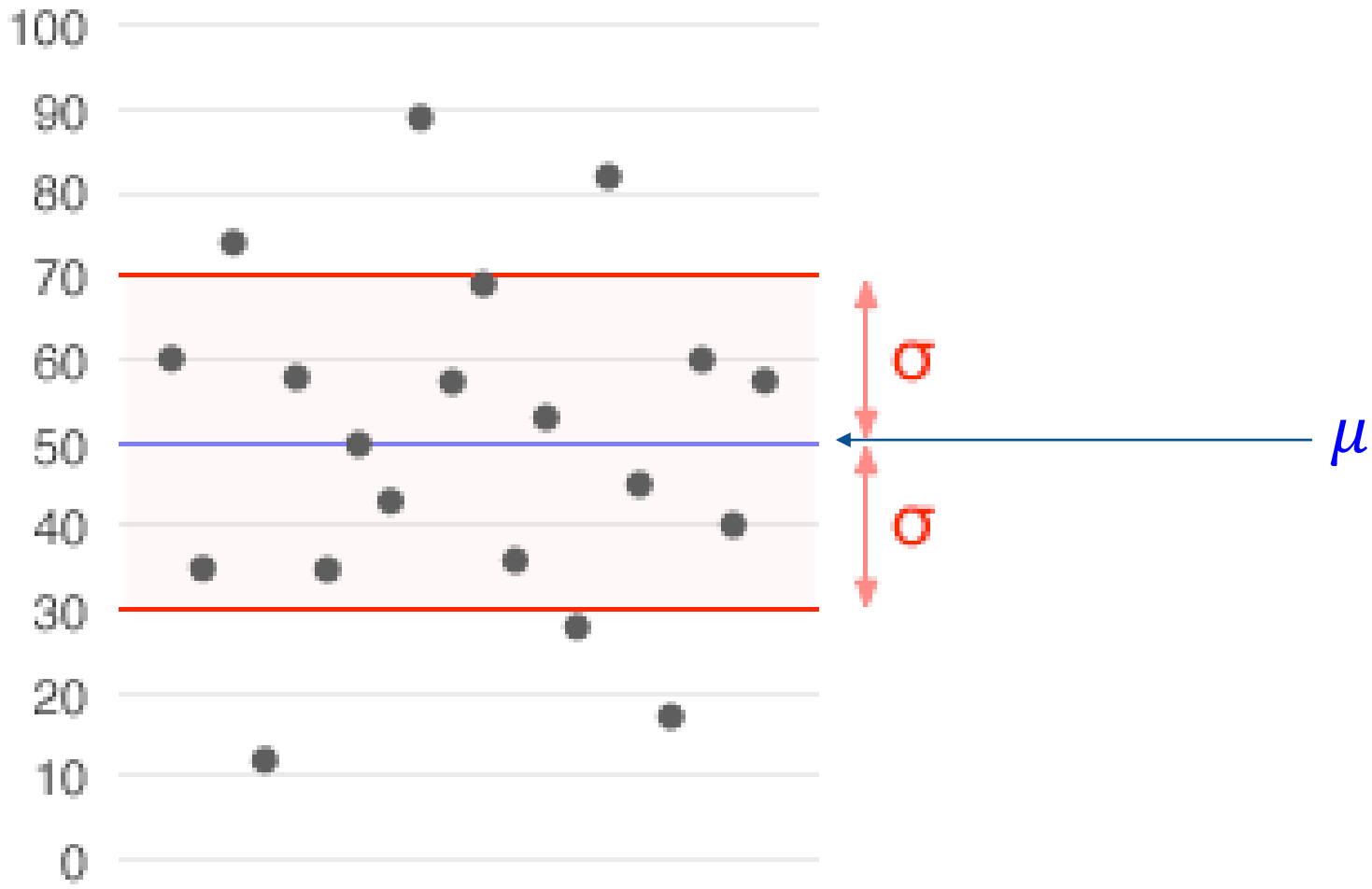
```c
#include <stdio.h>
#define MAXSIZE 100
int main() {
        int vet[MAXSIZE];
        int i,num;

        printf("Quanti elementi?");
        scanf("%d",&num);
        while(num > MAXSIZE){
                printf("Numero eccessivo!\n");
                printf("Quanti elementi?");
                scanf("%d", &num);
        }

        for(i=0; i < num; i++){
                printf("Fornire il valore n.%d:",i)
                scanf("%d",&vet[i]);
        }

        for(i=0; i < num; i++)
                if(vet[i] < 0)
                        vet[i] = -vet[i];

        for(i=0; i < num; i++)
                printf("Valore [%d]: %d\n",i, vet[i]);
}
```

Uso della costante definita

Uso del riempimento

Problema:
calcolo della media e della deviazione standard di un insieme di dati

Ogni punto sul diagramma rappresenta un particolare dato, il cui valore può essere letto sulla scala a sinistra.
La linea blu indica il valore medio (μ) dei dati.
La linea rossa indica la deviazione standard (σ) dei dati .



Problema:
calcolo della media e della deviazione standard di un insieme di dati

- Leggere da input la dimensione *n* e gli elementi di un array di numeri reali; fornire in uscita la media μ e la deviazione standard σ degli elementi presenti nell'array, dove:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2}$$

x_i ←→ `x[i]`

Problema (versione 2): calcolo della media e della deviazione standard di un insieme di dati

- Nelle stesse ipotesi dell'esercizio precedente, calcolare la media μ e la deviazione standard σ di un insieme di valori reali letti in un array, dove stavolta si usa l'espressione:

$$\sigma = \frac{1}{n}\sqrt{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}$$

- Che cosa è cambiato? È cambiato in meglio o in peggio?

Array bidimensionali

- Finora abbiamo considerato **array monodimensionali**, i quali richiedono un solo indice per l'individuazione di un elemento.
- Il C permette di definire anche **array bidimensionali**, in cui l'organizzazione degli elementi è di tipo matriciale.
- In questo caso, sono necessari due indici per identificare un elemento nell'array.
- Questo tipo strutturato permette di affrontare tutte quelle situazioni in cui è necessario lavorare con matrici, tabelle, ecc.

Definizione di un array bidimensionale

- Per definire un array bidimensionale, è necessario specificare:
  - il **nome** della variabile array
  - il **tipo** degli elementi
  - il **numero degli elementi presenti nelle due dimensioni** (cardinalità di riga e cardinalità di colonna dell'array)

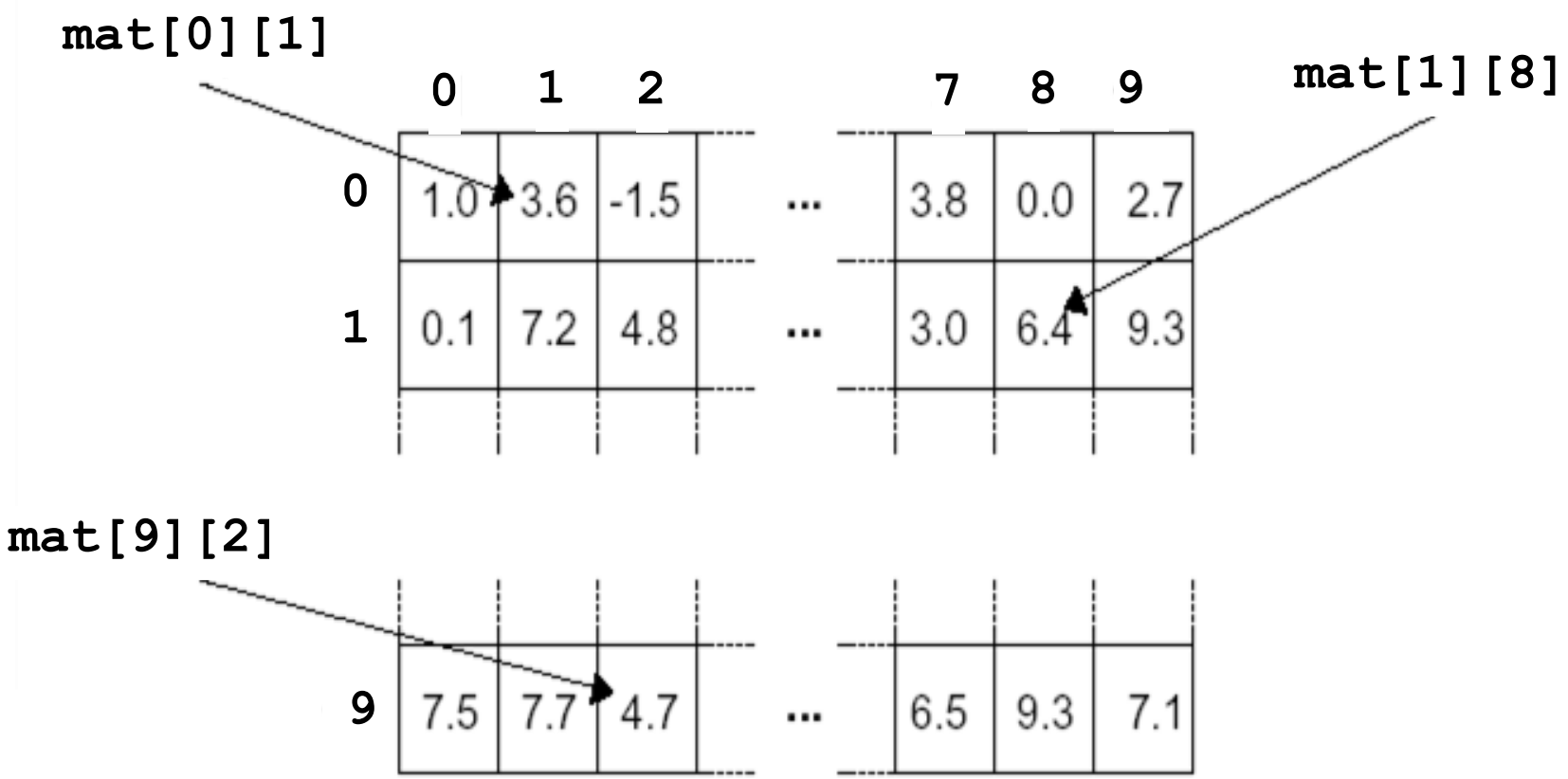
Esempio

- Definizione di una variabile array **`mat`** contenente 10x10 elementi **`double`**:

```
double mat[10][10];
```

- **Che differenza c'è rispetto ad un array monodimensionale di 100 elementi?**

Organizzazione di un array bidimensionale



Accesso agli elementi dell'array bidimensionale

- Per accedere ai singoli elementi di un array bidimensionale, è necessario specificare il nome della variabile array e gli indici di riga e di colonna che individuano l'elemento desiderato.
- **Esempi**:

```
mat[2][1]=3;
printf("il valore è: %d\n", mat[2][7]);
i=3;
j=5;
x=mat[i][j]*4+6;
```

Inizializzazione di un array bidimensionale

- Un array bidimensionale può essere inizializzato in fase di definizione:

```
int mat[2][3] = {{10,20,30},{40,50,60}};
```

- Altre inizializzazioni equivalenti:

```
int mat[2][3] = {10,20,30,40,50,60};

int mat[][3] = {{10,20,30},{40,50,60}};
```

Lettura e stampa di un array bidimensionale

- Anche nel caso bidimensionale, l'inizializzazione da input di una variabile array va realizzata realizzare tramite un'operazione di input per ciascuno degli elementi
- Analogamente, per stampare il contenuto di un array, è necessario fare la stampa di ognuno degli elementi.
- **Qual è il costrutto da utilizzare ?**
- Esempio:
  - leggere da input le dimensioni e gli elementi di un array bidimensionale e stampare il risultato della lettura

Array multidimensionali

- Il C permette la definizione di array multidimensionali con più di due indici. Esempio:
  ```
  int mat3[5][10][5];
  ```

- Con le dovute modifiche valgono le considerazioni sulla definizione, inizializzazione, assegnazione, accesso fatte per gli array monodimensionali e bidimensionali.