

# Laboratorio di programmazione per sistemi ciberfisici

## 10. Strutture

*Enrico Martini*  
*December 15, 2025*

## Definizione

La struttura è un'unità logica che *raggruppa più elementi*.

► Il comando `struct <nome> ...` crea un **nuovo tipo di dato**:

```
1 struct data{  
2     int giorno;  
3     int mese;  
4     int anno;  
5 };
```

Se una struttura viene definita all'interno di una funzione, è detta **struttura locale**, mentre se è definita all'esterno, è detta **struttura globale**.

## Accesso

Per accedere ad un membro della struttura si utilizza

`<nome_struttura>.<nome_membro>.`

```
1 struct data{
2     int giorno;
3     int mese;
4     int anno;
5 };
6
7 int main(){
8     struct data d;
9     scanf("%i/%i/%i",&d.giorno, &d.mese, &d.anno); // es: "16/12/2025"
10    // ...
11    printf("%i\n", d.mese); // "12"
12 }
```

## Inizializzazione

Le strutture possono essere inizializzate in modo simile a un array, oppure possono essere esplicitati i membri:

```
1 struct data x = {25, 12, 2025};  
2 struct data x = {25, 12};           // Il campo membro NON viene impostato a 0!  
3 struct data x = {.giorno = 25, .mese = 12, .anno = 2025};
```

Possono essere assegnati valori anche dopo aver dichiarato la struttura:

```
1 struct data x = {25, 12, 2025};  
2 x = (struct data) {26, 11, 2026};  
3 x = (struct data) { .giorno = 26, .anno = 2026 };
```

Se si usa la notazione `.<membro>` allora l'ordine non è importante

## Strutture e Funzioni

È possibile passare le strutture come argomento alle funzioni e restituirle dalle funzioni.

Esempio:

```
1  _Bool isCapodanno(struct data d){  
2      if(d.giorno == 1 && d.mese == 1)  
3          return 1;  
4      return 0;  
5  }
```

Il passaggio di strutture avviene **per copia**, a differenza degli array, che vengono passati per riferimento.

## Strutture e Array

Le strutture possono contenere array come membri. Se un membro di una struttura è di tipo array, anch'esso viene passato **per copia**.

```
1 struct esagono{  
2     float x[6];  
3     float y[6];  
4 };
```

Possono essere utilizzati array di strutture:

```
1 struct data A[10];
```

Possono essere inizializzati come array multidimensionali:

```
1 struct time runTime[5] = {{10, 25, 15}, {11, 12, 52}, {16, 24, 52}};
```

## Strutture di strutture

È possibile definire una struttura che contiene al suo interno altre strutture:

```
1 struct date{
2     int giorno; int mese; int anno;
3 };
4 struct time{
5     int ora; int minuto; int secondo;
6 };
7 struct dateTime{
8     struct date data; struct time tempo;
9 };
```

```
1 struct dateTime x = {{5, 3, 2025}, {16, 24, 52}};
2 printf("Sono le ore %i del giorno %i.\n", x.tempo.ora, x.data.giorno);
3 // "Sono le ore 16 del giorno 5."
```

## Esercizio 1

Scrivi un programma che accetta la data di oggi come input e visualizza la data di domani all'utente.



## Esercizio 2

Scrivi un programma che accetti in input due date e restituisca la data successiva tra le due.

## Esercizio 3

Creare una struttura `Rettangolo` con i membri `base` e `altezza`. Scrivere una funzione che calcoli l'area del rettangolo e la stampi.

## Esercizio 4

Creare una struttura `Punto` e una funzione `somma_puntiche`, dati due punti, restituisca il punto risultante dalla loro somma.

## Esercizio 5

Scrivere una funzione che aggiorna l'ora passata in input di un secondo (es.: 14:29→14:30).

```
1 14:29:10
```

```
2 14:29:11
```

```
1 23:59:59
```

```
2 00:00:00
```

## Esercizio 6

Scrivere una funzione `elapsed` che riceve come argomenti due strutture `time` e restituisce una struttura `time` che rappresenta il tempo trascorso tra i due istanti.

## Esercizio 7

Scrivere un programma che, date in input le informazioni relative a due triangoli (base e altezza), stampi l'area del triangolo maggiore.

## Esercizio 8

Scrivere una funzione `euler_distance` che calcoli la distanza tra due punti 3D, usando la distanza di Eulero:

$$d = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}$$

PS: importare la libreria `math.h` e utilizzare la funzione `sqrt()` per il calcolo della radice quadrata. Durante la compilazione aggiungere il flag `-lm`:

```
1 gcc -lm programma.c
```

## Esercizio 9

Creare una struttura `Poligono` che contiene un array di punti e un intero `numero_lati`. Scrivere una funzione che calcoli il perimetro del poligono.



## Esercizio 10

Dichiarare un array di struct Punto contenente 10 punti 2D. Scrivere una funzione che calcoli il baricentro dei punti:

$$x_{BAR} = \frac{1}{10} \sum_{i=0}^{10} x_i$$

$$y_{BAR} = \frac{1}{10} \sum_{i=0}^{10} y_i$$

## Esercizio 11

Calcolare il numero di giorni trascorsi tra due date A e B, usando la seguente formula:

$$N = \frac{1461 \cdot f(\text{anno}, \text{mese})}{4} + \frac{153 \cdot g(\text{mese})}{5} + \text{giorno}$$

$$f(\text{anno}, \text{mese}) = \begin{cases} \text{anno} - 1 & \text{se } \text{mese} \leq 2 \\ \text{anno} & \text{altrimenti} \end{cases}$$

$$g(\text{mese}) = \begin{cases} \text{mese} + 13 & \text{se } \text{mese} \leq 2 \\ \text{mese} + 1 & \text{altrimenti} \end{cases}$$