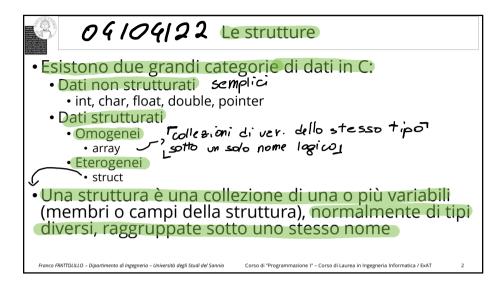


UNIVERSITÀ DEGLI STUDI DEL SANNIO Benevento DING DIPARTIMENTO DI INGEGNERIA

CORSO DI "PROGRAMMAZIONE I"

Prof. Franco FRATTOLILLO Dipartimento di Ingegneria Università degli Studi del Sannio

Franco FRATTOLILLO - Dipartimento di Ingegneria - Università degli Studi del Sannio





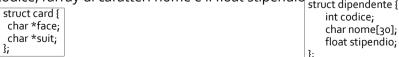
Perché usare le strutture

- Quando in un programma si vogliono rappresentare dati appartenenti al mondo reale, i tipi di dati semplici, quali interi, caratteri, ecc. non sono sufficienti
 - spesso ci si deve riferire ad oggetti che sono aggregati di dati
 - -> ad esempio, un libro è un'entità composta da un titolo, un autore, un editore, l'anno di pubblicazione, il testo
- Le strutture sono utili anche
- ->• per definire record da memorizzare in file
- ->• combinate con i puntatori, per creare strutture dati dinamiche (liste, alberi, grafi ...)



Esempi di strutture

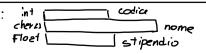
- Per dichiarare una struttura si usa la parola riservata struct
- La definizione di una struttura crea un nuovo tipo
- → dipendente è il nome della struttura
 - -> È un nuovo tipo che può essere usato per dichiarare variabili di tipo dipendente
 - -> Ogni variabile di tipo dipendente consta di 3 campi, l'intero codice, l'array di caratteri nome e il float stipendio struct dipendente {

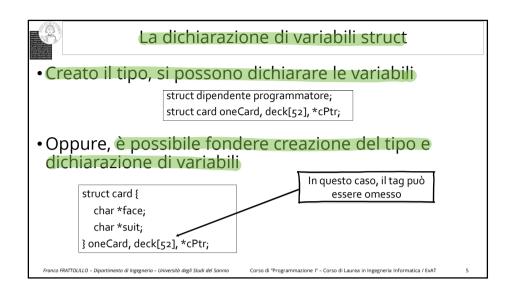


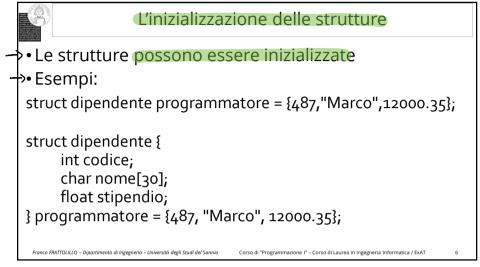
FRANCO

LILLO

DIPENDENTE:







```
L'assegnazione delle strutture

• Le strutture possono essere globalmente assegnate

i vettori no => pontetori
costenti elle teste

• Esempio:

struct dipendente prog1 = {487,"Marco",12000.35};

struct dipendente prog2;
...

prog2 = prog1;

Franco FRATTOLILLO - Dipartimento di Ingegneria - Università degli Studi del Sannio Corso di "Programmazione I" - Corso di Laurea in Ingegneria Informatica / ExAT 7
```



Altre operazioni sulle strutture

- Altre operazioni che possono essere effettuate su strutture sono:
 - ottenere l'indirizzo di una struttura (con &)
 - usare l'operatore size of per determinare l'ingombro di una struttura in termini di byte

Franco FRATTOLILLO - Dipartimento di Ingegneria - Università degli Studi del Sannio

Corso di "Programmazione I" - Corso di Laurea in Ingegneria Informatica / ExAT

```
Strutture ed array

    La dichiarazione di un array di strutture si realizza così:

struct matita {
                                  struct matita {
 int durezza:
                                    int durezza;
 char *fabbricante:
                                    char *fabbricante; -> puntetore e cerettere
 int numero:
                                    int numero;
};
                                  } m[3];
main() {
                              m = 1=1=
 struct matita m[3]:
 m[0].durezza = 2:
 m[2].numero = 400;
 m[1].fabbricante = "Modini";
 Franco FRATTOLILLO - Dipartimento di Ingegneria - Università degli Studi del Sannio
                                              Corso di "Programmazione I" - Corso di Laurea in Ingegneria Informatica / ExAT
```

```
Esempio

    Gli array di strutture sono utili al posto dei vettori

 paralleli o bidimensionali
  • ad esempio, un programma che conta le occorrenze delle parole chiavi del C, invece di essere organizzato con due
    vettori paralleli
                char *
                             auto | break
                                                    while
                                                       1

    può essere organizzato con un vettore di strutture

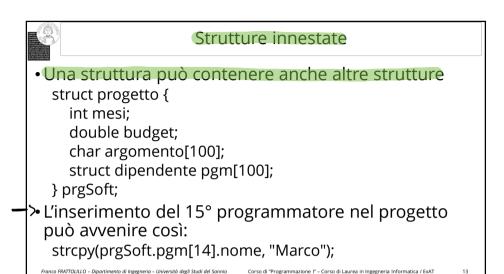
     · ogni elemento dell'array è una struttura a due campi che
      specifica una parola chiave ed il relativo numero di occorrenze
```

```
Esempio
char *word; => punt et ove e cevettere
struct key {
 int count:
} keytab[NKEY] = {"auto", 0,
                  "break". 0.
                  "while", 0};
for(i = 0; i < NKEY; i++)
 printf("%s è presente %d volte\n", keytab[i].word,
              keytab[i].count);
```

```
token -> scan F
                      token (- cher[]
linee -> gets
                      OCC.
```

```
struct Token ?
                    Token Testo [100];
    cher 5+1 [30];
    int occ ;
```

ρίρρο

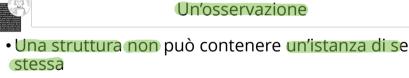


```
• Rappresentiamo mediante una struttura l'oggetto rettangolo

• In una rappresentazione cartesiana, un rettangolo è univocamente determinato dai suoi vertici in alto a destra ed in basso a sinistra_struct point {
    int x;
    int y;
};
struct rectangle {
    struct point pt1;
    struct point pt2;
};

**Franco FRATTOLILLO - Dipartimento di Ingegneria - Università degli Studi del Sannio

**Corso di "Programmazione l" - Corso di Laurea in Ingegneria Informatica / Ex-T 14
```



 Una struttura può però contenere un puntatore ad una struttura dello stesso tipo

• È questa la chiave per realizzare strutture dati dinamiche

Franco FRATTOLILLO - Dipartimento di Ingegneria - Università degli Studi del Sannio

Corso di "Programmazione I" - Corso di Laurea in Ingegneria Informatica / ExAT

```
struct matita {
  int durezza;
  char fornitore[40];
} m;

struct matita (*matPunt) = &m;
  (*matPunt).durezza = 3;
  strcpy((*matPunt).fornitore, "Mondini");
...

Franco FRATTOLILLO - Dipartimento di Ingegneria - Università degli Studi del Sannio Corso di "Programmazione l' - Corso di Laurea in Ingegneria Informatica / ENAT 16
```



L'operatore ->

- Per semplificare l'accesso ai membri di una struttura, quando si usano i puntatori, è disponibile l'operatore
- -> Freccia
- <puntatore struttura> -> <membro struttura>
- nell'esempio precedente, possiamo scrivere matPunt->durezza = 3: strcpy(matPunt->fornitore, "Mondini");
- nel caso di strutture innestate, l'operatore -> può essere utilizzato ripetutamente (è associativo a sinistra)

```
struct cartella {
...
struct matita *m;
}*q;
...
... q->m->fornitore ...
... q->m->durezza ...
```

Franco FRATTOLILLO – Dipartimento di Ingegneria – Università degli Studi del Sannio

Corso di "Programmazione I" – Corso di Laurea in Ingegneria Informatica / ExAT



Il passaggio delle strutture

- •Le strutture sono passate alle funzioni per valore
 - pertanto, la funzione opera su una copia locale della struttura
- È possibile il passaggio per riferimento, usando come parametro un puntatore alla struttura
 - in tal modo, si consente alla funzione di modificare la struttura

errey solo riferimento

Franco FRATTOLILLO – Dipartimento di Ingegneria – Università degli Studi del Sannio

Corso di "Programmazione I" - Corso di Laurea in Ingegneria Informatica / ExAT



L'aritmetica dei puntatori a strutture

- Sui puntatori a strutture è possibile applicare un'aritmetica, così come per gli altri puntatori
 - occorre soltanto tener presente che gli operatori d'accesso . e -> hanno la precedenza massima e che sono operatori associativi a sinistra ... ++p->len ...

struct Str{
int len;
char *str;
} *p;

struct Str s; s.len=2; s.str="pippo"; p=&s; p purtz S ... ++p->len ... /* è uguale a */ ... ++(p->len) ... /* che è diverso da */ ... (++p)->len ++p(->len) ... /* SBAGLIATO */

Per risolvere il problema usare le parentesi

ranco FRATTOLILLO – Dipartimento di Ingegneria – Università degli Studi del Sanni

Corso di "Programmazione I" – Corso di Laurea in Ingegneria Informatica / ExAT



La dichiarazione typedef

- typedef consente di dichiarare nuovi tipi sulla base di tipi esistenti
- Ad esempio: typedef char* Stringa
- definisce Stringa come un alias del tipo char*
- Pertanto, si può scrivere: Stringa str1, str2;
- Esempio:

typedef struct punto {...} PIXEL; PIXEL center, *origin;

/* invece di struct punto center, *origin; */

Franco FRATTOLILLO – Dipartimento di Ingegneria – Università degli Studi del Sannio

Corso di "Programmazione I" – Corso di Laurea in Ingegneria Informatica / ExAT

-



La dichiarazione typedef

- L'uso di typedef consente di aumentare la chiarezza e la leggibilità del programma
 - forse ...
- L'uso di typedef consente anche di elevare la portabilità del programma
 - dovendo portare su un altro sistema un programma che ha tipi dipendenti dall'architettura, è sufficiente modificare le typedef senza che vi sia la necessità di modificare le dichiarazioni di variabili

Franco FRATTOLILLO – Dipartimento di Ingegneria – Università degli Studi del Sannio

Corso di "Programmazione I" – Corso di Laurea in Ingegneria Informatica / ExAT