

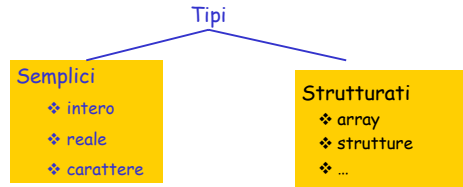
Strutture

Antonella Santone

1

Tipi di dati

- ❖ VALORI: un insieme dei valori del tipo
- ❖ OPERAZIONI: per operare su tali valori



2

Strutture

Aggregare elementi di tipo diverso

Esempio

```
struct data{
    int giorno;
    char mese[20];
    int anno;
};
```

Introduzione del nuovo tipo `data`

3

Dichiarazioni di variabili

Una volta definito il tipo di dato, possiamo dichiarare una variabile di quel tipo, permettendo però la keyword `struct`

Esempio

```
struct data oggi;
```

4

Sintassi generale

```
struct nome_struttura {
    tipo_membro1 nome_membro1;
    tipo_membro2 nome_membro2;
    ...
    tipo_membroN nome_membroN;
};
```

`tipo_membro`: può essere un tipo sia fondamentale che derivato

5

Ancora sulla sintassi

Si può dichiarare in una sola istruzione il tipo di dato `struct data` ed alcune variabili di quel tipo

```
struct data{
    int giorno;
    char mese[20];
    int anno;
} oggi, ieri;
```

Qualora solo queste variabili debbano essere dichiarate (non è necessario conoscere il tipo di dato `data`) l'identificatore `data` può essere omissso:

```
struct {
    int giorno;
    char mese[20];
    int anno;
} oggi, ieri;
```

6

Accedere ai membri delle strutture

Per accedere ai membri (o campi) di una struttura `struct`, il C fornisce l'operatore punto "."

Esempio

`oggi.anno` accede all'intero anno del dato `oggi` di tipo `data`

7

Accesso ai membri delle strutture (cont.)

```
oggi.giorno = 25;
strcpy(oggi.mese, "Dicembre");
oggi.anno = 2006;
```

Con le prime tre istruzioni si assegna la terna di valori `<25, "Dicembre", 2006>` alla variabile `oggi`

8

Sintassi generale per l'accesso

```
nome_variabile_struttura.nome_membro
```

9

Inizializzazione delle struct

Una struttura `struct` può essere pre-inizializzata al momento della dichiarazione, mettendo le costanti che inizializzano uno per uno tutti i campi della `struct` tra parentesi graffe

Esempio

```
struct data oggi = {25, "Dicembre", 2006};
```

10

Assegnazioni tra variabili struct

Si può riferire ad una variabile di tipo struttura nel suo insieme

```
struct data compleanno, oggi;
compleanno = oggi;
```

11

Errori

```
struct s1{
    int a;
};
struct s2{
    int a;
}
struct s1 bob;
struct s2 alex;
...
alex = bob;
```

ERRORE: tipi di dati discordi

12

Errori (cont.)

```
struct s1{
    int a;
};
struct s1 bob;
int i;
...
bob = i;
```

ERRORE: tipi di dati discordi

13

Errore tipico

Confrontare le strutture è un errore di sintassi, a causa delle differenti necessità di allineamento sui vari sistemi

```
#include <stdio.h>
main() {

    struct data{
        int giorno;
        char mese[20];
        int anno;
    };
    struct data oggi = {25, "dicembre", 2006};
    struct data ieri = {24, "dicembre", 2006};
    if (oggi==ieri) printf("OK");
}
```

Occorre considerare i singoli campi delle strutture

invalid operands to binary ==

14

Nota bene

```
#include <stdio.h>
main (){
    int mese;

    struct data{
        int giorno;
        char mese[20];
        int anno;
    };
    struct s1{
        int giorno;
    };
}
```

Corretto!

- strutture diverse possono avere campi con lo stesso nome;
- i nomi dei campi possono essere gli stessi di variabili e funzioni già utilizzate

Buona abitudine è però evitare di usare gli stessi nomi, potrebbe creare confusione

15

Esempio

```
#include<stdio.h>
#include<string.h>
main()
{
    struct automobile {
        char marca[20];
        char modello[20];
        int numero_vendute;
    };

    struct automobile a1, a2;
    strcpy(a1.marca, "Ferrari");
    strcpy(a1.modello, "F40");
    a1.numero_vendute = 200;
    strcpy(a2.marca, "Opel");
    strcpy(a2.modello, "Astra");
    a2.numero_vendute = 2000;
    printf ("Marca auto = %s\n", a1.marca);
    printf ("modello auto = %s\n", a1.modello);
    printf ("vendute = %d\n", a1.numero_vendute);
    printf ("marca auto = %s\n", a2.marca);
    printf ("modello auto = %s\n", a2.modello);
    printf ("vendute = %d\n", a2.numero_vendute);
}
```

16

Esercizio

Scrivere un programma che, dati due punti sul piano cartesiano, ne calcola la distanza

Siano $A(X_A, Y_A)$ e $B(X_B, Y_B)$ due punti del piano; per il teorema di Pitagora, la distanza di A da B è data dalla formula

$$\sqrt{(X_B - X_A)^2 + (Y_B - Y_A)^2}$$

17

Soluzione

```
#include <stdio.h>
#include <math.h>

main() {
    struct punto {
        double ascissa;
        double ordinata;
    };

    struct punto punti[2];
    double delta1, delta2;
    punti[0].ascissa = 5;
    punti[0].ordinata = 7;
    punti[1].ascissa = 2;
    punti[1].ordinata = 4;
    delta1 = punti[0].ascissa - punti[1].ascissa;
    delta2 = punti[0].ordinata - punti[1].ordinata;
    printf ("%lf\n", sqrt((delta1*delta1)+(delta2*delta2)));
}
```

18

Array di strutture

Esempio

```
struct studente {
    char nome[50];
    char cognome[50];
    int anno_immatricolazione;
};

struct studente elenco[100];

elenco[2].anno_immatricolazione=2006;
```

assegna 2006 nella variabile membro `anno_immatricolazione` dello studente i cui dati sono nella 3° struttura dell'array (l'indice dell'array parte sempre da 0!)

19

Esercizio

Si realizzi un programma C che legga da utente i dati relativi ad alcuni corsi. In particolare, per ogni corso vengono dati:

- **denominazione del corso:** una stringa di 20 caratteri che riporta il nome del corso;
- **cognome del docente:** una stringa di 15 caratteri che rappresenta il cognome del docente del corso;
- **iscritti:** un intero che indica il numero di studenti che frequentano il corso.

Il programma deve stampare la denominazione del corso e il cognome del docente relativi a tutti i corsi che hanno il numero di iscritti maggiore o uguale alla media aritmetica degli iscritti (calcolata su tutti i corsi).

20

Soluzione

Attenzione: abbiamo bisogno di un ARRAY di strutture !!!!

Esempio:

l'utente inserisce i seguenti dati per 3 corsi

```
elementi di informatica
santone
55

fisica
feoli
40

matematica
cardone
37
```

La media è di 44 quindi il programma stampa:
elementi di informatica
santone

21

Soluzione

```
#include <stdio.h>
#define N 30
main(){
    struct stud {
        char denominazione[21];
        char cognome_docente[16];
        int studenti;
    } corsi[N];
    int i, nc;
    float somma, media;
    printf("Inserisci il numero dei corsi ");
    scanf("%d", &nc);
    /* inserimento dati */
    for (i=0; i<nc; i++){
        printf("Inserisci il nome del corso : ");
        scanf("%s", corsi[i].denominazione);
        printf("Inserisci il cognome del docente : ");
        scanf("%s", corsi[i].cognome_docente);
        printf("Inserisci il numero degli iscritti : ");
        scanf("%d", &corsi[i].studenti);
    }
```

Nella soluzione proposta
consideriamo nomi composti da
una sola parola

22

Soluzione (cont.)

```
somma=0;
for (i=0; i < nc; i++)
    somma = somma + corsi[i].studenti;
media= somma/nc;
for (i=0; i < nc; i++)
    if (corsi[i].studenti>=media)
        printf("%s %s\n",corsi[i].denominazione,corsi[i].cognome_docente);
} /* chiude il main */
```

23

Selection Sort applicato alle struct

Attenzione: l'ordinamento è rispetto ad UN MEMBRO numerico della struttura !!!!

```
// Ordinamento decrescente di CD musicali per anno di uscita dell'album
#include<stdio.h>
#include<string.h>
#define N 3 //supponiamo di avere 3 CD
main(){
    struct CD{
        char cantante[15];
        char titolo[20];
        int anno;
        album[N];
    } CD;
    int i, j, pmax;
    struct CD max, temp;
    //riempimento dei CD
    strcpy(album[0].cantante, "Elisa");
    strcpy(album[0].titolo, "Lotus");
    album[0].anno=2003;

    strcpy(album[1].cantante, "Nannini");
    strcpy(album[1].titolo, "Gianna best");
    album[1].anno=2007;

    strcpy(album[2].cantante, "Pausini");
    strcpy(album[2].titolo, "Io canto");
    album[2].anno=2006;
```

24

Esempio struct CD (cont)

```
// Ordinamento del vettore album rispetto all'anno
for (i=0; i<N-1; i++){
    max = album[i];
    pmax = i;
    for (j = i+1; j < N; j++){
        if (album[j].anno > max.anno) { //il termine di confronto è l'anno
            max = album[j];
            pmax = j; }
    temp = album[i]; //scambio il CD della posizione i con quello
                    //della posizione pmax
    album[i] = album[pmax];
    album[pmax] = temp;
}

// stampa il vettore ordinato
for (i=0; i<N; i++) printf("cantante %s album %s anno %d\n",
album[i].cantante, album[i].titolo, album[i].anno);
}
```

25

Esercizio

Si realizza un programma C che consenta di registrare i dati relativi agli appelli di esame del mese corrente per un certo corso. In particolare, l'utente specifica:

- **denominazione del corso**: una stringa di 30 caratteri che riporta il nome del corso;
- **numero studenti**: un intero che indica il numero di studenti che si sono registrati per l'esame.

Il programma deve consentire di introdurre i risultati della prova scritta: **PER OGNI** studente specificare:

- **matricola**: un intero che indica il numero di matricola dello studente;
- **cognome**: una stringa di 20 caratteri che rappresenta il cognome;
- **voto**: un intero positivo minore o uguale a 30 che indica il voto conseguito dallo studente alla prova scritta.

Il programma deve stampare la lista degli studenti ammessi alla prova orale, ovvero: **matricola, cognome e voto dello studente per il quale è voto >= 18**. Preferibilmente, la lista degli ammessi deve essere ordinata per voto, a partire da quello più alto.

26

Esercizio: esempio di esecuzione

INPUT:

corso: Elementi di Informatica;

numero studenti:3.

Dati studente

- matricola: 14515;
- cognome: Rossi;
- voto : 24.

Dati studente

- matricola: 14528;
- cognome: Neri;
- voto : 15.

Dati studente

- matricola: 14524;
- cognome: Verdi;
- voto : 28.

OUTPUT:

Studenti ammessi alla prova orale dell'esame di Elementi di Informatica

Matricola: 14524, Cognome:Verdi, Voto:28

Matricola: 14515, Cognome:Rossi, Voto:24

27

Soluzione

```
#include <stdio.h>
#define N 30
main() {
    struct studente{
        char cognome[15];
        int matricola;
        int voto;
    } studenti[N];

    char denominazione[40];
    int i, ns;
    int j, pmax;
    struct studente max, temp;

    //inserimento nome del corso e numero degli studenti
    printf("corso ");
    gets(denominazione); //per nomi di più parole
    printf("numero studenti ");
    scanf("%d", &ns);

    //inserimento dati per ciascuno studente
    for (i=0; i<ns; i++){
        printf("\n Dati studente:\n matricola ");
        scanf("%d", &studenti[i].matricola);
        printf("cognome ");
        scanf("%s", studenti[i].cognome);
        printf("voto ");
        scanf("%d", &studenti[i].voto);
    }
}
```

Soluzione (cont.)

```
//ordinamento del vettore studenti rispetto a studenti[i].voto
for (i=0; i<ns-1; i++){
    max = studenti[i];
    pmax = i;
    for (j = i+1; j < ns; j++){
        if (studenti[j].voto > max.voto)
            { max = studenti[j];
              pmax = j; }
    temp = studenti[i];
    studenti[i] = studenti[pmax];
    studenti[pmax] = temp;
}
printf("studenti ammessi alla prova orale del corso %s\n", denominazione);
if (studente[0].voto<18)
    printf("nessuno");
else{
    for (i=0; i<ns; i++)
        if (studenti[i].voto>=18)
            printf("matricola %d, cognome %s, voto %d\n", studenti[i].matricola,
studenti[i].cognome, studenti[i].voto);
}}
```

29

Esercizio (II parte)

Estendere il programma con la possibilità di specificare risultati di esame per appelli di più corsi. Ad esempio:

corso: Elementi di Informatica;

numero studenti:2;

Dati studente

- matricola: 14515;
- cognome: Rossi;
- voto : 24.

Dati studente

- matricola: 14528;
- cognome: Neri;
- voto : 15.

corso: Ingegneria della conoscenza;

numero studenti:2;

Dati studente

- matricola: 14411;
- cognome: Esposito;
- voto : 25.

Dati studente

- matricola: 14428;
- cognome: Bianchi;
- voto : 30.

Output:

Studenti ammessi alla prova orale dell'esame di Elementi di Informatica

Matricola: 14515, Cognome:Rossi, Voto:24

Studenti ammessi alla prova orale dell'esame di Ingegneria della conoscenza

Matricola: 14411, Cognome:Esposito, Voto:25

Matricola: 14428, Cognome:Bianchi, Voto:30

Quante strutture occorrono?

30

Esempio di soluzione del programma esteso senza ordinamento

```
#include <stdio.h>
#define N 30
#define NE 2 //supponiamo 2 corsi

main(){
    struct studente{
        char cognome[15];
        int matricola;
        int voto;
    };

    struct appello{
        char denominazione[40];
        int numero_studenti;
        struct studente studenti[N]; //vettore di strutture
    } esame[NE];

    int i, j;
    char c, pausa;
```

Soluzione del programma esteso (cont)

```
//Inserimento dati

for(j=0; j<NE; j++){ //ciclo sui corsi
    printf("corso ");
    gets(esame[j].denominazione);
    printf("numero studenti ");
    scanf("%d", &esame[j].numero_studenti);
    scanf("%c",&pausa);
    for(i=0; i<esame[j].numero_studenti; i++){ //ciclo
    sull'esame di ogni studente
        printf("matricola ");
        scanf("%d",&esame[j].studenti[i].matricola);
        scanf("%c",&pausa);
        printf("cognome ");
        gets(esame[j].studenti[i].cognome);
        printf("voto ");
        scanf("%d", &esame[j].studenti[i].voto);
        scanf("%c",&pausa);
    }
}
```

Soluzione del programma esteso (cont)

```
// Stampa gli ammessi per ciascun corso
for(j=0; j<NE; j++){
    printf("\n Studenti ammessi alla prova orale
    del corso %s\n", esame[j].denominazione);
    for(i=0; i<esame[j].numero_studenti; i++){
        if (esame[j].studenti[i].voto>=18)
            printf("matricola %d, cognome %s,
            voto %d\n",
            esame[j].studenti[i].matricola,
            esame[j].studenti[i].cognome,
            esame[j].studenti[i].voto);
    }
}}
```

Esercizio

Completare il programma esteso con l'ordinamento degli studenti rispetto al voto