UNIVERSITÀ DI PARMA

Gli Array

When you have more data you win

Sommario

UNIVERSITÀ DI PARMA

- Array monodimensionali
  - Definizione
  - Uso
  - Inizializzazione
  - Errori
- Array multidimensionali

Limite variabili

UNIVERSITÀ DI PARMA

- Fino ad ora abbiamo visto variabili “singole”
  - Gestisco uno e un solo “dato”
- Non adatte a gestire grandi moli di dati
- Eppure informatica nasce proprio per trattare dati
  - Informazione + Automatica $\longrightarrow$ Informatica
- Soluzione C $\longrightarrow$ Array

Definizione Array

UNIVERSITÀ DI PARMA

- **In C** si definisce array una struttura dati che permette di memorizzare **un numero definito** di dati tutti dello **stesso tipo**
- Caratteristiche:
  - Come per le variabili l'array ha un **nome** e uno specifico **tipo di dato**
  - Ciascun dato è detto **elemento** o componente
  - Gli elementi sono **ordinati**
    - A ogni elemento è associato un **indice numerico**

Array monodimensionali

UNIVERSITÀ DI PARMA

- Definizione array monodimensionale (vettore)

```
<tipo array> <nome array>[<numero elementi>];
```

- Esempio

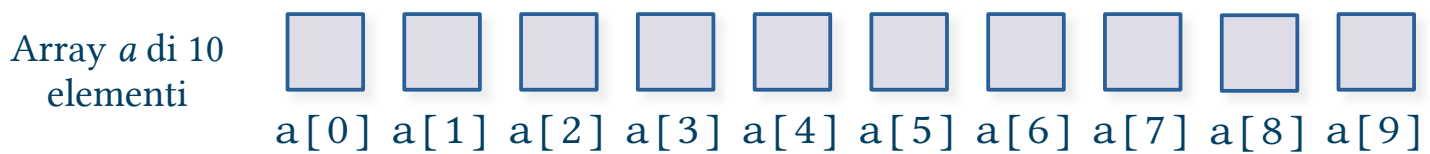
```
int numeri[100];

// definisco un array di nome “numeri” composto da
// 100 elementi di tipo int
```

Array monodimensionali

UNIVERSITÀ DI PARMA

- Definire un array di tipo x composto da N elementi equivale a
  - Definire N variabili di tipo x
    - Ma è piú pratico
- Come accedo ai singoli elementi di un array?
  - Gli elementi sono ordinati
  - Indice da 0 a N-1 ⟶ tra [e]



Array monodimensionali

UNIVERSITÀ DI PARMA

- Salvo indice, stessa sintassi variabili “regolari” per scrittura/lettura

```
int a [100];              → array di 100 interi
...
a[0] = 37;                → scrivo in primo elemento
...
printf(“%d”, a[8]);       → leggo nono elemento
...
scanf(“%d”, &a[99]);      → scrivo in ultimo elemento
```

Array monodimensionali

UNIVERSITÀ DI PARMA

- Anche gli array possono essere inizializzati quando si definiscono
- Qualche differenza:
  - **Inizializzazione completa:**

    ```
    float costanti[3] = {3.14, 2.718, 1.618};
    ```

  - **Inizializzazione completa con omissione dimensioni:**

    ```
    short primi[] = {1,3,5,7,11,13,17,19,23,29,31,37,41,43};
    ```

  - **Inizializzazione parziale:**

    ```
    char lettere[50] = {'P', 'i', 'e', 'r', 'o'};
    int tuttizeri[100]={0};  // C99 does not allow = {};
    float sparsevector[30]={[3]=4.5, [29]=1.7}; // C99
    ```

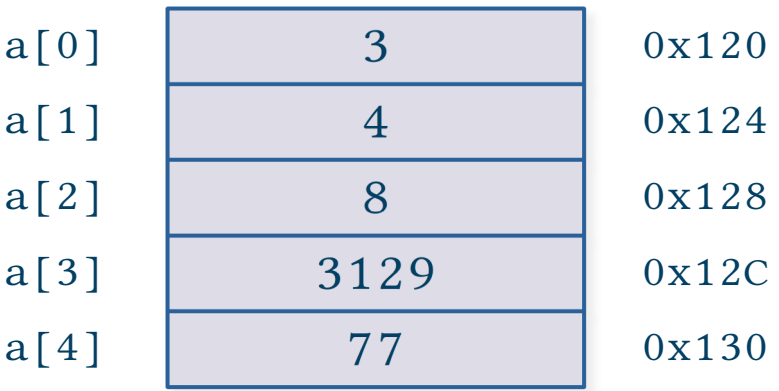
    In questi ultimi casi gli elementi non definiti sono comunque inizializzati a 0

Array e memoria

UNIVERSITÀ DI PARMA

- I singoli elementi di un array sono memorizzati consecutivamente in memoria:
- L'accesso ad un elemento avviene come: indirizzo dell'array+indice
  - Ci torneremo…

```
int a[5]={3,4,8,3129,77};
```



Array e sizeof()

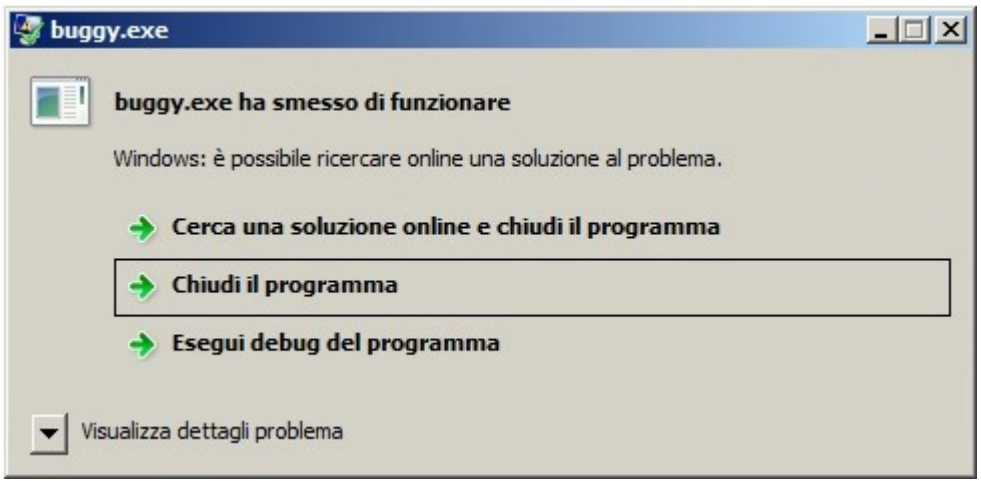
UNIVERSITÀ DI PARMA

- Anche per array posso usare macro sizeof()
  - Per singoli elementi ⟶ memoria occupata da singolo elemento
  - Per array ⟶ memoria occupata da tutto l'array
- Utile per "calcolare" numero elementi dell'array

Buffer Overrun

UNIVERSITÀ DI PARMA

- Iniziando a usare gli array vedremo spesso qualche errore
- Perché?



Buffer Overrun

UNIVERSITÀ DI PARMA

- Il C non controlla coerenza indice ↔ dimensione array

```
int dummy[100];
dummy[100]=7; // l'ultimo elemento ha indice 99
dummy[-1]=0;  // -1?
```

- Le ultime due righe sono un esempio di **buffer overrun**
  - malfunzionamento del programma
  - malfunzionamento del sistema

Array Multidimensionali

UNIVERSITÀ DI PARMA

- Gli array non sono limitati ad un solo indice
  - Ovvero a una sola dimensione
- Esempi di definizioni di array multidimensionali
  - `int matrice[10][9];  // 10 righe e 9 colonne?`
  - `long b[2][3][7][12][1024];`
- Anche in questo caso possibile inizializzare
  - `int m1[2][3]={3,4,8,1,2,7};`
  - `int m2[2][3]={{3,4,8},{1,2,7}};`

Array multidimensionali e memoria

UNIVERSITÀ DI PARMA

- Sebbene logicamente multidimensionali
- Memorizzati comunque sequenzialmente
- Posso vederli come
  - Array di array

```
int a[3][2]={3,4,8,3129,77,9};
```

a[0][0]	3	0x120
a[0][1]	4	0x124
a[1][0]	8	0x128
a[1][1]	3129	0x12C
a[2][0]	77	0x130
a[2][1]	9	0x134

UNIVERSITÀ DI PARMA

Gli Array

When you have more data you win