
Laboratorio di programmazione

Lezione IV

Tatiana Zolo

`zolo@cs.unipr.it`

GLI ARRAY (MONODIMENSIONALI)

→ **Definizione:** `type name[size]`.

`size` deve essere una costante positiva.

```
int A[10];
```

```
int float[4] = {6.5, 8, 4, 5.5};
```

→ **Accesso:** `type element = name[i]`.

`i` è l'indice dell'array e può assumere valori interi da `0` a `size - 1`.

→ **Assegnamento:** `name[i] = element`.

`element` deve essere di tipo `type`.

GLI ARRAY (MONODIMENSIONALI)

→ **Definizione:** `type name[size]`.

`size` deve essere una costante positiva.

```
int A[10];
```

```
int float[4] = {6.5, 8, 4, 5.5};
```

→ **Accesso:** `type element = name[i]`.

`i` è l'indice dell'array e può assumere valori interi da `0` a `size - 1`.

→ **Assegnamento:** `name[i] = element`.

`element` deve essere di tipo `type`.

→ un singolo elemento dell'array può essere utilizzato come una variabile semplice: `int n += A[0]*A[8]+1`.

→ Non è possibile assegnare un array ad un altro: è necessario assegnare individualmente ogni valore (es. con un ciclo "for").

→ Assenza di controllo dei limiti degli array, attenzione!

GLI ARRAY (MONODIMENSIONALI)

→ **Definizione:** `type name[size]`.

`size` deve essere una costante positiva.

```
int A[10];
```

```
int float[4] = {6.5, 8, 4, 5.5};
```

→ **Accesso:** `type element = name[i]`.

`i` è l'indice dell'array e può assumere valori interi da `0` a `size - 1`.

→ **Assegnamento:** `name[i] = element`.

`element` deve essere di tipo `type`.

→ un singolo elemento dell'array può essere utilizzato come una variabile semplice: `int n += A[0]*A[8]+1`.

→ Non è possibile assegnare un array ad un altro: è necessario assegnare individualmente ogni valore (es. con un ciclo "for").

→ Assenza di controllo dei limiti degli array, attenzione!

es. `array_min_max.cc`.

MATRICI (ARRAY BIDIMENSIONALI)

- I dati sono organizzati per righe e per colonne $\implies$ per la memorizzazione si utilizza una variable di tipo array specificando il numero di componenti per ciascuna delle due dimensioni che la costituiscono:

```
int mat[4][3];
```

mat è una variable strutturata che contiene 4 righe e 3 colonne, per un totale di 12 elementi.

- Per accedere a ciascun elemento si utilizzano due indici: il primo specifica la riga, il secondo la colonna:

```
int n = mat[1][1];
```

MATRICI (ARRAY BIDIMENSIONALI)

- I dati sono organizzati per righe e per colonne $\implies$ per la memorizzazione si utilizza una variable di tipo array specificando il numero di componenti per ciascuna delle due dimensioni che la costituiscono:

```
int mat[4][3];
```

mat è una variable strutturata che contiene 4 righe e 3 colonne, per un totale di 12 elementi.

- Per accedere a ciascun elemento si utilizzano due indici: il primo specifica la riga, il secondo la colonna:

```
int n = mat[1][1];
```

- In generale la dichiarazione degli **array multidimensionali** è

```
type name[dim1][dim2]...[dimk];
```

MATRICI (ARRAY BIDIMENSIONALI)

- I dati sono organizzati per righe e per colonne $\implies$ per la memorizzazione si utilizza una variable di tipo array specificando il numero di componenti per ciascuna delle due dimensioni che la costituiscono:

```
int mat[4][3];
```

mat è una variable strutturata che contiene 4 righe e 3 colonne, per un totale di 12 elementi.

- Per accedere a ciascun elemento si utilizzano due indici: il primo specifica la riga, il secondo la colonna:

```
int n = mat[1][1];
```

- In generale la dichiarazione degli **array multidimensionali** è

```
type name[dim1][dim2]...[dimk];
```

es. matrice.cc.

LA LIBRERIA IOSTREAM: I MANIPOLATORI

Elemento della classe iostream $\implies$ *stato di formattazione*.

Manipolatore: *modifica lo stato di formattazione*. È applicato all'oggetto stream come un dato, ma anziché provocare la lettura o scrittura di dati, esso modifica lo stato interno dell'oggetto stream.

LA LIBRERIA IOSTREAM: I MANIPOLATORI

Elemento della classe `iostream` $\implies$ *stato di formattazione*.

Manipolatore: **modifica lo stato di formattazione**. È applicato all'oggetto `stream` come un dato, ma anziché provocare la lettura o scrittura di dati, esso modifica lo stato interno dell'oggetto `stream`.

- Un valore in virgola mobile per default ha una precisione di 6 cifre: per modificarla `precision(int)` oppure `setprecision()` (è necessario file header `iomanip`).

LA LIBRERIA IOSTREAM: I MANIPOLATORI

Elemento della classe `iostream` $\implies$ *stato di formattazione*.

Manipolatore: **modifica lo stato di formattazione**. È applicato all'oggetto `stream` come un dato, ma anziché provocare la lettura o scrittura di dati, esso modifica lo stato interno dell'oggetto `stream`.

- ➔ Un valore in virgola mobile per default ha una precisione di 6 cifre: per modificarla `precision(int)` oppure `setprecision()` (è necessario file header `iomanip`).
- ➔ Un valore in virgola mobile per default è mostrato in notazione decimale fissa: per cambiare la visualizzazione nella notazione scientifica si usa `scientific`; per tornare alla visualizzazione decimale si usa `fixed`.

LA LIBRERIA IOSTREAM: I MANIPOLATORI

Elemento della classe `iostream` $\implies$ *stato di formattazione*.

Manipolatore: **modifica lo stato di formattazione**. È applicato all'oggetto `stream` come un dato, ma anziché provocare la lettura o scrittura di dati, esso modifica lo stato interno dell'oggetto `stream`.

- ➔ Un valore in virgola mobile per default ha una precisione di 6 cifre: per modificarla `precision(int)` oppure `setprecision()` (è necessario file header `iomanip`).
- ➔ Un valore in virgola mobile per default è mostrato in notazione decimale fissa: per cambiare la visualizzazione nella notazione scientifica si usa `scientific`; per tornare alla visualizzazione decimale si usa `fixed`.
- ➔ Si può controllare la larghezza di un valore numerico o di una stringa per l'output con `width(int)` oppure `setw()` (è necessario file header `iomanip`). Al contrario di tutti gli altri modificatori, non modifica lo stato di formattazione dell'oggetto `stream`.

LA LIBRERIA IOSTREAM: I MANIPOLATORI

Elemento della classe `iostream` $\implies$ *stato di formattazione*.

Manipolatore: **modifica lo stato di formattazione**. È applicato all'oggetto `stream` come un dato, ma anziché provocare la lettura o scrittura di dati, esso modifica lo stato interno dell'oggetto `stream`.

- Un valore in virgola mobile per default ha una precisione di 6 cifre: per modificarla `precision(int)` oppure `setprecision()` (è necessario file header `iomanip`).
- Un valore in virgola mobile per default è mostrato in notazione decimale fissa: per cambiare la visualizzazione nella notazione scientifica si usa `scientific`; per tornare alla visualizzazione decimale si usa `fixed`.
- Si può controllare la larghezza di un valore numerico o di una stringa per l'output con `width(int)` oppure `setw()` (è necessario file header `iomanip`). Al contrario di tutti gli altri modificatori, non modifica lo stato di formattazione dell'oggetto `stream`.
- es. `manip.cc`.

ESERCIZI

1. **Array**: leggi da standard input una sequenza di “n” numeri interi, con “n” dato di input. Calcolare la media e restituire la sequenza corrispondente traslata a media 0 (`media_array.cc`).
2. **Array**: carica i punteggi (da 1 a 10) di 2 prove effettuate da 3 concorrenti e determina la classifica sapendo che il punteggio totale di ogni concorrente è dato dalla media aritmetica delle due prove. Si visualizzino con una tabella sia i risultati parziali che il punteggio finale di ogni concorrente (`classifica.cc`).
3. **Matrici**: calcola la trasposta di una matrice le cui dimensioni ed elementi sono inseriti dall'utente (`trasposta.cc`).