4.4 Matrici

Nei paragrafi precedenti abbiamo trattato i vettori, detti anche matrici monodimensionali. Per la memorizzazione abbiamo usato una variabile di tipo array dichiarandone il numero di componenti, per esempio:

```
int vet[3];
```

Per accedere direttamente a ciascuno degli elementi del vettore si è utilizzato un indice che varia da zero a $n-1$. Nell'esempio n è uguale a 3.

In una matrice bidimensionale i dati sono organizzati per righe e per colonne, come se fossero inseriti in una tabella. Per la memorizzazione si utilizza una variabile di tipo array specificando il numero di componenti per ciascuna delle due dimensioni che la costituiscono:

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

La variabile strutturata `mat` che abbiamo dichiarato contiene 4 righe e 3 colonne per un totale di dodici elementi; per accedere a ciascuno di essi si utilizzano due indici: il primo specifica la riga il secondo la colonna. Gli indici variano rispettivamente tra 0 e $r-1$ e tra 0 e $c-1$, dove r e c sono il numero di righe e il numero di colonne. Abbiamo cioè

```
mat[0][0]  mat[0][1]  mat[0][2]
     mat[1][0]  mat[1][1]  mat[1][2]
     mat[2][0]  mat[2][1]  mat[2][2]
     mat[3][0]  mat[3][1]  mat[3][2]
```

Per esempio, `mat[1][2]` fa riferimento all'elemento presente nella seconda riga della terza colonna. Ogni colonna della matrice bidimensionale non è altro che un vettore.

Il formato generale della dichiarazione degli array multidimensionali è il seguente:

```
tipo nome[dimensione1][dimensione2]...[dimensioneN];
```

Per esempio, al fine di memorizzare i ricavi ottenuti dalla vendita di 10 prodotti in 5 punti vendita nei dodici mesi dell'anno, potremmo utilizzare la matrice tridimensionale `marketing` così dichiarata:

```
int marketing[10][5][12]
```

Scriviamo ora un programma che richiede all'utente i valori da inserire, li memorizza nella matrice bidimensionale `mat` e la visualizza (Listato 4.3).

```
/* Caricamento di una matrice */

#include <stdio.h>

int mat[4][3];

main()
{
int i, j;

printf("\n \n CARICAMENTO DELLA MATRICE \n \n");
for(i=0; i<4; i++)
  for(j=0; j<3; j++) {
    printf("Inserisci linea %d colonna %d val: ", i, j);
    scanf("%d", &mat[i][j]);
  };

/* Visualizzazione */
for(i=0; i<4; i++) {
  printf("\n");
  for(j=0; j<3; j++)
    printf("%5d", mat[i][j]);
}
}
```

Listato 4.3 Esempio di utilizzo di un array bidimensionale

Per effettuare il caricamento dei dati nella matrice utilizziamo due cicli, uno più esterno che mediante la variabile `i` fa la scansione delle righe da zero a 3 (4–1) e un altro che percorre, per mezzo della variabile `j`, le colonne da zero a 2 (3–1):

```
for(i=0; i<4; i++)
       for(j=0; j<3; j++) {
         printf("Inserisci linea %d colonna %d val:", i, j);
         scanf("%d", &mat[i][j]);
       };
```

Viene riempita tutta la prima riga poi la seconda e così via; se l'utente passa i valori 2, 55, 12, 98, 34... essi verranno inseriti negli elementi `mat[0][0]`,`mat[0][1]`,`mat[0][2]`,`mat[1][0]`,`mat[1][1]`...

Si può ottenere il caricamento per colonne invertendo semplicemente i due cicli:

```
for(j=0; j<3; j++)
       for(i=0; i<4; i++) {
         printf("Inserisci linea %d colonna %d val:", i, j);
         scanf("%d", &mat[i][j]);
       };
```

Il numero totale d'iterazioni è sempre uguale a 12 e gli indici `j` e `i` fanno la scansione delle colonne e delle righe della matrice senza fuoriuscire dai margini. La visualizzazione è ancora una volta ottenuta con due cicli `for` annidati uno nell'altro.

Nel programma precedente le dimensioni della matrice erano fissate a priori: modifichiamolo in modo da far decidere all'utente il numero delle righe e delle colonne, come nel Listato 4.4.

```
/* Caricamento di una matrice
   le cui dimensioni vengono decise dall'utente */

#include <stdio.h>

#define MAXLINEE 100
#define MAXCOLONNE 100
int mat[MAXLINEE][MAXCOLONNE];

main()
{
int n, m;
int i, j;

/* Richiesta delle dimensioni */
do {
  printf("\nNumero di linee: ");
  scanf("%d", &n);
}
while((n>=MAXLINEE) || (n<1));

do {
  printf("Numero di colonne: ");
  scanf("%d", &m);
}
while((m>=MAXCOLONNE) || (m<1));

printf("\n \n CARICAMENTO DELLA MATRICE \n \n");
for(i=0; i<n; i++)
  for(j=0; j<m; j++) {
    printf("Inserisci linea %d colonna %d val:", i, j);
    scanf("%d", &mat[i][j]);
  };

/* Visualizzazione */
for(i=0; i<n; i++) {
  printf("\n");
  for(j=0; j<m; j++)
    printf("%5d", mat[i][j]);
}
}
```

Listato 4.4 Inizializzazione di una matrice bidimensionale, seconda versione

La matrice viene definita con un massimo numero di linee e di colonne:

```
int mat[MAXLINEE][MAXCOLONNE];
```

dove `MAXLINEE` e `MAXCOLONNE` sono due costanti che abbiamo dichiarato precedentemente, il cui valore deve essere scelto in relazione alle massime dimensioni. Successivamente si richiede l'inserimento del valore di *n*, numero di linee che realmente verranno riempite:

```
do {
       printf("Numero di linee: ");
        scanf("%d", &n);
```

```
    }
    while((n>=MAXLINEE) || (n<1));
```

L'istruzione `scanf` viene inserita in un ciclo `do-while` in modo che se *n* è maggiore del numero di linee che costituiscono la matrice o è minore di 1 il valore non viene accettato e la richiesta viene ripetuta. Analogamente si procede per le colonne.