

Matrici



*Just Say
Ho!*

Merry Christmas



Programmazione elementare

Strutture di controllo base.

Conoscenza e uso dei vettori 1D

Algoritmi notevoli ricerca e
ordinamento.



Conoscenza delle librerie

Creazione e utilizzo



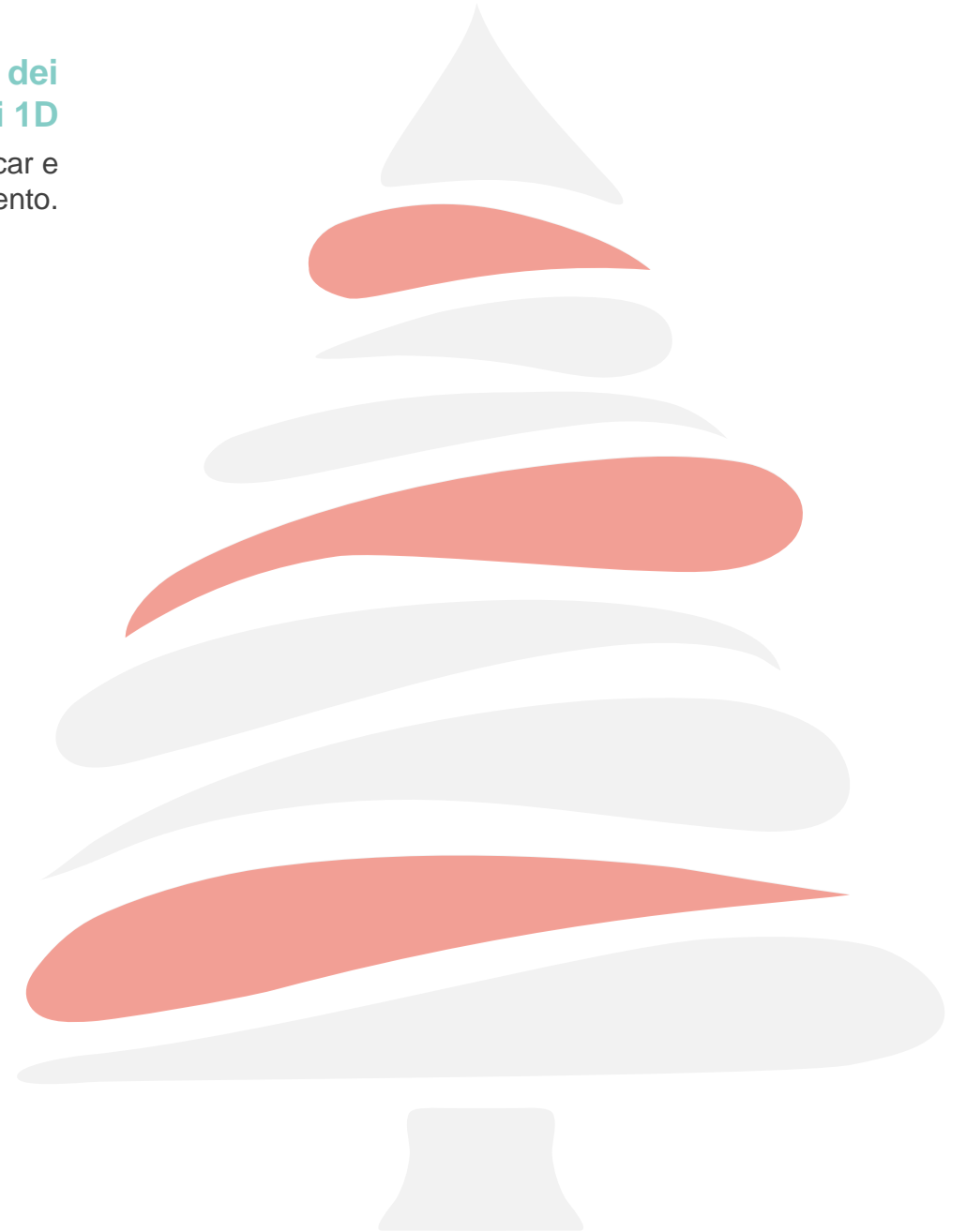
Conoscenza del passaggio di parametri

Tre tipi di passaggio



PREREQUISITI

Merry Christmas



Introduzione

La struttura dati vettore è anche detta **array monodimensionale (1D)**, perché ogni componente può essere selezionata mediante *un solo* indice.

In alcuni casi (come ad esempio accade in una *tabella*) si possono creare **array bidimensionali (2D)** nei quali sono necessari *due indici* per individuare una componente.



Array bidimensionale

In alcuni tipi di problemi può nascere la necessità di *memorizzare dati su due dimensioni* anziché su una sola. In questo caso il modello di dato da utilizzare si dice **array bidimensionale** o **matrice**.

Dal punto di vista della *rappresentazione logica* essa si presenta come una tabella formata da **righe** e **colonne**, come indicato per la matrice





0	5	5
1	2	2
2	0	1
3	2	12
4	18	54

Merry Christmas

Array bidimensionale

Esempio di matrice
con 5 righe e 2
colonne

NB: Anche le
matrici, come i
vettori, sono
memorizzate in
RAM.

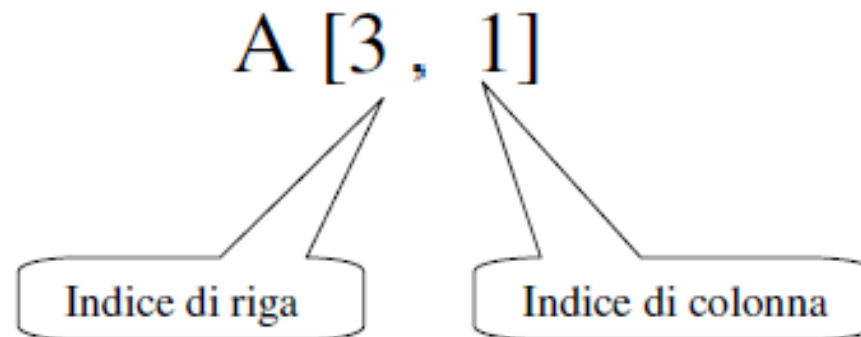
Matrice, indici, componenti

- Una matrice è una struttura dati contenente elementi detti **componenti**.
- Ogni componente è individuabile con due **indici** (o più di due se la matrice ha più di 2 dimensioni).
- Ad esempio la matrice della figura precedente, contiene 10 componenti.
- **Attenzione:** Nelle applicazioni pratiche può anche capitare di dover utilizzare **array a più di 2 dimensioni**.



Matrice, indici, componenti

- Supponiamo che la matrice precedente si chiami **A**. Nella **rappresentazione logica** la posizione di *ogni componente* è *individuata dai valori dei due indici* (riga e colonna), per cui la generica componente si indicherà con $mat[i,j]$, avendo assunto i come indice di riga e j come indice di colonna.



Caratteristiche della matrice

- Notare che:
- le componenti devono essere tutte dello **stesso tipo** (tutte interi, reali o caratteri)
- gli indici di riga e di colonna **partono sempre dal valore 0**
- gli indici possono essere espressioni (se ad es. $i=3$, $j=2$, $m[i+1][j-1]$ indica il contenuto di $m[4][1]$)
- il numero **righe*colonne** prende il nome di **dimensione della matrice**.



Dichiarazione di una matrice

- Per **dichiarare** una variabile di tipo matrice, analogamente a quanto fatto per l'array, scriviamo:

tipo[,] *nomeMatrice* = new *tipo*[*M,N*];

- dove:
- *tipo* è il tipo delle componenti;
- *nomeMatrice* è il nome collettivo
- *M* indica il numero delle righe
- *N* indica il numero delle colonne
- Ad esempio, per dichiarare una matrice A di interi di dimensione 5x2

int[,] A= new int[5,2];



Dichiarazione di una matrice

- Esempi di dichiarazioni:

```
int[,] matrix;  
double[,] mat;  
string[,,] cube;
```

- Esempi di creazioni (istanze):

```
int[,] matrix = new int[3, 4];  
double[,] mat = new double[8, 2];  
string[,,] cube = new string[5, 5, 5];
```



Accedere alle celle

- Per leggere un valore contenuto in una cella:

```
int[,] mat = {{1, 2}, {3, 4}}  
int element11 = mat[1, 1]; // element11 = 4
```

- Per assegnare un valore ad una cella:

```
int[,] mat = new int[2, 2];  
mat[1, 1] = 5;
```

- Per assegnare dei valori a tutte le celle

```
int[,] mat= new int[3, 4];  
for (int i=0; i<mat.GetLength(0); i++)  
    for (int j=0; j<mat.GetLength(1); j++)  
        mat[i, j] = i + j;
```

Numero di
righe

Numero di
colonne





OPERAZIONI BASE SU MATRICI



Operazioni base su matrici

- Data ad esempio, la matrice

```
int[,] m= new int[2,3];
```

- le **operazioni base** su essa sono:
- **lettura** di una componente (es. **Leggi** (m[i,j]));
- **stampa** di una componente (es. **Stampa** (m[i,j]));
- **assegnazione** di un valore ad una componente (es. m[i,j] = x);



Lettura di una matrice

- **ATTIVITÀ:** scrivere il metodo *LeggiMat()* che ricevendo come parametri due interi crei una matrice e la riempia di valori letti da input

```
static int[,] LeggiMat(int nR,int nC)
{
    int[,] dep = new int[nR, nC];
    for(int i=0;i<dep.GetLength (0);i++)
        for (int j = 0; j < dep.GetLength(1); j++)
        {
            dep[i, j] = Library.LeggiInt(string .Format("elem[{0},{1}]", i, j));
        }

    return dep;
}
```



Creare una matrice con valori Random

```
static Random rnd = new Random();
```

0 references

```
static int[,] CreaRndMat(int nr, int nc)
{
    int[,] mat = new int[nr, nc];
    for (int i = 0; i < mat.GetLength(0); i++)
        for (int j = 0; j < mat.GetLength(1); j++)
            mat[i, j] = rnd.Next(-10, 11);
    return mat;
}
```



Creare una matrice caricando da file

- Vedere in laboratorio



Stampa di una matrice

- **ATTIVITÀ:** scrivere il metodo *StampaMat()* che ricevendo come parametro una matrice la visualizzi

```
static void StampaMat(int[,] m)
{
    for (int i = 0; i < m.GetLength(0); i++)
    {
        for (int j = 0; j < m.GetLength(1); j++)
        {
            Console.Write("{0} ", m[i, j]);
        }
        Console.WriteLine();
    }
}
```



Main

```
static void Main(string[] args)
{
    int numR=Library.LeggiIntPos("num righe: ");
    int numC=Library.LeggiIntPos("num colonne: ");
    int[,] mat= LeggiMat (numR ,numC );
    StampaMat(mat);
}
```



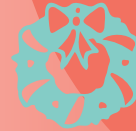


ESERCIZI

1 Data una matrice A rettangolare di interi, determinare e stampare il massimo di ogni sua riga



2 Data una matrice A rettangolare di interi, stabilire se è la matrice nulla (tutti zeri)



3 Data una matrice A rettangolare di interi, stabilire se si tratta della matrice identità (1 sulla diagonale principale, zero altrimenti).



4 Creare una matrice quadrata a dieci righe e dieci colonne, contenente i valori della tabellina pitagorica



Stampa di una matrice

- **ATTIVITÀ:** Scrivere i metodi che generino e visualizzino le matrici quadrate sotto riportate

1	...	1	...	1	...	1
0	...	0	...	0	...	0
1	...	1	...	1	...	1
0	...	0	...	0	...	0

1	...	0	...	0	...	1
1	...	0	...	0	...	1
1	...	0	...	0	...	1
1	...	0	...	0	...	1

1	...	0	...	1	...	0
1	...	0	...	1	...	0
1	...	0	...	1	...	0
1	...	0	...	1	...	0

1	...	1	...	1	...	1
0	...	0	...	0	...	0
1	...	1	...	1	...	1
0	...	0	...	0	...	0

1	...	1	...	1	...	1
1	...	0	...	0	...	1
1	...	0	...	0	...	1
1	...	1	...	1	...	1

1	...	0	...	0	...	1
0	...	1	...	1	...	0
0	...	1	...	1	...	0
1	...	0	...	0	...	1

0	...	0	...	0	...	0
0	...	1	...	1	...	0
0	...	1	...	1	...	0
0	...	0	...	0	...	0





Merry
Christmas

&

AND HAPPY

NEW YEAR!