INFORMATICA A

A.A. 2018-19

Laboratorio n°3

Dott. Michele Zanella

Ing. Gian Enrico Conti

Matrici

- Sono degli array bidimensionali (righe,colonne)

```
int matrix[10][10]
```

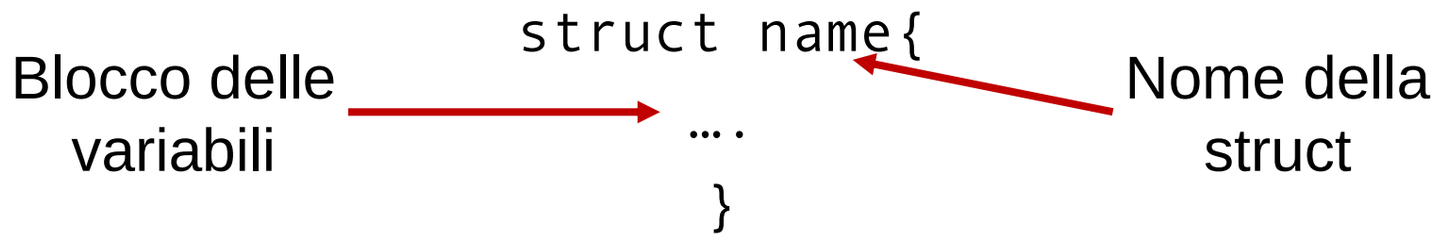
- Inizializzazione degli array e delle matrici:

```
char a[2] = {‘a’, ‘b’};

int m[2][2] = {
    {0, 1}, // Riga 0
    {2, 3}  // Riga 1
};
```

Struct

- Sono dei tipi di dato composti da una serie di variabili di vario tipo poste sotto lo stesso blocco di memoria, in questo modo esse sono accessibili da un singolo *puntatore* o tramite una variabile istanziata come il tipo della *struct*.



- Posso definire un nuovo tipo di dato che si riferisce alla struct in modo da poter istanziare delle variabili tramite l'operatore *typedef*:

```
typedef struct name{ …
        } type_name
```

Nome del nuovo tipo

- Per dichiarare una variabile del tipo da noi definito:

```
type_name var;
```

Struct

- Esempio, dichiariamo una struct per definire un punto cartesiano:

```
typedef struct point{
    int x; // Coordinata x
    int y; // Coordinata y
} point_t;

point_t p;
```

- Per accedere e assegnare ai campi della variabile *p* utilizziamo la notazione puntata ossia l'operatore "*.*" :

```
p.x = 10; // Coordinata x del punto p
p.y = 20; // Coordinata y del punto p

printf(" Le coordinate di p sono (%d,%d)\n", p.x, p.y);
```

Esercizio 3.1: Somma tra matrici

Scrivere un programma che date due matrici ne calcoli la matrice somma. Le matrici devono avere stesse dimensioni tra di loro.

Esempio:

$$\begin{bmatrix} 4 & 2 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 3 & 10 \\ 6 & 5 \end{bmatrix} = \begin{bmatrix} 7 & 12 \\ 6 & 6 \end{bmatrix}$$

Esercizio 3.2: Matrice - Riempimento concentrico

Scrivere un programma che riempia una matrice in modo concentrico, ricevendo gli interi da I/O, e che stampi la matrice formattata.

Esercizio 3.3: Rubrica del telefono

Si scriva un programma che implementi una semplice rubrica del telefono.

Requisiti:

- Possibilità di inserire/cancellare contatti
- Possibilità di stampare elenco contatti con informazioni
- Possibilità di ricercare un contatto dato il cognome

Hints:

- Riutilizzare eventuali funzioni contenuti in librerie (es. `string.h`)
- Settare un numero massimo `CONT_MAX` di contatti

Esercizio 3.3: Rubrica del telefono (cont'd)

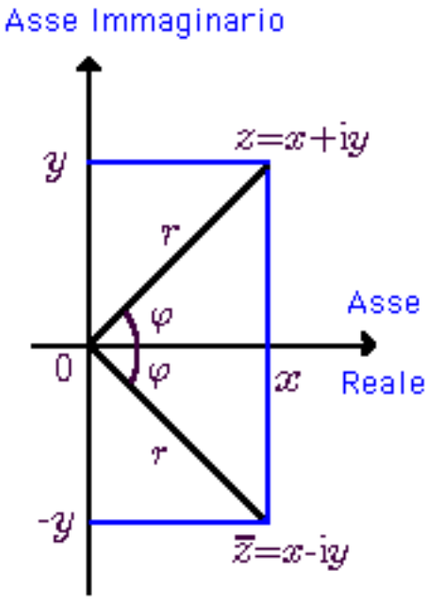
Contatto:

Definiamo un tipo di dato personalizzato *contatto*. Il tipo deve includere le informazioni relative a:

- Cognome
- Nome
- Età
- Numero di telefono

Esercizio a casa 3.1: Numeri complessi

1) si scriva una opportuna *struct* del C atta a rappresentare i numeri complessi.



Un numero complesso deve avere parte Reale e parte Immaginaria.

2) si dichiari una variabile N del tipo struct appena creato.

Esercizio 2: Numeri complessi (cont'd)

Si scrivano un frammento di codice che:

3) legga le componenti R e I del numero N

4) calcoli e stampi il Modulo di N acquisito

5) legga due numeri complessi N1 e N2 e ne calcoli la somma (la somma è a sua volta un numero complesso)

6) legga due numeri complessi N1 e N2 e dica se sono uguali in argomento (fase)

7) legga un array (detto "valori") di "MAX" numeri complessi e stampi il più grande in modulo