



UNIVERSITY OF TRENTO - Italy

Department of Information
and Communication Technology

Laboratorio 7

Pierluigi Roberti
Carmelo Ferrante

DISI – aa 2024-25
Università degli Studi di
Trento

pierluigi.roberti@unitn.it

Uso parola chiave typedef

Creazione di un alias di tipo (iniziale maiuscola)

Sintassi:

```
typedef tipo NomeNuovoTipo;
```

Esempio

```
typedef int Lunghezza;
```

```
typedef char Stringa[30];
```

Dichiarazione variabili basate sul nuovo tipo

```
Lunghezza l;
```

```
Stringa nome, cognome;
```

Variabili strutturate (struct)

- Sintassi

```
struct NomeStruttura {  
    Tipo1 nomeCampo1;  
    Tipo2 nomeCampo2;  
    ...  
    TipoN nomeCampoN;  
} s1, ... , sN;
```

← VARIABILI

- Accesso a un campo:

```
s1.nomeCampo1 = 0;
```

Uso di typedef e struct

Dichiarazione tipo di dato basato su struttura anonima

```
typedef struct {  
    Tipo1 nomeCampo1;  
    ...  
    TipoN nomeCampoN;  
} NomeTipo;
```

Il nuovo tipo di dato viene messo **prima del main** in modo che sia «visibile/usabile» in ogni punto del programma!

Dichiarazione tipo di dato basato su struttura NON anonima

```
typedef struct NomeTipo {  
    Tipo1 nomeCampo1;  
    ...  
    TipoN nomeCampoN;  
} NomeTipo;
```

Dichiarazione variabile basata sul nuovo tipo di dato definito

```
NomeTipo s1;
```

Esempi uso struct e typedef

Definizione dei dati relativi a 2 rette ($y = mx + q$)

//uso variabili singole

```
float diff;  
float m1, q1, m2, q2;  
m1=2;      q1=-3.0;      scanf ("%f", &m1) ;  
m2=-1;     q2=2.0;  
diff = m1-m2;
```

m1 

q1 



m2 

q2 



//uso di 2 array

```
float m[2], q[2];  
m[0]=2;  q[0]=-3.0;      scanf ("%f", &m[0]) ;  
m[1]=-1; q[1]=2.0;  
diff = m[0]-m[1];
```

m

0	1
	

q

0	1
	

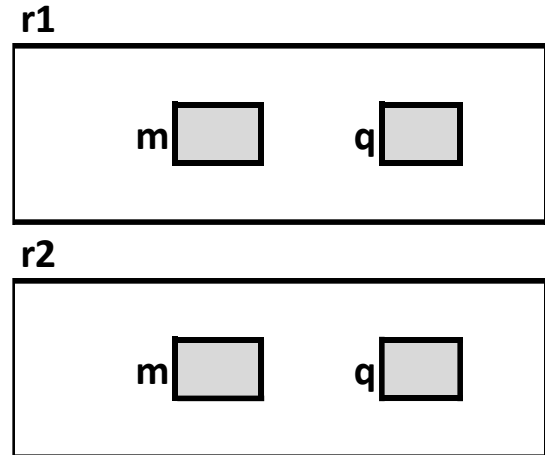
Esempi uso struct e typedef

Definizione dei dati relativi a 2 rette ($y = mx + q$)

```
//uso di 2 variabili struct
```

```
struct {  
    float m, q;  
} r1, r2;
```

```
r1.m=2;    r1.q=-3.0;    scanf("%f",&r1.m);  
r2.m=-1;   r2.q=2.0;  
diff = r1.m-r2.m;
```

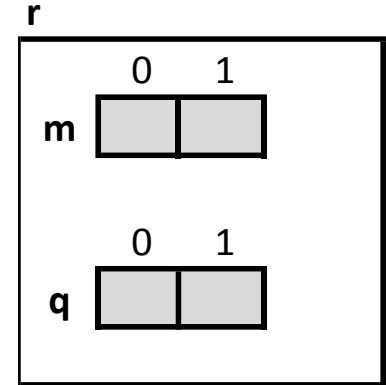


Esempi uso struct e typedef

Definizione dei dati relativi a 2 rette ($y = mx+q$)

//uso di 1 variabili struct con array

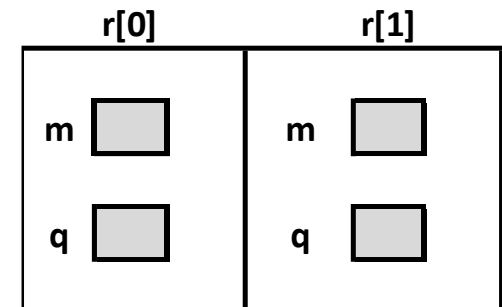
```
struct {  
    float m[2];  
    float q[2];  
} r;
```



```
r.m[0]=2;   r.q[0]=-3.0;   scanf("%f",&r.m[0]);  
r.m[1]=-1;  r.q[1]=2.0;  
diff = r.m[0]-r.m[1];
```

//uso di array di struct

```
struct{  
    float m,q;  
} r[2];
```



```
r[0].m=2;   r[0].q=-3.0;   scanf("%f",&r[0].m);  
r[1].m=-1;  r[1].q=2.0;  
diff = r[0].m-r[1].m;
```

Esempi uso struct e typedef

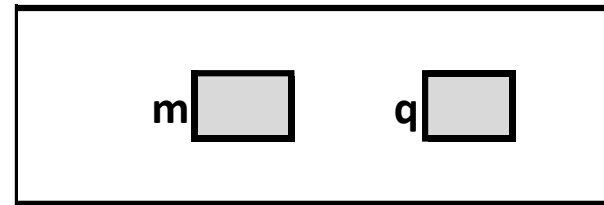
Definizione dei dati relativi a 2 rette ($y = mx + q$)

```
//uso di un tipo di dato (non anonimo)
```

```
//(fuori dal main)
```

```
typedef struct Retta{  
    float m,q;  
}Retta;
```

Retta



```
int main() {  
    Retta r1, r2;  
    r1.m=2; r1.q=-3.0;  scanf("%f",&r1.m);  
  
}
```


Esercizio 1

Scrivere un programma che

- genera in modo casuale le posizioni di NPUNTI punti P nello spazio 2D
- richiede all'utente le coordinate di un punto Q(x,y)
- stampa a video le coordinate del punto P più vicino al punto Q in termini di distanza euclidea

Note:

- I punti P e il punto Q sono di tipo **float**
- `#include <math.h>` per funzioni matematiche
- `sqrt(val)` restituisce la radice quadrata di val

Versione A: I punti P sono memorizzati in un **vettore** di **BI-dimensionale** di lunghezza NPUNTI

Versione B: I punti P sono delle **strutture** composte da due campi x e y

Esercizio 1 - VersioneA

```
#define N_DIMENSIONI 2  
#define N_PUNTI 100
```

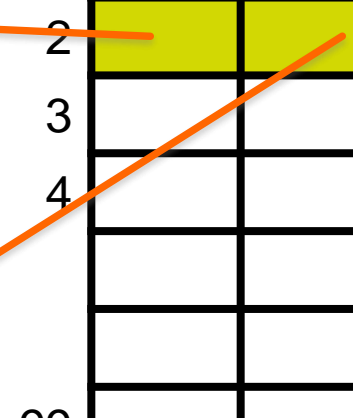
```
float punti[N_PUNTI][N_DIMENSIONI]
```

Coordinata x del punto P
con indice 2

Coordinata y del punto P
con indice 2

Terzo punto di coordinate (4,-2)
`punti[2][0] = 4;`
`punti[2][1] = -2;`

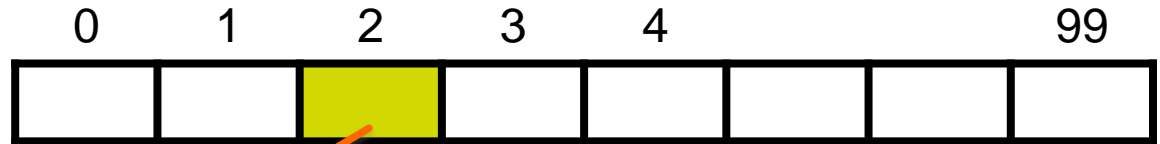
	x	y
	0	1
0		
1		
2		
3		
4		
99		



Esercizio 1 – Versione B

```
#define N_DIMENSIONI 2  
#define N_PUNTI 100  
typedef struct Tpunto{  
    float x, y;  
} Tpunto;
```

```
//nel main dichiarare:  
float punti[N_PUNTI][N_DIMENSIONI]  
Tpunto punti[N_PUNTI];
```



punto P con indice 2

Ogni elemento è
una struttura Tpunto

Terzo punto di coordinate (4,-2)

```
punti[2].x = 4;
```

```
punti[2].y = -2;
```

Soluzione esercizio 1 Versione B – 1/2

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h> /*libreria matematica*/
#define NPUNTI 100
typedef struct Tpunto{
    float x, y;
} Tpunto;

int main(int argc, char *argv[])
{
    Tpunto punti[NPUNTI];
    float x, y, dist_min, dist, deltax, deltax;
    int i, p_min;
    for(i=0 ; i<NPUNTI ; i++){
        punti[i].x = (rand()%100000)/1000.0;
        punti[i].y = 1.0*(rand()%100000)/1000;
    }
    // for(i=0 ; i<NPUNTI ; i++){
    //     printf("%f %f\n", punti[i].x, punti[i].y);}
```

Esercizio 2

Data una lista di voti di diversi studenti, calcolare

- la media dei voti di ciascuno studente
- la media dei voti di ciascun esame
- la media della media di tutti gli studenti

Ipotesi:

- gli studenti vengono riconosciuti con un numero intero; il numero di studenti è NUM_STUDENTI
- tutti gli studenti seguono gli stessi esami; gli esami sono riconosciuti da un indice (0, 1, 2, 3, ...); il numero di esami è NUM_ESAMI
- i voti sono numeri interi compresi tra 18 e 30 (inizializzare la matrice dei voti con valori casuali)
- tutti gli studenti non hanno tutti i voti (per esempio voto mancante = 0)

Esercizio 2

```
#define NUM_STUDENTI 4  
#define NUM_ESAMI 8
```

```
int tabella_voti[NUM_STUDENTI][NUM_ESAMI]
```

Voti da considerare
per il calcolo della
media dell'esame
numero 8

		Esami							
		0	1	2	3	4	5	6	7
Studenti	0								
	1			30					
	2					28			
	3								

Voto Studente 2 all'esame 3

```
tabella_voti[2][3]=30;
```

Voti da considerare
per il calcolo della
media dello
studente numero 4

Voto dello studente 3 all'esame 5

```
tabella_voti[3][5]=28;
```

Esercizio 3

Definite un nuovo tipo di dato *Tstudente* (tramite keyword **struct**) una struttura che contiene tre campi, ovvero *cognome*, *nome* e un vettore di interi (di dimensione **NUM_ESAMI**) che rappresenta il voto degli *esami*.

Vanno definiti con *#define* il numero massimo di esami (**NUM_ESAMI**), studenti (**NUM_STUDENTI**) e lunghezza di cognome e nome (**NUM_CAR** => 20)

Nel main definire una variabile *studente* array di tipo *Tstudente* composta da **NUM_STUDENTI**

Ipotesi:

- il numero di studenti è **NUM_STUDENTI** (pari a 4)
- tutti gli studenti seguono gli stessi esami; gli esami sono riconosciuti da un indice (0, 1, 2, 3, ...); il numero di esami è **NUM_ESAMI** (pari a 5)

Inizializzazione:

- inizializzare l'array *studente*
 - Inizializzando da tastiera il campo *nome* e *cognome*;
 - Assegnando dei valori casuali all'array *esami* (interi compresi tra 18 e 30)
- tutti gli studenti hanno un voto per ogni esame

Calcolare:

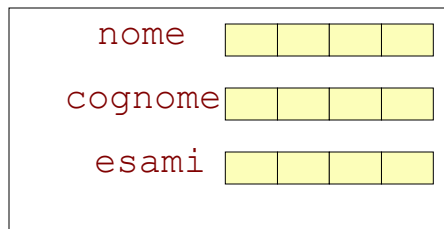
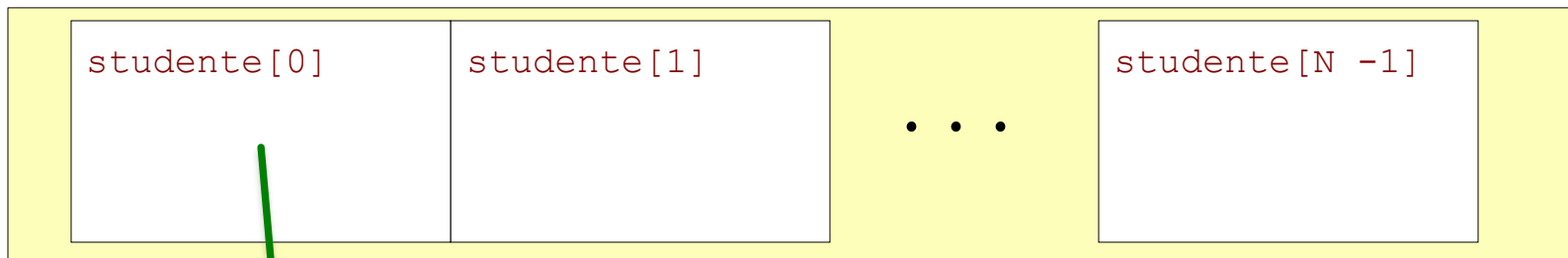
- la media dei voti di ciascuno studente
- la media dei voti di ciascun esame

Tstudente studente[NUM_STUDENTI]

Array di N elementi di tipo Sstudente

Nome dell'array: studente

Variabile
definita nel
MAIN

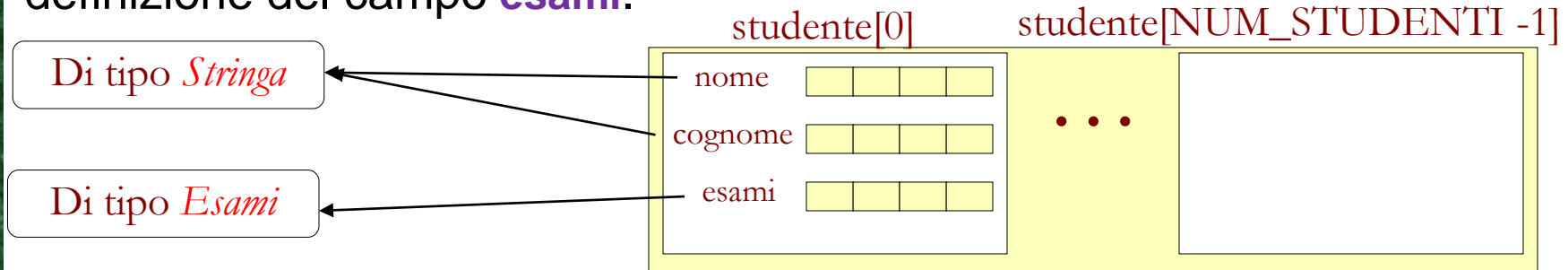


```
typedef struct Tstudente{  
    char cognome[NUM_CAR] ;  
    char nome[NUM_CAR] ;  
    int esami[NUM_ESAMI] ;  
}Tstudente ;
```

*Struct di tipo **Tstudente***

Esercizio 4

Estendere la struttura dell'esercizio precedente definendo un tipo *Stringa* per la definizione dei campi **nome** e **cognome** ed un tipo *Esami* per la definizione del campo **esami**.



- Per ogni **studente** inizializzare la struttura come segue:
 - Impostare tutti i voti a 0
 - nome composto da due caratteri casuali
 - cognome composto da due caratteri casuali
- Per ogni **studente** assegnare i voti come segue:
 - Generare un numero K compreso tra 0 e `NUM_ESAMI-1`
 - assegnare casualmente K voti (compresi tra 18 e 30)
- Visualizzazione nomi e dei voti degli studenti
- Calcolare la Media di ogni studente. NOTA: tenere conto del numero reali di esami superati, quelli con voto $\neq 0$ e del fatto che nel calcolo della media uno studente potrebbe non avere nessun esame superato

Esercizio 4 – Strutture dati

```
typedef char Stringa [NUM_CAR];
```

```
typedef int Esami [NUM_ESAMI];
```

```
typedef struct Tstudente {
```

```
    Stringa cognome;
```

```
    Stringa nome;
```

```
    Esami esami;
```

```
} Tstudente;
```

```
Tstudente studente [NUM_STUDENTI];
```