

# Fondamenti di Informatica T-1

## Modulo 2

---

# Obiettivi di questa esercitazione

---

1. Array

# Esercizio 1

(array)

---

- Creare un programma che legga da input un numero non noto a priori di interi (al più 10) terminati da 0. Tale sequenza può eventualmente contenere numeri ripetuti.
- Si memorizzi tale sequenza in un vettore di opportuna dimensione.
- Si stampino a video tutti i numeri per cui il successivo nel vettore è pari al numero stesso

# Esercizio 2

(array)

---

- Creare un programma che legga da input un numero non noto a priori di interi positivi (al più 10) terminati da 0.
- Qualora l'utente inserisca dei valori negativi, tali valori devono essere scartati e non considerati
- Si memorizzi tale sequenza in un vettore di opportuna dimensione.
- Si stampino a video tutti i numeri che sono allocati nel vettore in posizioni il cui indice è uguale al numero stesso

# Esercizio 3

(array)

---

- Creare un programma che legga da input un numero non noto a priori di interi positivi (al più 10) terminati da 0.
- Si memorizzi tale sequenza in un vettore di opportuna dimensione, rispettando l'ordine con cui i valori sono stati inseriti.
- Si memorizzino poi in un secondo vettore i valori del primo, avendo cura di inserirli in ordine inverso
- Si stampi infine il vettore coi valori in ordine invertito

# Esercizio 4

## (array)

---

- Realizzare un programma che legga da input una sequenza di interi positivi, terminati da 0.
- Tali numeri devono essere memorizzati in un array (di dimensione massima 10)
- Il programma quindi provveda a stampare a video tutti i numeri pari che sono memorizzati nell'array in una posizione con indice pari
- Estensione: si abbia cura di verificare che siano immessi numeri fino al limite di 10 elementi; dopo tale limite, il programma stampi un messaggio di errore, finchè la sequenza non è terminata da 0.
- Estensione: riprogettare il programma spostando la fase di acquisizione dei dati e l'algoritmo di stampa in due funzioni apposite.

# Esercizio 5

## (array)

---

Si scriva un programma che

- 1) richieda all'utente un valore **V** di soglia;
- 2) successivamente prenda in ingresso una sequenza di reali positivi terminata da 0 (massimo 10), e memorizzi in un vettore di float **M** (di dimensione fisica 10) SOLO i valori maggiori di V;
- 3) infine crei un secondo vettore **MED** in cui l'elemento *i*-esimo è calcolato come la media tra l'elemento *i*-esimo del vettore M e il valore V.

## Esercizio 5

(array)

---

Esempio: l'utente inserisce il valore 2.5 di soglia.

Poi inserisce la sequenza

1.3    4    5.2    9.5    2.2    1    0

Nel vettore M vengono quindi memorizzati solo

	0	1	2
M	4	5.2	9.5

Infine, il programma deve creare un secondo vettore MED in cui l'elemento i-esimo e' calcolato come la media tra l'elemento i-esimo del vettore M e il valore V.

	0	1	2
MED	3.25	3.85	6.0



# Esercizio 6

## (array)

---

Si scriva un programma che prenda in ingresso una sequenza di massimo 10 reali positivi terminata da 0, e la memorizzi in un vettore di float **NUM**.

Il programma deve creare un secondo vettore **MEDIE** in cui l'elemento *i*-esimo è calcolato come la media tra l'elemento *i*-esimo del vettore **NUM** e il suo successivo. Ovviamente la dimensione logica di medie sarà pari alla dimensione logica di **NUM** meno 1.

# Esercizio 6

(array)

---

Esempio: l'utente inserisce la sequenza

1.3 4 5.2 9.5 2.2 1 0

<b>NUM</b>	1.3	4	5.2	9.5	2.2	1	0
------------	-----	---	-----	-----	-----	---	---

<b>MEDIE</b>	2.65	4.60	7.35	5.85	1.6	0.5
--------------	------	------	------	------	-----	-----

# Esercizio 7

(array)

---

Scrivere un programma che

1) Legga da input due vettori **V1** e **V2** di interi di dimensione  $N$ ;

2) costruisca un terzo vettore **V3** di dimensione  $2N$  i cui elementi di posizione pari siano gli elementi del primo vettore e gli elementi di posizione dispari siano gli elementi del secondo vettore.

**v1**

4	5	9
---	---	---

**v2**

2	6	1
---	---	---

**v3**

0	1	2	3	4	5
4	2	5	6	9	1

# Esercizio 8

(array)

---

Scrivere un programma che, dato un vettore **NUM** di **N** interi positivi inseriti dall'utente, ne produca due **PAR** e **DIS** contenenti, rispettivamente, i numeri pari e dispari del vettore iniziale.

Si controlli che i numeri inseriti dall'utente siano positivi.

Per verificare se un numero è pari, si scriva una funzione che restituisca 1 se il numero è pari, 0 altrimenti

```
int pari(int n);
```

# Esercizio 8

(array)

---

Esempio: l'utente inserisce la sequenza

4 2 **-4** 5 6 **-9** 1 6  
          ↓     *scartati*     ↓

NUM 

4	2	5	6	1	6
---	---	---	---	---	---

PAR 

4	2	6	6		
---	---	---	---	--	--

***DIMENSIONE LOGICA 4***

DIS 

5	1				
---	---	--	--	--	--

***DIMENSIONE LOGICA 2***

# Esercizio 9

## (array)

---

- Creare un programma che legga da input una sequenza di interi positivi. In particolare, l'utente inserisce un numero iniziale con cui specifica quanti numeri è intenzionato ad inserire (al max 10). Di seguito poi inserisce i numeri, tutti in ordine strettamente crescente.
- In fase di lettura, il programma controlli che ogni numero sia effettivamente maggiore del precedente (si scartino i valori che non rispettano tale criterio).
- In un secondo vettore si calcoli la differenza percentuale tra un valore ed il successivo ( data dalla differenza tra i due valori, divisa poi per il primo valore e moltiplicato il tutto per 100)
- Si richieda poi all'utente un valore di soglia (in percentuale) , e si stampino a video tutte le coppie di valori il cui aumento dal primo al secondo valore risulta essere, in percentuale, maggiore della soglia specificata

# Esercizio 10

(array)

---

- Si vogliono elaborare alcuni dati metereologici, riguardanti alcune località sciistiche. Per ogni località, un'utente inserisce il codice della località e i cm. di manto nevoso (entrambi interi). Il codice 0, inserito come località, indica che l'utente non vuole inserire altri dati.
- Il programma deve memorizzare tali dati in due appositi vettori (uno per le località ed uno per la neve caduta)
- Il programma deve poi stampare a video i codici di tutte le località che risultino avere un manto nevoso inferiore alla media, calcolata sui valori inseriti

# Esercizio 10 - Soluzione

## (array)

---

```
#include <stdio.h>
#include <stdlib.h>
#define DIM 10

int main(void) {
    int cod_loc, size, i, tot;
    int loc[DIM], neve[DIM];
    float media;

    size = 0;
    do {
        printf("Inserisci codice localita': ");
        scanf("%d", &cod_loc);
        if ( cod_loc != 0 && size < DIM ) {
            loc[size] = cod_loc;
            printf("Manto nevoso (cm.): ");
            scanf("%d", &(neve[size]));
            size++;
        }
    } while (cod_loc!=0 && size<DIM);
```

...