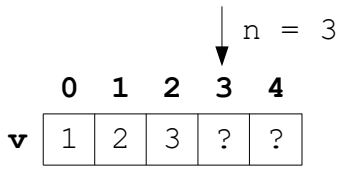
LABORATORIO FONDAMENTI DI INFORMATICA
22 OTTOBRE 2019 – Incontro 4 di 8 – Vettori e Matrici

Esercizio 1
Definire un vettore `v` di numeri interi che possa contenere al massimo `#define DIM0 5` numeri e un indice intero `n` che, inizialmente posto uguale a `0`, svolga la duplice funzione di indicare in un dato istante quale sia la prima cella disponibile per inserire un nuovo numero a partire dalla prima cella in poi e il conteggio di quanti siano i numeri presenti nel vettore.
Domandare all'utente i numeri da inserire uno per volta nel vettore fino a che l'utente inserisce il numero 0 oppure non vi sono più celle libere. Stampare infine a video gli `n` numeri inseriti nel vettore.



Vettore v in un istante in cui contiene n = 3 numeri.

```
Il vettore contiene n=0 numeri. Numero per cella 0? 1
…
Il vettore contiene n=4 numeri. Numero per cella 4? 5
Il vettore contiene n=5 numeri. Il vettore e' pieno.
Vettore con n=5 numeri: 1 2 3 4 5
```

Esercizio 2
Scrivere un programma che chieda all'utente `DIM1` = 5 numeri interi distinti da inserire in un primo vettore e quindi altri `DIM2` = 4 numeri interi distinti da inserire in un secondo vettore (`DIM1` e `DIM2` sono due costanti da definire con la direttiva `#define`).
Il programma deve stampare a video il contenuto di entrambi i vettori e poi trovare e stampare a video i numeri che sono presenti in entrambi i vettori (l'intersezione).
Si può supporre che sia l'utente a inserire numeri sempre distinti oppure si può implementare un controllo che richieda l'inserimento se un numero è già stato inserito.

```
vettore1: 1 4 5 6 7
vettore2: 4 5 9 1
intersezione: 1 4 5
```

Esercizio 3
Definita la costante `DIM` = 3, scrivere un programma che chieda all'utente `DIM*DIM` numeri interi e li inserisca in una matrice `DIM*DIM` rappresentata tramite un vettore bidimensionale. Dopo l'inserimento il programma deve moltiplicare per 2, cioè raddoppiare, ciascun numero presente nella matrice e ristampare a video la matrice "raddoppiata".

```
Matrice inserita:        Matrice raddoppiata:
 0  1  2                  0  2  4
 3  4  5                  6  8 10
 6  7  8                 12 14 16
```

Esercizio 4
Nel linguaggio C una stringa di testo viene rappresentata tramite un vettore di `char`. Per segnalare dove il testo finisce viene utilizzato il carattere terminatore `'\0'`. Quando si dimensiona il vettore, occorre sempre prevedere lo spazio anche per il carattere terminatore.
Scrivere un programma che chieda un nome all'utente di massimo 10 caratteri (escluso il terminatore) e lo inserisca in una stringa chiamata `nome` tramite la funzione `scanf` oppure la

funzione `gets`. Il programma deve poi calcolare la lunghezza in caratteri del nome, "scorrendo" il vettore cella per cella partendo dalla cella 0 fino a trovare quella che contiene il carattere terminatore `'\0'`. Stampare a video la lunghezza calcolata confrontandola con quella che viene restituita "in automatico" dalla funzione `strlen(nome)`.

NB:

- per utilizzare la funzione `strlen` occorre la direttiva `#include <string.h>`
- la funzione `scanf("%s", nome)` inserisce nella variabile `nome` il testo immesso dall'utente fino al primo spazio. Se si vogliono prevedere stringhe con spazi, si può utilizzare la funzione `gets(nome)`. Entrambe le funzioni aggiungono in automatico il carattere terminatore `'\0'`. Entrambe sono "non sicure" perché consentono all'utente di inserire testi di lunghezza superiore a quella supportata dal vettore.
- Per stampare con `printf` un testo contenente delle virgolette `"` o un backslash `\` occorre anteporgli il carattere di backslash: `printf("terminatore '\\0' \"fine\" ");`

0	1	2	3	4	5	6	7	8	9	10
J	o	h	n	\0						

```
Qual e' il tuo nome? John
Ciao John! La lunghezza del tuo nome e' di 4 caratteri.
Ok: la lunghezza calcolata cercando la posizione del carattere terminatore '\0' (4) coincide con lunghezza calcolata dalla funzione strlen (4).
```

Esercizio 5

Scrivere un programma che, definiti 3 vettori di numeri interi di dimensione massima MAX = 10, chieda all'utente quante celle `n` (con 1 <= `n` <= 10) voglia effettivamente usare. Il programma deve poi riempire i primi due vettori a partire dalla cella 0 con `n` numeri da 0 a 9 recuperati attraverso il generatore di numeri pseudo-casuali.

Nel terzo vettore il programma deve memorizzare per poi stampare a video la somma dei valori contenuti nelle celle "speculari" dei primi due vettori (la prima cella del primo vettore con l'ultima "significativa" del secondo, ecc.), nella stessa posizione utilizzata sul primo vettore.

Per utilizzare il generatore di numeri occorre includere le seguenti "librerie":

```
#include <stdlib.h>
#include <time.h>
```

Per generare numeri sempre diversi occorre inserire a inizio programma l'istruzione `srand`:

```
srand(time(NULL));
```

I numeri da 0 a 9 vengono generati tramite l'istruzione `rand()`:

```
vettore1[i] = rand() % 10;
```

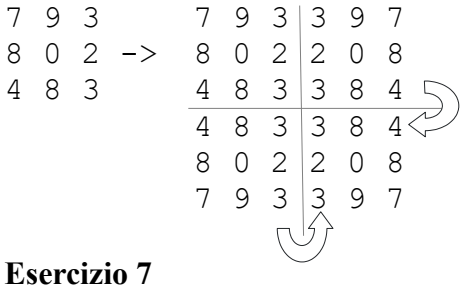
```
Lunghezza effettiva (1-10)? 4
vettore1:  7  9  3  8
vettore2:  0  2  4  8
vettore3: 15 13  5  8
```

Esercizio 6

Definita una costante DIM pari a 3, scrivere un programma che riempia una matrice DIM x DIM di numeri interi tramite il generatore di numeri pseudo-casuali (vedere l'esercizio precedente).

Tale matrice va replicata 4 volte in una seconda matrice (DIM x 2) x (DIM x 2), una volta per ciascun quadrante della seconda matrice. La replica deve essere però speculare (con ribaltamento

orizzontale e verticale rispetto alla mezzeria della matrice – vedere la figura sottostante).



Esercizio 7

Scrivere un programma che riempia un vettore di `NMAX` 100 numeri interi pseudo-casuali compresi tra 0 e 99 (utilizzare il generatore di numeri pseudo-casuali con `rand() % 100`).

Il programma deve stampare il contenuto di un secondo vettore di dieci elementi che deve contenere

- nella cella 0 quanti numeri del primo vettore sono compresi tra 0..9
- nella cella 1 quanti numeri del primo vettore sono compresi tra 10..19
- nella cella 2 quanti numeri del primo vettore sono compresi tra 20..29
- …
- nella cella 9 quanti numeri del primo vettore sono compresi tra 90..99

Suggerimento: si noti che il risultato (intero) di (X / 10) rappresenta la decina di appartenenza di un numero X. Ad esempio con X = 75 si ha che 75 / 10 = 7 quindi si può ottenere il numero della cella del secondo vettore dove aumentare il conteggio di quanti numeri sono compresi tra 70..79. Grazie a questo, si può evitare di scrivere 10 condizioni diverse, una per ogni decina.

```
(Risultato con NMAX = 10 invece che 100)
7 49 73 58 30 72 44 78 23 9
 0.. 9=2  10..19=0  20..29=1  30..39=1  40..49=2  50..59=1
60..69=0  70..79=3  80..89=0  90..99=0
```

Esercizio 8

Definire un tipo di dati strutturato denominato "`s_anagr`" per contenere i dati anagrafici di una persona, cioè il `nome` di massimo `NAMELEN` = 20 caratteri (+ 1 per il terminatore `'\0'`), il `sesso` `'M'` o `'F'` e l'età.

```
typedef struct {
     char nome[NAMELEN];
     char sesso;
     int eta;
} s_anagr;
```

Creare poi un vettore di `NANAGR` = 5 anagrafiche denominato "`v_anagr`"

```
s_anagr v_anagr[NANAGR] = {
     {"Topolino", 'M', 47}
     ,{"Minni", 'F', 37}
     ,{"Pluto", 'M', 17}
     ,{"Clarabella", 'F', 27}
     ,{"Pippo", 'M', 26}
};
```

Una volta popolato il vettore `v_anagr` occorre creare e stampare a video un nuovo vettore denominato "`v_finale`" di massimo `NANAGR` + 1 anagrafiche contenente:

- nella cella 0 la persona di sesso maschile più giovane contenuta in `v_anagr`
- nella cella 1 la persona di sesso femminile più giovane contenuta in `v_anagr`
- nelle celle dalla 2 in poi tutte le persone di sesso maschile 'M' con età superiore ai 25 anni contenute in `v_anagr`

Non serve con le persone fornite come esempio, ma in caso si voglia prevedere il caso che NON esista una persona più giovane da inserire nelle celle 0 e 1 (se ci sono solo maschi o solo femmine)

è possibile inserire al suo posto un'anagrafica "vuota" che abbia come nome "N/A" e come età -1.
Note sul linguaggio C:

- Per accedere al campo di una struttura si usa l'operatore punto " . "
  ```
  v_finale[i].eta
  ```
- Non funziona l'assegnamento diretto di una stringa
  ```
  v_anagr[i].nome = "Topolino"; //NO!
  ```
  occorre invece utilizzare la funzione strcpy(destinazione, sorgente)
  ```
  strcpy(v_anagr[i].nome, "Topolino");
  ```
- Se due variabili hanno la stessa struttura è possibile eseguire un assegnamento diretto per copiare i dati da una all'altra (altrimenti occorre copiare tutti i campi uno per uno)
  ```
  v_finale[i] = v_anagr[j];
  ```
- Non è possibile verificare direttamente se due variabili contengano gli stessi dati
  ```
  if (v_finale[0] == v_finale[2]) … //NO!
  ```
  ma occorre invece confrontare tutti i campi uno per uno
  ```
  if (v_finale[0].eta == v_finale[2].eta && v_finale[0].sesso
  == v_finale[2].sesso && strcmp(v_finale[0].nome,
  v_finale[2].nome) == 0) ...
  ```

```
Elenco anagrafiche:
"Topolino", 'M', 47
"Minni", 'F', 37
"Pluto", 'M', 17
"Clarabella", 'F', 27
"Pippo", 'M', 26
Cella 0: "Pluto", 'M', 17 (maschio piu' giovane)
Cella 1: "Clarabella", 'F', 27 (femmina piu' giovane)
Cella 2: "Topolino", 'M', 47 (maschio con piu' di 25 anni)
Cella 3: "Pippo", 'M', 26 (maschio con piu' di 25 anni)
```