Vettori, stringhe, matrici

Vettori

Un insieme di variabili ordinate

I vettori (array) sono variabili che al loro interno contengono più celle di un certo tipo.

Accedere alle celle

Le varie celle sono identificate dalla loro posizione all'interno del vettore

⚠ Attenzione, la prima cella si trova in posizione `0` (si dice che gli array in C sono zero-based). Non sarà lo stesso in MATLAB.
L'ultima posizione di un vettore è dunque quella in posizione **`n-1`**, dove **`n`** è la dimensione del vettore.

```c
int primes[5] = {2,3,5,7,11};
int i;

for(i=0;i<5;i++)
{
    printf("%d\n", primes[i]);
}
```

Esercizio 1

ES1: Scrivere un programma che chiede all'utente cinque numeri interi e che li stampa in ordine inverso.

Esercizio 1 - Soluzione

```c
int vett[5];
int i;

for(i=0;i<5;i++)
{
    printf("Inserisci il %d numero:", i);
    scanf("%d", &vett[i]);
}
for(i=4;i>=0;i--)
{
    printf("%d\n", vett[i]);
}
```

Esercizio 2

ES2: Scrivere un programma che chiede all'utente cinque numeri interi tra 0 e 100 e che ne stampa il massimo, il minimo e la media.

La directive **`#define`** permette di definire valori da usare in tutto il programma.

Questo è molto utile per sostituire valori *hard-coded* con dei simboli, e permette di non dover scorrere tutto il codice quando si cambia un parametro del nostro programma.

```c
#define ARRAY_SIZE 10

void main()
{
    int vett[ARRAY_SIZE];
    int i;

    for(i=0;i<ARRAY_SIZE;i++)
    {
        printf("Inserisci il %d numero:", i);
        scanf("%d", &vett[i]);
    }
}
```

L'operatore ternario **`?:`** permette di comprimere la sintassi di un **`if-else`**

```c
y = x ? a : b
```

⟷

```c
if(x)
{
    y = a;
}
else
{
    y = b;
}
```

Esercizio 2 - Soluzione

```c
int vett[ARRAY_SIZE];
int i;

for(i=0;i<ARRAY_SIZE;i++)
{
    do
    {
        printf("Inserisci il %d numero:", i);
        scanf("%d", &vett[i]);
    }while(vett[i]<0 || vett[i]>100);
}

int max=0, min=100, accumulator = 0;
float avg;

for(i=0;i<ARRAY_SIZE;i++)
{
    max = vett[i] > max ? vett[i] : max;
    min = vett[i] < min ? vett[i] : min;
    accumulator += vett[i];
}

printf("Il massimo è: %d\n", max);
printf("Il minimo è: %d\n", min);
printf("La media è: %f\n", (float)accumulator/ARRAY_SIZE);
```

Esercizio 3 (difficile)

ES3: Scrivere un programma che chiede all'utente cinque numeri interi positivi e che li stampa in ordine decrescente.

Esercizio 3 - Soluzione

1

```c
int vett[ARRAY_SIZE];
int i,j,max,posmax;

for(i=0;i<ARRAY_SIZE;i++)
{
    do
    {
        printf("Inserisci il %d numero:", i);
        scanf("%d", &vett[i]);
    }while(vett[i]<0);
}
```

2

```c
for(i=0;i<ARRAY_SIZE;i++)
{
    max = -1;
    for(j=i;j<ARRAY_SIZE;j++)
    {
        if(vett[j]>max)
        {
            max = vett[j];
            posmax = j;
        }
    }
    int temp = vett[i];
    vett[i] = max;
    vett[posmax] = temp;
}

for(i=0;i<ARRAY_SIZE;i++)
{
    printf("%d\n", vett[i]);
}
```

Stringhe (1)

Vettori di caratteri

Le stringhe in C sono definite come vettori di caratteri.

Ogni cella del vettore contiene un carattere della stringa.

```
char stringa[100];
```

c	i	a	o	!	\0

Terminatore

Poiché non possiamo sapere a priori quanto sia lunga una stringa, il carattere terminatore '`\0`' viene utilizzato per indicarne il termine.

⚠ Bisogna quindi fare attenzione che nel vettore ci sia abbastanza spazio per contenere anche questo carattere.

La stringa 'ciao' ha bisogno di un array di almeno 5 caratteri per poter essere salvata.

```
char stringa[5] = "ciao";
```

Esercizio 4

ES4: Scrivere un programma che data la stringa «stampami», salvata in un vettore di 100 caratteri, la stampi carattere per carattere.

```
void main()
{
    char stringa[100] = "stampami";

        ?

}
```

Esercizio 4 - Soluzione

```c
char stringa[100] = "stampami";
int i=0;
char temp = stringa[i];
while(temp != '\0')
{
    printf("%c\n", temp);
    i++;
    temp = stringa[i];
}
```

Esercizio 4 – Soluzione con strlen

ℹ️ La funzione **`strlen`** nella libreria **`string.h`** ritorna la lunghezza della stringa data.

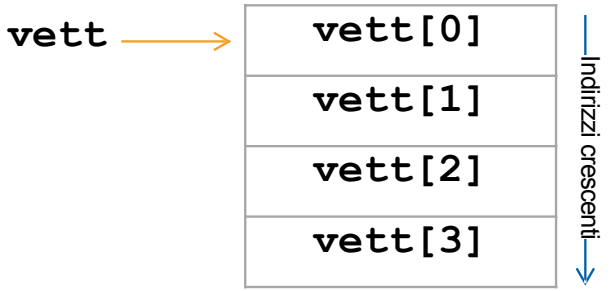
```c
#include <string.h>

                ⋮

char stringa[100] = "stampami";
int len = strlen(stringa);
int i;
for(i=0;i<len;i++)
{
    printf("%c\n", stringa[i]);
}
```

Stringhe (2)

I vettori in memoria

Tutte le celle costituenti un vettore sono salvate in regioni contigue in memoria, con l'identificatore del vettore (il nome della variabile vettore) che punta alla prima cella.

Noi non lo vedremo in dettaglio, è però utile saperlo in alcuni contesti.



Scanf e vettori

Quando usiamo la scanf, abbiamo visto che l'operatore **&** viene usato per ottenere l'indirizzo di memoria della variabile operando. Poiché i vettori sono già puntatori, **&** è omesso quando la variabile è di questo tipo.

```
char stringa1[100];
char stringa2[100];

scanf("%s", stringa1);
scanf("%s", &stringa2[0]);
```

Per le stringhe, omettere **&** è equivalente a indicare l'indirizzo della prima cella del vettore.

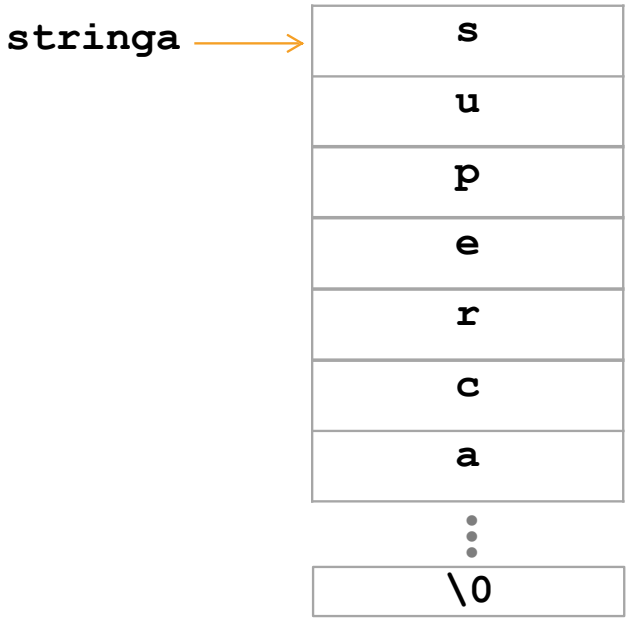
Esercizio 5

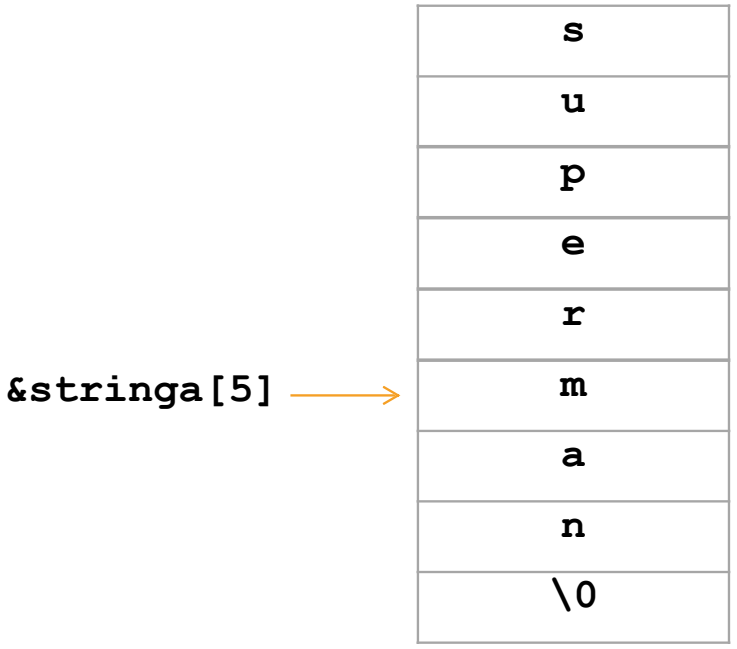
ES5: Cosa stampa questo programma?

```c
char stringa[100];

printf("Inserisci una stringa:");
scanf("%s", stringa); //VIENE INSERITO "supercalifragilistichespiralidoso"
printf("Inseriscine un'altra per buona misura:");
scanf("%s", &stringa[5]); //VIENE INSERITO "man"

printf("%s", stringa);
```

Esercizio 5 - Soluzione

```
Inserisci una stringa:supercalifragilistichespiralidoso
Inseriscine un'altra per buona misura:man
superman
```






```c
scanf("%s", stringa);
```

```c
scanf("%s", &stringa[5]);
```

Esercizio 5 - extra

ES5: E questa?

PROVATE VOI!!!

```
printf("%s", &stringa[9]);
```

Esercizio 6

ES6: Scrivere un programma che chiede ripetutamente un numero **`n`**. Se questo è pari, il programma stampa i primi **`n`** valori della sequenza di fibonacci. Se invece **`n`** è dispari o **`0`**, il programma si arresta.

FYI:

```
fib(i) = fib(i-1)+fib(i-2)

fib(0) = 0
fib(1) = 1
```

Cosa succede se **`n`** è negativo? Cambia qualcosa se è pari o dispari?

Esercizio 6 - Soluzione

```c
void main()
{
    int N;
    do
    {
        printf("Inserisci un numero intero N: ");
        scanf("%d", &N);

        if(N==0 || N%2 == 1)
        {
            printf("Fine\n");
        }
        else
        {
            int f0 = 0;
            int f1 = 1;
            int fN;
            int i;
            printf("%d\n%d\n", f0, f1);

            for(i=1;i<=N-2;i++)
            {
                fN = f0 + f1;
                printf("%d\n", fN);
                f0 = f1;
                f1 = fN;
            }
        }
    }
    while(N != 0 && N%2 == 0);
}
```

Matrici

Vettori di vettori

Le matrici, come i vettori, contengono un insieme di variabili dello stesso tipo. A differenza di questi, le celle nelle matrici sono identificate da due indici anziché uno solo.

```c
void main()
{
    int mat[5][5];

    printf("Inserisci in posizione 0,0: ");
    scanf("%d", &mat[0][0]);
}
```

Esercizio 7

ES1: Scrivere un programma che, data una matrice 8×8 rappresentante una scacchiera sulla quale sono presenti un Re ed una Regina, stampi «sì» se la Regina può mangiare il Re, «no» altrimenti.

SUGGERIMENTO: potete indicare il Re con un `1` e la Regina con un `2`, mentre lo `0` può indicare le caselle vuote.

Esercizio 7 - Soluzione

1

```
void main()
{
    int mat[8][8];
    fill_yes(&mat);

    //Find king's position
    int kx=-1,ky=-1;
    int i,j;
    for(i=0;i<8;i++)
    {
        for(j=0;j<8;j++)
        {
            if(mat[i][j]==1)
            {
                kx=j;
                ky=i;
            }
        }
    }

    if(kx<0 || ky<0)
    {
        printf("Non trovo il re.");
        return;
    }
```

2

```
//Check horizontal
for(j=0;j<8;j++)
{
    if(mat[ky][j]==2)
    {
        printf("Si'");
        return;
    }
}

//Check vertical
for(i=0;i<8;i++)
{
    if(mat[i][kx]==2)
    {
        printf("Si'");
        return;
    }
}
```

3

```
//Check diagonal 2/4
for(i=0;i<(ky-kx);i++)
{
    if(mat[(ky+kx)-i][i]==2)
    {
        printf("Si'");
        return;
    }
}

//Check diagonal 1/3
for(i=0;i<8-(ky-kx);i++)
{
    if(mat[(ky-kx)+i][i]==2)
    {
        printf("Si'");
        return;
    }
}

printf("No");
```

Esercizi aggiuntivi

Esercizio 8

ES8: Scrivere un programma che, data una stringa ed un numero intero, codifichi la stringa secondo il cifrario di Cesare.

TIP: fare uno shift sull'alfabeto di ogni lettera di tanti posti quanti indica il numero (la chiave).

Esercizio 9

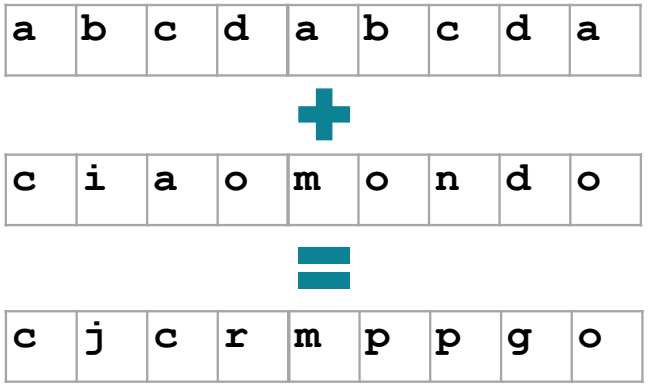
ES9: Scrivere un programma che, date due stringhe, codifichi la prima utilizzando il cifrario “snake cypher” con la seconda come chiave.

Chiave

a	b	c	d

Testo

c	i	a	o	m	o	n	d	o



Esercizio 10

ES10: Scrivere un programma che, dati i voti di n giudici per m tuffatori, ed i rispettivi coefficienti di difficoltà, calcoli i punteggi dei tuffatori.

Nelle gare di tuffi a 5 giudici, il punteggio finale e' pari al prodotto tra il coefficiente di difficoltà del tuffo e la somma dei voti ottenuta eliminando sia il voto più alto sia il voto più basso. Si scriva un programma che riceva in ingresso il coefficiente di difficoltà (valori da 1 a 4) e 5 voti da 1 a 10 e stampi ad output il punteggio ottenuto dal tuffatore, *con due numeri decimali.*

Ad esempio si considerino i seguenti voti (in rosso quelli eliminati):

~~8.0~~ 7.5 7.0 ~~6.5~~ 6.5

Definito come coefficiente di difficoltà 3.2, il voto finale sarà pari a:

$$(7.5 + 7.0 + 6.5) \cdot 3.4 = 67.20$$

Esercizio 11

ES11: Scrivere un programma che, dato un vettore di stringhe (una matrice di **`char`**), stampa a schermo quante delle stringhe nel vettore contengono una sottostringa determinata dall'utente.