

MATLAB

Alcune funzioni utili:

fprintf, length, size

Funzione fprintf

```
fprintf(fileID, format, var1, var2, ... )  
fprintf(format, var1, var2, ... )  
count = fprintf(...)
```

`fprintf(fileID, format, A, B)` salva il contenuto delle variabili A e B, per colonne (una dopo l'altra), secondo il formato *format* sul file specificato da *fileID*. La variabile A può essere sia uno scalare, sia un array, sia una stringa, come avviene quando si usa la funzione `disp()`.

`fprintf(format, A, B)` come sopra, ma visualizza il risultato sullo schermo

`count = fprintf(format, listavariabili)` assegna alla variabile `count` il numero di byte che sono stati stampati

Funzione fprintf

format

E' una stringa racchiusa fra apici che descrive come deve apparire il testo da stampare.

Le parti costanti del testo devono essere riportate così come devono apparire.

Se si deve stampare il contenuto di una variabile, si deve inserire una lettera corrispondente al tipo di dato preceduta da %.

Le principali sono: **%d** = intero, **%f** = reale,

%e = reale in formato esponenziale, **%s** = stringa

Possono essere presenti anche caratteri speciali (a capo **\n**, tabulazione **\t**, apici **' '**, ecc.)

Funzione fprintf: esempi

fprintf('Ciao! ') %% Stampa di una stringa

stampa su video

Ciao!

Qui si indica che il valore che sostituirà %d nell'output è il valore della variabile a

a=1

%% Stampa di una frase con variabili

fprintf('Contenuto della variabile a: %d', a);

stampa su video

Contenuto della variabile a: 1

Qui comparirà il valore di una variabile nell'output: %d specifica anche che sarà visualizzato come intero.

Funzione fprintf: esempi

```
a=[5 4 3 2 1];
```

```
fprintf('Valori: a(2)=%d, a(3)= %d', a(2), a(3));
```

stampa su video

```
Valori: a(2)=4, a(3)=3
```

```
a=[5 4 3 2 1];
```

```
fprintf('%d ', a);
```

stampa su video (un vettore viene stampato elemento per elemento replicando per ciascun elemento il formato indicato)

```
5 4 3 2 1
```

Funzione fprintf: esempi

```
a=[5 4 3 2 1; 2 3 4 5 6];
```

```
fprintf('%d ', a);
```

stampa su video (una matrice viene stampata per colonne replicando elemento per elemento il formato indicato)

```
5 2 4 3 3 4 2 5 1 6
```

(prima il contenuto della prima colonna, poi quello della seconda, ecc.)

Funzione fprintf

Principali caratteri speciali che possono apparire nella stringa di formato:

' '	apice singolo
%%	simbolo di percentuale
\\	backslash (barra inversa singola)
\b	backspace (cancella il carattere sulla sinistra)
\n	a capo
\t	tabulazione orizzontale

Funzione fprintf: esempi

```
fprintf('%d \t', a);
```

stampa su video (inserisce una tabulazione dopo ogni elemento)

```
5        4        3        2        1
```

```
p=15;
```

```
fprintf('Il %d%% della somma degli elementi di a è  
%f\n', p, sum(a)*p/100);
```

stampa su video

```
Il 15% della somma degli elementi di a è 2.25
```

Funzione fprintf: esempi

```
fprintf('Cesare disse: ' 'Il dado è tratto' ' \n');
```

stampa

```
Cesare disse: 'Il dado è tratto'
```

Funzioni `length` e `size`

La funzione **`length(v)`** fornisce come risultato la lunghezza (numero di elementi) del vettore v , sia che si tratti di un vettore riga che di un vettore colonna. Se v è uno scalare ha lunghezza 1.

La funzione **`size(a)`** fornisce come risultato il numero di righe e il numero di colonne di cui è costituita la matrice a .

NB Se a è uno scalare è costituito da 1 riga e 1 colonna. Se è un vettore riga con n elementi ha 1 riga e n colonne. Se è un vettore colonna con n elementi ha n righe e 1 colonna.

Funzioni length e size: esempi

Sono utili, ad esempio, quando si chiede all'utente di inserire dei dati da tastiera. L'utente potrebbe inserire uno scalare, un vettore, una matrice, ma il programma può saperlo solo 'interrogando' la variabile che contiene i dati inseriti dall'utente per conoscerne le dimensioni.

Ad esempio,

```
vet = input('inserisci un vettore') ;  
[r,c]=size(vet) ;  
while ((r~=1 && c~=1) || (c==1 && r==1) )  
    vet= input('Ti avevo detto un vettore: riprova!') ;  
    [r,c]=size(vet) ;  
end;
```

chiede di nuovo di inserire un vettore se non si è inserito un vettore ma, ad esempio, una matrice o uno scalare.

Funzioni length e size: esempi

```
vet = input('inserisci un vettore');  
[r,c]=size(vet);  
while ((r~=1 && c~=1) || (c==1 && r==1) )  
    vet= input('Ti avevo detto un vettore: riprova!');  
    [r,c]=size(vet);  
end;  
n=length(vet);  
for i=1:2:n  
    fprintf ('%d ', vet(i))  
end
```

Fa la stessa cosa poi stampa su video gli elementi che si trovano nelle posizioni dispari del vettore.

Funzioni length e size: esempi

```
a=[5 4 3 2 1];           %% a è un vettore riga
b=[2 4 5 7 9 2 5]';      %% b è un vettore colonna

fprintf('Il vettore a ha %d elementi ', length(a));
fprintf('mentre b ne ha %d.\n', length(b));
[nra,nca]=size(a); [nrb,ncb]=size(b);
fprintf('a ha dimensione %d x %d.\n', nra,nca);
fprintf('Invece b ha %d righe e %d colonne.', nrb,ncb);
```

stampa su video

Il vettore a ha 5 elementi mentre b ne ha 7.

a ha dimensione 1 x 5.

Invece b ha 7 righe e 1 colonne.

Estrazione di dati da una matrice

```
a = [1 2 3 4 5; 9 5 4 1 0; 10 1 7 2 1]
```

```
b = a(1,3)      % 3° elemento della prima riga
```

```
c = a(1,:)      % l'intera prima riga (c è un array  
                monodimensionale)
```

```
d = a(:,3)      % la terza colonna di a (d è un  
                vettore colonna)
```

Anche gli indici possono essere espressi tramite un array

```
1:n    % restituisce un array con tutti i numeri  
        interi da 1 a n
```

```
e = a(2, [[1:3] 5]) % vettore contenente i  
                    primi 3 elementi e il  
                    quinto della seconda riga  
                    di a
```

Esercizi

Scrivere un programma che legge un vettore riga da tastiera, lo visualizza come vettore colonna usando un ciclo for, ne calcola il minimo e il massimo, e poi visualizza separatamente, su due righe, i due risultati. Usare `fprintf` per produrre gli output.

Es. di esecuzione (in corsivo l'input da tastiera)

Inserisci un vettore riga: *[3 7 0]*

Hai inserito il seguente vettore:

3

7

0

Il valore massimo del vettore è 3

Il valore minimo del vettore è 0

Esercizi

Tenendo conto che le seguenti istruzioni scambiano il contenuto della variabile *a* con quello della variabile *b*

```
t = a;
```

a 1 5 b

```
a = b;
```

```
b = t;
```

t

Verificare che questo sia ciò che effettivamente avviene e poi scrivere un programma che legge un vettore da tastiera, ne inverte il contenuto e poi stampa il vettore così ottenuto (usare un ciclo for che scandisce il vettore e scambia prima il primo elemento con l'ultimo, poi il secondo con il penultimo e così via)

Es. di esecuzione (in corsivo l'input da tastiera)

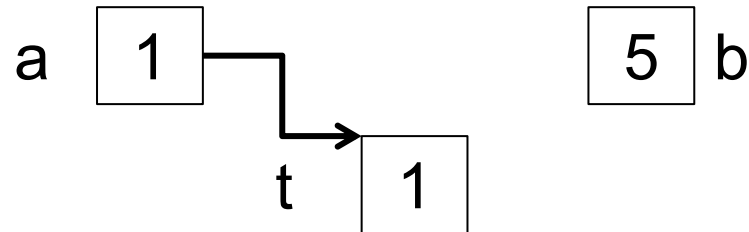
Inserisci un vettore: *[1 2 3 4 5 6 7]*

Il vettore invertito risulta 7 6 5 4 3 2 1

Esercizi

Tenendo conto che le seguenti istruzioni scambiano il contenuto della variabile *a* con quello della variabile *b*

```
t = a;  
a = b;  
b = t;
```



Verificare che questo sia ciò che effettivamente avviene e poi scrivere un programma che legge un vettore da tastiera, ne inverte il contenuto e poi stampa il vettore così ottenuto (usare un ciclo *for* che scandisce il vettore e scambia prima il primo elemento con l'ultimo, poi il secondo con il penultimo e così via)

Es. di esecuzione (in corsivo l'input da tastiera)

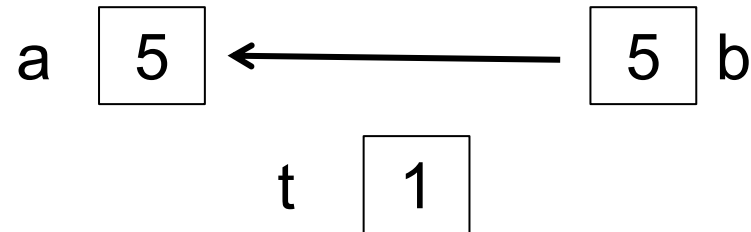
Inserisci un vettore: *[1 2 3 4 5 6 7]*

Il vettore invertito risulta 7 6 5 4 3 2 1

Esercizi

Tenendo conto che le seguenti istruzioni scambiano il contenuto della variabile *a* con quello della variabile *b*

```
t = a;  
a = b;  
b = t;
```



Verificare che questo sia ciò che effettivamente avviene e poi scrivere un programma che legge un vettore da tastiera, ne inverte il contenuto e poi stampa il vettore così ottenuto (usare un ciclo *for* che scandisce il vettore e scambia prima il primo elemento con l'ultimo, poi il secondo con il penultimo e così via)

Es. di esecuzione (in corsivo l'input da tastiera)

Inserisci un vettore: *[1 2 3 4 5 6 7]*

Il vettore invertito risulta 7 6 5 4 3 2 1

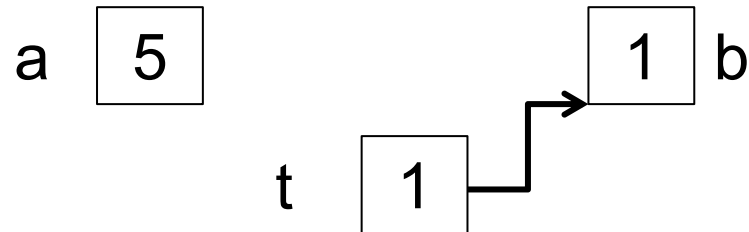
Esercizi

Tenendo conto che le seguenti istruzioni scambiano il contenuto della variabile *a* con quello della variabile *b*

```
t = a;
```

```
a = b;
```

```
b = t;
```



Verificare che questo sia ciò che effettivamente avviene e poi scrivere un programma che legge un vettore da tastiera, ne inverte il contenuto e poi stampa il vettore così ottenuto (usare un ciclo for che scandisce il vettore e scambia prima il primo elemento con l'ultimo, poi il secondo con il penultimo e così via)

Es. di esecuzione (in corsivo l'input da tastiera)

Inserisci un vettore: *[1 2 3 4 5 6 7]*

Il vettore invertito risulta 7 6 5 4 3 2 1