Università di Salerno
Dipartimento di Ingegneria Industriale

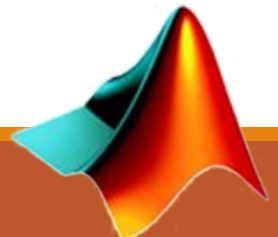
UNIVERSITÀ DEGLI STUDI DI SALERNO

Fondamenti di Informatica

Introduzione alla programmazione in MATLAB: Parte 2 – Possibili Soluzioni per gli Esercizi

Prof. Arcangelo Castiglione

A.A. 2016/17



MATLAB

Esercizi – 1/2
(Possibile Soluzione)

1) Supponendo che x = 6, calcolare a mano i risultati delle seguenti operazioni e controllarli con Matlab.

a) `z = (x < 10)`

b) `z = (x == 10)`

c) `z = (x >= 4)`

d) `z = (x ~= 7)`

```
(a) z = 1, (b) z = 0, (c) z = 1, (d) z = 1
```

Esercizi – 1/2
(Possibile Soluzione)

2) **Calcolare a mano i risultati delle seguenti operazioni e controllarli con Matlab.**

a) `z = 6 > 3 + 8`
b) `z = 6 + 3 > 8`
c) `z = 4 > (2 + 9)`
d) `z = (4 < 7) + 3`
e) `z = 4 < 7 + 3`
f) `z = (4 < 7) * 5`
g) `z = 4 < (7 * 5)`
h) `z = 2/5 >= 5`

```
(a) z = 0, (b) z = 1, (c) z = 0,
(d) z = 4, (e) z = 1, (f) z = 5,
(g) z = 1, (h) z = 0
```

Esercizi – 1/2
(Possibile Soluzione)

3) Supponendo che `x = [10, -2, 6, 5, -3]` e `y = [9, -3, 2, 5, -1]`, calcolare a mano i risultati delle seguenti operazioni e controllarli con Matlab.

a) `z = (x < 6)` b) `z = (x <= y)`

c) `z = (x == y)` d) `z = (x ~= y)`

```
(a) z = [0,1,0,1,1],   (b) z = [0,0,0,1,1],
(c) z = [0,0,0,1,0],   (d) z = [1,1,1,0,1]
```

Esercizi – 2/2
(Possibile Soluzione)

4) Dati i seguenti array x e y, utilizzare Matlab per trovare tutti gli elementi di x che sono maggiori dei corrispondenti elementi di y.

x = [-3, 0, 0, 2, 6, 8] y = [-5, -2, 0, 3, 4, 10]

```
>> x = [-3,0,0,2,6,8];
>> y = [-5,-2,0,3,4,10];
>> n = find(x>y)
n =
    1  2  5
```

Quindi, il primo, il secondo ed il quinto elemento di `x` sono maggiori dei corrispondenti elementi di `y`

Esercizi – 2/2
(Possibile Soluzione)

5) Il seguente array `prezzo` contiene i prezzi in lire di un determinato titolo azionario nel periodo di 10 giorni. Utilizzare Matlab per trovare il numero dei giorni in cui il prezzo è stato maggiore di 200 lire.

`prezzo = [190, 180, 220, 210, 250, 190, 170, 210, 270, 290]`

```
>>prezzo = [190,180,220,210,250,190,170,210,270,290];
>>length(find(prezzo>200))
ans =
    6
```

Esercizi – 2/2
(Possibile Soluzione)

6) I seguenti array `prezzo_A` e `prezzo_B` contengono i prezzi in lire di due titoli azionari nel periodo di 10 giorni. Utilizzare Matlab per trovare il numero dei giorni in cui il prezzo dell'azione A è stato maggiore di quello dell'azione B.

`prezzo_A = [190, 180, 220, 210, 250, 190, 170, 210, 270, 290]`

`prezzo_B = [220, 170, 200, 190, 240, 180, 160, 250, 280, 270]`

```
>>prezzo_A = [190,180,220,210,250,190,170,210,270,290];
>>prezzo_B = [220,170,200,190,240,180,160,250,280,270];
>>length(find(prezzo_A>prezzo_B))
ans =
    7
```

Esercizio 1

I seguenti array `prezzo_A`, `prezzo_B` e `prezzo_C` contengono i prezzi in lire di tre titoli azionari nel periodo di 10 giorni. Utilizzare Matlab per trovare:

a) il numero dei giorni in cui il prezzo dell'azione A è stato maggiore sia del prezzo di B sia del prezzo di C.

b) il numero dei giorni in cui il prezzo dell'azione A è stato maggiore del prezzo di B o del prezzo di C.

c) il numero dei giorni in cui il prezzo dell'azione A è stato maggiore del prezzo di B o del prezzo di C, ma non di entrambi.

```
prezzo_A = [190, 180, 220, 210, 250, 190, 170, 210, 270, 290]
prezzo_B = [220, 170, 200, 190, 240, 180, 160, 250, 280, 270]
prezzo_C = [170, 130, 220, 230, 190, 170, 200, 210, 240, 280]
```

Esercizio 1
(Possibile Soluzione)

a)

```
>>prezzo_A = [190,180,220,210,250,190,170,210,270,290];
>>prezzo_B = [220,170,200,190,240,180,160,250,280,270];
>>prezzo_C = [170,130,220,230,190,170,200,210,240,280];
>>length(find(prezzo_A>prezzo_B&prezzo_A>prezzo_C))
ans =
   4
```

b)

```
>>length(find(prezzo_A>prezzo_B|prezzo_A>prezzo_C))
ans =
   9
```

c)

```
>>length(find(xor(prezzo_A>prezzo_B,prezzo_A>prezzo_C)))
ans =
   5
```

Esercizio 2
(Possibile Soluzione)

Se x = [-3, 0, 0, 2, 5, 8] e y = [-5, -2, 0, 3, 4, 10], calcolare a mano i risultati delle seguenti operazioni e controllarli con Matlab.

a) `z = y < -x`

b) `z = x&y`

c) `z = x|y`

d) `z = xor(x,y)`

```
(a) z = [1,1,1,0,0,0], (b) z = [1,0,0,1,1,1]
(c) z = [1,1,0,1,1,1], (d) z = [0,1,0,0,0,0]
```

Esercizio 3

Siano e1 ed e2 sono due espressioni logiche. Le leggi di DeMorgan sulle espressioni logiche stabiliscono che:

NOT(e1 AND e2) implica (NOT e1) OR (NOT e2)

e

NOT(e1 OR e2) implica (NOT e1) AND (NOT e2)

Utilizzare queste leggi per trovare un'espressione equivalente per ciascuna delle seguenti espressioni; verificare l'equivalenza con Matlab.

a) `~((x < 10) & (x >= 6))`

b) `~((x == 2) | (x > 5))`

Esercizio 3
(Possibile Soluzione)

```
%(a)
A = (~((x<10)&(x>=6)))==((~(x<10))|(~(x>=6)))
%(b)
B = (~((x==2)|(x>5)))==((~(x==2))&(~(x>5)))
```

M-File Script

Eseguendo questo file per vari valori di x, otterremo che A = 1 e B = 1.
Ciò dimostra che le due identità sono vere.

Esercizio 4

Le seguenti espressioni sono equivalenti? Per verificare le risposte, utilizzare Matlab assegnando specifici valori alle variabili *a*, *b*, *c* e *d*.

a) 1. `(a==b) & ((b==c) | (a==c))`
   2. `(a==b) | ((b==c) & (a==c))`

b) 1. `(a<b) & ((a>c) | (a>d))`
   2. `(a<b) & (a>c) | ((a<c)) & (a>d))`

Esercizio 4
(Possibile Soluzione)

- Il seguente file di script restituisce `A = 0`, che indica falso. Quindi, le due espressioni non sono equivalenti

```
a = 1; b = 1; c = 3; d = 4;
A = ((a==b)&((b==c)|(a==c)))==((a==b)|((b==c)&(a==c)))
```

- Il seguente file script restituisce `B = 1`, che indica vero. Quindi, le due espressioni sono equivalenti

```
B = ((a<b)&((a>c)|(a>d)))==((a<b)&(a>c)|((a<b)&(a>d)))
```

Esercizio 5
(Possibile Soluzione)

Il prezzo in dollari di un determinato titolo azionario nel periodo di 10 giorni è dato del seguente array:

```
prezzo = [19, 18, 22, 21, 25, 19, 17, 21, 27, 29]
```

Un investitore possiede 1000 azioni all'inizio del periodo di 10 giorni e vuole comprare 100 azioni ogni giorno in cui il prezzo scende sotto i 20 dollari e vendere 100 azioni quando il prezzo supera i 25 dollari. Utilizzare Matlab per calcolare:

a) La spesa totale per acquistare le azioni;

b) L'importo totale derivante dalla vendita delle azioni;

c) Il numero totale di azioni che possiede l'investitore dopo 10 giorni.

d) L'incremento netto del valore del portafoglio azionario.

Per ottenere l'incremento netto del portafoglio azionario è necessario effettuare la **sottrazione** tra

- Prezzo del titolo azionario, al giorno di vendita (giorno 10), moltiplicato per il numero di titoli posseduti (calcolato al punto **c)**)
- Prezzo del titolo azionario, al giorno d'acquisto (giorno 1), moltiplicato per il numero di titoli posseduti (1000)

Esercizio 5
(Possibile Soluzione)

```
prezzo = [19,18,22,21,25,19,17,21,27,29];
costo_nuove_azioni = 100*sum(prezzo.*(prezzo<20))
importo_vendita = 100*sum(prezzo.*(prezzo>25))
variazione_azioni =100*(sum(prezzo<20)-sum(prezzo>25));
azioni_totali = 1000 + variazione_azioni
incremento_netto = prezzo(10)* azioni_totali - prezzo(1)*1000
```

- `costo_nuove_azioni = 7300`, `importo_vendita = 5600`, `azioni_totali = 1200` **ed** `incremento_netto = 15800`
- Quindi
  - La spesa totale per acquistare le azioni è stata di 7300$
  - L'importo totale derivante dalla vendita delle azioni è stato di 5600$
  - L'investitore dopo 10 giorni possiede 1200 azioni
  - L'incremento netto del valore del portafoglio azionario è stato di 15800$

Esercizio 6

(Possibile Soluzione)

L'altezza e la velocità di un oggetto che viene lanciato con una velocità iniziale v_0 e un angolo A (rispetto al piano orizzontale) sono date dalle seguenti formule:

$$h(t) = v_0 t \sin A - 0{,}5 g t^2$$

$$v(t) = \sqrt{v_0^2 - 2 v_0 g t \sin A + g^2 t^2}$$

Il termine g rappresenta l'accelerazione di gravità. L'oggetto cadrà al suolo quando $h(t) = 0$, nell'istante $t_{suolo} = 2(v_0/g)\sin A$. Siano dati i seguenti valori: $A = 30°$, $v_0 = 40$ m/sec e $g = 9{,}81$ m/sec^2. Utilizzare gli operatori logici e relazionali di Matlab per calcolare gli istanti in cui:

a) L'altezza non è minore di 15 m.

b) L'altezza non è minore di 15 m e contemporaneamente la velocità non è maggiore di 36 m/sec.

c) L'altezza è minore di 5 m o la velocità è maggiore di 35 m/sec.

Esercizio 6
(Possibile Soluzione)

M-File Script

```
v0 = 40; g = 9.81; A = 30*pi/180;
t_suolo = 2*v0*sin(A)/g;
t = [0:t_suolo/100:t_suolo];
h = v0*t*sin(A)-0.5*g*t.^2;
v = sqrt(v0^2-2*v0*g*sin(A)*t+g^2*t.^2);

%(a)
ua = find(h>=15);
t1a = (ua(1)-1)*(t_suolo/100)
t2a = ua(length(ua)-1)*(t_suolo/100)
%
%(b)
ub = find(h>=15&v<=36);
t1b = (ub(1)-1)*(t_suolo/100)
t2b = ub(length(ub)-1)*(t_suolo/100)
%
%(c)
uc = find(~(h<5|v>35));
t1c = (uc(1)-1)*(t_suolo/100)
t2c = uc(length(uc)-1)*(t_suolo/100)
```

Esercizio 6
(Possibile Soluzione)

- Quando eseguito, lo script produce i seguenti risultati
  - `t1a = 1.0194, t2a = 3.0581, t1b = 1.0601, t2b = 3.0173, t1c = 1.5494, t2c = 2.5280`
- Quindi
  a) L'altezza è di non meno di 15 metri per $1.0194 \leq t \leq 3.0581$ secondi
  b) L'altezza non è inferiore a 15 metri e la velocità è contemporaneamente non superiore a 36 m/sec per $1.0601 \leq t \leq 3.0173$ secondi
  c) L'altezza è inferiore a 5 metri o la velocità è superiore a 35 m/sec per $1.5494 \leq t \leq 2.5280$ secondi

Esercizio 7
(Possibile Soluzione)

Riscrivere le seguenti espressioni in modo da utilizzare una sola istruzione `if`.

```
if x < y
   if z < 10
      w = x*y*z
   end
end
```

```
if (x<y)&(z<10)
    w = x*y*z
end
```

Esercizio 8
(Possibile Soluzione)

- Scrivere una funzione, chiamata `maggiore3` (da memorizzare in un M-File Function), che prende in input i seguenti parametri numerici: `numero1`, `numero2` e `numero3`, e restituisce in output il maggiore di tali numeri

Esercizio 8
(Possibili Soluzioni)

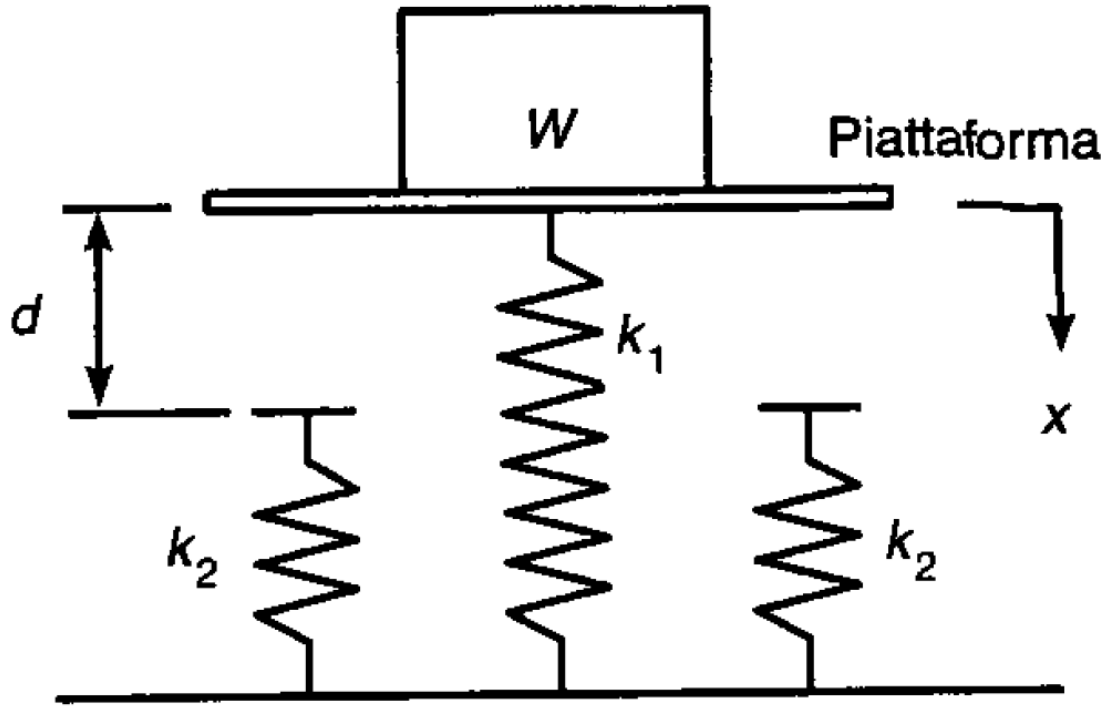
- Soluzione 1

```
function [ maggiore ] = maggiore3(numero1, numero2, numero3)
    if numero1 > numero2
        maggiore_temporaneo = numero1;
    else
        maggiore_temporaneo = numero2;
    end

    if numero3 > maggiore_temporaneo
        maggiore_temporaneo = numero3;
    end

    maggiore = maggiore_temporaneo;
end
```

- Soluzione 2 (Utilizzando funzioni built-in di MATLAB)

```
function [ maggiore ] = maggiore3(numero1, numero2, numero3)
     maggiore = max([numero1 numero2 numero3]);
end
```

Esercizio 9 – 1/3
(Possibile Soluzione)



Esercizio 9 – 2/3
(Possibile Soluzione)

La Figura 4.12 a) illustra un modello massa-molla del tipo utilizzato per progettare le sospensioni dei veicoli. Le molle esercitano una forza che è proporzionale alla loro compressione; il fattore di proporzionalità è la costante elastica k della molla. Le due molle laterali servono a fornire una resistenza aggiuntiva quando il peso W sollecita troppo la molla centrale. Se il peso viene appoggiato sulla piattaforma, il sistema si sposta a una distanza x prima di fermarsi. Affinché il sistema sia in equilibrio statico, la forza peso deve bilanciare le forze delle molle in questa nuova posizione, cioè:

$W = k_1 x$ se $x < d$

$W = k_1 x + 2k_2(x - d)$ se $x \geq d$

Esercizio 9 – 3/3
(Possibile Soluzione)

a) Creare un file di funzione che calcola la distanza x, utilizzando i parametri di input W, k_1, k_2 e d. Provare la funzione per i seguenti due casi, utilizzando i valori $k_1 = 10^4$ N/m; $k_2 = 1{,}5\times10^4$ N/m; $d = 0{,}1$ m.

$W = 500$ newton

$W = 2000$ newton

Esercizio 9
(Possibile Soluzione)

```
function distanza(W,k1,k2,d);
    if W < k1*d
        x = W/k1;
    else
        x = (W+2*k2*d)/(k1+2*k2);
    end
    disp('La distanza percorsa è: ')
    disp(x)
end
```

```
>>distanza(500,10000,15000,.1)
La distanza percorsa è:
    0.0500

>>distanza(2000,10000,15000,.1)
La distanza percorsa è:
    0.1250
```

Esercizio 10 – 1/2
(Possibile Soluzione)

Analizzare il sistema massa-molla descritto nel precedente Problema nel caso in cui il peso W viene lasciato cadere sulla piattaforma attaccata alla molla centrale. Se il peso cade da un'altezza h rispetto alla piattaforma, è possibile calcolare la compressione massima della molla x eguagliando l'energia potenziale di gravità Wh con l'energia potenziale immagazzinata nelle molle:

$$Wh = \frac{1}{2}k_1x^2 \qquad \text{se } x < d$$

Questa equazione può essere risolta in funzione di x:

$$x = \sqrt{\frac{2Wh}{k_1}} \qquad \text{se } x < d$$

e

$$Wh = \frac{1}{2}k_1x^2 + \frac{1}{2}(2k_2)(x-d)^2 \qquad \text{se } x \geq d$$

dalla quale è possibile ottenere la seguente equazione di secondo grado in x:

$$(k_1 + 2k_2)x^2 - 4k_2dx + 2k_2d_2 - 2Wh = 0 \qquad \text{se } x \geq d$$

Esercizio 10 – 2/2
(Possibile Soluzione)

Creare un file di funzione che calcola la compressione massima x dovuta al peso che cade da un'altezza h. I parametri di input della funzione sono k_1, k_2, d, W e h. Provare la funzione per i seguenti due casi, utilizzando i valori $k_1 = 10^4$ N/m; $k_2 = 1{,}5\times10^4$ N/m e $d = 0{,}1$ m.

$W = 100$ N, $h = 0{,}5$ m

$W = 2000$ N, $h = 0{,}5$ m

Esercizio 10 – 2/2
(Possibile Soluzione)

```
function x = spostamento(k1,k2,d,W,h)
    x1 = sqrt(2*W*h/k1);
    if x1 < d
        x = x1;
    else
        p = [k1+2*k2,-4*k2*d,2*k2*d^2-2*W*h];
        x = max(roots(p));
    end
end
```

```
>> spostamento(10000,15000,.1,100,.5)
ans =
    0.1000

>> spostamento(10000,15000,.1,2000,.5)
ans =
    0.2944
```

Esercizio 11
(Possibile Soluzione)

- Scrivere una funzione, chiamata `percentuale_sconto` (da memorizzare in un M-File Function), che prende in input l'`importo` dell'acquisto e restituisce in output la percentuale di sconto su tale importo
- Le percentuali di sconto devono essere così calcolate
  - Lo sconto verrà applicato se e solo se l'importo dell'acquisto è superiore a 299€
  - Se l'importo è superiore a 999€, la percentuale di sconto sarà del 5%, mentre, se tale importo è superiore a 1499€, allora la percentuale di sconto sarà dal 10%
  - La percentuale minima di sconto è 2%

Esercizio 11
(Possibile Soluzione)

```
function [perc_sconto] = percentuale_sconto(importo)
    if importo >= 299
        if importo >= 1499
            perc_sconto = 10;
        elseif importo >= 999
            perc_sconto = 5;
        else
            perc_sconto = 2;
        end
    else
        perc_sconto = 0;
end
```

Esercizio 12
(Possibile Soluzione)

- Scrivere una funzione, chiamata `stagione_anno` (da memorizzare in un M-file Function), che prende in input i seguenti parametri: `giorno` e `mese`, e restituisce in output **il codice della stagione** in cui tale data è collocata

- *Promemoria Stagioni*
  - *Inverno (Inizio: 23/12 - Fine: 20/03)* → codice `1`
  - *Primavera (Inizio: 21/03 – Fine: 21/06)* → codice `2`
  - *Estate (Inizio: 22/06 - Fine: 22/09)* → codice `3`
  - *Autunno (Inizio: 23/09 - Fine: 22/12)* → codice `4`
  - *Data non valida* → codice `-1`

Esercizio 12

(Possibile Soluzione)

```
function [ codice_stagione ] = stagione_anno(giorno, mese)
    if giorno >= 23 && mese == 12 % Inverno - parte 1
        codice_stagione = 1;
    elseif giorno >= 1 && (mese == 1 || mese == 2) % Inverno - parte 2
        codice_stagione = 1;
    elseif giorno <= 20 && mese == 3 % Inverno - parte 3
        codice_stagione = 1;
    elseif giorno >= 21 && mese == 3 % Primavera - parte 1
        codice_stagione = 2;
    elseif giorno >= 1 && (mese == 4 || mese == 5) % Primavera - parte 2
        codice_stagione = 2;
    elseif giorno <= 21 && mese == 6 % Primavera - parte 3
        codice_stagione = 2;
    elseif giorno >= 22 && mese == 6 % Estate - parte 1
        codice_stagione = 3;
    elseif giorno >= 1 && (mese == 7 || mese == 8) % Estate - parte 2
        codice_stagione = 3;
    elseif giorno <= 1 && mese == 9 % Estate - parte 3
        codice_stagione = 3;
    elseif giorno >= 23 && mese == 9 % Autunno - parte 1
        codice_stagione = 4;
    elseif giorno >= 1 && (mese == 10 || mese == 11) % Autunno - parte 2
        codice_stagione = 4;
    elseif giorno <= 22 && mese == 12 % Estate - parte 3
        codice_stagione = 4;
    else
        codice_stagione = -1;
    end
end
```