Università di Salerno
Dipartimento di Ingegneria Industriale

UNIVERSITÀ DEGLI STUDI DI SALERNO

ESERCIZI

Fondamenti di Informatica

Introduzione alla programmazione in MATLAB: Parte 3 – Possibili Soluzioni per gli Esercizi

Prof. Arcangelo Castiglione

A.A. 2016/17

MATLAB

Esercizio 1
(Possibile Soluzione)

- Scrivere un M-File Script MATLAB che generi il seguente output

```
     1     2     3     4     5     6     7     8     9    10

     1     2     3     4     5     6     7     8     9

     1     2     3     4     5     6     7     8

     1     2     3     4     5     6     7

     1     2     3     4     5     6

     1     2     3     4     5

     1     2     3     4

     1     2     3

     1     2

     1
```

```
i = 10;
while i > 0
    disp(1 : i);
    i = i - 1;
end
```

Esercizio 2
(Possibile Soluzione)

Esaminate il seguente file script. Riempite le righe della tabella successiva con i valori che sarebbero visualizzati subito dopo l'istruzione `while`, se questo file fosse eseguito. Scrivete i valori che assumono le variabili ogni volta che viene eseguita l'istruzione `while`. Poi, eseguite il file script e controllate i valori ottenuti con quelli che avete inserito nella tabella.

```
k = 1; b = -2; x = -1; y = -2;
while k <= 3
   k, b, x, y
   y = x^2 - 3;
   if y < b
      b = y;
   end
   x = x + 1;
   k = k + 1;
end
```

Passo I: 1 -2 -1 -2
Passo II: 2 -2 0 -2
Passo III: 3 -3 1 -3
Passo IV:
Passo V:

Passaggio	k	b	x	y
primo				
secondo				
terzo				
quarto				
quinto				

Esercizio 3

Il seguente prospetto fornisce i valori approssimati dei coefficienti di attrito statico μ per diversi materiali. Per mettere in movimento un peso W su una superficie orizzontale, occorre spingerlo con una forza F, dove $F = \mu W$. Scrivere un programma di Matlab utilizzando la struttura `switch` per calcolare la forza F. Il programma dovrà accettare come input il valore di W e i tipi di materiali.

Materiali	μ
Metallo su metallo	0,20
Legno su legno	0,35
Metallo su legno	0,40
Gomma su asfalto	0,70

Esercizio 3
(Possibile Soluzione)

```
W = input('Inserisci il peso (> 0): ');
materiali = input('Inserisci i materiali:' ,'s');
switch materiali
    case 'metallo su metallo'
        F = .2*W
    case 'legno su legno'
        F = .35*W
    case 'metallo su legno'
        F = .4*W
    case 'gomma su asfalto'
        F = .7*W
    otherwise
        disp('Scelta dei materiali non corretta!')
end
```

Esercizio 4

- Simulare una sequenza (**infinita**) di lanci di un dado a 6 facce
  - Se viene visualizzato 2 volte consecutive il numero 6, la simulazione si arresta
  - **Esempio di esecuzione:** 3 4 2 5 4 2 2 5 1 1 6 6

Esercizio 4
(Possibile Soluzione)

```
prec = -1;

while 1

    dado = ceil(6 * rand(1));
    disp(dado);

    if prec == 6 && dado == 6
        break;
    end
    prec = dado;

end
```

Esercizio 5
(Possibile Soluzione)

Riscrivere il seguente codice utilizzando un ciclo `while` per evitare di utilizzare il comando `break`.

```
for k = 1:10
   x = 50 - k^2;
   if x < 0
      break
   end
   y = sqrt(x)
end
```

```
x = 49;
k = 1;
while x>0
      y = sqrt(x)
      k = k+1;
      x = 50 - k^2;
end
```

Esercizio 6

- Scrivere una funzione `matrice_simmetrica` che prenda in input una matrice **A (quadrata)** e verifichi se essa è **simmetrica** o meno

- **NOTA:**
  - Una matrice si dice simmetrica se per ogni elemento **i, j** vale la seguente relazione **A(i, j)** = **A(j, i)**

$$\begin{bmatrix} 1 & 4 & 6 \\ 4 & 2 & 5 \\ 6 & 5 & 3 \end{bmatrix}$$

Esercizio 6
(Possibile Soluzione)

```
function [simm] = matrice_simmetrica(A)
    [nr, nc] = size(A);
    simm = 1;

    for i = 1:nr
        for j = i+1:nc
            if A(i, j) ~= A(j, i)
                simm = 0;
            end
        end
    end
end
```

Esercizio 7

- Scrivere una funzione `ricerca_elemento` che prenda in input una matrice **A** ed un intero **N** ed identifichi il numero di occorrenze di **N** in **A**

- **Esempio:** `ricerca_elemento(A, 5)` → `restituisce 2`

$$A = \begin{bmatrix} 1 & 4 & 6 \\ 4 & 2 & 5 \\ 6 & 5 & 3 \end{bmatrix}$$

Esercizio 7
(Possibile Soluzione)

```
function [occorrenze] = ricerca_elemento(A, N)
    [nr, nc] = size(A);
    occorrenze = 0;

    for i = 1:nr
        for j = 1:nc
            if A(i, j) == N
                occorrenze = occorrenze + 1;
            end
        end
    end
end
```

Esercizio 8

- Scrivere una funzione `sopra_media` che prenda in input una matrice **A**, ne calcoli la media e restituisca una matrice **B**, dove ogni elemento rispetta la seguente caratteristica

$$B(i,j) = \begin{cases} 1 & se\ A(i,j) \geq media_A \\ 0 & altrimenti \end{cases}$$

- **Esempio:** `ricerca_elemento(A)` → **output**

$$A = \begin{bmatrix} 1 & 4 & 6 \\ 4 & 2 & 5 \\ 6 & 5 & 3 \end{bmatrix} \qquad \mathrm{B} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \ (media_A = 4)$$

Esercizio 8
(Possibile Soluzione)

```
function [B] = sopra_media(A)
    % inizio calcolo media --- potrei sviluppare ed invocare
    % anche un'altra funzione, che calcoli la media di A
    [nr, nc] = size(A);
    media = 0;

    for r = 1:nr
        for c = 1:nc
            media = media + A(r,c);
        end
    end

    media = media / (nr * nc);
    % fine calcolo media

    for r = 1:nr
        for c = 1:nc
            if A(r,c) >= media
                B(r,c) = 1;
            else
                B(r,c) = 0;
            end
        end
    end
end
```

Esercizio 9 – 1/3

- **Come svolgere questo esercizio**
  1. **Analizzare ed approfondire lo svolgimento dell'Esempio 3**
     - http://www.di.unisa.it/~arccas/materiale/lezioni/Lezione_14.pdf (Da pag. 29 a pag. 51)
  2. **Definire le funzioni di seguito richieste**

Esercizio 9 – 2/3

Definizione di Funzioni

- Tenendo conto dello svolgimento dell'Esempio 3

a) Scrivere una funzione che
   - **Prende in input**
     - La matrice `m`, che rappresenta i costi ed i tempi di fabbricazione (Tabella 2.4)
     - Uno scalare `i`
   - **Restituisce in output**
     - Il costo di ogni processo per produrre una unità del prodotto `i`

NOTA: Le funzioni di tali esercizi possono invocare ulteriori funzioni sia viste a lezione e sia contenute negli esercizi precedenti oppure funzioni built-in o altre funzioni da voi definite

Esercizio 9 – 3/3
Definizione di Funzioni

- Tenendo conto dello svolgimento dell'Esempio 3

b) Scrivere una funzione che
  - **Prende in input**
    - La matrice `m`, che rappresenta i costi ed i tempi di fabbricazione (Tabella 2.4)
  - **Restituisce in output**
    - Un array contenente il costo (totale) per produrre una unità di ogni prodotto

NOTA: Le funzioni di tali esercizi possono invocare ulteriori funzioni sia viste a lezione e sia contenute negli esercizi precedenti oppure funzioni built-in o altre funzioni da voi definite

Esercizio 10 – 1/2

Un'azienda produce e vende carrelli da golf. Alla fine di ogni settimana, l'azienda trasporta i carrelli prodotti nel magazzino. Tutti i carrelli venduti sono prelevati dal magazzino. Ecco un semplice modello di questo processo:

$$I(k + 1) = P(k) + I(k) - S(k)$$

dove

$P(k)$ = numero di carrelli prodotti nella settimana k

$I(k)$ = numero di carrelli a magazzino nella settimana k

$S(k)$ = numero di carrelli venduti nella settimana k

Le vendite settimanali previste per 10 settimane sono elencate in questo prospetto:

Settimana	1	2	3	4	5	6	7	8	9	10
Vendite	50	55	60	70	70	75	80	80	90	55

Esercizio 10 – 2/2

La produzione di una settimana si basa sulle vendite della settimana precedente, quindi $P(k) = S(k-1)$. Si supponga che la produzione della prima settimana sia pari a 50 carrelli, cioè $P(1) = 50$. Scrivere un programma per calcolare e rappresentare in un diagramma il numero di carrelli a magazzino per ciascuna delle 10 settimane o finché il numero di unità a magazzino non sia minore di zero. Provare il programma per due casi: a) le scorte iniziali sono pari a 50 carrelli, cioè $I(1) = 50$; b) le scorte iniziali sono pari a 30 carrelli, cioè $I(1) = 30$.

c) Ripetere l'esercizio con il vincolo che la produzione della settimana successiva deve essere nulla se le scorte di carrelli superano 40 unità

Esercizio 10
(Possibile Soluzione)

a)

```
S = [50,55,60,70,70,75,80,80,90,55];
I(1) = 50;
P(1) = 50;
for k = 1:9
    I(k+1) = I(k)+P(k)-S(k);
    P(k+1) = S(k);
end
I
```

c)

```
S = [50,55,60,70,70,75,80,80,90,55];
I(1) = 50;
P(1) = 50;
k = 0;
while I>= 0
    k = k+1;
    I(k+1) = I(k)+P(k)-S(k);
    if I(k+1)>40
        P(k+1) = 0;
    else
        P(k+1) = S(k);
    end
end
disp(I)
```

Per quanto riguarda il punto **b)**, basta impostare `I(1) = 30` nel programma definito al punto **a)**

Esercizio 11 – 1/2

Gli ingegneri spesso devono calcolare la pressione e il volume di un gas all'interno di un contenitore. Per questi calcoli, di solito, viene utilizzata l'equazione di Van der Waals:

$$P = \frac{RT}{V-b} - \frac{a}{V^2}$$

Dove il termine b è una correzione per il volume delle molecole e il termine a/V^2 è una correzione per l'attrazione delle molecole. Il termine R è la costante dei gas perfetti, T è la temperatura assoluta e V è il volume specifico del gas. Il valore di R è uguale per tutti i gas e vale 0,082054 litri atm/mole °K. I valori di a e b dipendono dal tipo di gas.

Esercizio 11 – 2/2

Alcuni valori sono riportati nel seguente prospetto. Scrivere una funzione, utilizzando la struttura `switch`, per calcolare la pressione P dall'equazione di Van der Waals. Gli argomenti di input della funzione sono T e V e una variabile di stringa che contiene il nome di un gas elencato nel precedente prospetto. Provare la funzione per il gas Cl_2 alla temperatura T = 300 °K e un volume V = 20 litri/mole.

Gas	***a*** **(litri^2 atm/mole^2)**	***b*** **(litri/mole)**
Elio (He)	0,0341	0,0237
Idrogeno (H_2)	0,244	0,0266
Ossigeno (O_2)	1,36	0,0318
Cloro (Cl_2)	6,49	0,0562
Anidride carbonica (CO_2)	3,59	0,0427

Esercizio 11
(Possibile Soluzione)

```
function P = waals(T,V,gas)
    R = 0.08206;
    go = 1;
    switch gas
        case 'elio'
            a = 0.0341;b = 0.0237;
        case 'idrogeno'
            a = 0.244;b = 0.0266;
        case 'ossigeno'
            a = 1.36;b = 0.0318;
        case 'cloro'
            a = 6.49;b = 0.0562;
        case 'anidride carbonica'
            a = 3.59;b = 0.0427;
        otherwise
            disp('Errore: La tabella non contiene questo gas.')
            disp('Tutti i nomi dei gas devono essere in minuscolo.')
            go = 0;
    end
    if go == 1
        P = R*T/(V-b)-a/V^2;
    end
end
```

Esercizio 12 – 1/3

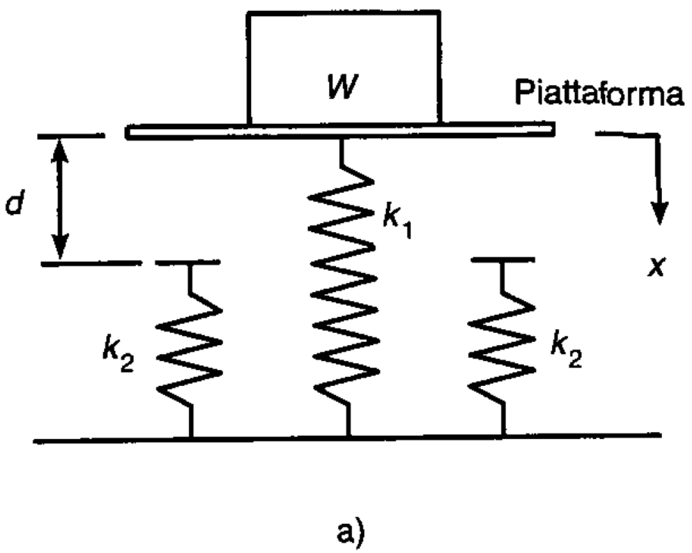
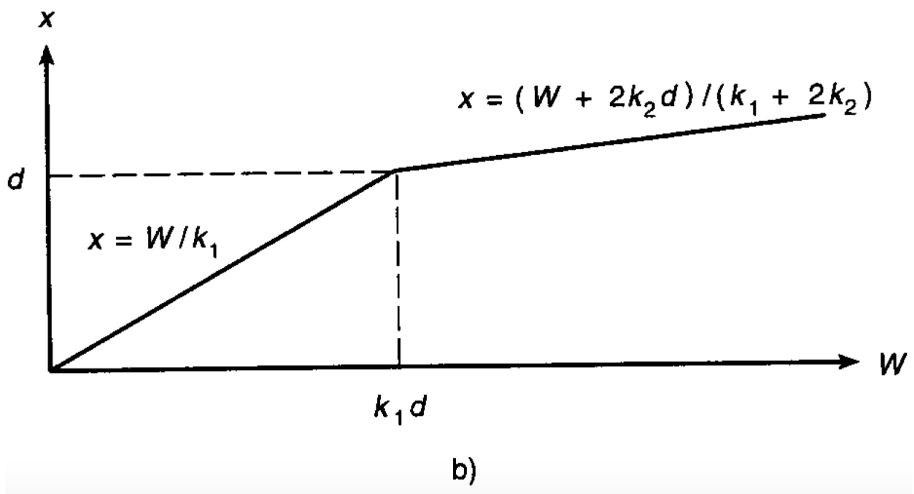


Figura 4.12



Esercizio 12 – 2/3

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:

$$W = k_1 x \qquad \text{se } x < d$$

$$W = k_1 x + 2k_2(x - d) \qquad \text{se } x \geq d$$

Esercizio 12 – 3/3

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 12
(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 13 – 1/2

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:

$$Wh = \frac{1}{2} k_1 x^2 \qquad \text{se } x < d$$

Questa equazione può essere risolta in funzione di x:

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

e

Esercizio 13 – 2/2

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:

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

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 13
(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
```

Crivello di Eratostene
(Esercizio 14)

- Scrivere una funzione MATLAB che prende in input un intero `n` e restituisce in output, <u>utilizzando il Crivello di Eratostene</u>, tutti i numeri primi compresi tra `1` ed `n`

Esempio d'uso

```
>> eratostene(100)

ans =

  Columns 1 through 14

     2     3     5     7    11    13    17    19    23    29    31    37    41    43

  Columns 15 through 25

    47    53    59    61    67    71    73    79    83    89    97

>>
```

Esercizio 14
(Possibile Soluzione)

```
function primi = eratostene(N)

primi = 2:N;
p=2;

while (p <= N)
    for i = 2*p:p:N
        primi(i - 1) = 0;
    end
    p = p + 1;
end

primi = primi(primi > 0);

end
```

Successione di Fibonacci

(Esercizio 15)

a) Calcolare mediante MATLAB, usando strutture di controllo selettive ed iterative, quante coppie di conigli si troveranno nel fosso dopo un anno

b) Scrivere una funzione MATLAB (basata strutture di controllo selettive ed iterative), che prende in input un generico numero di mesi `k` e restituisce in output quante coppie di conigli si troveranno nel fosso dopo tali mesi

c) Scrivere una funzione MATLAB (basata strutture di controllo selettive ed iterative), che prende in input il numero `n` di coppie iniziali ed un generico numero di mesi `k`, e restituisce in output quante coppie di conigli si troveranno nel fosso dopo tali mesi

```
>> fibonacci_iterativo(12)

ans =
     1   1   2   3   5   8   13   21   34   55   89   144
```

```
>> fibonacci_iterativo(2, 12)
ans =
     2   2   4   6   10   16   26   42   68   110   178   288
```

Esempi d'uso

Esercizio 15
(Possibile Soluzione)

```
function out = fibonacci_iterativo(t_iniziale, N)

    if t_iniziale == 0 || t_iniziale == 1
        second_num = 1;
    else
        second_num = t_iniziale;
    end

    if N == 1
        out = t_iniziale;
    elseif N == 2
        out = [t_iniziale second_num];
    else
        out = zeros(1,N);
        out(1:2) = [t_iniziale second_num];
        for idx = 3 : N
            out(idx) = out(idx-1) + out(idx-2);
        end
    end
end
```