



UNIVERSITÀ DEGLI STUDI DI SALERNO

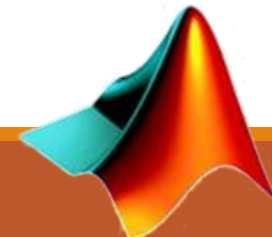


Fondamenti di Informatica

Programmazione in MATLAB | Gestione File | Grafici
Esercitazioni Per Casa | Possibili Soluzioni

Prof. Raffaele Pizzolante

A.A. 2016/17



MATLAB®

Esercitazioni per Casa

Indice

- Esercitazione per Casa | 1 | Possibili Soluzioni
 - Slide pp. 3–8
- Esercitazione per Casa | 2 | Possibili Soluzioni
 - Slide pp. 9–16
- Esercitazione per Casa | 3 | Possibili Soluzioni
 - Slide pp. 17–24

L

<<libreria.txt>>	Target 1 (ind. 1)	Target 2 (ind. 2)	Target 3 (ind. 3)
Genere 1 (indice 1)	15	10	21
Genere 2 (indice 2)	10	25	4
Genere 3 (indice 3)	5	21	7

G

<<guadagni.txt>>	Target 1	Target 2	Target 3
Guadagno	2.20	3.50	2.50

- In questa prova verranno utilizzati una matrice **L** ed un array riga **G**
 - Ogni cella di **L** rappresenta il numero di libri venduti, rivolti ad un certo Target (colonna) di un determinato Genere (riga)
 - Ogni cella di **G** rappresenta il Guadagno unitario per ciascun libro rivolto ad un determinato Target (colonna)
- La matrice **L** contiene esclusivamente dati numerici (evidenziati in arancio nell'esempio)
- L'array riga **G** contiene esclusivamente dati numerici (evidenziati in arancio nell'esempio)
- **N.B.** Per Target si intende una determinata fascia di età, ad es., *Bambini*, *Ragazzi* e *Adulti*

L

<<libreria.txt>>	Target 1 (ind. 1)	Target 2 (ind. 2)	Target 3 (ind. 3)
Genere 1 (indice 1)	15	10	21
Genere 2 (indice 2)	10	25	4
Genere 3 (indice 3)	5	21	7

G

<<guadagni.txt>>	Target 1	Target 2	Target 3
Guadagno	2.20	3.50	2.50

Esercizio 1

Scrivere una funzione `libri_venduti`, che prenda come argomento di input la matrice **L** (*libreria*) ed un intero `indice_target`, e restituisca come argomento di output il **numero totale di libri venduti** per il *Target* avente come indice `indice_target`

- **Esempio:** `libri_venduti(L, 1) → 30`

Possibile Soluzione

```
function [num_libri_venduti] = libri_venduti(L, indice_target)
    num_libri_venduti = sum(L(:, indice_target));
end
```

L

<<libreria.txt>>	Target 1 (ind. 1)	Target 2 (ind. 2)	Target 3 (ind. 3)
Genere 1 (indice 1)	15	10	21
Genere 2 (indice 2)	10	25	4
Genere 3 (indice 3)	5	21	7

G

<<guadagni.txt>>	Target 1	Target 2	Target 3
Guadagno	2.20	3.50	2.50

Esercizio 2

Scrivere una funzione `guadagno_medio_libro`, che prenda come argomento di input la matrice **L** (*libreria*) e l'array **G** (*guadagni*), e restituisca come argomento di output il **guadagno medio per ogni libro venduto**

- **Esempio:** `guadagno_medio_libro(L, G) → 2.8983`

Possibile Soluzione

```
function [guadagno_medio] = guadagno_medio_libro(L, G)
    [nr, nc] = size(L);
    somma_libri_venduti = 0;
    somma_guadagno = 0;

    for i = 1:nc
        somma_libri_venduti = somma_libri_venduti + libri_venduti(L, i);
        somma_guadagno = somma_guadagno + (libri_venduti(L, i) * G(i));
    end

    guadagno_medio = somma_guadagno / somma_libri_venduti;
end
```

L

<<libreria.txt>>	Target 1 (ind. 1)	Target 2 (ind. 2)	Target 3 (ind. 3)
Genere 1 (indice 1)	15	10	21
Genere 2 (indice 2)	10	25	4
Genere 3 (indice 3)	5	21	7

G

<<guadagni.txt>>	Target 1	Target 2	Target 3
Guadagno	2.20	3.50	2.50

Esercizio 3

Scrivere una funzione `genere_meno_venduto` che prenda come argomento di input la matrice **L** (*libreria*) e restituisca come argomento di output l'indice del genere meno venduto

- **Esempio:** `genere_meno_venduto(L) → 3`

Possibile Soluzione

```
function [indice] = genere_meno_venduto(L)
    [valore, indice] = min(sum(L, 2));
end
```

L

<<libreria.txt>>	Target 1 (ind. 1)	Target 2 (ind. 2)	Target 3 (ind. 3)
Genere 1 (indice 1)	15	10	21
Genere 2 (indice 2)	10	25	4
Genere 3 (indice 3)	5	21	7

G

<<guadagni.txt>>	Target 1	Target 2	Target 3
Guadagno	2.20	3.50	2.50

Esercizio 4

Scrivere una funzione `target_piu_redditizio` che prenda in input la matrice **L** (*libreria*) e l'array riga **G** (*guadagni*), e restituisca come argomento di output l'indice del target più redditizio

- **Esempio:** `target_piu_redditizio(L, G) → 2`

Possibile Soluzione

```
function [indice] = target_piu_redditizio(L, G)
    [valore, indice] = max(sum(L) .* G);
end
```

L

<<libreria.txt>>	Target 1 (ind. 1)	Target 2 (ind. 2)	Target 3 (ind. 3)
Genere 1 (indice 1)	15	10	21
Genere 2 (indice 2)	10	25	4
Genere 3 (indice 3)	5	21	7

G

<<guadagni.txt>>	Target 1	Target 2	Target 3
Guadagno	2.20	3.50	2.50

Esercizio 5

Scrivere un M-File Script chiamato `libreria_script.m` che effettui le seguenti operazioni

1. Importi la matrice `L` dal file `libreria.txt`
2. Importi l'array `G` dal file `guadagni.txt`
3. Invochi la funzione dell'Esercizio 4 (chiamata `target_piu_redditizio`) con gli argomenti di input: `L` e `G`, ed infine mostri a video l'output della funzione stessa

NOTA: I file `libreria.txt` e `guadagni.txt` contengono solo dati numerici. È utilizzato il separatore virgola (,) per separare le colonne. Si assuma che i file siano memorizzati all'interno della **Current Directory**

Possibile Soluzione

```
L = importdata('libreria.txt');
G = importdata('guadagni.txt');

target_redditizio = target_piu_redditizio(L, G)
```

M

<<magazzini.txt>>	Pantalone (ind. 1)	Camicia (ind. 2)	Maglia (ind. 3)
Roma (indice 1)	4	25	18
Milano (indice 2)	15	3	19
Napoli (indice 3)	20	10	21

P

<<prezzi.txt>>	Pantalone	Camicia	Maglia
Prezzo	40	30	20

- In questa prova verranno utilizzati una matrice **M** ed un array **P**
 - Ogni cella di **M** rappresenta la quantità di un certo prodotto (colonna) disponibile in una determinata sede del magazzino (riga)
 - Ogni cella di **P** rappresenta l'importo necessario per uno specifico prodotto (colonna)
- La matrice **M** contiene esclusivamente dati numerici (evidenziati in arancio nell'esempio)
- L'array **P** contiene esclusivamente dati numerici (evidenziati in arancio nell'esempio)

M	<<magazzini.txt>>	Pantalone (ind. 1)	Camicia (ind. 2)	Maglia (ind. 3)
	Roma (indice 1)	4	25	18
	Milano (indice 2)	15	3	19
	Napoli (indice 3)	20	10	21
P	<<prezzi.txt>>	Pantalone	Camicia	Maglia
	Prezzo	40	30	20

Esercizio 1

- Scrivere una funzione `scorte`, che prenda in input la matrice **M** (*magazzini*) e restituisca un array **S**, in cui ogni elemento **S (i)** è definito come segue

$$S(i) = \begin{cases} 1 & \text{se è necessario fare un rifornimento nella sede con indice } i \\ 0 & \text{altrimenti} \end{cases}$$

- Si noti che **un rifornimento è necessario** se vi sono uno o più prodotti la cui **quantità** presente in magazzino è **inferiore a 5**
- Nell'esempio, la funzione `scorte` restituirà $S = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$
- **Nota**: È indifferente che **S** sia un vettore riga o un vettore colonna

M	<<magazzini.txt>>	Pantalone (ind. 1)	Camicia (ind. 2)	Maglia (ind. 3)
	Roma (indice 1)	4	25	18
	Milano (indice 2)	15	3	19
	Napoli (indice 3)	20	10	21
P	<<prezzi.txt>>	Pantalone	Camicia	Maglia
	Prezzo	40	30	20

Esercizio 1 - Possibile Soluzione

```
function [S] = scorte(M)
    [nr, nc] = size(M);

    for i=1:nr
        min_sede = min(M(:,i));

        if (min_sede < 5)
            S(i) = 1;
        else
            S(i) = 0;
        end
    end
    S = S';
end
```

Esempio d'uso

```
>> S = scorte(M)
```

```
S =
```

```
1
1
0
```

M	<<magazzini.txt>>	Pantalone (ind. 1)	Camicia (ind. 2)	Maglia (ind. 3)
	Roma (indice 1)	4	25	18
	Milano (indice 2)	15	3	19
	Napoli (indice 3)	20	10	21
P	<<prezzi.txt>>	Pantalone	Camicia	Maglia
	Prezzo	40	30	20

Esercizio 2

- Scrivere una funzione chiamata `importi`, che prenda in input la matrice **M** (*magazzini*) e l'array **P** (*prezzi*), e restituisca in output un array **I**
- Ogni elemento dell'array **I** dovrà contenere l'importo totale ricavabile dalla vendita della merce contenuta in un determinato magazzino
- **Esempio**
 - $I(1) = (4 * 40) + (25 * 30) + (18 * 20) \rightarrow$ magazzino **Roma**
 - $I(2) = (15 * 40) + (3 * 30) + (19 * 20) \rightarrow$ magazzino **Milano**
 - $I(3) = (20 * 40) + (10 * 30) + (21 * 20) \rightarrow$ magazzino **Napoli**
- **Nota**: È indifferente che **I** sia un vettore riga o un vettore colonna

M	<<magazzini.txt>>	Pantalone (ind. 1)	Camicia (ind. 2)	Maglia (ind. 3)
	Roma (indice 1)	4	25	18
	Milano (indice 2)	15	3	19
	Napoli (indice 3)	20	10	21
P	<<prezzi.txt>>	Pantalone	Camicia	Maglia
	Prezzo	40	30	20

Esercizio 2 - Possibile Soluzione

```
function [I] = importi(M, P)
    [nr, nc] = size(M);

    for i = 1:nr
        I(i) = sum(M(i,:) .* P);
    end

    I = I';
end
```

Esempio d'uso

```
>> I = importi(M, P)

I =

    1270
    1070
    1520
```

M	<<magazzini.txt>>	Pantalone (ind. 1)	Camicia (ind. 2)	Maglia (ind. 3)
	Roma (indice 1)	4	25	18
	Milano (indice 2)	15	3	19
	Napoli (indice 3)	20	10	21
P	<<prezzi.txt>>	Pantalone	Camicia	Maglia
	Prezzo	40	30	20

Esercizio 3

- Scrivere una funzione chiamata `piu_fornito`, che prenda in input la matrice **M** (*magazzini*) e restituisca in output l'indice del magazzino che ha più prodotti

Possibile Soluzione

```
function [indice_max] = piu_fornito(M)
[nr, nc] = size(M);

max = sum(M(1,:));
indice_max = 1;

for i=2:nr
    if max < sum(M(i,:))
        max = sum(M(i,:));
        indice_max = i;
    end
end
end
```

Esempio d'uso

```
>> indice_piu_fornito = piu_fornito(M)

indice_piu_fornito =

    3
```

M	<<magazzini.txt>>	Pantalone (ind. 1)	Camicia (ind. 2)	Maglia (ind. 3)
	Roma (indice 1)	4	25	18
	Milano (indice 2)	15	3	19
	Napoli (indice 3)	20	10	21
P	<<prezzi.txt>>	Pantalone	Camicia	Maglia
	Prezzo	40	30	20

Esercizio 4

- Scrivere una funzione chiamata `totale_camicie`, che prenda in input la matrice **M** (*magazzini*) e restituisca in output il **numero totale di camicie** (in tutti i magazzini)

Possibile Soluzione

```
function [num_camicie] = totale_camicie(M)
    num_camicie = sum(M(:,2));
end
```

Esempio d'uso

```
>> camicie = totale_camicie(M)

camicie =
    38
```

M	<<magazzini.txt>>	Pantalone (ind. 1)	Camicia (ind. 2)	Maglia (ind. 3)
	Roma (indice 1)	4	25	18
	Milano (indice 2)	15	3	19
	Napoli (indice 3)	20	10	21
P	<<prezzi.txt>>	Pantalone	Camicia	Maglia
	Prezzo	40	30	20

Esercizio 5

Scrivere un M-File Script chiamato `magazzini_script.m` che effettui le seguenti operazioni

1. Importi la matrice M dal file `magazzini.txt`
2. Importi l'array P dal file `prezzi.txt`
3. Invochi la funzione dell'Esercizio 3 (chiamata `piu_fornito`) con argomento di input M, e mostri a video l'output della funzione stessa

NOTA: I file `magazzini.txt` e `prezzi.txt` contengono solo dati numerici. È utilizzato il separatore virgola (,) per separare le colonne. Si assuma che i file siano memorizzati all'interno della **Current Directory**

Possibile Soluzione

```
M = importdata('magazzini.txt');
P = importdata('prezzi.txt');

piu_fornito(M)
```

C

Piatti

<<costi_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	30	6	11	78	100	13
Ristorante 2 (ind. 2)	25	42	25	12	35	91
Ristorante 3 (ind. 3)	8	48	20	22	14	68
Ristorante 4 (ind. 4)	15	18	24	40	72	4
Ristorante 5 (ind. 5)	16	69	83	31	33	81
Ristorante 6 (ind. 6)	9	13	15	58	19	52

Costi per Piatto

T

Piatti

<<tipo_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	2	1	2	3	1	4
Ristorante 2 (ind. 2)	1	1	2	1	3	4
Ristorante 3 (ind. 3)	3	1	1	1	2	3
Ristorante 4 (ind. 4)	4	1	4	1	2	2
Ristorante 5 (ind. 5)	1	1	2	3	2	3
Ristorante 6 (ind. 6)	3	4	2	1	4	4

Tipologia Piatto: 1 Antipasto, 2 Primo, 3 Secondo, 4 Contorno

NOTA: Negli esercizi possono essere utilizzate funzioni viste a lezione (negli esempi), funzioni realizzate negli esercizi precedenti e/o funzioni built-in di MATLAB

- In questa esercitazione verranno utilizzate una matrice **C** ed una matrice **T**
- La matrice **C** rappresenta i **costi per ciascun piatto** offerto da **ciascun ristorante**
 - **C(indice_ristorante, indice_piatto)** rappresenta il costo del Piatto avente indice **indice_piatto**, offerto dal Ristorante avente indice **indice_ristorante**
 - **Esempio:** **C(4, 2)** → ha valore 18 e indica che il Piatto 2 offerto dal Ristorante 4 costa 18 €
- La matrice **T** specifica, per ogni piatto fornito da ciascun ristorante, la relativa tipologia
 - **Esempio:** **T(6, 3)** → ha valore 2 e indica che il Piatto 3 fornito dal Ristorante 6 è di tipo 2
- Le matrici contengono esclusivamente dati numerici (evidenziati in arancio nell'esempio)

C

Piatti

<<costi_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	30	6	11	78	100	13
Ristorante 2 (ind. 2)	25	42	25	12	35	91
Ristorante 3 (ind. 3)	8	48	20	22	14	68
Ristorante 4 (ind. 4)	15	18	24	40	72	4
Ristorante 5 (ind. 5)	16	69	83	31	33	81
Ristorante 6 (ind. 6)	9	13	15	58	19	52

Costi per Piatto

T

Piatti

<<tipo_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	2	1	2	3	1	4
Ristorante 2 (ind. 2)	1	1	2	1	3	4
Ristorante 3 (ind. 3)	3	1	1	1	2	3
Ristorante 4 (ind. 4)	4	1	4	1	2	2
Ristorante 5 (ind. 5)	1	1	2	3	2	3
Ristorante 6 (ind. 6)	3	4	2	1	4	4

Tipologia Piatto: 1 Antipasto, 2 Primo, 3 Secondo, 4 Contorno

NOTA: Negli esercizi possono essere utilizzate funzioni viste a lezione (negli esempi), funzioni realizzate negli esercizi precedenti e/o funzioni built-in di MATLAB

Esercizio 1

Scrivere una funzione chiamata `piatto_economico`, che prenda come argomenti di input: la matrice `C` (*costi*), la matrice `T` (*tipo pasti*) ed un intero `indice_tipo_piatto`, e restituisca come argomento di output il costo del piatto, indicato da `indice_tipo_piatto`, più economico

- **Esempio:** `piatto_economico(C, T, 3)` → restituisce 8

Possibile Soluzione

```
function [ costo_piatto_economico ] = piatto_economico(C, T, indice_tipo_piatto)
    costo_piatto_economico = min(C(find(T == indice_tipo_piatto)));
end
```

C

Piatti

<<costi_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	30	6	11	78	100	13
Ristorante 2 (ind. 2)	25	42	25	12	35	91
Ristorante 3 (ind. 3)	8	48	20	22	14	68
Ristorante 4 (ind. 4)	15	18	24	40	72	4
Ristorante 5 (ind. 5)	16	69	83	31	33	81
Ristorante 6 (ind. 6)	9	13	15	58	19	52

Costi per Piatto

T

Piatti

<<tipo_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	2	1	2	3	1	4
Ristorante 2 (ind. 2)	1	1	2	1	3	4
Ristorante 3 (ind. 3)	3	1	1	1	2	3
Ristorante 4 (ind. 4)	4	1	4	1	2	2
Ristorante 5 (ind. 5)	1	1	2	3	2	3
Ristorante 6 (ind. 6)	3	4	2	1	4	4

Tipologia Piatto: 1 Antipasto, 2 Primo, 3 Secondo, 4 Contorno

NOTA: Negli esercizi possono essere utilizzate funzioni viste a lezione (negli esempi), funzioni realizzate negli esercizi precedenti e/o funzioni built-in di MATLAB

Esercizio 2

Scrivere una funzione chiamata `menu_completo`, che prenda come argomento di input la matrice `T` (*tipo pasti*) ed un intero `indice_ristorante`, e restituisca come argomento di output

- 1, se il Ristorante avente indice `indice_ristorante` offre un menù completo
- 0, altrimenti

N.B. Per menù completo si intende che il ristorante fornisce almeno una portata per ciascun tipo di piatto

- **Esempio 1:** `menu_completo(T, 1) → restituisce 1`
- **Esempio 2:** `menu_completo(T, 3) → restituisce 0`

Possibile Soluzione 1/2

```
function [ ris ] = menu_completo(T, indice_ristorante)
    tipologie_piatti = 1:4;
    riga_T = T(indice_ristorante, :);
    intersezione = intersect(riga_T, tipologie_piatti);

    ris = isequal(intersezione, tipologie_piatti);
end
```

C

Piatti

<<costi_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	30	6	11	78	100	13
Ristorante 2 (ind. 2)	25	42	25	12	35	91
Ristorante 3 (ind. 3)	8	48	20	22	14	68
Ristorante 4 (ind. 4)	15	18	24	40	72	4
Ristorante 5 (ind. 5)	16	69	83	31	33	81
Ristorante 6 (ind. 6)	9	13	15	58	19	52

Costi per Piatto

T

Piatti

<<tipo_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	2	1	2	3	1	4
Ristorante 2 (ind. 2)	1	1	2	1	3	4
Ristorante 3 (ind. 3)	3	1	1	1	2	3
Ristorante 4 (ind. 4)	4	1	4	1	2	2
Ristorante 5 (ind. 5)	1	1	2	3	2	3
Ristorante 6 (ind. 6)	3	4	2	1	4	4

Tipologia Piatto: 1 Antipasto, 2 Primo, 3 Secondo, 4 Contorno

NOTA: Negli esercizi possono essere utilizzate funzioni viste a lezione (negli esempi), funzioni realizzate negli esercizi precedenti e/o funzioni built-in di MATLAB

Esercizio 2

Scrivere una funzione chiamata `menu_completo`, che prenda come argomento di input la matrice `T` (*tipo pasti*) ed un intero `indice_ristorante`, e restituisca come argomento di output

- 1, se il Ristorante avente indice `indice_ristorante` offre un menù completo
- 0, altrimenti

N.B. Per menù completo si intende che il ristorante fornisce almeno una portata per ciascun tipo di piatto

- **Esempio 1:** `menu_completo(T, 1) → restituisce 1`
- **Esempio 2:** `menu_completo(T, 3) → restituisce 0`

Possibile Soluzione 2/2

```
function [ ris ] = menu_completo(T, indice_ristorante)
    tipologie_piatti = 1:4;
    somma = 0;

    for indice_tipologia = 1:length(tipologie_piatti)
        if length(find(T(indice_ristorante, :) == indice_tipologia)) >= 1
            somma = somma + 1;
        end
    end

    ris = somma == 4;
end
```

C

Piatti

<<costi_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	30	6	11	78	100	13
Ristorante 2 (ind. 2)	25	42	25	12	35	91
Ristorante 3 (ind. 3)	8	48	20	22	14	68
Ristorante 4 (ind. 4)	15	18	24	40	72	4
Ristorante 5 (ind. 5)	16	69	83	31	33	81
Ristorante 6 (ind. 6)	9	13	15	58	19	52

Costi per Piatto

T

Piatti

<<tipo_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	2	1	2	3	1	4
Ristorante 2 (ind. 2)	1	1	2	1	3	4
Ristorante 3 (ind. 3)	3	1	1	1	2	3
Ristorante 4 (ind. 4)	4	1	4	1	2	2
Ristorante 5 (ind. 5)	1	1	2	3	2	3
Ristorante 6 (ind. 6)	3	4	2	1	4	4

Tipologia Piatto: 1 Antipasto, 2 Primo, 3 Secondo, 4 Contorno

NOTA: Negli esercizi possono essere utilizzate funzioni viste a lezione (negli esempi), funzioni realizzate negli esercizi precedenti e/o funzioni built-in di MATLAB

Esercizio 3

Scrivere una funzione chiamata `ampia_scelta`, che prenda come argomenti di input la matrice `T` (*tipo pasti*) ed un intero `indice_tipo_piatto`, e restituisca come argomento di output l'indice del ristorante che offre la più ampia scelta associata al tipo di piatto selezionato

- **Esempio:** `ampia_scelta(T, 4)` → restituisce 6

Possibile Soluzione

```
function [ indice_ristorante ] = ampia_scelta(T, indice_tipo_piatto)
    [num_ristoranti, num_piatti] = size(T);

    for indice_ristorante = 1:num_ristoranti
        x = find(T(indice_ristorante, :) == indice_tipo_piatto);
        ristoranti(indice_ristorante) = length(x);
    end

    [valore, indice_ristorante] = max(ristoranti);
end
```

C

Piatti

<<costi_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	30	6	11	78	100	13
Ristorante 2 (ind. 2)	25	42	25	12	35	91
Ristorante 3 (ind. 3)	8	48	20	22	14	68
Ristorante 4 (ind. 4)	15	18	24	40	72	4
Ristorante 5 (ind. 5)	16	69	83	31	33	81
Ristorante 6 (ind. 6)	9	13	15	58	19	52

Costi per Piatto

T

Piatti

<<tipo_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	2	1	2	3	1	4
Ristorante 2 (ind. 2)	1	1	2	1	3	4
Ristorante 3 (ind. 3)	3	1	1	1	2	3
Ristorante 4 (ind. 4)	4	1	4	1	2	2
Ristorante 5 (ind. 5)	1	1	2	3	2	3
Ristorante 6 (ind. 6)	3	4	2	1	4	4

Tipologia Piatto: 1 Antipasto, 2 Primo, 3 Secondo, 4 Contorno

NOTA: Negli esercizi possono essere utilizzate funzioni viste a lezione (negli esempi), funzioni realizzate negli esercizi precedenti e/o funzioni built-in di MATLAB

Esercizio 4

Scrivere una funzione chiamata `tipo_piatto_caro`, che prenda come argomenti di input la matrice `C` (*costi*) e la matrice `T` (*tipi pasti*), e restituisca come argomento di output la tipologia del piatto più caro tra tutti i ristoranti

- Esempio:** `tipo_piatto_caro(C, T)` → restituisce 1 (dato che il piatto più caro ha prezzo di 100€ ed è della tipologia 1)

Possibile Soluzione

```
function [ tipo_piatto ] = tipo_piatto_caro(C, T)
    tipo_piatto = T(find(C == max(max(C)))) ;
end
```

C

Piatti

<<costi_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	30	6	11	78	100	13
Ristorante 2 (ind. 2)	25	42	25	12	35	91
Ristorante 3 (ind. 3)	8	48	20	22	14	68
Ristorante 4 (ind. 4)	15	18	24	40	72	4
Ristorante 5 (ind. 5)	16	69	83	31	33	81
Ristorante 6 (ind. 6)	9	13	15	58	19	52

Costi per Piatto

T

Piatti

<<tipo_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	2	1	2	3	1	4
Ristorante 2 (ind. 2)	1	1	2	1	3	4
Ristorante 3 (ind. 3)	3	1	1	1	2	3
Ristorante 4 (ind. 4)	4	1	4	1	2	2
Ristorante 5 (ind. 5)	1	1	2	3	2	3
Ristorante 6 (ind. 6)	3	4	2	1	4	4

Tipologia Piatto: 1 Antipasto, 2 Primo, 3 Secondo, 4 Contorno

NOTA: Negli esercizi possono essere utilizzate funzioni viste a lezione (negli esempi), funzioni realizzate negli esercizi precedenti e/o funzioni built-in di MATLAB

Esercizio 5

- Scrivere un M-File Script chiamato `ristoranti_script.m` che effettui le seguenti operazioni
1. Importi la matrice C dal file `costi_piatti.txt` (Mostrato in seguito)
 2. Importi la matrice T dal file `tipo_piatti.txt` (Mostrato in seguito)
 3. Invochi la funzione dell'Esercizio 4 (chiamata `tipo_piatto_caro`) con gli argomenti di input: C e T, e mostri a video il risultato della funzione stessa
 4. Generi un grafico a barre con le seguenti caratteristiche
 1. Asse X → Rappresenta gli indici dei piatti
 2. Asse Y → Rappresenta il costo del piatto più caro (tenendo conto di tutti i ristoranti), in base all'indice del piatto stesso (specificato sull'asse X)

NOTA: I file `costi_piatti.txt` e `tipo_piatti.txt` (mostrati in seguito) contengono solo dati numerici. È utilizzato il separatore virgola (,) per separare le colonne. Si assuma che i file siano memorizzati all'interno della **Current Directory**

C

Piatti

<<costi_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	30	6	11	78	100	13
Ristorante 2 (ind. 2)	25	42	25	12	35	91
Ristorante 3 (ind. 3)	8	48	20	22	14	68
Ristorante 4 (ind. 4)	15	18	24	40	72	4
Ristorante 5 (ind. 5)	16	69	83	31	33	81
Ristorante 6 (ind. 6)	9	13	15	58	19	52

Costi per Piatto

T

Piatti

<<tipo_piatti.txt>>	1	2	3	4	5	6
Ristorante 1 (ind. 1)	2	1	2	3	1	4
Ristorante 2 (ind. 2)	1	1	2	1	3	4
Ristorante 3 (ind. 3)	3	1	1	1	2	3
Ristorante 4 (ind. 4)	4	1	4	1	2	2
Ristorante 5 (ind. 5)	1	1	2	3	2	3
Ristorante 6 (ind. 6)	3	4	2	1	4	4

Tipologia Piatto: 1 Antipasto, 2 Primo, 3 Secondo, 4 Contorno

NOTA: Negli esercizi possono essere utilizzate funzioni viste a lezione (negli esempi), funzioni realizzate negli esercizi precedenti e/o funzioni built-in di MATLAB

Esercizio 5

Possibile Soluzione (contenuto del file ristoranti_script.m)

```
C = importdata('costi_piatto.txt');
T = importdata('tipo_piatti.txt');

tipo_piatto_caro(C, T)

[num_ristoranti, num_piatti] = size(C);

x = 1:num_piatti;
y = max(C);

plot(x, y)
```