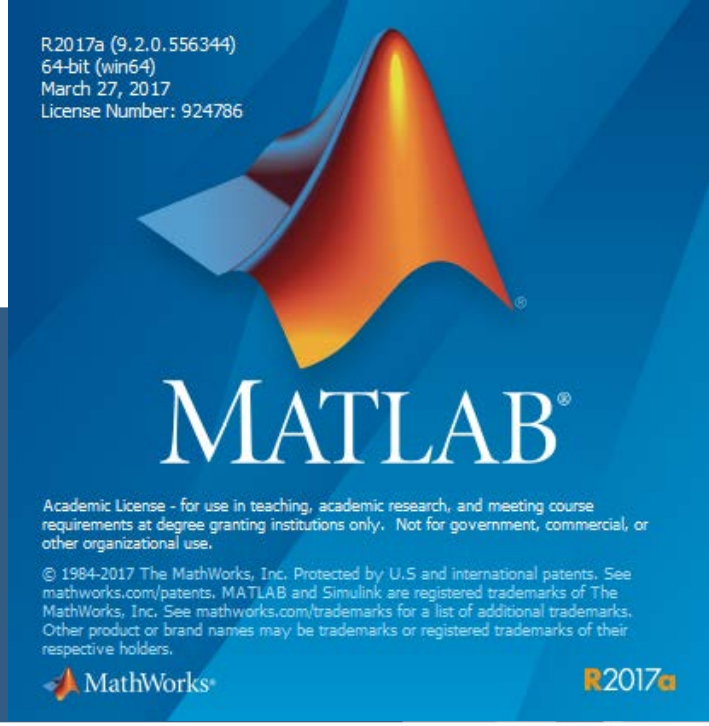
POLITECNICO DI MILANO

Lab







Esercitazione 00

Introduzione a Matlab

Corso di Strumentazione e Controllo di Impianti Chimici

Prof. Davide Manca

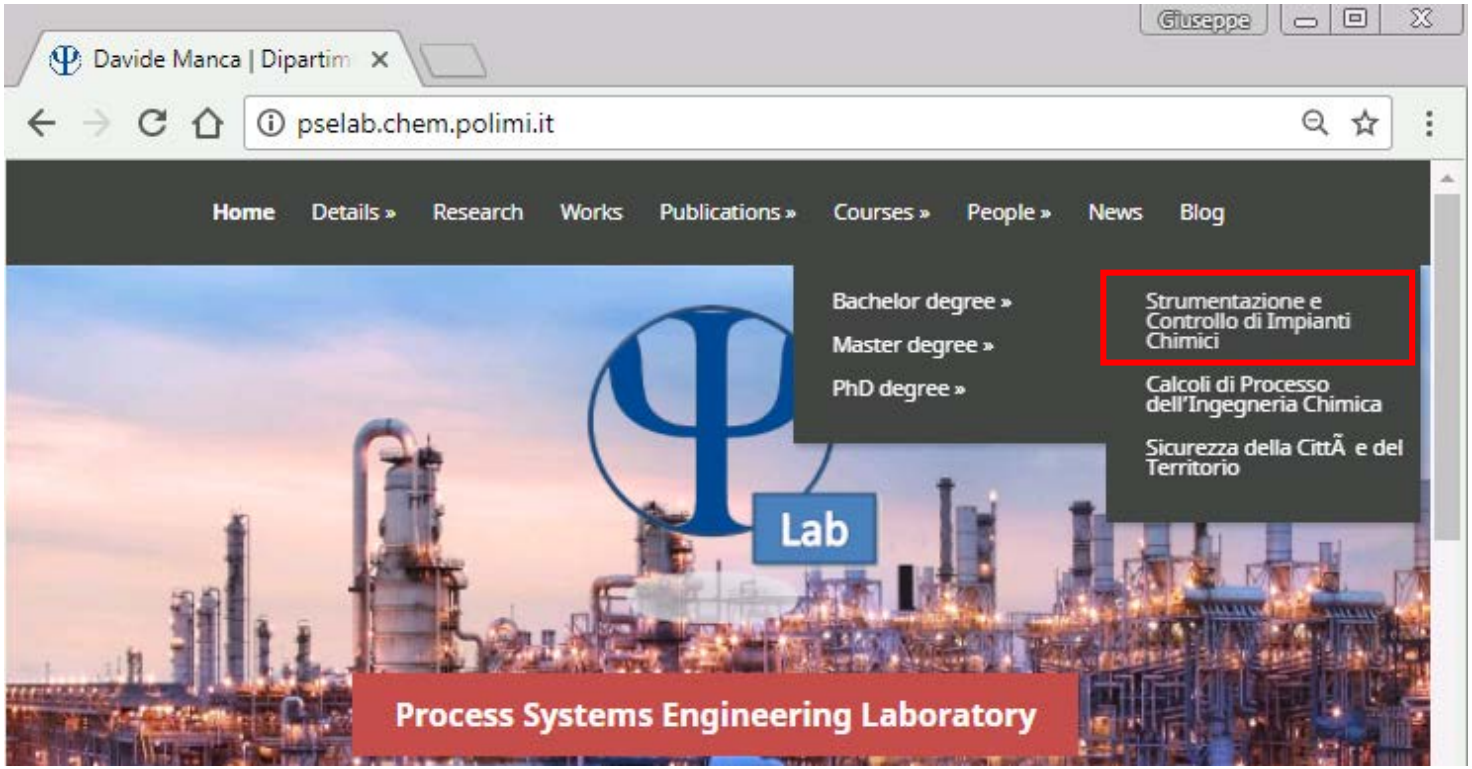
Tutor: Giuseppe Pesenti

Tutor:

Giuseppe Pesenti – giuseppe.pesenti@polimi.it

Materiale del corso:

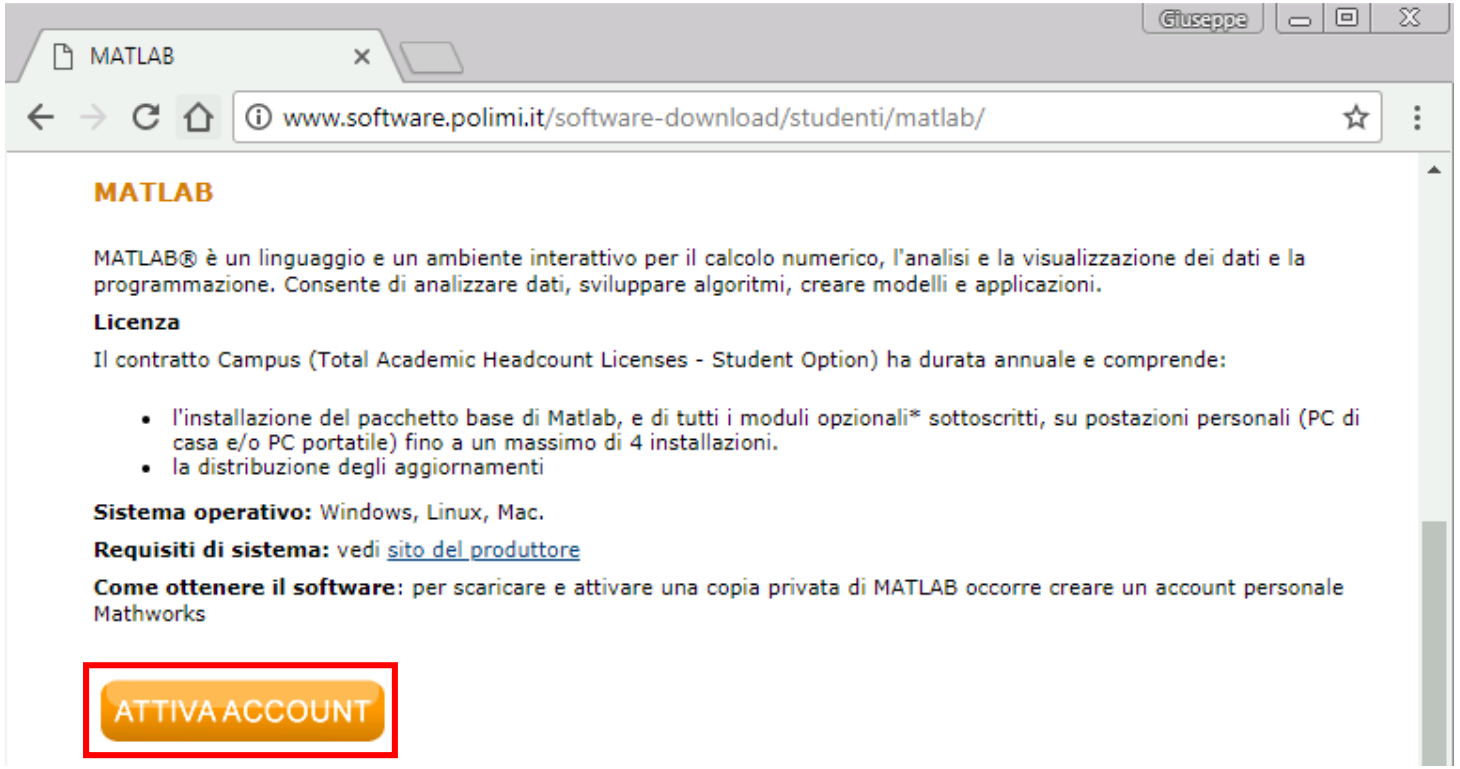
PSE-Lab – Courses – Bachelor Degree – Strumentazione e Controllo di Impianti Chimici



Matlab

Matlab: ambiente di calcolo numerico sviluppato da Mathworks.
Utilizza l'omonimo linguaggio di programmazione.

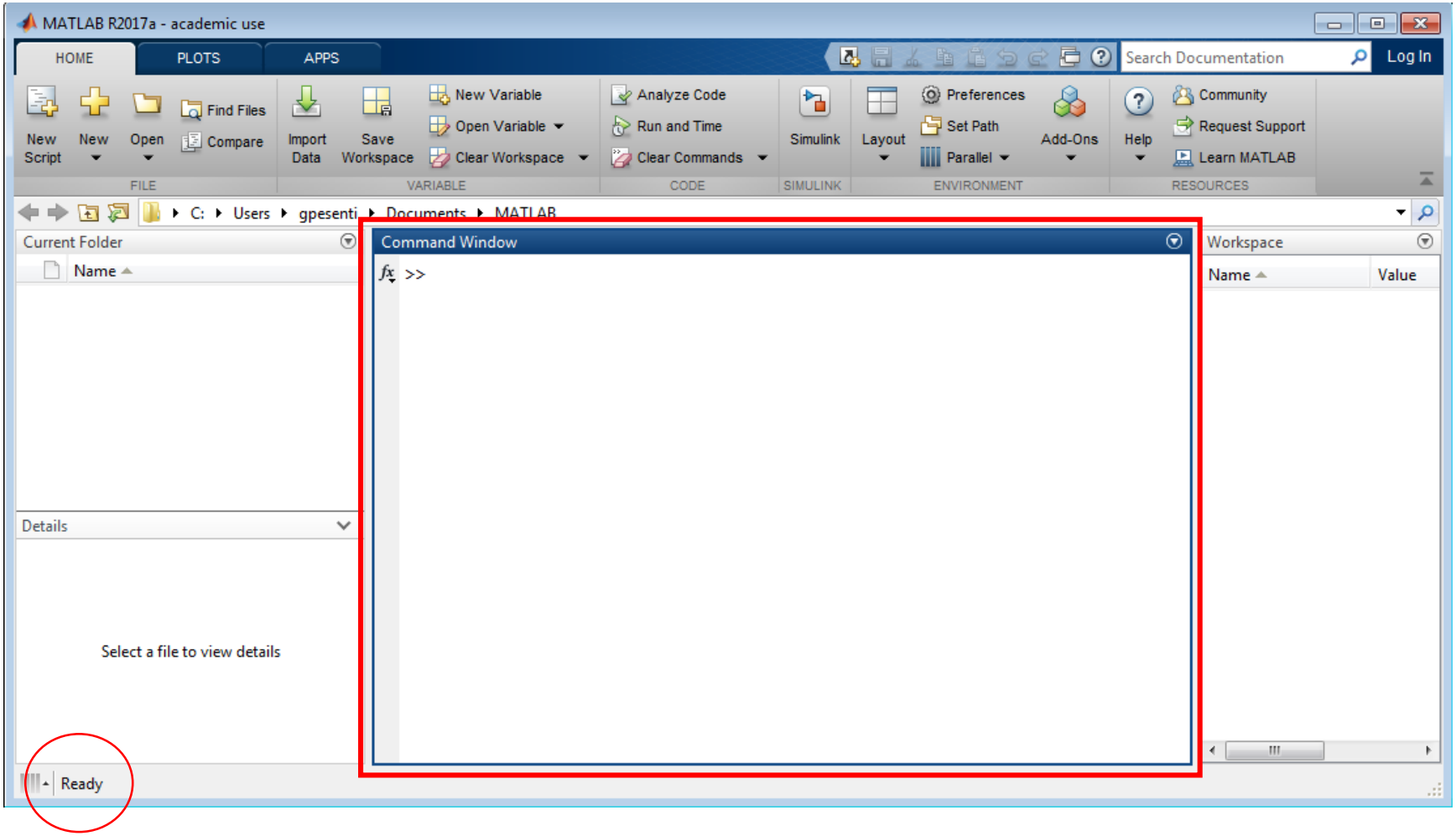
Scaricare e installare Matlab:
Software Polimi – Studenti – Matlab Academy e Matlab - Matlab



Command window

Command window:

Interfaccia stile riga di comando, per eseguire immediatamente dei comandi



Calcoli di base

Usare Matlab come una calcolatrice:

+ - * / ^

Esercizi: effettuare le seguenti operazioni nella command window di Matlab

- 5+6+7 = 18
- 12*(-2.5+1.99) = -6.1200
- 56^34 = 2.7441e+59
- 7.56/(3^4) = 0.0933
- 3e7*2e-2 = 600000
- 22/0 = Inf

exp() log() log10()

- 3.67*exp(8.56) = 1.9153e+04
- log10(1e5) = 5
- exp(log(17)) = 17

Nota: Matlab di default opera in doppia precisione

Vettori e matrici

Creare vettori:

Vettore riga **[1 2]** o **[1, 2]**

Vettore colonna **[1; 2]**

Esercizi: creare i seguenti vettori

- [1 2.5 4 5.5]
- [-1.5; -0.75; 0; 0.75]
- [2^0 2^1 2^2 2^3 2^4]

Creare matrici:

[1 2; 3 4]

Esercizi: creare le seguenti matrici

- [2/5 3/5 4/5; 7/10 8/10 9/10]
- [1 10; 100 1000]

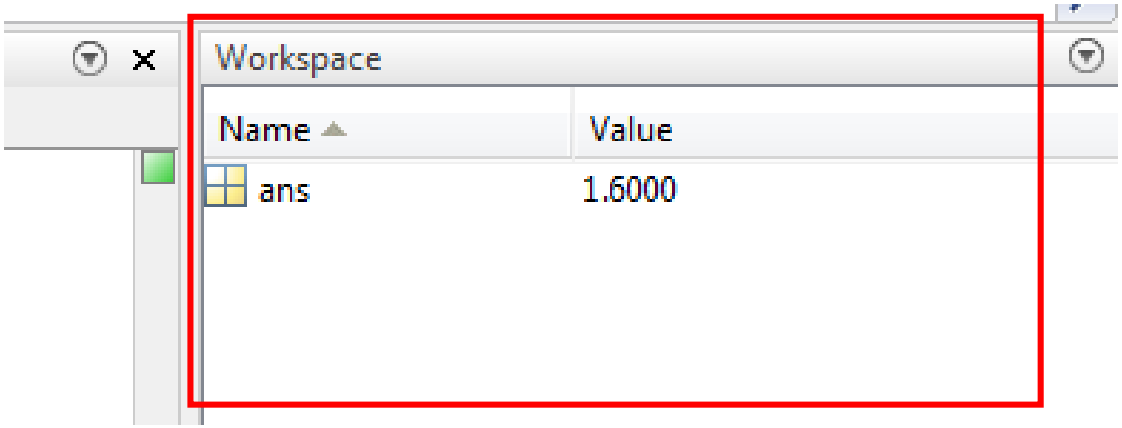
Variabile ans


```
>> 2*2

ans =

     4
```

Risultato dei calcoli memorizzato come **ans**

Esercizio: utilizzare la variabile ans

```
>>  2 * 2          = 4
>>  ans + 4        = 8
>>  ans / 5        = 1.60000
```



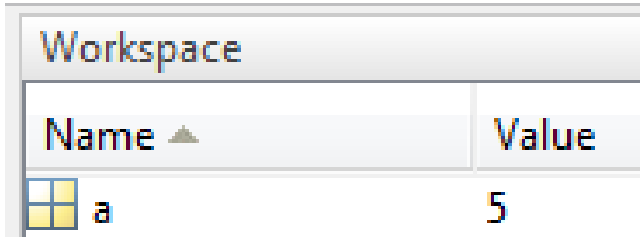
Variabili

Memorizzare un valore in un'area di memoria:

a = 5

assegna alla variabile a il **valore** a destra dell'uguale

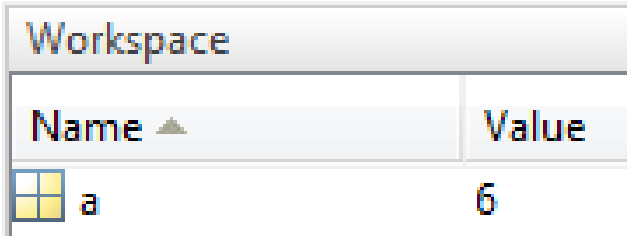
nota: Matlab è **case-sensitive**



a = a + 1

assegna semplicemente il valore

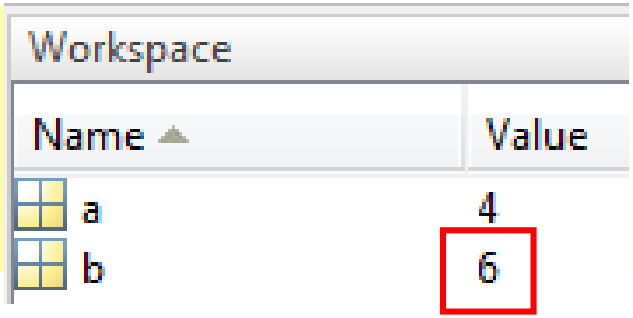
- **non** è un'uguaglianza
- **non** definisce identità o riferimenti duraturi



Esercizio: definire e utilizzare le variabili **a** e **b**

```
>>  a = 3
>>  b = 2 * a
>>  a = a + 1
```

Una variabile cambia **solo** con un'assegnazione



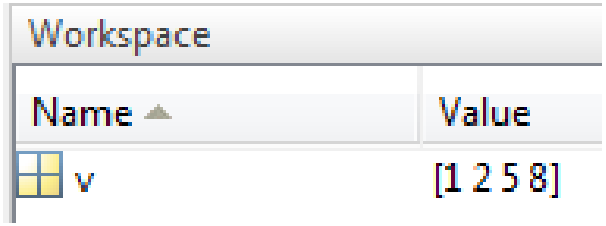
Per sopprimere l'output: **;**

Vettori, variabili e matrici

Memorizzare un vettore o una matrice in un'area di memoria:

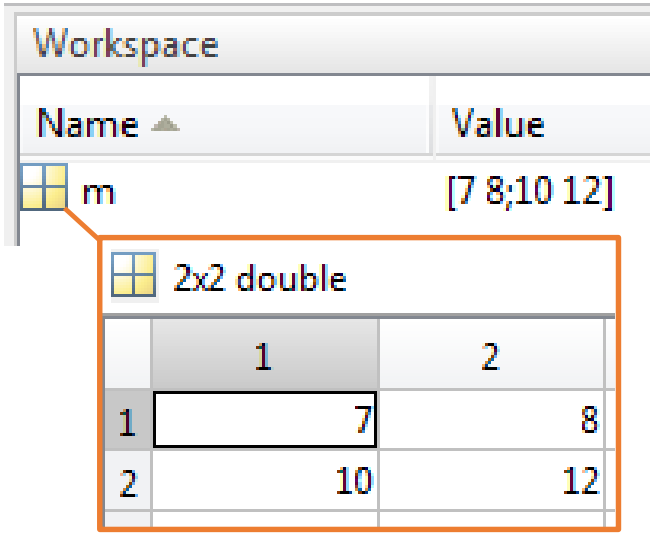
v = [1 2 5 8];

assegna alla variabile v il **vettore di valori** a destra



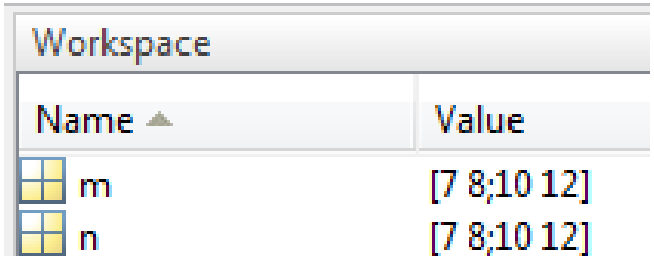
m = [7 8; 10 12];

assegna a m la **matrice di valori** a destra dell'uguale



n = m;

assegna a n l'**intera matrice di valori** assegnata a m



Creazione di vettori e matrici speciali

Creare vettori di punti equi spaziati:

- **>> 0:100**
- **>> 0:0.01:100**
- **>> linspace(2017, 2018, 365)**

Creare vettori e matrici

- **zeros( )**
- ones( )
- eye( )
- rand( )
- magic( )

specificando le dimensioni

- (m, n) rappresenta una matrice m x n con m righe, n colonne
- (n,1) rappresenta un vettore colonna
- (1,n) rappresenta un vettore riga
- (n) sottintende (n, n) e rappresenta una matrice quadrata n x n

Esercizio: creare un vettore di numeri pari da 2 a 100

>> 2:2:100

Esercizio: creare un vettore di 100 punti equispaziati tra 3 e 47

>> linspace(3, 47, 100)

Esercizio: creare una matrice di zeri 5x5
e assegnare 10 al secondo elemento della quarta riga

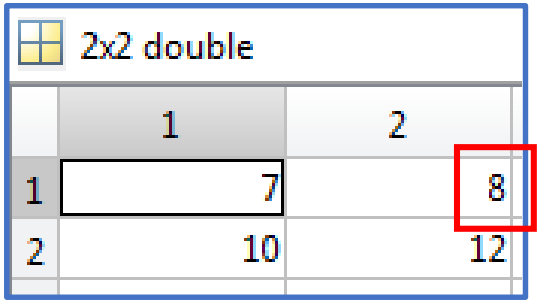
>> M = zeros(5);
>> M(4,2) = 10;

Accesso a elementi di vettori e matrici

Accedere/selezionare il valore di un singolo elemento:

m (1 , 2) = 8

riga colonna



nota: il conteggio degli elementi parte da 1

Accedere/selezionare più valori contemporaneamente:

mat(1:3, 2) **mat([1 3], 3)** **mat(1, :)**

Esercizio: creare una matrice di zeri 4x4, quindi sostituire alla seconda riga un vettore con i valori che determinano tre intervalli equispaziati tra 3 e 12

```
>>  n = zeros(4);
>>  v = linspace(3,  12,  4);
>>  n(2,  :) = v;
```

Per trasporre una matrice o vettore: '

Esercizio: trasporre il vettore creato e sostituirlo anche alla prima colonna della matrice

```
>>  v = v';
>>  n(:,  1) = v;
```

Operazioni tra vettori e matrici

Matlab contiene già funzioni per gestire le operazioni applicate a vettori e matrici

Operazione:	**Condizione:**
A'	-
A + B, A – B	le matrici/vettori A e B hanno le stesse dimensioni
A .* B, A ./ B	le matrici/vettori A e B hanno le stesse dimensioni
A * B	le colonne di A sono tante quante le righe di B
	Per matrici quadrate:
det(A)	**-**
A^(-1) o **inv(A)**	A è invertibile
A / B	B è invertibile, in quanto si calcola **B*inv(A)**
A \ B	A è invertibile, in quanto si calcola **inv(A)*B**

Esercizio: risolvere il seguente sistema lineare

$$\begin{cases} x - 3y + z = 2 \\ 3x - 4y + z = 0 \\ 2y - z = 1 \end{cases}$$

Ax=b

$A^{-1}*A*x=A^{-1}*b$

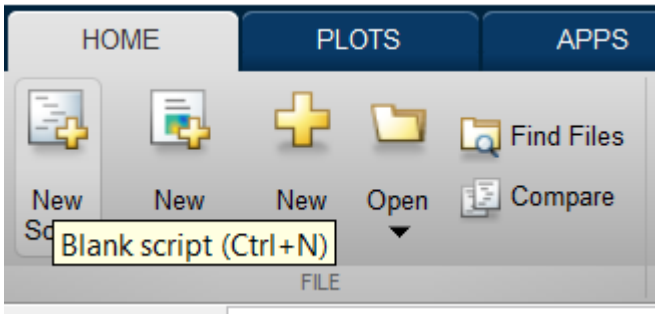
$x=A^{-1}*b$

```
>> A=[1 -3 1; 3 -4 1; 0 2 -1];
>> b=[2 0 1]';
>> x=inv(A)*b;        >> x=A\b;
```

x = -5
y = -8
z = -17

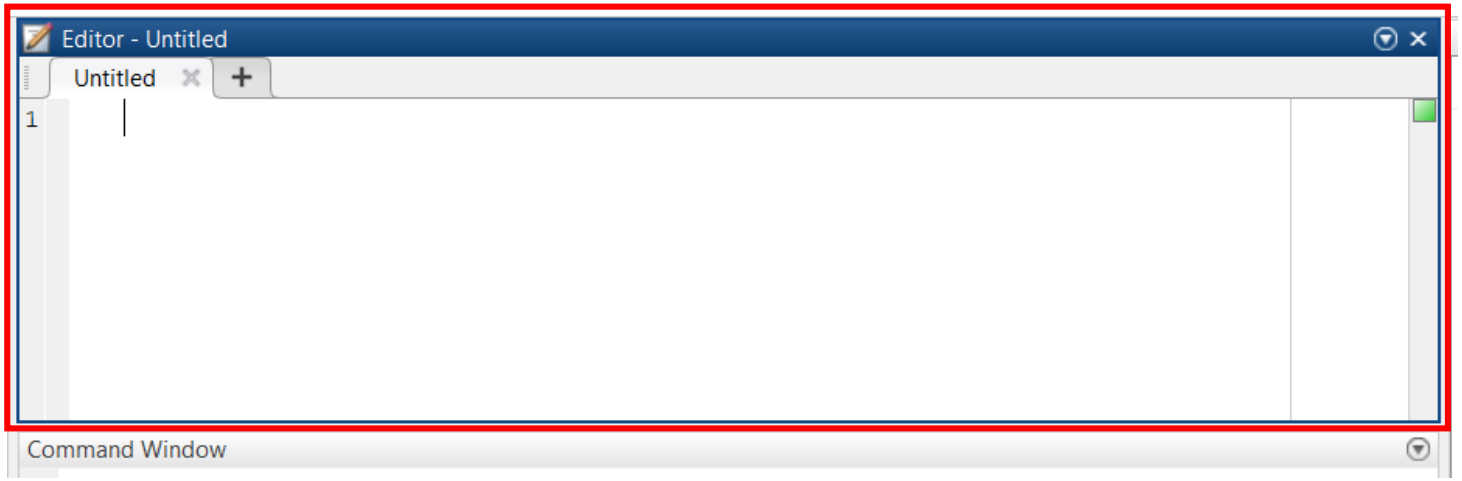
Script

Scrivere un insieme di comandi



Script:

- salvato come file .m
- **eseguito in blocco**
  nella command window
- modificabile ed eseguibile più volte

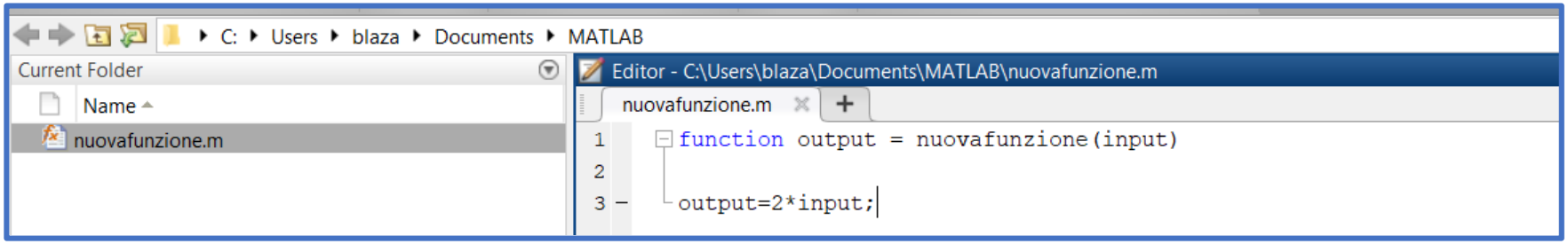


Comandi utili:

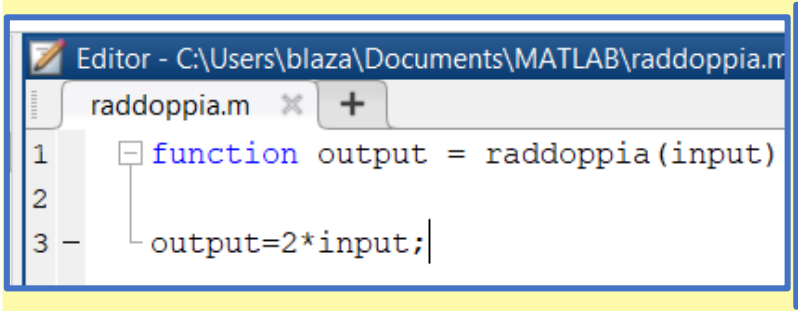
- clear
- disp
- clc
- %

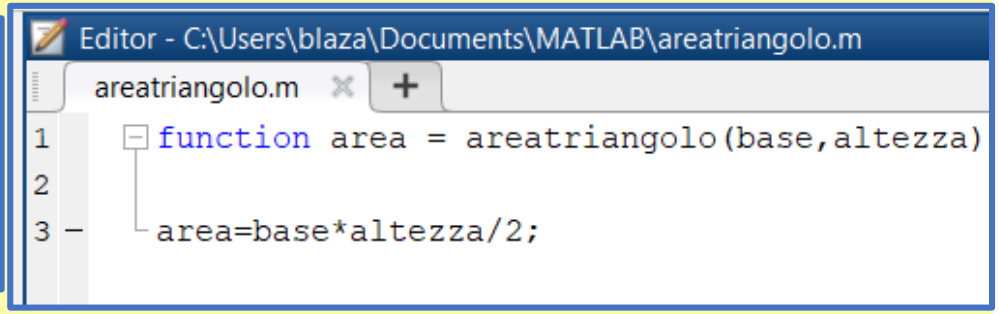
Funzioni

Creare funzioni



Esercizio: creare la funzione **raddoppia()**, che raddoppia il valore di input. Usare raddoppia(5)





Esercizio: creare la funzione **areatriangolo()**, che calcola l'area di un triangolo ricevendo come input base e altezza. Usare areatriangolo(6,4)

Attenzione:

- Alle funzioni sono passati solo i valori (**copia** distinta)
- Debug
- Visibilità (scope) delle variabili
- Global

Test logici e if

- a = 5 è un'assegnazione
- 2^2 == 4 è un **test logico**

Test logici:

== **~=** **>** **<** **<=** **>=**

Costrutto if:

if *espressione*
...
end

Costrutto if completo:

if *espressione*
...
elseif *espressione*
...
else
...
end

Esercizio: creare una funzione che calcola la radice quadrata di un numero, usando ^(1/2)

radicequadrata.m*

```
1   function rad = radicequadrata(radicando)
2
3   if radicando > 0
4       rad = radicando^(1/2);
5   elseif radicando == 0
6       rad = 0;
7   else
8       error('radicando negativo')
9   end
10
11  end
```

funzione **sqrt()**

Cicli while e for

Costrutto while:

while *espressione*
  ...
end

```
x=0;
while x <= 5
   x=x+1;
end
```

Costrutto for:

for *variabile*=***vettoreriga***
  ...
end

```
for i=[1 3 5]
   disp(i)
end
```

Esercizio: calcolare la somma dei numeri da 1 a 100

```
somma=0;
for n=1:100
   somma=somma+n;
end
disp(somma)
```

Esercizio: trovare n per cui la somma dei numeri da 1 a n è maggiore di 365

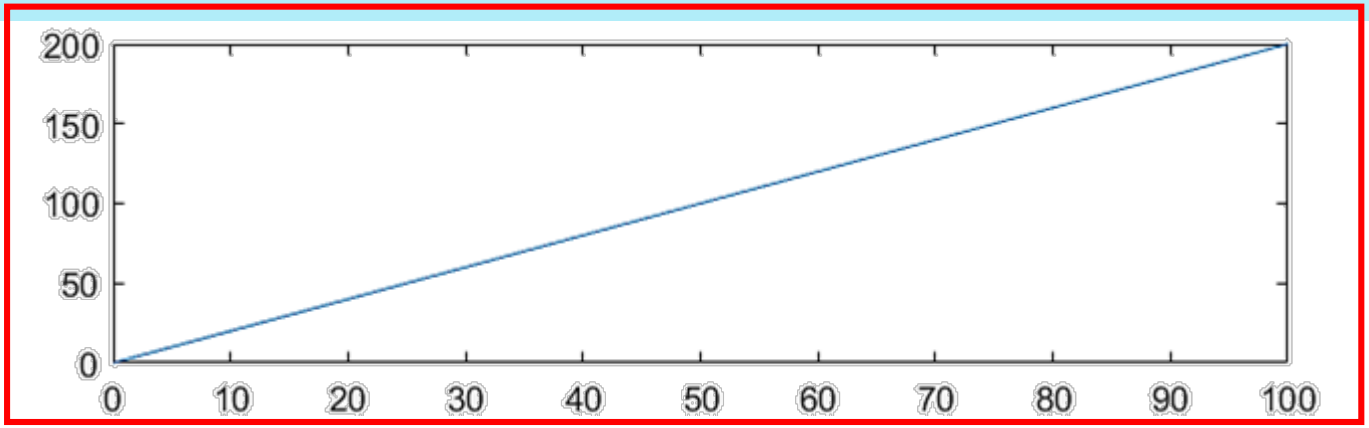
```
somma=0;
n=0;
while somma <= 365
   n=n+1;
   somma=somma+n;
end
disp(n)
```

Grafici

Esempio: calcolare raddoppia() per ogni numero da 0 a 100 con intervallo 0.5

```
>>  x=0:0.5:100;                    % creare un vettore
>>  n=length(x);                    % trovare la lunghezza del vettore
>>  y=zeros(1,n);                   % vettore y preallocato per migliorare prestazioni

>>  for i=1:n                       % ciclo for per scorrere gli n elementi di x
        elemento = x(i);            % elemento corrente i-esimo da raddoppiare
        y(i) = raddoppia(elemento); % elemento raddoppiato memorizzato in y
    end
```

>> **plot(x,y)**



Built-in functions

Matlab contiene una vasta libreria di funzioni:

- length
- size
- max
- min
- sum
- isequal
- ...

- **help**

Command Window

```
>> help length
 length   Length of vector.
    length(X) returns the length of vector X.  It is equivalent
    to MAX(SIZE(X)) for non-empty arrays and 0 for empty ones.

    See also numel.

    Reference page for length
    Other functions named length
```

Esercizi

Esercizio 1. Creare la seguente matrice 3x4 usando due cicli for innestati:
[1 2 3 4; 5 6 7 8; 9 10 11 12]

```
m=zeros(3,4);                    n=n+1;
n=0;                             m(i,j)=n;
for i=1:3                      end
   for j=1:4                 end
```

Esercizio 2. Creare una funzione che calcola il valore assoluto di un valore input

Esercizio 3. Creare una funzione che, dato un vettore in input, calcola la somma dei suoi elementi

Esercizio 4. Trovare n tale per cui la somma dei numeri da 1 a n non divisibili per 7 è maggiore di 100 000

Esercizio 5. Creare una funzione che controlla se due vettori sono identici