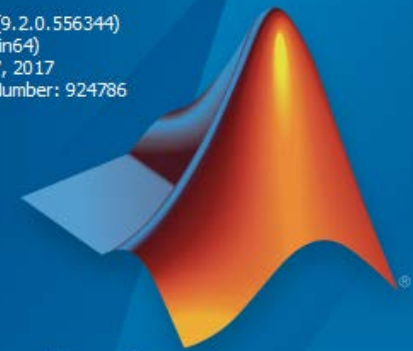




 POLITECNICO DI MILANO



R2017a (9.2.0.556344)
64-bit (win64)
March 27, 2017
License Number: 924786



MATLAB[®]

Academic License - for use in teaching, academic research, and meeting course requirements at degree granting institutions only. Not for government, commercial, or other organizational use.

© 1984-2017 The MathWorks, Inc. Protected by U.S and international patents. See mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

 MathWorks[®]

R2017a



Esercitazione 00

Introduzione a Matlab

Corso di Strumentazione e Controllo di Impianti Chimici

Prof. Davide Manca

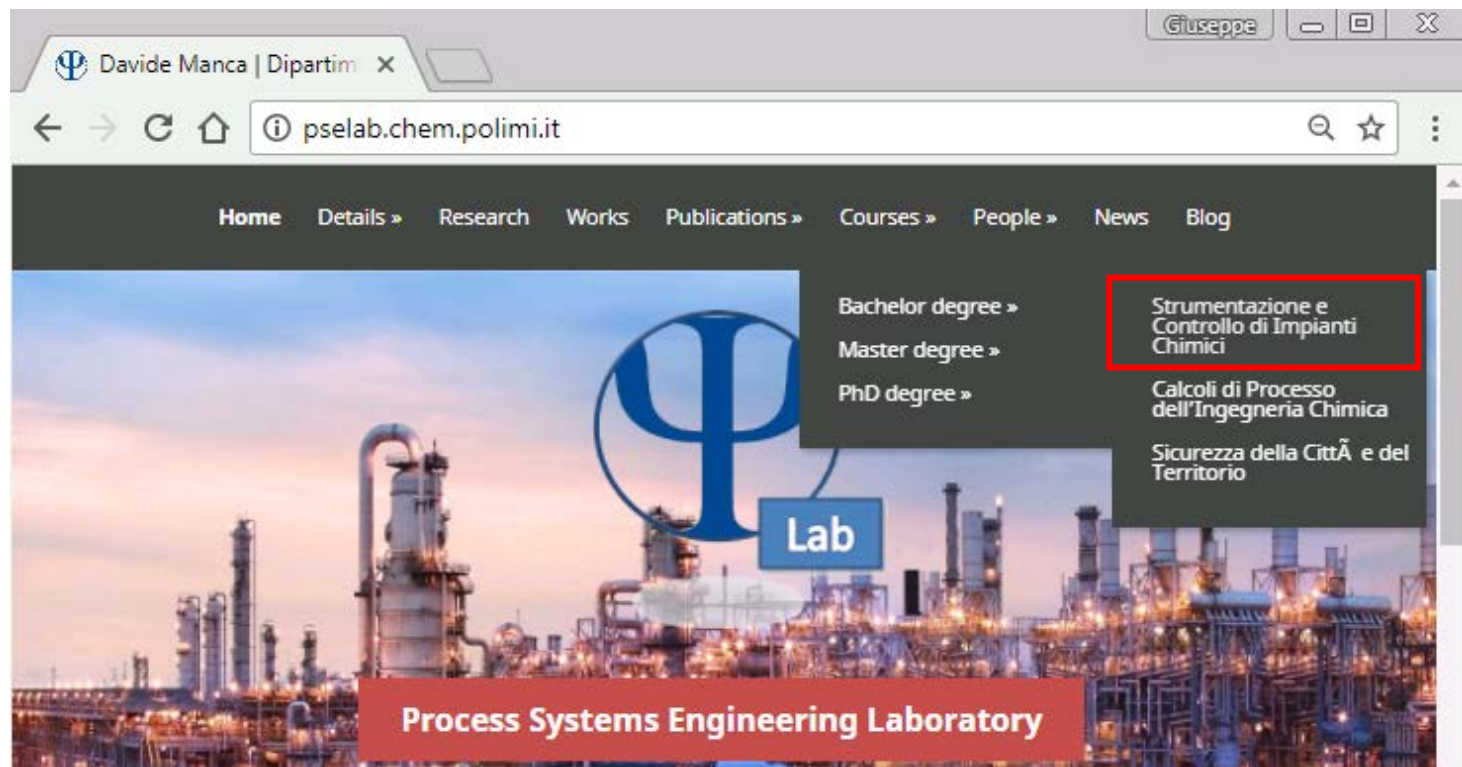
Tutor: Giuseppe Pesenti

Tutor:

Giuseppe Pesenti – giuseppe.pesenti@polimi.it

Materiale del corso:

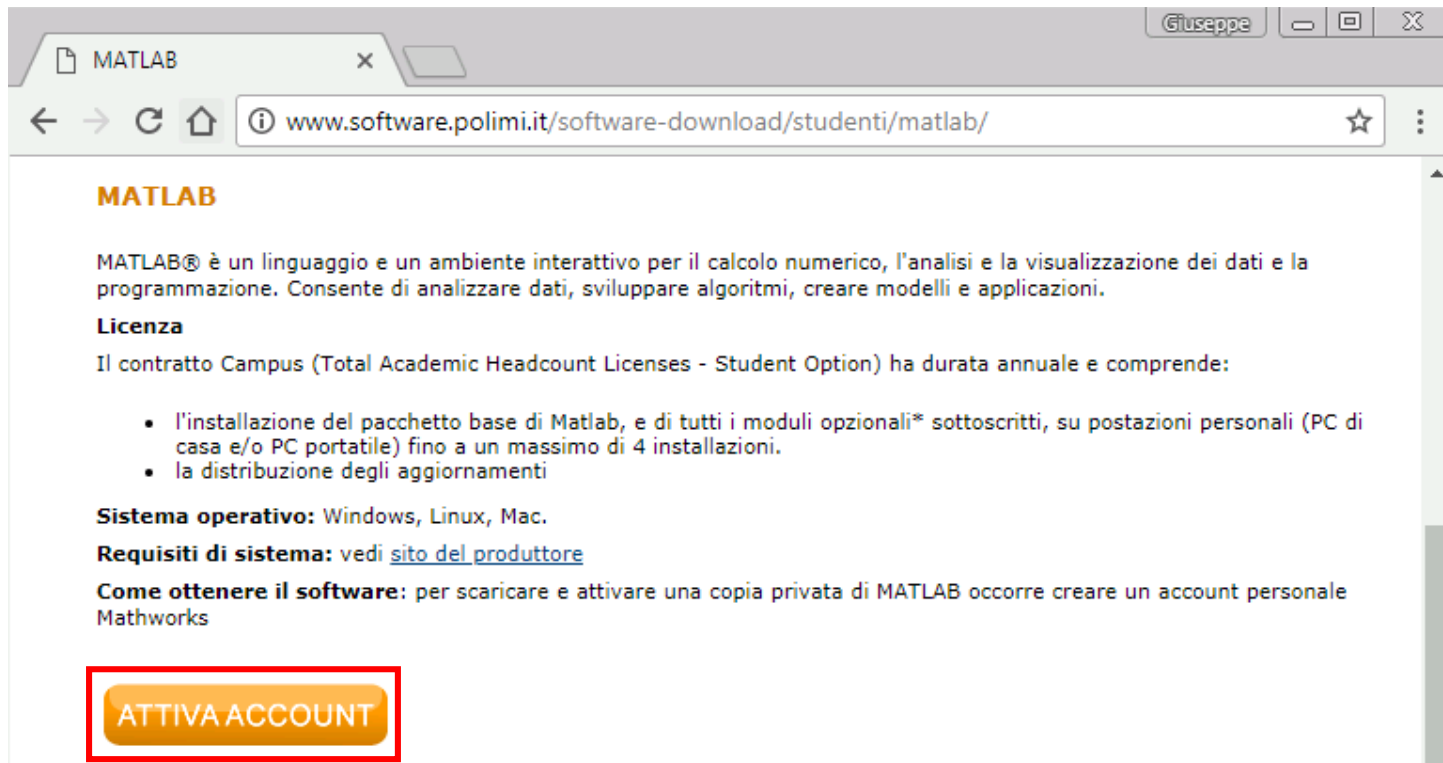
PSE-Lab – Courses – Bachelor Degree – Strumentazione e Controllo di Impianti Chimici



Matlab: 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



MATLAB

MATLAB® è un linguaggio e un ambiente interattivo per il calcolo numerico, l'analisi e la visualizzazione dei dati e la programmazione. Consente di analizzare dati, sviluppare algoritmi, creare modelli e applicazioni.

Licenza

Il contratto Campus (Total Academic Headcount Licenses - Student Option) ha durata annuale e comprende:

- l'installazione del pacchetto base di Matlab, e di tutti i moduli opzionali* sottoscritti, su postazioni personali (PC di casa e/o PC portatile) fino a un massimo di 4 installazioni.
- la distribuzione degli aggiornamenti

Sistema operativo: Windows, Linux, Mac.

Requisiti di sistema: vedi [sito del produttore](#)

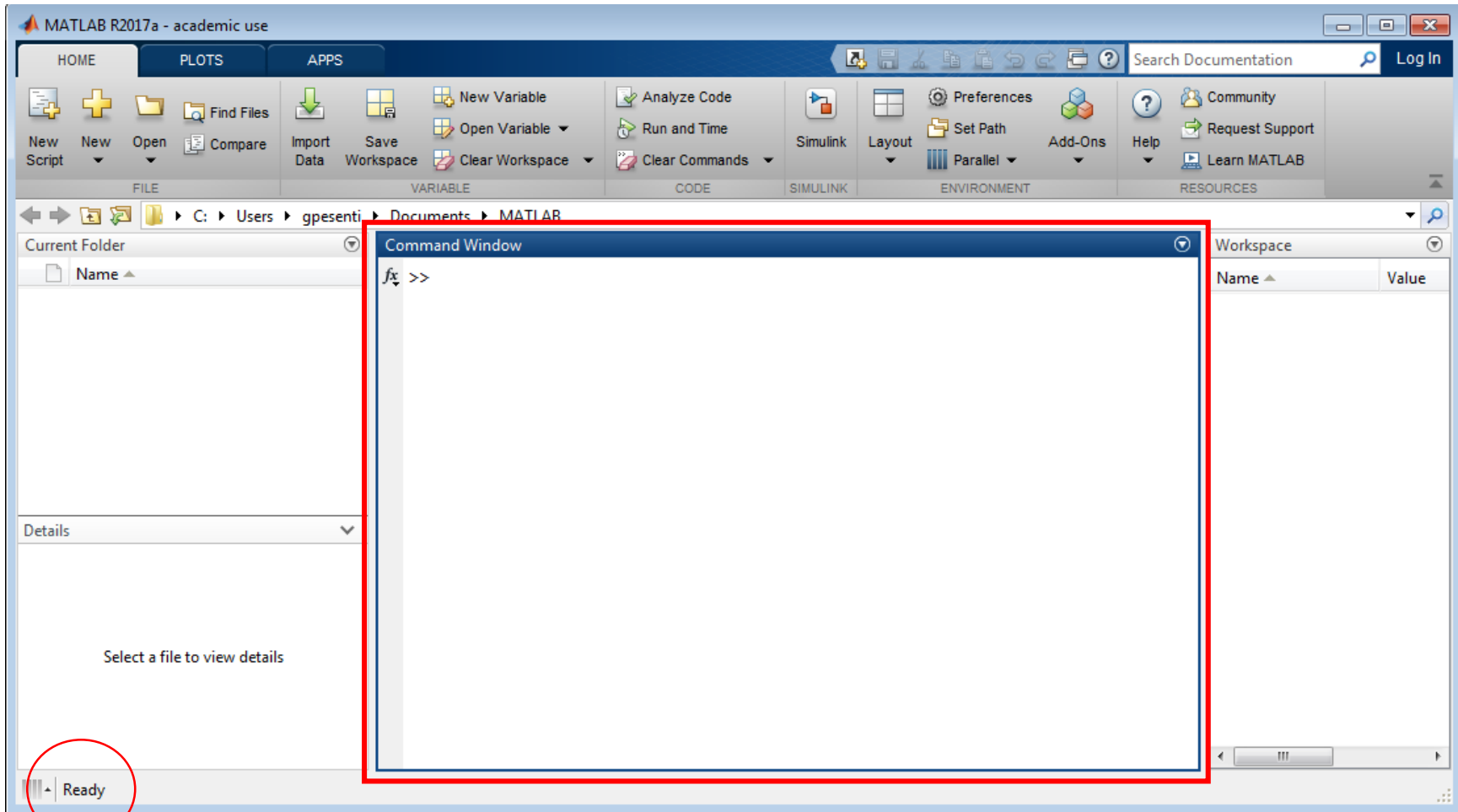
Come ottenere il software: per scaricare e attivare una copia privata di MATLAB occorre creare un account personale Mathworks

ATTIVA ACCOUNT



Command window:

Interfaccia stile riga di comando, per eseguire immediatamente dei comandi





Usare Matlab come una calcolatrice:

+ - * / ^

Esercizi: effettuare le seguenti operazioni nella command window di Matlab

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

exp() log() log10()

- $3.67*\text{exp}(8.56) = 1.9153e+04$
- $\text{log}_{10}(1e5) = 5$
- $\text{exp}(\text{log}(17)) = 17$

Nota: Matlab di default opera in doppia precisione



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

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



```
>> 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
```

Name	Value
ans	1.6000



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: ;

Workspace	
Name ▲	Value
 a	5

Workspace	
Name ▲	Value
 a	6

Workspace	
Name ▲	Value
 a	4
 b	6



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

Workspace	
Name ▲	Value
v	[1 2 5 8]

Workspace	
Name ▲	Value
m	[7 8;10 12]

2x2 double		
	1	2
1	7	8
2	10	12

Workspace	
Name ▲	Value
m	[7 8;10 12]
n	[7 8;10 12]

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 $m \times n$ 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 $n \times n$

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 5×5 `>> M = zeros(5);`
e assegnare 10 al secondo elemento della quarta riga `>> M(4,2) = 10;`

Accedere/selezionare il valore di un singolo elemento:

$m(1, 2) = 8$
riga colonna

nota: il conteggio degli elementi parte da 1

		2x2 double	
		1	2
1	7		8
2	10		12

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;
```



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

Operazione:

A'

$A + B, A - B$

$A .* B, A ./ B$

$A * B$

$\det(A)$

$A^{(-1)}$ o $\text{inv}(A)$

A / B

$A \setminus B$

Condizione:

-

le matrici/vettori A e B hanno le stesse dimensioni

le matrici/vettori A e B hanno le stesse dimensioni

le colonne di A sono tante quante le righe di B

Per matrici quadrate:

-

A è invertibile

B è invertibile, in quanto si calcola $B * \text{inv}(A)$

A è invertibile, in quanto si calcola $\text{inv}(A) * B$

Esercizio: risolvere il seguente sistema lineare

$$x - 3y + z = 2$$

$$3x - 4y + z = 0$$

$$2y - z = 1$$

$$Ax = b$$

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

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

$$\gg A = [1 \ -3 \ 1; \ 3 \ -4 \ 1; \ 0 \ 2 \ -1];$$

$$\gg b = [2 \ 0 \ 1]';$$

$$\gg x = \text{inv}(A) * b;$$

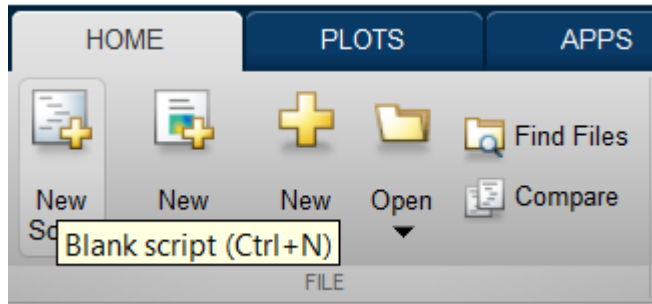
$$\gg x = A \setminus b;$$

$$x = -5$$

$$y = -8$$

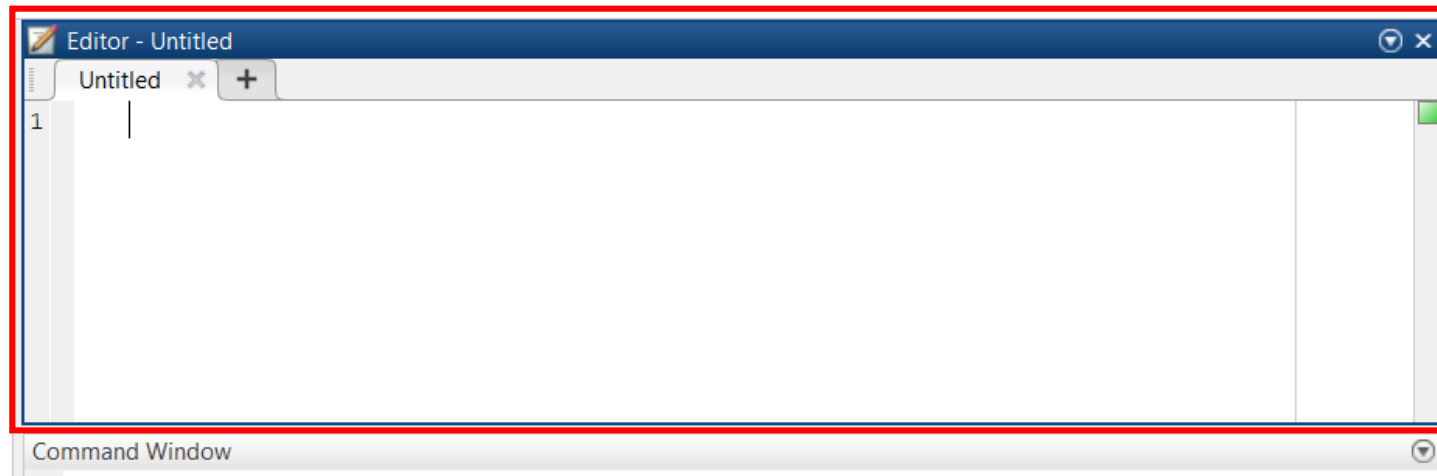
$$z = -17$$

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
- %



Creare funzioni

```
Editor - C:\Users\blaza\Documents\MATLAB\nuovafunzione.m
nuovafunzione.m x +
1 function output = nuovafunzione(input)
2
3 output=2*input;
```

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

```
Editor - C:\Users\blaza\Documents\MATLAB\raddoppia.m
raddoppia.m x +
1 function output = raddoppia(input)
2
3 output=2*input;
```

```
Editor - C:\Users\blaza\Documents\MATLAB\areatriangolo.m
areatriangolo.m x +
1 function area = areatriangolo(base,altezza)
2
3 area=base*altezza/2;
```

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



- $a = 5$ 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* x +
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()**



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)
```

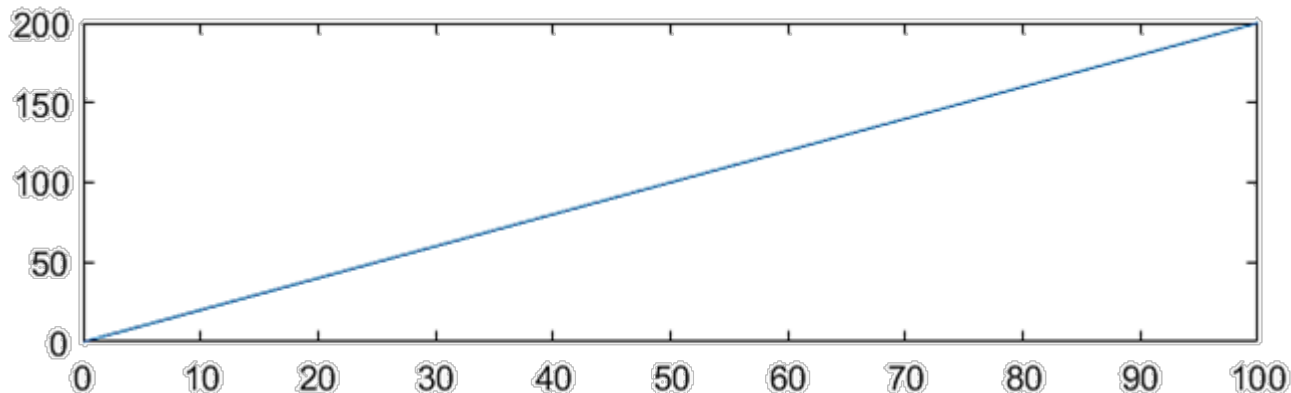



Esempio: calcolare raddoppia() per ogni numero da 0 a 100 con intervallo 0.5

```
>> 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)
```





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
```

**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