Comandi utili

Assegnamento

```
a = 5;
b = 3
```

La differenza tra `a = 5` e `a = 5;` è che il punto e virgola non stampa il risultato a video.

Console

- `disp('Hello, World!')` stampa a video `Hello, World!`
- `disp(a)` stampa a video il valore di `a`
- `clear` pulisce tutte le variabili dalla memoria
- `clc` pulisce la console

Operazioni

- `+` somma
- `-` sottrazione
- `*` moltiplicazione
- `.*` moltiplicazione elemento per elemento
- `/` divisione
- `./` divisione elemento per elemento
- `^` elevamento a potenza
- `sqrt()` radice quadrata
- `sin()`, `cos()`, `tan()` seno, coseno, tangente
- `log()` logaritmo naturale
- `log10()` logaritmo in base 10
- `exp()` esponenziale
- `abs()` valore assoluto
- `round()` arrotondamento
- `floor()` arrotondamento per difetto
- `ceil()` arrotondamento per eccesso
- `mod()` resto della divisione
- `rand()` numero casuale tra 0 e 1

Array e operazioni su array

- `a = [1, 2, 3]` definisce un array
- `a(1)` accede al primo elemento dell'array
- `a(1:2)` accede ai primi due elementi dell'array
- `a(1:2) = [4, 5]` modifica i primi due elementi dell'array
- `length(a)` restituisce la lunghezza dell'array
- `size(a)` restituisce la dimensione dell'array
- `a'` trasposta dell'array
- `a + 1` somma 1 a tutti gli elementi dell'array
- `a + b` somma due array

  > *(Così per tutte le operazioni)*

- `zeros(3)` array di 3 elementi tutti a 0

- `ones(3)` array di 3 elementi tutti a 1
- `rand(3)` array di 3 elementi casuali tra 0 e 1
- `a = [1, 2, 3]; b = [4, 5, 6]; c = [a, b]` concatenazione di array
- `1:5` array da 1 a 5
- `linspace(0, 1, 5)` array di 5 elementi tra 0 e 1

Funzioni

```
function y = f(x)
    y = x^2;
end
```

- `function` definisce una funzione
- `y = f(x)` definisce la funzione `f` che restituisce `y` in base a `x`
- `end` chiude la definizione della funzione
- `f(5)` chiama la funzione `f` con `x = 5`, che restituirà `25` su `y`

Grafici

```
x = linspace(0, 2*pi, 100);
y = sin(x);
plot(x, y);
```

- `linspace(0, 2*pi, 100)` definisce un array di 100 elementi tra 0 e 2π
- `sin(x)` calcola il seno di tutti gli elementi di `x`
- `plot(x, y)` disegna il grafico di `y` in funzione di `x`

Comandi utili

- `figure` crea una nuova figura
- `plot(x, y, 'r-')` disegna il grafico di `y` in funzione di `x` con linea rossa continua
- `semilogy(x, y)` disegna un grafico con scala logaritmica sull'asse y
- `hold on` mantiene il grafico attuale e ne disegna uno nuovo sopra
- `xlabel('x')` etichetta l'asse x
- `ylabel('y')` etichetta l'asse y
- `title('Titolo')` titolo del grafico
- `legend('sin(x)')` legenda del grafico
- `grid on` mostra la griglia
- `subplot(2, 1, 1)` crea un grafico in una griglia 2x1 nella prima posizione, `subplot(2, 1, 2)` nella seconda
- `plot(x, y, 'r-', x, z, 'b--')` disegna due grafici nello stesso grafico

Cicli

```
for i = 1:5
    disp(i);
end
```

- `for` inizia un ciclo
- `i = 1:5` definisce un ciclo da 1 a 5
- `disp(i)` stampa a video il valore di `i`

- `end` chiude il ciclo

```
iterations = 1:100;
for i = iterations
    disp(i);
end
```

In questo caso `iterations` è un array di 100 elementi da 1 a 100 e il ciclo `for` scorre tutti gli elementi dell'array.

Condizioni

```
if a > 5
    disp('a è maggiore di 5');
elseif a == 5
    disp('a è uguale a 5');
else
    disp('a è minore di 5');
end
```

Funzioni matematiche

```
f = @(x) x^2;
f(5)
```

- `f = @(x) x^2` definisce una funzione anonima `f` che restituisce `x^2`
- `f(5)` chiama la funzione `f` con `x = 5`, che restituirà `25`

```
f = @(x) x^2;
x = linspace(0, 10, 100);
y = f(x);
plot(x, y);
```

- `x = linspace(0, 10, 100)` definisce un array di 100 elementi tra 0 e 10
- `f(x)` calcola `f` per tutti gli elementi di `x`
- `plot(x, y)` disegna il grafico di `y` in funzione di `x`