Corso Pratico Python

[Modulo 1- 2026]

2. Linguaggio Python

2.1. Variabili

2.1.1. Visibilità, Interprete

2.1.2. Classi, Moduli

2.1.3. Liste, Array, Dizionari, Tuple.

2.1.4. Argomenti, Kwargs

Richiedente: Dott. Roberto Felici [CNR-ISM] • Insegnante: Dott. Scigè J. Liu [INAF-IAPS]

2.1.4. Argomenti, KWargs,

Posizionali e Nominali [+Argomenti Variabili (*args)]

Posizionali:
•Il valore è assegnato in base all'ordine (es. func(a, b)).
Nominali (Keyword):
•Il valore è assegnato esplicitamente al
•nome del parametro (es. func(a=1, b=2)).
Default:
•È possibile assegnare valori predefiniti
•che vengono usati se l'argomento è omesso.

```
def saluta(nome, messaggio="Buongiorno"):
    return f"{messaggio}, {nome}!"

print(saluta("Roberto")) # Usa il default
```

**args → Argomenti Posizionali Variabili*
•**Flessibilità:** Permette a una funzione di accettare un numero qualsiasi di argomenti posizionali.
•**Struttura:** All'interno della funzione, args è trattato come una **Tupla**.
•**Sintassi:** Si usa l'asterisco * prima del nome del parametro.

```
def somma_tutti(*numeri):
    # 'numeri' è una tupla
    return sum(numeri)

print(somma_tutti(10, 20, 30, 40)) # Output: 100
```

2.1.4. Argomenti, KWargs,

Posizionali e Nominali [+Argomenti Variabili (*args)]

**args → Argomenti Posizionali Variabili*

- **Flessibilità:** Permette a una funzione di accettare un numero qualsiasi di argomenti posizionali.
- **Struttura:** All'interno della funzione, args è trattato come una **Tupla**.
- **Sintassi:** Si usa l'asterisco * prima del nome del parametro.

```python
def somma_tutti(*numeri):
    # 'numeri' è una tupla
    return sum(numeri)

print(somma_tutti(10, 20, 30, 40)) # Output: 100
```

***Ordine**: Vengono catturati tutti gli argomenti posizionali non assegnati ad altri parametri.*

***Iterabilità**: Essendo una tupla, è possibile scorrere gli elementi con un ciclo for.*

***Unpacking**: L'operatore * può essere usato anche per "spacchettare" una lista/tupla esistente in una chiamata a funzione.*

```python
dati = [1, 2, 3]
# Trasforma la lista in argomenti separati
risultato = somma_tutti(*dati)
```

2.1.4. Argomenti, KWargs [** kwargs]

** **kwargs Argomenti Nominali Variabili**

- **Mappatura:**
- Permette di passare argomenti identificati da una chiave.
- **Struttura:**
- All'interno della funzione, kwargs è come un **Dizionario**.
- **Sintassi:**
- Si usano due asterischi ** prima del nome del parametro.

```
def crea_profilo(**info):
    # 'info' è un dizionario (chiave-valore)
    for chiave, valore in info.items():
        print(f"{chiave}: {valore}")

crea_profilo(nome="Scigè", ruolo="Docente", anno=2026)
```

- **Configurazioni:** Ideale per passare opzioni o parametri di configurazione opzionali.
- **Estendibilità:** Permette di aggiungere nuovi parametri alla chiamata senza rompere la compatibilità della funzione.
- **Accesso Rapido:** Utilizza la logica di accesso istantaneo tipica dei Dizionari.

```
def config_grafico(**opzioni):
    colore = opzioni.get("colore", "blu")
    spessore = opzioni.get("spessore", 1)
```

2.1.4. Argomenti, KWargs,

L'Ordine dei Parametri

Gerarchia Obbligatoria nella Definizione

Per evitare errori di sintassi, Python impone un ordine preciso quando si mescolano diversi tipi di argomenti:

1. **Argomenti Standard** (posizionali).
2. ***args** (posizionali variabili).
3. **Argomenti con Default**.
4. ****kwargs** (nominali variabili).

```
def funzione_complessa(p1, *args, p2="Default", **kwargs):
    print(p1, args, p2, kwargs)
```