Guida Completa alla Formattazione dell'Output in Python

Utilizzo avanzato della funzione `print()`

Documentazione Tecnica Python

20 ottobre 2026

La funzione `print()` in Python offre un'ampia gamma di opzioni per la formattazione dell'output. Questa guida raccoglie in modo sistematico tutte le modalità disponibili: dai parametri nativi della funzione alle `f-string`, fino alle sequenze di escape ANSI.

1 Parametri Nativi della Funzione `print()`

La sintassi completa della funzione `print()` include i seguenti parametri con valore predefinito:

```
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
```

1.1 Esempi d'uso dei parametri

```
# 'sep': definisce il separatore tra gli argomenti (default: spazio)
print("02", "10", "2026", sep="-")
# Output: 02-10-2026

# 'end': definisce il carattere finale (default: \n)
print("Elaborazione in corso", end="... ")
print("Fatto!")
# Output: Elaborazione in corso... Fatto!

# 'file': reindirizza l'output su file o stream
with open("log.txt", "w", encoding="utf-8") as f:
    print("Messaggio salvato su file log", file=f)

# 'flush': forza lo svuotamento immediato del buffer di output
print("Caricamento in corso...", flush=True)
```

2 Formattazione con F-String (Python 3.6+)

Le **f-string** costituiscono il metodo moderno, più efficiente e leggibile per formattare testo e variabili.

2.1 Sintassi di Base e Operazioni

```
nome = "Alice"
eta = 30

# Inserimento semplice di variabili ed espressioni
print(f"Ciao {nome}, l'anno prossimo avrai {eta + 1} anni.")

# Conversione di rappresentazione: !r (repr), !s (str), !a (ascii)
```

```python
print(f"Stringa con virgolette: {nome!r}") # Output: 'Alice'

# Debugging rapido con '=' (stampa nome variabile e valore)
print(f"{nome=}, {eta=}") # Output: nome='Alice', eta=30
```

2.2 Allineamento e Spaziatura

```python
testo = "Python"

print(f"'{testo:<10}'")  # Allineato a sinistra (larghezza 10)
print(f"'{testo:>10}'")  # Allineato a destra
print(f"'{testo:^10}'")  # Centrato
print(f"'{testo:*^10}'") # Centrato con riempimento (*)
```

2.3 Numeri Decimali e Virgola Mobile

```python
pi = 3.14159265

print(f"{pi:.2f}")        # Arrotondato a 2 cifre decimali (3.14)
print(f"{12.345:>10.2f}") # Larghezza totale 10, 2 cifre decimali, allineato a
    destra
print(f"{0.25:.1%}")      # Formattazione percentuale (25.0%)
```

2.4 Numeri Interi, Separatori e Basi Numeriche

```python
numero = 1000000

print(f"{numero:_}")     # Separatore migliaia con underscore (1_000_000)
print(f"{numero:,}")     # Separatore migliaia con virgola (1,000,000)
print(f"{42:05d}")       # Padding con zeri (00042)

# Conversione di base
valore = 255
print(f"Binario: {valore:b}")     # 11111111
print(f"Ottale: {valore:o}")      # 377
print(f"Esadecimale: {valore:x}") # ff (o :X per FF)
```

2.5 Date e Orari

```python
import datetime
ora_attuale = datetime.datetime.now()

print(f"{ora_attuale:%d/%m/%Y %H:%M}") # es. 20/10/2026 08:55
```

3 Metodo `.format()`

Utilizzato per la compatibilità con versioni precedenti a Python 3.6 o quando la stringa di formato è separata dai dati.

```python
# Per posizione
print("Ciao {}, hai {} messaggi.".format("Luca", 5))

# Per indice esplicito
```

```
print("{1} e piu alto di {0}.".format("Marco", "Paolo"))

# Per nome dell'argomento
print("Utente: {username}, Ruolo: {role}".format(username="admin", role="Root"))

# Unpacking di dizionari (**)
dati = {"prodotto": "Computer", "prezzo": 899.99}
print("Articolo: {prodotto} - Prezzo: {prezzo:.2f}EUR".format(**dati))
```

4 Operatore % (Sintassi Legacy)

```
nome = "Giulia"
eta = 22
altezza = 1.756

# %s = stringa, %d = intero, %f = float
print("Nome: %s | Eta: %d | Altezza: %.2f m" % (nome, eta, altezza))
```

5 Sequenze di Escape e Codici ANSI

5.1 Sequenze di Escape Standard

- `\n` : Nuova riga
- `\t` : Tabulazione
- `\\` : Barra rovesciata (\)
- `\"` / `\'` : Virgolette / Apostrofo

5.2 Colori e Stili di Testo nel Terminale (ANSI)

```
GRASSETTO = "\033[1m"
SOTTOLINEATO = "\033[4m"

ROSSO = "\033[31m"
VERDE = "\033[32m"
GIALLO = "\033[33m"
RESET = "\033[0m"

print(f"{GRASSETTO}{VERDE}Operazione completata con successo!{RESET}")
print(f"{ROSSO}Attenzione: errore di sistema.{RESET}")
```

6 Sintesi dei Formattatori nei Placeholder

Specificatore	Descrizione	Esempio di Output
`:<N`	Allinea a sinistra su N caratteri	`f"{'a':<5}"` $\rightarrow$ `'a    '`
`:>N`	Allinea a destra su N caratteri	`f"{'a':>5}"` $\rightarrow$ `'    a'`
`:^N`	Centra il testo su N caratteri	`f"{'a':^5}"` $\rightarrow$ `'  a  '`
`:0N`	Padding con zeri iniziali	`f"{5:03d}"` $\rightarrow$ `'005'`
`:.Nf`	Precisione float a N decimali	`f"{3.14159:.2f}"` $\rightarrow$ `'3.14'`
`:,`	Separatore delle migliaia (virgola)	`f"{1000000:,}"` $\rightarrow$ `'1,000,000'`
`:b` / `:x`	Conversione in Binario / Esadecimale	`f"{10:b}"` $\rightarrow$ `'1010'`

Tabella 1: Principali opzioni del mini-linguaggio di formattazione.