Introduzione a Python

Francesco Gobbi

I.I.S.S. Galileo Galilei, Ostiglia

23 settembre 2025

Cos'è un oggetto in Python?

- In Python ogni valore è un oggetto con:
  - **Tipo** (class), es. `int`, `str`, `list`, ...
  - **Identità** unica in memoria (`id(obj)`)
  - **Valore** o contenuto
- Gli oggetti supportano metodi e attributi:
  - Esempio: stringhe hanno `.upper()`, liste hanno `.append()`...
- Anche funzioni, classi e moduli sono oggetti

```
1 # Diversi oggetti
2 a = 10                    # int
3 b = "ciao"               # str
4 c = [1,2,3]               # list
5 print(type(a), id(a))
6 print(type(b), id(b))
7 print(type(c), id(c))
```

Variabili come etichette

- Una **variabile** non contiene direttamente il valore, ma un riferimento all'oggetto in memoria
- Assegnamento crea o aggiorna l'etichetta:

```
1 a = 100
2 # id prima
3 print(id(a))
4 a = 200
5 # id dopo (nuovo oggetto)
6 print(id(a))
```

Cambiando il valore, Python crea un nuovo oggetto e sposta l'etichetta.

Alias e condivisione

- Più variabili possono puntare allo **stesso** oggetto (alias)
- Attenzione alle modifiche: mutazione vs riassegnamento

```
a = [1,2]
b = a                    # b alias di a
print(a is b)            # True
b.append(3)              # muta l'oggetto condiviso
print(a, b)              # [1,2,3] [1,2,3]

# Riassegnamento di b
b = [9,9]
print(a, b)              # [1,2,3] [9,9]
```

Esempio con dizionari mutabili:

```
d1 = {'x':1}
d2 = d1
d2['y'] = 2
print(d1)                # {'x':1,'y':2}
```

Identità (`is`) vs uguaglianza (==)

- `is` verifica se due variabili puntano allo **stesso** oggetto
- == verifica se il contenuto è **equivalente**

```
x = [1,2]
y = [1,2]
print(x == y)  # True, stesso contenuto
print(x is y)  # False, diversi oggetti

s1 = "hello"
s2 = "hello"
print(s1 == s2, s1 is s2)  # True, True (interning)
```

Mutabilità vs immutabilità

- Oggetti **mutabili**: liste, dizionari, set → si modificano internamente
- Oggetti **immutabili**: int, float, str, tuple → ogni modifica genera un *nuovo* oggetto

```
1  # Stringhe (immutabili)
2  s = "pippo"
3  t = s.replace('i', 'a')
4  print(id(s), id(t))  # id diversi
5
6  # Tuple (immutabili)
7  tup = (1,2)
8  # tup[0] = 0  # Errore
9
10 # Liste (mutabili)
11 l = [1,2]
12 l.append(3)
13 print(id(l))          # id invariato
```

Riassegnamento di variabile

- L'operazione = cambia l'etichetta su un nuovo oggetto
- Non tocca gli altri alias

```
1 a = 5
2 b = a
3 b = 10                # b punta ora a 10
4 print(a, b)           # 5 10
```

Riassegnamento non è mutazione.

Funzioni e classi sono oggetti

- Anche funzioni e classi hanno tipo e id

```
1  def saluta():
2      print("Ciao!")
3
4  sal = saluta              # alias funzione
5  print(type(sal), id(sal))
6  sal()                     # chiama saluta()
7
8  class Persona:
9      pass
10 P = Persona               # alias classe
11 print(type(P), id(P))
```

Riepilogo

- In Python **tutto è un oggetto**
- Le variabili sono etichette che referenziano oggetti
- `id()` mostra identità, `type()` il tipo
- Distinguere sempre:
  - **Mutazione** (modifica oggetto) vs **Riassegnamento** (nuovo oggetto)
  - `is` vs `==`