10 – Filter Function

La funzione `filter()` in Python serve per **selezionare elementi da un iterabile** in base a una condizione. Restituisce solo gli elementi per i quali la funzione di controllo ritorna `True`. È utile quando vuoi **filtrare** dati senza scrivere cicli `for` complessi.

Sintassi

```
filter(funzione, iterabile)
```

- **funzione**: una funzione che prende un elemento e restituisce `True` o `False`.
- **iterabile**: lista, tupla o qualsiasi oggetto iterabile.
- Restituisce un oggetto `filter`, che può essere convertito in lista, tupla o set.

Esempio base

Se vogliamo selezionare solo i numeri pari da una lista:

```
numeri = [1, 2, 3, 4, 5, 6]
pari = filter(lambda x: x % 2 == 0, numeri)
print(list(pari))  # [2, 4, 6]
```

Usare filter con una funzione definita

```
def maggiore_di_tre(x):
    return x > 3

numeri = [1, 2, 3, 4, 5]
risultato = filter(maggiore_di_tre, numeri)
print(list(risultato))  # [4, 5]
```

Filter su più tipi di iterabili

`filter()` funziona su qualsiasi iterabile, inclusi set o tuple:

```
valori = (10, 15, 20, 25)
filtrati = filter(lambda x: x % 10 == 0, valori)
print(tuple(filtrati))  # (10, 20)
```

Vantaggi

- Permette di filtrare elementi in modo conciso.
- Funziona bene con funzioni lambda per controlli rapidi.
- Mantiene il codice leggibile rispetto a un ciclo con `if`.

Nota

Come `map()`, `filter()` restituisce un oggetto iterabile; per ottenere una lista o una tupla bisogna convertire esplicitamente con `list()` o `tuple()`.

Vuoi procedere con il **paragrafo 11 – List Comprehensions**?