17 – Arrays

In Python, un **array** è una struttura dati simile a una lista, ma progettata per contenere **elementi dello stesso tipo** in modo più efficiente, sia in termini di memoria sia di prestazioni. Gli array sono particolarmente utili quando si lavora con **grandi quantità di dati numerici**.

Python offre array tramite il modulo `array` della libreria standard, oppure tramite librerie esterne come **NumPy** per array avanzati multidimensionali.

Creazione di un array con il modulo `array`

```
from array import array

# array di interi ('i')
numeri = array('i', [1, 2, 3, 4, 5])
print(numeri)  # array('i', [1, 2, 3, 4, 5])
```

- Il primo parametro `'i'` indica il tipo degli elementi (`'i'` = interi, `'f'` = float, ecc.).
- Tutti gli elementi devono essere dello stesso tipo.

Accesso e modifica degli elementi

Come le liste, gli array supportano indice e slicing:

```
print(numeri[0])   # 1
numeri[1] = 20
print(numeri)      # array('i', [1, 20, 3, 4, 5])
```

Aggiungere elementi

```
numeri.append(6)      # aggiunge alla fine
numeri.insert(2, 15) # inserisce alla posizione 2
print(numeri)         # array('i', [1, 20, 15, 3, 4, 5, 6])
```

Rimuovere elementi

```
numeri.remove(20)   # rimuove il primo valore uguale a 20
ultimo = numeri.pop()  # rimuove l'ultimo elemento
print(numeri)
```

Vantaggi degli array

- Più **efficienti in memoria** rispetto alle liste per grandi quantità di dati numerici.
- Operazioni matematiche più rapide con librerie come **NumPy**.
- Adatti per algoritmi numerici o scientifici.

Nota

- Per piccoli insiemi di dati o dati eterogenei, le liste sono più pratiche.
- Gli array standard sono limitati ai tipi di dati primitivi, mentre NumPy offre array multidimensionali e funzioni avanzate.

Vuoi procedere con il **paragrafo 18 – Sets**?