**📘 Set Operations – Demonstration & Explanation**

**✅ 1. Create and Modify a Set**

sec\_data = {1, 2, 3, 4, 6}

sec\_data.add(11) # Adds element 11

sec\_data.discard(1) # Removes element 1 (no error if not found)

print(sec\_data)

.add(x) adds an element to the set.  
.discard(x) removes the element if present; no error if absent.

**✅ 2. Set Operations: Union, Intersection, Difference, Symmetric Difference**

set\_1 = {x for x in range(1, 5)} # {1, 2, 3, 4}

set\_2 = {x for x in range(1, 6) if x % 2 == 0} # {2, 4}

print(f"Set 1: {set\_1} | Set 2: {set\_2}")

union\_set = set\_1.union(set\_2)

intersection\_set = set\_1.intersection(set\_2)

difference\_set = set\_1.difference(set\_2)

sym\_diff = set\_1.symmetric\_difference(set\_2)

print(f"Union: {union\_set}")

print(f"Intersection: {intersection\_set}")

print(f"Difference: {difference\_set}")

print(f"Symmetric Difference: {sym\_diff}")

* **Union**: All unique elements from both sets.
* **Intersection**: Common elements.
* **Difference**: Elements in set\_1 not in set\_2.
* **Symmetric Difference**: Elements in one set but not both.

**✅ 3. Set Comprehension – Squares**

set\_data = {x\*\*2 for x in range(1, 11)}

print(sorted(set\_data)) # [1, 4, 9, ..., 100]

Uses **set comprehension** to create a set of perfect squares.

**✅ 4. Remove Duplicates from a List using Set**

set\_data = {1, 1, 1, 1, 2, 3, 3, 2}

print(set(set\_data)) # Output: {1, 2, 3}

Sets **automatically eliminate duplicates**.

**✅ 5. Subset & Superset Checks**

set\_data = {x for x in range(5)} # {0, 1, 2, 3, 4}

set\_data2 = {y for y in range(3)} # {0, 1, 2}

if set\_data2.issubset(set\_data):

print("Set 2 is a subset of Set 1")

if set\_data.issuperset(set\_data2):

print("Set 1 is a superset of Set 2")

* .issubset() checks if one set is contained within another.
* .issuperset() checks if it contains another set.

**✅ 6. Frozenset**

frozen = frozenset(set\_data)

print(f"Frozenset: {frozen}")

A **frozenset** is an **immutable set** — useful as dictionary keys or for fixed collections.

**✅ 7. Convert Set to List, Modify, and Back to Set**

set\_data3 = {x for x in range(5)} # {0, 1, 2, 3, 4}

list\_data = list(set\_data3)

list\_data.append(6)

set\_data2 = frozenset(list\_data)

print(f"Modified set: {set\_data2}")

Converts set to list for modification (append), then back to frozenset.

**✅ 8. Iterate Over a Set**

set\_2 = {x for x in range(1, 5)}

for x in set\_2:

print(x)

**For loop** iteration over elements in a set (unordered).

**✅ 9. Remove Elements Until Set is Empty**

set\_2 = {x for x in range(1, 5)}

for x in set\_2.copy():

set\_2.discard(x)

print(f"Set after removal: {set\_2}")

.copy() used to avoid modifying the set during iteration.

**✅ 10. Set Copy and Modification**

set\_2 = {x for x in range(1, 5)}

set\_2\_copy = set\_2.copy()

set\_2\_copy.add(11)

print(f"Original Set: {set\_2}")

print(f"Modified Copy: {set\_2\_copy}")

Demonstrates shallow copying and modification of sets.

**✅ 11. Update with Symmetric Difference**

set\_ = {1, 2, 3, 4}

set\_2 = {1, 2, 3, 4}

set\_.add(12)

sym\_diff = set\_.symmetric\_difference(set\_2)

print(f"Symmetric Difference: {sym\_diff}")

Returns elements **only in one of the sets**.

**✅ 12. Membership Testing in a Set**

set\_ = {1, 2, 3, 4}

print("Is 1 in the set?", 1 in set\_)

print("Is 2 in the set?", 2 in set\_)

print("Is 50 in the set?", 50 in set\_)

Fast **O(1)** membership test using in.

**🧾 Summary of Common Set Methods**

| **Method** | **Description** |
| --- | --- |
| add(elem) | Adds an element |
| discard(elem) | Removes an element if present |
| remove(elem) | Same as discard but raises error if missing |
| union(other) | All elements in either set |
| intersection(other) | Elements in both sets |
| difference(other) | Elements in current but not in other |
| symmetric\_difference() | Elements in one set but not both |
| issubset(other) | Checks if a set is contained in another |
| issuperset(other) | Checks if it contains another set |
| copy() | Returns a shallow copy |
| clear() | Removes all elements |