20/02/26 ॥ गणपती बाप्पा मोरया ॥

3. set (HashSet)

↳ It is a collection that stores unique elements only.

Properties of Set:
1. No duplicates
2. No indexing
3. Unordered
4. Very fast lookup
5. Mutable

Use a Set when you require:
1. uniqueness
2. fast membership check
3. removing duplicates.

```
s = set()          # creating an empty set
s = {1,2,3}        # creating set with values
```

Imp: `s = {}` # this creates a dictionary, not a set

Example: Duplicates removed automatically.

```
s = {1,2,2,3,1}
print(s)
```

Output: {1,2,3}

- Add element

```
s.add(5)          #adds element
                  # does nothing if
                    already present
```

Output : {1, 2, 3, 5}

- Remove element

```
s.remove(2)       # if element is not
                    present ⇒ ERROR
```

instead use,

```
s.discard(2)      # no error if element
                    is missing.
```

Output : {1, 3, 5}

- Membership check

```
x in s
```

Example :

```
if 5 in s:
    print("Present")
```

O(1) average case

- Loop through set

```
for val in s:
    print(val)
```

→

usage of sets in DSA.

1. Duplicate Detection

```
arr = [1,2,3,4,2]

seen = set()
for num in arr:
    if num in seen:
        print("Duplicate found")
    seen.add(num)

print("Non-duplicates:", seen)
```

o/p: Duplicate found
Non-duplicates: {1,2,3}

2. Remove Duplicates

```
arr = [1,1,2,3,2]
unique = list(set(arr))

print(unique)
```

o/p: [1,2,3]

3. Visited Array in Graphs

```
visited = set()

def dfs(node):
    if node in visited:
        return
    visited.add(node)

print(visited)
```

DFS logic

4. Cycle Detection

Used in :

- graphs
- linked list problems
- duplicate detection

Imp: Sets are unordered.
Elements must be immutable.

A python set is a hash-based collection of unique elements that provides O(1) average time for insertion, deletion and lookup.