UNIT 8 – SETS

What is a Set?

A set is an unordered collection of items.

Every set element is unique (no duplicates) and

must be immutable.

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

Output:

```
>>> {1, 2, 3}
```

Properties of Sets

1. Unordered: No index access
2. Unique: Automatically removes duplicates
3. Mutable: You can add/remove items

Set Methods

Adding & Removing Elements

```
s = {1, 2}
s.add(3)        # Adds 3
s.remove(2)     # Removes 2 (Error if not found)
s.discard(10)   # No error if 10 not found
s.pop()         # Removes random element
```

TIP!
Use discard() to avoid Errors!

Interview Question

Q: Difference between remove() and discard()?

A: remove() raises KeyError if item is missing, discard() does nothing.

Set Operations - I

1. Union (|)

Combines elements from both sets.

```
A = {1, 2, 3}
B = {3, 4, 5}
print(A | B)
```

Output:

```
>>> {1, 2, 3, 4, 5}
```

2. Intersection (&)

Common elements in both sets.

```
print(A & B)
```

Output:

```
>>> {3}
```

Set Operations - II

3. Difference (-)

Elements in A but NOT in B.

```
A = {1, 2, 3}
B = {3, 4, 5}
print(A - B)
```

Output:

```
>>> {1, 2}
```

4. Symmetric Difference (^)

Elements in either A or B, but NOT in both.

```
print(A ^ B)
```

Output:

```
>>> {1, 2, 4, 5}
```

Sets: Summary & Exercises

Quick Revision
- {} is a dictionary, set() is an empty set!
- Sets are great for removing duplicates.
- Sets are unordered - no indexing!

Mini Exercise

Q1: Given list [1,2,2,3,4,4,5], how to get unique values?

Ans: set([1,2,2,3,4,4,5])

Interview Question

Q: Can a set contain a list?

Ans: No, because lists are mutable (unhashable).

Intermediate: File Handling

Reading & Writing Files

```
# Writing
with open('test.txt', 'w') as f:
    f.write('Hello Python!')

# Reading
with open('test.txt', 'r') as f:
    print(f.read())
```

Output:

```
>>> Hello Python!
```

TIP!
Always use 'with' to auto-close files!

Intermediate: Exception Handling

Try-Except Blocks

Handling errors gracefully.

```
try:
    x = 1 / 0
except ZeroDivisionError:
    print('Cannot divide by zero!')
```

Output:

```
>>> Cannot divide by zero!
```

Intermediate: OOP Basics

Classes & Objects

```
class Student:
    def __init__(self, name):
        self.name = name

s1 = Student('Rahul')
print(s1.name)
```

Output:

```
>>> Rahul
```

Memory Trick

Class = Blueprint (Map of a house)

Object = Actual House built from map