## 1️⃣ Dictionary in Python

### Definition

A **dictionary** is a built-in data type in Python used to store **data in key–value pairs**.

Each key is unique and maps to a value.

Dictionaries are **unordered**, **mutable (changeable)**, and **don't allow duplicate keys**.

**Example:**

```
student = {
    "name": "Saumya Singh",
    "age": 25,
    "city": "Sultanpur"
}
```

Here:

- `"name"`, `"age"`, `"city"` → **keys**
- `"Saumya Singh"`, `21`, `"Delhi"` → **values**

### Accessing Values

You can access a value using its key:

```
print(student["name"])     # Saumya Singh
print(student["city"])     # Delhi
```

### Adding or Updating Values

You can add new key-value pairs or modify existing ones:

```
student["college"] = "ABC University"   # add new key-value
```

```
student["age"] = 22                    # update existing value
print(student)
```

**Removing Items**

```
student.pop("city")      # removes key 'city'
print(student)
```

**Dictionary Methods**

| Method | Description | Example |
|---|---|---|
| `.keys()` | Returns all keys | `student.keys()` |
| `.values()` | Returns all values | `student.values()` |
| `.items()` | Returns all key-value pairs as tuples | `student.items()` |
| `.get(key)` | Returns value of a key safely | `student.get("name")` |
| `.update(new_dict)` | Updates dictionary with another | `student.update({"city": "Lucknow"})` |

**Nested Dictionary**

You can store another dictionary inside a dictionary.

**Example:**

```
profile = {
    "username": "saumya1singh",
    "details": {
        "followers": 1200,
        "verified": True
    }
}
```

```
print(profile["details"]["followers"])   # 1200
```

**Practice Question 1**

Create a dictionary named `marks` to store marks of 3 subjects.
Add the subjects one by one and print the final dictionary.

**Example:**

```
Input:
Maths → 90
Science → 85
English → 88

Output:
{'Maths': 90, 'Science': 85, 'English': 88}
```

## 2️⃣ Sets in Python

**Definition**

A **set** is a collection of **unordered and unique items**. Sets automatically remove duplicate elements and are written using **curly braces `{ }`**.

**Example:**

```
languages = {"Python", "Java", "C++", "Python"}
print(languages)
# Output: {'C++', 'Java', 'Python'}
```

**Creating a Set**

```
empty_set = set()      # Empty set
nums = {1, 2, 3, 4}    # Non-empty set
```

**Set Properties**

- Unordered → no fixed index positions
- Unique → no duplicates
- Mutable → elements can be added or removed
- Cannot contain mutable elements like lists or dictionaries

---

**Adding and Removing Elements**

```
nums = {1, 2, 3}
nums.add(4)
nums.remove(2)
print(nums)   # {1, 3, 4}
```

**Other Useful Methods**

| Method | Description | Example |
|---|---|---|
| `.add(el)` | Adds an element | `nums.add(5)` |
| `.remove(el)` | Removes element | `nums.remove(1)` |
| `.clear()` | Empties the set | `nums.clear()` |
| `.pop()` | Removes a random element | `nums.pop()` |
| `.union(set2)` | Combines both sets | `{1,2}.union({2,3}) → {1,2,3}` |
| `.intersection(set2)` | Common elements of both sets | `{1,2,3}.intersection({2,3,4}) → {2,3}` |

**Practice Question 2**

You are given a list of programming languages:
`["Python", "Java", "C++", "Python", "Java", "C"]`
Convert it into a set and print how many **unique languages** Divya knows.

**Expected Output:**

```
{'Python', 'Java', 'C++', 'C'}
Divya knows 4 unique languages.
```

## 3️⃣ Difference Between Dictionary and Set

| Feature | Dictionary | Set |
|---|---|---|
| **Structure** | Stores data as key–value pairs | Stores unique values only |
| **Syntax** | `{ "key": value }` | `{value1, value2, ...}` |
| **Mutable** | Yes | Yes |
| **Duplicates** | Keys are unique | All elements unique |
| **Indexing** | Not supported | Not supported |

## ✅ Summary

- **Dictionary:** Stores key–value pairs, mutable, unique keys.
- **Set:** Stores unique, unordered elements.
- Both are very useful for **data handling and searching operations**.

## Mini Assignment

1. Create a dictionary storing meanings of 3 English words.
2. Create a set of numbers and show union and intersection with another set.

3. Try to add both integer `9` and float `9.0` to a set and observe what happens.
   *(Hint: You can convert one into a string to make both unique.)*