📘 Python Dictionary

📘 1. What is a Dictionary?

A **dictionary** is an unordered, mutable collection of **key-value** pairs in Python. It's like a real dictionary where you look up a *word (key)* to get its *meaning (value)*.

```
student = {
    "name": "Darshan",
    "age": 22,
    "skills": ["Python", "Node.js"],
    "is_active": True
}
```

🔤 2. Dictionary Syntax

```
my_dict = {
    "key1": "value1",
    "key2": "value2"
}
```

- Keys must be **immutable** (str, int, tuple)
- Values can be **anything** (int, list, dict, etc.)

🧰 3. Dictionary Methods (Zero to Hero)

Method	Description
`dict.get(key[, default])`	Returns value for key, else default
`dict.keys()`	Returns view object of all keys
`dict.values()`	Returns view object of all values
`dict.items()`	Returns view object of key-value pairs
`dict.update(other_dict)`	Adds items from another dict
`dict.pop(key)`	Removes item with key
`dict.popitem()`	Removes last inserted item
`dict.clear()`	Removes all items
`dict.copy()`	Returns shallow copy

Method	Description
`dict.setdefault(k, v)`	Returns value of key; if not present, inserts with value v
`fromkeys(seq, val)`	Creates dict from sequence with same value

🎯 4. Accessing and Modifying Values

```
person = {"name": "Amit", "city": "Delhi"}
print(person["name"])        # Access
person["age"] = 25           # Add new key
person["city"] = "Mumbai"    # Modify
```

🔁 5. Looping Through Dictionary

```
for k, v in person.items():
    print(f"{k} → {v}")
```

🧱 6. Nested Dictionary

```
student = {
    "name": "Darshan",
    "grades": {
        "math": 90,
        "science": 85
    }
}
print(student["grades"]["math"])  # → 90
```

🏢 7. Multi-Dictionaries (List of Dicts)

```
students = [
    {"name": "Darshan", "age": 22},
    {"name": "Ravi", "age": 21},
    {"name": "Neha", "age": 23}
]

for student in students:
    print(student["name"])
```

🎨 8. Dictionary Comprehension

```
squares = {x: x*x for x in range(1, 6)}
print(squares)  # → {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

With condition:

```
even_squares = {x: x*x for x in range(10) if x % 2 == 0}
```

📦 9. Real-World Use Cases

- 🔐 Config Settings: `{"host": "localhost", "port": 8080}`
- 👥 Storing user data
- 📊 Frequency Counter
- 📚 JSON parsing
- ☑️ Caching

🧠 10. Practice Tips

Task	Example
Merge two dicts	`{**dict1, **dict2}`
Find key with max value	`max(my_dict, key=my_dict.get)`
Reverse key-value	`{v: k for k, v in my_dict.items()}`
Filter dict	`{k: v for k, v in d.items() if v > 10}`
Dict from two lists	`dict(zip(keys, values))`

🧪 11. Dictionary Quiz (Sample)

1. What does `dict.get("age", 0)` return if age doesn't exist? → `0`
2. What does `dict1.update(dict2)` do? → Adds/overwrites keys from `dict2` into `dict1`.
3. Which is mutable – keys or values? → Values only. Keys must be immutable.

🚀 12. Mini Challenges

1. **Frequency Counter** – Count character frequencies in a string.
2. **Nested Lookup** – Access data from nested dicts.
3. **Reverse Dict** – Convert `{a:1, b:2}` to `{1:a, 2:b}`
4. **Student Grader** – Store and average marks for multiple students.

5. **JSON Formatter** – Pretty-print any nested dictionary.

🔥 Bonus Tricks

```
# Sort by value
sorted_dict = dict(sorted(my_dict.items(), key=lambda item: item[1]))

# Swap keys & values
inverted = {v: k for k, v in my_dict.items()}
```

📚 Cheatsheet Summary

```
my_dict = {"name": "Darshan", "age": 22}

my_dict["name"]              # Access
my_dict.get("city", "N/A")   # Safe access
my_dict["age"] = 23          # Update
my_dict["city"] = "Ahmedabad"  # Add
my_dict.pop("age")           # Remove
len(my_dict)                 # Length
"name" in my_dict            # Check key
list(my_dict.keys())         # All keys
list(my_dict.values())       # All values
```

📦 Python Dictionary Mastery Kit

🛠️ 30+ Practice Problems (Basic to Advanced)

◇ Level 1: Basics

1. Create a dictionary to store a student's name, age, and grade.
2. Add a new key `"gender"` with value `"male"`.
3. Update the grade to `"A+"`.
4. Check if a key `"age"` exists in the dictionary.
5. Delete a key from the dictionary.
6. Get the value of a non-existent key using `.get()`.
7. Loop through dictionary keys and values.

◇ Level 2: Intermediate

8. Create a dictionary from two lists using `zip()`.
9. Merge two dictionaries.
10. Sort a dictionary by its values.

11. Find the key with the maximum value.
12. Count frequency of each letter in a word.
13. Invert a dictionary (values as keys).
14. Filter a dictionary to keep only values > 50.
15. Use `setdefault()` to insert default values.

◇ Level 3: Nested + Advanced

16. Store marks of multiple students in nested dicts.
17. Access a value inside nested dictionaries.
18. Update a nested value.
19. Add a new student to the nested dictionary.
20. Create a list of student names from nested dict.
21. Count number of students scoring > 90.
22. Flatten a nested dictionary.
23. Group words by their first letter using dict.
24. Create a dictionary of squares using dict comp.
25. Track login attempts using usernames.

◇ Bonus Challenges

26. Build a frequency counter for words in a paragraph.
27. Remove duplicates from a list using dict keys.
28. Write a program to simulate a phonebook.
29. Track inventory of a store using dict.
30. Build a scorecard for a cricket match.
31. Check if two dictionaries are equal.
32. Merge list of dicts to a single dict.

🛠 Mini Projects

☑ 1. Student Report Card

Store multiple students with marks in subjects, compute average, topper, and failing students.

☑ 2. Library Management

Track book inventory, users, and who borrowed what.

☑ 3. Online Store

Track products, prices, stock, and customer carts using nested dicts.

☑ 4. Movie Database

Store movies, genres, and cast info using dictionaries.

☑ 5. Survey Analyzer

Process survey data like {question: [responses]} and find most common answers.

❓ 15+ MCQs & Quiz

1. What is the output of `dict.get("x", 0)` if `"x"` is not present? a) KeyError b) None c) 0 ☑ d) "x"
2. Which data type cannot be used as a dictionary key? a) str b) int c) tuple d) list ☑
3. `dict.items()` returns: a) keys b) values c) key-value pairs ☑ d) list
4. What is the output of `{"a":1} == {"a":1}`? ☑ True
5. Which method removes the last inserted item? a) pop b) del c) remove d) popitem ☑
6. What does `setdefault("x", 100)` do? a) Always sets x = 100 b) Only if x not exists ☑
7. What happens if you try to use a list as a key? ☑ TypeError (unhashable type)
8. How do you copy a dictionary? ☑ dict.copy()
9. What is the default return of `dict.get("missing")`? ☑ None
10. Can dictionary keys be repeated? ❌ No. Only last value is kept.

📦 JSON & API Case Studies using Dicts

🗂 JSON Parsing

```python
import json

json_str = '{"name": "Darshan", "age": 22}'
data = json.loads(json_str)
print(data["name"])  # Darshan
```

🌐 API Response Example

```python
response = {
    "status": "success",
    "data": {
        "user": {
            "id": 101,
            "name": "Darshan Vasani",
            "skills": ["Python", "React"]
        }
    }
}
print(response["data"]["user"]["skills"][0])
```

☑ API Usage Scenario

- **Weather API**: Extract temperature, humidity from JSON.
- **Github API**: Extract repo names, star counts using nested dicts.
- **Topmate API**: Parse booked sessions, time slots using dict.
