Dictionary Loop

Table of contents

Introduction

- In Python, you can loop through a dictionary to access its **keys**, **values**, or **both (key-value pairs)**.

- This is especially useful when you want to process or **display all elements** of a dictionary.

Looping Through Keys

- By default, looping over a dictionary using a **for loop** will iterate through its **keys**.

Python	Output

`person = {"name": "Riya", "age": 25, "city": "Pune"}`
 	
`for key in person:`
`  print(key)`
`  print(person[key])`
	Name
	Riya
	Age
	25
	City
	Pune

Looping Through Keys

- By default, looping over a dictionary using a **for loop** will iterate through its **keys**.
- You can also use **.keys()** explicitly.

Python	Output

`person = {"name": "Riya", "age": 25, "city": "Pune"}`
 	
`for key in person.keys():`
`  print(key)`
	Name
	Riya
	Age

Looping Through Values

- To loop only through the **values**, use the **.values()** method
- This is useful when you're only interested in the **data** and not in the keys.

Python	Output

`person = {"name": "Riya", "age": 25, "city": "Pune"}`
 	
`for value in person.values():`
`    print(value)`
	Riya
	25
	Pune

Looping Through Key-Value Pairs

- To access **both keys and values** at once, use the **.items()** method:
- This is the **most powerful** form of **dictionary iteration** and is commonly used in **data processing tasks**.

Python	Output

`person = {"name": "Riya", "age": 25, "city": "Pune"}`
 	
`for key, value in person.items():`
`  print(f"{key} → {value}")`
	`name → Riya`
	`age → 25`
	`city → Pune`

Looping Using enumerate()

- Though not very common for dictionaries, you can use **enumerate()** with **.items()** to also get the index.

Python	**Output**

`person = {"name": "Riya", "age": 25, "city": "Pune"}`
 	
`for index, (key, value) in enumerate(person.items()):`
`  print(f"{index}: {key} = {value}")`
	`0: name = Riya`
	`1: age = 25`
	`2: city = Pune`

Looping Using Dictionary Comprehension

- **Dictionary comprehension** is used when you need to **transform** or **filter data** efficiently using minimal code.

Python

```
person = {"name": "Riya", "age": 25, "city": "Pune"}

comp = {k: v for k, v in person.items() if type(v) is str}
print(comp)
```

Output

```
{'name': 'Riya', 'city': 'Pune'}
```

WATCH

Level up your coding with each episode in this focused Python series.

Next Video!

Nested Dictionary