Looping in Sets & Set Comprehension

Table of contents

01

Looping Over Set

Looping Over Set

- Python provides several ways to **loop through a set**.
- Sets are **unordered** collections of **unique elements**, looping through them follows *different* characteristics compared to **lists, strings, tuples or dictionaries.**

Why Loop Through a Set?

- To **access** each item in the set **one by one**.
- To **apply operations** to each item.
- To **filter** or transform data.

Looping Set – using for()

The most common way to iterate through a set is by using a **for** loop.

Python	Output

`my_set = {1, 2, 3, 4, 5}`
 	
`for item in my_set:`
`    print(item)`
	1
	2
	3
	4
	5

Note: Since sets are *unordered*, the output order may *vary each time* the loop runs.

Looping Set – using while()

- Sets **do not support indexing**, using a while loop directly is tricky.
- Convert the **set** into a **list** first.

Python	Output

`my_set = {10, 20, 30, 40, 50}`
`set_list = list(my_set)`
`i = 0`
 	
`while i < len(set_list):`
`    print(set_list[i])`
`    i += 1`
	
	10
	20
	30
	40
	50

Looping Nested Set

- Sets do **not** allow **mutable elements** (e.g., *lists*), but they can **contain immutable elements** like *tuples*.

Python	Output

`nested_set = {(1, 2), (3, 4), (5, 6)}`
 	
`for tup in nested_set:`
`  print(f"Tuple: {tup}")`
`  for value in tup:`
`  print(f"  Value: {value}")`
	`Tuple: (3, 4)`
	`  Value: 3`
	`  Value: 4`
	`Tuple: (5, 6)`
	`  Value: 5`
	`  Value: 6`
	`Tuple: (1, 2)`
	`  Value: 1`
	`  Value: 2`

02

Set Comprehension

Set Comprehension

- Set comprehension is a **quick** and **efficient way** to create **sets** using a single line of code.
- It is similar to **list comprehension**, but instead of creating a **list**, it creates a **set**.

Syntax

```
{expression for item in iterable if condition}
```

- **expression** → What you want to add to the set.
- **item** → Each value from the iterable (like a list, tuple, range, or another set).
- **if condition** (optional) → Filters the values that go into the new set.

Set Comprehension – Examples

- Creating a Set from a List

Python	**Output**

`numbers = [1, 2, 3, 4, 5]`
`new_set = {num for num in numbers}`
`print(new_set)`
	`{1, 2, 3, 4, 5}`

Set Comprehension – Examples

- Keeping Only Even Numbers

Python	**Output**

`numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]`
`even_numbers = {num for num in numbers if num % 2 == 0}`
`print(even_numbers)`
	`{2, 4, 6, 8, 10}`

Set Comprehension – Examples

- Extracting Unique Letters from a Word

Python	**Output**

`word = "banana"`
`unique_letters = {letter for letter in word}`
`print(unique_letters)`
	`{'a', 'n', 'b'}`

Set Comprehension – Examples

- Convert Words to Uppercase

Python	**Output**

`words = {"apple", "banana", "cherry"}`
`uppercase_words = {word.upper() for word in words}`
`print(uppercase_words)`
	`{'APPLE', 'CHERRY', 'BANANA'}`

Practice Set - 2

Download Link in **Description** and **Pinned Comment**

WATCH

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

Next Video!

Practice Set - 2
Solution