IBM

[**Python for Data Science, AI & Development**](https://www.coursera.org/learn/python-for-applied-data-science-ai/home/welcome)

**Module 2: Python Data Structures**

Table of Contents

[Dictionaries 4](#_Toc187604810)

[Characteristics 4](#_Toc187604811)

[Accessing Dictionary Items 4](#_Toc187604812)

[Modifying a Dictionary 5](#_Toc187604813)

[1. Add or Update: 5](#_Toc187604814)

[2. Remove Items: 5](#_Toc187604815)

[3. Clear All Items: 5](#_Toc187604816)

[Dictionary Methods 6](#_Toc187604817)

[Get length of dictionary in Python 6](#_Toc187604818)

[Looping Through Dictionaries 7](#_Toc187604819)

[1. Looping through keys: 7](#_Toc187604820)

[2. Looping through values: 7](#_Toc187604821)

[3. Looping through key-value pairs: 7](#_Toc187604822)

[Dictionary Comprehensions 7](#_Toc187604823)

[Nested Dictionaries 8](#_Toc187604824)

[Use Cases of Dictionaries 8](#_Toc187604825)

[1. Lookup Tables: 8](#_Toc187604826)

[2. Counting Items: 8](#_Toc187604827)

[3. Storing Configurations: 8](#_Toc187604828)

[Summary 8](#_Toc187604829)

[Sets 9](#_Toc187604830)

[Creating a Set in Python 9](#_Toc187604831)

[Operations on Sets 9](#_Toc187604832)

[Basic Operations 9](#_Toc187604833)

[Set Operations 10](#_Toc187604834)

[1. Union : 10](#_Toc187604835)

[1. Intersection 10](#_Toc187604836)

[2. Difference 10](#_Toc187604837)

[3. Difference\_update 10](#_Toc187604838)

[4. Symmetric Difference 10](#_Toc187604839)

[5. Isdisjoint() 11](#_Toc187604840)

[6. Issubset() 11](#_Toc187604841)

[7. issuperset() 11](#_Toc187604842)

[Accessing a Set in Python 11](#_Toc187604843)

[Frozen Sets in Python 12](#_Toc187604844)

[Key Points to Remember 13](#_Toc187604845)

# Dictionaries

A **dictionary** in Python is an **unordered, mutable collection** of key-value pairs. Each key in a dictionary is unique, and it is associated with a value. Dictionaries are optimized for retrieving values when the key is known.

In[Python](https://www.geeksforgeeks.org/python-programming-language), a dictionary can be created by ***placing a sequence of elements within curly {} braces, separated by a ‘comma’.***

Dict = {"key1": 1, "key2": "2", "key3": [3, 3, 3], "key4": (4, 4, 4), ('key5'): 5, (0, 1): 6}

We can also create dictionary ***using dict() constructor***.



## Characteristics

* From Python 3.7 Version onward, Python dictionary is Ordered.
* **Dictionary keys are case sensitive:**the same name but different cases of Key will be treated distinctly.
* **Keys must be immutable:** This means keys can be strings, numbers, or tuples but not lists.
* **Keys must be unique:** Duplicate keys are not allowed and any duplicate key will overwrite the previous value.
* Dictionary internally uses [Hashing](https://www.geeksforgeeks.org/introduction-to-hashing-2/). Hence, operations like search, insert, delete can be performed in **Constant Time**.

## Accessing Dictionary Items

We can access a value from a dictionary by using the **key**within square brackets or **get ()** method.

A close-up of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

Using ***get ()*** method is safer as it doesn’t raise an error if the key is missing.

## Modifying a Dictionary

### Add or Update:

A screenshot of a computer

Description automatically generated

### Remove Items:

Using del:



Using pop():

A screenshot of a computer program

Description automatically generated

Using popitem():

* Removes the last inserted key-value pair (Python 3.7+).

A close-up of a computer code

Description automatically generated

### Clear All Items:

Clears all the key-value pairs of dictionaries

A close up of a hashtag

Description automatically generated

## Dictionary Methods

A screenshot of a dictionary

Description automatically generated

A screenshot of a computer code

Description automatically generated

## Get length of dictionary in Python

Python provides multiple methods to get the length, and we can apply these methods to both simple and nested dictionaries.

A computer code with text

Description automatically generated with medium confidence

A screenshot of a computer code

Description automatically generated

## Looping Through Dictionaries

### Looping through keys:

A close up of words

Description automatically generated gives all the keys of pair.

### Looping through values:

 gives all the values of pair.

### Looping through key-value pairs:

A close-up of a word

Description automatically generated gives all the key value pairs.

## Dictionary Comprehensions

A concise way to create dictionaries.

A computer code with text

Description automatically generated with medium confidence

## Nested Dictionaries

Dictionaries can contain other dictionaries.

A screenshot of a computer code

Description automatically generated

## Use Cases of Dictionaries

### Lookup Tables:

Quick retrieval of information.

A close up of a word

Description automatically generated

### Counting Items:

A screenshot of a computer code

Description automatically generated

A white background with black text

Description automatically generated

### Storing Configurations:

A close-up of a text

Description automatically generated

## Summary

Python dictionaries are an essential data structure for handling real-world data efficiently:

* They are flexible, allowing hierarchical and dynamic data storage.
* Operations like search, insert, and delete are fast due to ***hashing.***
* With practical applications ranging from contact books to data analytics, dictionaries are an indispensable tool for Python developers.

Mastering dictionaries will significantly improve your programming efficiency and problem-solving abilities.

# Sets

A **set** in Python is a collection of unique, unordered, and unindexed elements. It is a built-in data type designed to hold multiple items in a single variable, ensuring that each element is distinct. Sets are mutable, meaning you can add or remove items, but their elements must be immutable (like integers, strings, or tuples).

## Creating a Set in Python

In Python, the most basic and efficient method for creating a set is using curly braces. We can also create set using set() constructor.

A screenshot of a computer code

Description automatically generated

Python Sets can be created by using the built-in [**set()** function](https://www.geeksforgeeks.org/python-set-function/) with an iterable object or a sequence by placing the sequence inside curly braces, separated by a ‘comma’.

***Note:*** *A Python set cannot have mutable elements like a list or dictionary, as it is immutable.*

## Operations on Sets

## Basic Operations

* **Adding Elements:** Use add() to add a single element.

**A screenshot of a computer code

Description automatically generated**

* **Removing Elements:** Use remove() or discard(). The difference is that remove() raises an error if the element is not found, whereas discard() does not.

**A white background with black text

Description automatically generated**

We can also use pop() to remove elements in a set but it can remove random elements**.**

* **Clearing and Deleting**

**clear():** Removes all elements from the set.

**del:** Deletes the set entirely.

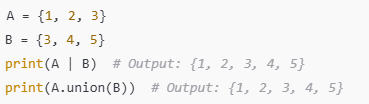
A computer screen shot of a number

Description automatically generated

## Set Operations

### 1. Union :

Combines all unique elements from two sets.



### Intersection

Finds common elements between sets.

A close-up of a computer code

Description automatically generated

### Difference

Elements in one set but not in another.

A screenshot of a computer code

Description automatically generated

### Difference\_update

If A and B are two sets. The set **difference ()** method will get the (A – B) and will return a new set. The set **difference\_update ()** method modifies the existing set. If (A – B) is performed, then A gets modified into (A – B), and if (B – A) is performed, then B gets modified into (B – A).

### Symmetric Difference

Elements in either of the sets but not both.

A close-up of a number

Description automatically generated

### Isdisjoint()

It checks whether the two sets are disjoint or not, if it is disjoint then it returns True otherwise it will return False. Two sets are said to be disjoint when their intersection is null.

A white background with black text

Description automatically generated

### Issubset()

This returns True if all elements of a set A are present in another set B which is passed as an argument, and returns False if all elements are not present in [Python](https://www.geeksforgeeks.org/python-programming-language/).

A black and white text

Description automatically generated

### issuperset()

**Python Set issuperset() method** returns True if all elements of a set B are in set A. Then Set A is the superset of set B.A group of black text

Description automatically generated

## Accessing a Set in Python

We can loop through a set to access set items as set is unindexed and do not support accessing elements by indexing. Also we can use [in keyword](https://www.geeksforgeeks.org/python-in-keyword/) which is membership operator to check if an item exists in a set.

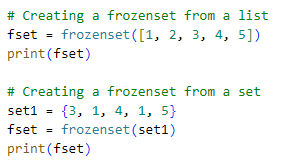
A screenshot of a computer code

Description automatically generatedSets MethodsA screenshot of a computer program

Description automatically generated

## Frozen Sets in Python

A [frozenset](https://www.geeksforgeeks.org/frozenset-in-python/)in Python is a built-in data type that is similar to a set but with one key difference that is immutability. This means that once a frozenset is created, we cannot modify its elements that is we cannot add, remove or change any items in it. Like regular sets, a frozenset cannot contain duplicate elements.

## Key Points to Remember

* Sets are best used when you need to store unique elements.
* They are unordered, so operations involving indexing are not supported.
* Use frozen sets when immutability is required.