# Sets in Python-Complete Tutorial

Python / March 20, 2020

This **Sets in Python-Complete Tutorial** is made for you to give complete knowledge of the **Sets in Python**. At the end of this tutorial, you will have full knowledge of the **Sets in Python**.

Hello, & Welcome! to the **Sets in Python -Complete Tutorial**.

### Part 12- Sets in Python -Complete Tutorial

### In this article of Python Tutorial, you will learn following-

1. What is Set in Python?
2. How to Access an Item in Sets?
3. Change an Item in Sets.
4. Add an item in Sets.
5. Remove an Item from Sets.
6. Concatenation of two Sets.
7. Python Set Operation.

## What is Set in Python?

A set is a collection of unordered and unindexed items. It is similar to list but Sets doesn't allow duplicate items. A set is defined by curly braces {}. Sets perform different mathematical operations like union, intersection, and differences.

Let's see how to create a set in python.

```
my_set={"orange","red","green","yellow"}
print(my_set)

OUTPUT-
{'yellow', 'red', 'green', 'orange'}
```

In the output, items are not in the same order as we put in the set. This shows that sets are an unordered collection of an item.

## How to Access an Item in Sets?

As we discuss, the set is an unindexed collection of items, that's why you can not access items in sets. But you can check that specific item is present in the set or not by using '**in**'

keyword. You can loop through by **for** loop.

Let's understand with the help of an example by using **for loop**–

```
my_set={"orange","red","green","yellow"}
for i in my_set:
   print(i)

OUTPUT-
orange
green
red
yellow
```

Using '**in**' Keyword-

```
my_set={"orange","red","green","yellow"}
print("red" in my_set)

OUTPUT-
True
```

Here, we are checking that "red" is present in the set or not, and we are getting output as True.

## Change an Item in Sets.

After creating a set, you can't change or modify its element. You can add an item in sets but not update any item.

## Add an item in Sets.

You can add an item to the sets. If you want to add only one item, then you can add it by using **add()** method. But if you want to add more than one item, then use the **update()** method.

Let's have a look at **add()** method example-

```
my_set={"orange","red","green","yellow"}
my_set.add("Black")
print(my_set)

OUTPUT-
{'yellow', 'red', 'Black', 'orange', 'green'}
```

Let's have a look at **update()** method-

```
my_set={"orange","red","green","yellow"}
my_set.update(["Black","White","Pink"])

OUTPUT-
{'Pink', 'yellow', 'red', 'Black', 'orange', 'green', 'White'}
```

Here, by using **the update()** method, we are adding more than one item in the set.

# Remove an Item from Sets.

If you want to remove any item from set, then there are multiple ways in python to remove an item. We will discuss all the methods one by one.

## Using remove() method-

You can delete any item from the set by using the remove() method.

Let's see in the example-

```
my_set={"orange","red","green","yellow"}
my_set.remove("red")

OUTPUT-
{'yellow', 'green', 'orange'}
```

## Using discard() method-

Discard() method works mostly the same as remove() method. You can remove the item from the set by using the discard() method.

Let's see in the example-

```
my_set={"orange","red","green","yellow"}
my_set.discard("red")

output-
{'yellow', 'green', 'orange'}
```

## Using pop() method-

The pop() method removes the last element of the set. But, as we know that set is unordered and you don't know which item is the last item in the set.

Let's see in the example below-

```
my_set={"orange","red","green","yellow"}
i=my_set.pop()
print(i)         #deleted item
print(my_set)


OUTPUT-
'yellow'
{'red', 'green', 'orange'}
```

Here, "yellow" is deleted from the list. we are storing a deleted value in **"i"** and printing it to check which item is deleted.

## Using clear() method-

The clear() method makes a set empty, which means no item will be there after using the clear() method.

Let's see in the example-

```
my_set={"orange","red","green","yellow"}
my_set.clear()
print(my_set)

OUTPUT-
set()
```

In the output, we get an empty set.

### Using del Keyword-

If you wanna delete your set completely then you should use **del** keyword.

Let's understand with the help of an example-

```
my_set={"orange","red","green","yellow"}
del my_set
print(my_set)

OUTPUT-
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'my_set' is not defined
```

## Concatenation of two Sets.

If you wanna join two sets into one, then you can do it by using an **update()** method and by **union()** method.

Let's have a look at **update()** method-

```
my_set1={"orange","red","green","yellow"}
my_set2={"Black","White","Pink"}
my_set1.update(my_set2)
print(my_set1)


OUTPUT-
{'Pink', 'yellow', 'red', 'Black', 'orange', 'green', 'White'}
```

Here, we have joined two sets into one by using the update() method.

Let's have a look at union() method example-

```
my_set1={"orange","red","green","yellow"}
my_set2={"Black","White","Pink"}
my_set3=my_set1.union(my_set2)
print(my_set3)

OUTPUT-
{'Pink', 'yellow', 'red', 'Black', 'orange', 'green', 'White'}
```

Here, both method **update()** and **union()** doesn't allow duplicate items in a set.

# Python Set Operation.

Set is used to perform different set operations like union, intersection and symmetric difference.

## Union of Set

Union of set means joins or concatenate two sets into one. It combines all the items of both sets into one.

You can perform a union in python by using | symbol, or you can use union() method.

Let's understand with the help of an example-

```
X={1,2,3,4,5,6}
Y={6,7,8,9,10}
print(X|Y)

OUTPUT-
{1,2,3,4,5,6,7,8,9,10}
```

## The intersection of Set

The intersection of set means, select only common items from both the sets. It doesn't take all the items from the set, only take common items.

You can perform an intersection by using & operator or by using an intersection() method.

```
X={1,2,3,4,5,6}
Y={6,7,8,9,10}
print(X & Y)

OUTPUT-
{6}
```

## Set Difference

A difference of two sets means X-Y (means items that are in X but not in Y), similarly Y-X (means items that are in Y not in X).

You can perform a difference by using – operator or by using the difference() method.

```
X={1,2,3,4,5,6}
Y={5,6,7,8,9,10}
print(X-Y)

OUTPUT-
{1,2,3,4}
```

That's all for the **Sets in Python**. I hope now you have a better understanding of the Sets in Python.

Congratulations! You successfully learned the **Sets in** Python.

In the next tutorial, we will start learning **Classes in Python**.

Till then, Enjoy Learning Python!

All the Best!

Are you ML Beginner and confused, from where to start ML, then read my BLOG – How do I learn Machine Learning?

If you are looking for Machine Learning Algorithms, then read my Blog – Top 5 Machine Learning Algorithm.

If you are wondering about Machine Learning, read this Blog- What is Machine Learning?

Thank YOU!

## Though of the Day...

> *"Live as if you were to die tomorrow. Learn as if you were to live forever."*

> – *Mahatma Gandhi*

← Previous Post Next Post →

# Leave a Comment

Your email address will not be published. Required fields are marked *

Type here..

Name*

Email*

Website

Save my name, email, and website in this browser for the next time I comment.

Post Comment »

## Follow Us on Twitter-

## Follow us on Pinterest-

## Our Facebook Group-