Sets

Python Sets

это неупорядоченная, неизменяемая* и неиндексированная коллекция and **do not** allow duplicate values.

Set *items* are unchangeable, but you can remove items and add new items.

```
thisset = {"apple", "banana", "cherry"}
print(thisset)
```

A set can contain different data types:

```
set1 = {"abc", 34, True, 40, "male"}
```

Once a set is created, you cannot change its items, but you can add new items.

add() method.

To add one item to a set use the `add()` method.

```
thisset = {"apple", "banana", "cherry"}
thisset.add("orange")
print(thisset)
```

update() method.

To add items from another set into the current set, use the `update()` method.

```
thisset = {"apple", "banana", "cherry"}
tropical = {"pineapple", "mango", "papaya"}
thisset.update(tropical)
print(thisset)
```

The object in the `update()` method does not have to be a set, it can be any iterable object (tuples, lists, dictionaries etc.).

```
thisset = {"apple", "banana", "cherry"}
mylist = ["kiwi", "orange"]
thisset.update(mylist)
print(thisset)
```

remove() or discard() method.

To remove an item in a set, use the `remove()`, or the `discard()` method.

```
thisset = {"apple", "banana", "cherry"}
```

```
thisset.remove("banana")
print(thisset)

thisset = {"apple", "banana", "cherry"}
thisset.discard("banana")
print(thisset)
```

Если удаляемый элемент не существует, функция discard() НЕ выдаст ошибку.

Вы также можете использовать `pop()` метод для удаления элемента, но этот метод удалит *последний* элемент. Помните, что наборы неупорядочены, поэтому вы не будете знать, какой элемент будет удален.
Возвращаемое значение `pop()` метода — удаленный элемент.

```
thisset = {"apple", "banana", "cherry"}
x = thisset.pop()
print(x)
print(thisset)

cherry
{'banana', 'apple'}
```

`clear()` method – empties the set
`del()` method – will delete the set completely

`union()` method
You can use the `union()` method that returns a new set containing all items from both sets:

```
set1 = {"a", "b" , "c"}
set2 = {1, 2, 3}
set3 = set1.union(set2)
print(set3)
```

Keep ONLY the Duplicates

The `intersection_update()` method will keep only the items that are present in both sets.

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
x.intersection_update(y)
print(x)

{'apple'}

# or
```

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
z = x.intersection(y)
print(z)
```

Keep All, But NOT the Duplicates

The `symmetric_difference_update()` method will keep only the elements that are NOT present in both sets.

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
x.symmetric_difference_update(y)
print(x)

{'google', 'banana', 'microsoft', 'cherry'}

# or

x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
z = x.symmetric_difference(y)
print(z)
```

Convert set to list:

```
# Python3 program to convert a
# set into a list
my_set = {'Geeks', 'for', 'geeks'}

s = list(my_set)
print(s)
```

Convert list to set:

```
# sample_list is defined list
sample_list = [1,2,3,'seeker',3,7.5]
# set() to convert list to set
sample_set  = set(sample_list)
print(sample_set) #printing set
```

Set Methods

Method	Description
add()	Adds an element to the set
clear()	Removes all the elements from the set
copy()	Returns a copy of the set
difference()	Returns a set containing the difference between two or more sets
difference_update()	Removes the items in this set that are also included in another, specified set
discard()	Remove the specified item
intersection()	Returns a set, that is the intersection of two other sets
intersection_update()	Removes the items in this set that are not present in other, specified set(s)
isdisjoint()	Returns whether two sets have a intersection or not
issubset()	Returns whether another set contains this set or not
issuperset()	Returns whether this set contains another set or not
pop()	Removes an element from the set
remove()	Removes the specified element
symmetric_difference()	Returns a set with the symmetric differences of two sets
symmetric_difference_update()	inserts the symmetric differences from this set and another
union()	Return a set containing the union of sets
update()	Update the set with the union of this set and others