

Set Operations

symmetric_difference

- prints the elements from either of the sets, but not in both.
- prints uncommon elements from either of sets.
- removes common elements and prints uncommon elements.

```
In [1]: A = {1,2,3,4,5,6}
        B = {5,6,7,8,9}
        A.symmetric_difference(B)
```

```
Out[1]: {1, 2, 3, 4, 7, 8, 9}
```

Using symmetric difference short cut operator ^

```
In [2]: A^B
```

```
Out[2]: {1, 2, 3, 4, 7, 8, 9}
```

symmetric_difference_update

- it updates the original set to keep only the elements that are NOT common (i.e the uncommon ones).
- symmetric_difference_update = symmetric_difference() + assign back to the same set.

```
In [4]: A = {1,2,3,4,5,6}
        B = {5,6,7,8,9}
        A.symmetric_difference_update(B)
        print (A)
```

```
{1, 2, 3, 4, 7, 8, 9}
```

```
In [5]: B.symmetric_difference_update(A)
        print(B)
```

```
{1, 2, 3, 4, 5, 6}
```

difference between symmetric_difference vs symmetric_difference_update

- consider school note book as an example:

- `symmetric_difference()` takes notes everyday on a new page - creates a new set
- `symmetric_difference_update()` erases and re-writes on the same page - update the same set

```
In [6]: A = {1,2,3,4}
        B = {3,4,5,6}
```

```
In [13]: C = A.symmetric_difference(B)
          print(C)      # creates a new set
          print(A)      # unchanged

{1, 2, 5, 6}
{1, 2, 3, 4}
```

```
In [9]: A = {1,2,3,4}
        B = {3,4,5,6}
```

```
In [11]: C = A.symmetric_difference_update(B)
          print(C)
```

None

```
In [14]: print(A) # update the same set

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

issubset()

- checks if all elements of one set are present in another set.

```
In [15]: A = {1,2,3,4,5,6}
          B = {3,4,5}
          C = {10,20,30}
```

```
In [16]: A.issubset(B)
```

Out[16]: False

```
In [17]: B.issubset(A)
```

Out[17]: True

```
In [18]: C.issubset(A)
```

Out[18]: False

superset()

- checks if a set contains all elements of another set.

```
In [19]: A = {1,2,3,4,5,6}
          B = {3,4,5}
          C = {10,20,30}
```

```
In [20]: A.issuperset(B)
```

```
Out[20]: True
```

```
In [21]: B.issuperset(A)
```

```
Out[21]: False
```

isdisjoint()

- checks if two sets have nothing in common
- no shared elements

```
In [22]: A = {1,2,3,4,5,6}  
B = {3,4,5}  
C = {10,20,30}
```

```
In [23]: C.isdisjoint(A)
```

```
Out[23]: True
```

```
In [24]: B.isdisjoint(A)
```

```
Out[24]: False
```

other built-in functions

```
In [25]: a = {1,2,3,4,5,6,7,8,9,10}  
a
```

```
Out[25]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

```
In [26]: sum(a)
```

```
Out[26]: 55
```

```
In [27]: max(a)
```

```
Out[27]: 10
```

```
In [28]: min(a)
```

```
Out[28]: 1
```

```
In [29]: len(a)
```

```
Out[29]: 10
```

```
In [30]: sorted(a)
```

```
Out[30]: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
In [32]: sorted(a)
```

```
Out[32]: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
In [33]: sorted(a, reverse=True)
```

```
Out[33]: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
```

Dictionary

- dictionary is a collection of key-value pairs.
- Keys must be unique in a dictionary, duplicate values are allowed.

```
In [36]: mydict = dict()  
print(mydict)
```

```
{}
```

```
In [37]: type(mydict)
```

```
Out[37]: dict
```

```
In [38]: mydict = {}
```

```
In [39]: print(mydict)
```

```
{}
```

```
In [40]: type(mydict)
```

```
Out[40]: dict
```

```
In [41]: mydict = {1:'one', 2:'two', 3:'three', 4:'four'}  
mydict
```

```
Out[41]: {1: 'one', 2: 'two', 3: 'three', 4: 'four'}
```

```
In [42]: mydict = {'A':'one' , 'B':'two' , 'C':'three'} # dictionary with character keys  
mydict
```

```
Out[42]: {'A': 'one', 'B': 'two', 'C': 'three'}
```

```
In [43]: mydict = {1:'one' , 'A':'two' , 3:'three'} # dictionary with mixed keys  
mydict
```

```
Out[43]: {1: 'one', 'A': 'two', 3: 'three'}
```

```
In [46]: mydict = {1:'one' , 2:'two' , 'A':['asif' , 'john' , 'Maria']}  
mydict
```

```
Out[46]: {1: 'one', 2: 'two', 'A': ['asif', 'john', 'Maria']}
```

```
In [47]: keys = {'a' , 'b' , 'c' , 'd'}  
mydict3 = dict.fromkeys(keys) # Create a dictionary from a sequence of keys
```

```
mydict3
```

```
Out[47]: {'b': None, 'c': None, 'd': None, 'a': None}
```

```
In [48]: keys = {'a' , 'b' , 'c' , 'd'}  
value = 10  
mydict3 = dict.fromkeys(keys , value) # Create a dictionary from a sequence of  
mydict3
```

```
Out[48]: {'b': 10, 'c': 10, 'd': 10, 'a': 10}
```

range

```
In [49]: range(10)
```

```
Out[49]: range(0, 10)
```

```
In [50]: range(10,20)
```

```
Out[50]: range(10, 20)
```

```
In [51]: list(range(10,20))
```

```
Out[51]: [10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
```

```
In [52]: list(range(10,20,5))
```

```
Out[52]: [10, 15]
```

```
In [ ]:
```