Python Sets

A set is an unordered collection of items. Every set element is unique (no duplicates) and must be immutable (cannot be changed).

However, a set itself is mutable. We can add or remove items from it.

Sets can also be used to perform mathematical set operations like union, intersection, symmetric difference, etc.

Characterstics:

- Unordered
- Mutable
- No Duplicates
- Can't contain mutable data types

Creating sets

In [10]:

```
# empty
s = set()
print(s)
print(type(s))
# 1D and 2D
s1 = {1,2,3}
print(s1)
# cannot create a 2d set because the set is itself mutable but it cannot store mutable datan type
#s2 = {1,2,3,{4,5}}
#print(s2)
# homo and hetero
s3 = {1,'hello',4.5,(1,2,3)}
print(s3)
# using type conversion

s4 = set([1,2,3])
print(s4)
```

```
# Duplicates not allowed
s5 = {1,1,2,2,3,3}
print(s5)
# set can't have mutable items
# s6 = {1,2,[3,4]}
# print(s6)
```

```
set()
<class 'set'>
{1, 2, 3}
{1, 'hello', (1, 2, 3), 4.5}
{1, 2, 3}
{1, 2, 3}
```

In [12]:

```
# sets are unordered -
s1 = {1,2,3}
s2 = {3,2,1}

print(s1 == s2)
```

```
True
```

Accessing items

In [23]:

```
# Since it is unordered indexing and slicing will not work
s={1,2,3,5.34}
# s[1:3]
```

Editing items

In [29]:

```
# since it is unordered so editing is also not possible
s={234,45,52,11}
# s[0]=23
```

Adding items

In [48]:

```
# using add method - we can add new elemnent
s={34,354,2,1.34,23,True}
```

```
s.add(False)
print(s)

# using update we can add multiple items by using a list
s.update([23,3+5j,False])
print(s)
```

```
{False, 1.34, 34, 2, 354, True, 23}
{False, 1.34, 34, 2, 354, True, (3+5j), 23}
```

Deleting items

In [79]:

```
# using del keyword , we can only delete whole set
s={34,45,67,234,6}
# del s[1] - not possible as it is unordered
del s

# discard - delete specific element if exist else print current set
s={34,767,23,88}
s.discard(24)
print(s)

# remove - it also remove element from set but give error if not present in set
s={34,67,899,346,341}
s.remove(34)
print(s)

# pop - remove any element from set
s={12,134,14,55}
s.pop()
print(s)

# clear - empty the set
s.clear()
print(s)
```

```
{88, 34, 767, 23}
{67, 899, 341, 346}
{14, 134, 55}
set()
```

Set operations

In [82]:

```
s1 = {1,2,3,4,5}
s2 = {4,5,6,7,8}

# union
print(s1 | s2)
# Intersection(&)
print(s1 & s2)
# Difference(-)
print(s1 - s2)
print(s2 - s1)
# Symmetric Difference(^)
print(s1 ^ s2)
# Membership Test
print(1 not in s1)
# Iteration
for i in s1:
  print(i)
```

```
{1, 2, 3, 4, 5, 6, 7, 8}
{4, 5}
{1, 2, 3}
{8, 6, 7}
{1, 2, 3, 6, 7, 8}
False
1
2
3
4
5
```

Set functions

In [88]:

```
# len/sum/min/max/sorted
s = {3,1,4,5,2,7}
print(len(s),sum(s),min(s),max(s),sorted(s,reverse=True))
```

```
6 22 1 7 [7, 5, 4, 3, 2, 1]
```

In [110…

```
s1 = {1,2,3,4,5}
s2 = {4,5,6,7,8}

# union/update - s1|s2
print(s1.union(s2))
# update permanently change the set
s1.update(s2)
print(s1)
print(s2)
# intersection/intersection_update
s1 = {1,2,3,4,5}
s2 = {4,5,6,7,8}

print(s1.intersection(s2))

print(s1.intersection_update(s2))
print(s1)
print(s2)
```

```
{1, 2, 3, 4, 5, 6, 7, 8}
{1, 2, 3, 4, 5, 6, 7, 8}
{4, 5, 6, 7, 8}
{4, 5}
None
{4, 5}
{4, 5, 6, 7, 8}
```

In [112…

```
# difference/difference_update
s1 = {1,2,3,4,5}
s2 = {4,5,6,7,8}

print(s1.difference(s2))

print(s1.difference_update(s2))
print(s1)
print(s2)
# symmetric_difference/symmetric_difference_update
s1 = {1,2,3,4,5}
s2 = {4,5,6,7,8}
```

```
print(s1.symmetric_difference(s2))

print(s1.symmetric_difference_update(s2))
print(s1)
print(s2)
```

```
{1, 2, 3}
None
{1, 2, 3}
{4, 5, 6, 7, 8}
{1, 2, 3, 6, 7, 8}
None
{1, 2, 3, 6, 7, 8}
{4, 5, 6, 7, 8}
```

In [104…

```
# isdisjoint/issubset/issuperset
s1 = {1,2,3,4}
s2 = {7,8,5,6}

print(s1.isdisjoint(s2))

# superset check method
s1 = {1,2,3,4,5}
s2 = {3,4,5}

print(s1.issuperset(s2))
# copy
s1 = {1,2,3}
s2 = s1.copy()

print(s1,id(s1))
print(s2,id(s2))
```

```
True
True
{1, 2, 3} 2270688948768
{1, 2, 3} 2270688948320
```

Frozen set

Frozen set is just an immutable version of a Python set object

- what works and what does not
- works -> all read functions
- does't work -> write operations

In [117…

```
# create frozenset
fs1 = frozenset([1,2,3])
fs2 = frozenset([3,4,5])

fs1 | fs2
```

Out[117…

```
frozenset({1, 2, 3, 4, 5})
```

In [124…

```
# When to use
# 2D sets
fs = frozenset([1,2,frozenset([3,4])])
fs
```

Out[124…

```
frozenset({1, 2, frozenset({3, 4})})
```

In [ ]:

```

```

Set comprehension

In [130…

```
# examples

{i**2 for i in range(1,11) if i>5}
```

Out[130…

```
{36, 49, 64, 81, 100}
```

END