Python - Sets

unordered collections of unique data elements

curly braces

set

no indexing

In [1]:

```
1 help(set)
```

```
Help on class set in module builtins:

class set(object)
 |  set() -> new empty set object
 |  set(iterable) -> new set object
 |
 |  Build an unordered collection of unique elements.
 |
 |  Methods defined here:
 |
 |  __and__(self, value, /)
 |      Return self&value.
 |
 |  __contains__(...)
 |      x.__contains__(y) <==> y in x.
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
```

In [2]:

```
1 print(dir(set))
```

```
['__and__', '__class__', '__contains__', '__delattr__', '__dir__', '__
doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__gt_
_', '__hash__', '__iand__', '__init__', '__init_subclass__', '__ior_
_', '__isub__', '__iter__', '__ixor__', '__le__', '__len__', '__lt__',
'__ne__', '__new__', '__or__', '__rand__', '__reduce__', '__reduce_ex_
_', '__repr__', '__ror__', '__rsub__', '__rxor__', '__setattr__', '__s
izeof__', '__str__', '__sub__', '__subclasshook__', '__xor__', 'add',
'clear', 'copy', 'difference', 'difference_update', 'discard', 'inters
ection', 'intersection_update', 'isdisjoint', 'issubset', 'issuperse
t', 'pop', 'remove', 'symmetric_difference', 'symmetric_difference_upd
ate', 'union', 'update']
```

In [3]:

```
1 setf = ['add', 'clear', 'copy', 'difference', 'difference_update', 'discard', 'i
2         'intersection_update', 'isdisjoint', 'issubset', 'issuperset', 'pop', 'r
3         'symmetric_difference_update', 'union', 'update']
```

add : singleton elment

In [4]:

```
a = {1,2,3,4,5}
```

In [5]:

```
a
```

Out[5]:

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

In [6]:

```
type(a)
```

Out[6]:

```
set
```

In [7]:

```
b = {10, 9, 8, 7, 6, 5}
b
```

Out[7]:

```
{5, 6, 7, 8, 9, 10}
```

In [8]:

```
c = {'a','b','c'}
c
```

Out[8]:

```
{'a', 'b', 'c'}
```

In [9]:

```
d = {'d','c','b','a'}
d
```

Out[9]:

```
{'a', 'b', 'c', 'd'}
```

In [10]:

```
b.add(11)
b
```

Out[10]:

```
{5, 6, 7, 8, 9, 10, 11}
```

In [11]:

```
b.add(11)
b
```

Out[11]:

```
{5, 6, 7, 8, 9, 10, 11}
```

In [13]:

```
f = list(b)
f.append(10)
```

In [14]:

```
f
```

Out[14]:

```
[5, 6, 7, 8, 9, 10, 11, 10]
```

In [15]:

```
g = set(f)
g
```

Out[15]:

```
{5, 6, 7, 8, 9, 10, 11}
```

In [21]:

```
k = {11,}
k.update({12, 13, 14, 15, 16})
```

In [22]:

```
k
```

Out[22]:

```
{11, 12, 13, 14, 15, 16}
```

In [25]:

```
r = {'abc','bcd'}
r.update({'cde','efg','jhi'})
```

In [26]:

```
r
```

Out[26]:

```
{'abc', 'bcd', 'cde', 'efg', 'jhi'}
```

mathematical terms

In [27]:

```
a = {1, 2, 3, 4}
b = {4, 5, 6, 7}
```

common elements

In [28]:

```
a.intersection(b)
```

Out[28]:

```
{4}
```

In [29]:

```
b.intersection(a)
```

Out[29]:

```
{4}
```

In [30]:

```
a & b , b & a
```

Out[30]:

```
({4}, {4})
```

combine elements of sets

In [31]:

```
a.union(b)
```

Out[31]:

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

In [32]:

```
b.union(a)
```

Out[32]:

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

In [33]:

```
a | b, b | a
```

Out[33]:

```
({1, 2, 3, 4, 5, 6, 7}, {1, 2, 3, 4, 5, 6, 7})
```

In [37]:

```
1 (a&b)
```

Out[37]:

```
{4}
```

In [35]:

```
1 a&b
```

Out[35]:

```
{4}
```

diff in set a compared to set b

In [38]:

```
1 a - b
```

Out[38]:

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

In [39]:

```
1 a.difference(b)
```

Out[39]:

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

diff in set b compared to set a

In [40]:

```
1 b - a
```

Out[40]:

```
{5, 6, 7}
```

In [41]:

```
1 b.difference(a)
```

Out[41]:

```
{5, 6, 7}
```

symmetricdifference__

In [42]:

```
1 a.symmetric_difference(b)
```

Out[42]:

```
{1, 2, 3, 5, 6, 7}
```

In [43]:

```
1 d
```

Out[43]:

```
{'a', 'b', 'c', 'd'}
```

remove

In [47]:

```
1 d.remove('d')
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call
 last)
<ipython-input-47-afcd16718406> in <module>
----> 1 d.remove('d')

KeyError: 'd'
```

In [46]:

```
1 d
```

Out[46]:

```
{'a', 'b', 'c'}
```

discard

In [48]:

```
1 d.discard('a')
2 d
```

Out[48]:

```
{'b', 'c'}
```

In [49]:

```
1 d.discard('a')
```

In [50]:

```
1 help('set.discard')
```

```
Help on method_descriptor in set:

set.discard = discard(...)
    Remove an element from a set if it is a member.

    If the element is not a member, do nothing.
```

pop

In [52]:

```
1 f
```

Out[52]:

```
[5, 6, 7, 8, 9, 10, 11, 10]
```

In [53]:

```
1 f.pop()
```

Out[53]:

```
10
```

In [54]:

```
1 f.pop(2)
```

Out[54]:

```
7
```

In [55]:

```
1 g
```

Out[55]:

```
{5, 6, 7, 8, 9, 10, 11}
```

In [56]:

```
1 g.pop()
```

Out[56]:

```
5
```

In [57]:

```
1 g.pop(8)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call
 last)
<ipython-input-57-d134420af6f9> in <module>
----> 1 g.pop(8)

TypeError: pop() takes no arguments (1 given)
```

In [61]:

```
alp = 'a b c d e f g h i j k l m n o p q r s t u v w x y z'.split()
alps = set(alp)
alps
```

Out[61]:

```
{'a',
 'b',
 'c',
 'd',
 'e',
 'f',
 'g',
 'h',
 'i',
 'j',
 'k',
 'l',
 'm',
 'n',
 'o',
 'p',
 'q',
 'r',
 's',
 't',
 'u',
 'v',
 'w',
 'x',
 'y',
 'z'}
```

In [59]:

```
vow = {'a','e','i','o','u'}
vow
```

Out[59]:

```
{'a', 'e', 'i', 'o', 'u'}
```

In [62]:

```
vow.issubset(alps)
```

Out[62]:

```
True
```

In [63]:

```
alps.issubset(vow)
```

Out[63]:

```
False
```

In [64]:

```
alps.issuperset(vow)
```

Out[64]:

```
True
```

subset (set) ------> set(superset)(subset)

In [65]:

```
a
```

Out[65]:

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

In [66]:

```
b
```

Out[66]:

```
{4, 5, 6, 7}
```

In [67]:

```
a & b
```

Out[67]:

```
{4}
```

In [68]:

```
a.intersection_update(b)
```

In [69]:

```
b
```

Out[69]:

```
{4, 5, 6, 7}
```

In [70]:

```
a
```

Out[70]:

```
{4}
```

In [71]:

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

In [72]:

```
a.difference_update(b)
```

In [73]:

```
a
```

Out[73]:

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

In [74]:

```
a.add(4)
```

In [77]:

```
a.symmetric_difference_update(b)
a
```

Out[77]:

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

In [78]:

```
l = a.copy()
```

In [79]:

```
l
```

Out[79]:

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

In [80]:

```
l.clear()
```

In [81]:

```
l
```

Out[81]:

```
set()
```