Dictionaries

- It stores collection of various types of data
- Dictionaries are unordered and changeable.
- Dictionaries have pair of keys and values which is seperated with ':'
- Keys are act as index of values in dictionary
- It is represented as flower brackets. Syntax: dict = {'key1':'vlaue1','key2':'value2'}

In [1]:

```
dic = {'name':'reddy','age':22,'grade':'A','Phno':123456}
print(dic)
```

```
{'name': 'reddy', 'age': 22, 'grade': 'A', 'Phno': 123456}
```

In [2]:

```
dic['grade']
```

Out[2]:

```
'A'
```

In [3]:

```
dic['Phno']
```

Out[3]:

```
123456
```

In [4]:

```
dic['Phno'] = 9786754534
```

In [5]:

```
dic
```

Out[5]:

```
{'name': 'reddy', 'age': 22, 'grade': 'A', 'Phno': 9786754534}
```

In [6]:

```
print(dir(dic))
```

```
['__class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__do
c__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__',
'__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__',
'__len__', '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__
repr__', '__reversed__', '__setattr__', '__setitem__', '__sizeof__', '__str_
_', '__subclasshook__', 'clear', 'copy', 'fromkeys', 'get', 'items', 'keys',
'pop', 'popitem', 'setdefault', 'update', 'values']
```

In [7]:

```
dic['Phno']
```

Out[7]:

```
9786754534
```

In [8]:

```
# get()
dic.get('Phno')
```

Out[8]:

```
9786754534
```

In [9]:

```
# keys()
dic.keys()
```

Out[9]:

```
dict_keys(['name', 'age', 'grade', 'Phno'])
```

In [10]:

```
# values()
dic.values()
```

Out[10]:

```
dict_values(['reddy', 22, 'A', 9786754534])
```

In [11]:

```
# items()
dic.items()
```

Out[11]:

```
dict_items([('name', 'reddy'), ('age', 22), ('grade', 'A'), ('Phno', 9786754
534)])
```

In [12]:

```
# setdefalut()
dic.setdefault('addr','guntur')
```

Out[12]:

```
'guntur'
```

In [14]:

```
print(dic)
```

```
{'name': 'reddy', 'age': 22, 'grade': 'A', 'Phno': 9786754534, 'addr': 'gunt
ur'}
```

In [16]:

```
dic.setdefault('marks')
```

In [18]:

```
print(dic)
```

```
{'name': 'reddy', 'age': 22, 'grade': 'A', 'Phno': 9786754534, 'addr': 'gunt
ur', 'marks': None}
```

In [21]:

```
dic['marks'] = 560
```

In [23]:

```
print(dic)
```

```
{'name': 'reddy', 'age': 22, 'grade': 'A', 'Phno': 9786754534, 'addr': 'gunt
ur', 'marks': 560}
```

In [24]:

```
# update()
dic.update({'marks':780})
```

In [25]:

```
print(dic)
```

```
{'name': 'reddy', 'age': 22, 'grade': 'A', 'Phno': 9786754534, 'addr': 'gunt
ur', 'marks': 780}
```

In [26]:

```
dic.update({'rollno':2210,'projectid':123})
```

In [28]:

```
print(dic)
```

```
{'name': 'reddy', 'age': 22, 'grade': 'A', 'Phno': 9786754534, 'addr': 'gunt
ur', 'marks': 780, 'rollno': 2210, 'projectid': 123}
```

In [30]:

```
# pop()
dic.pop('marks')
```

Out[30]:

```
780
```

In [31]:

```
dic
```

Out[31]:

```
{'name': 'reddy',
 'age': 22,
 'grade': 'A',
 'Phno': 9786754534,
 'addr': 'guntur',
 'rollno': 2210,
 'projectid': 123}
```

In [32]:

```
# popitem()
dic.popitem()
```

Out[32]:

```
('projectid', 123)
```

In [33]:

```
dic
```

Out[33]:

```
{'name': 'reddy',
 'age': 22,
 'grade': 'A',
 'Phno': 9786754534,
 'addr': 'guntur',
 'rollno': 2210}
```

In [34]:

```
# fromkeys()
x = ('key1','key2','key3')
dict.fromkeys(x)
```

Out[34]:

```
{'key1': None, 'key2': None, 'key3': None}
```

In [35]:

```
x = ('key1','key2','key3')
y = 0
dict.fromkeys(x, y)
```

Out[35]:

```
{'key1': 0, 'key2': 0, 'key3': 0}
```

In [44]:

```
x = ('key1','key2','key3')
y = (1,2,3)
dict2 = dict.fromkeys(x, y)
```

In [46]:

```
dict2
```

Out[46]:

```
{'key1': (1, 2, 3), 'key2': (1, 2, 3), 'key3': (1, 2, 3)}
```

In [47]:

```
dict2.update({'key1':1})
```

In [48]:

```
dict2
```

Out[48]:

```
{'key1': 1, 'key2': (1, 2, 3), 'key3': (1, 2, 3)}
```

In [49]:

```
dict2.update({'key2':2,'key3':3})
```

In [50]:

```
dict2
```

Out[50]:

```
{'key1': 1, 'key2': 2, 'key3': 3}
```

In [51]:

```
#clear()
dict2.clear()
```

In [52]:

```
dict2
```

Out[52]:

```
{}
```

Nested Dictionary

- Dictionary of list
- Dictionary of dictionary

In [53]:

```
# Dictionary of list

dict1 = {'std1':[120,'xyz','cse'],'std2':[121,'abc','ece'],'std3':[123,'mno','civil']}
dict1
```

Out[53]:

```
{'std1': [120, 'xyz', 'cse'],
 'std2': [121, 'abc', 'ece'],
 'std3': [123, 'mno', 'civil']}
```

In [54]:

```
dict1.get('std2')
```

Out[54]:

```
[121, 'abc', 'ece']
```

In [55]:

```
dict1['std2']
```

Out[55]:

```
[121, 'abc', 'ece']
```

In [56]:

```
dict1['std2'][0]
```

Out[56]:

```
121
```

In [57]:

```
dict1['std2'][2]
```

Out[57]:

```
'ece'
```

In [61]:

```
# Dictionary of dictionary
dict1 = {'std1':{120,'xyz','cse'},'std2':{121,'abc','ece'},'std3':{123,'mno','civil'}}
print(dict1)
```

```
{'std1': {120, 'xyz', 'cse'}, 'std2': {'ece', 121, 'abc'}, 'std3': {'civil',
123, 'mno'}}
```

In [62]:

```
dict1['std3']
```

Out[62]:

```
{123, 'civil', 'mno'}
```

In [66]:

```python
# input : lst = [1,3,1,2,3,3,2,4,1,1,4]
# output : {1:4, 3:3, 2:2, 4:2}

n = int(input("Enter length of the list: "))
lst = []
for i in range(n):
    v = int(input("Enter value: "))
    lst.append(v)
print(lst)
dic = {}
for i in lst: #i=1, i=3, i=1, i=2, i=3, i=3, i=2,i=4
    if i not in dic:
        dic[i] = 1 # {1:1,3:1,2:1,4:1}
    else:
        dic[i] += 1 # dic[i]=dic[i]+1 --> {1:4,3:3,2:2,4:2}
print(dic)
```

```
Enter length of the list: 11
Enter value: 1
Enter value: 2
Enter value: 1
Enter value: 3
Enter value: 3
Enter value: 1
Enter value: 2
Enter value: 4
Enter value: 4
Enter value: 2
Enter value: 1
[1, 2, 1, 3, 3, 1, 2, 4, 4, 2, 1]
{1: 4, 2: 3, 3: 2, 4: 2}
```

In [65]:

```python
n = int(input("Enter length of the list: "))
lst = []
for i in range(n):
    v = int(input("Enter value: "))
    lst.append(v)
lst
```

```
Enter length of the list: 5
Enter value: 1
Enter value: 2
Enter value: 3
Enter value: 4
Enter value: 5
```

Out[65]:

```
[1, 2, 3, 4, 5]
```

In [67]:

```
n = int(input("Enter length of the list: "))
lst = []
for i in range(n):
    v = int(input("Enter value: "))
    lst.append(v)
lst
dic = {}
for i in lst:
    dic[i] = lst.count(i)
print(dic)
```

```
Enter length of the list: 5
Enter value: 3
Enter value: 4
Enter value: 3
Enter value: 3
Enter value: 4
{3: 3, 4: 2}
```

Set

- A set is collection of data type that is iterable.
- It is murable(not changeable).
- Set class represents the mathematical notation of a set.

In [68]:

```
s = set()
s
```

Out[68]:

```
set()
```

In [69]:

```
type(s)
```

Out[69]:

```
set
```

In [70]:

```
d = dict()
type(d)
```

Out[70]:

```
dict
```

In [72]:

```
set1 = {4,6,2,9,7,1,3,4,1}
print(set1)
```

```
{1, 2, 3, 4, 6, 7, 9}
```

In [73]:

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

```
['__and__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc_
_', '__eq__', '__format__', '__ge__', '__getattribute__', '__gt__', '__hash_
_', '__iand__', '__init__', '__init_subclass__', '__ior__', '__isub__', '__i
ter__', '__ixor__', '__le__', '__len__', '__lt__', '__ne__', '__new__', '__o
r__', '__rand__', '__reduce__', '__reduce_ex__', '__repr__', '__ror__', '__r
sub__', '__rxor__', '__setattr__', '__sizeof__', '__str__', '__sub__', '__su
bclasshook__', '__xor__', 'add', 'clear', 'copy', 'difference', 'difference_
update', 'discard', 'intersection', 'intersection_update', 'isdisjoint', 'is
subset', 'issuperset', 'pop', 'remove', 'symmetric_difference', 'symmetric_d
ifference_update', 'union', 'update']
```

In [74]:

```
set1
```

Out[74]:

```
{1, 2, 3, 4, 6, 7, 9}
```

In [75]:

```
set1[0]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-75-c38563f1af7a> in <module>
----> 1 set1[0]

TypeError: 'set' object is not subscriptable
```

In [76]:

```
# add()
set1.add(13)
```

In [77]:

```
print(set1)
```

```
{1, 2, 3, 4, 6, 7, 9, 13}
```

In [78]:

```
# update()
set1.update([8,10,12])
```

In [80]:

```
print(set1)
```

```
{1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13}
```

In [81]:

```
# discard()
set1.discard(9)
```

In [82]:

```
print(set1)
```

```
{1, 2, 3, 4, 6, 7, 8, 10, 12, 13}
```

In [83]:

```
set1[4] = 15
print(set1)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-83-41e7b56ac10b> in <module>
----> 1 set1[4] = 15
      2 print(set1)

TypeError: 'set' object does not support item assignment
```

In [84]:

```
set1
```

Out[84]:

```
{1, 2, 3, 4, 6, 7, 8, 10, 12, 13}
```

In [85]:

```
set1.discard(11)
```

In [86]:

```
set1
```

Out[86]:

```
{1, 2, 3, 4, 6, 7, 8, 10, 12, 13}
```

In [87]:

```
# remove()
set1.remove(10)
```

In [88]:

```
set1
```

Out[88]:

```
{1, 2, 3, 4, 6, 7, 8, 12, 13}
```

In [89]:

```
set1.remove(20)
set1
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-89-352040a58872> in <module>
----> 1 set1.remove(20)
      2 set1

KeyError: 20
```

In [90]:

```
set1
```

Out[90]:

```
{1, 2, 3, 4, 6, 7, 8, 12, 13}
```

In [91]:

```
# pop()
set1.pop()
set1
```

Out[91]:

```
{2, 3, 4, 6, 7, 8, 12, 13}
```

In [93]:

```
set1.pop()
```

Out[93]:

```
2
```

In [94]:

```
set1
```

Out[94]:

```
{3, 4, 6, 7, 8, 12, 13}
```

In [95]:

```
# clear()
set1.clear()
```

In [96]:

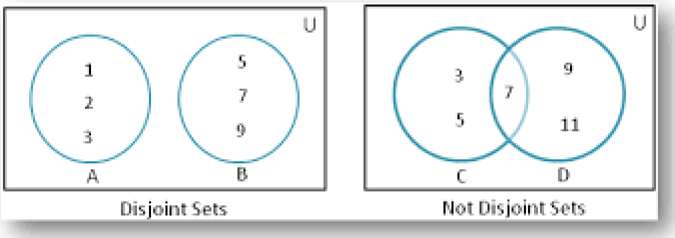
```
set1
```

Out[96]:

```
set()
```

Disjoint

- Two sets are said to be disjoint if they don't have any common elements.



In [98]:

```
# isdisjoint()

A = {1,2,3}
B = {5,7,9}
print(A.isdisjoint(B))
print(B.isdisjoint(A))
```

```
True
True
```

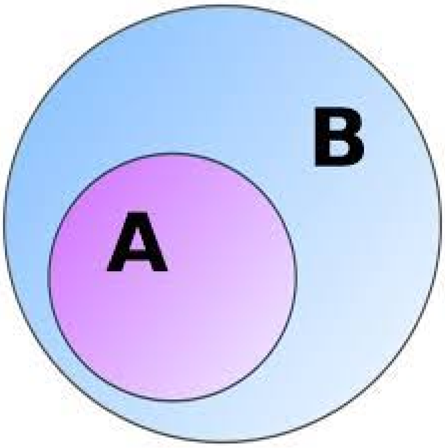
In [99]:

```
A = {3,7,5}
B = {6,7,9}
print(A.isdisjoint(B))
print(B.isdisjoint(A))
```

```
False
False
```

Superset

- Set B is said to be the superset of set A if all elements of A are in set B.



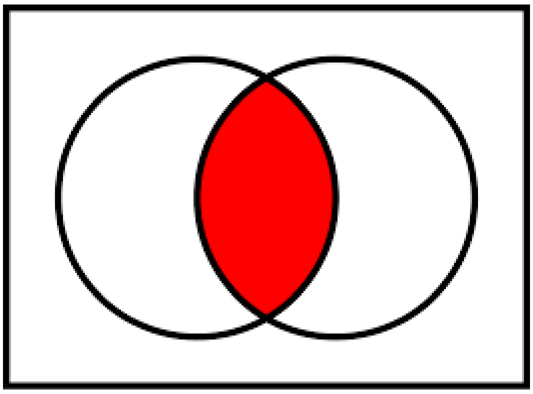
In [101]:

```
# issuperset()
A = {1,3,5,7,9} # superset
B = {1,5,9} # subset
print(A.issuperset(B))
print(B.issuperset(A))
```

```
True
False
```

Intersection()

- The intersection() method returns a set that contains the similarity between two or more sets.



In [102]:

```
A = {1,3,5,7,9}
B = {1,5,8,10}
print(A.intersection(B))
```

```
{1, 5}
```

In [103]:

```
print(A)
print(B)
```

```
{1, 3, 5, 7, 9}
{8, 1, 10, 5}
```

In [104]:

```
# intersection_update()
A = {1,3,5,7,9}
B = {1,5,8,10}
print(A.intersection_update(B))
```

```
None
```

In [105]:

```
print(A)
```

```
{1, 5}
```

In [106]:

```
print(B)
```

```
{8, 1, 10, 5}
```

In [107]:

```
print(B.intersection_update(A))
```

```
None
```

In [108]:

```
print(B)
```

```
{1, 5}
```

In [109]:

```
print(A)
print(B)
```

```
{1, 5}
{1, 5}
```

In [ ]:

```

```