In [1]:

```
"""Dictionary is an unorder collection of items. Its contains the data using key-value
 pair"""
```

Out[1]:

```
'Dictionary is an unorder collection of items. Its contains the data using
key-value pair'
```

In [2]:

```
#dictionary with simple key-value
student = {"name" : "Rahat", "id": 24066, "age" : 26.8}
print(student)
```

```
{'name': 'Rahat', 'id': 24066, 'age': 26.8}
```

In [3]:

```
#dictionary with complex data as value

complex_dict = {"name" : "Adi", "results" : [('math', 3.5), ('phy', 4)]}
print(complex_dict)
```

```
{'name': 'Adi', 'results': [('math', 3.5), ('phy', 4)]}
```

In [5]:

```
#dictionary with no data

empty_dict = dict()
print(empty_dict)
```

```
{}
```

In [6]:

```
#dictionary creation with List

dict_list = dict([("Rahat", 3.68), ("Baser", 3.86)])
print(dict_list)
```

```
{'Rahat': 3.68, 'Baser': 3.86}
```

In [7]:

```
#dictionary access

print(dict_list["Rahat"])
```

```
3.68
```

In [9]:

```
"""If key is not found"""

print(dict_list["rahat"])
```

```
---------------------------------------------------------------------------
-
KeyError                                  Traceback (most recent call las
t)
<ipython-input-9-b769c8297942> in <module>()
      1 """If key is not found"""
      2
----> 3 print(dict_list["rahat"])

KeyError: 'rahat'
```

In [10]:

```
#dictionary access with get method

print(dict_list.get("Rahat"))
```

```
3.68
```

In [11]:

```
print(dict_list.get("rahat")) #the key is not available in dict.
```

```
None
```

In [13]:

```
"""We can access any element by dict[key] or dict.get() function. If the key is not ava
ilable, then direct access make key exception. On the other hand get function couldn't
 through exception instead of return a None"""
```

Out[13]:

```
"We can access any element by dict[key] or dict.get() function. If the key
is not available, then direct access make key exception. On the other hand
get function couldn't through exception instead of return a None"
```

In [15]:

```
#adding element in dictionary

dict_list["hasina"] = 2.74
print(dict_list)
```

```
{'Rahat': 3.68, 'Baser': 3.86, 'hasina': 2.74}
```

In [16]:

```
#updating element in dictionary

dict_list["hasina"] = 2.28
print(dict_list)
```

```
{'Rahat': 3.68, 'Baser': 3.86, 'hasina': 2.28}
```

In [17]:

```
#deleting an item from dictionary using pop function

dict_list.pop("Baser")
print(dict_list)
```

```
{'Rahat': 3.68, 'hasina': 2.28}
```

In [23]:

```
dict_list.pop("Baser")

"""If the item is not in the dictionary. it will through exception"""
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last
t)
<ipython-input-23-84e989cf00f3> in <module>()
----> 1 dict_list.pop("Baser")
      2
      3 """If the item is not in the dictionary. it will through exceptio
n"""

KeyError: 'Baser'
```

In [24]:

```
del dict_list["Baser"]

"""If the item is not in the dictionary. it will through exception"""
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last
t)
<ipython-input-24-4cdc8d87b5c5> in <module>()
----> 1 del dict_list["Baser"]

KeyError: 'Baser'
```

In [26]:

```
#deleting all items using clear

dict_list.clear()
print(dict_list)
```

```
{}
```

In [27]:

```
#deleting all items using del function

dict_list = dict([("Rahat", 3.68), ("Baser", 3.86)])
print(dict_list)

del dict_list
print(dict_list)
```

```
{'Rahat': 3.68, 'Baser': 3.86}


---------------------------------------------------------------------------
-
NameError                                 Traceback (most recent call las
t)
<ipython-input-27-0d51e1fff17a> in <module>()
      5
      6 del dict_list
----> 7 print(dict_list)

NameError: name 'dict_list' is not defined
```

In [29]:

```
"""Major difference between del function and clear is: clear remove all the elements an
d make the dictionary empty otherwise del function remove the dictionary from the memor
y location"""
```

Out[29]:

```
'Major difference between del function and clear is: clear remove all the
elements and make the dictionary empty otherwise del function remove the d
ictionary from the memory location'
```

In [31]:

```
"""Dictionary Methods..."""

square = {2: 4, 3: 9, 4: 16, 5: 25}
```

In [32]:

```
#copy method

new_dict = square.copy()
print(new_dict)
```

```
{2: 4, 3: 9, 4: 16, 5: 25}
```

In [34]:

```
#fromkeys(sequence, value)

new_dict = {}.fromkeys(['rahat', 'munna', 'munni'], 0)
print(new_dict)
```

```
{'rahat': 0, 'munna': 0, 'munni': 0}
```

In [36]:

```
#items function return all the elemennts and their value as the list of pair(tuple)

print(square.items())
```

```
dict_items([(2, 4), (3, 9), (4, 16), (5, 25)])
```

In [37]:

```
#keys function return all the keys in a list

print(square.keys())
```

```
dict_keys([2, 3, 4, 5])
```

In [38]:

```
#values function return all the values in a list

print(square.values())
```

```
dict_values([4, 9, 16, 25])
```

In [39]:

```
#get all the build in methods of dictionary

d = {}
print(dir(d))
```

```
['__class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__
doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem_
_', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__
le__', '__len__', '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_e
x__', '__repr__', '__setattr__', '__setitem__', '__sizeof__', '__str__',
'__subclasshook__', 'clear', 'copy', 'fromkeys', 'get', 'items', 'keys',
'pop', 'popitem', 'setdefault', 'update', 'values']
```

In [43]:

```
"""Dictionary Comprehension"""

st_dict = {"a" : 1, "b": 2, "c" : 3, "d": 4}
```

In [45]:

```
#create dict using loop

new_dict = {key:value for key, value in st_dict.items() if value%2 == 0}
print(new_dict)
```

```
{'b': 2, 'd': 4}
```