Dict Category Data Type (Collection Data Types)

In [ ]:
```
> dict is the pre-defined class the pupose of the dict data type is that " to store
> An object of dict is maintains insertion order
> An object of dict is never contions indices. so that we can't perform the both in
> To store the data ---> (Key,value)
> key value ---> Dict class
> symbolic notations ---> {}
> syntax name ---> {(key1:val1,key2:val2,....keyn:valn)}
                   key1,key2....keyn are called value of key --> they must be alwa
                   val1,val2,...val-n are represents values of values
> An object of dict belongs to mutable. In details the values of key belongs to imm
> In the python programming we can creat two types of dict objects:
    a).Empty dict
    b).Non-empty dict
```

In [ ]:
```
INDEX:
-----
> What is the dict
> Purpose of the dict
> Notations for dict
> syntax for creating dict
> Types of dict objects
> Memory Management of dict
> Pre-defined fuctions dict
> Programming Examples
> Inner OR Nested dict
> Pre-defined fuctions in Nested or inner dict
> Combinations of dict with set, list and tuple
> Programming examples
```

In [4]:
```
d1={10:1.2,20:3.5,30:1.2, 40:4.5}
print(d1,type(d1))
```

```
{10: 1.2, 20: 3.5, 30: 1.2, 40: 4.5} <class 'dict'>
```

In [12]:
```
d2={10:"Mahaboob",20:"MRIIRS",30:"CDOE", 40:"Khan"}
print(d2,type(d2))
```

```
{10: 'Mahaboob', 20: 'MRIIRS', 30: 'CDOE', 40: 'Khan'} <class 'dict'>
```

In [10]:
```
len(d1)
```

Out[10]: 4

In [14]:
```
len(d2)
```

Out[14]: 4

In [16]:
```
d3={10:1.2, 10:2.3, 10:3.4, 10:5.6}
print(d3,type(d3))
```

```
{10: 5.6} <class 'dict'>
```

In [18]:
```
d2={10:"Mahaboob",20:"MRIIRS",30:"CDOE", 40:"Khan"}
print(d2,type(d2))
d2[0]
```

```
{10: 'Mahaboob', 20: 'MRIIRS', 30: 'CDOE', 40: 'Khan'} <class 'dict'>
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[18], line 3
      1 d2={10:"Mahaboob",20:"MRIIRS",30:"CDOE", 40:"Khan"}
      2 print(d2,type(d2))
----> 3 d2[0]

KeyError: 0
```

In [20]:
```
d2={10:"Mahaboob",20:"MRIIRS",30:"CDOE", 40:"Khan"}
d2[10] #key itself consider of an index
```

Out[20]: 'Mahaboob'

In [22]:
```
d2[20] #key itself consider of an index
```

Out[22]: 'MRIIRS'

In [24]:
```
d2[30] #key itself consider of an index
```

Out[24]: 'CDOE'

In [4]:
```
d2={10:"Mahaboob",20:"MRIIRS",30:"CDOE", 40:"Khan"}
print(d2,type(d2))
for x,y in enumerate(d2):
    print(x,"--->",y)
```

```
{10: 'Mahaboob', 20: 'MRIIRS', 30: 'CDOE', 40: 'Khan'} <class 'dict'>
0 ---> 10
1 ---> 20
2 ---> 30
3 ---> 40
```

In [6]:
```
d1={10:1.2,20:3.5,30:1.2, 40:4.5}
print(d1,type(d1))
for x,y in enumerate(d1):
    print(x,"--->",y)
```

```
{10: 1.2, 20: 3.5, 30: 1.2, 40: 4.5} <class 'dict'>
0 ---> 10
1 ---> 20
2 ---> 30
3 ---> 40
```

In [8]:
```
d1={}
print(d1,type(d1))
len(d1)
```

```
{} <class 'dict'>
```

Out[8]: 0

In [10]:
```
# How to add data in dict (key,value empty)
#dictobj [key]=value
d1[100]=1.2
d1[101]=2.3
d1[102]=3.4
d1[103]=4.5
print(d1,type(d1))
```

```
{100: 1.2, 101: 2.3, 102: 3.4, 103: 4.5} <class 'dict'>
```

In [16]:
```
d1[100]=12.34 # modifiding the value of key[100]
print(d1,type(d1))
```

```
{100: 12.34, 101: 2.3, 102: 3.4, 103: 4.5} <class 'dict'>
```

In [18]:
```
d1[200]=4.8 #adding the entery
print(d1,type(d1))
```

```
{100: 12.34, 101: 2.3, 102: 3.4, 103: 4.5, 200: 4.8} <class 'dict'>
```

Pre-defined fuctions dict:

> We know that in a dictionary object, we learned how to insert key-value pairs and how to modify the value by passing the key. Along with these operations, we can also perform different actions using predefined functions of a dictionary object.

In [8]:
```
#Clear()
d1={10:1.2,20:2.3,30:3.4,40:4.5}
d1.clear()
print(d1,type(d1))
```

```
{} <class 'dict'>
```

In [10]:
```
len(d1)
```

Out[10]:
```
0
```

In [12]:
```
print({}.clear())
```

```
None
```

In [14]:
```
print(dict().clear())
```

```
None
```

In [16]:
```
# Copy()
d1={10:1.2,20:2.3,30:3.4,40:4.5}
print(d1,type(d1),id(d1))
```

```
{10: 1.2, 20: 2.3, 30: 3.4, 40: 4.5} <class 'dict'> 1566288452800
```

In [18]:
```
d2=d1.copy()
print(d2,type(d2),id(d2))
```

```
{10: 1.2, 20: 2.3, 30: 3.4, 40: 4.5} <class 'dict'> 1566308446656
```

In [22]:
```
#pop()
d1={10:1.2,20:2.3,30:3.4,40:4.5}
```

```
d1.pop(30)
print(d1,type(d1),id(d1))
```

```
{10: 1.2, 20: 2.3, 40: 4.5} <class 'dict'> 1566322495040
```

In [24]:
```
d1.pop(10)
print(d1,type(d1),id(d1))
```

```
{20: 2.3, 40: 4.5} <class 'dict'> 1566322495040
```

In [26]:
```
d1.pop(40)
print(d1,type(d1),id(d1))
```

```
{20: 2.3} <class 'dict'> 1566322495040
```

In [28]:
```
#popitem()
d1={10:1.2,20:2.3,30:3.4,40:4.5}
d1.popitem() # when i don't pass any key here, it always removes the last element
print(d1,type(d1),id(d1))
```

```
{10: 1.2, 20: 2.3, 30: 3.4} <class 'dict'> 1566322583104
```

In [30]:
```
d1.popitem() # when i don't pass any key here, it always removes the last element
print(d1,type(d1),id(d1))
```

```
{10: 1.2, 20: 2.3} <class 'dict'> 1566322583104
```

In [32]:
```
d1.popitem() # when i don't pass any key here, it always removes the last element
print(d1,type(d1),id(d1))
```

```
{10: 1.2} <class 'dict'> 1566322583104
```

In [36]:
```
#get()
d1={10:1.2,20:2.3,30:3.4,40:4.5}
x=d1.get(10)
print(x)
```

```
1.2
```

In [38]:
```
x=d1.get(20)
print(x)
```

```
2.3
```

In [40]:
```
x=d1.get(30)
print(x)
```

```
3.4
```

In [42]:
```
x=d1.get(40)
print(x)
```

```
4.5
```

In [44]:
```
x=d1.get(100) #
print(x)
```

```
None
```

In [52]:
```
# keys(): This function is used to obtain all the keys from a dictionary.
# The result is stored in the variable varname, and its type is <class 'dict_keys'>
```

```
d1={10:1.2,20:2.3,30:3.4,40:4.5}
ks=d1.keys()
print(ks,type(ks))
```

```
dict_keys([10, 20, 30, 40]) <class 'dict_keys'>
```

In [48]:
```
for k in ks:
    print(k)
```

```
10
20
30
40
```

In [54]:
```
# values(): This function is used to obtain all the values from a dictionary.
# The result is stored in the variable varname, and its type is <class 'dict_values
d1={10:1.2,20:2.3,30:3.4,40:4.5}
vs=d1.values()
print(vs,type(vs))
```

```
dict_values([1.2, 2.3, 3.4, 4.5]) <class 'dict_values'>
```

In [58]:
```
for v in vs:
    print(v)
```

```
1.2
2.3
3.4
4.5
```

In [60]:
```
for v in d1.values():
    print(v)
```

```
1.2
2.3
3.4
4.5
```

In [62]:
```
#items(): This function is used to obtain all key-value pairs from a dictionary.
# The result is stored in the variable varname, and its type is <class 'dict_items'
d1={10:1.2,20:2.3,30:3.4,40:4.5}
MRIIRS=d1.items()
print(MRIIRS,type(MRIIRS))
```

```
dict_items([(10, 1.2), (20, 2.3), (30, 3.4), (40, 4.5)]) <class 'dict_items'>
```

In [64]:
```
for CDOE in MRIIRS:
    print(CDOE)
```

```
(10, 1.2)
(20, 2.3)
(30, 3.4)
(40, 4.5)
```

In [ ]: