Dictionary Data Structure

- Rules and guidelines of dictionary data structure
  - Dictionary are key-pair elements just like JSON
  - Dictionary is listed by {}
  - Each element is the combination of key : value pair
  - Dictionary keys are immutable and Key's values can be mutable or immutable
  - Dictionary can be created using {} and dict function.
  - We can not have duplicate key in the dictionary

In [1]:
```
print("All Functions from Dictionary: ",[i for i in dir(dict) if "__" not in i])
```

```
All Functions from Dictionary:  ['clear', 'copy', 'fromkeys', 'get', 'items', 'key
s', 'pop', 'popitem', 'setdefault', 'update', 'values']
```

In [2]:
```
dct = {}
```

In [3]:
```
type(dct)
```

Out[3]:
```
dict
```

In [4]:
```
d = dict()
```

In [5]:
```
type(d)
```

Out[5]:
```
dict
```

In [6]:
```
employeeDB = {"EmpID" : [101,102,103,104,105],
              "EmpName" : ["A","B","C","D"],
              "City" : ["Houson","Tempa","Delhi",'Chennai'],
              "Grade" : ["A","B","A","B"],
              "Salary" : [100,200,350,230]}
```

In [7]:
```
employeeDB
```

Out[7]:
```
{'EmpID': [101, 102, 103, 104, 105],
 'EmpName': ['A', 'B', 'C', 'D'],
 'City': ['Houson', 'Tempa', 'Delhi', 'Chennai'],
 'Grade': ['A', 'B', 'A', 'B'],
 'Salary': [100, 200, 350, 230]}
```

In [8]:
```
# can you display only the keys
```

In [9]:
```
employeeDB.keys()
```
Out[9]:
```
dict_keys(['EmpID', 'EmpName', 'City', 'Grade', 'Salary'])
```
In [10]:
```
# can you display the values
```
In [11]:
```
employeeDB.values()
```
Out[11]:
```
dict_values([[101, 102, 103, 104, 105], ['A', 'B', 'C', 'D'], ['Houson', 'Tempa',
'Delhi', 'Chennai'], ['A', 'B', 'A', 'B'], [100, 200, 350, 230]])
```
In [12]:
```
# can you display the key and its values
```
In [13]:
```
employeeDB.items() # returning tuple, where first item is key and 2nd item are
```
Out[13]:
```
dict_items([('EmpID', [101, 102, 103, 104, 105]), ('EmpName', ['A', 'B', 'C',
'D']), ('City', ['Houson', 'Tempa', 'Delhi', 'Chennai']), ('Grade', ['A', 'B',
'A', 'B']), ('Salary', [100, 200, 350, 230])])
```
In [14]:
```
# can you display the value of specific key
```
In [15]:
```
employeeDB['Salary']
```
Out[15]:
```
[100, 200, 350, 230]
```
In [16]:
```
employeeDB["EmpID"]
```
Out[16]:
```
[101, 102, 103, 104, 105]
```
In [17]:
```
employeeDB["EmpName"]
```
Out[17]:
```
['A', 'B', 'C', 'D']
```
In [18]:
```
employeeDB["City"]
```
Out[18]:
```
['Houson', 'Tempa', 'Delhi', 'Chennai']
```
In [19]:
```
employeeDB["Grade"]
```
Out[19]:
```
['A', 'B', 'A', 'B']
```

In [20]:
```
employeeDB
```
Out[20]:
```
{'EmpID': [101, 102, 103, 104, 105],
 'EmpName': ['A', 'B', 'C', 'D'],
 'City': ['Houson', 'Tempa', 'Delhi', 'Chennai'],
 'Grade': ['A', 'B', 'A', 'B'],
 'Salary': [100, 200, 350, 230]}
```
In [21]:
```
import pandas as pd
```
In [23]:
```
employeeDB = {"EmpID" : [101,102,103,104,105],
              "EmpName" : ["A","B","C","D","E"],
              "City" : ["Houson","Tempa","Delhi",'Chennai',"Panjim"],
              "Grade" : ["A","B","A","B","A"],
              "Salary" : [100,200,350,230,150]}
```
In [24]:
```
df = pd.DataFrame(employeeDB)
```
In [25]:
```
df
```
Out[25]:

	EmpID	EmpName	City	Grade	Salary
0	101	A	Houson	A	100
1	102	B	Tempa	B	200
2	103	C	Delhi	A	350
3	104	D	Chennai	B	230
4	105	E	Panjim	A	150

In [26]:
```
df.to_csv("Employee.csv", index = False)
```
In [27]:
```
employeeDB
```
Out[27]:
```
{'EmpID': [101, 102, 103, 104, 105],
 'EmpName': ['A', 'B', 'C', 'D', 'E'],
 'City': ['Houson', 'Tempa', 'Delhi', 'Chennai', 'Panjim'],
 'Grade': ['A', 'B', 'A', 'B', 'A'],
 'Salary': [100, 200, 350, 230, 150]}
```
In [28]:
```
dct = dict()
```
In [29]:
```
dct["A"] = 10,20,30,40
```

```
In [30]: dct
Out[30]: {'A': (10, 20, 30, 40)}

In [31]: dct["D"] = ["Abhishek","Ajeet"]

In [32]: dct
Out[32]: {'A': (10, 20, 30, 40), 'D': ['Abhishek', 'Ajeet']}

In [33]: dct["X"] = employeeDB

In [34]: dct
Out[34]: {'A': (10, 20, 30, 40),
          'D': ['Abhishek', 'Ajeet'],
          'X': {'EmpID': [101, 102, 103, 104, 105],
           'EmpName': ['A', 'B', 'C', 'D', 'E'],
           'City': ['Houson', 'Tempa', 'Delhi', 'Chennai', 'Panjim'],
           'Grade': ['A', 'B', 'A', 'B', 'A'],
           'Salary': [100, 200, 350, 230, 150]}}

In [35]: dct["Z"] = {"Name" : "Dharmendra","son" : ["Sunny","Bobby"],"Pet" : ["Cat","Dog

In [36]: dct
Out[36]: {'A': (10, 20, 30, 40),
          'D': ['Abhishek', 'Ajeet'],
          'X': {'EmpID': [101, 102, 103, 104, 105],
           'EmpName': ['A', 'B', 'C', 'D', 'E'],
           'City': ['Houson', 'Tempa', 'Delhi', 'Chennai', 'Panjim'],
           'Grade': ['A', 'B', 'A', 'B', 'A'],
           'Salary': [100, 200, 350, 230, 150]},
          'Z': {'Name': 'Dharmendra', 'son': ['Sunny', 'Bobby'], 'Pet': ['Cat', 'Dog']}}

In [37]: dct["Z"]
Out[37]: {'Name': 'Dharmendra', 'son': ['Sunny', 'Bobby'], 'Pet': ['Cat', 'Dog']}

In [39]: dct["Name"]
         ---------------------------------------------------------------------------
         KeyError                                  Traceback (most recent call last)
         Cell In[39], line 1
         ----> 1 dct["Name"]

         KeyError: 'Name'
```

```
In [41]: dct.keys()

Out[41]: dict_keys(['A', 'D', 'X', 'Z'])

In [42]: dct.get('Name')

In [43]: employeeDB

Out[43]: {'EmpID': [101, 102, 103, 104, 105],
          'EmpName': ['A', 'B', 'C', 'D', 'E'],
          'City': ['Houson', 'Tempa', 'Delhi', 'Chennai', 'Panjim'],
          'Grade': ['A', 'B', 'A', 'B', 'A'],
          'Salary': [100, 200, 350, 230, 150]}

In [44]: employeeDB["Address"]

---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[44], line 1
----> 1 employeeDB["Address"]

KeyError: 'Address'

In [45]: employeeDB.get("Address")

In [46]: employeeDB.get("City")

Out[46]: ['Houson', 'Tempa', 'Delhi', 'Chennai', 'Panjim']

In [47]: employeeDB

Out[47]: {'EmpID': [101, 102, 103, 104, 105],
          'EmpName': ['A', 'B', 'C', 'D', 'E'],
          'City': ['Houson', 'Tempa', 'Delhi', 'Chennai', 'Panjim'],
          'Grade': ['A', 'B', 'A', 'B', 'A'],
          'Salary': [100, 200, 350, 230, 150]}

In [49]: employeeDB['EmpID'].append(999)

In [50]: employeeDB

Out[50]: {'EmpID': [101, 102, 103, 104, 105, 999],
          'EmpName': ['A', 'B', 'C', 'D', 'E'],
          'City': ['Houson', 'Tempa', 'Delhi', 'Chennai', 'Panjim'],
          'Grade': ['A', 'B', 'A', 'B', 'A'],
          'Salary': [100, 200, 350, 230, 150]}
```

```
In [53]: employeeDB['City'][-3] = "New Delhi"
```

```
In [54]: employeeDB
```

```
Out[54]: {'EmpID': [101, 102, 103, 104, 105, 999],
          'EmpName': ['A', 'B', 'C', 'D', 'E'],
          'City': ['Houson', 'Tempa', 'New Delhi', 'Chennai', 'Panjim'],
          'Grade': ['A', 'B', 'A', 'B', 'A'],
          'Salary': [100, 200, 350, 230, 150]}
```

```
In [55]: dct
```

```
Out[55]: {'A': (10, 20, 30, 40),
          'D': ['Abhishek', 'Ajeet'],
          'X': {'EmpID': [101, 102, 103, 104, 105, 999],
           'EmpName': ['A', 'B', 'C', 'D', 'E'],
           'City': ['Houson', 'Tempa', 'New Delhi', 'Chennai', 'Panjim'],
           'Grade': ['A', 'B', 'A', 'B', 'A'],
           'Salary': [100, 200, 350, 230, 150]},
          'Z': {'Name': 'Dharmendra', 'son': ['Sunny', 'Bobby'], 'Pet': ['Cat', 'Dog']}}
```

```
In [57]: dct["Z"]['Pet']
```

```
Out[57]: ['Cat', 'Dog']
```

```
In [60]: dct["X"]["Salary"] = dct["X"]["Salary"][::-1]
```

```
In [61]: dct
```

```
Out[61]: {'A': (10, 20, 30, 40),
          'D': ['Abhishek', 'Ajeet'],
          'X': {'EmpID': [101, 102, 103, 104, 105, 999],
           'EmpName': ['A', 'B', 'C', 'D', 'E'],
           'City': ['Houson', 'Tempa', 'New Delhi', 'Chennai', 'Panjim'],
           'Grade': ['A', 'B', 'A', 'B', 'A'],
           'Salary': [150, 230, 350, 200, 100]},
          'Z': {'Name': 'Dharmendra', 'son': ['Sunny', 'Bobby'], 'Pet': ['Cat', 'Dog']}}
```

How to add a key and its values

```
In [62]: dct["Address"] = ["Sec-11","Sec-12","Sec-13"]
```

In [63]:
```
dct
```
Out[63]:
```
{'A': (10, 20, 30, 40),
 'D': ['Abhishek', 'Ajeet'],
 'X': {'EmpID': [101, 102, 103, 104, 105, 999],
  'EmpName': ['A', 'B', 'C', 'D', 'E'],
  'City': ['Houson', 'Tempa', 'New Delhi', 'Chennai', 'Panjim'],
  'Grade': ['A', 'B', 'A', 'B', 'A'],
  'Salary': [150, 230, 350, 200, 100]},
 'Z': {'Name': 'Dharmendra', 'son': ['Sunny', 'Bobby'], 'Pet': ['Cat', 'Dog']},
 'Address': ['Sec-11', 'Sec-12', 'Sec-13']}
```

In [64]:
```
dct["D"] = ["Abhinav","Aniket","Ankit"]
```

In [65]:
```
dct
```
Out[65]:
```
{'A': (10, 20, 30, 40),
 'D': ['Abhinav', 'Aniket', 'Ankit'],
 'X': {'EmpID': [101, 102, 103, 104, 105, 999],
  'EmpName': ['A', 'B', 'C', 'D', 'E'],
  'City': ['Houson', 'Tempa', 'New Delhi', 'Chennai', 'Panjim'],
  'Grade': ['A', 'B', 'A', 'B', 'A'],
  'Salary': [150, 230, 350, 200, 100]},
 'Z': {'Name': 'Dharmendra', 'son': ['Sunny', 'Bobby'], 'Pet': ['Cat', 'Dog']},
 'Address': ['Sec-11', 'Sec-12', 'Sec-13']}
```

In [71]:
```
d = {1:12,2:22,3:30,4:40,5:"A",2:200} # latest key value will be consider
```

In [72]:
```
m = {1:100,2:20,"A":[1,2],"C":20}
```

In [73]:
```
d
```
Out[73]:
```
{1: 12, 2: 200, 3: 30, 4: 40, 5: 'A'}
```

In [74]:
```
m
```
Out[74]:
```
{1: 100, 2: 20, 'A': [1, 2], 'C': 20}
```

Update Function

In [75]:
```
d
```
Out[75]:
```
{1: 12, 2: 200, 3: 30, 4: 40, 5: 'A'}
```

In [76]:
```
d.update({1:100,4:4000})
```

```
In [77]: d
```

```
Out[77]: {1: 100, 2: 200, 3: 30, 4: 4000, 5: 'A'}
```

```
In [79]: d.update({100:1000})
```

```
In [80]: d
```

```
Out[80]: {1: 100, 2: 200, 3: 30, 4: 4000, 5: 'A', 100: 1000}
```

```
In [82]: d.update({1:200})
```

```
In [83]: d
```

```
Out[83]: {1: 200, 2: 200, 3: 30, 4: 4000, 5: 'A', 100: 1000}
```

```
In [84]: employeeDB
```

```
Out[84]: {'EmpID': [101, 102, 103, 104, 105, 999],
          'EmpName': ['A', 'B', 'C', 'D', 'E'],
          'City': ['Houson', 'Tempa', 'New Delhi', 'Chennai', 'Panjim'],
          'Grade': ['A', 'B', 'A', 'B', 'A'],
          'Salary': [150, 230, 350, 200, 100]}
```

- Number : Immutable
- Strings : Immutable
- Tuple : Immutable

```
In [86]: dct = {"a" : 100, [10,20] : 10}
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[86], line 1
----> 1 dct = {"a" : 100, [10,20] : 10}

TypeError: unhashable type: 'list'
```

```
In [87]: dct = {"a" : 100, (10,20) : 10}
```

```
In [93]: dct[(10,20)]
```

```
Out[93]: 10
```

fromkeys

Using fromkey we can create a dictionary very quickly.

In [92]:
```
d
```

Out[92]:
```
{1: 200, 2: 200, 3: 30, 4: 4000, 5: 'A', 100: 1000}
```

In [94]:
```
lst = ["Abhishek","Eric","Corey","Lucas","Modi"]
```

In [95]:
```
dict.fromkeys(lst)
```

Out[95]:
```
{'Abhishek': None, 'Eric': None, 'Corey': None, 'Lucas': None, 'Modi': None}
```

In [96]:
```
dict.fromkeys(lst,10)
```

Out[96]:
```
{'Abhishek': 10, 'Eric': 10, 'Corey': 10, 'Lucas': 10, 'Modi': 10}
```

SetDefault

If you want to create a key but you are not sure about the its values then you can use setdefault.

In [97]:
```
d
```

Out[97]:
```
{1: 200, 2: 200, 3: 30, 4: 4000, 5: 'A', 100: 1000}
```

In [98]:
```
d.setdefault('PAN')
```

In [99]:
```
d
```

Out[99]:
```
{1: 200, 2: 200, 3: 30, 4: 4000, 5: 'A', 100: 1000, 'PAN': None}
```

In [100]:
```
d.setdefault("EmpID","AAA-DDD")
```

Out[100]:
```
'AAA-DDD'
```

In [101]:
```
d
```

Out[101]:
```
{1: 200,
 2: 200,
 3: 30,
 4: 4000,
 5: 'A',
 100: 1000,
 'PAN': None,
 'EmpID': 'AAA-DDD'}
```

In [102]:
```python
d.setdefault('Mobile',"+91 xxxx-xxxx-xx")
```

Out[102]:
```
'+91 xxxx-xxxx-xx'
```

In [103]:
```python
print(d)
```

```
{1: 200, 2: 200, 3: 30, 4: 4000, 5: 'A', 100: 1000, 'PAN': None, 'EmpID': 'AAA-DD
D', 'Mobile': '+91 xxxx-xxxx-xx'}
```

In [104]:
```python
employeeDB
```

Out[104]:
```
{'EmpID': [101, 102, 103, 104, 105, 999],
 'EmpName': ['A', 'B', 'C', 'D', 'E'],
 'City': ['Houson', 'Tempa', 'New Delhi', 'Chennai', 'Panjim'],
 'Grade': ['A', 'B', 'A', 'B', 'A'],
 'Salary': [150, 230, 350, 200, 100]}
```

Dictionary Pop

using pop method we can remove a key and its value, if key is not the part of your dictionary then raise an error.

In [105]:
```python
employeeDB.pop?
```

In [106]:
```python
employeeDB.pop("City")
```

Out[106]:
```
['Houson', 'Tempa', 'New Delhi', 'Chennai', 'Panjim']
```

In [107]:
```python
employeeDB
```

Out[107]:
```
{'EmpID': [101, 102, 103, 104, 105, 999],
 'EmpName': ['A', 'B', 'C', 'D', 'E'],
 'Grade': ['A', 'B', 'A', 'B', 'A'],
 'Salary': [150, 230, 350, 200, 100]}
```

In [108]:
```python
employeeDB.pop("College")
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[108], line 1
----> 1 employeeDB.pop("College")

KeyError: 'College'
```

In [109]:
```python
dct
```

Out[109]:
```
{'a': 100, (10, 20): 10}
```

```
In [110]: d
```

```
Out[110]: {1: 200,
           2: 200,
           3: 30,
           4: 4000,
           5: 'A',
           100: 1000,
           'PAN': None,
           'EmpID': 'AAA-DDD',
           'Mobile': '+91 xxxx-xxxx-xx'}
```

Dictonary popitem : LIFO (Last In First Out)

This function works based on LIFO and return an output in the form tuple, first element will be your key and 2nd element will be your key's values.

```
In [111]: d.popitem()
```

```
Out[111]: ('Mobile', '+91 xxxx-xxxx-xx')
```

```
In [112]: d
```

```
Out[112]: {1: 200,
           2: 200,
           3: 30,
           4: 4000,
           5: 'A',
           100: 1000,
           'PAN': None,
           'EmpID': 'AAA-DDD'}
```

```
In [113]: d.popitem()
```

```
Out[113]: ('EmpID', 'AAA-DDD')
```

```
In [114]: d
```

```
Out[114]: {1: 200, 2: 200, 3: 30, 4: 4000, 5: 'A', 100: 1000, 'PAN': None}
```

```
In [115]: d.popitem()
```

```
Out[115]: ('PAN', None)
```

```
In [116]: d.popitem()
```

```
Out[116]: (100, 1000)
```

```
In [117]: d
```

```
Out[117]: {1: 200, 2: 200, 3: 30, 4: 4000, 5: 'A'}
```

```
In [119]:
print("All Functions from Dictionary: ",[i for i in dir(dict) if "__" not in i]
print()
print("All Functions from list: ",[i for i in dir(list) if "__" not in i])
print()
print("All Functions from tuple: ",[i for i in dir(tuple) if "__" not in i])
print()
print("All Functions from set: ",[i for i in dir(set) if "__" not in i])
```

```
All Functions from Dictionary:  ['clear', 'copy', 'fromkeys', 'get', 'items', 'key
s', 'pop', 'popitem', 'setdefault', 'update', 'values']

All Functions from list:  ['append', 'clear', 'copy', 'count', 'extend', 'index',
'insert', 'pop', 'remove', 'reverse', 'sort']

All Functions from tuple:  ['count', 'index']

All Functions from set:  ['add', 'clear', 'copy', 'difference', 'difference_updat
e', 'discard', 'intersection', 'intersection_update', 'isdisjoint', 'issubset', 'i
ssuperset', 'pop', 'remove', 'symmetric_difference', 'symmetric_difference_updat
e', 'union', 'update']
```

```
In [ ]:
```