## Dict

Values in the list can be accessed using integers called indices

Dictionary access values by means of integers or other python objects called keys which indicates where in the Dictionary is given value is found.

We can index the dictionary by key, to fetch and change the keys associated values

In python it is illegal to assign a position to a list, but we can assign a position to dictionary

We can define a dictionary explicitly as a series of key/value pairs separated by comma.

```
#Create empty dict and assign to d
d = {}
print(type(d)) # <class 'dict'>
```

```
#Access the list using index
lst = ["User 01", "User 02"]
print(lst[0]) # User 01
print(lst[1]) # User 02
```

```
#Access the dict using key
d = {}
d[0] = "User 01"
d[1] = "User 02"
print(d[0]) #User 01
print(d[1]) #User 02
print(type(d)) #<class 'dict'>
print(d) # {0: 'User 01', 1: 'User 02'}
```

```
# Assigning a position to a list is not possible
l = ["Admin 01", "Admin 02"]
#      0,         1
print(l[0]) # Admin 01
l[2] = ["Admin 03"]
print(l) # IndexError: list assignment index out of range
```

```
# Assigning a position to a dict is possible
d = {1:"Admin 01", 2:"Admin 02"}
print(d[1]) # Admin 01
d[3] = ["Admin 03"]
print(d) # {1: 'Admin 01', 2: 'Admin 02', 3: ['Admin 03']}
```

```
#Defining the Dictionary explicitly with key/value pairs and separated by comma,
#In dict duplicate keys are not allowed
d={1: "User 01", 2 : "User 02", 3 : "User 03"}
print(d)
# {1: 'User01', 2: 'User 02', 3: 'User 03'}
print(type(d)) # <class 'dict'>

#Accessing the list of items using list method
print(d.items())
# dict_items([(1, 'User 01'), (2, 'User 02'), (3, 'User 03')])

#Accessing the list of keys using keys method
print(d.keys())
# dict_keys([1, 2, 3])

#Accessing the value using values method
print(d.values())
# dict_values(['User 01', 'User 02', 'User 03'])
```

```
#Accessing the keys using keys method, to access keys we iterate using for loop
d={1: "User 01", 2 : "User 02", 3 : "User 03"}
k = d.keys()
for i in k:print(i)

#Accessing the values
v = d.values()
for i in v:print(i)

#Accessing the key
print(d[2])
```

```
#Delete the Key
#del can be used to remove an entry (key/value) from a dictionary
dictt={1: "User 01", 2 : "User 02", 3 : "User 03"}

del dictt[2]
print(dictt) # {1: 'User 01', 3: 'User 03'}
```

```
# update the dict with new dict
d1 = {1: "NameOne", 2: "NameTwo", 3: "NameThree"}
d2 = {4:"NameFour", 5:"NameFive", 6:"NameSix"}
d1.update(d2)
print(d1)

{1: 'NameOne', 2: 'NameTwo', 3: 'NameThree', 4: 'NameFour', 5: 'NameFive', 6: 'NameSix'}
```

```
# update the dict value
d1 = {1:"NameOne",2:"NameTwo"}
d2 = {2:"Sai Kiran"}
d1.update(d2)
print(d1)  # {1: 'NameOne', 2: 'Sai Kiran'}
```

```
# update dict using tuple data
d1 = {1: "NameOne", 2: "NameTwo", 3: "NameThree"}
d1.update([(1, "Sai Kiran"), (2, "Sai Kumar")])
print(d1) # {1: 'Sai Kiran', 2: 'Sai Kumar', 3: 'NameThree'}
```

```
# update dict using list data
d2 = {1: "NameOne", 2: "NameTwo", 3: "NameThree"}
d2.update([[1, "Sai Ram"], [2, "Sai Ram"]])
print(d2) # {1: 'Sai Ram', 2: 'Sai Ram', 3: 'NameThree'}
```

```
# update method adds new data at the end of list
# if we know the key in advance, the key value will be changed
d1 = {1:"NameOne", 2:"NameTwo", 3:"NameThree", 5:"NameFour"}
d2 = {4:"SaiKiran"}
d1.update(d2)
print(d1) # {1: 'NameOne', 2: 'NameTwo', 3: 'NameThree', 5: 'NameFour', 4: 'SaiKiran'}
```

```
# using pop we can remove the key and value, here the value will be shown after the key
deletes, if there is no key we will get key error

dictionary = {1: 'NameOne', 2:'NameTwo', 3:'NameThree'}
print(dictionary) # {1: 'NameOne', 2: 'NameTwo', 3: 'NameThree'}
print(dictionary.pop(3)) # NameThree
print(dictionary) # {1: 'NameOne', 2: 'NameTwo' }
```

```
# in case of pop, we can see the value after the key is deleted
# but when we use del keyword, we cannot see any value
dictionary = {1: 'NameOne', 2:'NameTwo', 3:'NameThree'}
del dictionary[3]
print(dictionary) # {1: 'NameOne', 2: 'NameTwo'}
```

```
# update the key in dictionary
# dictionary[new_key] = dictionary.pop(old_key)
# update the key integer to another key integer
dictionary = {1: 'NameOne', 2:'NameTwo', 3:'NameThree'}
dictionary[101] = dictionary.pop(1)
print(dictionary) # {2: 'NameTwo', 3: 'NameThree', 101: 'NameOne'}
```

```
# dictionary[new_key] = dictionary.pop(old_key)
# update the key integer to string
dictionary = {1: 'NameOne', 2:'NameTwo', 3:'NameThree'}
dictionary['SaiKumar'] = dictionary.pop(2)
print(dictionary) # {1: 'NameOne', 3: 'NameThree', 'SaiKumar': 'NameTwo'}
```

```
# dictionary[new_key] = dictionary.pop(old_key)
# updating multiple keys at a time
dictionary = {1: 'NameOne', 2:'NameTwo', 3:'NameThree', 4:'NameFour'}
dictionary[102] = dictionary.pop(2)
dictionary[101] = dictionary.pop(1)
print(dictionary) # {3: 'NameThree', 4: 'NameFour', 102: 'NameTwo', 101: 'NameOne'}
```