Assignment 8 - Python

Dictionary

Dictionary Dictionary is a mutable data type in Python. A python dictionary is a collection of key and value pairs separated by a colon (:) & enclosed in curly braces {}. Keys must be unique in a dictionary, duplicate values are allowed.

```python
In [1]: mydict = dict() # empty dictionary
        mydict
```

```
Out[1]: {}
```

```python
In [2]: mydict = {} # empty dictionary
        mydict
```

```
Out[2]: {}
```

```python
In [3]: mydict = {1:'one' , 2:'two' , 3:'three'} # dictionary with integer keys
        mydict
```

```
Out[3]: {1: 'one', 2: 'two', 3: 'three'}
```

```python
In [4]: mydict = dict({1:'one' , 2:'two' , 3:'three'}) # Create dictionary using dict()
        mydict
```

```
Out[4]: {1: 'one', 2: 'two', 3: 'three'}
```

```python
In [5]: mydict = {'A':'one' , 'B':'two' , 'C':'three'} # dictionary with character keys
        mydict
```

```
Out[5]: {'A': 'one', 'B': 'two', 'C': 'three'}
```

```python
In [6]: mydict = {1:'one' , 'A':'two' , 3:'three'} # dictionary with mixed keys
        mydict
```

```
Out[6]: {1: 'one', 'A': 'two', 3: 'three'}
```

```python
In [7]: mydict.keys() # Return Dictionary Keys using keys() method
```

```
Out[7]: dict_keys([1, 'A', 3])
```

```python
In [8]: mydict.values() # Return Dictionary Values using values() method
```

```
Out[8]: dict_values(['one', 'two', 'three'])
```

```python
In [9]: mydict.items() # Access each key-value pair within a dictionary
```

```
Out[9]: dict_items([(1, 'one'), ('A', 'two'), (3, 'three')])
```

```
In [10]:  mydict = {1:'one' , 2:'two' , 'A':['asif' , 'john' , 'Maria']} # dictionary with
          mydict
```

```
Out[10]:  {1: 'one', 2: 'two', 'A': ['asif', 'john', 'Maria']}
```

```
In [13]:  mydict = {1:'one' , 2:'two' , 'A':['asif' , 'john' , 'Maria'], 'B':('Bat' , 'ca')}
          mydict
```

```
Out[13]:  {1: 'one', 2: 'two', 'A': ['asif', 'john', 'Maria'], 'B': ('Bat', 'ca')}
```

```
In [14]:  mydict = {1:'one' , 2:'two' , 'A':{'Name':'asif' , 'Age' :20}, 'B':('Bat' , 'ca')}
          mydict
```

```
Out[14]:  {1: 'one', 2: 'two', 'A': {'Name': 'asif', 'Age': 20}, 'B': ('Bat', 'ca')}
```

```
In [15]:  keys = {'a' , 'b' , 'c' , 'd'}
          mydict3 = dict.fromkeys(keys) # Create a dictionary from a sequence of keys
          mydict3
```

```
Out[15]:  {'a': None, 'b': None, 'c': None, 'd': None}
```

```
In [17]:  keys = {'a' , 'b' , 'c' , 'd'}
          value = 10
          mydict3 = dict.fromkeys(keys , value) # Create a dictionary from a sequence of
          mydict3
```

```
Out[17]:  {'a': 10, 'b': 10, 'c': 10, 'd': 10}
```

```
In [18]:  keys = {'a' , 'b' , 'c' , 'd'}
          value = [10,20,30]
          mydict3 = dict.fromkeys(keys , value) # Create a dictionary from a sequence of
          mydict3
```

```
Out[18]:  {'a': [10, 20, 30], 'b': [10, 20, 30], 'c': [10, 20, 30], 'd': [10, 20, 30]}
```

```
In [19]:  value.append(40)
          mydict3
```

```
Out[19]:  {'a': [10, 20, 30, 40],
           'b': [10, 20, 30, 40],
           'c': [10, 20, 30, 40],
           'd': [10, 20, 30, 40]}
```

Accessing Items

```
In [20]:  mydict = {1:'one' , 2:'two' , 3:'three' , 4:'four'}
          mydict
```

```
Out[20]:  {1: 'one', 2: 'two', 3: 'three', 4: 'four'}
```

```
In [21]:  mydict[1] # Access item using key
```

```
Out[21]:  'one'
```

```
In [22]:  mydict.get(1) # Access item using get() method
```

```
Out[22]:  'one'
```

```
In [23]:  mydict1 = {'Name':'Asif' , 'ID': 74123 , 'DOB': 1991 , 'job' :'Analyst'}
          mydict1
```

```
Out[23]:  {'Name': 'Asif', 'ID': 74123, 'DOB': 1991, 'job': 'Analyst'}
```

```
In [24]:  mydict1['Name'] # Access item using key
```

```
Out[24]:  'Asif'
```

```
In [25]:  mydict1.get('job') # Access item using get() method
```

```
Out[25]:  'Analyst'
```

Add, Remove & Change Items

```
In [26]:  mydict1 = {'Name':'Asif' , 'ID': 12345 , 'DOB': 1991 , 'Address' : 'Hilsinki'}
          mydict1
```

```
Out[26]:  {'Name': 'Asif', 'ID': 12345, 'DOB': 1991, 'Address': 'Hilsinki'}
```

```
In [27]:  mydict1['DOB'] = 1992 # Changing Dictionary Items
          mydict1['Address'] = 'Delhi'
          mydict1
```

```
Out[27]:  {'Name': 'Asif', 'ID': 12345, 'DOB': 1992, 'Address': 'Delhi'}
```

```
In [28]:  dict1 = {'DOB':1995}
          mydict1.update(dict1)
          mydict1
```

```
Out[28]:  {'Name': 'Asif', 'ID': 12345, 'DOB': 1995, 'Address': 'Delhi'}
```

```
In [29]:  mydict1['Job'] = 'Analyst' # Adding items in the dictionary
          mydict1
```

```
Out[29]:  {'Name': 'Asif',
           'ID': 12345,
           'DOB': 1995,
           'Address': 'Delhi',
           'Job': 'Analyst'}
```

```
In [30]:  mydict1.pop('Job') # Removing items in the dictionary using Pop method
          mydict1
```

```
Out[30]:  {'Name': 'Asif', 'ID': 12345, 'DOB': 1995, 'Address': 'Delhi'}
```

```
In [31]: mydict1.popitem() # A random item is removed
```

```
Out[31]: ('Address', 'Delhi')
```

```
In [32]: mydict1
```

```
Out[32]: {'Name': 'Asif', 'ID': 12345, 'DOB': 1995}
```

```
In [33]: del[mydict1['ID']] # Removing item using del method
         mydict1
```

```
Out[33]: {'Name': 'Asif', 'DOB': 1995}
```

```
In [34]: mydict1.clear() # Delete all items of the dictionary using clear method
         mydict1
```

```
Out[34]: {}
```

```
In [35]: del mydict1 # Delete the dictionary object
         mydict1
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[35], line 2
      1 del mydict1 # Delete the dictionary object
----> 2 mydict1

NameError: name 'mydict1' is not defined
```

Copy Dictionary

```
In [36]: mydict = {'Name':'Asif' , 'ID': 12345 , 'DOB': 1991 , 'Address' : 'Hilsinki'}
         mydict
```

```
Out[36]: {'Name': 'Asif', 'ID': 12345, 'DOB': 1991, 'Address': 'Hilsinki'}
```

```
In [37]: mydict1 = mydict # Create a new reference "mydict1"
```

```
In [38]: id(mydict) , id(mydict1) # The address of both mydict & mydict1 will be the same
```

```
Out[38]: (2196057901888, 2196057901888)
```

```
In [39]: mydict2 = mydict.copy() # Create a copy of the dictionary
```

```
In [40]: id(mydict2) # The address of mydict2 will be different from mydict because mydic
```

```
Out[40]: 2196057946752
```

```
In [41]: mydict['Address'] = 'Mumbai'
```

In [42]: `mydict`

Out[42]: `{'Name': 'Asif', 'ID': 12345, 'DOB': 1991, 'Address': 'Mumbai'}`

In [43]: `mydict1 # mydict1 will be also impacted as it is pointing to the same dictionary`

Out[43]: `{'Name': 'Asif', 'ID': 12345, 'DOB': 1991, 'Address': 'Mumbai'}`

In [44]: `mydict2 # Copy of list won't be impacted due to the changes made in the original`

Out[44]: `{'Name': 'Asif', 'ID': 12345, 'DOB': 1991, 'Address': 'Hilsinki'}`

Loop through a Dictionary

In [45]:
```python
mydict1 = {'Name':'Asif' , 'ID': 12345 , 'DOB': 1991 , 'Address' : 'Hilsinki'}
mydict1
```

Out[45]: `{'Name': 'Asif', 'ID': 12345, 'DOB': 1991, 'Address': 'Hilsinki'}`

In [46]:
```python
for i in mydict1:
        print(i , ':' , mydict1[i]) # Key & value pair
```

```
Name : Asif
ID : 12345
DOB : 1991
Address : Hilsinki
```

In [47]:
```python
for i in mydict1:
        print(mydict1[i]) # Dictionary items
```

```
Asif
12345
1991
Hilsinki
```

Dictionary Membership

In [48]:
```python
mydict1 = {'Name':'Asif' , 'ID': 12345 , 'DOB': 1991 , 'Job': 'Analyst'}
mydict1
```

Out[48]: `{'Name': 'Asif', 'ID': 12345, 'DOB': 1991, 'Job': 'Analyst'}`

In [49]: `'Name' in mydict1 # Test if a key is in a dictionary or not.`

Out[49]: `True`

In [50]: `'Asif' in mydict1 # Membership test can be only done for keys.`

Out[50]: `False`

In [51]: `'ID' in mydict1`

Out[51]: True

In [52]: `'Address' in mydict1`

Out[52]: False

All / Any

The all() method returns: True - If all all keys of the dictionary are true False - If any key of the dictionary is false The any() function returns True if any key of the dictionary is True. If not, any() returns False.

In [53]:
```python
mydict1 = {'Name':'Asif' , 'ID': 12345 , 'DOB': 1991 , 'Job': 'Analyst'}
mydict1
```

Out[53]: `{'Name': 'Asif', 'ID': 12345, 'DOB': 1991, 'Job': 'Analyst'}`

In [54]: `all(mydict1) # Will Return false as one value is false (Value 0)`

Out[54]: True