Dictionary

- Collections of Items
- Every item is a pair of key(supported only immutable) and value
- don't have default index
- {}
- key and values are seperated with ':'
- items are seperated with ','

In [1]:

```
# syntax of Dictionary
# variable ={key1:value1,key2:value2,...}
```

In [2]:

```
# declare empty Dictionary
a= {}
```

In [3]:

```
print(type(a))
```

```
<class 'dict'>
```

In [4]:

```
b =dict()
```

In [5]:

```
print(type(b))
```

```
<class 'dict'>
```

In [6]:

```
students ={"001":"ramu","002":"ravi","003":"krishna"}
```

In [8]:

```
print(students)
```

```
{'001': 'ramu', '002': 'ravi', '003': 'krishna'}
```

In [9]:

```
temp ={'one':"check me",1:123,[1,2,3,4,5]:'not working'}
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-9-aaa8e10e19cb> in <module>
----> 1 temp ={'one':"check me",1:123,[1,2,3,4,5]:'not working'}

TypeError: unhashable type: 'list'
```

In [10]:

```
temp1 ={1:123,2.2:"hello","one":[123,234,345],(111,22,3):{1,2,3,4}}
```

In [11]:

```
temp1
```

Out[11]:

```
{1: 123, 2.2: 'hello', 'one': [123, 234, 345], (111, 22, 3): {1, 2, 3, 4}}
```

In [12]:

```
# Access dict values
temp1[1]
```

Out[12]:

```
123
```

In [14]:

```
temp1[2.2]
```

Out[14]:

```
'hello'
```

In [16]:

```
temp1["one"]
```

Out[16]:

```
[123, 234, 345]
```

In [17]:

```
temp1[(111,22,3)]
```

Out[17]:

```
{1, 2, 3, 4}
```

In [19]:

```
# Update values in dict
temp1[1]= "hello"
```

In [21]:

```
temp1[1]
```

Out[21]:

```
'hello'
```

In [22]:

```
temp1[2] = "python programming"
```

In [23]:

```
temp1
```

Out[23]:

```
{1: 'hello',
 2.2: 'hello',
 'one': [123, 234, 345],
 (111, 22, 3): {1, 2, 3, 4},
 2: 'python programming'}
```

In [24]:

```
print(temp1)
```

```
{1: 'hello', 2.2: 'hello', 'one': [123, 234, 345], (111, 22, 3): {1, 2, 3,
4}, 2: 'python programming'}
```

In [25]:

```
# Delete
del temp1[2]
```

In [26]:

```
print(temp1)
```

```
{1: 'hello', 2.2: 'hello', 'one': [123, 234, 345], (111, 22, 3): {1, 2, 3,
4}}
```

In [27]:

```
del temp1[1],temp1[2.2]
```

In [28]:

```
print(temp1)
```

```
{'one': [123, 234, 345], (111, 22, 3): {1, 2, 3, 4}}
```

In [ ]:

```
# Dictionary Methods
# dir(dictionary function)
# dir(dict variable)
# dir(dict value)
```

In [29]:

```
print(dir(dict))
```

```
['__class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__do
c__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__',
'__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__',
'__len__', '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__
repr__', '__reversed__', '__setattr__', '__setitem__', '__sizeof__', '__str_
_', '__subclasshook__', 'clear', 'copy', 'fromkeys', 'get', 'items', 'keys',
'pop', 'popitem', 'setdefault', 'update', 'values']
```

In [30]:

```
print(dir(temp1))
```

```
['__class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__do
c__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__',
'__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__',
'__len__', '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__
repr__', '__reversed__', '__setattr__', '__setitem__', '__sizeof__', '__str_
_', '__subclasshook__', 'clear', 'copy', 'fromkeys', 'get', 'items', 'keys',
'pop', 'popitem', 'setdefault', 'update', 'values']
```

In [31]:

```
print(dir({1:1,2:2}))
```

```
['__class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__do
c__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__',
'__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__',
'__len__', '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__
repr__', '__reversed__', '__setattr__', '__setitem__', '__sizeof__', '__str_
_', '__subclasshook__', 'clear', 'copy', 'fromkeys', 'get', 'items', 'keys',
'pop', 'popitem', 'setdefault', 'update', 'values']
```

In [32]:

```
# Method synatax
#  variable.methodname(arg1,arg2,..)
#  value.methodname(arg1,arg2,...)
```

In [33]:

```
#items()
temp1.items()
```

Out[33]:

```
dict_items([('one', [123, 234, 345]), ((111, 22, 3), {1, 2, 3, 4})])
```

In [34]:

```
# keys()
temp1.keys()
```

Out[34]:

```
dict_keys(['one', (111, 22, 3)])
```

In [35]:

```
# values()
temp1.values()
```

Out[35]:

```
dict_values([[123, 234, 345], {1, 2, 3, 4}])
```

In [36]:

```
# fromkeys()
numbers=[1,2,3,4,5,6]
c={}
c.fromkeys(numbers)
```

Out[36]:

```
{1: None, 2: None, 3: None, 4: None, 5: None, 6: None}
```

In [37]:

```
d={}
d.fromkeys(list(range(5,11)),"python")
```

Out[37]:

```
{5: 'python', 6: 'python', 7: 'python', 8: 'python', 9: 'python', 10: 'pytho
n'}
```

In [38]:

```
# setdefault()
print(temp1)
```

```
{'one': [123, 234, 345], (111, 22, 3): {1, 2, 3, 4}}
```

In [39]:

```
temp1.setdefault('one',"hello check me")
```

Out[39]:

```
[123, 234, 345]
```

In [40]:

```
print(temp1)
```

```
{'one': [123, 234, 345], (111, 22, 3): {1, 2, 3, 4}}
```

In [41]:

```
temp1.setdefault('two',"hello check me once")
```

Out[41]:

```
'hello check me once'
```

In [42]:

```
print(temp1)
```

```
{'one': [123, 234, 345], (111, 22, 3): {1, 2, 3, 4}, 'two': 'hello check me
once'}
```

In [43]:

```
# get()
temp1.get('one')
```

Out[43]:

```
[123, 234, 345]
```

In [44]:

```
temp1.get('three',"I am new one")
```

Out[44]:

```
'I am new one'
```

In [45]:

```
temp1.get('zz',"not found")
```

Out[45]:

```
'not found'
```

In [46]:

```
print(temp1)
```

```
{'one': [123, 234, 345], (111, 22, 3): {1, 2, 3, 4}, 'two': 'hello check me
once'}
```

In [48]:

```
# update()
temp1.update({'name':"narayana"})
```

In [49]:

```
print(temp1)
```

```
{'one': [123, 234, 345], (111, 22, 3): {1, 2, 3, 4}, 'two': 'hello check me
once', 'name': 'narayana'}
```

In [50]:

```
temp1.update({"two":"hello guys"})
```

In [51]:

```
print(temp1)
```

```
{'one': [123, 234, 345], (111, 22, 3): {1, 2, 3, 4}, 'two': 'hello guys', 'n
ame': 'narayana'}
```

In [53]:

```
# copy()
temp2 = temp1  # it will store only address of temp1
temp3={}
temp3=temp1.copy()
```

In [54]:

```
print(temp3)
```

```
{'one': [123, 234, 345], (111, 22, 3): {1, 2, 3, 4}, 'two': 'hello guys', 'n
ame': 'narayana'}
```

In [55]:

```
new=temp1
```

In [56]:

```
print(id(new))
```

```
2635946666752
```

In [57]:

```
print(id(temp1))
```

```
2635946666752
```

In [59]:

```
# pop()
temp1.pop("one")
```

Out[59]:

```
[123, 234, 345]
```

In [60]:

```
print(temp1)
```

```
{(111, 22, 3): {1, 2, 3, 4}, 'two': 'hello guys', 'name': 'narayana'}
```

In [61]:

```
# popitem()
temp1.popitem()
```

Out[61]:

```
('name', 'narayana')
```

In [62]:

```
print(temp1)
```

```
{(111, 22, 3): {1, 2, 3, 4}, 'two': 'hello guys'}
```

In [63]:

```
# clear()
temp1.clear()
```

In [64]:

```
print(temp1)
```

```
{}
```

In [65]:

```
# find length of dict
len(temp1)
```

Out[65]:

```
0
```

In [66]:

```
print(len(students))
```

```
3
```

In [67]:

```
# How to print values without using values() methods
print(students)
```

```
{'001': 'ramu', '002': 'ravi', '003': 'krishna'}
```

In [70]:

```
for value in students:
    print(students[value],end='\t')
```

```
ramu    ravi    krishna
```

In [73]:

```
# sort keys ascending
data={2:'56',1:'2',5:'58',3:'96',4:'32'}
```

In [74]:

```
sorted(data.keys())
```

Out[74]:

```
[1, 2, 3, 4, 5]
```

TASKS

- words frequence
- character frequence

In [ ]:

```
'''# word freq
1) dynamic string
2) create empty  dict
3) ex:  hello hi i am hello ravi am why
    {"hello":2,"hi":1,"i":1,"am":2,"ravi":1,"why":1}
'''
```

File Handling

- Create File
- Write File
- Read File

In [76]:

```
# steps to perform operation on files
'''
step-1)open file   #create connection b/w file & python
step-2)perform operation on file
step-3)close file  #disconnect the connection b/w file & python
'''
```

```
...
```

In [ ]:

```
'''
X mode Rules:-
----------------
-> if file is already existed then it will returns ERROR
-> read() not working
-> write() is working
'''
```

In [78]:

```
# open("filename.extention","mode")
# Craete New file
file  = open("apssdc.txt","x") # x ---> create mode
print(file)
# no operation
file.close()
```

...

In [85]:

```
file1=open('movie.c','x')
print(file1)
file1.close()
```

```
<_io.TextIOWrapper name='movie.c' mode='x' encoding='cp1252'>
```

In [80]:

```
# write File mode(w)
'''
w mode Rules:-
----------------
-> if file is already existed then file is overwrited(deleted after created new file)
-> if file not existed then create new file
-> write() is working
-> read() not working
'''
```

...

In [88]:

```
file_temp = open("college.txt","w")
print(file_temp)
data ="123456"
file_temp.write(str(data))  # write() accepted only string datatype
file_temp.close()
print("Successfully completed")
```

...