Day 10 Dictionary and Sets in Python

Yesterday Recap

- List
  - Accessing
  - Methods
- Tuple
  - Accessing
  - Methods

Todays Objectives

- Dictionary
  - Accessing the elements
  - Dictionary Methods
- Sets
  - Set Methods

Dictionary

It is used to store non-homogenous group of data in the form of Key:value

Properties

- It is used to store data as K:V in `{}`
- It is iterable
- Ordered
  - Python 3.6 is unOrdered
  - Python 3.6+ in ordered
- Key should be unique and it is immutable

In [1]:
```
d1 = {}
d2 = dict()


print(type(d1), type(d2))
```

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

In [2]:
```
d1 = {'RollNo': [1,2,3,4,5,6]}


print(d1)
```

```
{'RollNo': [1, 2, 3, 4, 5, 6]}
```

In [23]:
```
d1 = {'1234567890': ['APSSDC', 'Vijayawada', '01/01/2000', '9876543210'],
      '9876543210': ['Python', 'Earth', '01/01/1994', '0123456789'],
      '1234567890': ['APSSDC', 'Visakhapatnam', '01/01/2000', '9876543210']

print(d1)
```

```
{'1234567890': ['APSSDC', 'Visakhapatnam', '01/01/2000', '9876543210'], '98
76543210': ['Python', 'Earth', '01/01/1994', '0123456789']}
```

In [5]:
```
d2 = {[1,2,3]: 'Python'}
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-5-1efc00009c8b> in <module>
----> 1 d2 = {[1,2,3]: 'Python'}

TypeError: unhashable type: 'list'
```

In [6]:
```
d2 = {(1,2,3): 'Python'}
```

In [7]:
```
print(d2)
```

```
{(1, 2, 3): 'Python'}
```

Accessing the pairs from the dict

In [8]:
```
print(d1)
```

```
{'1234567890': ['APSSDC', 'Visakhapatnam', '01/01/2000', '9876543210'], '98
76543210': ['Python', 'Earth', '01/01/1994', '0123456789']}
```

In [9]:
```
print(d1['1234567890'])
```

```
['APSSDC', 'Visakhapatnam', '01/01/2000', '9876543210']
```

In [10]:
```
print(d1['123456789'])
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-10-f54622f17d58> in <module>
----> 1 print(d1['123456789'])

KeyError: '123456789'
```

In [24]:
```
d1['1234567890'] = ['Apssdc', 'Tadepalli', '01/01/2014', '1234567890']
d1['123456789'] = ['APSSDC', 'Visakhapatnam', '01/01/2000', '9876543210']


print(d1)
```

```
{'1234567890': ['Apssdc', 'Tadepalli', '01/01/2014', '1234567890'], '987654
3210': ['Python', 'Earth', '01/01/1994', '0123456789'], '123456789': ['APSS
DC', 'Visakhapatnam', '01/01/2000', '9876543210']}
```

In [12]:
```
print(d1['1234567890'])
```

```
['Apssdc', 'Tadepalli', '01/01/2014', '1234567890']
```

In [13]:
```
print(d1['1234567890'][2])
```

```
01/01/2014
```

In [14]:
```
d1['1234567890'][2] = '07/07/2014'

print(d1)
```

```
{'1234567890': ['Apssdc', 'Tadepalli', '07/07/2014', '1234567890'], '987654
3210': ['Python', 'Earth', '01/01/1994', '0123456789'], '123456789': ['APSS
DC', 'Visakhapatnam', '01/01/2000', '9876543210']}
```

In [15]:
```
d1['1234567890'][3] = '95135741236'

print(d1)
```

```
{'1234567890': ['Apssdc', 'Tadepalli', '07/07/2014', '95135741236'], '98765
43210': ['Python', 'Earth', '01/01/1994', '0123456789'], '123456789': ['APS
SDC', 'Visakhapatnam', '01/01/2000', '9876543210']}
```

In [18]:
```
d1['1234567890'] = sorted(d1['1234567890'])

print(d1)
```

```
{'1234567890': ['01/01/2014', '1234567890', 'Apssdc', 'Tadepalli'], '987654
3210': ['Python', 'Earth', '01/01/1994', '0123456789'], '123456789': ['APSS
DC', 'Visakhapatnam', '01/01/2000', '9876543210']}
```

In [19]:
```
1 d1 = {123:456, 132:344, 123:456}
```

In [20]:
```
1 print(d1)
```

```
{123: 456, 132: 344}
```

In [21]:
```
1 print(d1[123])
```

```
456
```

Dictionary Methods

In [25]:
```
1 print(d1)
```

```
{'1234567890': ['Apssdc', 'Tadepalli', '01/01/2014', '1234567890'], '987654
3210': ['Python', 'Earth', '01/01/1994', '0123456789'], '123456789': ['APSS
DC', 'Visakhapatnam', '01/01/2000', '9876543210']}
```

In [26]:
```
1 print(d1.keys())
```

```
dict_keys(['1234567890', '9876543210', '123456789'])
```

In [27]:
```
1 print(d1.values())
```

```
dict_values([['Apssdc', 'Tadepalli', '01/01/2014', '1234567890'], ['Pytho
n', 'Earth', '01/01/1994', '0123456789'], ['APSSDC', 'Visakhapatnam', '01/0
1/2000', '9876543210']])
```

In [28]:
```
1 print(d1.items())
```

```
dict_items([('1234567890', ['Apssdc', 'Tadepalli', '01/01/2014', '123456789
0']), ('9876543210', ['Python', 'Earth', '01/01/1994', '0123456789']), ('12
3456789', ['APSSDC', 'Visakhapatnam', '01/01/2000', '9876543210'])])
```

In [29]:
```
1 d1.get('1234567890')
```

Out[29]:
```
['Apssdc', 'Tadepalli', '01/01/2014', '1234567890']
```

In [30]:
```
1 print(d1.get('123'))
```

```
None
```

In [31]:
```
1 print(d1.get('123', 'Key not Avaliable'))
```

```
Key not Avaliable
```

In [32]:
```
print(d1.get('1234567890', 'Key not Avaliable'))
```

```
['Apssdc', 'Tadepalli', '01/01/2014', '1234567890']
```

In [49]:
```
m21 = {'2_1': [25, 55, 65, 77, 80, 60]}
m31 = {'2_1': [35, 55, 65, 77, 80, 60], '3_1': [1,2,4,5,6]}
m32 = {'2_1': [50, 55, 65, 77, 80, 60], '3_2': [6,5,4,3,2,1]}
```

In [50]:
```
print(m21)
```

```
{'2_1': [25, 55, 65, 77, 80, 60]}
```

In [51]:
```
m21.update(m31)
print(m21)
```

```
{'2_1': [35, 55, 65, 77, 80, 60], '3_1': [1, 2, 4, 5, 6]}
```

In [52]:
```
m21.update(m32)

print(m21)
```

```
{'2_1': [50, 55, 65, 77, 80, 60], '3_1': [1, 2, 4, 5, 6], '3_2': [6, 5, 4,
3, 2, 1]}
```

In [45]:
```
copy = Mark2_1.copy()

print(copy)
```

```
{'RollNO': [50, 55, 65, 77, 80, 60], '3_1': [1, 2, 4, 5, 6], '3_2': [6, 5,
4, 3, 2, 1]}
```

In [53]:
```
m21 = m31.copy()

print(m21)
```

```
{'2_1': [35, 55, 65, 77, 80, 60], '3_1': [1, 2, 4, 5, 6]}
```

In [54]:
```
m21.clear()

print(m21)
```

```
{}
```

In [55]:
```
print(m31.pop('3_1'))

print(m31)
```

```
[1, 2, 4, 5, 6]
{'2_1': [35, 55, 65, 77, 80, 60]}
```

In [56]:
```
print(m31.pop('3_1'))

print(m31)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-56-887351cc3e4f> in <module>
----> 1 print(m31.pop('3_1'))
      2
      3 print(m31)

KeyError: '3_1'
```

In [59]:
```
print(m32)
```

```
{'2_1': [50, 55, 65, 77, 80, 60], '3_2': [6, 5, 4, 3, 2, 1]}
```

In [60]:
```
print(m32.popitem())

print(m32)
```

```
('3_2', [6, 5, 4, 3, 2, 1])
{'2_1': [50, 55, 65, 77, 80, 60]}
```

In [61]:
```
print(m21.popitem())

print(m21)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-61-b14506c61eec> in <module>
----> 1 print(m21.popitem())
      2
      3 print(m21)

KeyError: 'popitem(): dictionary is empty'
```

In [63]:
```
li = [1,2,3]


a, b, c = li

print(a,b,c)
```

```
1 2 3
```

In [64]:
```
1 a,b = [1,2,3]
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-64-e99d840cfee5> in <module>
----> 1 a,b = [1,2,3]

ValueError: too many values to unpack (expected 2)
```

In [65]:
```
1 a,b,c,d = [1,2,3]
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-65-c8f31c2d8086> in <module>
----> 1 a,b,c,d = [1,2,3]

ValueError: not enough values to unpack (expected 4, got 3)
```

In [66]:
```
1 a,b,c,d = (1,1,2,3)
2
3 print(a,b,c,d)
```

```
1 1 2 3
```

In [67]:
```
li2 = [a,b,c,d]

print(li2)
```

```
[1, 1, 2, 3]
```

In [68]:
```
1 print(len(d1))
```

```
3
```

In [69]:
```
1 print(max(d1))
```

```
9876543210
```

In [70]:
```
1 print(min(d1))
```

```
123456789
```

Iterating

In [71]:
```
for pair in d1:
    print(pair)
```

```
1234567890
9876543210
123456789
```

In [73]:
```
for val in d1.values():
    print(val)
```

```
['Apssdc', 'Tadepalli', '01/01/2014', '1234567890']
['Python', 'Earth', '01/01/1994', '0123456789']
['APSSDC', 'Visakhapatnam', '01/01/2000', '9876543210']
```

In [74]:
```
for pair in d1:
    print(d1[pair])
```

```
['Apssdc', 'Tadepalli', '01/01/2014', '1234567890']
['Python', 'Earth', '01/01/1994', '0123456789']
['APSSDC', 'Visakhapatnam', '01/01/2000', '9876543210']
```

In [75]:
```
for pair in d1:
    print(d1.get(pair))
```

```
['Apssdc', 'Tadepalli', '01/01/2014', '1234567890']
['Python', 'Earth', '01/01/1994', '0123456789']
['APSSDC', 'Visakhapatnam', '01/01/2000', '9876543210']
```

In [76]:
```
for pair in d1.items():
    print(pair)
```

```
('1234567890', ['Apssdc', 'Tadepalli', '01/01/2014', '1234567890'])
('9876543210', ['Python', 'Earth', '01/01/1994', '0123456789'])
('123456789', ['APSSDC', 'Visakhapatnam', '01/01/2000', '9876543210'])
```

In [77]:
```
for pair in d1.items():
    print(pair[1])
```

```
['Apssdc', 'Tadepalli', '01/01/2014', '1234567890']
['Python', 'Earth', '01/01/1994', '0123456789']
['APSSDC', 'Visakhapatnam', '01/01/2000', '9876543210']
```

In [78]:

```
for key, value in d1.items():
    print(key)
    print(value)
    print('*' * 10)
```

```
1234567890
['Apssdc', 'Tadepalli', '01/01/2014', '1234567890']
**********
9876543210
['Python', 'Earth', '01/01/1994', '0123456789']
**********
123456789
['APSSDC', 'Visakhapatnam', '01/01/2000', '9876543210']
**********
```

{1:1, 2:4, 3:9, ... 100: 10000}

In [79]:

```
sq = {}


for i in range(1, 101):
    sq[i] = i ** 2


print(sq)
```

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100, 11: 1
21, 12: 144, 13: 169, 14: 196, 15: 225, 16: 256, 17: 289, 18: 324, 19: 361,
20: 400, 21: 441, 22: 484, 23: 529, 24: 576, 25: 625, 26: 676, 27: 729, 28:
784, 29: 841, 30: 900, 31: 961, 32: 1024, 33: 1089, 34: 1156, 35: 1225, 36:
1296, 37: 1369, 38: 1444, 39: 1521, 40: 1600, 41: 1681, 42: 1764, 43: 1849,
44: 1936, 45: 2025, 46: 2116, 47: 2209, 48: 2304, 49: 2401, 50: 2500, 51: 2
601, 52: 2704, 53: 2809, 54: 2916, 55: 3025, 56: 3136, 57: 3249, 58: 3364,
59: 3481, 60: 3600, 61: 3721, 62: 3844, 63: 3969, 64: 4096, 65: 4225, 66: 4
356, 67: 4489, 68: 4624, 69: 4761, 70: 4900, 71: 5041, 72: 5184, 73: 5329,
74: 5476, 75: 5625, 76: 5776, 77: 5929, 78: 6084, 79: 6241, 80: 6400, 81: 6
561, 82: 6724, 83: 6889, 84: 7056, 85: 7225, 86: 7396, 87: 7569, 88: 7744,
89: 7921, 90: 8100, 91: 8281, 92: 8464, 93: 8649, 94: 8836, 95: 9025, 96: 9
216, 97: 9409, 98: 9604, 99: 9801, 100: 10000}
```

In [82]:

```
for key in sq:
    if key % 2 == 0:
        print(sq[key], end = '--')
```

```
4--16--36--64--100--144--196--256--324--400--484--576--676--784--900--1024-
-1156--1296--1444--1600--1764--1936--2116--2304--2500--2704--2916--3136--33
64--3600--3844--4096--4356--4624--4900--5184--5476--5776--6084--6400--6724-
-7056--7396--7744--8100--8464--8836--9216--9604--10000--
```

Task

- Character frequency inside the string
- word frequency inside the string

In [83]:

```
s = """Python is an interpreted high-level general-purpose programming la
Developer: Python Software Foundation
Stable release: 3.9.5 / 3 May 2021; 19 days ago
Preview release: 3.10.0b1 / 3 May 2021; 19 days ago
Typing discipline: Duck, dynamic, strong typing; gradual (since 3.5, but
First appeared: February 1991; 30 years ago
Paradigm: Multi-paradigm: object-oriented, procedural (imperative), funct
```

char, char_count

- [['p', 5], ['a', 10]]
- {'p':5, 'a':10}

Contact Application

{'Name': ['Mobile1','mobile1', 'email', 'DOB', 'website']}

Take the input from the user

- 1. Create a contact -> Key:Value
- 2. add the contact to existing contact -> Key -> Name
- 3. edit the contact -> Key -> value
- 4. delete the contact -> Name
- 5. view the contact -> Name

In [88]:

```
option = input("""Press the options below
1. Create a contact
2. add the contact to existing contact
3. edit the contact
4. delete the contact""")
```

```
Press the options below
1. Create a contact
2. add the contact to existing contact
3. edit the contact
4. delete the contact2
```

In [85]:

```
contact = {}
```

In [86]:

```
if option == '1':
    key = input("Enter Contact Name")
    value = input("Enter detail with space sep").split()
    contact[key] = value
```

```
Enter Contact NamePython
Enter detail with space sep9876543210 python@gmail.com 01-01-1997
```

In [87]:
```
print(contact)
```

```
{'Python': ['9876543210', 'python@gmail.com', '01-01-1997']}
```

In [90]:
```
if option == '2':
    name = input('Enter the name to add the existing')
    if name in contact:
        data = input("Enter the data to update").split()
        contact[name].extend(data)
    else:
        print(name, "contact is not avaliable")
```

```
Enter the name to add the existingPython
Enter the data to update1234567890
```

In [91]:
```
print(contact)
```

```
{'Python': ['9876543210', 'python@gmail.com', '01-01-1997', '1234567890']}
```

In [93]:
```
name = input()

if name in contact.keys():
    print(contact.pop(name))
else:
    print('name is not avaliable')
```

```
Python
```

In [94]:
```
print(contact)
```

```
{}
```

Day10 Outcomes

1. Dictionary
2. Dictoinary Methods