Dictionary in Python

01418112 Fundamental Programming Concepts

Dictionaries and Lists

Collection of objects

Mutable

Dynamic structures:
Grow and shrink as needed.

Nested:
A list can contain another list.
A dictionary can contain another dictionary.
A dictionary can also contain a list, and vice versa.

Dictionaries and Lists

Dictionaries differ from lists primarily in how elements are accessed:

List elements are accessed by their position in the list, via indexing.

Dictionary elements are accessed via keys.

ความหมายของ Dictionary

- เป็นโครงสร้างข้อมูลที่มีลักษณะเป็น associative array
- ประกอบขึ้นจากชุดของคู่ลำดับ key กับ value เพื่อเชื่อมโยง (mapping) key เข้าหา value

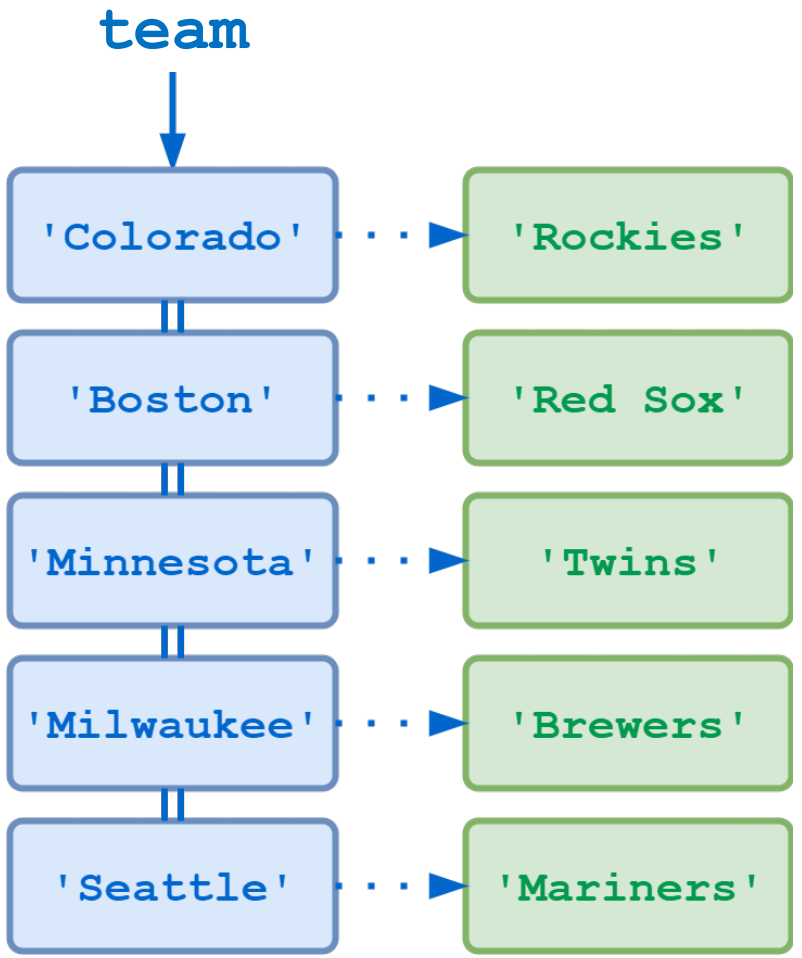
นิยาม Dictionary ในโปรแกรม

- เขียนคู่ลำดับ key กับ value ไว้ภายในสัญลักษณ์ {}
- คั่นระหว่างแต่ละชุดของคู่ลำดับ key กับ value ด้วยสัญลักษณ์ , (comma)
- ในแต่ละคู่ลำดับ คั่นระหว่าง key กับ value ด้วยสัญลักษณ์ : (colon)

d = { <key>: <value>, <key>: <value>, . . . <key>: <value> }

ตัวอย่างการนิยาม Dictionary

```
team = {
    'Colorado' : 'Rockies',
    'Boston'   : 'Red Sox',
    'Minnesota': 'Twins',
    'Milwaukee': 'Brewers',
    'Seattle'  : 'Mariners'
}
```



นิยาม Dictionary ด้วยฟังก์ชัน dict()

อาร์กิวเมนต์ของฟังก์ชัน `dict()` อยู่ในรูปของ sequence ของคู่ลำดับ key กับ value

ในที่นี้อยู่ในรูปลิสต์ของ tuples หรือ ค่าที่เขียนอยู่ระหว่าง () และคั่นระหว่างค่าด้วย ,

d = dict([(<key>, <value>), (<key>, <value), . . . (<key>, <value>)])

```
team = dict([
    ('Colorado', 'Rockies'),
    ('Boston', 'Red Sox'),
    ('Minnesota', 'Twins'),
    ('Milwaukee', 'Brewers'),
    ('Seattle', 'Mariners')
])
```

ตัวแปร Dictionary

```
>>>team
{'Colorado': 'Rockies', 'Boston': 'Red Sox', 'Minnesota': 'Twins',
'Milwaukee': 'Brewers', 'Seattle': 'Mariners’}
>>>print(team)
{'Colorado': 'Rockies', 'Boston': 'Red Sox', 'Minnesota': 'Twins',
'Milwaukee': 'Brewers', 'Seattle': 'Mariners'}
```

ข้อมูลใน dictionary จะถูกแสดงตามลำดับที่นิยาม

การเข้าถึง value ใน Dictionary

- การเข้าถึง value ใน dictionary ไม่มีความสัมพันธ์กับลำดับที่นิยาม
- การเข้าถึง value ใน dictionary ไม่สามารถเข้าถึงโดยตำแหน่งเหมือนลิสต์ได้

```
>>>team[1]
Traceback (most recent call last):
  Python Shell, prompt 5, line 1
builtins.KeyError: 1
```

การเข้าถึง value ใน Dictionary

- การเข้าถึง value ใน dictionary ทำโดยระบุ key ระหว่าง [] ของตัวแปร dictionary

```
>>>team['Boston']
'Red Sox'
>>>team['Seattle']
'Mariners'
>>>team['New York']
Traceback (most recent call last):
  Python Shell, prompt 9, line 1
builtins.KeyError: 'New York'
```

การเพิ่มคู่ลำดับใหม่ใน Dictionary

- การเพิ่มคู่ลำดับใหม่ใน dictionary ได้แก่ การกำหนด key พร้อมกับ value ใหม่
- คู่ลำดับใหม่จะเพิ่มเป็นคู่ลำดับด้านท้ายเสมอ

```
>>>team['Kansas City'] = 'Royals'
>>>team
{'Colorado': 'Rockies', 'Boston': 'Red Sox', 'Minnesota': 'Twins',
'Milwaukee': 'Brewers', 'Seattle': 'Mariners', 'Kansas City': 'Royals'}
```

Key ของ Dictionary

- key ของ dictionary ไม่ใช่ index เพื่อเข้าถึง value

```
>>>d = {0: 'a', 1: 'b', 2: 'c', 3: 'd'}
>>>d
{0: 'a', 1: 'b', 2: 'c', 3: 'd'}
>>>d[0]
'a'
>>>d[3]
'd'
>>>d[4]
Traceback (most recent call last):
  Python Shell, prompt 16, line 1
builtins.KeyError: 4
```

Key ของ Dictionary

- dictionary เป็นการจับคู่ระหว่าง key กับ value แต่ละชุด โดยไม่คำนึงถึงลำดับการนิยาม

```
>>>d = {3: 'd', 2: 'c', 1: 'b', 0: 'a'}
>>>d
{3: 'd', 2: 'c', 1: 'b', 0: 'a'}
>>>d[3]
'd'
>>>d[0]
'a'
```

Key ของ Dictionary

- key อาจอยู่ในคลาส int, float, bool หรืออื่น ๆ ที่มีสมบัติ immutable

```
>>>foo = {42: 'aaa', 2.78: 'bbb', True: 'ccc'}
>>>foo[42]
'aaa'
>>>foo[2.78]
'bbb'
>>>foo[True]
'ccc'
```

สร้าง Dictionary จาก Dictionary ว่าง

```
>>>person = {}
>>>type(person)
<class 'dict'>
>>>bool(person)
False
```

สร้าง Dictionary จาก Dictionary ว่าง

```
>>>person['fname'] = 'Joe'
>>>person['lname'] = 'Fonebone'
>>>person['age'] = 51
>>>person['spouse'] = 'Edna'
>>>person['children'] = ['Ralph', 'Betty', 'Joey']
>>>person['pets'] = {'dog': 'Fido', 'cat': 'Sox'}
>>>print(person)
{'fname': 'Joe', 'lname': 'Fonebone', 'age': 51, 'spouse': 'Edna', 'children':
['Ralph', 'Betty', 'Joey'], 'pets': {'dog': 'Fido', 'cat': 'Sox'}}
```

สร้าง Dictionary จาก Dictionary ว่าง

```
>>>person['lname']
'Fonebone'
>>>person['children']
['Ralph', 'Betty', 'Joey']
>>>person['children'][2]
'Joey'
>>>person['pets']
{'dog': 'Fido', 'cat': 'Sox'}
>>>person['pets']['dog']
'Fido'
```

Value ของ key ใน Dictionary

- key ของคู่ลำดับใน dictionary ต้องไม่ซ้ำกัน (unique) ขณะที่ value อาจซ้ำกันได้

```
>>>d = {0: 'a', 1: 'a', 2: 'a', 3: 'a'}
>>>d
{0: 'a', 1: 'a', 2: 'a', 3: 'a'}
>>>d[0] == d[1] == d[2]
True
>>>d[1] = 'b'
>>>d
{0: 'a', 1: 'b', 2: 'a', 3: 'a'}
```

Value ของ key ใน Dictionary

- ในการนิยามตัวแปร dictionary หากมีคู่ลำดับใน dictionary ที่มี key ซ้ำกัน การนิยามหลังสุดจะแทนที่การนิยามก่อนหน้า

```
>>>d = {0: 'a', 1: 'b', 2: 'c', 1: 'd'}
>>>d
{0: 'a', 1: 'd', 2: 'c'}
>>>d[1]
'd'
```

ตัวดำเนินการ in และ not in

```
>>>team = {'Colorado': 'Rockies', 'Boston': 'Red Sox', 'Minnesota':
'Twins', 'Milwaukee': 'Brewers', 'Seattle'  : 'Mariners'}
>>>'Boston' in team
True
>>>'Los Angeles' not in team
True
>>>if 'Los Angeles' in team and team['Los Angeles']:
        print(team['Los Angeles'])
>>>
```

method d.get(<key>[, <default>])

ค้นหา <key> ใน dictionary d ถ้าพบ <key> คืนค่า value มิฉะนั้นคืนค่า None หรือคืนค่า default ถ้าระบุ

```
>>>m = {1: 'Jan', 2: 'Feb', 3: 'Mar', 4: 'Apr', 5: 'May', 6: 'Jun', 7:
'Jul', 8: 'Aug', 9: 'Sep', 10: 'Oct', 11: 'Nov', 12: 'Dec'}
>>>m.get(5)
'May'
>>>m.get(-5, 'Not found')
'Not found'
```

method d.keys()

คืนค่า key ทั้งหมดใน dictionary d

```
>>>m.keys()
dict_keys([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])
>>>type(m.keys())
<class 'dict_keys'>
>>>ls = list(m.keys())
>>>ls
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
>>>list(m)
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
```

method d.values()

คืนค่า `value` ทั้งหมดใน dictionary `d`

```
>>>m.values()
dict_values(['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep',
'Oct', 'Nov', 'Dec'])
>>>ls = list(m.values())
>>>ls
['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct',
'Nov', 'Dec']
```

method d.items()

คืนค่าลิสต์ของ tuples ที่ประกอบด้วยคู่ลำดับทั้งหมดใน dictionary d

```
>>>m.items()
dict_items([(1, 'Jan'), (2, 'Feb'), (3: 'Mar'), (4, 'Apr'), (5, 'May'), (6, 'Jun'), (7, 'Jul'), (8, 'Aug'), (9, 'Sep'), (10, 'Oct'), (11, 'Nov'), (12, 'Dec')])
>>>ls = list(m.items())
>>>ls
[(1, 'Jan'), (2, 'Feb'), (3: 'Mar'), (4, 'Apr'), (5, 'May'), (6, 'Jun'), (7, 'Jul'), (8, 'Aug'), (9, 'Sep'), (10, 'Oct'), (11, 'Nov'), (12, 'Dec')]
```

method d.pop(<key>[, <default>])

ถ้าพบ <key> จะลบ <key> ออกจาก dictionary d พร้อมคืนค่า value ของ <key> มิฉะนั้นจะเกิดข้อผิดพลาด หรือคืนค่า default ถ้าระบุ

```
>>>m.pop(7)
'Jul'
>>> m.pop(0)
Traceback (most recent call last):
  Python Shell, prompt 39, line 1
builtins.KeyError: 0
>>>m.pop(0, 'Not found')
'Not found'
```

method d.popitem()

ลบ key กับ value ลำดับสุดท้ายจาก dictionary d และคืนค่า tuples ของคู่ลำดับดังกล่าว

```
>>>m.popitem()
(12, 'Dec')
>>>x = m.popitem()
>>>type(x)
<class 'tuple'>
>>>m
{1: 'Jan', 2: 'Feb', 3: 'Mar', 4: 'Apr', 5: 'May', 6: 'Jun', 8: 'Aug', 9:
'Sep', 10: 'Oct'}
>>>len(m)
9
```

method d.clear()

ลบ key กับ value ทั้งหมดจาก dictionary d

```
>>>m.clear()
>>>m
{}
>>>len(m)
0
>>>m.popitem()
Traceback (most recent call last):
  Python Shell, prompt 56, line 1
builtins.KeyError: 'popitem(): dictionary is empty'
```

method d.update(<obj>)

รวม dictionary d กับ dictionary <obj> หรือ iterable <obj> ซึ่งอยู่ในรูปของคู่ลำดับ key กับ value

```
>>>d1 = {'a': 10, 'b': 20, 'c': 30}
>>>d2 = {'a': 10, 'b': 20, 'c': 30}
>>>d1.update(d2)
>>>d1
{'a': 10, 'b': 20, 'c': 30}
>>>d3 = {'A': 100, 'B': 200}
>>>d1.update(d3)
>>>d1
{'a': 10, 'b': 20, 'c': 30, 'A': 100, 'B': 200}
```

method d.update(<obj>)

```
>>>d3
{'A': 100, 'B': 200}
>>>d1.update([('b', 50), ('k', 80)])
>>>d1
{'a': 10, 'b': 50, 'c': 30, 'A': 100, 'B': 200, 'k': 80}
```

ตัวอย่างโปรแกรม dict-1.py

```
d1 = {'a': 10, 'b': 20, 'c': 30}                    #  1
for x in d1:                                        #  2
    print(x)                                        #  3
d2 = {x: x**2 for x in [1, 2, 3, 4]}                #  4
print(d2)                                           #  5
for k, v in d2.items():                             #  6
    print(k, v)                                     #  7
```

ตัวอย่างโปรแกรม dict-1.py

```
d3 = {x: y for x in [1, 2, 3] for y in [3, 1, 4] if x!=y}    #  8
print(d3)                                                    #  9
for x in [1, 2, 3]:                                          # 10
    for y in [3, 1, 4]:                                      # 11
        if x!=y:                                             # 12
            print(x, y)                                      # 13
ls = [x+1 for x in [1, 2, 3, 4]]                             # 14
print(ls)                                                    # 15
```

ตัวอย่างโปรแกรม toStringDate.py

```
def strDate(m, d, y):                                                    #  1
    month = {1: 'Jan', 2: 'Feb', 3: 'Mar', 4: 'Apr',
             5: 'May', 6: 'Jun', 7: 'Jul', 8: 'Aug',
             9: 'Sep', 10: 'Oct', 11: 'Nov', 12: 'Dec'}                  #  2

    print("{:s} {:d}, {:4d}.".format(month[int(m)], int(d), int(y)))     #  3

dateStr = input("Enter a date (mm/dd/yyyy): ")                           #  4
monthStr, dayStr, yearStr = dateStr.split("/")                           #  5
strDate(monthStr, dayStr, yearStr)                                       #  6
```

ตัวอย่างโปรแกรม dict-2.py

```
def inputScore(subj):                                   #  1
    print("input {} scores >>".format(subj))            #  2
    n = int(input('Enter number of students: '))        #  3
    for i in range(n):                                  #  4
        x = int(input())                                #  5
        scores[subj].append(x)                          #  6
scores = {'Math': list(),
          'Science': list(),
          'English': list()}                            #  7
print(scores)                                           #  8
for subj in list(scores):                               #  9
    inputScore(subj)                                    # 10
print(scores)                                           # 11
```

ตัวอย่างโปรแกรม dict-3.py

```
rank = {}                                                          #  1
inputStr = [('Ann', 3), ('Bob', 5), ('John', 1), ('Noon', 6), ('Bow', 8),
            ('Ken', 4), ('Jane', 2), ('Paul', 7), ('Mary', 9)]     #  2
for nm, num in inputStr:                                           #  3
    rank[nm] = int(num)                                            #  4
srank = list(rank)                                                 #  5
srank.sort()                                                       #  6
for x in srank:                                                    #  7
    print('{:4s} {}'.format(x, rank.get(x)))                       #  8
```

References

- https://docs.python.org/3/tutorial/datastructures.html
- https://realpython.com/python-data-types/