# Lesson 7 Python Container Type and Related Operation

## 1. Container Introduction

Object is the abstract of data in Python. All the data in Python program is represented by object or the relationship between objects. A Container is an object that contains the references to other objects, such as Tuple, List, Dictionary, etc. These referenced objects compose the object value of the container.

There are two common types of containers, including sequence (List, Tuple, etc.) and map (Dictionary). In sequence, each element has subscript and they are organized. In map, each element has its name (key), and they are unordered. Apart from sequence and map, “**Set**” is also a container.

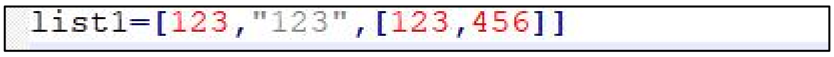
## 2. Container Introduction

### 2.1 List

1) Syntax: [ 123 , "456" , classObjectA , [123,"456"] ]

2) Format: list is surrounded by square bracket and its elements are separated by comma. The type of the elements is diverse, and they can be nested, and read and written by subscript.

3) Feature: There is no unified type of the elements in List, and they can be added, deleted, modified and searched. We can delete (pop) the elements based on subscript, remove elements based on value, insert elements into some position and append the elements to the end of List.



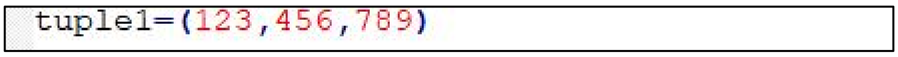
```
list1=[123,"123",[123,456]]
```

## 2.2 Tuple

1）Syntax：（123，456，789，100）。Syntax: (123，456，789，100)

2）Format: surrounded by bracket and the elements are separated by comma. The elements can be nested, but the type of elements must be unified. They can be read by subscript, but cannot be written, added or deleted.

3）Feature: it is considered as immutable list and all the operations on it are similar to List. The only difference is that Tuple can only be searched. It cannot be added(append and insert), delete(remove and pop) and modify (assign value to elements). In other words, Tuple is a list whose elements should be unified and immutable, and content can only be read.



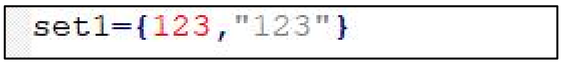
```
tuple1=(123,456,789)
```

## 2.3 Set

1）Syntax: { 1，"23"，classObjectA }

2）Format: it is surrounded by curly bracket. Its elements are separated by comma. The type of the elements is diverse, and they can be nested, and read and written by subscript

3）Feature: apart from automatic deduplication and not supporting sorting, others are the same as list.



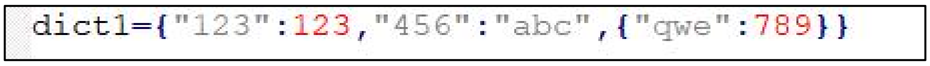
```
set1={123,"123"}
```

## 2.4 Dictionary

1）Syntax: {"name" ："zhangsan"，  "age"：50}

2) Format: it is surrounded by curly bracket and the elements are separated by comma. The elements should be in unified format (When the elements are separated by semicolon, the front part should be string and the latter part can be any types.) and Key must be string.  If there is no semicolon within some element, dictionary will be considered as Set.

3) Feature:dictionary is the most flexible container. It doesn't support sorting, but support nesting. Access the elements through [ ] and get and the elements cannot use subscript. Delete the elements by pop key. (without remove)



```
dict1={"123":123,"456":"abc",{"qwe":789}}
```

## 3. Basic Operation of Container

### 3.1 Basic Operation of List

| Type | Keyword/ function/ method | Meaning |
|---|---|---|
| Add | append() | Append elements to the end |
| | insert() | Insert the data into the designated position |
| Delete | clear() | Clear the list |
| | pop() | Pop up the data at the end by default |
| | pop(index) | Pop up the designated index data |

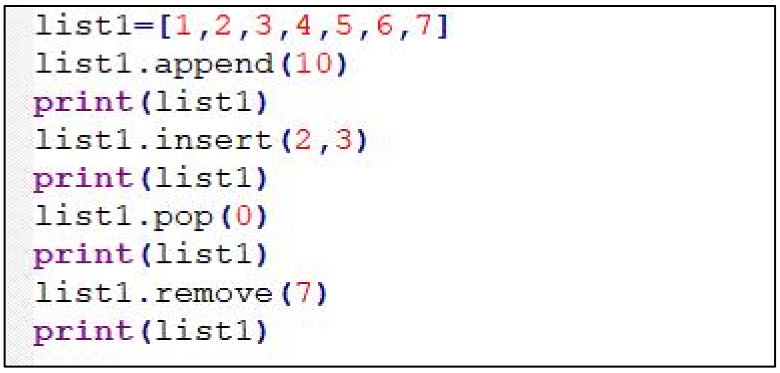
| | | |
|---|---|---|
| | remove (data) | Remove the designated data |
| Modify | extend(list 2) | Add the data of list 2 to the end of the current list |
| Check | count(data) | Count the frequency of occurrence of the data |
| | index(content) | Check the position of the content |
| Other | copy() | Copy a piece of list |
| | sort() | sort |
| | reverse() | Reverse the list |

1）append：Add the elements to the end of the list

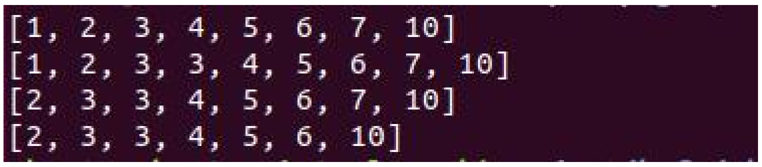
2）insert：Insert the elements into the designated position.

3）pop：Pop up the last element. And you can also specify the position of the pop-up element

4）remove：Delete the designated elements



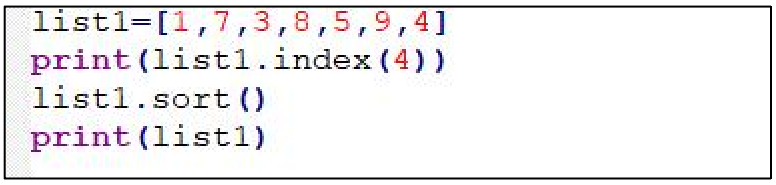
```
list1=[1,2,3,4,5,6,7]
list1.append(10)
print(list1)
list1.insert(2,3)
print(list1)
list1.pop(0)
print(list1)
list1.remove(7)
print(list1)
```



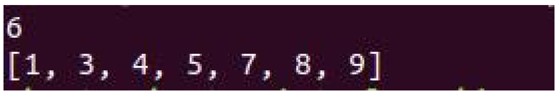
```
[1, 2, 3, 4, 5, 6, 7, 10]
[1, 2, 3, 3, 4, 5, 6, 7, 10]
[2, 3, 3, 4, 5, 6, 7, 10]
[2, 3, 3, 4, 5, 6, 10]
```

5) index：Check the position of the designated elements

6) sort：Sort the content in ascending order, or in descending order.



```
list1=[1,7,3,8,5,9,4]
print(list1.index(4))
list1.sort()
print(list1)
```



```
6
[1, 3, 4, 5, 7, 8, 9]
```

## 3.2 Basic Operation of Tuple

The operations on tuple are similar to those on list, but they are still different.

1) Tuple doesn’t have append(), extend(), insert(), etc., therefore we cannot add element to the tuple.

2) remove() and pop() are also not contained, hence we cannot delete elements from the tuple.

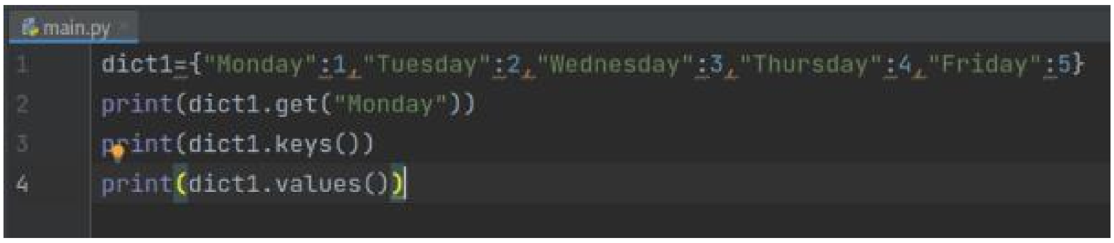
## 3.3 Basic Operation of Dictionary

| Method | Usage |
|---|---|
| get (key, default=None) | Return the value of designated keys. If the value cannot be find in the dictionary, default value will be returned. |
| keys() | Return all the keys of a dictionary as list |
| values() | Return all the values of the dictionary as list |
| update(dict2) | Update the key or value of dictionary dict to dict |
| items() | Return tuple data, keys or value, that can be traversed |

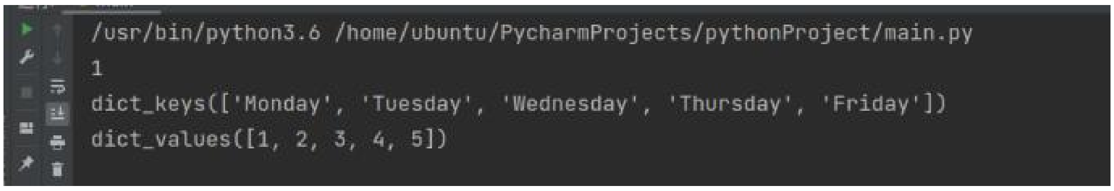
3) get: Return the value of designated key

4) keys: Return all the keys of the dictionary

5) values：Return all the values of the dictionary



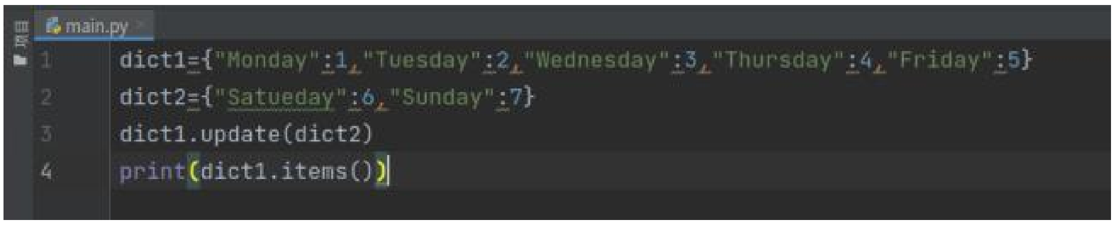
```
dict1={"Monday":1,"Tuesday":2,"Wednesday":3,"Thursday":4,"Friday":5}
print(dict1.get("Monday"))
print(dict1.keys())
print(dict1.values())
```



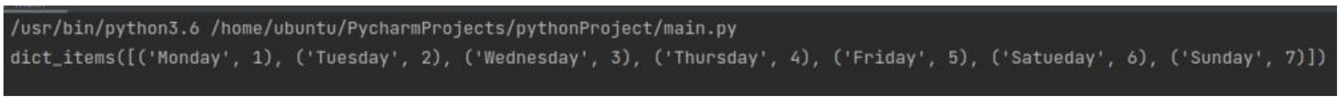
```
/usr/bin/python3.6 /home/ubuntu/PycharmProjects/pythonProject/main.py
1
dict_keys(['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday'])
dict_values([1, 2, 3, 4, 5])
```

6) Update：Update the existing key-value pairs or add new key-value pairs

7) items：Return an array of key-value tuples that can be traversed as a list



```
dict1={"Monday":1,"Tuesday":2,"Wednesday":3,"Thursday":4,"Friday":5}
dict2={"Satueday":6,"Sunday":7}
dict1.update(dict2)
print(dict1.items())
```



```
/usr/bin/python3.6 /home/ubuntu/PycharmProjects/pythonProject/main.py
dict_items([('Monday', 1), ('Tuesday', 2), ('Wednesday', 3), ('Thursday', 4), ('Friday', 5), ('Satueday', 6), ('Sunday', 7)])
```