Lecture#6 Python Programming

Python Dictionary(Dict): Update, Cmp, Len, Sort, Copy, Items, str Example

What is a Dictionary in Python?

A **Dictionary in Python** is the unordered and changeable collection of data values that holds key-value pairs. Each key-value pair in the dictionary maps the key to its associated value making it more optimized. A Dictionary in python is declared by enclosing a comma-separated list of key-value pairs using curly braces({}). Python Dictionary is classified into two elements: Keys and Values.

- Keys will be a single element
- Values can be a list or list within a list, numbers, etc.

In this Python tutorial, you will learn:

- What is a Dictionary in Python?
- Syntax for Python Dictionary:
- Properties of Dictionary Keys
- Python Dictionary Methods
- Updating Dictionary
- Check if a given key already exists in a dictionary
- Python Dictionary in-built Functions
- Variable Types
- Python List cmp() Method
- Dictionary Str(dict)
- Merging Dictionaries
- Merge two dictionaries using update() method
- Merging dictionaries using ** method (From Python 3.5 onwards)
- Dictionary Membership Test

Syntax for Python Dictionary

```
Dict = { ' Tim': 18,  xyz,.. }
```

Dictionary is listed in curly brackets, inside these curly brackets, keys and values are declared. Each key is separated from its value by a colon (:), while commas separate each element.

Properties of Dictionary Keys

There are two important points while using dictionary keys

- More than one entry per key is not allowed (no duplicate key is allowed)
- The values in the dictionary can be of any type, while the keys must be immutable like numbers, tuples, or strings.
- Dictionary keys are case sensitive- Same key name but with the different cases are treated as different keys in Python dictionaries.

Python 2 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print (Dict['Tiffany'])
```

Python 3 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print((Dict['Tiffany']))
```

- In code, we have dictionary name “Dict”
- We declared the name and age of the person in the dictionary, where name is “Keys” and age is the”value”
- Now run the code
- It retrieves the age of tiffany from the dictionary.

Python Dictionary Methods

Copying dictionary

You can also copy the entire dictionary to a new dictionary. For example, here we have copied our original dictionary to the new dictionary name “Boys” and “Girls”.

Python 2 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
Boys = {'Tim': 18,'Charlie':12,'Robert':25}
Girls = {'Tiffany':22}
studentX=Boys.copy()
studentY=Girls.copy()
print studentX
print studentY
```

Python 3 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
Boys = {'Tim': 18,'Charlie':12,'Robert':25}
Girls = {'Tiffany':22}
studentX=Boys.copy()
studentY=Girls.copy()
print(studentX)
print(studentY)
```

- We have the original dictionary (Dict) with the name and age of the boys and girls together
- But we want boys list separate from girls list, so we defined the element of boys and girls in a separate dictionary name “Boys” and “Girls.”
- Now again we have created new dictionary name “student X” and “student Y,” where all the keys and values of boy dictionary are copied into student X, and the girls will be copied in studentY
- So now you don’t have to look into the whole list in the main dictionary(Dict) to check who is a boy and who is girl, you just have to print student X if you want boys list and StudentY if you want girls list
- So, when you run the student X and studentY dictionary, it will give all the elements present in the dictionary of “boys” and “girls” separately

Updating Dictionary

You can also update a dictionary by adding a new entry or a key-value pair to an existing entry or by deleting an existing entry. Here in the example, we will add another name, “Sarah” to our existing dictionary.

Python 2 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
Dict.update({"Sarah":9})
print Dict
```

Python 3 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
Dict.update({"Sarah":9})
print(Dict)
```

- Our existing dictionary “Dict” does not have the name “Sarah.”
- We use the method Dict.update to add Sarah to our existing dictionary
- Now run the code, it adds Sarah to our existing dictionary

Delete Keys from the dictionary

Python dictionary gives you the liberty to delete any element from the dictionary list. Suppose you don’t want the name Charlie in the list, so you can remove the key element by the following code.

Python 2 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
del Dict ['Charlie']
print Dict
```

Python 3 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
del Dict ['Charlie']
print(Dict)
```

When you run this code, it should print the dictionary list without Charlie.

- We used the code del Dict
- When code executed, it has deleted the Charlie from the main dictionary

Dictionary items() Method

The items() method returns a list of tuple pairs (Keys, Value) in the dictionary.

Python 2 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print "Students Name: %s" % Dict.items()
```

Python 3 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print("Students Name: %s" % list(Dict.items()))
```

- We use the code items() method for our Dict.
- When code was executed, it returns a list of items (keys and values) from the dictionary

Check if a given key already exists in a dictionary

For a given list, you can also check whether our child dictionary exists in the main dictionary or not. Here we have two sub-dictionaries “Boys” and “Girls”, now we want to check whether our dictionary Boys exist in our main “Dict” or not. For that, we use the for loop method with else if method.

Python 2 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
Boys = {'Tim': 18,'Charlie':12,'Robert':25}
Girls = {'Tiffany':22}
for key in Boys.keys():
    if key in Dict.keys():
        print True
    else:
        print False
```

Python 3 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
Boys = {'Tim': 18,'Charlie':12,'Robert':25}
Girls = {'Tiffany':22}
for key in list(Boys.keys()):
    if key in list(Dict.keys()):
        print(True)
    else:
        print(False)
```

- The forloop in code checks each key in the main dictionary for Boys keys
- If it exists in the main dictionary, it should print true or else it should print false
- When you execute the code, it will print “True” for three times, as we got three elements in our “Boys” dictionary
- So it indicates that the “Boys” exist in our main dictionary (Dict)

Sorting the Dictionary

In the dictionary, you can also sort the elements. For example, if we want to print the name of the elements of our dictionary alphabetically, we have to use the for a loop. It will sort each element of the dictionary accordingly.

Python 2 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
Boys = {'Tim': 18,'Charlie':12,'Robert':25}
Girls = {'Tiffany':22}
Students = Dict.keys()
Students.sort()
for S in Students:
      print":".join((S,str(Dict[S])))
```

Python 3 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
Boys = {'Tim': 18,'Charlie':12,'Robert':25}
Girls = {'Tiffany':22}
Students = list(Dict.keys())
Students.sort()
for S in Students:
      print(":".join((S,str(Dict[S]))))
```

- We declared the variable students for our dictionary “Dict.”
- Then we use the code Students.sort, which will sort the element inside our dictionary
- But to sort each element in the dictionary, we run the for a loop by declaring variable S
- Now, when we execute the code, the for loop will call each element from the dictionary, and it will print the string and value in an order

Python Dictionary in-built Functions

Dictionary len() Method

The len() function gives the number of pairs in the dictionary.

For example,

Python 2 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print "Length : %d" % len (Dict)
```

Python 3 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print("Length : %d" % len (Dict))
```

When len (Dict) function is executed it gives the output at “4” as there are four elements in our dictionary

Variable Types

Python does not require to explicitly declare the reserve memory space; it happens automatically. The assign values to variable “=” equal sign are used. The code to determine the variable type is ” %type (Dict).”

Python 2 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print "variable Type: %s" %type (Dict)
```

Python 3 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print("variable Type: %s" %type (Dict))
```

- Use the code %type to know the variable type
- When code was executed, it tells a variable type is a dictionary

Python List cmp() Method

The compare method cmp() is used in Python to compare values and keys of two dictionaries. If method returns 0 if both dictionaries are equal, 1 if dic1 > dict2 and -1 if dict1 < dict2.

Python 2 Example

```
Boys = {'Tim': 18,'Charlie':12,'Robert':25}
Girls = {'Tiffany':22}
print cmp(Girls, Boys)
```

Python 3 Example

```
cmp is not supported in Python 3
```

- We have two dictionary name, “Boys” and “Girls.”
- Whichever you declare first in code “cmp(Girls, Boys)” will be considered as dictionary 1. In our case, we declared “Girls” first, so it will be considered as dictionary 1 and boys as dictionary 2
- When code is executed, it prints out -1, It indicates that our dictionary 1 is less than dictionary 2.

Dictionary Str(dict)

With Str() method, you can make a dictionary into a printable string format.

Python 2 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print "printable string:%s" % str (Dict)
```

Python 3 Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print("printable string:%s" % str (Dict))
```

- Use the code % str (Dict)
- It will return the dictionary elements into a printable string format

Here is the list of all Dictionary Methods

Method	Description	Syntax
copy()	Copy the entire dictionary to new dictionary	dict.copy()
update()	Update a dictionary by adding a new entry or a key-value pair to anexisting entry or by deleting an existing entry.	Dict.update([other])
items()	Returns a list of tuple pairs (Keys, Value) in the dictionary.	dictionary.items()
sort()	You can sort the elements	dictionary.sort()
len()	Gives the number of pairs in the dictionary.	len(dict)
cmp()	Compare the values and keys of two dictionaries	cmp(dict1, dict2)
Str()	Make a dictionary into a printable string format	Str(dict)

Merging Dictionaries

Here will understand how to merge two given dictionaries into a single dictionary.

I have two dictionaries as shown below :

Dictionary1 : my_dict1

```
my_dict1 = {"username": "XYZ", "email": "xyz@gmail.com", "location":"Mumbai"}
```

Dictionary 2 : my_dict2

```
my_dict2 = {"firstName" : "Nick", "lastName": "Price"}
```

Let us merge both these dictionaries my_dict1 and my_dict2 and create a single dictionary with namemy_dict.

Merge two dictionaries using update() method

The update() method will help us to merge one dictionary with another. In the example, we will update the my_dict1 with my_dict2. After using update() method the my_dict1 will have the contents of my_dict2 as shown below:

```
my_dict1 = {"username": "XYZ", "email": "xyz@gmail.com", "location":"Mumbai"}

my_dict2 = {"firstName" : "Nick", "lastName": "Price"}

my_dict1.update(my_dict2)

print(my_dict1)
```

Output:

```
{'username': 'XYZ', 'email': 'xyz@gmail.com', 'location': 'Mumbai',
'firstName': 'Nick', 'lastName': 'Price'}
```

Merging dictionaries using ** method (From Python 3.5 onwards)

The ** is called Kwargs in Python, and it will work with Python version 3.5+. Using **, we can merge two dictionaries, and it will return the merged dictionary. Making use of ** in front of the variable will replace the variable with all its content.

Here is a working example of using ** to merge two directories.

```
my_dict1 = {"username": "XYZ", "email": "xyz@gmail.com",
"location":"Mumbai"}

my_dict2 = {"firstName" : "Nick", "lastName": "Price"}

my_dict =  {**my_dict1, **my_dict2}

print(my_dict)
```

Output:

```
{'username': 'XYZ', 'email': 'xyz@gmail.com', 'location': 'Mumbai',
'firstName': 'Nick', 'lastName': 'Price'}
```

Dictionary Membership Test

You can test if a key in the present inside a dictionary or not. This test can be performed only on the key of a dictionary and not the value. The membership test is done using the **in** keyword. When you check the key in the dictionary using the **in** keyword, the expression returns true if the key is present and false if not.

Here is an example that shows member ship test on a dictionary.

```
my_dict = {"username": "XYZ", "email": "xyz@gmail.com", "location":"Mumbai"}
print("email" in my_dict)
print("location" in my_dict)
print("test" in my_dict)
```

Output:

```
True
True
False
```

Summary:

- Dictionaries in a programming language is a type of data-structure used to store information connected in some way.

- Python Dictionary are defined into two elements Keys and Values.
- Dictionaries do not store their information in any particular order, so you may not get your information back in the same order you entered it.
- Keys will be a single element
- Values can be a list or list within a list, numbers, etc.
- More than one entry per key is not allowed (no duplicate key is allowed)
- The values in the dictionary can be of any type, while the keys must be immutable like numbers, tuples, or strings.
- Dictionary keys are case sensitive- Same key name but with the different cases are treated as different keys in Python dictionaries.