Python Programming

RGM College of Engineering & Technology (Autonomous)

Department of Computer Science & Engineering

Academic Year : 2020-2021

DICTIONARY DATA TYPE - 1

UNIT – V:

Dictionaries: Creation of Dictionary objects, Accessing elements of dictionary, Basic operations on Dictionary - Updating the Dictionary, Deleting the elements from Dictionary. Important functions of Dictionary – dict(), len(), clear(), get(), pop(), popitem(), keys(),values(), items(), copy(), setdefault(). Illustrative examples on all the above topics.

Topics Covered:

1. Introduction
2. Creation of Dictionary objects
3. Accessing data from the Dictionary
4. Updating the Dictionary
5. Deleting the elements from Dictionary
6. Important functions of Dictionary

   i) dict() ii) len() iii) clear() iv) get() v) pop() vi) popitem() vii) keys()

   viii) values() ix) items() x) copy() xi) setdefault() xii) update()
7. Dictionary Comprehension
8. Example Programs

Guido Van Rossum

Learning Mantra

If you really strong in the basics, then remaining things will become so easy.

Agenda:

1. Introduction

2. Creation of Dictionary objects

3. Accessing data from the dictionary

1.INTRODUCTION

Need of Dictionary data type:

- We can use List, Tuple and Set to represent a group of individual objects as a single entity.
- If we want to represent a group of objects as key-value pairs then we should go for Dictionary.

Eg:

Roll_no & name

Phone_number & address

Ip_address & domain name

Key features of Dictionary Data type:

1. Duplicate keys are not allowed but values can be duplicated.
2. Heterogeneous objects are allowed for both key and values.
3. insertion order is not preserved.
4. Dictionaries are mutable.
5. Dictionaries are dynamic.
6. indexing and slicing concepts are not applicable.

Note:

- In C++ and Java Dictionaries are known as "Map" where as in Perl and Ruby it is known as "Hash".

2. Creation of Dictionary objects

I. If you want to create an empty dictionary, we use the following approach:

Eg:

```
d = {}
print(type(d))
```

Output:

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

II. We can create an empty dictionary using dict() function also.

Eg:

```
d = dict()

print(type(d))
```

Output:

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

- We can add entries into a dictionary as follows:

d[key] = value

Eg:

```
d[100]="karthi"
d[200]="sahasra"
d[300]="sri"
d['rgm'] = 'Nandyal'
print(d)       #{100: 'karthi', 200: 'sahasra', 300: 'sri', 'rgm' : 'Nandyal'}
```

Output:

```
{100: 'karthi', 200: 'sahasra', 300: 'sri', 'rgm': 'Nandyal'}
```

III. If we know data in advance then we can create dictionary as follows:

Eg:

```
d={100:'karthi' ,200:'sahasra', 300:'sri'}

print(d)
```

Output:

```
{100: 'karthi', 200: 'sahasra', 300: 'sri'}
```

3. Accessing data from the dictionary

- We can access data by using keys.

Eg:

```
d={'a':'apple' ,'b':'banana', 'c':'cat'}

print(d['b'])
```

Output:

```
banana
```

- If the specified key is not available then we will get KeyError.

Eg:

```
d={'a':'apple' ,'b':'banana', 'c':'cat'}

print(d['z'])
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-13-15b56079ad88> in <module>
      1 d={'a':'apple' ,'b':'banana', 'c':'cat'}
----> 2 print(d['z'])

KeyError: 'z'
```

- We can prevent this by checking whether key is already available or not by using **has_key()** function (or) by using **in** operator.
- **d.has_key(400) ➔** returns 1 if key is available otherwise returns 0

Note:

- **has_key()** function is available only in Python 2 but not in Python 3.
- Hence, compulsory we have to use **in** operator.

Eg:

```
d={'a':'apple' ,'b':'banana', 'c':'cat'}
if 'b' in d:
    print(d['b'])
```

Output: banana

Eg:

```
d={'a':'apple' ,'b':'banana', 'c':'cat'}
if 'z' in d:
    print(d['z'])
```

 # If the key is not there in the dictionary, it wont give any KeyError.

Example Program :

Q. Write a Python program to enter name and percentage marks in a dictionary and display information on the screen.

```
rec={}
n=int(input("Enter number of students: "))
i=1
while i <= n:
    name=input("Enter Student Name: ")
    marks=input("Enter % of Marks of Student: ")
    rec[name]=marks
    i=i+1
print("Name of Student","\t","% of Marks")
for x in rec:
    print("\t",x,"\t",rec[x])        # x ➔ key rec[x] ➔ value
```

```
Enter number of students: 3
Enter Student Name: sourav
Enter % of Marks of Student: 89
Enter Student Name: sachin
Enter % of Marks of Student: 77
Enter Student Name: dravid
Enter % of Marks of Student: 77
Name of Student          % of Marks
        sourav          89
        sachin          77
        dravid          77
```

Any question?

If you try to practice programs yourself, then you will learn many things automatically

Spend few minutes and then enjoy the study

Thank You