# Dictionary Data Type

## Date: 07-05-2020 Day 1

**Topics Covered:**

### Introduction

1. **Creation of Dictionary objects**

### Accessing data from the Dictionary

1. **Updating the Dictionary**

### Deleting the elements from Dictionary

1. **Important functions of Dictionary**
   1. dict()
   2. len()
   3. clear()
   4. get()
   5. pop()
   6. popitem()
   7. keys()
   8. values()
   9. items()
   10. copy()
   11. setdefault()
   12. update()

### Dictionary Comprehension Example Programs

1. **Introduction**

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:

rollno name

phone number--address ipaddress domain name

#### Key features of Dictionary Data type :

* 1. Duplicate keys are not allowed but values can be duplicated.
  2. Hetrogeneous 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"**.

### Creation of Dictionary objects

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

In [1]:

d **=** {}

print(type(d))

<class 'dict'>

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

In [2]:

d **=** dict()

print(type(d))

<class 'dict'>

We can add entries into a dictionary as follows:

#### d[key] = value

In [8]:

d[100]**=**"karthi" d[200]**=**"sahasra" d[300]**=**"sri"

d['rgm'] **=** 'Nandyal'

print(d) *#{100: 'karthi', 200: 'sahasra', 300: 'sri', 'rgm' : 'Nandyal'}*

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

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

In [4]:

d**=**{100:'karthi' ,200:'sahasra', 300:'sri'} print(d)

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

### Accessing data from the dictionary

We can access data by using keys.

In [12]:

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

banana

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

In [13]:

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.

In [16]:

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

**if** 'b' **in** d:

print(d['b'])

banana

In [18]:

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 key err*

#### Example Program :

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

In [23]:

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 | |

## Date: 08-05-2020 Day 2

### Updating the Dictionary

#### Syntax: d[key]=value

If the key is not available then a new entry will be added to the dictionary with the specified key-value pair.

If the key is already available then old value will be replaced with new value.

#### Eg :

In [1]:

d**=**{100:"karthi",200:"sahasra",300:"sri"} print(d)

d[400]**=**"sachin" print(d)

d[100]**=**"sourav" print(d)

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

{100: 'karthi', 200: 'sahasra', 300: 'sri', 400: 'sachin'}

{100: 'sourav', 200: 'sahasra', 300: 'sri', 400: 'sachin'}

### Deleting the elements from Dictionary

#### Syntax : del d[key]

It deletes entry associated with the specified key.

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

#### Eg :

In [2]:

d**=**{100:"karthi",200:"sahasra",300:"sri"} print(d)

**del** d[100] print(d) **del** d[400]

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

{200: 'sahasra', 300: 'sri'}

#### ---------------------------------------------------------------------------

**KeyError** Traceback (most recent call last)

**<ipython-input-2-a42fad35d4cc>** in <module> 3 **del** d**[100]**

1. print**(**d**)**

#### ----> 5 del d[400]

**KeyError**: 400

**Note :** Let us discuss about few more functions related to delete the contents of a dictionary.

#### 1. clear():

This function is used to remove all entries from the dictionary.

#### Eg :

In [3]:

d**=**{100:"karthi",200:"sahasra",300:"sri"} print(d)

d.clear() print(d)

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

{}

#### 2.del:

To delete total dictionary, we can use **del** command .Now we cannot access dictionary **d. Eg :**

In [4]:

d**=**{100:"karthi",200:"sahasra",300:"sri"} print(d)

**del** d

print(d) *# d can not access so we will get NameError*

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

#### ---------------------------------------------------------------------------

**NameError** Traceback (most recent call last)

**<ipython-input-4-a93a2726b01d>** in <module>

1. print**(**d**)**

#### del d

**----> 4** print**(**d**)**

**NameError**: name 'd' is not defined

#### Example:

In [7]:

list **=** ['sourav','sachin','rahul']

d**=**{100:list}

print(d)

*# here, value is a list which cinsists of multiple objects which are a*

{100: ['sourav', 'sachin', 'rahul']}

### 6. Important functions of Dictionary

#### dict():

This function is used to create a dictionary.

In [13]:

d**=**dict()

print(d)

d**=**dict({100:"karthi",200:"saha"}) print(d)

d**=**dict([(100,"karthi"),(200,"saha"),(300,"sri")]) print(d)

d**=**dict(((100,"karthi"),(200,"saha"),(300,"sri"))) print(d)

d**=**dict({(100,"karthi"),(200,"saha"),(300,"sri")}) print(d)

d**=**dict({[100,"karthi"],[200,"saha"],[300,"sri"]}) print(d)

*#It creates empty dictionary*

*#It creates dictionary with specifie*

*#It creates dictionary with the given #It creates dictionary with the given #It creates dictionary with the given #It creates dictionary with the given*

{}

{100: 'karthi', 200: 'saha'}

{100: 'karthi', 200: 'saha', 300: 'sri'}

{100: 'karthi', 200: 'saha', 300: 'sri'}

{300: 'sri', 200: 'saha', 100: 'karthi'}

#### ---------------------------------------------------------------------------

**TypeError** Traceback (most recent call last)

**<ipython-input-13-683bd03bcacb>** in <module>

#### 9 d=dict({(100,"karthi"),(200,"saha"),(300,"sri")}) #It creates di ctionary with the given set of tuple elements

10 print**(**d**)**

#### ---> 11 d=dict({[100,"karthi"],[200,"saha"],[300,"sri"]}) #It creates di ctionary with the given set of list elements

12 print**(**d**)**

**TypeError**: unhashable type: 'list'

#### Note :

Compulsory internally we need to take tuple only is acceptable. If you take list it gives the above specified error.

#### If the key values are available in the form of tuple, then all those tuple values can be coverted into dictionary by using 'dict()' function.

1. **len()**

Returns the number of items in the dictionary.

In [17]:

d**=**dict({100:"karthi",200:"saha"})

print(d)

print(len(d))

*#It creates dictionary with specified elements*

{100: 'karthi', 200: 'saha'}

2

#### clear():

To remove all elements from the dictionary.

In [16]:

d**=**dict({100:"karthi",200:"saha"})

print(d) d.clear() print(d)

*#It creates dictionary with specified elements*

{100: 'karthi', 200: 'saha'}

{}

#### get():

To get the value associated with the key.

Two forms of **get()** method is available in Python.

#### d.get(key)

If the key is available then returns the corresponding value otherwise returns None.It wont raise any error.

In [18]:

d**=**dict({100:"karthi",200:"saha"})

print(d.get(100))

*#It creates dictionary with specified elements*

karthi

In [19]:

d**=**dict({100:"karthi",200:"saha"})

print(d.get(500))

*#It creates dictionary with specified elements*

None

#### d.get(key,defaultvalue)

If the key is available then returns the corresponding value otherwise returns default value.

In [21]:

d**=**dict({100:"karthi",200:"saha"})

print(d.get(100,'ravan'))

*#It creates dictionary with specified elements*

karthi

In [14]:

d**=**dict({100:"karthi",200:"saha"})

print(d.get(500,'ravan')) print(d)

*#It creates dictionary with specified elements*

ravan

{100: 'karthi', 200: 'saha'}

#### Another Example :

In [23]:

d**=**{100:"karthi",200:"saha",300:"sri"} print(d[100]) *#karthi*

print(d[400]) *#KeyError:400*

print(d.get(100)) *#karthi*

print(d.get(400)) *#None*

print(d.get(100,"Guest")) *#karthi*

print(d.get(400,"Guest")) *#Guest*

karthi

#### ---------------------------------------------------------------------------

**KeyError** Traceback (most recent call last)

#### <ipython-input-23-2890397dced0> in <module> 1 d={100:"karthi",200:"saha",300:"sri"}

2 print**(**d**[100]) #karthi**

#### ----> 3 print(d[400]) #KeyError:400

1. print**(**d**.**get**(100)) #karthi**
2. print**(**d**.**get**(400)) #None KeyError**: 400

In [24]:

d**=**{100:"karthi",200:"saha",300:"sri"} print(d[100]) *#karthi*

*#print(d[400]) #KeyError:400*

print(d.get(100)) *#karthi*

print(d.get(400)) *#None*

print(d.get(100,"Guest")) *#karthi*

print(d.get(400,"Guest")) *#Guest*

karthi karthi None

karthi Guest

#### pop(): Syntax : d.pop(key)

It removes the entry associated with the specified key and returns the corresponding value.

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

d**=**{100:"karthi",200:"saha",300:"sri"} print(d)

print(d.pop(100)) print(d)

print(d.pop(400))

{100: 'karthi', 200: 'saha', 300: 'sri'} karthi

{200: 'saha', 300: 'sri'}

#### ---------------------------------------------------------------------------

**KeyError** Traceback (most recent call last)

**<ipython-input-26-787766bb18c2>** in <module>

1. print**(**d**.**pop**(100))**
2. print**(**d**)**

**----> 5** print**(**d**.**pop**(400))**

**KeyError**: 400

#### popitem():

It removes an arbitrary item(key-value) from the dictionaty and returns it.

In [29]:

d**=**{100:"karthi",200:"saha",300:"sri"} print(d)

print(d.popitem()) print(d.popitem()) print(d)

print(d.pop(400)) *# KeyError*

{100: 'karthi', 200: 'saha', 300: 'sri'}

(300, 'sri')

(200, 'saha')

{100: 'karthi'}

#### ---------------------------------------------------------------------------

**KeyError** Traceback (most recent call last)

**<ipython-input-29-4185b7c5bad9>** in <module>

1. print**(**d**.**popitem**())**
2. print**(**d**)**

**----> 6** print**(**d**.**pop**(400)) # KeyError KeyError**: 400

If the dictionary is empty then we will get **KeyError.**

In [28]:

d **=**{}

print(d.popitem())

*#KeyError: 'popitem(): dictionary is empty'*

#### ---------------------------------------------------------------------------

**KeyError** Traceback (most recent call last)

**<ipython-input-28-14f741a4e5d5>** in <module> 1 d **={}**

**----> 2** print**(**d**.**popitem**())**

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

In [30]:

d**=**{100:"karthi",200:"saha",300:"sri"} print(d)

print(d.popitem()) print(d.popitem()) print(d.popitem()) print(d.popitem()) print(d)

{100: 'karthi', 200: 'saha', 300: 'sri'}

(300, 'sri')

(200, 'saha')

(100, 'karthi')

#### ---------------------------------------------------------------------------

**KeyError** Traceback (most recent call last)

**<ipython-input-30-17881d89d74e>** in <module>

1. print**(**d**.**popitem**())**
2. print**(**d**.**popitem**())**

**----> 6** print**(**d**.**popitem**())**

7 print**(**d**)**

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

#### keys():

It returns all keys associated with dictionary.

#### Eg :

In [33]:

d**=**{100:"karthi",200:"saha",300:"sri"} print(d.keys())

**for** key **in** d.keys(): print(key)

dict\_keys([100, 200, 300])

100

200

300

#### values():

It returns all values associated with the dictionary.

In [34]:

d**=**{100:"karthi",200:"saha",300:"sri"} print(d.values())

**for** key **in** d.values(): print(key)

dict\_values(['karthi', 'saha', 'sri']) karthi

saha sri

#### items():

It returns list of tuples representing key-value pairs like as shown below.

#### [(k,v),(k,v),(k,v)]

In [38]:

d**=**{100:"karthi",200:"saha",300:"sri"} list **=** d.items()

print(list)

dict\_items([(100, 'karthi'), (200, 'saha'), (300, 'sri')])

In [35]:

d**=**{100:"karthi",200:"saha",300:"sri"}

**for** k,v **in** d.items(): print(k,"--",v)

|  |  |  |
| --- | --- | --- |
| 100 | -- | karthi |
| 200 | -- | saha |
| 300 | -- | sri |

#### copy():

This method is used to create exactly duplicate dictionary(cloned copy).

In [37]:

d**=**{100:"karthi",200:"saha",300:"sri"} d1**=**d.copy()

print(d1) print(d)

{100: 'karthi', 200: 'saha', 300: 'sri'}

{100: 'karthi', 200: 'saha', 300: 'sri'}

#### setdefault():

**Syntax : d.setdefault(k,v)**

If the key is already available then this function returns the corresponding value.

If the key is not available then the specified key-value will be added as new item to the dictionary.

#### Eg :

In [39]:

d**=**{100:"karthi",200:"saha",300:"sri"} print(d.setdefault(400,"sourav"))

print(d)

print(d.setdefault(100,"sachin")) print(d)

sourav

{100: 'karthi', 200: 'saha', 300: 'sri', 400: 'sourav'} karthi

{100: 'karthi', 200: 'saha', 300: 'sri', 400: 'sourav'}

#### update(): Syntax : d.update(x)

All items present in the dictionary x will be added to dictionary **d**.

In [42]:

d**=**{100:"karthi",200:"saha",300:"sri"}

d1 **=**{'a':'apple', 'b':'banana'} d.update(d1)

print(d)

{100: 'karthi', 200: 'saha', 300: 'sri', 'a': 'apple', 'b': 'banana'}

In [44]:

d**=**{100:"karthi",200:"saha",300:"sri"}

d1 **=**{'a':'apple', 'b':'banana'} d2 **=** {777:'A', 888:'B'}

d.update(d1,d2)

print(d)

*# For ipdate method. you need to pass single argument only.*

#### ---------------------------------------------------------------------------

**TypeError** Traceback (most recent call last)

**<ipython-input-44-58a2bfd142f6>** in <module> 2 d1 **={'a':'apple', 'b':'banana'}**

#### 3 d2 = {777:'A', 888:'B'}

**----> 4** d**.**update**(**d1**,**d2**) # For update method. you need to pass si ngle argument only.**

5 print**(**d**)**

**TypeError**: update expected at most 1 arguments, got 2

In [46]:

d**=**{100:"karthi",200:"saha",300:"sri"}

d1 **=**{'a':'apple', 'b':'banana'} d2 **=** {777:'A', 888:'B'}

d.update([(777,'A')]) *# For ipdate method. you can pass list of tuple as an argument. i.e*

print(d)

{100: 'karthi', 200: 'saha', 300: 'sri', 777: 'A'}

In [47]:

d**=**{100:"karthi",200:"saha",300:"sri"}

d1 **=**{'a':'apple', 'b':'banana'} d2 **=** {777:'A', 888:'B'}

d.update([(777,'A'),(888,'B'),(999,'C')]) *# you can add any no.of list of tuple elements.*

print(d)

{100: 'karthi', 200: 'saha', 300: 'sri', 777: 'A', 888: 'B', 999: 'C'}

## Date: 09-05-2020 Day 3

#### Example Programs

**Q 1. Write a program to take dictionary from the keyboard and print the sum of values.**

In [3]:

d**=**eval(input("Enter dictionary:")) s**=**sum(d.values())

print("Sum= ",s)

Enter dictionary:{'A':100,'B':200,'c':300} Sum= 600

d**=**eval(input("Enter dictionary:")) s**=**sum(d.values())

print("Sum= ",s)

Enter dictionary:'A':100,'B':200,'c':300 Traceback **(most recent call last)**:

File "C:\Users\HP\Anaconda3\lib\site-packages\IPython\core\interactiveshel l.py", line 3296, in run\_code

exec(code\_obj, self.user\_global\_ns, self.user\_ns)

#### File "<ipython-input-4-7372dea074de>", line 1, in <module> d=eval(input("Enter dictionary:"))

**File "<string>", line 1**

#### 'A':100,'B':200,'c':300

**^**

**SyntaxError:** invalid syntax

In [5]:

l **=** [10,20,30,40]

s **=** sum(l) *# sum() function works on list also*

print('Sum is : ',s)

Sum is : 100

In [6]:

l **=** (10,20,30,40)

s **=** sum(l) *# sum() function works on tuple also*

print('Sum is : ',s)

Sum is : 100

In [7]:

l **=** {10,20,30,40}

s **=** sum(l) *# sum() function works on set also*

print('Sum is : ',s)

Sum is : 100

**Note :** sum() function can work on any sequence.

#### Q 2. Write a program to find number of occurrences of each letter present in the given string.

word**=**input("Enter any word: ")

d**=**{}

**for** x **in** word: d[x]**=**d.get(x,0)**+**1

**for** k,v **in** d.items():

*# we are creating dictionary with the given word*

*====>*

print(k,"occurred ",v," times")

Enter any word: mississippi

|  |  |  |  |
| --- | --- | --- | --- |
| m | occurred | 1 | times |
| i | occurred | 4 | times |
| s | occurred | 4 | times |
| p | occurred | 2 | times |

In [10]:

word**=**input("Enter any word: ")

d**=**{}

**for** x **in** word: d[x]**=**d.get(x,0)**+**1

**for** k,v **in** sorted(d.items()):

*# we are creating dictionary with the given word*

*====>*

*# To sort all the items of the dictionary in alphabetical o*

print(k,"occurred ",v," times")

Enter any word: mississippi

|  |  |  |  |
| --- | --- | --- | --- |
| i | occurred | 4 | times |
| m | occurred | 1 | times |
| p | occurred | 2 | times |
| s | occurred | 4 | times |

#### Q 3. Write a program to find number of occurrences of each vowel present in the given string.

In [11]:

word**=**input("Enter any word: ") vowels**=**{'a','e','i','o','u'} d**=**{}

**for** x **in** word:

**if** x **in** vowels: d[x]**=**d.get(x,0)**+**1

**for** k,v **in** sorted(d.items()):

print(k,"occurred ",v," times")

Enter any word: doganimaldoganimal

|  |  |  |  |
| --- | --- | --- | --- |
| a | occurred | 4 | times |
| i | occurred | 2 | times |
| o | occurred | 2 | times |

#### Q 4. Write a program to accept student name and marks from the keyboard and creates a dictionary. Also display student marks by taking student name as input.

In [12]:

n**=**int(input("Enter the number of students: ")) d**=**{}

**for** i **in** range(n):

name**=**input("Enter Student Name: ")

marks**=**input("Enter Student Marks: ")

d[name]**=**marks *# assigninng values to the keys of the dictionary 'd'*

**while True**:

name**=**input("Enter Student Name to get Marks: ") marks**=**d.get(name,**-**1)

**if** marks**== -**1:

print("Student Not Found")

**else**:

print("The Marks of",name,"are",marks) *# print('The marks of {} :{}'.format(name,*

option**=**input("Do you want to find another student marks[Yes|No]")

**if** option**==**"No":

**break**

print("Thanks for using our application")

Enter the number of students: 5 Enter Student Name: Sourav

Enter Student Marks: 90

Enter Student Name: Sachin Enter Student Marks: 87

Enter Student Name: Rahul Enter Student Marks: 86

Enter Student Name: Parthiv Enter Student Marks: 56

Enter Student Name: Robin Enter Student Marks: 66

Enter Student Name to get Marks: Sourav The Marks of Sourav are 90

Do you want to find another student marks[Yes|No]Y Enter Student Name to get Marks: Robin

The Marks of Robin are 66

Do you want to find another student marks[Yes|No]y Enter Student Name to get Marks: karthi

Student Not Found

Do you want to find another student marks[Yes|No]No Thanks for using our application

**7. Dictionary Comprehension**

Comprehension concept applicable for dictionaries also.

In [20]:

squares**=**{x:x**\***x **for** x **in** range(1,6)} print(squares)

doubles**=**{x:2**\***x **for** x **in** range(1,6)} print(doubles)

{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}

{1: 2, 2: 4, 3: 6, 4: 8, 5: 10}

In [ ]: