

**Dictionary Data Structure**

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

Duplicate keys are not allowed but values can be duplicated. Hetrogeneous objects are allowed for both key and values. insertion order is not preserved

Dictionaries are mutable Dictionaries are dynamic

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"

### How to create Dictionary?

d={} or d=dict()

we are creating empty dictionary. We can add entries as follows

d[100]="durga" d[200]="ravi" d[300]="shiva"

print(d) #{100: 'durga', 200: 'ravi', 300: 'shiva'}

If we know data in advance then we can create dictionary as follows d={100:'durga' ,200:'ravi', 300:'shiva'}

d={key:value, key:value}



### How to access data from the dictionary?

We can access data by using keys.

d={100:'durga' ,200:'ravi', 300:'shiva'} print(d[100]) #durga

print(d[300]) #shiva

If the specified key is not available then we will get KeyError print(d[400]) # KeyError: 400

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

But has\_key() function is available only in Python 2 but not in Python 3. Hence compulsory we have to use in operator.

if 400 in d:

print(d[400])

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

|  |
| --- |
| **1) rec={}** |
| **2) n=int(input("Enter number of students: "))** |
| **3) i=1** |
| **4) while i <=n:** |
| **5) name=input("Enter Student Name: ")** |
| **6) marks=input("Enter % of Marks of Student: ")** |
| **7) rec[name]=marks** |
| **8) i=i+1** |
| **9) print("Name of Student","\t","% of marks")** |
| **10) for x in rec:** |
| **11) print("\t",x,"\t\t",rec[x])** |
| **12)** |
| **13) Output** |
| **14) D:\Python\_classes>py test.py** |
| **15) Enter number of students: 3** |
| **16) Enter Student Name: durga** |
| **17) Enter % of Marks of Student: 60%** |
| **18) Enter Student Name: ravi** |
| **19) Enter % of Marks of Student: 70%** |
| **20) Enter Student Name: shiva** |



|  |  |  |  |
| --- | --- | --- | --- |
| **21) Enter % of Marks of Student: 80%** | | | |
| **22) Name of Student** | | | **% of marks** |
| **23)** | **durga** | **60%** | |
| **24)** | **ravi** | **70 %** | |
| **25)** | **shiva** | **80%** | |

## How to update dictionaries?

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:

|  |
| --- |
| **1. d={100:"durga",200:"ravi",300:"shiva"}** |
| **2. print(d)** |
| **3. d[400]="pavan"** |
| **4. print(d)** |
| **5. d[100]="sunny"** |
| **6. print(d)** |
| **7.** |
| **8. Output** |
| **9. {100: 'durga', 200: 'ravi', 300: 'shiva'}** |
| **10. {100: 'durga', 200: 'ravi', 300: 'shiva', 400: 'pavan'}** |
| **11. {100: 'sunny', 200: 'ravi', 300: 'shiva', 400: 'pavan'}** |

## How to delete elements from dictionary?

#### del d[key]

It deletes entry associated with the specified key. If the key is not available then we will get KeyError

Eg:

|  |  |
| --- | --- |
| **1.** | **d={100:"durga",200:"ravi",300:"shiva"}** |
| **2.** | **print(d)** |
| **3.** | **del d[100]** |
| **4.** | **print(d)** |
| **5.** | **del d[400]** |
| **6.** | |
| **7.** | **Output** |
| **8.** | **{100: 'durga', 200: 'ravi', 300: 'shiva'}** |



**9. {200: 'ravi', 300: 'shiva'}**

10. **KeyError: 400**

#### d.clear()

To remove all entries from the dictionary Eg:

|  |  |
| --- | --- |
| **1.** | **d={100:"durga",200:"ravi",300:"shiva"}** |
| **2.** | **print(d)** |
| **3.** | **d.clear()** |
| **4.** | **print(d)** |
| **5.** | |
| **6.** | **Output** |
| **7.** | **{100: 'durga', 200: 'ravi', 300: 'shiva'}** |
| 8. | **{}** |

#### del d

To delete total dictionary.Now we cannot access d Eg:

|  |  |
| --- | --- |
| **1.** | **d={100:"durga",200:"ravi",300:"shiva"}** |
| **2.** | **print(d)** |
| **3.** | **del d** |
| **4.** | **print(d)** |
| **5.** | |
| **6.** | **Output** |
| **7.** | **{100: 'durga', 200: 'ravi', 300: 'shiva'}** |
| **8.** | **NameError: name 'd' is not defined** |

# Important functions of dictionary:

#### dict():

To create a dictionary

d=dict() ===>It creates empty dictionary

d=dict({100:"durga",200:"ravi"}) ==>It creates dictionary with specified elements d=dict([(100,"durga"),(200,"shiva"),(300,"ravi")])==>It creates dictionary with the given list of tuple elements



#### len()

Returns the number of items in the dictionary

#### clear():

To remove all elements from the dictionary

#### get():

To get the value associated with the key

d.get(key)

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

d.get(key,defaultvalue)

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

Eg:

d={100:"durga",200:"ravi",300:"shiva"} print(d[100]) ==>durga

print(d[400]) ==>KeyError:400 print(d.get(100)) ==durga print(d.get(400)) ==>None print(d.get(100,"Guest")) ==durga print(d.get(400,"Guest")) ==>Guest

#### pop():

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 Eg:

|  |  |
| --- | --- |
| **1)** | **d={100:"durga",200:"ravi",300:"shiva"}** |
| **2)** | **print(d.pop(100))** |
| **3)** | **print(d)** |
| **4)** | **print(d.pop(400))** |
| **5)** | |
| **6)** | **Output** |



**7) durga**

**8) {200: 'ravi', 300: 'shiva'}**

**9) KeyError: 400**

#### popitem():

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

|  |  |
| --- | --- |
| **1)** | **d={100:"durga",200:"ravi",300:"shiva"}** |
| **2)** | **print(d)** |
| **3)** | **print(d.popitem())** |
| **4)** | **print(d)** |
| **5)** | |
| **6)** | **Output** |
| **7)** | **{100: 'durga', 200: 'ravi', 300: 'shiva'}** |
| **8)** | **(300, 'shiva')** |
| **9)** | **{100: 'durga', 200: 'ravi'}** |

If the dictionary is empty then we will get KeyError d={}

print(d.popitem()) ==>KeyError: 'popitem(): dictionary is empty'

#### keys():

It returns all keys associated eith dictionary Eg:

|  |
| --- |
| **1) d={100:"durga",200:"ravi",300:"shiva"}** |
| **2) print(d.keys())** |
| **3) for k in d.keys():** |
| **4) print(k)** |
| **5)** |
| **6) Output** |
| **7) dict\_keys([100, 200, 300])** |
| **8) 100** |
| **9) 200** |
| **10) 300** |

#### values():

It returns all values associated with the dictionary



Eg:

|  |
| --- |
| **1. d={100:"durga",200:"ravi",300:"shiva"}** |
| **2. print(d.values())** |
| **3. for v in d.values():** |
| **4. print(v)** |
| **5.** |
| **6. Output** |
| **7. dict\_values(['durga', 'ravi', 'shiva'])** |
| **8. durga** |
| **9. ravi** |
| **10. shiva** |

#### items():

It returns list of tuples representing key-value pairs. [(k,v),(k,v),(k,v)]

Eg:

|  |  |
| --- | --- |
| **1.** | **d={100:"durga",200:"ravi",300:"shiva"}** |
| **2.** | **for k,v in d.items():** |
| **3.** | **print(k,"--",v)** |
| **4.** | |
| **5.** | **Output** |
| **6.** | **100 -- durga** |
| **7.** | **200 -- ravi** |
| **8.** | **300 -- shiva** |

#### copy():

To create exactly duplicate dictionary(cloned copy) d1=d.copy();

#### setdefault():

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:

|  |
| --- |
| **1. d={100:"durga",200:"ravi",300:"shiva"}** |
| **2. print(d.setdefault(400,"pavan"))** |
| **3. print(d)** |
| **4. print(d.setdefault(100,"sachin"))** |
| **5. print(d)** |
| **6.** |
| **7. Output** |
| **8. pavan** |
| **9. {100: 'durga', 200: 'ravi', 300: 'shiva', 400: 'pavan'}** |
| **10. durga** |
| **11. {100: 'durga', 200: 'ravi', 300: 'shiva', 400: 'pavan'}** |

#### update():

d.update(x)

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

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

|  |  |
| --- | --- |
| **1.** | **d=eval(input("Enter dictionary:"))** |
| **2.** | **s=sum(d.values())** |
| **3.** | **print("Sum= ",s)** |
| **4.** | |
| **5.** | **Output** |
| **6.** | **D:\Python\_classes>py test.py** |
| **7.** | **Enter dictionary:{'A':100,'B':200,'C':300}** |
| **8.** | **Sum= 600** |

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

|  |
| --- |
| **1. word=input("Enter any word: ")** |
| **2. d={}** |
| **3. for x in word:** |
| **4. d[x]=d.get(x,0)+1** |
| **5. for k,v in d.items():** |
| **6. print(k,"occurred ",v," times")** |
| **7.** |
| **8. Output** |
| **9. D:\Python\_classes>py test.py** |
| **10. Enter any word: mississippi** |
| **11. m occurred 1 times** |
| **12. i occurred 4 times** |
| **13. s occurred 4 times** |



**14. p occurred 2 times**

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

**1. word=input("Enter any word: ") 3. d={}**

**5. if x in vowels:**

**7. for k,v in sorted(d.items()): 9.**

**11. D:\Python\_classes>py test.py**

**13. a occurred 4 times**

**15. o occurred 2 times**

**14. i occurred 2 times**

**12. Enter any word: doganimaldoganimal**

**10. Output**

**8. print(k,"occurred ",v," times")**

**d[x]=d.get(x,0)+1**

**6.**

**4. for x in word:**

**2. vowels={'a','e','i','o','u'}**

**Q. 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?**

**1) n=int(input("Enter the number of students: "))**

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

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

**7) while True:**

**9) marks=d.get(name,-1)**

**11)**

**print("Student Not Found")**

**13)**

**print("The Marks of",name,"are",marks)**

**15) if option=="No":**

**17) print("Thanks for using our application")**

**19) Output**

**21) Enter the number of students: 5**

**23) Enter Student Marks: 90**

**22) Enter Student Name: sunny**

**20) D:\Python\_classes>py test.py**

**18)**

**break**

**16)**

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

**12) else:**

**10) if marks== -1:**

**8) name=input("Enter Student Name to get Marks: ")**

**6) d[name]=marks**

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

**2) d={}**



|  |
| --- |
| **24) Enter Student Name: banny** |
| **25) Enter Student Marks: 80** |
| **26) Enter Student Name: chinny** |
| **27) Enter Student Marks: 70** |
| **28) Enter Student Name: pinny** |
| **29) Enter Student Marks: 60** |
| **30) Enter Student Name: vinny** |
| **31) Enter Student Marks: 50** |
| **32) Enter Student Name to get Marks: sunny** |
| **33) The Marks of sunny are 90** |
| **34) Do you want to find another student marks[Yes|No]Yes** |
| **35) Enter Student Name to get Marks: durga** |
| **36) Student Not Found** |
| **37) Do you want to find another student marks[Yes|No]No** |
| **38) Thanks for using our application** |

# Dictionary Comprehension:

Comprehension concept applicable for dictionaries also.

|  |  |
| --- | --- |
| **1.** | **squares={x:x\*x for x in range(1,6)}** |
| **2.** | **print(squares)** |
| **3.** | **doubles={x:2\*x for x in range(1,6)}** |
| **4.** | **print(doubles)** |
| **5.** | |
| **6.** | **Output** |
| **7.** | **{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}** |
| **8.** | **{1: 2, 2: 4, 3: 6, 4: 8, 5: 10}** |