

## Lecture-24

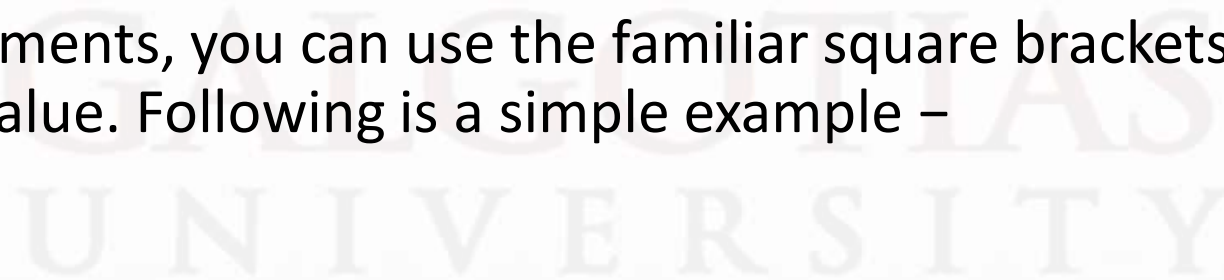
### **Built-in dictionary functions and methods :**

#### **Dictionary**

A dictionary object is an unordered collection of data in a key:value pair form. A collection of such pairs is enclosed in curly brackets. For example: {1:"Steve", 2:"Bill", 3:"Ram", 4: "Farha"}

#### **Accessing Values in Dictionary**

To access dictionary elements, you can use the familiar square brackets along with the key to obtain its value. Following is a simple example –



# School of Basic and Applied Sciences

**Course Code : BSCM 304**

**Course Name: Programming Using Python**

```
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}  
print "dict['Name']: ", dict['Name']  
print "dict['Age']: ", dict['Age']
```

When the above code is executed, it produces the following result –

```
dict['Name']: Zara  
dict['Age']: 7
```

If we attempt to access a data item with a key, which is not part of the dictionary, we get an error as follows –

```
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}  
print "dict['Alice']: ", dict['Alice']
```

When the above code is executed, it produces the following result –

```
dict['Alice']:
```

```
Traceback (most recent call last):
```

```
  File "test.py", line 4, in <module>  
    print "dict['Alice']: ", dict['Alice'];
```

```
KeyError: 'Alice'
```

GALGOTIAS  
UNIVERSITY

## Updating Dictionary:

You can update a dictionary by adding a new entry or a key-value pair, modifying an existing entry, or deleting an existing entry as shown below in the simple example –

```
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}  
dict['Age'] = 8; # update existing entry  
dict['School'] = "DPS School"; # Add new entry  
print "dict['Age']: ", dict['Age']  
print "dict['School']: ", dict['School']
```

When the above code is executed, it produces the following result –

```
dict['Age']: 8  
dict['School']: DPS School
```



## Delete Dictionary Elements

You can either remove individual dictionary elements or clear the entire contents of a dictionary. You can also delete entire dictionary in a single operation.

To explicitly remove an entire dictionary, just use the **del** statement. Following is a simple example –

```
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
del dict['Name'];           # remove entry with key 'Name'
dict.clear();              # remove all entries in dict
del dict ;                 # delete entire dictionary

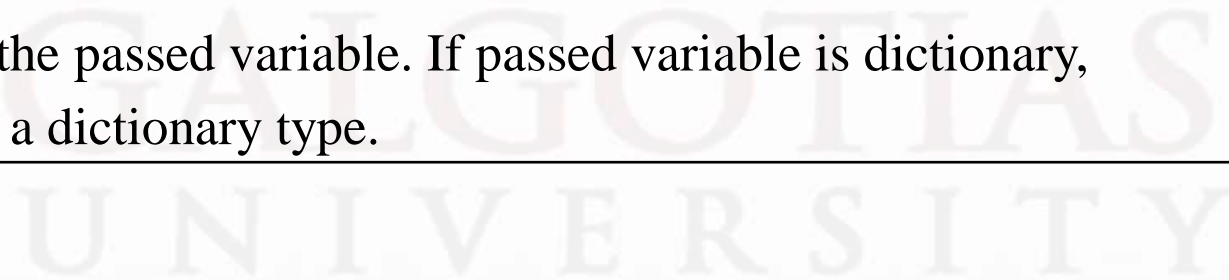
print "dict['Age']: ", dict['Age']
print "dict['School']: ", dict['School']
```

This produces the following result. Note that an exception is raised because after **del dict** dictionary does not exist any more –

```
dict['Age']:
Traceback (most recent call last):
  File "test.py", line 8, in <module>
    print "dict['Age']: ", dict['Age'];
TypeError: 'type' object is unsubscriptable
```

## Python includes the following dictionary functions –

Sr.No	Function with Description
1	<a href="#"><u>cmp(dict1, dict2)</u></a> Compares elements of both dict.
2	<a href="#"><u>len(dict)</u></a> Gives the total length of the dictionary. This would be equal to the number of items in the dictionary.
3	<a href="#"><u>str(dict)</u></a> Produces a printable string representation of a dictionary
4	<a href="#"><u>type(variable)</u></a> Returns the type of the passed variable. If passed variable is dictionary, then it would return a dictionary type.



## 1. cmp(dict1, dict2):

Python dictionary method **cmp()** compares two dictionaries based on key and values.

### Syntax

```
cmp(dict1, dict2)
```

### Example:

```
dict1 = {'Name': 'Zara', 'Age': 7};
```

```
dict2 = {'Name': 'Mahnaz', 'Age': 27};
```

```
dict3 = {'Name': 'Abid', 'Age': 27};
```

```
dict4 = {'Name': 'Zara', 'Age': 7};
```

```
print "Return Value : %d" % cmp (dict1, dict2)
```

```
print "Return Value : %d" % cmp (dict2, dict3)
```

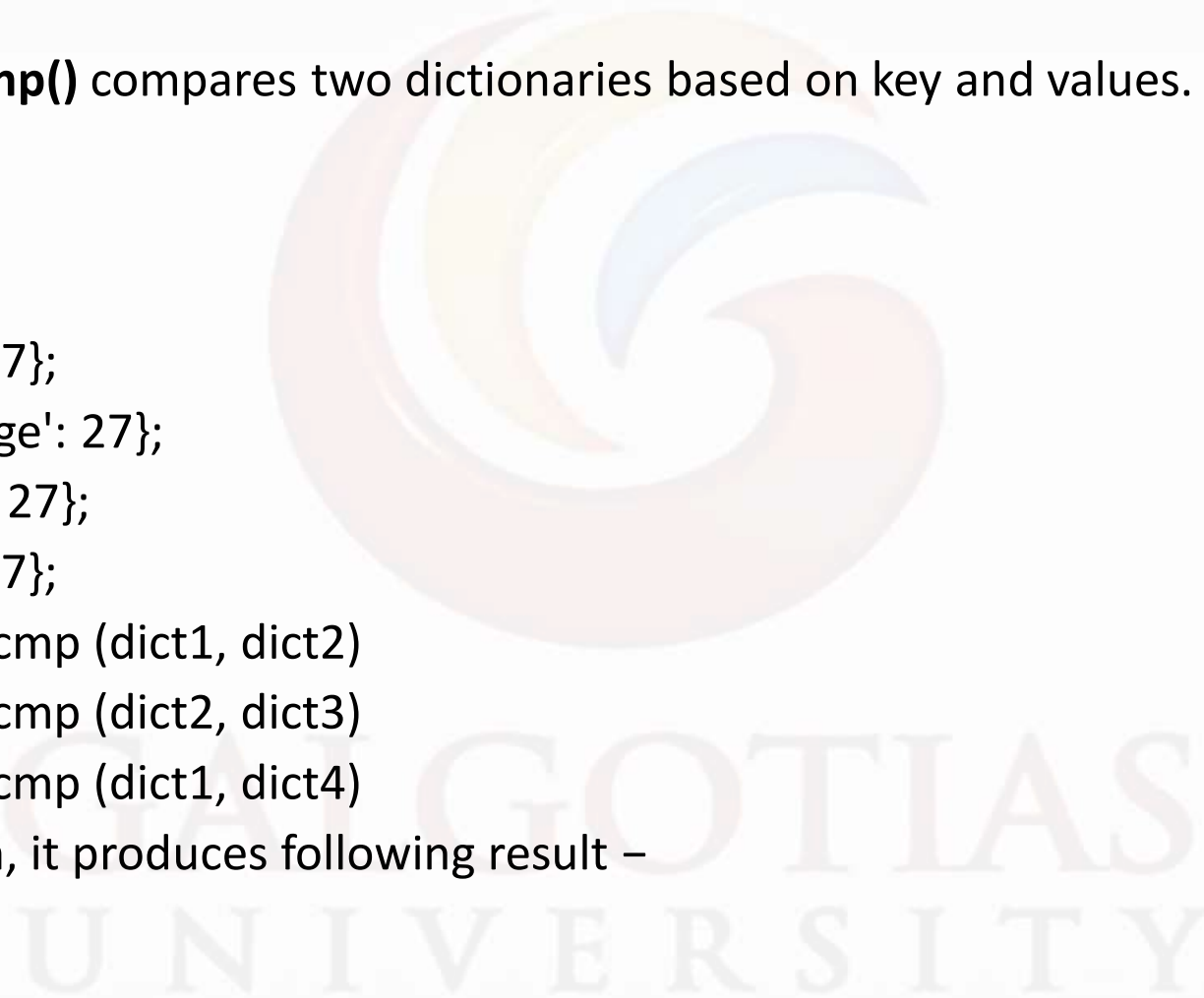
```
print "Return Value : %d" % cmp (dict1, dict4)
```

When we run above program, it produces following result –

```
Return Value : -1
```

```
Return Value : 1
```

```
Return Value : 0
```



## 2. len(dict):

Python dictionary method **len()** gives the total length of the dictionary. This would be equal to the number of items in the dictionary.

### Syntax

```
len(dict)
```

### Example

```
dict = {'Name': 'Zara', 'Age': 7};  
print "Length : %d" % len (dict)
```

When we run above program, it produces following result –

```
Length : 2
```

GALGOTIAS  
UNIVERSITY

## 3. str(dict):

Python dictionary method **str()** produces a printable string representation of a dictionary.

### Syntax

```
str(dict)
```

### Example

The following example shows the usage of str() method.

```
dict = {'Name': 'Zara', 'Age': 7};  
print "Equivalent String : %s" % str (dict)
```

When we run above program, it produces following result –

```
Equivalent String : {'Age': 7, 'Name': 'Zara'}
```

SAI GOTIAS  
UNIVERSITY



## 4. type(variable):

Python dictionary method **type()** returns the type of the passed variable. If passed variable is dictionary then it would return a dictionary type.

### Syntax

```
type(dict)
```

### • Example

- `dict = {'Name': 'Zara', 'Age': 7};`
- `print "Variable Type : %s" % type (dict)`
- When we run above program, it produces following result –
- Variable Type : <type 'dict'>

GALGOTIAS  
UNIVERSITY

## Python includes following dictionary methods:

Sr.No	Methods with Description
1	<a href="#"><u>dict.clear()</u></a> Removes all elements of dictionary dict
2	<a href="#"><u>dict.copy()</u></a> Returns a shallow copy of dictionary dict
3	<a href="#"><u>dict.fromkeys()</u></a> Create a new dictionary with keys from seq and values <i>set to value</i> .
4	<a href="#"><u>dict.get(key, default=None)</u></a> For key key, returns value or default if key not in dictionary
5	<a href="#"><u>dict.has_key(key)</u></a> Returns true if key in dictionary <i>dict</i> , <i>false</i> otherwise

# School of Basic and Applied Sciences

Course Code : BSCM 304

Course Name: Programming Using Python

Sr.No	Methods with Description
6	<a href="#">dict.items()</a> Returns a list of <i>dict's</i> (key, value) tuple pairs
7	<a href="#">dict.keys()</a> Returns list of dictionary <i>dict's</i> keys
8	<a href="#">dict.setdefault(key, default=None)</a> Similar to <code>get()</code> , but will set <code>dict[key]=default</code> if key is not already in <i>dict</i>
9	<a href="#">dict.update(dict2)</a> Adds dictionary <i>dict2's</i> key-values pairs to <i>dict</i>
10	<a href="#">dict.values()</a> Returns list of dictionary <i>dict's</i> values

## 1. dict.clear():

Python dictionary method **clear()** removes all items from the dictionary.

Syntax

```
dict.clear()
```

### **Example:**

```
dict = {'Name': 'Zara', 'Age': 7};  
print('Start dictionary=',dict)  
print "Start Len : %d" % len(dict)  
dict.clear()  
print "End Len : %d" % len(dict)  
print('End dictionary=',dict)
```

When we run above program, it produces following result –

```
('Start dictionary=', {'Age': 7, 'Name': 'Zara'})
```

```
Start Len : 2
```

```
End Len : 0
```

```
('End dictionary=', {})
```

GALGOTIAS  
UNIVERSITY

## 2. dict.copy():

Python dictionary method **copy()** returns a shallow copy of the dictionary.

Syntax

```
dict.copy()
```

**Example:**

```
dict1 = {'Name': 'Zara', 'Age': 7};
```

```
dict2 = dict1.copy()
```

```
print "New Dictionary : %s" % str(dict2)
```

When we run above program, it produces following result –

```
New Dictionary : {'Age': 7, 'Name': 'Zara'}
```

SAI GOTIAS  
UNIVERSITY

## 3. dict.fromkeys():

Python dictionary method **fromkeys()** creates a new dictionary with keys from *seq* and *values* set to value.

### **Syntax**

```
dict.fromkeys(seq[, value])
```

### **Example:**

```
seq = ('name', 'age', 'sex')
```

```
dict = dict.fromkeys(seq)
```

```
print "New Dictionary : %s" % str(dict)
```

```
dict = dict.fromkeys(seq, 10)
```

```
print "New Dictionary : %s" % str(dict)
```

When we run above program, it produces following result –

```
New Dictionary : {'age': None, 'name': None, 'sex': None}
```

```
New Dictionary : {'age': 10, 'name': 10, 'sex': 10}
```

## 4. dict.get(key, default=None):

Python dictionary method **get()** returns a value for the given key. If key is not available then returns default value None.

### **Syntax**

```
dict.get(key, default = None)
```

### **Example**

```
dict = {'Name': 'Zabra', 'Age': 7}
print "Value : %s" % dict.get('Age')
print "Value : %s" % dict.get('Education', "Never")
print "Value : %s" % dict.get('Education')
```

When we run above program, it produces following result –

Value : 7

Value : Never

Value : None

GALGOTIAS  
UNIVERSITY

## 5. dict.has\_key(key):

Python dictionary method **has\_key()** returns true if a given *key* is available in the dictionary, otherwise it returns a false.

### **Syntax**

```
dict.has_key(key)
```

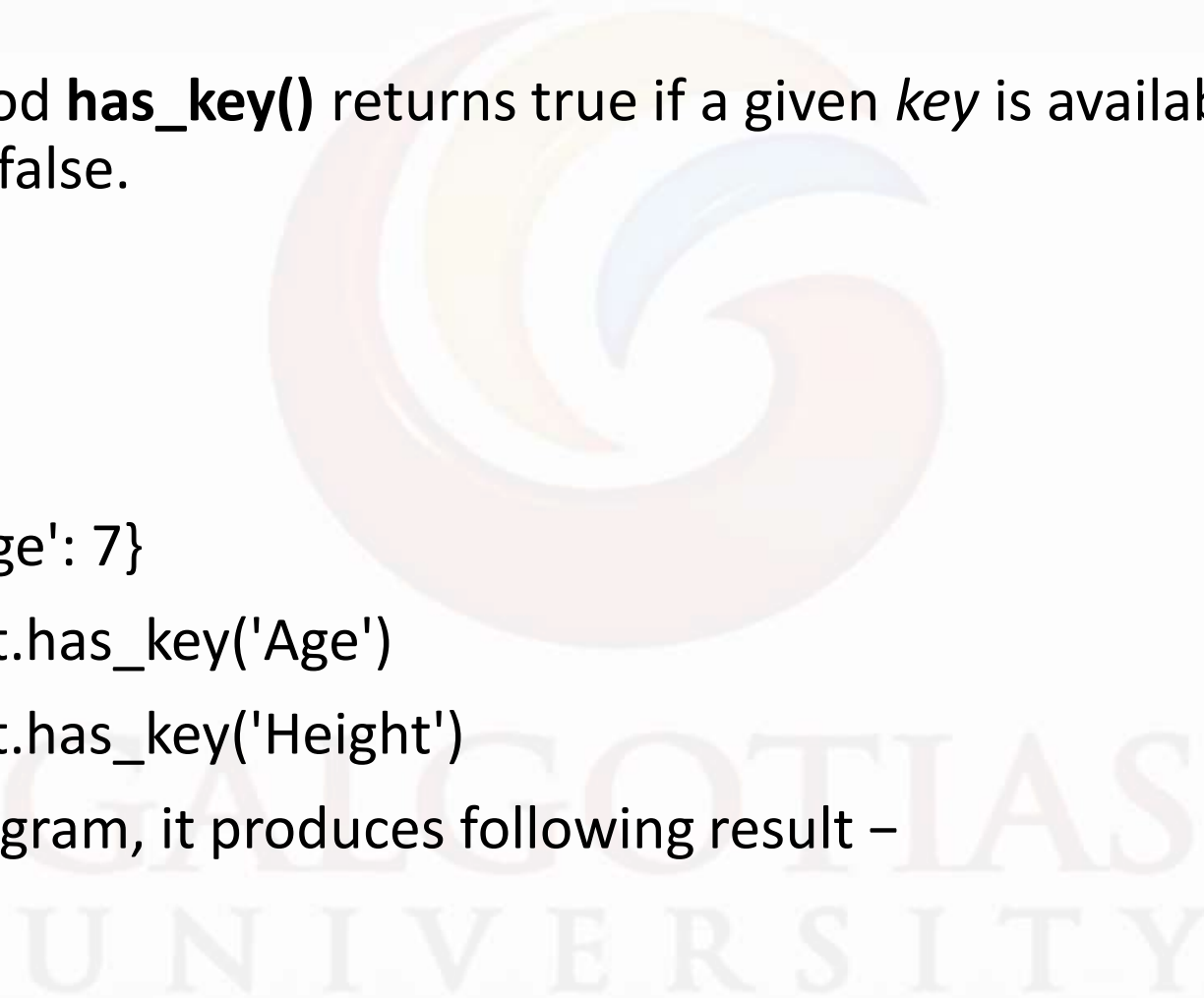
### **Example**

```
dict = {'Name': 'Zara', 'Age': 7}
print "Value : %s" % dict.has_key('Age')
print "Value : %s" % dict.has_key('Height')
```

When we run above program, it produces following result –

```
Value : True
```

```
Value : False
```





## 6. dict.items():

Python dictionary method **items()** returns a list of dict's (key, value) tuple pairs

### **Syntax**

```
dict.items()
```

### **Example**

```
dict = {'Name': 'Zara', 'Age': 7}
print "Value : %s" % dict.items()
```

When we run above program, it produces following result –

```
Value : [('Age', 7), ('Name', 'Zara')]
```



## 7. dict.setdefault(key, default=None):

Python dictionary method **setdefault()** is similar to `get()`, but will set `dict[key]=default` if key is not already in dict.

Syntax

```
dict.setdefault(key, default=None)
```

**Example**

```
dict = {'Name': 'Zara', 'Age': 7}
```

```
print "Value : %s" % dict.setdefault('Age', None)
```

```
print "Value : %s" % dict.setdefault('Sex', None)
```

When we run above program, it produces following result –

```
Value : 7
```

```
Value : None
```

## 8. dict.update(dict2):

Python dictionary method **update()** adds dictionary dict2's key-values pairs in to dict. This function does not return anything.

### Syntax

```
dict.update(dict2)
```

### Example

```
dict = {'Name': 'Zara', 'Age': 7}
```

```
dict2 = {'Sex': 'female' }
```

```
dict.update(dict2)
```

```
print "Value : %s" % dict
```

When we run above program, it produces following result –

```
Value : {'Age': 7, 'Name': 'Zara', 'Sex': 'female'}
```

## 9. dict.keys():

Python dictionary method **keys()** returns a list of all the available keys in the dictionary.

### **Syntax**

```
dict.keys()
```

### **Example**

```
dict = {'Name': 'Zara', 'Age': 7}
print "Value : %s" % dict.keys()
```

When we run above program, it produces following result –

```
Value : ['Age', 'Name']
```

## 10. dict.values():

Python dictionary method **values()** returns a list of all the values available in a given dictionary.

### **Syntax**

```
dict.values()
```

### **Example**

```
dict = {'Name': 'Zara', 'Age': 7}
print "Value : %s" % dict.values()
```

When we run above program, it produces following result –

```
Value : [7, 'Zara']
```

GALGOTIAS  
UNIVERSITY

## References:

1. Introduction to Computation and Programming using Python, by John Guttag, PHI Publisher
2. Fundamentals of Python first Programmes by Kenneth A Lambert, Copyrighted material Course Technology Inc. 1 st edition (6th February 2009)
3. <https://www.tutorialspoint.com/python/index.htm>
4. <https://www.geeksforgeeks.org/python-programming-language>

GALGOTIAS  
UNIVERSITY

**\*\*\*END OF THE LECTURE\*\*\***

**\*\*\*THANK YOU\*\*\***

GALGOTIAS  
UNIVERSITY