



# DICTIONARIES



**WHAT IS DICTIONARY?**



## WHAT IS DICTIONARY?

A dictionary is like a list, but more in general. In a list, index value is an integer, while in a dictionary index value can be any other data type and are called keys. The key will be used as a string as it is easy to recall. A dictionary is an extremely useful data storage construct for storing and retrieving all key value pairs, where each element is accessed (or indexed) by a unique key. However, dictionary keys are not in sequences and hence maintain no left-to right order.



## KEY VALYE PAIR

We can refer to a dictionary as a mapping between a set of indices (which are called keys) and a set of values. Each key maps a value. The association of a key and a value is called a key-value pair.

Syntax:

```
my_dict    =    {'key1':    'value1','key2':  
'value2','key3': 'value3'...'keyn': 'valuen'}
```



# DICTIONARIES

- ✓ Curley brackets are used to represent a dictionary.
- ✓ Each pair in the dictionary is represented by a key and value separated by a colon.
- ✓ Multiple pairs are separated by vcomas





# DICTIONARIES

- ✓ A dictionary is an unordered collection of key-value pairs.
- ✓ A dictionary has a length, specifically the number of keyvalue pairs.
- ✓ A dictionary provides fast look up by key.
- ✓ The keys must be immutable object types.

# STATE DIAGRAM

>>> A={1:"one",2:"two",3:"three"}

A =

1  
2  
3

one  
two  
three

KEYS

VALUES

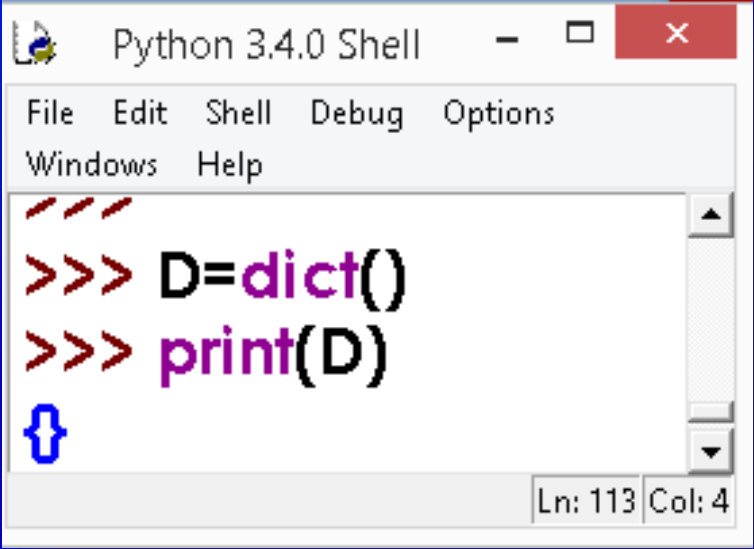


**CREATING A DICTIONARY – dict()**



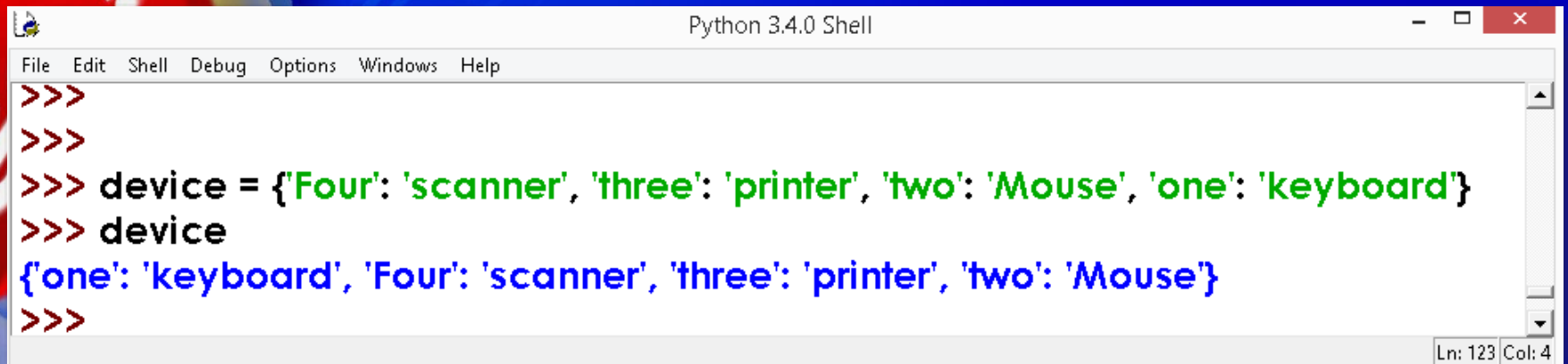
## CREATING DICTIONARY – dict()

The function `dict ( )` is used to create a new dictionary with no items. This function is called built-in function. We can also create dictionary using `{}`.

A screenshot of a Python 3.4.0 Shell window. The window has a title bar with the text 'Python 3.4.0 Shell' and standard window controls. Below the title bar is a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'. The main area of the window contains a Python prompt '>>>' followed by the code 'D=dict()' and 'print(D)' on two lines. The output of the code is an empty dictionary '{}'. The status bar at the bottom right shows 'Ln: 113 Col: 4'.

```
Python 3.4.0 Shell
File Edit Shell Debug Options
Windows Help
>>> D=dict()
>>> print(D)
{}
Ln: 113 Col: 4
```

# CREATING DICTIONARAY – Example

A screenshot of a Python 3.4.0 Shell window. The window has a title bar with the text "Python 3.4.0 Shell" and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with options: File, Edit, Shell, Debug, Options, Windows, and Help. The main area of the window contains a Python interactive prompt with the following code:

```
>>>  
>>>  
>>> device = {'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'}  
>>> device  
{'one': 'keyboard', 'Four': 'scanner', 'three': 'printer', 'two': 'Mouse'}  
>>>
```

The output of the dictionary is displayed in blue text. At the bottom right of the window, a status bar shows "Ln: 123 Col: 4".

```
Python 3.4.0 Shell  
File Edit Shell Debug Options Windows Help  
>>>  
>>>  
>>> device = {'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'}  
>>> device  
{'one': 'keyboard', 'Four': 'scanner', 'three': 'printer', 'two': 'Mouse'}  
>>>  
Ln: 123 Col: 4
```



# CREATING AND TRAVERSING DICTIONARAY

# CREATING AND TRAVERSING DICTIONARAY

```
*Python 3.4.0: dictex.py - C:/Python34/dictex.py*
File Edit Format Run Options Windows Help
def Creating_Dictionary():
    device = {'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'}
    for i in device:
        print(device[i])
Creating_Dictionary()
Ln: 7 Col: 0
```

OUT PUT

```
Python 3.4...
File Edit Shell Debug
Options Windows Help
scanner
keyboard
printer
Mouse
>>> |
Ln: 129 Col: 4
```

# CREATING AND TRAVERSING DICTIONARY

```
Python 3.4.0: dict2.py - C:/Python34/dict2.py
File Edit Format Run Options Windows Help

def create_dict():
    D=dict()
    D["one"]="C++"
    D["two"]="Java"
    D["three"]="Python"
    D["four"]="Pascal"
    for i in D:
        print(D[i])
create_dict()

Ln: 10 Col: 0
```

OUT PUT

```
Python 3.4.0 Shell
File Edit Shell Debug Options Windows Help

>>>
Java
Pascal
C++
Python
>>>

Ln: 13 Col: 4
```





## **DICTIONARAY – BUILT IN METHODS**

## DICTIONARAY – BUILT IN METHODS

Dictionary Method	Meaning
<code>dict.clear()</code>	Removes all the elements of the dictionary
<code>dict.copy()</code>	Returns (shallow)copy of dictionary.
<code>dict.get(key, default=None)</code>	for key key, returns value or default if key not in dictionary (note that default's default is None)
<code>dict.items()</code>	returns a list of dict's (key, value) tuple pairs


## DICTIONARAY – BUILT IN METHODS

Dictionary Method	Meaning
<code>dict.keys()</code>	returns list of dictionary dict's keys
<code>dict.setdefault key, default=None</code>	similar to <code>get()</code> , but will set <code>dict[key]=default</code> if key is not already in dict
<code>dict.update(dict2)</code>	adds dictionary dict2's key-values pairs to dict
<code>dict.values()</code>	returns list of dictionary dict's values

## DICTIONARAY – BUILT IN METHODS

Dictionary Method	Meaning
<code>dict.pop()</code>	returns list of dictionary dict's keys
<code>dict.popitem()</code>	similar to <code>get()</code> , but will set <code>dict[key]=default</code> if key is not already in dict

# dict.clear() METHOD



```
Python 3.4.0: dict3.py - C:/Python34/dict3.py
File Edit Format Run Options Windows Help

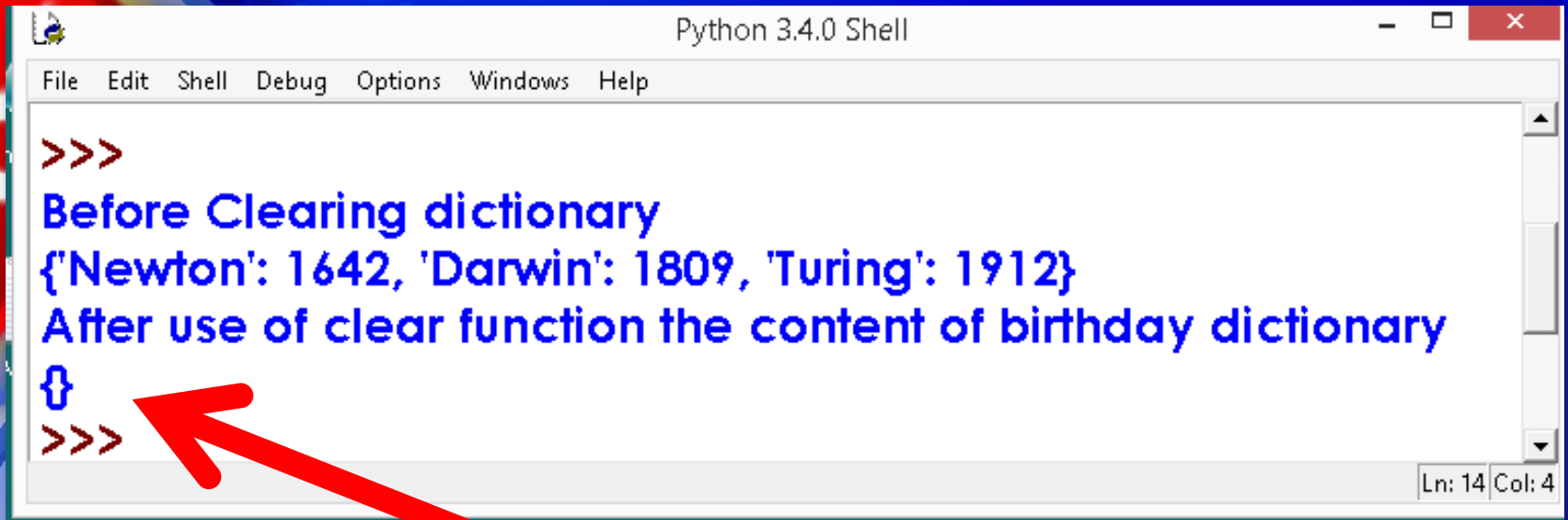
def create_dict():
    birthday={
        'Newton':1642,
        'Darwin':1809,
        'Turing':1912
    }
    print("Before Clearing dictionary")
    print(birthday)
    birthday.clear()
    print("After use of clear function the content of birthday dictionary")
    print(birthday)
create_dict()
```

clear method

OUTPUT is in next slide!



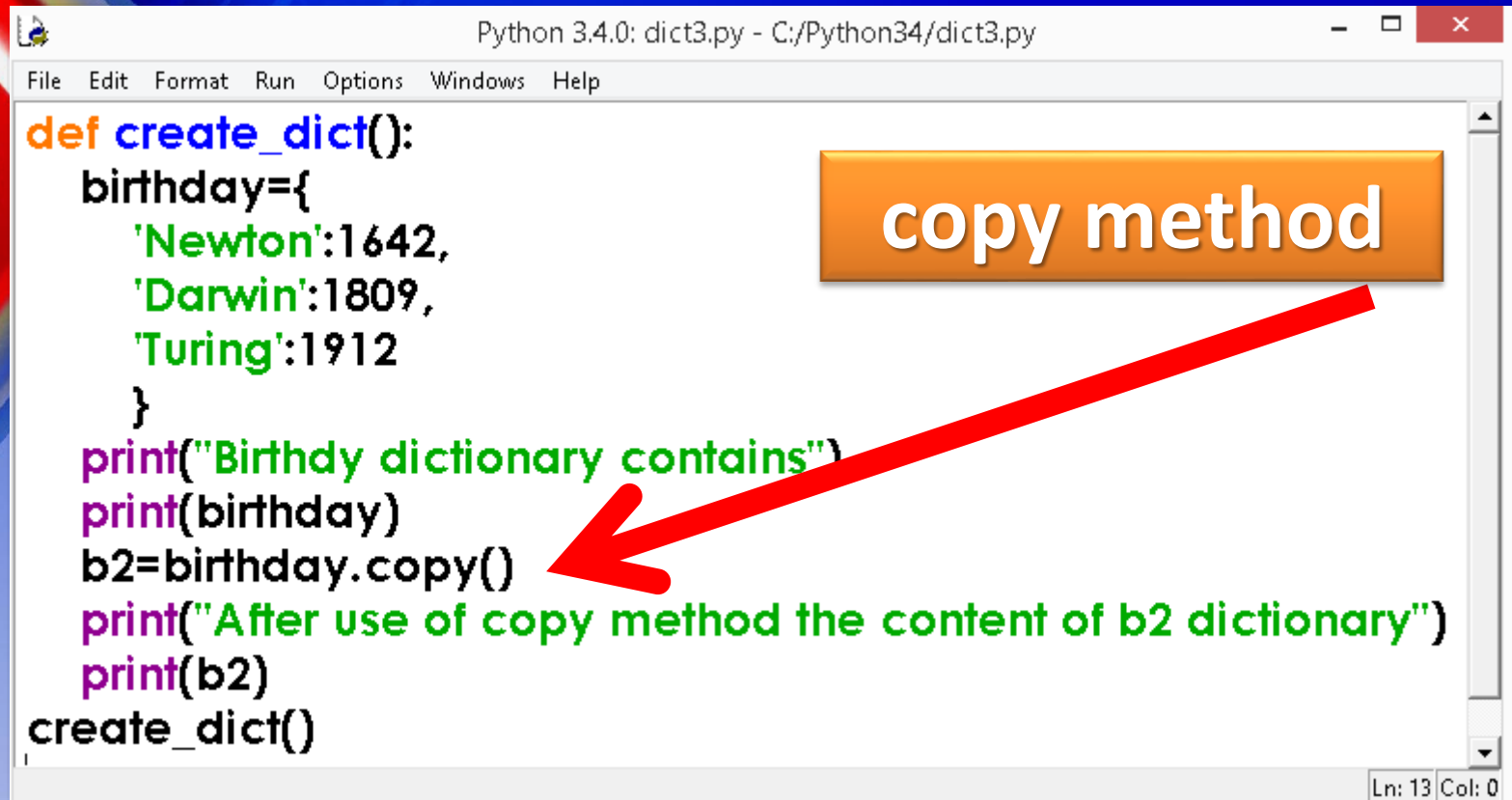
## dict.clear() METHOD - OUTPUT

A screenshot of a Python 3.4.0 Shell window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'. The main text area shows a Python prompt '>>>' followed by blue text: 'Before Clearing dictionary', '{\'Newton\': 1642, \'Darwin\': 1809, \'Turing\': 1912}', 'After use of clear function the content of birthday dictionary', and an empty dictionary '{}'. Another blue '>>>' prompt is on the next line. A large red arrow points from the bottom of the window to an orange box at the bottom of the slide. The status bar at the bottom right of the window shows 'Ln: 14 Col: 4'.

```
>>>
Before Clearing dictionary
{'Newton': 1642, 'Darwin': 1809, 'Turing': 1912}
After use of clear function the content of birthday dictionary
{}
>>>
```

**Birthday dictionary cleared**

# dict.copy() METHOD



```
Python 3.4.0: dict3.py - C:/Python34/dict3.py
File Edit Format Run Options Windows Help

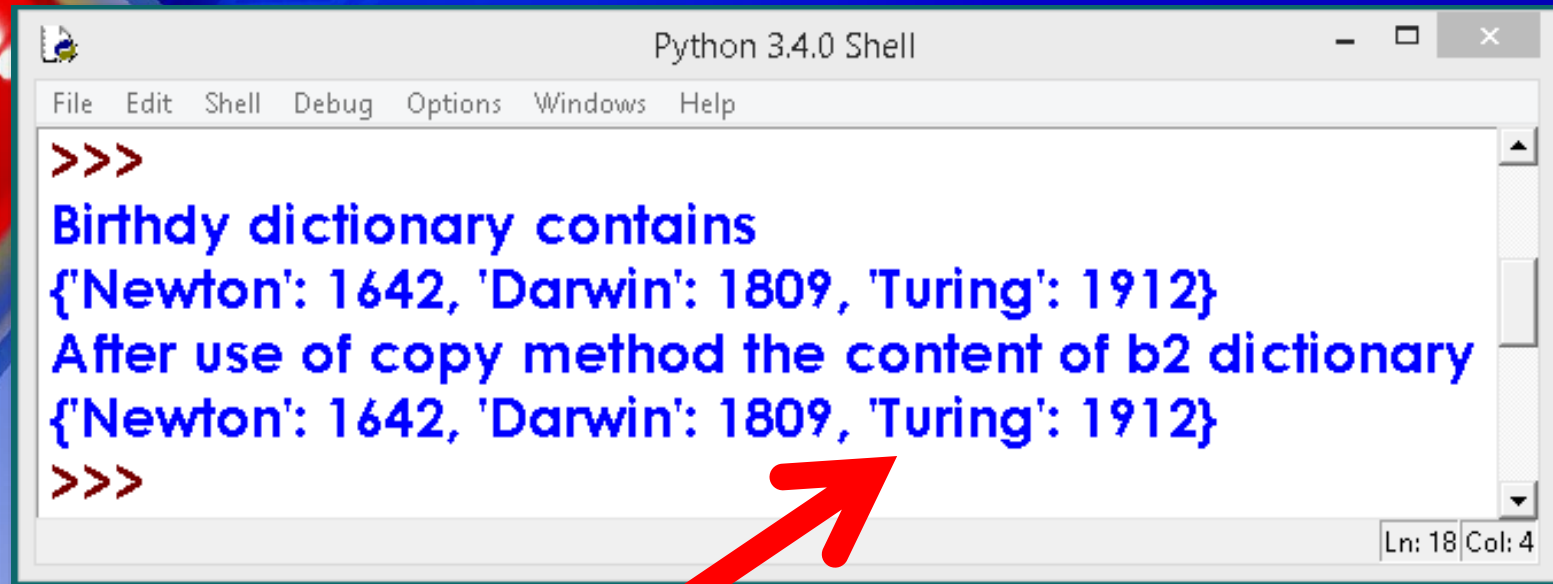
def create_dict():
    birthday={
        'Newton':1642,
        'Darwin':1809,
        'Turing':1912
    }
    print("Birthdy dictionary contains")
    print(birthday)
    b2=birthday.copy()
    print("After use of copy method the content of b2 dictionary")
    print(b2)
create_dict()
```

copy method

Ln: 13 Col: 0

OUTPUT is in next slide!

## dict.copy() METHOD - OUTPUT




A screenshot of a Python 3.4.0 Shell window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'. The main text area shows a Python prompt '>>>' followed by blue text: 'Birthdy dictionary contains', '{\'Newton\': 1642, \'Darwin\': 1809, \'Turing\': 1912}', 'After use of copy method the content of b2 dictionary', '{\'Newton\': 1642, \'Darwin\': 1809, \'Turing\': 1912}', and another '>>>' prompt. A red arrow points from the bottom of the window to an orange box below. The status bar at the bottom right shows 'Ln: 18 Col: 4'.

```
>>>
Birthdy dictionary contains
{'Newton': 1642, 'Darwin': 1809, 'Turing': 1912}
After use of copy method the content of b2 dictionary
{'Newton': 1642, 'Darwin': 1809, 'Turing': 1912}
>>>
```

Ln: 18 Col: 4

copy method creates b2 dictionary

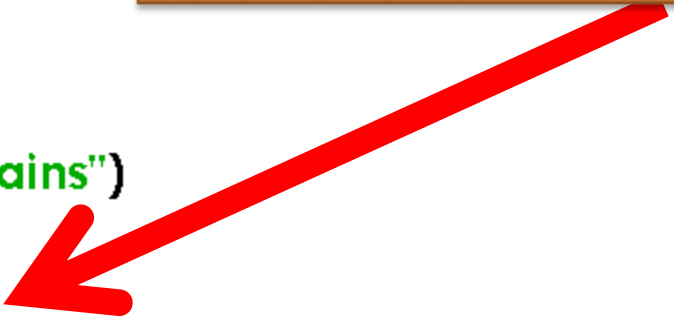
# dict.get() METHOD



```
Python 3.4.0: dict4.py - C:/Python34/dict4.py
File Edit Format Run Options Windows Help

def create_dict():
    birthday={
        'Newton':1642,
        'Darwin':1809,
        'Turing':1912
    }
    print("Birthdy dictionary contains")
    print(birthday)
    b2=birthday.get('Newton')
    print("After use of get method the content of b2 dictionary")
    print(b2)
create_dict()
```

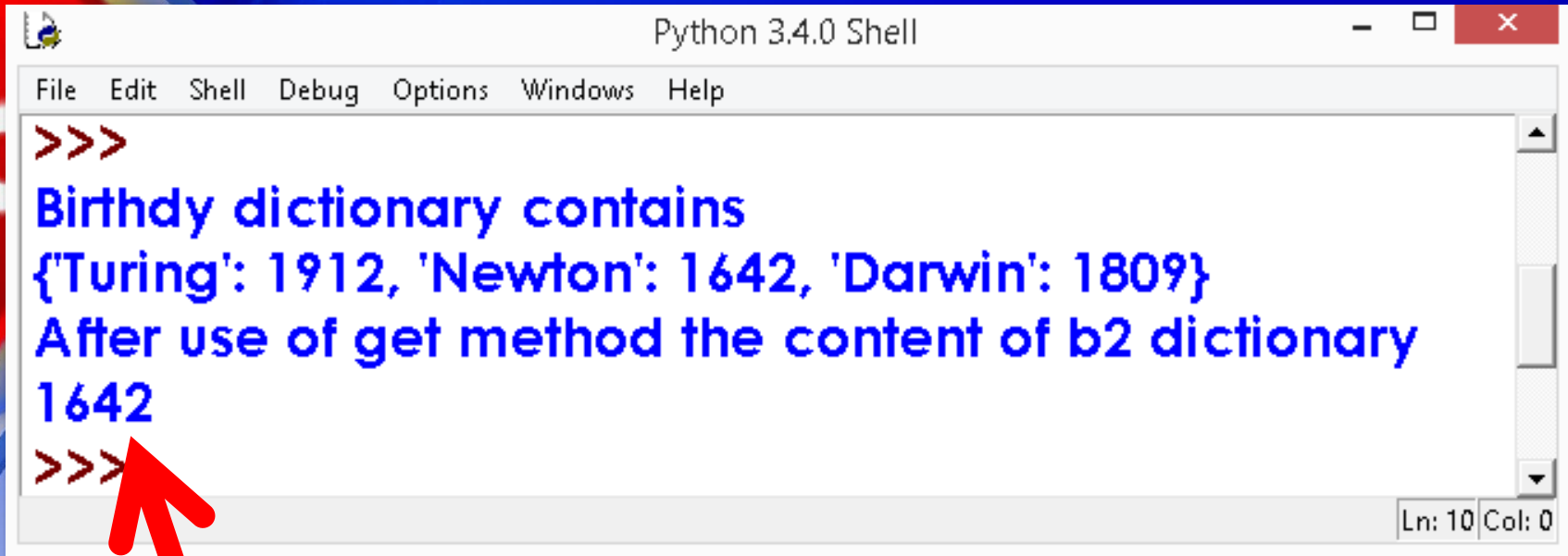
get method



Ln: 13 Col: 0

OUTPUT is in next slide!

## dict.get() METHOD - OUTPUT


A screenshot of a Python 3.4.0 Shell window. The window has a title bar with the text 'Python 3.4.0 Shell' and standard window controls. Below the title bar is a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'. The main text area contains the following text: '>>>' followed by 'Birthdy dictionary contains' (note the typo), then a dictionary literal '{'Turing': 1912, 'Newton': 1642, 'Darwin': 1809}', then 'After use of get method the content of b2 dictionary' (note the typo), then the value '1642', and finally another '>>>' prompt. A large red arrow points from the bottom of the window towards the text 'Creating a b2 dictionary using get method' in the block below. The status bar at the bottom right shows 'Ln: 10 Col: 0'.

```
>>>
Birthdy dictionary contains
{'Turing': 1912, 'Newton': 1642, 'Darwin': 1809}
After use of get method the content of b2 dictionary
1642
>>>
```

Creating a b2 dictionary using get method




# dict.items() METHOD



```
Python 3.4.0: dict4.py - C:/Python34/dict4.py
File Edit Format Run Options Windows Help

def create_dict():
    birthday={
        'Newton':1642,
        'Darwin':1809,
        'Turing':1912
    }
    print("Birthdy dictionary contains")
    print(birthday)
    b2=birthday.items()
    print("After use of items method the content of b2 dictionary")
    print(b2)
create_dict()
```

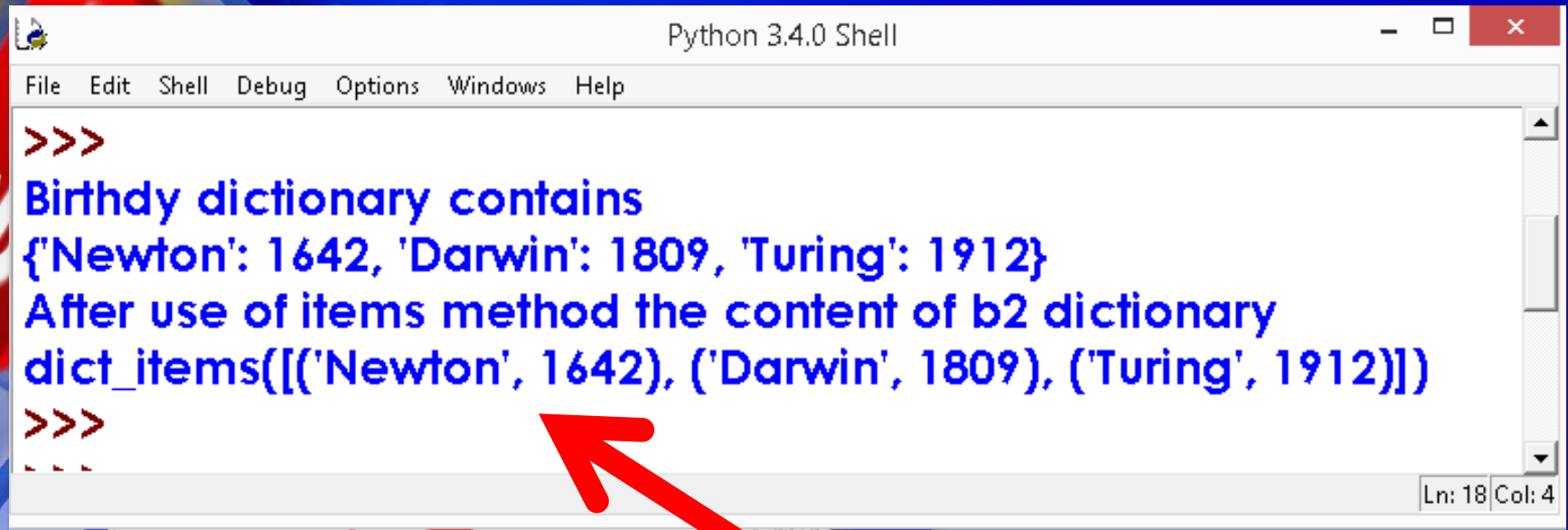
items method



Ln: 12 Col: 13

OUTPUT is in next slide!

## dict.items() METHOD - OUTPUT



A screenshot of a Python 3.4.0 Shell window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'. The main text area shows a Python prompt '>>>' followed by blue text: 'Birthdy dictionary contains', '{'Newton': 1642, 'Darwin': 1809, 'Turing': 1912}', 'After use of items method the content of b2 dictionary', and 'dict\_items([('Newton', 1642), ('Darwin', 1809), ('Turing', 1912)])'. Another '>>>' prompt is on the next line. A red arrow points from the bottom of the window to a green box at the bottom of the slide. The status bar at the bottom right of the window shows 'Ln: 18 Col: 4'.

```
>>>
Birthdy dictionary contains
{'Newton': 1642, 'Darwin': 1809, 'Turing': 1912}
After use of items method the content of b2 dictionary
dict_items([('Newton', 1642), ('Darwin', 1809), ('Turing', 1912)])
>>>
```

items method returns dictionary content

## dict.keys() METHOD



```
*Python 3.4.0: dict8.py - C:/Python34/dict8.py*
File Edit Format Run Options Windows Help

def create_dict():
    birthday={
        'Newton':1642,
        'Darwin':1809,
        'Turing':1912
    }
    print('dictionary keys are:',birthday.keys())
create_dict()
```

Ln: 11 Col: 0

OUTPUT is in next slide!

## dict.keys() METHOD - OUTPUT

A screenshot of a Python 3.4.0 Shell window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'. The main text area shows a prompt '>>>' followed by the text 'dictionary keys are: dict\_keys(['Darwin', 'Turing', 'Newton'])' in blue, and another prompt '>>>'. A red arrow points from the text 'keys method returns dictionary keys' below to the 'dict\_keys' part of the output. The status bar at the bottom right shows 'Ln: 13 Col: 4'.

```
Python 3.4.0 Shell
File Edit Shell Debug Options Windows Help
>>>
dictionary keys are: dict_keys(['Darwin', 'Turing', 'Newton'])
>>>
Ln: 13 Col: 4
```

keys method returns dictionary keys

# dict.update() METHOD

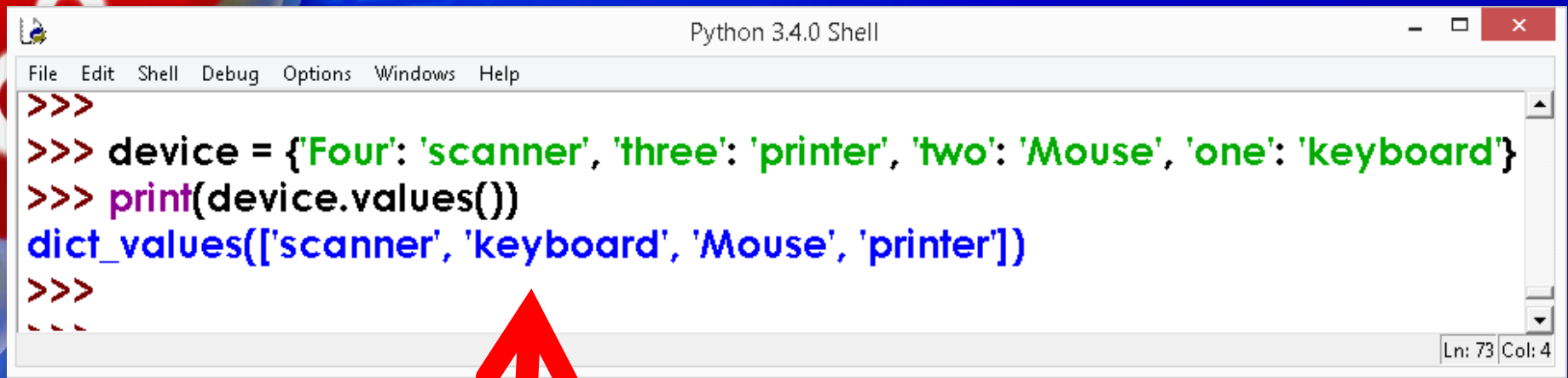
update method

```
Python 3.4.0 Shell
File Edit Shell Debug Options Windows Help
>>> device = {'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'}
>>> dev1={'Five':'Computer','Six':'CPU','Seven':'RAM'}
>>> device.update(dev1)
>>> print(device)
{'Four': 'scanner', 'two': 'Mouse', 'three': 'printer', 'Five': 'Computer', 'one': 'keyb
oard', 'Six': 'CPU', 'Seven': 'RAM'}
>>>
>>>
```

Ln: 67 Col: 4



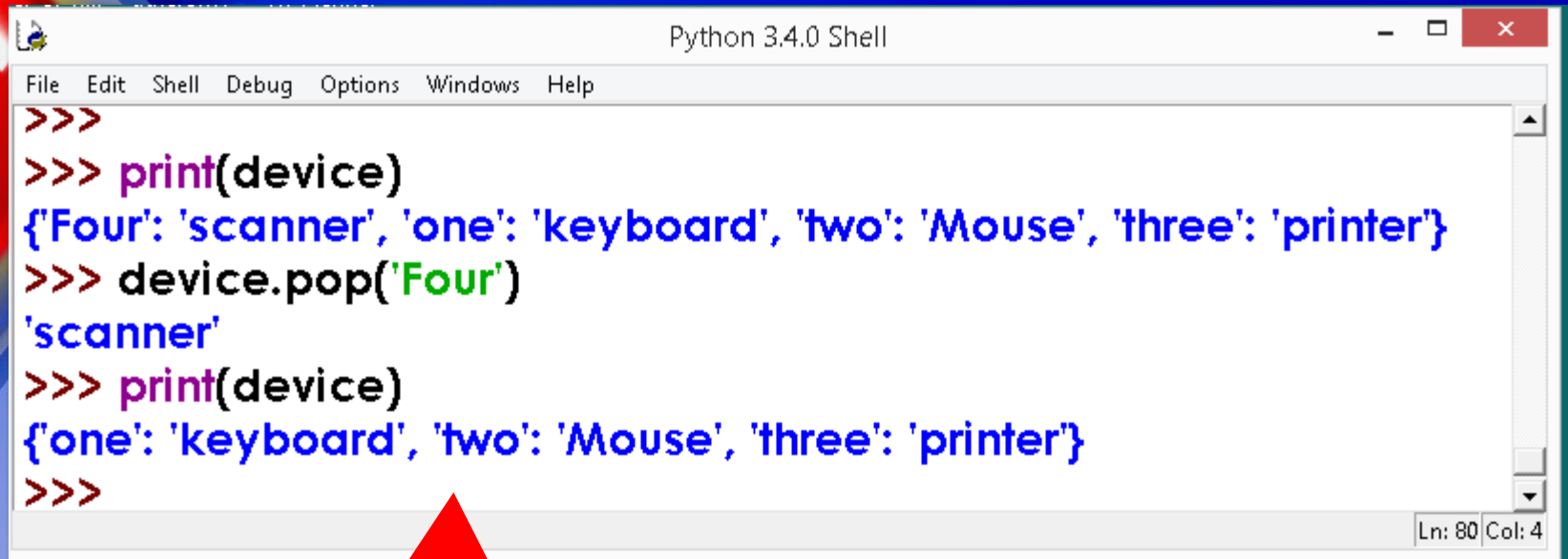
## dict.values() METHOD

A screenshot of a Python 3.4.0 Shell window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'. The main text area shows a Python script: three prompt characters '>>>' followed by 'device = {'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'}', then 'print(device.values())', and finally 'dict\_values(['scanner', 'keyboard', 'Mouse', 'printer'])'. Below this is another set of three prompt characters '>>>'. A large red arrow points from the bottom of the window up to the 'dict\_values' line. The status bar at the bottom right shows 'Ln: 73 Col: 4'.

```
>>>  
>>> device = {'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'}  
>>> print(device.values())  
dict_values(['scanner', 'keyboard', 'Mouse', 'printer'])  
>>>
```

values method returns dictionary values

## dict.pop() METHOD




A screenshot of a Python 3.4.0 Shell window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'. The main text area shows the following code:   
>>>   
>>> print(device)   
{'Four': 'scanner', 'one': 'keyboard', 'two': 'Mouse', 'three': 'printer'}   
>>> device.pop('Four')   
'scanner'   
>>> print(device)   
{'one': 'keyboard', 'two': 'Mouse', 'three': 'printer'}   
>>>   
A red arrow points from the bottom of the shell window to a text box below. The status bar at the bottom right of the shell window shows 'Ln: 80 Col: 4'.

```
>>>
>>> print(device)
{'Four': 'scanner', 'one': 'keyboard', 'two': 'Mouse', 'three': 'printer'}
>>> device.pop('Four')
'scanner'
>>> print(device)
{'one': 'keyboard', 'two': 'Mouse', 'three': 'printer'}
>>>
```

Ln: 80 Col: 4

pop method removes specified key values from the dictionary

## dict.popitem() METHOD



```
Python 3.4.0 Shell
File Edit Shell Debug Options Windows Help
>>> print(device)
{'one': 'keyboard', 'two': 'Mouse', 'three': 'printer'}
>>> device.popitem()
('one', 'keyboard')
>>> print(device)
{'two': 'Mouse', 'three': 'printer'}
>>>
```

Ln: 86 Col: 4

popitem method removes values/items from the dictionary



**K.V.Sesha Sain [Kasyap]**

**Corporate Trainer| Technical Trainer|  
Company Specific Trainer |Certified Skilled Trainer  
M.sc Comp, OCP, JCP, MDSE  
Oracle Certified Professional (2) |  
Java Certified Professional  
16+ Years of Experience**

Email:brainsncrowns@gmail.com

Mobile: 6300340023