# Python

Python
Dictionary

# Python Dictionary

An effective data structure for storing data in Python is dictionaries, in which can simulate the real-life data arrangement where some specific value exists for some particular key.

- Python Dictionary is used to store the data in a key-value pair format.
- It is the mutable data-structure.
- The elements Keys and values is employed to create the dictionary.
- Keys must consist of just one element.
- Value can be any type such as list, tuple, integer, etc.

In other words, we can say that a dictionary is the collection of key-value pairs where the value can be of any Python object. In contrast, the keys are the immutable Python object, i.e., Numbers, string, or tuple. Dictionary entries are ordered as of Python version 3.7. In Python 3.6 and before, dictionaries are generally unordered.

## Creating the Dictionary

The simplest approach to create a Python dictionary is by using curly brackets {}, but there are other methods as well. The dictionary can be created by using multiple key-value pairs enclosed with the curly brackets {}, and each key is separated from its value by the colon (:). The syntax to define the dictionary is given below.

**Syntax:**

```
Dict = {"Name": "Chris", "Age": 20}
```

In the above dictionary **Dict**, The keys **Name** and **Age** are the strings which comes under the category of an immutable object.

Let's see an example to create a dictionary and print its content.

**Code**

```
Employee = {"Name": 'John', "Age": 29, "salary":25000,"Company":"GOOGLE"}
print(type(Employee))
print("printing Employee data .... ")
print(Employee)
```

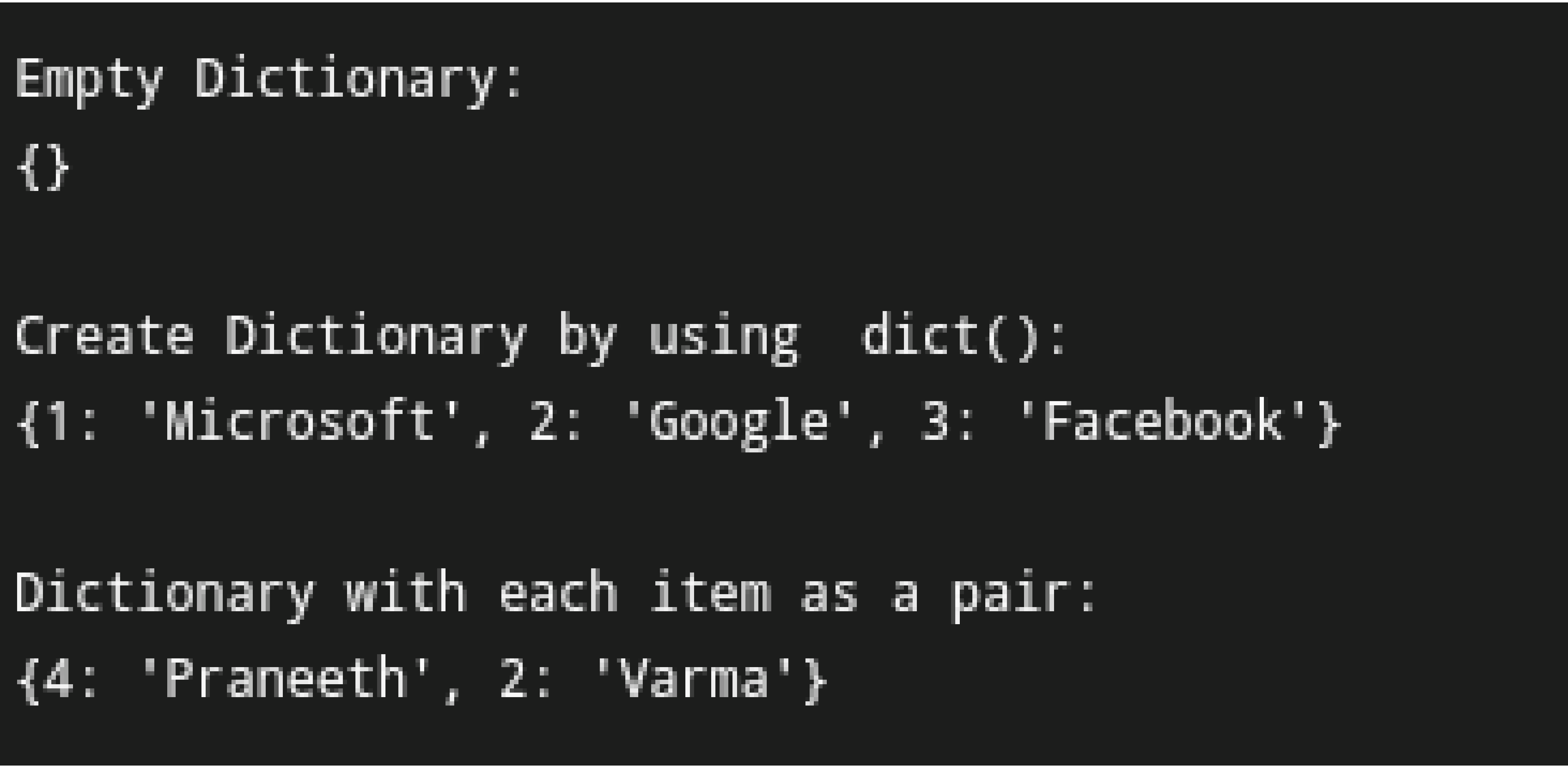
**Output**

```
<class 'dict'>
printing Employee data ....
{'Name': 'John', 'Age': 29, 'salary': 25000, 'Company': 'GOOGLE'}
```

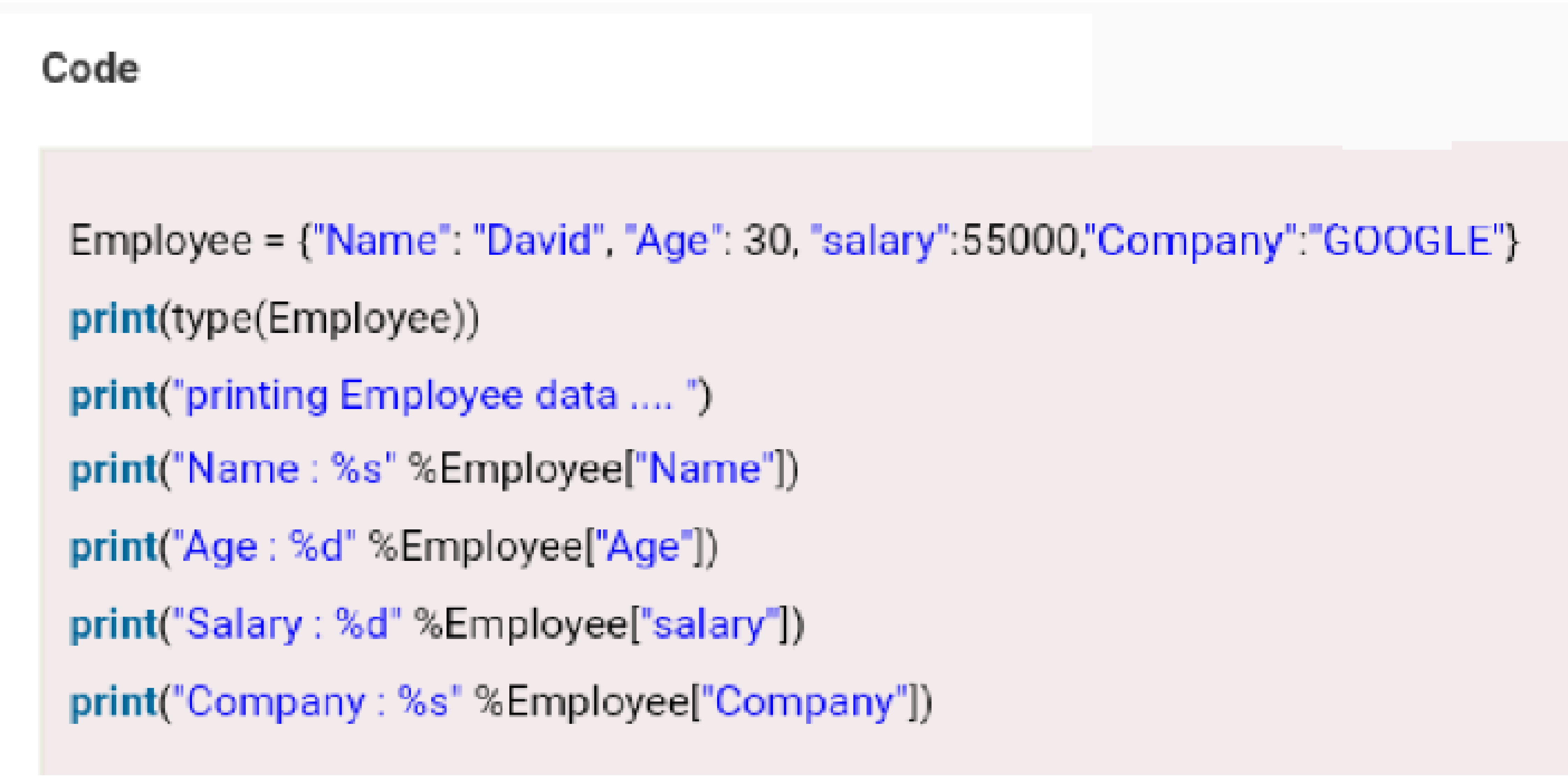
Python provides the built-in function **dict()** method which is also used to create the dictionary.

## Code

```python
# Creating an empty Dictionary
Dict = {}
print("Empty Dictionary: ")
print(Dict)

# Creating a Dictionary
# with dict() method
Dict = dict({1: 'Microsoft', 2: 'Google', 3:'Facebook'})
print("\nCreate Dictionary by using  dict(): ")
print(Dict)

# Creating a Dictionary
# with each item as a Pair
Dict = dict([(4, 'Praneeth'), (2, 'Varma')])
print("\nDictionary with each item as a pair: ")
print(Dict)
```



```
Empty Dictionary:
{}

Create Dictionary by using  dict():
{1: 'Microsoft', 2: 'Google', 3: 'Facebook'}

Dictionary with each item as a pair:
{4: 'Praneeth', 2: 'Varma'}
```



Code

```
Employee = {"Name": "David", "Age": 30, "salary":55000,"Company":"GOOGLE"}
print(type(Employee))
print("printing Employee data .... ")
print("Name : %s" %Employee["Name"])
print("Age : %d" %Employee["Age"])
print("Salary : %d" %Employee["salary"])
print("Company : %s" %Employee["Company"])
```

**Output**

```
<class 'dict'>
printing Employee data ....
Name : David
Age : 30
Salary : 55000
Company : GOOGLE
```

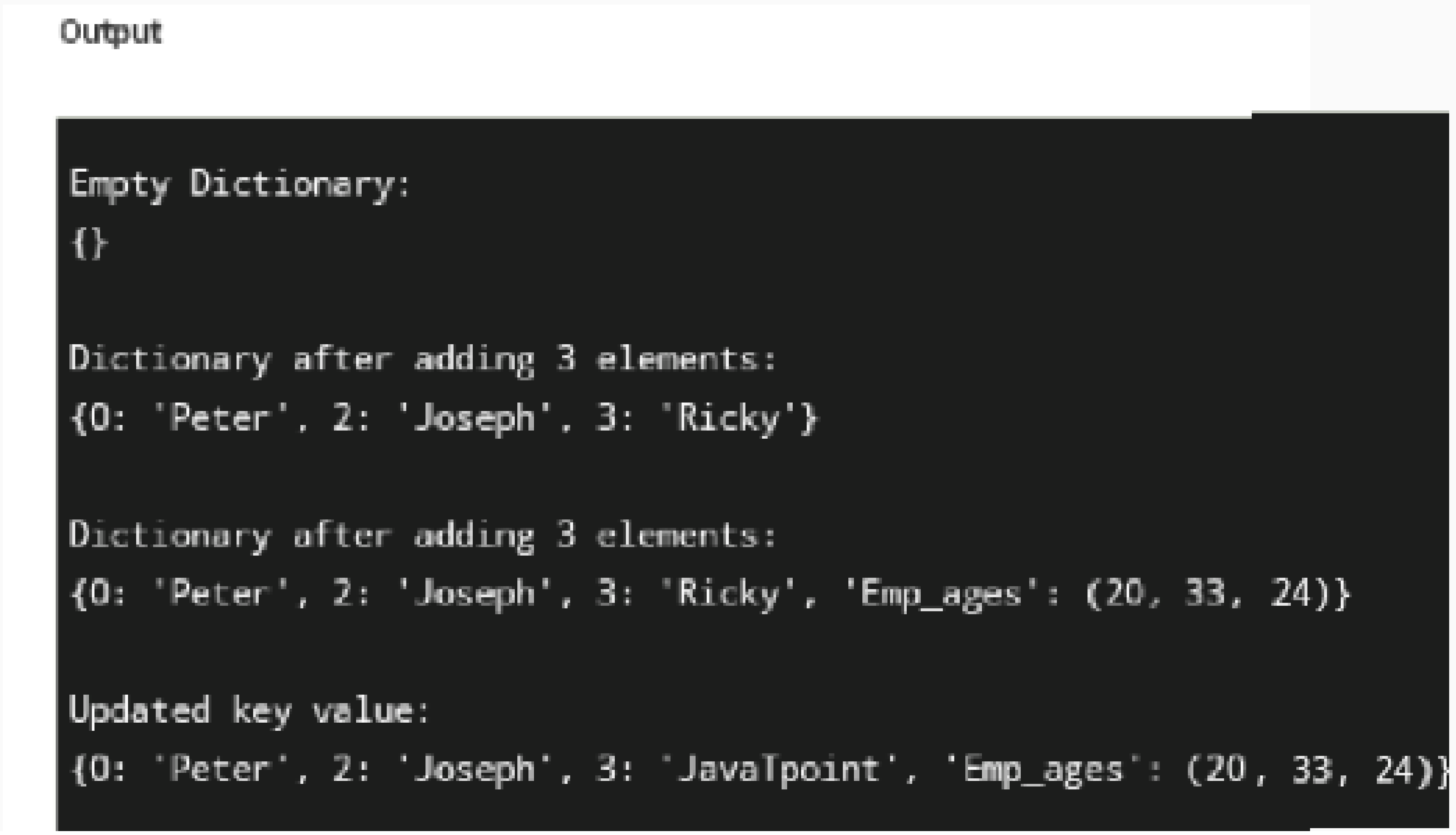
## Adding Dictionary Values

The dictionary is a mutable data type, and its values can be updated by using the specific keys. The value can be updated along with key **Dict[key] = value**. The update() method is also used to update an existing value.

**Note: If the key-value already present in the dictionary, the value gets updated. Otherwise, the new keys added in the dictionary.**

**Code**

```
# Creating an empty Dictionary
Dict = {}
print("Empty Dictionary: ")
print(Dict)

# Adding elements to dictionary one at a time
Dict[0] = 'Peter'
Dict[2] = 'Joseph'
Dict[3] = 'Ricky'
print("\nDictionary after adding 3 elements: ")
print(Dict)

# Adding set of values
# with a single Key
# The Emp_ages doesn't exist to dictionary
Dict['Emp_ages'] = 20, 33, 24
print("\nDictionary after adding 3 elements: ")
print(Dict)

# Updating existing Key's Value
Dict[3] = 'JavaTpoint'
print("\nUpdated key value: ")
print(Dict)
```



Output

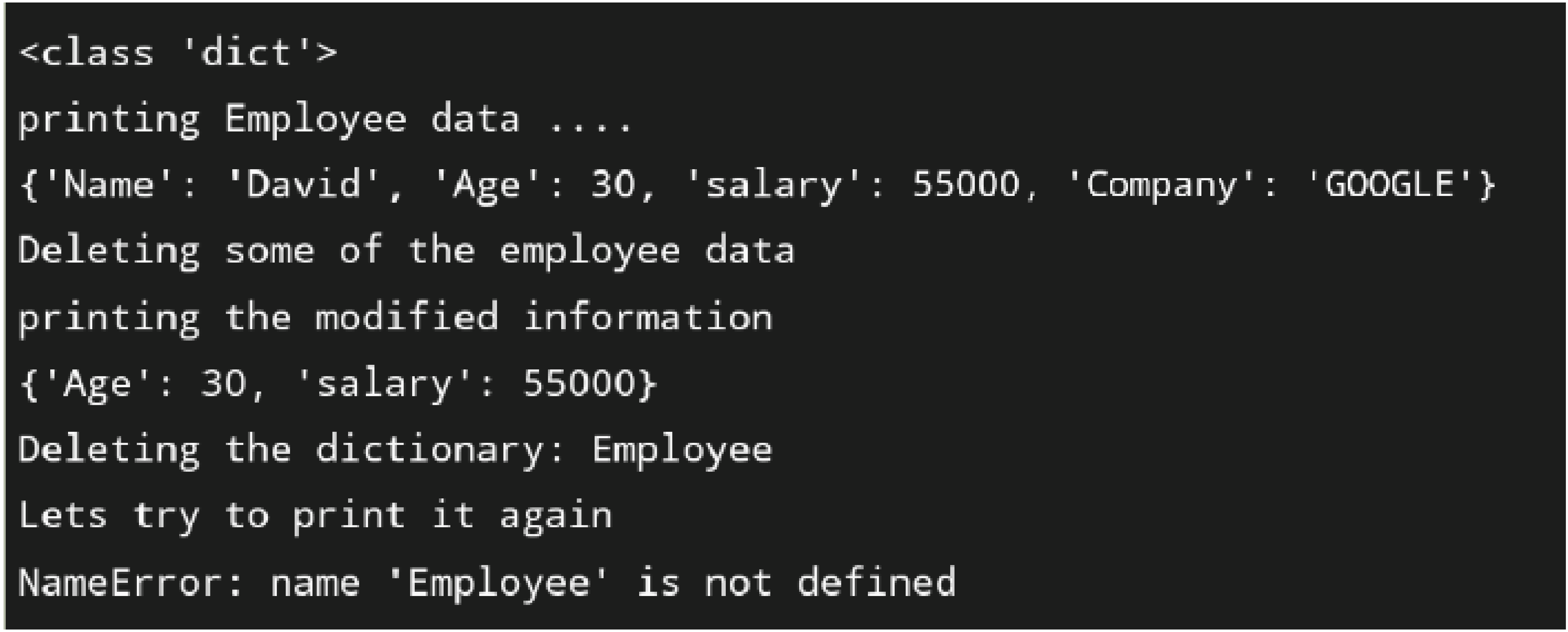
```
Empty Dictionary:
{}

Dictionary after adding 3 elements:
{0: 'Peter', 2: 'Joseph', 3: 'Ricky'}

Dictionary after adding 3 elements:
{0: 'Peter', 2: 'Joseph', 3: 'Ricky', 'Emp_ages': (20, 33, 24)}

Updated key value:
{0: 'Peter', 2: 'Joseph', 3: 'JavaTpoint', 'Emp_ages': (20, 33, 24)}
```

## Deleting Elements using del Keyword

The items of the dictionary can be deleted by using the **del** keyword

**Code**

```
Employee = {"Name": "David", "Age": 30, "salary":55000,"Company":"GOOGLE"}
print(type(Employee))
print("printing Employee data .... ")
print(Employee)
print("Deleting some of the employee data")
del Employee["Name"]
del Employee["Company"]
print("printing the modified information ")
print(Employee)
print("Deleting the dictionary: Employee");
del Employee
print("Lets try to print it again ");
print(Employee)
```


```
<class 'dict'>
printing Employee data ....
{'Name': 'David', 'Age': 30, 'salary': 55000, 'Company': 'GOOGLE'}
Deleting some of the employee data
printing the modified information
{'Age': 30, 'salary': 55000}
Deleting the dictionary: Employee
Lets try to print it again
NameError: name 'Employee' is not defined
```

## Deleting Elements using pop() Method

The **pop()** method accepts the key as an argument and remove the associated value. Consider the following example.

**Code**

```
# Creating a Dictionary
Dict = {1: 'JavaTpoint', 2: 'Learning', 3: 'Website'}
# Deleting a key
# using pop() method
pop_key = Dict.pop(2)
print(Dict)
```

```
{1: 'JavaTpoint', 3: 'Website'}
```

Python also provides a built-in methods popitem() and clear() method for remove elements from the dictionary. The popitem() removes the arbitrary element from a dictionary, whereas the clear() method removes all elements to the whole dictionary.

## Iterating Dictionary

A dictionary can be iterated using for loop as given below.

### Example 1

**Code**

```
# for loop to print all the keys of a dictionary
Employee = {"Name": "John", "Age": 29, "salary":25000,"Company":"GOOGLE"}
for x in Employee:
    print(x)
```

**Output**

```
Name
Age
salary
Company
```

**Code**

```
print the values of the dictionary by using values() method.
 Employee = {"Name": "John", "Age": 29, "salary":25000,"Company":"GOOGLE"}
 for x in Employee.values():
    print(x)
```

**Output**

```
John
29
25000
GOOGLE
```