# PRACTICAL III

**DATE: 01/02/24**

## Tuples and Dictionary in python

**AIM:** To learn and execute Tuples and Dictionary in python.

## INTRODUCTION:

Python Tuple is a collection of objects separated by commas. In some ways, a tuple is similar to a Python list in terms of indexing, nested objects, and repetition but the main difference between both is Python tuple is immutable, unlike the Python list which is mutable.

### Creating Python Tuples

There are various ways by which you can create a tuple in Python. They are as follows:

- Using round brackets
- With one item
- Tuple Constructor

### Create Tuples using Round Brackets ()

```
var = ("Geeks", "for", "Geeks")

print(var)
```

### Create a Tuple With One Item

```
values : tuple[int | str, ...] = (1,2,4,"Geek")

print(values)
```

### Tuple Constructor in Python

```
tuple_constructor = tuple(("dsa", "developement", "deep learning"))

print(tuple_constructor)
```

### What is Immutable in Tuples?

Tuples in Python are similar to Python lists but not entirely. Tuples are immutable and ordered and allow duplicate values. Some Characteristics of Tuples in Python.

- We can find items in a tuple since finding any item does not make changes in the tuple.
- One cannot add items to a tuple once it is created.

- Tuples cannot be appended or extended.
- We cannot remove items from a tuple once it is created.

### Accessing Values in Python Tuples

Tuples in Python provide two ways by which we can access the elements of a tuple.

- Using a positive index
- Using a negative index

### Python Access Tuple using a Positive Index

```
var = ("Geeks", "for", "Geeks")

print("Value in Var[0] = ", var[0])

print("Value in Var[1] = ", var[1])

print("Value in Var[2] = ", var[2])
```

### Access Tuple using Negative Index

```
var = (1, 2, 3)

print("Value in Var[-1] = ", var[-1])

print("Value in Var[-2] = ", var[-2])

print("Value in Var[-3] = ", var[-3])
```

### Different Operations Related to Tuples

- Concatenation
- Nesting
- Repetition
- Slicing
- Deleting
- Finding the length
- Multiple Data Types with tuples
- Conversion of lists to tuples
- Tuples in a Loop

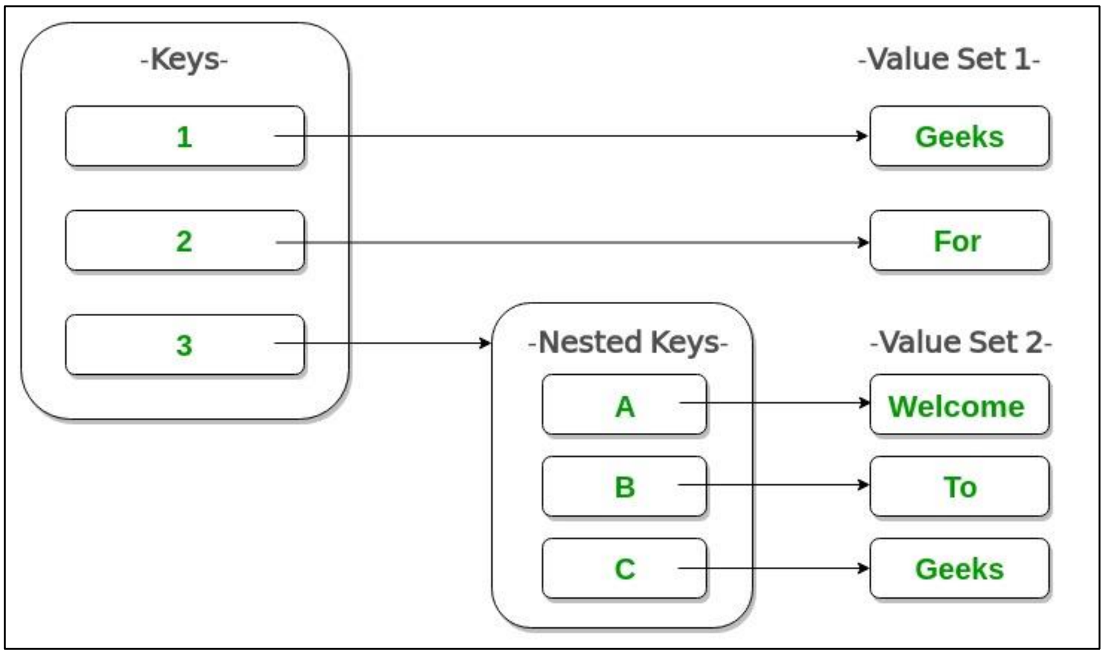
**What is a Python Dictionary?**

Dictionaries in Python is a data structure, used to store values in key:value format. This makes it different from lists, tuples, and arrays as in a dictionary each key has an associated value.

**How to Create a Dictionary**

- In Python, a dictionary can be created by placing a sequence of elements within curly {} braces, separated by a ‘comma’.
- The dictionary holds pairs of values, one being the Key and the other corresponding pair element being its Key:value.
- Values in a dictionary can be of any data type and can be duplicated, whereas keys can’t be repeated and must be immutable.
- Note – Dictionary keys are case sensitive, the same name but different cases of Key will be treated distinctly.

**Nested Dictionaries**



**Example:**

```
Dict = {1: 'Geeks', 2: 'For',
        3: {'A': 'Welcome', 'B': 'To', 'C': 'Geeks'}}
print(Dict)
```

## Adding Elements to a Dictionary

The addition of elements can be done in multiple ways. One value at a time can be added to a Dictionary by defining value along with the key e.g. Dict[Key] = ‘Value’.

Updating an existing value in a Dictionary can be done by using the built-in update() method. Nested key values can also be added to an existing Dictionary.

Note- While adding a value, if the key-value already exists, the value gets updated otherwise a new Key with the value is added to the Dictionary.

## Accessing Elements of a Dictionary

To access the items of a dictionary refer to its key name. Key can be used inside square brackets.

## Deleting Elements using ‘del’ Keyword

The items of the dictionary can be deleted by using the del keyword as given below.

**Example:**

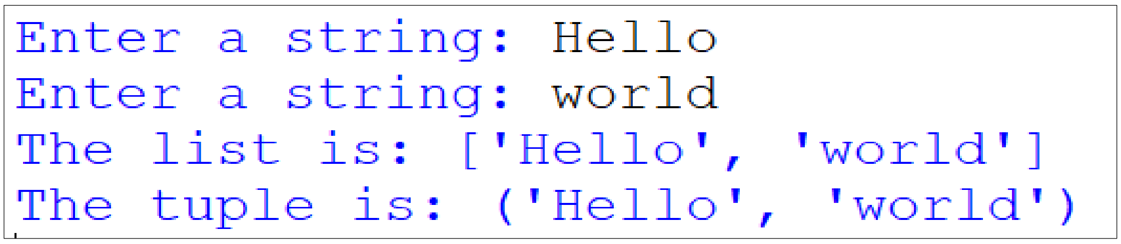
```
Dict = {1: 'Geeks', 'name': 'For', 3: 'Geeks'}

print("Dictionary =")

print(Dict)

del(Dict[1])

print("Data after deletion Dictionary=")

print(Dict)
```

| | |
|---|---|
| dic.clear() | Remove all the elements from the dictionary |
| dict.copy() | Returns a copy of the dictionary |
| dict.get(key, default = “None”) | Returns the value of specified key |
| dict.items() | Returns a list containing a tuple for each key value pair |

| | |
|---|---|
| dict.keys() | Returns a list containing dictionary’s keys |
| dict.update(dict2) | Updates dictionary with specified key-value pairs |
| dict.values() | Returns a list of all the values of dictionary |
| pop() | Remove the element with specified key |
| popItem() | Removes the last inserted key-value pair |
| dict.has_key(key) | returns true if the dictionary contains the specified key. |
| dict.get(key, default = “None”) | used to get the value specified for the passed key. |

**Q1. Write a Python program to create a tuple and display the tuple.**

```
list1=[]
for i in range (0,2):
    a=(input("Enter a string: "))
    list1.append(a)
print("The tuple is:" ,tuple(list1))
```

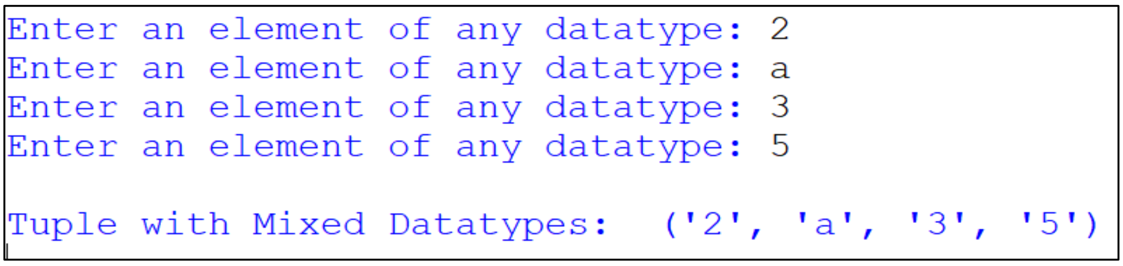


```
Enter a string: Hello
Enter a string: world
The list is: ['Hello', 'world']
The tuple is: ('Hello', 'world')
```

**Figure 1: Output of Q1**

**Q2. Write a Python program to create a tuple with different data types.**

```
list1=[]
for i in range (0,4):
    a=(input("Enter an element of any datatype: "))
    list1.append(a)
print("\nTuple with Mixed Datatypes: " ,tuple(list1))
```

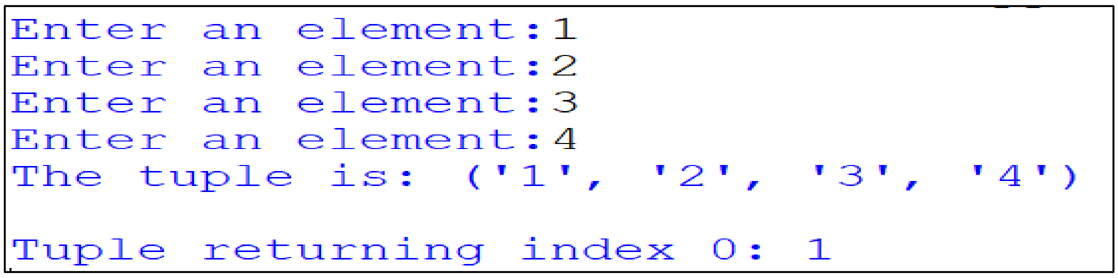


```
Enter an element of any datatype: 2
Enter an element of any datatype: a
Enter an element of any datatype: 3
Enter an element of any datatype: 5

Tuple with Mixed Datatypes:  ('2', 'a', '3', '5')
```

**Figure 2: Output of Q2**

**Q3. Write a Python program to create a tuple with numbers and print one item.**

```
list1=[]
for i in range (0,4):
    a=(input("Enter an element:"))
    list1.append(a)
tup=tuple(list1)
print("The tuple is:" ,tup)
print("\nTuple returning index 0:",tup[0])
```

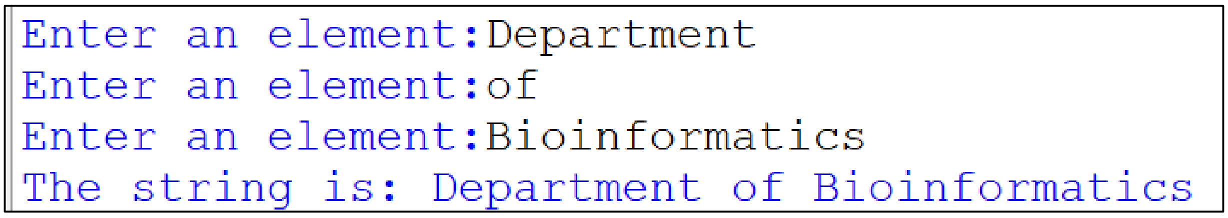


```
Enter an element:1
Enter an element:2
Enter an element:3
Enter an element:4
The tuple is: ('1', '2', '3', '4')

Tuple returning index 0: 1
```

**Figure 3: Output of Q3**

**Q4. Write a Python program to convert a tuple to a string.**

```
list1=[]
for i in range (0,3):
    a=(input("Enter an element:"))
    list1.append(a)
tup=tuple(list1)
string1= str(' '.join(tup))
print("The string is:" ,string1)
```

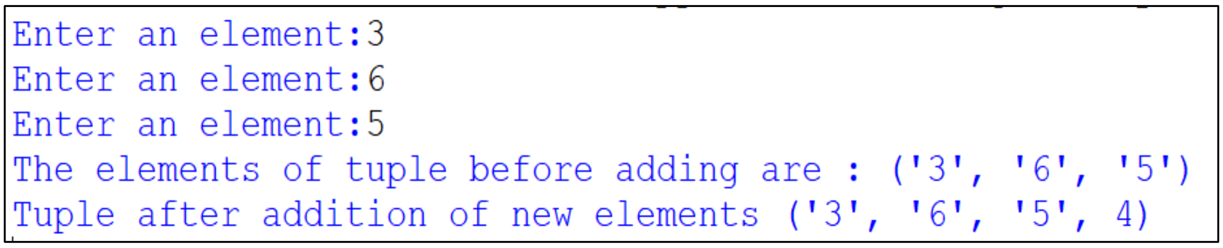


```
Enter an element:Department
Enter an element:of
Enter an element:Bioinformatics
The string is: Department of Bioinformatics
```

**Figure 4: Output of Q4**

**Q5. Write a Python program to add an item in a tuple.**

```
list1=[]
for i in range (0,3):
    a=(input("Enter an element:"))
    list1.append(a)
tup=tuple(list1)
print("The elements of tuple before adding are :", tup)
tup += (4,)
print("Tuple after addition of new elements", tup)
```

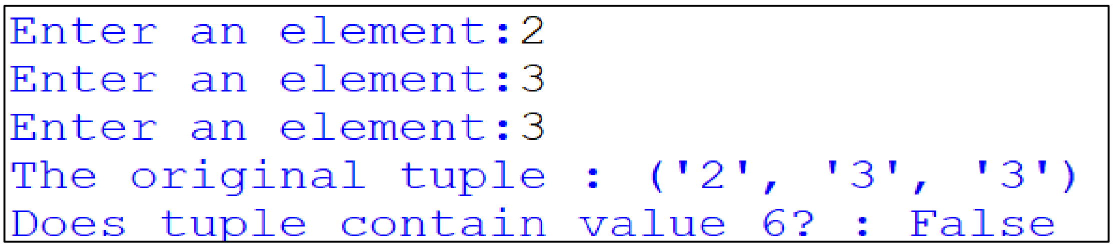


```
Enter an element:3
Enter an element:6
Enter an element:5
The elements of tuple before adding are : ('3', '6', '5')
Tuple after addition of new elements ('3', '6', '5', 4)
```

**Figure 5: Output of Q5**

**Q6. Write a Python program to check whether an element exists within a tuple.**

```
list1=[]
for i in range (0,3):
    a=(input("Enter an element:"))
    list1.append(a)
tup=tuple(list1)
print("The original tuple : " + str(tup))
N = 0
res = N in tup
print("Does tuple contain value 6? : " + str(res))
```

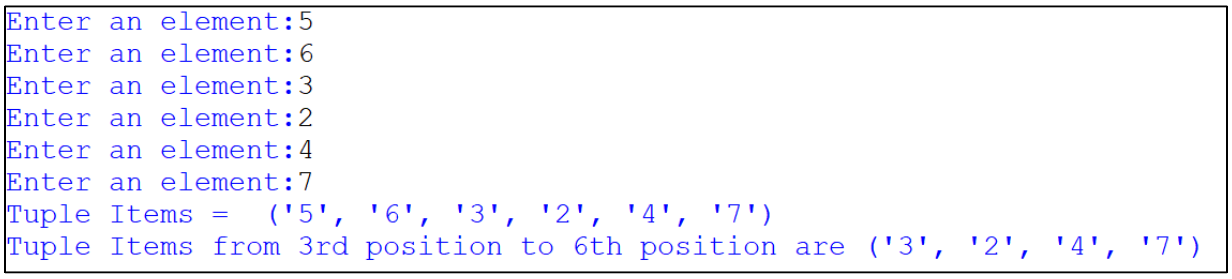


```
Enter an element:2
Enter an element:3
Enter an element:3
The original tuple : ('2', '3', '3')
Does tuple contain value 6? : False
```

**Figure 6: Output of Q6**

**Q7. Write a Python program to slice a tuple.**

```
list1=[]
for i in range (0,6):
    a=(input("Enter an element:"))
    list1.append(a)
tup=tuple(list1)
print("Tuple Items = ", tup)
slice1 = tup[2:6]
print("Tuple Items from 3rd position to 6th position are" ,slice1)
```



```
Enter an element:5
Enter an element:6
Enter an element:3
Enter an element:2
Enter an element:4
Enter an element:7
Tuple Items =  ('5', '6', '3', '2', '4', '7')
Tuple Items from 3rd position to 6th position are ('3', '2', '4', '7')
```

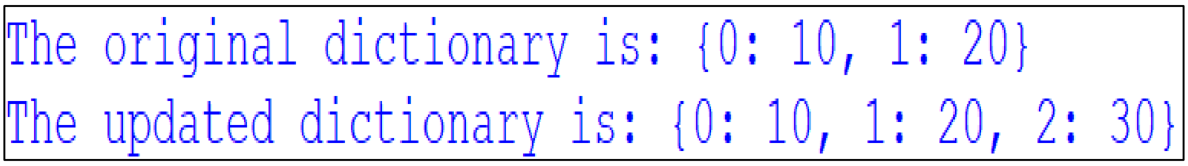
**Figure 7: Output of Q7**

**Q8. Write a Python script to add a key to a dictionary.**

**Sample Dictionary : {0: 10, 1: 20}**

**Expected Result : {0: 10, 1: 20, 2: 30}**

```
dict= {0: 10, 1: 20}
print("The original dictionary is:" ,dict)
dict.update({2: 30})
print("The updated dictionary is:" ,dict)
```



```
The original dictionary is: {0: 10, 1: 20}
The updated dictionary is: {0: 10, 1: 20, 2: 30}
```

**Figure 8: Output of Q8**

**Q9. Write a Python script to concatenate the following dictionaries to create a new one.**

**Sample Dictionary :**

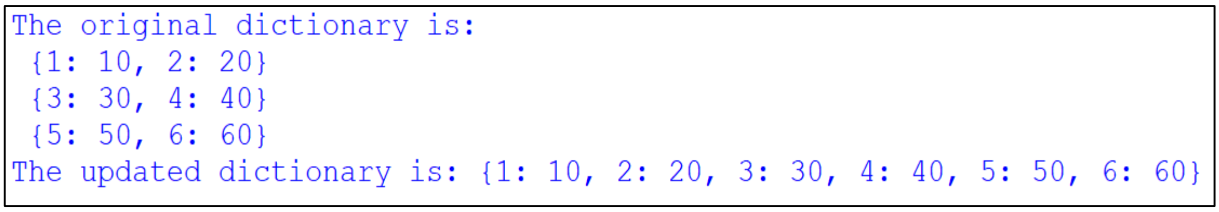
**dic1={1:10, 2:20}**

**dic2={3:30, 4:40}**

**dic3={5:50,6:60}**

**Expected Result : {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}**

```
dic1={1:10, 2:20}
dic2={3:30, 4:40}
dic3={5:50,6:60}
print("The original dictionary is:\n" ,dic1,"\n", dic2,"\n", dic3)
dic1.update(dic2)
dic1.update(dic3)
print("The updated dictionary is:" ,dic1)
```

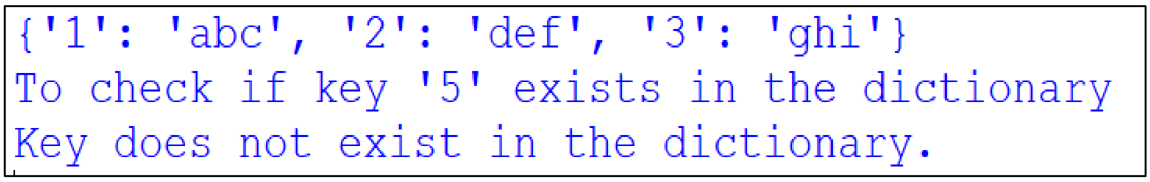


```
The original dictionary is:
 {1: 10, 2: 20}
 {3: 30, 4: 40}
 {5: 50, 6: 60}
The updated dictionary is: {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
```

**Figure 9: Output of Q9**

**Q10. Write a Python script to check whether a given key already exists in a dictionary.**

```
my_dict = {'1': 'abc', '2': 'def', '3': 'ghi'}
print(my_dict)
print("To check if key '5' exists in the dictionary")
if '5' in my_dict:
    print("Key exists in the dictionary.")
else:
    print("Key does not exist in the dictionary.")
```



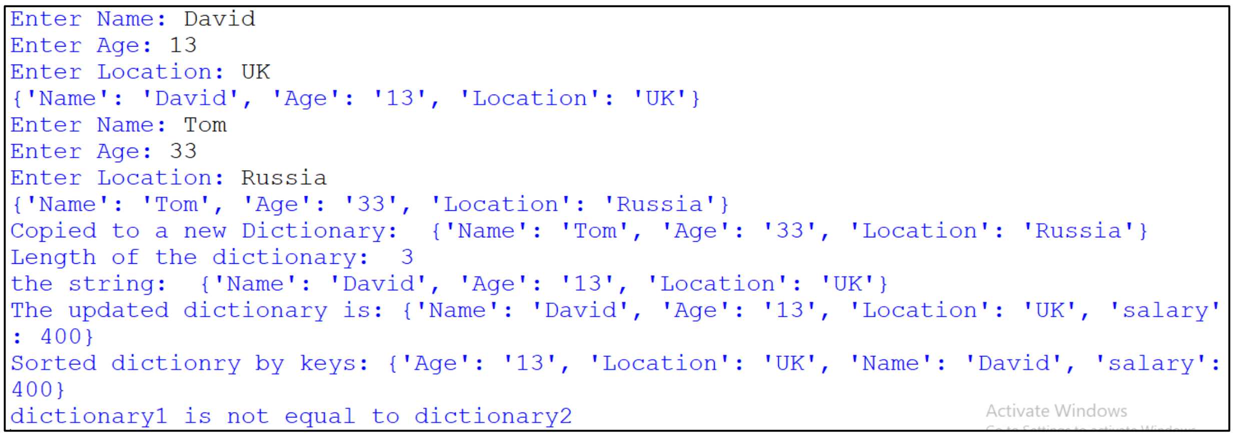
```
{'1': 'abc', '2': 'def', '3': 'ghi'}
To check if key '5' exists in the dictionary
Key does not exist in the dictionary.
```

**Figure 10: Output of Q10**

**Q11. Write a python script to perform all in-built methods in the dictionary.**

```
#first dictionary
dictionary = {}
keys = ['Name', 'Age', 'Location']
for key in keys:
    value = input(f"Enter {key}: ")
    dictionary[key] = value
print (dictionary)
#copy function
dictionary2 = dictionary.copy()
print("Copied to a new Dictionary: ", dictionary2)
#length function
print("Length of the dictionary: ",len(dictionary))
#string function
print("the string: ", str(dictionary))
#update function
dictionary.update({"salary":400})
```

```
print("The updated dictionary is:" ,dictionary)
#sort function
myKeys = list(dictionary.keys())
myKeys.sort()
sorted_dict = {i: dictionary[i] for i in myKeys}
print("Sorted dictionary by keys:" ,sorted_dict)
#compare function
if dictionary == dictionary2:
   print ("dictionary1 is equal to dictionary2")
else:
   print ("dictionary1 is not equal to dictionary2")
```



```
Enter Name: David
Enter Age: 13
Enter Location: UK
{'Name': 'David', 'Age': '13', 'Location': 'UK'}
Enter Name: Tom
Enter Age: 33
Enter Location: Russia
{'Name': 'Tom', 'Age': '33', 'Location': 'Russia'}
Copied to a new Dictionary:  {'Name': 'Tom', 'Age': '33', 'Location': 'Russia'}
Length of the dictionary:  3
the string:  {'Name': 'David', 'Age': '13', 'Location': 'UK'}
The updated dictionary is: {'Name': 'David', 'Age': '13', 'Location': 'UK', 'salary'
: 400}
Sorted dictionry by keys: {'Age': '13', 'Location': 'UK', 'Name': 'David', 'salary':
400}
dictionary1 is not equal to dictionary2
```

**Figure 11: Output of Q11**