Chapter 5: Tuple

Prepared by: Hanan Hardan

Python Tuple

- Tuples are used to store multiple items in a single variable
- A tuple is a collection which is ordered and unchangeable.
- They are written within round brackets.

Example:

```
mytuple = ('C++', 'Python', 'C#')
print (mytuple)
print (type(mytuple))
```

```
>>>
('C++', 'Python', 'C#')
<class 'tuple'>
```

* W3Schools

Tuple Characteristics

- Ordered:
  - It means that the items in a tuple has a defined order and cannot be changed.
- Unchangeable:
  - We cannot change, add or remove items in a created tuple.
- Indexed:
  - Each item in a tuple has an index [0], [1], …etc.
- Allow Duplicates:
  - Tuples can have several items with the same value, but of course difference indecies.

* W3Schools

Example 1

```
mytuple = ('C++', 'Python', 'C#', 'C++', 'Oracle')
print (mytuple)
print (mytuple[0])
print (mytuple[3])
```

```
>>>
('C++', 'Python', 'C#', 'C++', 'Oracle')
C++
C++
>>>
```

Tuple Length

- len function is used to get number of items in a tuple

Example:

```
mytuple = ('C++', 'Python', 'C#', 'C++', 'Oracle')
print (mytuple)
print (len(mytuple))
```

```
>>>
('C++', 'Python', 'C#', 'C++', 'Oracle')
5
>>>
```

Create a Tuple with one item

- To create a tuple with one item you have to add a comma after it, otherwise it won't be considered a tuple.

- **Example:**

```
mytuple = ('C++',)
print (mytuple)
print (type(mytuple))

NotATuple = ('C++')
print (NotATuple)
print (type (NotATuple))
```

```
>>>
('C++',)
<class 'tuple'>
C++
<class 'str'>
>>>
```

Tuple Items – Data types

- A tuple can contain items of all data types. It can include items of the same data type, or of different data type.

Example:

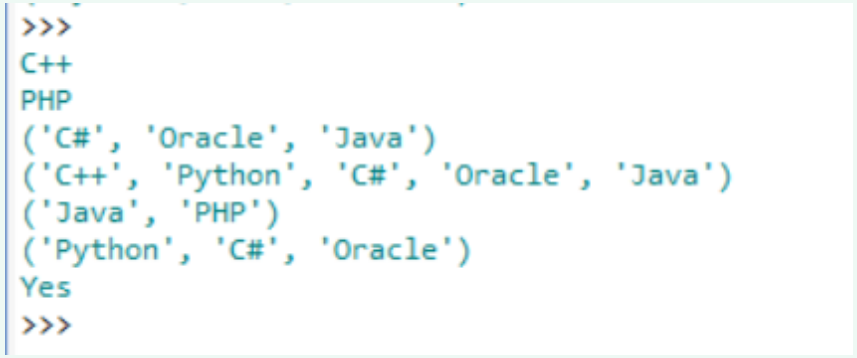
```
tuple1 = ('C++', 'Python', 'C#')
tuple2 = (1,2,3,4)
tuple3 = (True, False, True)
tuple4 = ('Banana', True, 30, 5.6, 'apple')
print (tuple1)
print (tuple2)
print (tuple3)
print (tuple4)
print (type (tuple4[0]))
print (type (tuple4[1]))
print (type (tuple4[2]))
print (type (tuple4[3]))
```

```
>>>
('C++', 'Python', 'C#')
(1, 2, 3, 4)
(True, False, True)
('Banana', True, 30, 5.6, 'apple')
<class 'str'>
<class 'bool'>
<class 'int'>
<class 'float'>
>>>
```

Access Tuple Items

Example:

```
mytuple = ('C++', 'Python', 'C#', 'Oracle', 'Java','PHP')
print (mytuple[0])
print (mytuple[-1])
print (mytuple[2:5])
print (mytuple[:5])
print (mytuple[4:])
print (mytuple[-5:-2])
if 'Python' in mytuple:
    print ('Yes')
```



Packing and unpacking tuples

- Packing: Creating a tuple by adding items in one variable

  fruits = ('apple', 'banana', 'strawberry')
- Unpacking: extracting the items from a tuple into variables

Example:

```
fruits = ('apple', 'banana', 'strawberry')
print (fruits)
(green, yellow, red) = fruits
print (green)
print (yellow)
print (red)
```

```
>>>
('apple', 'banana', 'strawberry')
apple
banana
strawberry
>>>
```

Loop Tuples

- You can loop tuples as described earlier in strings and arrays in two ways:

Example:

```
fruits = ('apple', 'banana', 'strawberry')
for i in fruits:
    print (i)
for j in range(len(fruits)):
    print (fruits[j])
```

```
>>>
apple
banana
strawberry
apple
banana
strawberry
>>>
```

Join Tuples

- **Add:**

```
fruits = ('apple', 'banana', 'strawberry')
Qty = (10, 6, 8)
Result = fruits + Qty
print (Result)
```

```
>>>
('apple', 'banana', 'strawberry', 10, 6, 8)
>>>
```

- **Multiplty:**

```
fruits = ('apple', 'banana', 'strawberry')
Result = fruits * 3
print (Result)
```

```
>>>
('apple', 'banana', 'strawberry', 'apple', 'banana', 'strawberry', 'apple', 'banana', 'strawberry')
>>>
```

Tuple Methods

- **count(itemvalue):** specifies number of times an item value appears in a tuple.

```
fruits = ('apple', 'banana', 'strawberry', 'apple')
print (fruits.count('apple'))
```

```
>>>
2
>>>
```

- **index(itemvalue):** returns the position of a certain item value in a tuple

```
fruits = ('apple', 'banana', 'strawberry', 'apple')
print (fruits.index('apple'))
print (fruits.index('strawberry'))
```

```
>>>
0
2
>>>
```