Practical Learning 3 - 2
Tuples

Tuples

- A tuple in Python is similar to a list. The difference between the two is that we cannot change the elements of a tuple once it is assigned whereas, in a list, elements can be changed.
- A tuple is a collection which is ordered and unchangeable.
- Tuples are written with round brackets.

1. Create a new python file named LE3Tuples_LastnameFirstname.

Creating a tuple

A tuple is created by placing all the items (elements) inside parentheses (), separated by commas. The parentheses are optional; however, it is a good practice to use them.

A tuple can have any number of items and they may be of different types (integer, float, list, string, etc.).

2. Type the following and run the following code:

```
# Empty tuple
my_tuple = ()
print(my_tuple)  # Output: ()

# Tuple having integers
my_tuple = (1, 2, 3)
print(my_tuple)  # Output: (1, 2, 3)

# tuple with mixed datatypes
my_tuple = (1, "Hello", 3.4)
print(my_tuple)  # Output: (1, "Hello", 3.4)

# nested tuple
my_tuple = ("mouse", [8, 4, 6], (1, 2, 3))

# Output: ("mouse", [8, 4, 6], (1, 2, 3))
print(my_tuple)
```

A tuple can also be created without using parentheses. This is known as tuple packing.

```
my_tuple = 3, 4.6, "dog"
print(my_tuple)   # Output: 3, 4.6, "dog"

# tuple unpacking is also possible
a, b, c = my_tuple

print(a)      # 3
print(b)      # 4.6
print(c)      # dog
```

Access Tuple Elements

There are various ways in which we can access the elements of a tuple.

Indexing

We can use the index operator [] to access an item in a tuple where the index starts from 0.

So, a tuple having 6 elements will have indices from 0 to 5. Trying to access an element outside of tuple (for example, 6, 7, ...) will raise an **IndexError.**

The index must be an integer; so, we cannot use float or other types. This will result in **TypeError.**

3.

```
my_tuple = ('p','e','r','m','i','t')

print(my_tuple[0])   # 'p'
print(my_tuple[5])   # 't'

# IndexError: list index out of range
# print(my_tuple[6])

# Index must be an integer
# TypeError: list indices must be integers, not float
# my_tuple[2.0]

# nested tuple
n_tuple = ("mouse", [8, 4, 6], (1, 2, 3))

# nested index
print(n_tuple[0][3])       # 's'
print(n_tuple[1][1])       # 4
```

Negative in Indexing
Python allows negative indexing for its sequences.

The index of -1 refers to the last item, -2 to the second last item and so on.

4.
```
my_tuple = ('p','e','r','m','i','t')

# Output: 't'
print(my_tuple[-1])

# Output: 'p'
print(my_tuple[-6])
```

Slicing

We can access a range of items in a tuple by using the slicing operator - colon ":".

5.
```
my_tuple = ('p','y','t','h','o','n','-','2','L')

# elements 2nd to 4th
# Output: ('y', 'h', 'o')
print(my_tuple[1:4])

# elements beginning to 2nd
# Output: ('p', 'y')
print(my_tuple[:-7])

# elements 8th to end
# Output: ('2', 'L')
print(my_tuple[7:])

# elements beginning to end
# Output: ('p', 'y', 't', 'h', 'o', 'n', '-', '2', 'L')
print(my_tuple[:])
```

Changing a Tuple

Unlike lists, tuples are immutable.

This means that elements of a tuple cannot be changed once it has been assigned. But, if the element is itself a mutable datatype like list, its nested items can be changed.

We can also assign a tuple to different values (reassignment).

6.

```
my_tuple = (4, 2, 3, [6, 5])


# TypeError: 'tuple' object does not support item assignment
# my_tuple[1] = 9

# However, item of mutable element can be changed
my_tuple[3][0] = 9    # Output: (4, 2, 3, [9, 5])
print(my_tuple)

# Tuples can be reassigned
my_tuple = ('p','y','t','h','o','n','-','2','L')

# Output: ('p', 'y', 't', 'h', 'o', 'n', '-', '2', 'L')
print(my_tuple)
```

We can use + operator to combine two tuples. This is also called concatenation.

We can also repeat the elements in a tuple for a given number of times using the * operator.

Both + and * operations result in a new tuple.

7.

```
# Concatenation
# Output: (1, 2, 3, 4, 5, 6)
print((1, 2, 3) + (4, 5, 6))

# Repeat
# Output: ('MCL', 'MCL', 'MCL')
print(("MCL",) * 3)
```

Deleting a Tuple

As discussed above, we cannot change the elements in a tuple. That also means we cannot delete or remove items from a tuple.

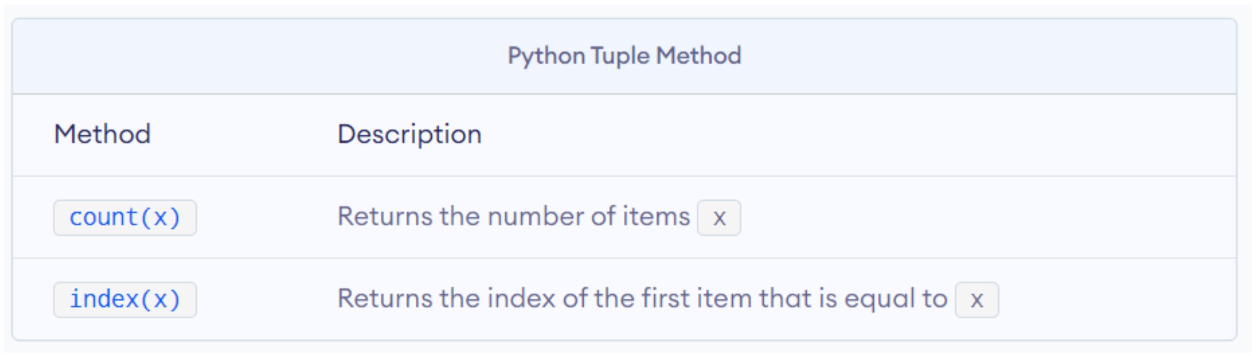
But deleting a tuple entirely is possible using the keyword del.

8.

```
my_tuple = ('p','y','t','h','o','n','-','2','L')

# can't delete items
# TypeError: 'tuple' object doesn't support item deletion
# del my_tuple[3]

# Can delete an entire tuple
del my_tuple

# NameError: name 'my_tuple' is not defined
print(my_tuple)
```

Tuple Methods

Methods that add items or remove items are not available with tuple. Only the following two methods are available.



Python Tuple Method

Method	Description
`count(x)`	Returns the number of items `x`
`index(x)`	Returns the index of the first item that is equal to `x`

Some examples of Python tuple methods:

9. Comment the last code. (The code that generates an error)

```
my_tuple = ('a','p','p','l','e',)

print(my_tuple.count('p'))  # Output: 2
print(my_tuple.index('l'))  # Output: 3
```

10. Upload your work.