Tuple

- A tuple is an ordered, immutable collection of items in Python.
- A tuple can hold elements of different data types.
- A tuple supports indexing and slicing.
- A tuple allows duplicates.
- Once a tuple is created, its elements can't be changed, added, or removed.
- A tuple only supports the `count` and `index` functions.

Tuple Creation

In [ ]:
```
t = ()  # Empty tuple
t1 = tuple()  # Another way to create an empty tuple
```

In [ ]:
```
type(t1)  # Check the type of t1, should return <class 'tuple'>
```

Out[ ]:
```
tuple
```

In [ ]:
```
t2 = (10, 20, 30)  # Tuple with integer elements
t2  # Display the tuple
```

Out[ ]:
```
(10, 20, 30)
```

In [ ]:
```
t3 = ('one', 'two', 'three', 'four')  # Tuple with string elements
t3  # Display the tuple
```

Out[ ]:
```
('one', 'two', 'three', 'four')
```

In [ ]:
```
t4 = (1, 2.7, True, 'three', 1+2j)  # Tuple holding different data types
t4  # Display the tuple
```

Out[ ]:
```
(1, 2.7, True, 'three', (1+2j))
```

In [ ]:
```
t5 = (1, (2, 3), 4, 56)  # Nested tuple
t5  # Display the tuple
```

Out[ ]:
```
(1, (2, 3), 4, 56)
```

In [ ]:
```
t6 = (99, 0, [1, 23, 4], 88)  # List inside the tuple
t6  # Display the tuple
```

Out[ ]:
```
(99, 0, [1, 23, 4], 88)
```

In [ ]:
```
len(t6)  # Get the length of the tuple t6
```

Out[ ]: 4

Indexing and Nested Indexing

- In nested indexing, we can access elements within nested iterables.

In [ ]:
```
t3  # Display the tuple
```

Out[ ]: ('one', 'two', 'three', 'four')

In [ ]:
```
t3[0]  # Access the first element in t3
```

Out[ ]: 'one'

In [ ]:
```
t4  # Display the tuple
```

Out[ ]: (1, 2.7, True, 'three', (1+2j))

In [ ]:
```
t4[0]  # Access the first element in t4
```

Out[ ]: 1

In [ ]:
```
t6  # Display the tuple
```

Out[ ]: (99, 0, [1, 23, 4], 88)

In [ ]:
```
t6[2][0]  # Access the first element of the list inside the tuple
```

Out[ ]: 1

In [ ]:
```
t5  # Display the tuple
```

Out[ ]: (1, (2, 3), 4, 56)

In [ ]:
```
t5[1][0]  # Access the first element of the nested tuple inside t5
```

Out[ ]: 2

In [ ]:
```
t4  # Display the tuple
```

Out[ ]: (1, 2.7, True, 'three', (1+2j))

In [ ]:
```
t4[3][0]  # Attempt to access the first character of the fourth element (a string)
```

Out[ ]: 't'

Slicing

- Slicing allows you to retrieve a portion of a tuple.

In [ ]:
```
t7 = ('one', 'two', 'three', 'four', 'five', 'six') # Tuple with multiple string el
```

In [ ]:
```
t7  # Display the tuple
```

Out[ ]:
```
('one', 'two', 'three', 'four', 'five', 'six')
```

In [ ]:
```
t7[1:3]  # Slice the tuple to get elements from index 1 to 2
```

Out[ ]:
```
('two', 'three')
```

In [ ]:
```
t7[:4]  # Slice the tuple to get elements from the start up to index 3
```

Out[ ]:
```
('one', 'two', 'three', 'four')
```

In [ ]:
```
t7[::-1]  # Reverse the tuple using slicing
```

Out[ ]:
```
('six', 'five', 'four', 'three', 'two', 'one')
```

Remove & Changes

- Tuple elements cannot be removed or changed since tuples are immutable.

In [ ]:
```
t2  # Display the tuple
```

Out[ ]:
```
(10, 20, 30)
```

In [ ]:
```
del t2[1]  # Attempt to delete an item from a tuple will raise an error
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[157], line 1
----> 1 del t2[1]  # Attempt to delete an item from a tuple will raise an error

TypeError: 'tuple' object doesn't support item deletion
```

In [ ]:
```
del t2[0]  # Attempt to delete another item from a tuple will raise an error
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[159], line 1
----> 1 del t2[0]  # Attempt to delete another item from a tuple will raise an error

TypeError: 'tuple' object doesn't support item deletion
```

In [ ]:
```
del t2  # Delete the entire tuple
t2  # Attempt to access t2 after deletion will raise an error
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[161], line 2
      1 del t2  # Delete the entire tuple
----> 2 t2  # Attempt to access t2 after deletion will raise an error

NameError: name 't2' is not defined
```

In [ ]:
```
t4  # Display the tuple
```

Out[ ]:
```
(1, 2.7, True, 'three', (1+2j))
```

In [ ]:
```
t4[0] = 100  # Attempt to change an item in a tuple will raise an error
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[165], line 1
----> 1 t4[0] = 100  # Attempt to change an item in a tuple will raise an error

TypeError: 'tuple' object does not support item assignment
```

In [ ]:
```
t7  # Display the tuple
```

Out[ ]:
```
('one', 'two', 'three', 'four', 'five', 'six')
```

In [ ]:
```
t7.clear()  # Attempt to clear a tuple will raise an error as clear() is not suppor
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[169], line 1
----> 1 t7.clear()  # Attempt to clear a tuple will raise an error as clear() is not
supported

AttributeError: 'tuple' object has no attribute 'clear'
```

In [ ]:
```
t8 = t2.copy()  # Attempt to copy a tuple will raise an error as copy() is not supp
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[171], line 1
----> 1 t8 = t2.copy()  # Attempt to copy a tuple will raise an error as copy() is n
ot supported

NameError: name 't2' is not defined
```

Loop Through a Tuple

- You can loop through the elements of a tuple.

In [ ]:
```
for i in t6:
    print(i)  # Print each element in t6
```

```
99
0
[1, 23, 4]
88
```

In [ ]:
```
for i in enumerate(t5):
    print(i)  # Print each element with its index from t5
```

```
(0, 1)
(1, (2, 3))
(2, 4)
(3, 56)
```

Index()

- The index() method is supported in tuples.
- With the index() function, we can get the index number of a specified value.

In [ ]:
```
t9 = (100, 200, 300, 400)  # Tuple with integer elements
```

In [ ]:
```
t9.index(400)  # Get the index of the value 400 in t9
```

Out[ ]:
```
3
```

In [ ]:
```
t9.index(500)  # Attempting to get the index of a non-existent value will raise an
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[121], line 1
----> 1 t9.index(500)  # Attempting to get the index of a non-existent value will ra
ise an error

ValueError: tuple.index(x): x not in tuple
```

Count

- The count() function is supported in tuples.
- count() checks the number of occurrences of a specified value.

In [ ]:
```
t9
```

Out[ ]:
```
(100, 200, 300, 400)
```

In [ ]:
```
t9.count(400)  # Count how many times 400 appears in t9
```

Out[ ]:
```
1
```

In [ ]:
```
t9.count(500)  # Attempt to count a value that doesn't exist will return 0
```

Out[ ]:
```
0
```

Tuple Membership

- The in operator is used to check for membership in a tuple.
- It checks whether a specified value is present in the iterable or not.

In [ ]:
```
t9  # Display the tuple
```

Out[ ]:
```
(100, 200, 300, 400)
```

In [ ]:
```
400 in t9  # Check if 400 is in t9
```

Out[ ]:
```
True
```

In [ ]:
```
500 in t9  # Check if 500 is in t9
```

Out[ ]:
```
False
```

Sort

- Sorting is possible in tuples, but unlike lists, tuples don't have a built-in sort function.
- To sort a tuple, you need to pass it to the sorted() function, which returns a list of sorted values.

In [ ]:
```
t10 = (1, 44, 56, 100, 0)  # Tuple with integer elements
```

In [ ]:
```
sorted(t10)  # Sort the tuple, returns a sorted list
```

Out[ ]:
```
[0, 1, 44, 56, 100]
```

In [ ]:
```
sorted(t10, reverse=True)  # Sort the tuple in reverse order, returns a sorted list
```

Out[ ]:
```
[100, 56, 44, 1, 0]
```