Tuples

A tuple is another sequence data type in Python. It is similar to a list in terms of indexing and slicing but has a few key differences. Tuples are useful for storing collections of related values that should not be changed after they are created. They can be used to represent things like coordinates, dates, and other types of fixed data.

Creating a Tuple

Tuples are created using parentheses () and comma , separates the values in a tuple. Example:

```
In [1]: my_tuple = (1, 2, 3, 'four', 'five')
        print(my_tuple) # Output: (1, 2, 3, 'four', 'five')
```

```
(1, 2, 3, 'four', 'five')
```

Accessing Tuple Values

Tuple values are accessed using indexing, just like lists. Example:

```
In [2]: print(my_tuple[0]) # Output: 1
        print(my_tuple[-1]) # Output: 'five'
```

```
1
five
```

Tuple Slicing

Tuple slicing is done just like list slicing. Example:

```
In [3]: # Slicing a tuple
        print(my_tuple[1:4]) # Output: (2, 3, 'four')
```

```
(2, 3, 'four')
```

Tuple Packing and Unpacking

Tuple packing is when we take values and pack them into a tuple. Tuple unpacking is the opposite where we take a tuple and unpack it into individual values.

```
In [4]: # Tuple packing
        my_packed_tuple = 1, 2, 3
        print(my_packed_tuple) # Output: (1, 2, 3)

        # Tuple unpacking
```

```
a, b, c = my_packed_tuple
print(a, b, c) # Output: 1 2 3
```

```
(1, 2, 3)
1 2 3
```

Tuple Methods

Tuples have fewer built-in methods compared to lists. Here are some common tuple methods:

count(): Returns the number of times a specified value appears in the tuple. Example:

In [5]:
```
# Counting the number of times a value appears in the tuple
my_tuple = (1, 2, 2, 3, 2)
count = my_tuple.count(2)
print(count) # Output: 3
```

```
3
```

len(): Returns the number of elements in a tuple. Example:

In [6]:
```
# Finding the length of a tuple
my_tuple = (1, 2, 3)
length = len(my_tuple)
print(length) # Output: 3
```

```
3
```

sorted(): Returns a new sorted list from the items in an iterable. Example:

In [7]:
```
# Sorting a tuple
my_tuple = (3, 1, 4, 1, 5, 9, 2, 6, 5, 3)
sorted_tuple = sorted(my_tuple)
print(sorted_tuple) # Output: [1, 1, 2, 3, 3, 4, 5, 5, 6, 9]
print(type(sorted_tuple))
```

```
[1, 1, 2, 3, 3, 4, 5, 5, 6, 9]
<class 'list'>
```

min() and max(): Return the minimum and maximum values in a tuple, respectively. Example:

In [8]:
```
# Finding the minimum and maximum values in a tuple
my_tuple = (3, 1, 4, 1, 5, 9, 2, 6, 5, 3)

min_value = min(my_tuple)
max_value = max(my_tuple)

print(min_value) # Output: 1
print(max_value) # Output: 9
```

```
1
9
```

index(): Returns the index of the first occurrence of a specified value in the tuple. Example:

In [9]:
```
# Finding the index of a value in the tuple
my_tuple = (1, 2, 3, 4, 5)

index = my_tuple.index(3)

print(index) # Output: 2
```

```
2
```

any() and **all():** Return True if at least one or all of the elements in a tuple, respectively, evaluate to True. Otherwise, they return False.

In [10]:
```
# Checking if any or all elements in a tuple are True
my_tuple = (True, False, True)

any_true = any(my_tuple)
all_true = all(my_tuple)

print(any_true) # Output: True
print(all_true) # Output: False
```

```
True
False
```

Note : Immutable nature Tuples are immutable, which means once a tuple is created, we cannot change its values. This is different from lists which can be modified after creation. For Example:

In [11]:
```
# Tuples are immutable
my_tuple = (1, 2, 3)
my_tuple[1] = 4 # Raises a TypeError
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[11], line 3
      1 # Tuples are immutable
      2 my_tuple = (1, 2, 3)
----> 3 my_tuple[1] = 4 # Raises a TypeError

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

Conclusion

In summary, tuples are a type of data structure in Python that are similar to lists, but with a few key differences. Tuples are immutable, meaning that once they are created, their values cannot be changed. They are created using parentheses instead of square brackets, and they can contain any type of data.