Tuples

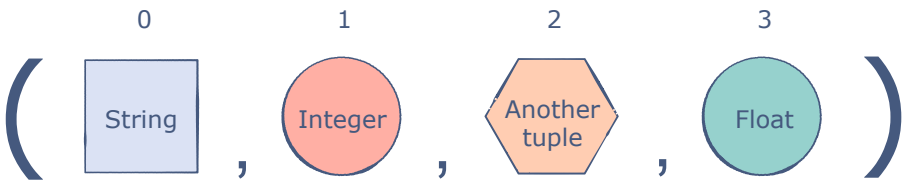
This lesson highlights the key features of the tuple data structure in Python.

We'll cover the following

Structure

A tuple is very similar to a list, except for the fact that its contents cannot be changed. In other words, a tuple is **immutable**. However, it can contain mutable elements like a list. These elements can be altered.

The contents of a tuple are enclosed in parentheses, `()`. They are also ordered, and hence, follow the linear index notation.



Length: 4

Creating a Tuple

Tuples can be created similar to lists. All the indexing and slicing operations apply to it as well:

```
car = ("Ford", "Raptor", 2019, "Red")
print(car)
```

```
# Length
print(len(car))


# Indexing
print(car[1])

# Slicing
print(car[2:])
```

Merging Tuples

Tuples can be merged using the `+` operator:

```
hero1 = ("Batman", "Bruce Wayne")
hero2 = ("Wonder Woman", "Diana Prince")
awesome_team = hero1 + hero2
print(awesome_team)
```

Nested Tuples

In the previous coding example, instead of merging the two tuples, we could create a new tuple with these two tuples as its members:

```
hero1 = ("Batman", "Bruce Wayne")
hero2 = ("Wonder Woman", "Diana Prince")
awesome_team = (hero1, hero2)
print(awesome_team)
```

Search

We can check whether an element exists in a tuple by using the `in` operator:

```
cities = ("London", "Paris", "Los Angeles", "Tokyo")
print("Moscow" in cities)
```

The `index()` function can give us the index of a particular value:

```
cities = ("London", "Paris", "Los Angeles", "Tokyo")
print(cities.index("Tokyo"))
```

Immutability

Since tuples are immutable, we can’t add or delete elements from them. Furthermore, it isn’t possible to append another tuple to an existing tuple.

And we’re done with tuples! Next up, we have the famous **dictionary**.