TUPLES

A tuple in Python is an immutable sequence of elements, enclosed in parentheses `()`. Tuples are similar to lists, but unlike lists, they cannot be modified once created. Tuples are commonly used to store related pieces of data together.

Here's an example of creating a tuple in Python:

```
my_tuple = (1, 2, 3, 'a', 'b', 'c')
```

In the above example, `my_tuple` is a tuple that contains integers and strings. Tuples can store elements of different data types.

You can access individual elements of a tuple using indexing, just like lists. For example:

```
print(my_tuple[0])  # Output: 1
print(my_tuple[3])  # Output: 'a'
```

Tuples also support slicing, which allows you to extract a portion of the tuple. For example:

```
print(my_tuple[2:5])  # Output: (3, 'a', 'b')
```

Since tuples are immutable, you cannot modify their elements or add/remove elements once created. However, you can concatenate tuples or create new tuples based on existing ones.

```
tuple1 = (1, 2, 3)
tuple2 = ('a', 'b', 'c')
concatenated_tuple = tuple1 + tuple2
print(concatenated_tuple)  # Output: (1, 2, 3, 'a', 'b', 'c')
```

Tuples are often used when you want to ensure that the data remains unchanged throughout your program or when you want to return multiple values from a function.

```
# 1. Create two tuples t1 with elements (1, 2, 3) and t2 with
elements (4, 5, 6). Concatenate them into a single tuple t3 and
print t3.

t1 = (1, 2, 3)
t2 = (4, 5, 6)
t3 = t1 + t2
print("Concatenated tuple:", t3)
```

```
Concatenated tuple: (1, 2, 3, 4, 5, 6)
```

```
# 2. Create a tuple t with elements (10, 20, 30, 40, 50). Print the
first and last element.

t = (10, 20, 30, 40, 50)
print("First element:", t[0])
print("Last element:", t[-1])
```

```
First element: 10
Last element: 50
```

```python
# 3. Create a tuple t with elements (10, 20, 30, 40, 50). Print the
length of the tuple.

t = (10, 20, 30, 40, 50)
print("Length of the tuple:", len(t))
```

```
Length of the tuple: 5
```

```python
# 4. Create a tuple t = (5, 1, 8, 3, 9, 2) and find the maximum and
minimum elements.

t = (5, 1, 8, 3, 9, 2)
print("Maximum element:", max(t))
print("Minimum element:", min(t))
```

```
Maximum element: 9
Minimum element: 1
```

```python
# 5. Create a tuple t with elements (1, 2, 3, 4, 5) and print the
element at index 3.

def element_at_index(tpl, index):
    return tpl[index]
# Example usage
tpl = (1, 2, 3, 4, 5)
index = 3
print(f"Element at index {index}->", element_at_index(tpl, index))
```

```
Element at index 3-> 4
```