TOPSTechnologies

Accessing Tuple

Presented for :

TOPs Technologies

Presented by :

Sumit B Yadav

Que 1

Accessing Tuples elements using positive and negative

1) Positive Indexing :

Starts from 0 for the first element and increases by 1 for subsequent elements.
Use the index in square brackets [] to access the element.

```
my_tuple = (10,20,30,40,50)
print(my_tuple[0])
output : 10
print(my_tuple[2])
output : 30
```

2) Negitive Indexing:

Starts from -1 for the last element and decreases by 1 as you move left.
Useful to access elements from the end of the tuple.

```
my_tuple = (10,20,30,40,50)
print(my_tuple[-1])
output : 50
print(my_tuple{-3])
output : 30
```

Accessing Tuple Elements

You can use either positive or negative indexing to access any element in the tuple.

example -

```
my_tuple = ('a', 'b', 'c', 'd', 'e')

# Positive indexing
print(my_tuple[1])  # Output: 'b'

# Negative indexing
print(my_tuple[-2])  # Output: 'd'
```

Que. 2

tuple_name[start:stop:step]

- start: The index to start slicing (inclusive).
- stop: The index to stop slicing (exclusive).
- step: The interval between elements (optional).

Basic Slicing -

```
my_tuple = (10, 20, 30, 40, 50)

# Slice from index 1 to 3 (exclusive)
print(my_tuple[1:3])  # Output: (20, 30)

# Slice from index 0 to 4 (exclusive)
print(my_tuple[0:4])  # Output: (10, 20, 30, 40)
```

2. Omitting Start or Stop

- If start is omitted, it defaults to the beginning (0).
- If stop is omitted, it defaults to the end of the tuple.

python

Example -

```
my_tuple = (10, 20, 30, 40, 50)

# Slice from the beginning to index 3 (exclusive)
print(my_tuple[:3])  # Output: (10, 20, 30)

# Slice from index 2 to the end
print(my_tuple[2:])  # Output: (30, 40, 50)
```

3. Negative Indexing

You can use negative indexes for slicing as well.

python

example -

```
my_tuple = (10, 20, 30, 40, 50)

# Slice using negative indexes
print(my_tuple[-4:-1])  # Output: (20, 30, 40)

# Slice from index -3 to the end
print(my_tuple[-3:])  # Output: (30, 40, 50)
```

4. Step Parameter

The step specifies the stride of the slicing. If omitted, it defaults to 1.

example -

```
my_tuple = (10, 20, 30, 40, 50)

# Slice with a step of 2
print(my_tuple[0:5:2])  # Output: (10, 30, 50)

# Reverse the tuple using a negative step
print(my_tuple[::-1])  # Output: (50, 40, 30, 20, 10)
```

5. Combining Positive and Negative Indexing

example-

```
my_tuple = (10, 20, 30, 40, 50)

# Slice from index 1 to the second last element
print(my_tuple[1:-1])  # Output: (20, 30, 40)
```

3. Membership Testing

You can check if an element exists in a tuple using the in keyword, which returns True if the element is present and False otherwise.

Example:

```
tuple1 = (1, 2, 3, 4, 5)

# Check if an element exists in the tuple
print(3 in tuple1)  # Output: True
print(6 in tuple1)  # Output: False
```