Key Points About Tuples in Python

- **Definition**: Tuples are immutable data structures used to store multiple items, often of different data types, in a single variable.
- **Declaration**: Tuples are declared using parentheses `()` or just commas (best practice is parentheses):
  - Example: `my_tuple = (1, "string", 4.5, True)`
- **Accessing Elements**: Use zero-based indexing to access tuple elements:
  - Example: `my_tuple[1]` returns `"string"` .
- **Checking Type**: Use `type()` to confirm an object is a tuple:
  - Example: `type(my_tuple)` returns `<class 'tuple'>` .
- **Built-in Methods**:
  - `count(value)` : Returns the number of occurrences of a value.
    - Example: `my_tuple.count("string")` returns `1` .
  - `index(value)` : Returns the index of the first occurrence of a value.
    - Example: `my_tuple.index(4.5)` returns `2` .
- **Iteration**: Use a `for` loop to iterate through tuple elements:
  - Example:

    ```
    for x in my_tuple:
        print(x)
    ```

- **Immutability**: Tuples are immutable, meaning their elements cannot be changed after declaration.
  - Attempting to assign a new value to a tuple element results in a `TypeError` .
  - Example:

    ```
    my_tuple[0] = 5  # Raises TypeError
    ```

- **Use Case**: Tuples are preferred when the data structure should remain constant throughout the program.