**Python Collections: list, tuple, set, dict**

Python has four main collection data types:  
✅ **list** → Ordered, Mutable, Allows Duplicates  
✅ **tuple** → Ordered, Immutable, Allows Duplicates  
✅ **set** → Unordered, Mutable, No Duplicates  
✅ **dict** → Key-Value Pairs, Ordered (Python 3.7+), Mutable

**1. List (list)**

A **list** is an **ordered, mutable** collection that allows **duplicate** values.

**✅ Characteristics**

* **Ordered** → Elements have a fixed order.
* **Mutable** → Can modify elements.
* **Duplicates Allowed** → Same values can appear multiple times.

**📌 Example**

fruits = ["apple", "banana", "cherry", "apple"]

print(fruits) # Output: ['apple', 'banana', 'cherry', 'apple']

**🛠️ Common List Operations**

fruits.append("mango") # Adds 'mango' to the end

fruits.insert(1, "grape") # Inserts 'grape' at index 1

fruits.remove("banana") # Removes 'banana'

fruits.pop() # Removes last element

fruits.sort() # Sorts the list

print(len(fruits)) # Gets the length

**2. Tuple (tuple)**

A **tuple** is an **ordered, immutable** collection that allows **duplicates**.

**✅ Characteristics**

* **Ordered** → Elements remain in a fixed order.
* **Immutable** → Cannot be modified after creation.
* **Duplicates Allowed** → Same values can appear multiple times.

**📌 Example**

coordinates = (10, 20, 30)

print(coordinates) # Output: (10, 20, 30)

**🛠️ Tuple Operations**

print(coordinates[1]) # Accessing elements (Output: 20)

print(len(coordinates)) # Getting tuple length

**Tuples are faster than lists** because they are immutable.

✅ **Use tuples when data should not change** (e.g., geographic coordinates).

**3. Set (set)**

A **set** is an **unordered, mutable** collection that contains **unique elements**.

**✅ Characteristics**

* **Unordered** → No fixed order.
* **Mutable** → Can add or remove elements.
* **No Duplicates** → Automatically removes duplicates.

**📌 Example**

unique\_numbers = {1, 2, 3, 3, 4}

print(unique\_numbers) # Output: {1, 2, 3, 4} (No duplicates!)

**🛠️ Set Operations**

unique\_numbers.add(5) # Adds an element

unique\_numbers.remove(3) # Removes element (Error if not found)

unique\_numbers.discard(10) # Removes element (No error if not found)

print(len(unique\_numbers)) # Gets the length

**Set Mathematical Operations**

A = {1, 2, 3}

B = {3, 4, 5}

print(A | B) # Union: {1, 2, 3, 4, 5}

print(A & B) # Intersection: {3}

print(A - B) # Difference: {1, 2}

print(A ^ B) # Symmetric Difference: {1, 2, 4, 5}

✅ **Use sets when you need unique values and fast membership checks**.

**4. Dictionary (dict)**

A **dictionary** is an **ordered (Python 3.7+), mutable** collection of **key-value pairs**.

**✅ Characteristics**

* **Key-Value Pairs** → Each item has a key and a value.
* **Ordered** (Python 3.7+) → Maintains insertion order.
* **Mutable** → Can modify elements.
* **Keys Must Be Unique** → No duplicate keys allowed.

**📌 Example**

student = {"name": "Alice", "age": 25, "city": "New York"}

print(student["name"]) # Output: Alice

**🛠️ Dictionary Operations**

student["age"] = 26 # Updating a value

student["course"] = "Python" # Adding a new key-value pair

student.pop("city") # Removing an item

print(student.keys()) # Getting all keys

print(student.values()) # Getting all values

✅ **Use dictionaries when you need key-value storage for quick lookups.**

**Comparison Table**

| **Feature** | **list** | **tuple** | **set** | **dict** |
| --- | --- | --- | --- | --- |
| **Ordered?** | ✅ Yes | ✅ Yes | ❌ No | ✅ Yes (3.7+) |
| **Mutable?** | ✅ Yes | ❌ No | ✅ Yes | ✅ Yes |
| **Duplicates?** | ✅ Yes | ✅ Yes | ❌ No | ❌ No (Keys) |
| **Indexing?** | ✅ Yes | ✅ Yes | ❌ No | ✅ Keys |
| **Usage** | General storage | Fixed data | Unique values | Key-value storage |

**When to Use Each?**

* **list** → When you need an **ordered, modifiable collection**.
* **tuple** → When you need an **ordered, immutable collection** (e.g., coordinates, config values).
* **set** → When you need **unique, unordered elements** and fast lookups.
* **dict** → When you need to store **key-value pairs** for fast retrieval.