**Week 8 – Notes from the Lecture**

**Syntax for Creating a Tuple or Dictionary**

* Tuple: A tuple is created by placing values inside parentheses `()` and separating them with commas. Tuples are immutable, meaning their elements cannot be changed after creation.

Syntax:

my\_tuple = (value1, value2, value3)

Example:

my\_tuple = (1, "apple", 3.14)

Tuples can also be created without parentheses, using just commas:

my\_tuple = 1, "apple", 3.14

* Dictionary: A dictionary is created using curly braces `{}` and consists of key-value pairs. Each key is separated from its value by a colon `:`, and the pairs are separated by commas. Dictionaries are mutable.

Syntax:

my\_dict = {key1: value1, key2: value2, key3: value3}

Example:

my\_dict = {"name": "Alice", "age": 25, "city": "New York"}

**What Are Keys and Values in the Context of Dictionaries?**

* Keys: Keys in a dictionary are unique identifiers used to access the associated values. Each key must be unique and immutable (e.g., strings, numbers, or tuples).
* Values: Values are the data or information associated with each key. They can be of any data type (strings, numbers, lists, other dictionaries, etc.), and they do not need to be unique.

Example:

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

In this dictionary:

- Keys: `"name"`, `"age"`, `"city"`

- Values: `"Alice"`, `25`, `"New York"`

You access values by referencing their keys:

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

**Guidelines for Choosing Between Data Structures (Tuple, List, Dictionary)**

1. Tuple:

* Use when:
* The order of elements matters.
* You need immutable data that should not be changed after assignment (e.g., coordinates, fixed collections of constants).
* You want a simple, lightweight container with a fixed size.

- Example Use Case: Storing coordinates `(x, y)`, RGB color values `(255, 0, 0)`.

2. List:

* Use when:
* You need a mutable sequence of elements that can be modified, added to, or removed from.
* The order of elements matters.
* You need to store a collection of items that may change over time (e.g., shopping list, queue of tasks).

- Example Use Case: Storing a list of user inputs, inventory items, or numbers for further calculations.

3. Dictionary:

* Use when:
* You need to store key-value pairs where each key is associated with a specific value.
* You want to quickly access data based on a unique identifier (key) rather than index positions.
* The collection needs to support fast lookups and doesn’t rely on the order of items.

- Example Use Case: Storing user profiles (with keys like `"name"`, `"age"`, `"email"`), or mapping product names to prices.

**Summary:**

* Tuple: Use for fixed, ordered, immutable collections.
* List: Use for mutable, ordered collections where elements may change.
* Dictionary: Use for unordered collections of key-value pairs with unique keys and fast lookups.