**Understanding Comprehensions and Sets in Python**

Python provides **comprehensions** as a concise way to create collections like **lists, dictionaries, and sets**. Let's explore comprehensions and sets in detail.

**1. List Comprehensions**

**Definition:**

List comprehensions provide a **shorter and more readable** way to create lists.

**Example Usage:**

# Using a loop

numbers = []

for i in range(5):

numbers.append(i \* 2)

# Using list comprehension (same result)

numbers = [i \* 2 for i in range(5)]

print(numbers) # Output: [0, 2, 4, 6, 8]

* More **efficient and readable** than a for loop.
* Can include **conditions**.

**Example with Condition:**

even\_numbers = [x for x in range(10) if x % 2 == 0]

print(even\_numbers) # Output: [0, 2, 4, 6, 8]

**2. Dictionary Comprehensions**

**Definition:**

Creates dictionaries in a compact way using key-value pairs.

**Example Usage:**

squares = {x: x\*\*2 for x in range(5)}

print(squares) # Output: {0: 0, 1: 1, 2: 4, 3: 9, 4: 16}

* Useful for **transforming existing dictionaries**.

**Example - Swapping Keys and Values in a Dictionary:**

original = {'a': 1, 'b': 2, 'c': 3}

swapped = {v: k for k, v in original.items()}

print(swapped) # Output: {1: 'a', 2: 'b', 3: 'c'}

**3. Set Comprehensions**

**Definition:**

Creates a **set** using comprehension, removing duplicates automatically.

**Example Usage:**

numbers = {x % 3 for x in range(10)}

print(numbers) # Output: {0, 1, 2}

* **Removes duplicates** automatically.
* **Faster and more concise** than using a loop.

**4. Nested Comprehensions**

**Definition:**

Comprehensions can be nested for multi-dimensional data structures.

**Example Usage:**

matrix = [[j for j in range(3)] for i in range(3)]

print(matrix)

# Output: [[0, 1, 2], [0, 1, 2], [0, 1, 2]]

**5. Sets in Python**

**Definition:**

A **set** is an **unordered collection of unique elements**.

**Example Usage:**

# Creating a set

my\_set = {1, 2, 3, 3, 2}

print(my\_set) # Output: {1, 2, 3}

# Adding elements

my\_set.add(4)

# Removing elements

my\_set.remove(2)

# Checking membership

print(3 in my\_set) # Output: True

**Set Operations:**

A = {1, 2, 3}

B = {3, 4, 5}

# Union

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

# Intersection

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

# Difference

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

**Summary Table**

| **Concept** | **Definition** | **Example** |
| --- | --- | --- |
| **List Comprehension** | Creates a list using a compact syntax | [x\*\*2 for x in range(5)] |
| **Dictionary Comprehension** | Creates a dictionary with key-value pairs | {x: x\*\*2 for x in range(5)} |
| **Set Comprehension** | Creates a set with unique values | {x % 3 for x in range(10)} |
| **Sets** | Unordered collection of unique items | {1, 2, 3} |
| **Set Operations** | Union, Intersection, Difference | `A |