**Module – 3 (Collections, functions and Modules)**

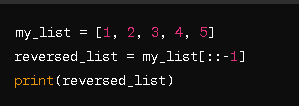
**Ketan Pillai**

**1) What is List? How will you reverse a list?**

Ans - A list is like a collection of items, kind of like a shopping list. It can hold anything you want, like numbers, words, or even other lists.

To reverse a list, you just flip it around so that the last item becomes the first, and the first becomes the last. Imagine you're stacking books and then you decide to take them off one by one, but in the opposite order. That's reversing a list.

In code, you could do it like this in Python:

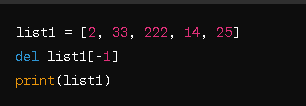


This `[:: -1]` part tells Python to step through the list backwards, reversing it. So, if `my\_list` is `[1, 2, 3, 4, 5]`, `reversed\_list` will be `[5, 4, 3, 2, 1]`.

**2) How will you remove last object from a list?  
Suppose list1 is [2, 33, 222, 14, and 25], what is list1 [-1]?**

**Ans -** To remove the last object from a list, you simply delete it, like tossing out the last item on your shopping list before you leave the store.

In Python, you can do it like this:



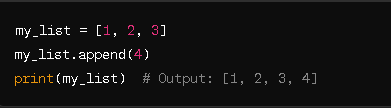
This will give you `[2, 33, 222, 14]` because it removes the last item, which is `25`.

As for `list1[-1]`, it just gives you the last item in the list, so `list1[-1]` would give you `25`. It's like asking, "What's the last thing on the list?"

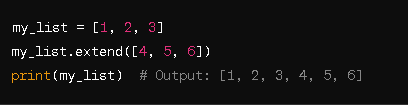
**3) Differentiate between append () and extend () methods?**

Ans - In Python, both `append()` and `extend()` are methods used to add elements to a list, but they work a bit differently:

- `append()`: It adds a single element to the end of the list. It's like putting one more thing at the end of your shopping list.



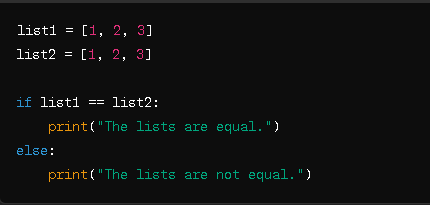
- `extend()`: It adds multiple elements to the end of the list by appending each item from the iterable you provide. It's like combining two lists together.



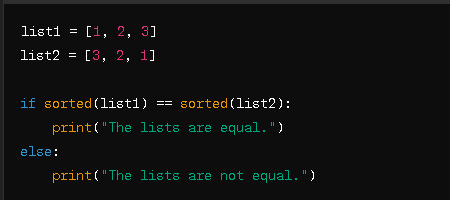
So, `append()` adds one thing at a time, while `extend()` adds a bunch of things all at once.

**5) How will you compare two lists?**

Ans - To compare two lists in Python, you can use the `==` operator. It checks if both lists have the same elements in the same order.

Here's how you can do it:

This will print "The lists are equal." because both `list1` and `list2` have the same elements in the same order.

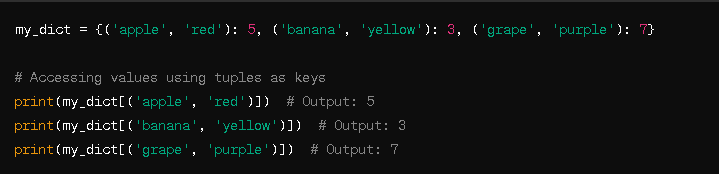
If the lists have the same elements but in a different order, Python will consider them different. If you want to check if they have the same elements regardless of order, you can sort them first:

This will still print "The lists are equal." because after sorting, both lists will have the same elements.

* **How will you create dictionary using tuples in Python?**

Ans - In Python, you can create a dictionary using tuples as keys. Each key-value pair in the dictionary represents a mapping between a tuple (key) and a value.

Here's how you do it:

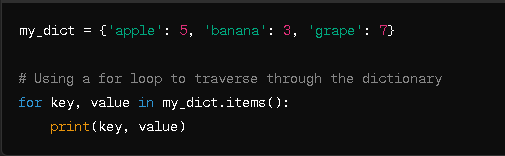


In this example, each tuple `(key)` represents a fruit and its color, and the corresponding value represents the quantity of that fruit. So, when you access the dictionary with a tuple as a key, you get the associated value.

* **How Do You Traverse Through A Dictionary Object In Python?**

**Ans -** Traversing through a dictionary in Python means going through each key-value pair in the dictionary. You can do this using a loop.

Here's how you can traverse through a dictionary:

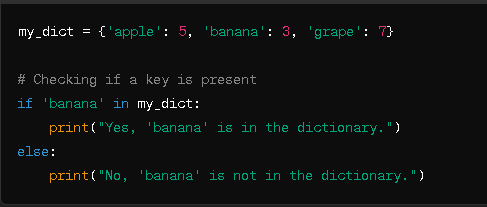


In this loop, `key` represents each key in the dictionary, and `value` represents the corresponding value. By using `.items()` method, Python gives you both the keys and values as pairs to work with in the loop. So, you can print them or perform any operation you want inside the loop.

* **How Do You Check The Presence Of A Key In A Dictionary?**

**Ans –** To check if a key is present in a dictionary in Python, you can use the `in` keyword.

Here's how you do it:



This will print "Yes, 'banana' is in the dictionary." because 'banana' is a key in `my\_dict`. If you want to check for a key that's not present, you'd get the "No" message.