Chapter 5 - Lists

Lists

List Operators

Learning Objectives - List Operators

- **Modify lists with slice (`:`), `in`, `+`, `*`, and `len`**
- **Identify the return values for the above operations**
- **Describe an empty list**

List Concatenation

List Concatenation

You can use the concatenation operator (`+`) to combine two lists together. This is similar to string concatenation from the lesson on data types.

```
list_1 = [1, 2, 3]
list_2 = [4, 5, 6]

print(list_1 + list_2)
```

challenge

What happens if you:

- Change the `print` statement to be `print(list_2 + list_1)`?
- Change the `print` statement to be `print(list_1 + [4])`?
- Change the `print` statement to be `print(list_1 + 4)`?

List Repetition

List Repetition

You can use the repetition operator (`*`) to a list. This is similar to string repetition from the lesson on data types.

```
list_1 = ["Hi!"]

print(list_1 * 4)
```

challenge

What happens if you:

- Change the `print` statement to be `print(list_1 * 100)`?
- Change the `print` statement to be `print(list_1 * 0)`?
- Change the `print` statement to be `print(list_1 * -1)`?

▼ **What does `[]` mean?**

The `[]` is called an empty list. This is a list that has no elements. If you use the `*` operator and 0 or a negative integer on a list, it will produce an empty list.

The In Operator

The In Operator

The `in` operator tells you if a value is present in a list. `in` is a boolean operator, so it will return `True` or `False`.

```
my_list = ["red", "orange", "yellow", "green"]

print("red" in my_list)
```

challenge

What happens if you:

- Change the `print` statement to be `print("black" in my_list)`?
- Change the `print` statement to be `print("Red" in my_list)`?
- Change the `print` statement to be `print(my_list in my_list)`?

List Length

List Length

Python has a function (`len`) to determine the length of a list.

```
list_1 = [12, 66, 52, 97, 28, 41, 7]
list_2 = [68, True, 34, False, 41.897, "apple"]

if len(list_1) > len(list_2):
    print("list_1 is longer than list_2")
elif len(list_1) == len(list_2):
    print("list_1 and list_2 are the same length")
else:
    print("list_2 is longer than list_1")
```

▼ **The Empty List**

We already talked about the empty list, `[]`. It is a list with no elements. You can define an empty list with code if `len(my_list) == 0` is true, then the list is an empty list.

challenge

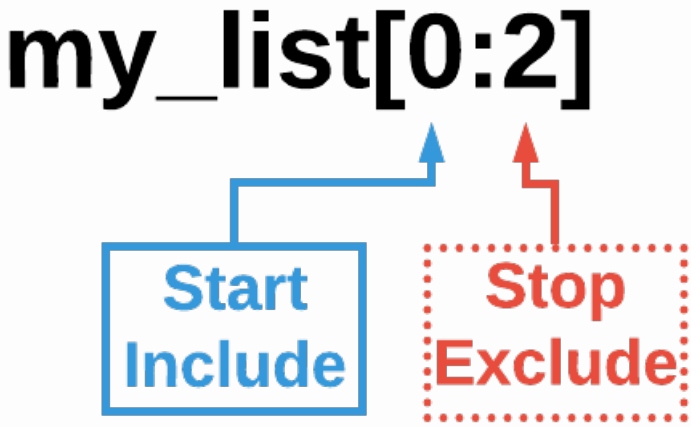
What happens if you:

- Remove one of the elements from `list_1`?
- Add a few elements to `list_2`?
- Change `list_2` to `list_2 = [i for i in range(0, 20)]`?

The Slice Operator

The Slice Operator

The slice operator (`:`) returns a portion of the list. Provide numbers around the slice operator to indicate where you start and stop. The slice operator includes the first number, but does **not include** the second number. The slice operator does not modify the original list. Instead, it returns a portion of the list.



List Slice

```
my_list = ["red", "orange", "yellow", "green"]
my_slice = my_list[0:2]

print(my_slice)
```

challenge

What happens if you:

- Change the slice to be `my_list[1:2]`?
- Change the slice to be `my_list[0:len(my_list)]`?
- Change the slice to be `my_list[1:1]`?
- Change the slice to be `my_list[:2]`?

▼ **Slice Defaults**

If no number is used for the starting point in a slice `my_list[:2]`, Python will default to 0. If no number is used for the stopping point `my_list[2:]`, Python will default to the end of the list. Using no numbers on a slice `my_list[:]`, Python will default to 0 for the start and the end of the list as the stopping point. In short, Python will return the entire list.