1. What exactly is []?

Solution –

This defines a data type “list”. We can store collection of data inside lists.

For Example –

List1 = [“Mango”, “Apple”,2023,True,34.5]

1. In a list of values stored in a variable called spam, how would you assign the value 'hello' as the third value? (Assume [2, 4, 6, 8, 10] are in spam.)

Solution –

Spam[2] = ‘hello’

Let's pretend the spam includes the list ['a', 'b', 'c', 'd'] for the next three queries.

1. What is the value of spam[int(int('3' \* 2) / 11)]?

Solution –

Output: 3

1. What is the value of spam[-1]?

Solution –

Output: ‘d’

1. What is the value of spam[:2]?

Solution –

Values from index 0 to 1 (2 values from the beginning)

Output: [‘a’,’b’]

Let's pretend bacon has the list [3.14, 'cat,' 11, 'cat,' True] for the next three questions.

1. What is the value of bacon.index('cat')?

Solution –

Output: 1

1. How does bacon.append(99) change the look of the list value in bacon?

Solution –

It adds integer value 99 at the end of the list.

Output:

[3.14, 'cat', 11, 'cat’, True,99]

1. How does bacon.remove('cat') change the look of the list in bacon?

Solution –

Output:

[3.14, 'cat', 11, True, 99]

9. What are the list concatenation and list replication operators?

Solution –

**List concatenation – “+”**

This is used to combine the values present in two or more lists.

For Example –

List1 = [1,2,3,4,5]

List2 = [6,7,8,9,10]

List\_concat = List1+List2

Print(List\_concat)

Output:

[1,2,3,4,5,6,7,8,9,10]

**List Replication – “\*”**

This is used to create a new list by repeating the elements of a list to mentioned amount of times.

For Example –

List1 = [1,2,3,4,5]

List\_Replicate = List1 \* 2

Print(List\_Replicate)

Output:

[1,2,3,4,5,1,2,3,4,5]

1. What is difference between the list methods append() and insert()?

Solution –

Append() – append allows us to add a value at the end of the list

Insert() – insert allows us to add a value at desired index position

1. What are the two methods for removing items from a list?

Solution –

Remove() – remove allows us to remove items by value.

Pop() – pop allows us to remove items by index.

1. Describe how list values and string values are identical.

Solution –

Both lists and strings are stored in sequence.

Both can be accessed using their index values/position.

Both can be iterated using loops to access the values.

We can check the length of list and a string using a method len().

For Example –

List1 = [“Hello”,100,True,3.3]

String1= ‘Hello World’

Print(List1[1])

Output: 100

Print(String1[1])

Output: e

1. What's the difference between tuples and lists?

Solution –

Lists = Defined with square brackets “[]”

Tuples = Defined with braces “()”

Lists – Mutable

Tuples - Immutable

1. How do you type a tuple value that only contains the integer 42?

Solution –

Tup1 = (42)

1. How do you get a list value's tuple form? How do you get a tuple value's list form?

Solution –

Tup1 = tuple(list1)

List1 = list(tup1)

1. Variables that "contain" list values are not necessarily lists themselves. Instead, what do they contain?

Solution –

Variables that "contain" list values are not necessarily lists themselves, instead they store a reference to the list in memory.

For Example –

List1 = [1,2,3,4,5]

List2=List1

List1.append(6)

Print(List1)

Print(List2)

Once we append a value into List1 it also updates List2

1. How do you distinguish between copy.copy() and copy.deepcopy()?

Solution –

Copy.copy() – Shallow Copy

A shallow copy doesn't create a copy of nested objects, instead it just copies the reference of nested objects.

For Example –

import copy

old\_list = [[1, 1, 1], [2, 2, 2], [3, 3, 3]]

new\_list = copy.copy(old\_list)

old\_list[1][1] = 'AA'

print("Old list:", old\_list)

print("New list:", new\_list)

Output:

Old\_list = [[1,1,1],[2,AA,2],[3,3,3]]

New\_list = [[1,1,1],[2,AA,2],[3,3,3]]

Copy.deepcopy() – Deep Copy

A deep copy creates independent copy of original object and all its nested objects.

For Example –

import copy

old\_list = [[1, 1, 1], [2, 2, 2], [3, 3, 3]]

new\_list = copy.deepcopy(old\_list)

old\_list[1][1] = 'AA'

print("Old list:", old\_list)

print("New list:", new\_list)

Output:

Old\_list = [[1,1,1],[2,AA,2],[3,3,3]]

New\_list = [[1,1,1],[2,2,2],[3,3,3]]