1. What exactly is []?

Ans:

The symbol “[“ is used to indicate start of a list while “]” is used to indicate end of a list in python.

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.)

Ans:

We can assign value using indexing.

e.g. 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)]?

Ans: The value of spam[3] is “d”.

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

Ans: The value of spam[-1] is “d” -> the last element.

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

Ans: The value of spam[:2] returns list of values of index 0 and 1 i.e. [“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')?

Ans: The value of bacon.index(“cat”) is 1. (The first appearance)

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

Ans: The append assigns value at the end of the list like this : [3.14, “cat”, 11, “cat”, True, 99]

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

Ans: If the remove is called for once the bacon list will look like [3.14,11,’cat’, True,99]

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

Ans:

In Python, the + symbol serves as the list concatenation operator, allowing us to merge multiple lists into one. On the other hand, the \* symbol acts as the list replication operator. It enables us to generate a new list by duplicating the elements of an existing list multiple times.

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

Ans:

The append() method in Python is like adding an item to the end of a list. It changes the list itself by putting the item at the very end. In contrast, the insert() method is used to add an item at a particular position in the list. You can choose where exactly to insert the item by specifying the desired position. The other items in the list are shifted to make room for the new item.

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

Ans: Delete and pop are two methods used for removing items from a list.

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

Ans:

Lists and strings have some similarities. They are both like a long line of items, where you can find each item in order. You can use numbers to find specific items in both lists and strings. For example, if you have a list of fruits, you can use a number to get the fruit at that position. The same goes for strings, where you can use a number to find a specific letter in a word. Both lists and strings can also be combined together. You can put two lists or strings side by side to make a bigger list or string. So, they are similar in how you can organize and combine things, but they also have their own unique features.

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

Ans:

Tuples and lists are both used to store collections of items in Python, but they have some key differences. The main difference is that lists are mutable, which means their elements can be changed, added, or removed, while tuples are immutable, and their elements cannot be modified once they are defined. This means that you can modify a list by adding or removing elements, but with a tuple, you cannot make any changes to the elements once the tuple is created. Another difference is the syntax: lists are enclosed in square brackets [], while tuples are enclosed in parentheses ().

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

Ans:

To create a tuple with the integer value 42, we can use ``` t = (42,)```

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

Ans:

We can use tuple(m\_list) function to convert m\_list to tuple. Similarly we can use list(m\_tuple) to convert tuple to list.

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

Ans:

In Python, variables that hold list values are like signposts pointing to the actual list stored in the computer's memory. They don't directly hold the entire list, but instead remember where the list is located. So, when we use a variable with a list value, it's like following a signpost to find and work with the actual list.

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

Ans:

copy.copy() creates a copy of an object, meaning it creates a new object that references the same nested objects as the original. Changes to the nested objects will affect both the copy and the original. copy.deepcopy() creates a deep copy of an object, meaning it creates a new object with its own independent copies of all nested objects. Changes to the nested objects in the copy will not affect the original.