1. What exactly is []?

Ans:

The notation [] typically represents an empty list in many programming languages, including Python. A list is a data structure that can store multiple items in a specific order. The empty brackets [] denote a list that contains no elements.

2. 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:

To assign the value 'hello' as the third value in the list stored in the variable called spam, you can use the following code:

spam = [2, 4, 6, 8, 10] # Original list

spam[2] = 'hello' # Assign 'hello' to the third value

After executing this code, the spam list will be updated to [2, 4, 'hello', 8, 10], with 'hello' as the third value.

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

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

Ans:

To determine the value of spam[int(int('3' \* 2) / 11)], we'll break down the expression step by step.

1. '3' \* 2 will result in the string '33'.
2. int('33') will convert the string '33' to the integer 33.
3. 33 / 11 will perform integer division and give the result 3 (since both 33 and 11 are integers).
4. int(3) will simply return 3.

Therefore, spam[int(int('3' \* 2) / 11)] is equivalent to spam[3].

Assuming the list spam is ['a', 'b', 'c', 'd'], the value of spam[3] will be the fourth element of the list, which is 'd'. So, the value of spam[int(int('3' \* 2) / 11)] is 'd'.

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

Ans: If the list spam is ['a', 'b', 'c', 'd'], then spam[-1] refers to the last element of the list.

In this case, the value of spam[-1] would be 'd'.

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

Ans: If the spam list includes the values ['a', 'b', 'c', 'd'], the value of spam[:2] would be ['a', 'b'].

In Python, when slicing a list, the syntax spam[:2] means to select elements starting from index 0 up to, but not including, index 2. In this case, it selects the elements at index 0 and index 1, which are 'a' and 'b', respectively.

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

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

Ans: The value of bacon.index('cat') is 1.

In Python, the index() method is used to find the first occurrence of a specified element in a list and returns its index. In this case, the first occurrence of the string 'cat' in the list bacon is at index 1.

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

Ans: If you execute the command bacon.append(99) on the list bacon with the initial values [3.14, 'cat,' 11, 'cat,' True], it will add the value 99 at the end of the list. The updated list would look like this:

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

The append() method in Python adds an element to the end of a list. In this case, it adds the value 99 as the last item in the list bacon.

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

Ans: If you execute the command bacon.remove('cat') on the list bacon with the initial values [3.14, 'cat,' 11, 'cat,' True], it will remove the first occurrence of the value 'cat' from the list. The updated list would look like this:

[3.14, 11, 'cat', True]

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

Ans: In Python, the list concatenation operator is the plus sign (+), and the list replication operator is the asterisk (\*).

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

Ans: The append() and insert() methods are used in Python to add elements to a list, but they differ in how they add elements and where they place them within the list.

append() method:

The append() method is used to add an element to the end of a list. It takes a single argument, which is the element you want to add, and appends it to the end of the list.

insert() method:

The insert() method is used to add an element at a specific index position within a list. It takes two arguments: the index position where you want to insert the element and the element itself.

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

Ans:

There are several methods for removing items from a list, but two commonly used methods are:

Using the remove() method: The remove() method allows you to remove the first occurrence of a specified value from a list. You need to provide the value you want to remove as an argument to the remove() method. If the value is present multiple times in the list, only the first occurrence will be removed.

Using the pop() method: The pop() method removes and returns the element at a specified index from a list. If you don't provide an index, pop() will remove and return the last element of the list. Here's an example:

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

Ans: The similarity between Lists and Strings in Python is that both are sequences. The differences between them are that firstly, Lists are mutable but Strings are immutable. Secondly, elements of a list can be of different types whereas a String only contains characters that are all of String type.

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

Ans: The primary difference between tuples and lists is that tuples are immutable as opposed to lists which are mutable. Therefore, it is possible to change a list but not a tuple. The contents of a tuple cannot change once they have been created in Python due to the immutability of tuples.

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

Ans: To create a tuple with the integer value 42, you can use the following syntax in Python:

my\_tuple = (42,)

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

Ans: To convert a list value into its tuple form, you can use the tuple() function.

To convert a tuple value into its list form, you can use the list() function.

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

Ans: Variables that "contain" list values are not actually storing the entire list itself, but rather a reference or pointer to the list. In many programming languages, including Python, variables that hold list values are actually storing memory addresses that point to the location in memory where the list is stored.

In simpler terms, the variable contains a reference to the list's memory location, rather than the actual list values. This means that if you assign the list to another variable or pass it as a function argument, you're actually passing around the memory address, not making a copy of the entire list.

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

Ans:

copy() create reference to original object. If you change copied object - you change the original object. . deepcopy() creates new object and does real copying of original object to new one. Changing new deepcopied object doesn't affect original object.