1. What exactly is []?

Answer: - [] represents an empty list in Python. It is a data structure that can store multiple values of different types in a single variable.

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.)

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

Answer: - To assign the value 'hello' as the third value in the list stored in the variable spam, you can use the index notation and assign the value directly:

spam[2] = 'hello'

The index starts from 0, so the third value corresponds to index 2.

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

Answer: - The value of spam[int(int('3' * 2) / 11)] can be calculated as follows:

- '3' * 2 results in the string '33'.
- int('33') converts the string '33' to an integer 33.
- 33 / 11 performs integer division and gives the result 3.
- Finally, spam[3] accesses the value at index 3 in the spam list.

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

Answer: - The value of spam[-1] gives the last value in the spam list. In this case, it would be 'd'.

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

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

Answer: - The value of spam[:2] is a sublist that includes all elements from the beginning up to, but not including, index 2. Therefore, it would be ['a', 'b'].

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

Answer: - The value of bacon.index('cat') returns the index of the first occurrence of the element 'cat' in the bacon list. In this case, it would be 1.

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

Answer: - The bacon.append(99) method adds the value 99 at the end of the bacon list, changing the list to [3.14, 'cat', 11, 'cat', True, 99].

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

Answer: - The bacon.remove('cat') method removes the first occurrence of the element 'cat' from the bacon list. After executing this method, the list becomes [3.14, 11, 'cat', True].

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

Answer: - The list concatenation operator is +, which combines two lists to create a new list. The list replication operator is *, which replicates a list a certain number of times to create a new list.

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

Answer: - The append() method is used to add an element to the end of a list, while the insert() method is used to insert an element at a specific index within the list.

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

Answer: - The two methods for removing items from a list are remove() and pop(). The remove() method removes the first occurrence of a specific value, while the pop() method removes an element at a given index and returns its value.

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

Answer: - List values and string values are similar in that they both can be indexed and sliced using square brackets. They are both ordered sequences of elements. However, lists can contain elements of different data types and are mutable (can be changed), while strings are immutable (cannot be changed) and can only contain characters.

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

Answer: - Tuples and lists are both used to store multiple items in a single variable, but they have some differences:

- Lists are mutable, meaning their elements can be changed, added, or removed. Tuples are immutable and cannot be modified once created.
- Lists are represented with square brackets [], while tuples use parentheses ().
- Lists are typically used for collections of similar items, while tuples are used for collections of different types of items.

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

Answer: - To create a tuple value that only contains the integer 42, you can use parentheses:

my_tuple = (42,)

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

Answer: - To convert a list value to its tuple form, you can use the tuple() function. For example:

my_list = [1, 2, 3]

my_tuple = tuple(my_list)

To convert a tuple value to its list form, you can use the list() function. For example:

my_tuple = (1, 2, 3)

my_list = list(my_tuple)

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

contain?

Answer: - Variables that "contain" list values are themselves references to the list object. In other words, they hold the memory address of the list object. Modifying the list through one variable will affect all variables that reference the same list.

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

Answer: - The copy.copy() function creates a shallow copy of an object, including lists. It creates a new object with a new reference, but the elements inside the copied object still refer to the same objects as the original.

On the other hand, the copy.deepcopy() function creates a deep copy of an object, including lists. It creates a completely independent copy, including all the nested objects. Modifying the copied object will not affect the original.