1. What exactly is []?

It represents an empty list.

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.)

spam[2] = “hello”

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)]?

It’s spam[3] which in the above case is ‘d’

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

‘d’

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

[‘a’,’b’]

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')?

1

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

This will append 99 to the end of the list and the list bacon will become [3.14, 'cat', 11, 'cat', True, 99]

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

This will remove the first occurrence of ‘cat’ from the list bacon, and it will become [3.14, 11, 'cat', True]

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

The list concatenation operator in Python is +, which combines two lists into one.

For example, [1, 2] + [3, 4] results in [1, 2, 3, 4].

The list replication operator is *, which replicates a list by a specified number of times.

For example, [1, 2] * 3 results in [1, 2, 1, 2, 1, 2]

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

The append() method adds a single element to the end of a list, while the insert() method inserts a single element at a specified position in the list, shifting existing elements to the right.

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

The two methods for removing items from a list are remove() and pop(). The remove() method removes the first occurrence of a specified value from the list, while the pop() method removes and returns the item at a specified index, or the last item if no index is specified.

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

List values and string values share several characteristics: both are sequences of elements, accessed by index, and can be sliced and concatenated. Additionally, both support iteration and various built-in operations like len() and membership (in).

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

Tuples and lists are both ordered collections of items, but they have some key differences. Lists are mutable, meaning their elements can be changed, added, or removed after the list is created. Tuples, on the other hand, are immutable; once created, their elements cannot be changed. Additionally, lists are typically used for homogeneous data, while tuples are often used for heterogeneous data and for cases where immutability is desired.

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

(42,)

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

To convert a list value to its tuple form, you can use the tuple() function, passing the list as an argument. Conversely, to convert a tuple value to 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?

Variables that "contain" list values are references to the list objects stored in memory. In other words, they contain memory addresses that point to the location of the list in memory.

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

copy.copy() performs a shallow copy of a list, meaning it creates a new list object but does not recursively clone the elements within the list. This means that if the list contains mutable objects (like other lists or dictionaries), changes to those objects within the copied list will affect the original list. copy.deepcopy() performs a deep copy of a list, recursively copying all elements within the list as well. This ensures that changes made to elements in the copied list do not affect the original list, as they operate on separate objects in memory.