1. What exactly is []?

Ans: In Python, [] represents an empty list. A list is a mutable, ordered collection of elements that can hold various data types, including numbers, strings, and other objects. An empty list, denoted by [], simply means that the list 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.) Let's pretend the spam includes the list ['a', 'b', 'c', 'd'] for the next three queries.

Ans: To assign the value 'hello' as the third value in a list stored in a variable called spam, you can use indexing to access the third element (remembering that Python uses 0-based indexing) and then assign the new value to it.

Spam[2] = ‘hello’

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

Ans: 3

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

Ans: spam[-1] will return the value of the last element of the list ‘spam’.

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.

Ans: spam[:2] will give first two elements of the spam list as the output in a new list. So the answer is

[3.14, ‘cat’]

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

Ans: 1

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

Ans: Using append on the bacom list, it will add 99 at the end of the list.

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

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

Ans: remove statement removes the first occurrence of ‘cat’ from the list bacon.

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

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

Ans: ‘+’ is the list concatenation operator.

‘\*’ is the list replication operator.

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

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

append() takes single argument as the element you want to add to the list, and appends it at the end of the list.

insert() takes two arguments, first argument is the index number where you want to insert the element in the list and second argument is the element itself.

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

Ans: remove() and pop() methods are used for removing items from a list.

remove() removes the first occurrence of the argument item, while pop() removes the item from the index number which was passed as argument.

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

Ans: Lists and strings are both data structures in Python, but they have some important differences. However, there are also similarities between them. Here are some ways in which list values and string values are similar: 1. Indexing 2. Slicing 3.Sequences 4.Iteration 5.Concatenation 6.Length

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

Ans: Tuple is an immutable data structure while list is a mutable data structure.

Lists are created using [] square brackets while tuples are used using () parantheses.

Lists have more built-in methods for manipulation while tuples have fewer methods because of their immutability.

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

Ans: To create a tuple that contains the integer 42, you can enclose the integer in parentheses () followed by a comma ,. The comma , is necessary to distinguish it as a tuple with a single element. Without the comma, Python would interpret the parentheses as regular parentheses, and you would not have a tuple.

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 to a tuple or a tuple to a list in Python, you can use the tuple() constructor to convert a list to a tuple and the list() constructor to convert a tuple to a list.

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

Ans: Variables that "contain" list values in Python do not actually contain the list data directly. Instead, they contain references or pointers to the list data in memory. In Python, variables are essentially labels or names that reference objects in memory, rather than containers that hold the data themselves.

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

Ans: the key difference is that copy.copy() creates a shallow copy that shares references to nested objects, while copy.deepcopy() creates a deep copy that recursively duplicates all objects, resulting in a completely independent copy. The choice between them depends on whether you need a shallow or deep copy based on the structure and relationships within the original object.