1. What exactly is []?

Ans: It denotes 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.)

Ans: spam = [2, 4, 6, 8, 10]

The third value has index 2. Thus, we need to insert the desired value, 'hello', at index 2.

spam = [2, 4, 6, 8, 10]

spam.insert(2, 'hello')

print(spam)

Output -

[2, 4, 'hello', 6, 8, 10]

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

spam = ['a', 'b', 'c', 'd']

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

Ans: 'd'

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

Ans: 'd'

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

Ans: ['a', 'b']

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

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

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

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

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

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

Ans: The list concatenation operator is + and the list replication operator is \*

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

Ans: append() takes the element as the argument and adds it to the end of the list. insert() takes tow arguments (index, element) and adds the element at the specified index.

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

Ans: The two methods are pop() and remove().

pop() takes the index of the element to be removed as the argument.

remove() takes the element to be removed as the argument.

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

Ans: Both lists and strings are ordered collections. The elements in a string and list have indices. Lists and strings can be concatenated and spliced.

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

Ans: Lists and tuples are collections. But the difference lies in the fact that lists are mutable (i.e. the elements of a list can be changed) while tuples are immutable.

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

Ans: t = (42,). The element should be followed by a comma in order for Python to recognize it as a tuple.

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

Ans: This can be achieved by using the list() and tuple() functions.

For example,

tuple(['a', 'b', 'c']) gives ('a', 'b', 'c') as the output (list value's tuple form)

list(('a', 'b', 'c')) gives ['a', 'b', 'c'] as the output (tuple value's list form)

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

Ans: When storing mutable entities such as lists or dictionaries, the variable contains references to the mutable entities rather than storing the entities themselves. In the case of immutable entities ( integers, string, tuples etc.), the value itself i stored in the variable.

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

Ans: copy.copy() returns a shallow copy of an object, and copy.deepcopy() returns a deep copy of an object. The difference between shallow copy and deep copy is seen in the case of compound objects such as lists. In a shallow copy, a new compound object is created and is populated with references to the child objects found in the original. When a change is made to the shallow copy, it is reflected in the original as well. In a deep copy, a new compound object is created and it is recursively populated with copies of the child objects found in the original. A change made to a deep copy does not affect the original.