1. What exactly is []?

In python, a list is created by placing elements inside square brackets [], separated by commas. And a list can have any number of items and they may be of different types.

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.insert(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)]?

‘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?

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

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

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

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

List concatenation: +

List replication: \*

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

append() appends an element into at the end of a list

insert() insert an element into a list at a specified index

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

I can find 4 methods of doing that: pop(), remove(), clear(), del a\_list[<index>]

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

Lists are very much like strings but can hold different data types. The both are ordered collections of components. The values that make up a list are called elements, and the values that make up a string are called characters.

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

The main difference is that tuples are immutable (static) object while lists are immutable (dynamic). Lists are specified by square brackets, and tuples are specified by round brackets.

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?

Get a list value in tuple form: list(<a tuple>)

Get a tuple value in list form: tuple(<a list>)

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

They contain a pointer to (reference of) the location of memory where the first element of list located.

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

In Python, assignment statements do not copy objects, they create bindings between a target and an object. A copy sometimes is needed so one can change one copy without changing the other. There are two ways to create copies: Deep copy and Shallow copy.

In order to make these copy, we use copy module.

Deep copy is a process in which the copying process occurs recursively. It means first constructing a new collection object and then recursively populating it with copies of the child objects found in the original. In case of deep copy, a copy of object is copied in other object, which means that any changes made to a copy of object do not reflect in the original object. This is implemented using copy.deepcopy() function.

A shallow copy means constructing a new collection object and then populating it with references to the child objects found in the original. The copying process does not recurse and therefore won’t create copies of the child objects themselves. In case of shallow copy, a reference of object is copied in another object. It means that any changes made to a copy of object do reflect in the original object. This is implemented using copy.copy() function.

Note: The difference between shallow and deep copy is only relevant for compound objects (objects that contain other objects, such as lists or class instances).