1. What exactly is []?

[] is 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.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?

The most conventional method to perform the list concatenation, the use of **“+” operator** can easily add the whole of one list behind the other list and hence perform the concatenation.

**The \* operator** replicates the elements in the list.

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

The only difference between append() and insert() is that insert function allows **us to add a specific element at a specified index of the list** unlike append() where we can add the element only at end of the list.

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

The methods are **remove(), pop()**.

The **pop**() **method** removes the element at the specified position. Syntax. list.**pop**(pos).

The **remove**() **method** takes a single element as an argument and removes it from the list.

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

The values that make up a list are called **its elements**. ... Lists are similar to strings, which are ordered collections of characters, except that the elements of a list can have any type and for any one list, the items can be of different types. List and string both are used **to store value /data** and both are sequence. string are used to store character in single or double quotes. list are mutable and string are immutable. The similarity between Lists and Strings in Python is **that both are sequences**. The differences between them are that firstly, Lists are mutable but Strings are immutable. Secondly, elements of a list can be of different types whereas a String only contains characters that are all of String type.

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

The key difference between the tuples and lists is that while **the tuples are immutable objects the lists are mutable**. This means that tuples cannot be changed while the lists can be modified. Tuples are more memory efficient than the lists.

The declaration or initialization is different. The list can be initialized with ‘[ ]’. While tuple can initialized with ‘( )’.

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

hit=(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 tuple into list in Python, **call list() builtin function and pass the tuple as argument to the function**.

To convert a list into tuple in Python, **call tuple() builtin function and pass the list as argument to the function**.

**we have mentioned three common and most widely used methods to convert lists to a tuple in python programming.**

1. 1) Using tuple() builtin function.
2. 2) Using loop inside the tuple.
3. 3) Unpack list inside the parenthesis.

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

Variables will **contain references** to list values rather than list values themselves. But for strings and integer values, variables simply contain the string or integer value.

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

Essentially, sometimes you may want to have the original values unchanged and only modify the new values or vice versa. In Python, there are two ways to create copies:

1. Shallow Copy
2. Deep Copy

## Shallow Copy

A shallow copy creates a new object which stores the reference of the original elements.

So, a shallow copy doesn't create a copy of nested objects, instead it just copies the reference of nested objects. This means, a copy process does not recurse or create copies of nested objects itself.

## Deep Copy

A deep copy creates a new object and recursively adds the copies of nested objects present in the original elements.

Let’s continue with example 2. However, we are going to create deep copy using deepcopy() function present in copy module. The deep copy creates independent copy of original object and all its nested objects.