1. What exactly is []?

Answer-1. This [] is called square brackets. This are used to define list, store values inside them.

Ex- a = [2, 3, 4, 6, 44]

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.)

Answer-2. We can assign the value ‘hello’ as the third value by these steps-:

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

spam.insert(2,’hello’)

print(spam)

[2, 4, ’hello’, 6, 8, 10]

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)]?

Answer-3. The value is ‘d’.

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

Answer-4. The value is ‘d’.

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

Answer-5. The value is [‘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')?

Answer-6. ## There is error because of comma in ‘cat,’. So the answer for the question is from list

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

The value of bacon.index('cat') is 1.

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

Answer-7. It adds the value 99, so list looks like, bacon = [3.14, 'cat', 11, 'cat', True, 99].

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

Answer-8. It removes the ‘cat’, list look like, bacon = [3.14, 11, 'cat', True, 99].

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

Answer-9. We can use operator + for concatenation and \* for replication of list.

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

Answer-10. Method append() adds value in the end of list, whereas by using insert() we can add value at any index place.

Eg. a = [4,5,6,7,8] ## append method

a.append(879)

print(a)

[4,5,6,7,8,879]

B = [8,9,65,45] ## insert method

B.insert(1,222)

print(B)

[8,222,9,65,45]

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

Answer-11. We can remove items from list using remove() and pop() methods.

Eg. ## using remove method.

g= [5,6,7,8,2]

g.remove(8) ## by passing the value in remove(), we can remove that item from list.

print(g)

[5,6,7,2]

## by using pop()

h=[8,9,4,6,3]

h.pop(1) ## we can remove the items by passing its index value in pop.

print(h)

[8,4,6,3]

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

Answer-12. List and string values are identical as we can use indexing and slicing methods on them.

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

Answer-13. Difference between tuples and lists is tuples are immutable and list are mutable.

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

Answer-14. Tuple = (42)

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

Answer-15. To get list values as tuple form, we have to pass list in tuple().

Eg. a= [2,3,4,5,6]

tuple(a) ## pass list in tuple()

(2,3,4,5,6) ## output in tuple form

To get tuple value’s in list form, we have to pass tuple in list[].

b = (3,2,1,6,5)

list(b) ## pass tuple in list[]

[3,2,1,6,5] ## output in list form

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

Answer-16. They contain address of the list values that we have assign to the variables.

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

Answer-17. Difference between copy.copy() and copy.deepcopy() is that-:

copy.copy() only stores refrence of values of the original that means any changes made to the copied object will be reflected in the original.

In copy.deepcopy() copy the original object values that means any changes made to the copied object will not be reflected in the original.