1. What exactly is []?

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.)

Spam[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?

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

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

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

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

List Concatenation==> + operator and List replication==> \* operator

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

Append() method add the element to the list always at the end.

A=[1,2,3,4] Insert() method add the elements to the list at the specified index

A.append(5) A.insert(2,5)

A=[1,2,3,4,5] A=[1,2,5,3,4]

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

remove() method and pop() method

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

Lists are similar to strings, which are ordered collections of characters.

L=[10,20,30,40]

S=[‘10’,’20’,’30’,40’]

L[1]==> 10

S[1]==> ‘10’

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

tuple is immutable where as list is mutable

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?

tuple form==> tuple(list)

list form==> list(tuple)

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

tuple,string,integer,floats

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

copy () create reference to original object. If you change copied object - you change the original object. . deepcopy() creates new object and does real copying of original object to new one. Changing new deepcopied object does not affect original object.

***Copy.copy()***

a = [[1, 2, 3], [4, 5, 6]]

id(a) == id(b) id(a[0]) == id(b[0])

False True

***copy.deepcopy()***

b = copy.deepcopy(a)

id(a) == id(b) id(a[0]) == id(b[0])  
False False