1. What exactly is []?

A: It is defined as empty list.

1. In a list of values stored in a variable called spam, how would you assign the value &#39;hello&#39; as the third value? (Assume [2, 4, 6, 8, 10] are in spam.)

A:

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

spam[2]='hello'

spam

Out[4]: [2, 4, 'hello', 8, 10]

Q3. What is the value of spam[int(int(&#39;3&#39; \* 2) / 11)]?

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

spam[int(int('3'\* 2)/11)]

Out[13]: 'd'

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

‘d’

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

A: [‘a’, ‘b’]

Let&#39;s pretend bacon has the list [3.14, &#39;cat,&#39; 11, &#39;cat,&#39; True] for the next three questions.

6. What is the value of bacon.index(&#39;cat&#39;)?

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

bacon.index('cat')

1

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

A:bacon.append(99)

bacon

Out[18]: [3.14, 'cat', 11, 'cat', True, 99]

8. How does bacon.remove(&#39;cat&#39;) change the look of the list in bacon?

A:bacon.remove('cat')

bacon

Out[20]: [3.14, 11, 'cat', True, 99]

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

A: List concatenation: +

List replication: \*

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

A: append(): data can be inserted at the end only

Insert(): data can be inserted at any specific location

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

A: del bacon, remove

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

A:

* can be concatenated
* can be replicated
* used under for loops
* passed through len functions

13. the difference between tuples and lists?

A:

Tuples are not mutable and presented by ()

Lists are mutable; can be changed by addition, deletion and presented by []

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

A: l=(42,)

l

Out[25]: (42,)

15. How do you get a list value&#39;s tuple form? How do you get a tuple value&#39;s list form?

A: a=['a', 8, 90,'b']

tuple(a)

b=(10,7,3,2,1)

list(b)

16. Variables that &quot;contain&quot; list values are not necessarily lists themselves. Instead, what do they

contain?

A: reference of list values

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

A:

import copy

li1 = [1, 2, [3, 5], 4]

# using copy for shallow copy

li2 = copy.copy(li1)

print("li2 ID: ", id(li2), "Value: ", li2)

# using deepcopy for deepcopy

li3 = copy.deepcopy(li1)

print("li3 ID: ", id(li3), "Value: ", li3)