1. What exactly is []?

Answer:

This is an 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.)

Answer:

We can apply the index operation to assign the value ‘hello’ as the third value of the spam variable.

spam[2] = ‘hello’

However, in this way we will be losing the original third value of the spam variable, thereby following the operation we can apply to keep all the original values along with the new value as the third value:

spam = spam[0:2] + ['hello'] + spam[2:]

or,

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)]?

Answer:

spam[int(int('3' \* 2) / 11)] = ‘d’

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

Answer:

spam[-1] = ‘d’

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

Answer:

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')?

Answer:

1

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

Answer:

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

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

Answer:

If we consider the original values of bacon, then the output will be as follows:

[3.14, 11, 'cat', True]

If we consider the appended value of bacon, then the output will be as follows:

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

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

Answer:

-> the operator for list concatenation is + (plus)

-> the operator for list replication is \* (multiplication)

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

-> append() will add values only to the end of a list

-> insert() can add values anywhere in the list by providing index number

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

-> del statement and remove method of list can be using for removing items from a list

Example:

Let’s say, lst = [‘a’, ‘b’, ‘c’, ‘d’, ‘e’]

Now, removing ‘b’ from lst using del statement and remove method:

-> del lst[1]

-> lst.remove(‘b’)

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

-> Both list and string are using len() function to find the length

-> Both have similar index and slice operation

-> Both are using + for concatenation and \* for repetition

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

|  |  |
| --- | --- |
| tuples | lists |
| tuples are immutable | lists which are mutable |
| Cannot add, remove or insert values once they have created. | Can add, remove or insert values into list after even they have created. |
| Declared with parenthesis, (). | Declared with square brackets, []. |

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

Answer:

(42,)

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

Answer:

-> We can pass the entire list as a parameter within the tuple() function

-> We can pass the entire tuple as a parameter within the list() function

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

Answer:

They contain references to list values.

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

Answer:

-> The copy() returns a shallow copy of the list

-> deepcopy() returns a deep copy of the list

-> The difference can be observed only for compound object (like list of list).

-> When we modify nested object, changes will get reflect in shallow copy but not in deep copy.

Example:

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

b = a

shallow = copy.copy(a)

deep = copy.deepcopy(a)

a[5].append(21)

Result:

a = [1, 2, 3, 4, 5, [1, 2, 3, 4, 5, 21]]

shallow = [1, 2, 3, 4, 5, [1, 2, 3, 4, 5, 21]]

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