1. What exactly is []?

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.)

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)]?

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

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

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')?

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

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

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

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

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

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

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

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

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

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

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

***Answers :***

1. [] represents an empty list in Python. It is a built-in data structure that can hold a collection of items, separated by commas and enclosed within square brackets.
2. To assign the value 'hello' as the third value in the list stored in the variable spam, you can use the index notation and assignment operator as follows:

spam[2] = 'hello'

This will replace the existing value at index 2 with 'hello'.

1. The value of spam[int(int('3' \* 2) / 11)] is 'd'. Here, '3' \* 2 creates the string '33', then int('33') converts it to the integer 33. Dividing 33 by 11 results in 3. The index 3 in the list spam corresponds to the value 'd'.
2. The value of spam[-1] is 'd'. The negative index -1 refers to the last item in the list spam.
3. The value of spam[:2] is ['a', 'b']. This is a list slicing operation that retrieves the elements from index 0 up to index 2 (exclusive), resulting in the first two elements of the list.
4. The value of bacon.index('cat') is 1. The index() method returns the index of the first occurrence of the specified value in the list bacon. In this case, 'cat' is first found at index 1.
5. The bacon.append(99) method appends the value 99 to the end of the list bacon. After calling this method, the list bacon will look like [3.14, 'cat', 11, 'cat', True, 99].
6. The bacon.remove('cat') method removes the first occurrence of the value 'cat' from the list bacon. After calling this method, the list bacon will look like [3.14, 11, 'cat', True].
7. The list concatenation operator is +, which allows you to combine two or more lists into a single list. The list replication operator is \*, which allows you to create a new list by repeating the elements of an existing list a certain number of times.
8. The append() method is used to add an item at the end of a list, while the insert() method allows you to insert an item at a specific position in the list, shifting the existing elements to accommodate the new item.
9. The two methods for removing items from a list are remove() and pop(). The remove() method removes the first occurrence of a specified value from the list, while the pop() method removes an item at a specific index and returns its value.
10. Both list values and string values can be indexed and sliced. They are ordered sequences of elements, where each element can be accessed by its index. Additionally, both lists and strings are iterable, meaning you can loop over their elements using loops or comprehensions.
11. Tuples and lists are both used to store collections of items. The main difference is that tuples are immutable, meaning their elements cannot be changed once defined, whereas lists are mutable, allowing elements to be modified, added, or removed.
12. To create a tuple value that only contains the integer 42, you can use parentheses to enclose the value:

my\_tuple = (42,)

The comma after the integer is necessary to indicate that it is a tuple with a single element.

1. To get a list value's tuple form, you can use the tuple() function. For example:

my\_list = [1, 2, 3]

my\_tuple = tuple(my\_list)

To get a tuple value's list form, you can use the list() function. For example:

my\_tuple = (1, 2, 3)

my\_list = list(my\_tuple)

1. Variables that "contain" list values are actually referencing the memory address where the list is stored. In Python, variables are references to objects, so a variable that contains a list value holds a reference to the list object in memory.
2. The copy.copy() function creates a shallow copy of a list, which means that a new list is created with references to the same objects as the original list. If the objects within the list are mutable, changes made to those objects will be reflected in both the original and copied list. On the other hand, copy.deepcopy() creates a deep copy of a list, creating a new list with new objects that are copies of the original objects. Any modifications made to the copied list or its objects will not affect the original list.