**What exactly is []?**

[] is an empty list in Python. It represents a list with no elements.

**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.)**

assign 'hello' as the third value by using indexing:

spam[2] = 'hello'

After this operation, spam will be [2, 4, 'hello', 8, 10].

**What is the value of spam[int(int('3' \* 2) / 11)]?**

Let's break it down:

'3' \* 2 results in '33'.

int('33') converts it to the integer 33.

33 / 11 results in 3.0.

int(3.0) converts it to 3.

So, spam[3] is 'd'.

The value is 'd'.

**What is the value of spam[-1]?**

spam[-1] refers to the last element in the list, which is 'd'.

**What is the value of spam[:2]?**

spam[:2] returns a slice of the list from the start up to (but not including) index 2. So, it returns ['a', 'b'].

**What is the value of bacon.index('cat')?**

bacon.index('cat') returns the index of the first occurrence of 'cat' in the list, which is 1.

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

bacon.append(99) adds 99 to the end of the list. After this operation, bacon will be [3.14, 'cat', 11, 'cat', True, 99].

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

bacon.remove('cat') removes the first occurrence of 'cat' from the list. After this operation, bacon will be [3.14, 11, 'cat', True, 99].

1. **What are the list concatenation and list replication operators?**

The list concatenation operator is +, which combines two lists.

The list replication operator is \*, which repeats a list a specified number of times.

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

append() adds an element to the end of the list.

insert() adds an element at a specified position in the list.

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

The two methods are:

remove(): Removes the first occurrence of a specified value.

pop(): Removes an element at a specified index (or the last element if no index is specified).

1. **Describe how list values and string values are identical.**

Both lists and strings are sequences, meaning they can be indexed, sliced, and iterated over. They also support operations like concatenation and replication.

1. **What's the difference between tuples and lists?**

Tuples are immutable (cannot be changed after creation), while lists are mutable (can be modified).

Tuples are defined using parentheses (), while lists are defined using square brackets [].

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

You can type it as (42,). The comma is necessary to distinguish it from a single integer in parentheses.

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

**To convert a list to a tuple, use tuple():**

my\_tuple = tuple(my\_list)

**To convert a tuple to a list, use list():**

my\_list = list(my\_tuple)

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

Variables contain references to list values, not the actual list itself. This means that multiple variables can reference the same list in memory.

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

copy.copy() creates a shallow copy of an object, meaning it copies the object but not the objects it references (e.g., nested lists are not deeply copied).

copy.deepcopy() creates a deep copy of an object, meaning it copies the object and all objects it references recursively.