1. What exactly is `[]`?

`[]` represents an empty list in Python, which is a data structure used to store an ordered sequence of elements.

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`.)

You can assign the value `'hello'` to the third position (index 2) in the list `spam` like this:

```python

spam[2] = 'hello'

```

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

The value of this expression is `'d'`, as `int('3' \* 2)` evaluates to `33`, and `33 / 11` is `3`. So, `spam[3]` corresponds to the value `'d'`.

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

The value of `spam[-1]` is `'d'`, which represents the last element in the list.

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

The value of `spam[:2]` is `['a', 'b']`, which is a sublist containing the first two elements of the list.

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

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

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

`bacon.append(99)` adds the value `99` to the end of the list `bacon`, so the list becomes `[3.14, 'cat', 11, 'cat', True, 99]`.

8. 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 `bacon`. After this operation, the list becomes `[3.14, 11, 'cat', True]`.

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

The list concatenation operator is `+`, which combines two lists into one. The list replication operator is `\*`, which creates a new list by repeating the original list a specified number of times.

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

- `append()`: Adds an element to the end of the list.

- `insert()`: Inserts an element at a specific index in the list, shifting other elements to make space.

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

Two methods for removing items from a list are:

- `remove()`: Removes the first occurrence of a specific value.

- `pop()`: Removes an item at a specific index and returns its value.

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

Both list values and string values are ordered sequences of elements. Each element in a list or a string has an index that determines its position within the sequence. Indexing and slicing operations work similarly on both lists and strings.

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

Tuples and lists are both ordered sequences, but the main difference is that lists are mutable (can be changed after creation), while tuples are immutable (cannot be changed after creation).

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

You can create a tuple containing the integer 42 like this: `(42,)`

15. 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: `tuple(my\_list)`

To convert a tuple to a list: `list(my\_tuple)`

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

Variables that "contain" list values actually contain references to the list objects in memory. They don't store the list directly, but rather a pointer to its location in memory.

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

`copy.copy()` creates a shallow copy of an object, including a copy of the object itself and references to the objects it contains. `copy.deepcopy()` creates a deep copy, which means it recursively copies all the nested objects as well, creating a completely independent copy.