1. **What exactly is []?**
    - [] is an empty list in Python. Lists are used to store multiple items in a single variable.
2. **In a list of values stored in a variable called spam, how would you assign the value 'hello' as the third value?**
    - If spam is [2, 4, 6, 8, 10], you would assign 'hello' as the third value (index 2) using:

spam[2] = 'hello'

3. **What is the value of spam[int(int('3' * 2) / 11)]?**
    - The expression '3' * 2 repeats the string '3' twice, giving '33'.
    - int('33') converts '33' to the integer 33.
    - Dividing 33 by 11 gives 3.
    - So, spam[int(int('33') / 11)] is equivalent to spam[3].
    - With spam = ['a', 'b', 'c', 'd'], spam[3] is 'd'.
4. **What is the value of spam[-1]?**
    - spam[-1] accesses the last element of the list. In spam = ['a', 'b', 'c', 'd'], the last element is 'd'.
5. **What is the value of spam[:2]?**
    - spam[:2] returns the first two elements of the list. In spam = ['a', 'b', 'c', 'd'], spam[:2] is ['a', 'b'].
6. **What is the value of bacon.index('cat')?**
    - The method index('cat') returns the index of the first occurrence of 'cat' in the list bacon = [3.14, 'cat', 11, 'cat', True]. The first 'cat' occurs at index 1.
7. **How does bacon.append(99) change the look of the list value in bacon?**
    - The append(99) method adds 99 to the end of the list. So, if bacon = [3.14, 'cat', 11, 'cat', True], after bacon.append(99), the list becomes:

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

8. **How does bacon.remove('cat') change the look of the list in bacon?**
    - The remove('cat') method removes the first occurrence of 'cat' from the list. If bacon = [3.14, 'cat', 11, 'cat', True], after bacon.remove('cat'), 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.
    - The list replication operator is *, which repeats the 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(index, item) adds an element at a specific index in the list.

11. **What are the two methods for removing items from a list?**
    - remove(item) removes the first occurrence of the item from the list.
    - pop(index) removes the item at the specified index and returns it. If no index is specified, it removes the last item.
12. **Describe how list values and string values are identical.**
    - Both lists and strings are sequences of items. You can access elements using indexing, slice them, and iterate through them in loops.
13. **What's the difference between tuples and lists?**
    - Lists are mutable, meaning their values can be changed after creation.
    - Tuples are immutable, meaning their values cannot be changed after creation.
14. **How do you type a tuple value that only contains the integer 42?**
    - To create a tuple with a single value, you must include a comma: (42,).
15. **How do you get a list value's tuple form? How do you get a tuple value's list form?**
    - Use tuple(list) to convert a list to a tuple.
    - Use list(tuple) to convert a tuple to a list.
16. **Variables that "contain" list values are not necessarily lists themselves. Instead, what do they contain?**
    - Variables that "contain" lists actually contain references to the list objects in memory.
17. **How do you distinguish between copy.copy() and copy.deepcopy()?**
    - copy.copy() creates a shallow copy of the object, meaning nested objects are not fully copied.
    - copy.deepcopy() creates a deep copy, meaning all nested objects are recursively copied, resulting in a completely independent copy.