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()?

**ANS:  
Here are the responses to your queries:**

**1. `[]` denotes an empty list in Python.**

**2. To assign the value 'hello' as the third value in the list stored in the variable `spam`, you would use the index `2` since Python uses zero-based indexing. The assignment would look like this:**

**```python**

**spam[2] = 'hello'**

**```**

**3. The value of `spam[int(int('3' \* 2) / 11)]` can be calculated as follows:**

**- `int('3' \* 2)` results in the integer `33`.**

**- `int('33') / 11` results in `3.0`.**

**- So, `spam[3]` would access the fourth element of the list, which is `'d'`.**

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

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

**6. `bacon.index('cat')` returns the index of the first occurrence of 'cat' in the list `bacon`. In this case, it returns `1`.**

**7. `bacon.append(99)` adds the value `99` to the end of the list `bacon`.**

**8. `bacon.remove('cat')` removes the first occurrence of 'cat' from the list `bacon`.**

**9. The list concatenation operator is `+`, and the list replication operator is `\*`.**

**10. The `append()` method adds an item to the end of the list, while the `insert()` method inserts an item at a specified position in the list.**

**11. The two methods for removing items from a list are `remove()` and `pop()`.**

**12. List values and string values are identical in that they are both sequences of elements. They support similar operations like indexing, slicing, and iteration.**

**13. The main difference between tuples and lists is that tuples are immutable (unchangeable), while lists are mutable (changeable).**

**14. To create a tuple value that only contains the integer `42`, you would write `(42,)`.**

**15. To get a list value's tuple form, you can use the `tuple()` constructor. To get a tuple value's list form, you can use the `list()` constructor.**

**16. Variables that "contain" list values are actually storing references to the list objects in memory.**

**17. `copy.copy()` creates a shallow copy of a list, meaning it duplicates the top-level structure of the list, but not the nested objects within it. `copy.deepcopy()` creates a deep copy, duplicating the list as well as all the nested objects within it recursively.**