# Assignment Answers

## 1. What exactly is []?

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

spam[2] = 'hello'  
After this, spam becomes [2, 4, 'hello', 8, 10].

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

The value is 'd'.

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

The value is 'd'.

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

The value is ['a', 'b'].

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

The value is 1.

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

It adds 99 to the end of the list: [3.14, 'cat', 11, 'cat', True, 99].

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

It removes the first occurrence of 'cat': [3.14, 11, 'cat', True].

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

Concatenation: + (e.g., [1, 2] + [3, 4] results in [1, 2, 3, 4]).  
Replication: \* (e.g., [1, 2] \* 3 results in [1, 2, 1, 2, 1, 2]).

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

append(x) adds x to the end of the list.  
insert(i, x) inserts x at index i, shifting other elements.

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

remove(x): Removes the first occurrence of x.  
pop(i): Removes and returns the element at index i (or last if i is omitted).

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

Both are sequences, support indexing, slicing, iteration, and can be concatenated or replicated.

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

Lists ([]) are mutable (can be changed).  
Tuples (()) are immutable (cannot be changed).

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

(42,)

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

Convert list to tuple: tuple(my\_list).  
Convert tuple to list: list(my\_tuple).

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

They contain references (or memory addresses) pointing to the list objects.

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

copy.copy() creates a shallow copy (copies references but not nested objects).  
copy.deepcopy() creates a deep copy (duplicates all nested objects).