Assignment 4

1. What exactly is [ ]?

Ans: The empty list value, which is a list value that contains no items. This is similar to how '' is the empty string value.

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.)

Ans: spam[2] = 'hello' (Notice that the third value in a list is at index 2 because the first index is 0.)

```
spam=[2, 4, 6, 8, 10]
print(spam)
print("------------------------")
spam[3]=["Hello"]
print(spam)
```

```
[2, 4, 6, 8, 10]
------------------------
[2, 4, 6, ['Hello'], 10]
```

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)]?

Ans: 'd' (Note that '3' * 2 is the string '33', which is passed to int() before being divided by 11. This eventually evaluates to 3. Expressions can be used wherever values are used.)

```
spam= ["a", "b", "c", "d"]
spam=[int(int('3' * 2) / 11)]
print(spam)
```

```
[3]
```

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

Ans: 'd' (Negative indexes count from the end.)

```
spam= ["a", "b", "c", "d"]
print(spam[-1])
```

```
d
```

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

Ans: ['a', 'b']

```
spam= ["a", "b", "c", "d"]
print(spam[0:2])
```

```
['a', 'b']
```

Let's pretend bacon has the list [3.14, 'cat,' 11, True] for the next three questions.

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

Ans: 1

```
bacon=[3.14, "cat", 11, True]
print(bacon.index("cat"))
```

```
1
```

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

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

```
bacon=[3.14, "cat", 11, True]
print(bacon)
print("------------------------")
bacon.append(99)
print(bacon)
```

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

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

Ans: [3.14, 11, True]

```
bacon=[3.14, "cat", 11,True]
print(bacon)
print("------------------------")
bacon.remove("cat")
print(bacon)
```

```
[3.14, 'cat', 11, True]
------------------------
[3.14, 11, True]
```

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

Ans: The operator for list concatenation is +, while the operator for replication is *. (This is the same as for strings.)

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

Ans: While append() will add values only to the end of a list, insert() can add them anywhere in the list.

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

Ans: The del statement and the remove() list method are two ways to remove values from a list.

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

Ans: Both lists and strings can be passed to len(), have indexes and slices, be used in for loops, be concatenated or replicated, and be used with the in and not in operators.

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

Ans: Lists are mutable; they can have values added, removed, or changed. Tuples are immutable; they cannot be changed at all. Also, tuples are written using parentheses, (and), while lists use the square brackets, [and].

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

Ans: (42,) (The trailing comma is mandatory.)

```
tup=int(42)
print(tup)
```

```
42
```

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

Ans: The tuple() and list() functions, respectively

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

Ans: They contain references to list values.

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

Ans: The copy.copy() function will do a shallow copy of a list, while the copy.deepcopy() function will do a deep copy of a list. That is, only copy.deepcopy() will duplicate any lists inside the list.