The *extend* method vs the + operator.

- **+ creates a fresh list (with a new memory reference)**
- ***extend* operates on list `li` in place.**

```
>>> li.extend([9, 8, 7])
>>>li
[1, 2, 'i', 3, 4, 5, 'a', 9, 8, 7]
```

***Confusing*:**

- **Extend takes a list as an argument.**
- **Append takes a singleton as an argument.**

```
>>> li.append([10, 11, 12])
>>> li
[1, 2, 'i', 3, 4, 5, 'a', 9, 8, 7, [10, 11, 12]]
```

Operations on Lists Only 3

```
>>> li = ['a', 'b', 'c', 'b']

>>> li.index('b')      # index of first occurrence
1

>>> li.count('b')      # number of occurrences
2

>>> li.remove('b')     # remove first occurrence
>>> li
  ['a', 'c', 'b']
```

Operations on Lists Only 4

```
>>> li = [5, 2, 6, 8]

>>> li.reverse()     # reverse the list *in place*
>>> li
  [8, 6, 2, 5]

>>> li.sort()        # sort the list *in place*
>>> li
  [2, 5, 6, 8]

>>> li.sort(some_function)
    # sort in place using user-defined comparison
```

Tuples vs. Lists

- **Lists slower but more powerful than tuples.**
  - Lists can be modified, and they have lots of handy operations we can perform on them.
  - Tuples are immutable and have fewer features.

- **To convert between tuples and lists use the list() and tuple() functions:**

```
li = list(tu)
tu = tuple(li)
```