## 9.7. Mutating Methods

You've seen some methods already, like the count and index methods. Methods are either mutating or non-mutating. Mutating methods are ones that change the object after the method has been used. Non-mutating methods do not change the object after the method has been used.

The count and index methods are both non-mutating. Count returns the number of occurances of the argument given but does not change the original string or list. Similarly, index returns the leftmost occurance of the argument but does not change the original string or list. Below we'll talk about list methods in general. Keep an eye out for methods that are mutating!

## 9.7.1. List Methods

The dot operator can also be used to access built-in methods of list objects. append is a list method which adds the argument passed to it to the end of the list. Continuing with this example, we show several other list methods. Many of them are easy to understand.

```
Original - 1 of 1
                      Save & Run
                                                                       Show in CodeLens
        1 \text{ mylist} = []
        2 mylist.append(5)
        3 mylist.append(27)
        4 mylist.append(3)
        5 mylist.append(12)
        6 print(mylist)
        8 \text{ mylist.insert} (1, 12)
        9 print(mylist)
       10 print (mylist.count(12))
       12 print (mylist.index(3))
       13 print (mylist.count(5))
       15 mylist.reverse()
       [5, 27, 3, 12]
       [5, 12, 27, 3, 12]
       3
       [12, 3, 27, 12, 5]
ning Lists"> (Cloning Lists html)
                                                                                      9.8. Append versus Con
       [3, 12, 12, 27]
       [3, 12, 12]
```

Activity: 1 -- ActiveCode (ac8\_6\_1)

There are two ways to use the pop method. The first, with no parameter, will remove and return the last item of the list. If you provide a parameter for the position, pop will remove and return the item at that position. Either way the list is changed.

The following table provides a summary of the list methods shown above. The column labeled result gives an explanation as to what the return value is as it relates to the new value of the list. The word **mutator** means that the list is changed by the method but nothing is returned (actually None is returned). A **hybrid** method is one that not only changes the list but also returns a value as its result. Finally, if the result is simply a return, then the list is unchanged by the method.

Be sure to experiment with these methods to gain a better understanding of what they do.

Method	Parameters	Result	Description
append	item	mutator	Adds a new item to the end of a list
insert	position, item	mutator	Inserts a new item at the position given
рор	none	hybrid	Removes and returns the last item
рор	position	hybrid	Removes and returns the item at position
sort	none	mutator	Modifies a list to be sorted
reverse	none	mutator	Modifies a list to be in reverse order
index	item	return idx	Returns the position of first occurrence of item
count	item	return ct	Returns the number of occurrences of item
remove	item	mutator	Removes the first occurrence of item

Details for these and others can be found in the Python Documentation (http://docs.python.org/py3k/library/stdtypes.html#sequence-types-str-bytes-bytearray-list-tuple-range).

It is important to remember that methods like append, sort, and reverse all return None. They change the list; they don't produce a new list. So, while we did reassignment to increment a number, as in x = x + 1, doing the analogous thing with these operations will lose the entire list contents (see line 8 below).

ning Lists"> (CloningLists.html) Save & Run

Load History

Show CodeLens

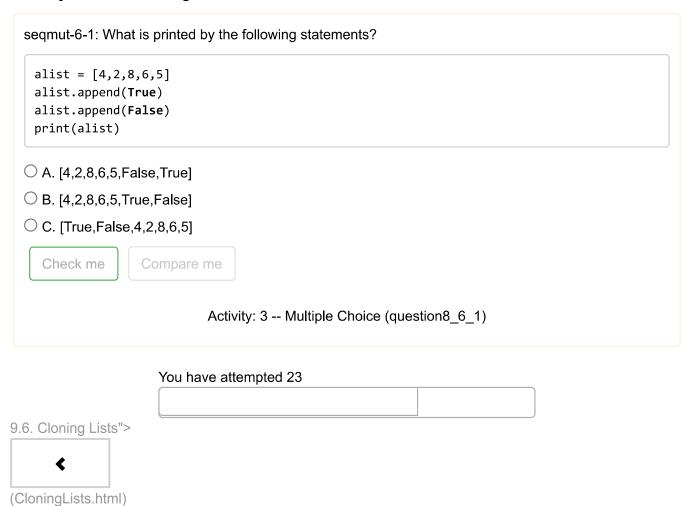
9.8. Append versus Con

1 mylist = []
2 mylist.append(5)

```
3 mylist.append(27)
4 mylist.append(3)
5 mylist.append(12)
6 print(mylist)
7
8 mylist = mylist.sort() #probably an error
9 print(mylist)
10

Activity: 2 -- ActiveCode (ac8_6_2)
```

## **Check your understanding**



ning Lists"> (Cloning Lists.html)

(Appendversus Concatenate">

(Appendversus Concatenate.html)

9.8. Append versus Con

## Mark as Completed

© Copyright 2017 bradleymiller. Created using Runestone (http://runestoneinteractive.org/) 4.1.17.

| Back to top

ning Lists"> (CloningLists.html)

9.8. Append versus Con