**COMP 141: List Functions**

Assume L, L1, and L2 are lists; and p and q are integers.

| | |
|---|---|
| len(L) | Returns the length of L. |
| L1 + L2 | Returns a **new** list consisting of all the items of L1 followed immediately by all the items of L2.<br>Note: just doing L1 + L2 doesn't change L1 or L2. You must store this new list somewhere if you want to use it later, by saying something like L = L1 + L2 (L will be a new list). |
| L[p] | Returns the item at index p in list L. (Indices start at zero!) |
| L[p:q] | Returns the sublist consisting of all items in L starting at index p and ending one before index q.<br>Note: if either p or q (or both) is left out, Python will assume p=0 (beginning of the list) and q=len(s) (the end of the list).<br>Ex: L[1:] will return L with the first item left out.<br>Using negative numbers for p and/or q counts from the end of the list:<br>Ex: L[-1] returns the last item in L; L[-2:] returns the last two items in L. |
| item in L | Returns True if item occurs somewhere in list L, False otherwise. item can be variable or a constant.<br>Ex: 3 in [1, 2, 3] returns True.<br>Ex: num = 6<br>num in [1, 2, 3] returns False. |
| item not in L | Returns False if item x occurs somewhere in list L, True otherwise. |
| L.index(item) | Similar to "item in L," but returns the lowest index at which the item is found in list L, and an error if L doesn't contain item. Think of this as looking through the list from left to right until the item is found (this left to right ordering is important if the item occurs more than once in L). |
| L.insert(p, item) | Inserts item into list L at position p, shifting elements to the right as necessary.<br>**Note: this changes L, so you don't have to do L = L.insert(p, item)** |
| L.append(item) | Attaches item to the end of list L. Equivalent to L = L + [item].<br>**Note: this changes L, so you don't have to do L = L.append(item)** |
| L.remove(item) | Removes the first instance of item in list L, but gives an error if there is no such item in L.<br>**Note: this changes L, so you don't have to do L = L.remove(item)** |

Notice that len() is the only function that is not "attached" to a list with a period.