

Lists

Sequences

- An object that holds multiple items, stored one right after the other
 - Can use different operations to manipulate and view data
- Two main types of sequences
 - Lists
 - Items can be changed or added
 - Tuples
 - Items *cannot* be changed or added

Lists

- An object that contains multiple data items
- Lists are dynamic structures
 - Contents inside can be changed
 - New contents can be added
 - Items can also be removed from the list
- Python lists can contain multiple data types
 - Ex. can have a list that is ints and strings

Lists Anatomy

- Lists are denoted by square brackets
 - You put whatever items inside the brackets, separating each item with a comma
- You can create an empty list to add items to it later with just an empty set of brackets
 - You add items with the built-in Python append function

```
# list of integers  
numbers = [2, 4, 6, 8]
```

```
# list of strings  
names = ["Kortni", "Devin", "Maggie"]
```

```
# mixed list  
mix = ["Kortni", 61, "Devin", 72, "Maggie", 10]
```

```
# creates an empty list  
empty_list = []
```

```
# adds item to list  
empty_list.append("Hello")  
empty_list.append("World")
```

Displaying Subsets

- Can create a subset of a list using the colon in brackets
 - Can print or create a whole new list from the sublist
- Number before colon: starts at that index, includes the value at that index
- Number after colon: ends at that index, does not include value at that index

<code>numbers = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]</code>	
<code>print(numbers[:])</code>	<code>[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]</code>
<code>print(numbers[:3])</code>	<code>[0, 1, 2]</code>
<code>print(numbers[1:])</code>	<code>[1, 2, 3, 4, 5, 6, 7, 8, 9]</code>
<code>print(numbers[1:5])</code>	<code>[1, 2, 3, 4]</code>
<code>print(numbers[:9])</code>	<code>[0, 1, 2, 3, 4, 5, 6, 7, 8]</code>

Searching Lists

- You can check to see if an item is in the list via if-statements
- Check if a variable (either input or statically set) is in the list via “in” command, comparing it to the list you want to check
- You can check if something’s *not* in the list by adding “not” before the keyword “in”
 - if search not in names

```
search = input("Enter in a name: ")  
  
# checks to see if that's in the list  
if search in names:  
    print("That name is in the list.")  
  
else:  
    print("That name is not in the list.")
```

Manipulating List Items

- You can change data in a list and place them back in their place
 - You can also just change the contents entirely
- You cannot use the “for ... in list” loop to do this -- it only allows using the list items, not changing

```
numbers[0] = 10  
[10, 5, 7, 9]
```

```
# list before  
print(numbers)  
  
# list length  
length = len(numbers)  
  
for i in range(length):  
    # alters each item  
    numbers[i] = numbers[i] + 1  
  
# list after  
print()  
print(numbers)
```

```
[2, 4, 6, 8]
```

```
[3, 5, 7, 9]
```


Lists: Notes

Table 8-1 A few of the list methods

Method	Description
<code>append(item)</code>	Adds <i>item</i> to the end of the list.
<code>index(item)</code>	Returns the index of the first element whose value is equal to <i>item</i> . A <code>ValueError</code> exception is raised if <i>item</i> is not found in the list.
<code>insert(index, item)</code>	Inserts <i>item</i> into the list at the specified <i>index</i> . When an item is inserted into a list, the list is expanded in size to accommodate the new item. The item that was previously at the specified index, and all the items after it, are shifted by one position toward the end of the list. No exceptions will occur if you specify an invalid index. If you specify an index beyond the end of the list, the item will be added to the end of the list. If you use a negative index that specifies an invalid position, the item will be inserted at the beginning of the list.
<code>sort()</code>	Sorts the items in the list so they appear in ascending order (from the lowest value to the highest value).
<code>remove(item)</code>	Removes the first occurrence of <i>item</i> from the list. A <code>ValueError</code> exception is raised if <i>item</i> is not found in the list.
<code>reverse()</code>	Reverses the order of the items in the list.