List Cheat Sheet

Lists allow us to store multiple python objects in one variable, in an order of our choosing. To create a list use the `[]` or the `list()` coercion function:

```
test_list = [1,2,3]
list(1,2,3)
```

Getting something out of a list:

List items are kind of like excel columns, where each item has an index number associated with it:

Index	Item
0	1
1	2.5
2	3
3	This'
4	[]

An item of a list can be accessed by using the "index of that item". this is done by typing the name of the item, and using the `[]` with an index inbetween:

```
name_of_list[index_you_want]
```

where the index is a integer pointing to the location of that item on the list. This brings us to pet peeve number one (aka `pet_peeve[0]`):

Python lists are 0 indexed, therefore you use 0 to get the 1st item in a list)

```
test_list = [1,2,3]
test[0] # will output 1
test[0] = 0 # will replace 1 with zero
test[-1] # using negative numbers will start from the end of the list and
count backwards
```

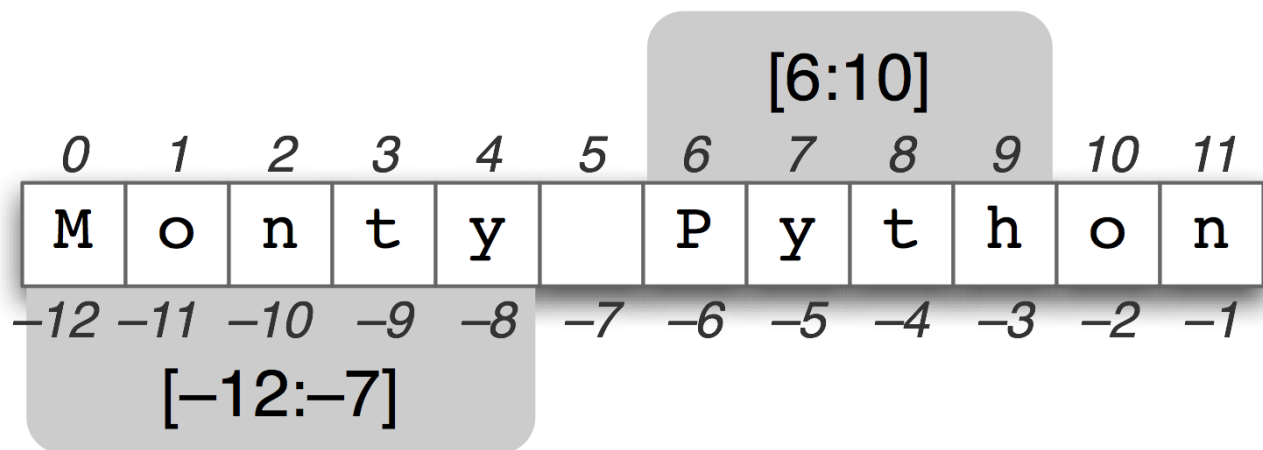*Note:* You can index strings as well, but you can put anything new into a string like line three above

Slicing more than one thing from a list:

You can slice out multiple values in a list by defining the start and stop index of the slice:

```
name_of_list[start_index:stop_index]
```

However there is a catch, Pet Peeve number 2 (aka `pet_peeve[1]`), always slice **_one index past the one you want!_**

Here is an example of what is going on

credit: https://www.learntowish.com/python-strings/

Note you can leave left or right side empty to either start from the beginning, or go all the way to the end

```
test_list=[1,2,3,4]
test_list[1:3] # outputs [2,3]
test_list[:-1] # outputs [1,2,3]
test_list[:] # outputs all of the
```

Pet Peeve 3: Copy lists properly

If you would like to make a copy of a list dont just assign that list a new variable name, use the `[:]` or the `.copy()` method. just using:

```
list_copy = list_original
```

is just giving a second nickname ot list original. If you change the copy, then you change the original and vice versa. These two options work:

```
list_copy = list_original[:]
list_copy2 = list_original.copy()
```