1 Strings

	So you want to...	Code
Get substrings	Get n-th character in a string	`string[n]`
	Get the characters between position n and position m (not including m) in a string	`string[n:m]`
	Get first n characters (not including n) in a string	`string[:n]`
	Get the last character in a string	`string[-1]`
	Get the last n characters in a string	`string[-n:]`
Find/count substrings	Find the first position where as substring occurs (-1 if substring doesn't exist in string)	`string.find(substr)`
	Count the number of times a substring occurs	`string.count(substr)`
Concatenate	Concatenate two strings	`new_string = string_1 + string_2`
	Concatenate a list of strings	`new_string = ''.join(list_of_strings)`
	Concatenate a list of strings, separated by a space	`new_string = ' '.join(list_of_strings)`
Split	Split a strings into pieces based on some substring delimiter (delimiter not included in pieces)	`list_of_substrs = string.split(delim)`
Modify	Change the character at the n-th position to, say, `'Z'`	`string = string[:n] + 'Z' + string[n+1:]`
	Replace all occurrences of substring A in a string with substring B	`string = string.replace(substr_A, substr_B)`
	Delete all occurrences of substring A	`string = string.replace(substr_A, '')`

2 Looping

So you want to...	Code
Repeat for n number of times	`for i in range(n):` `  #do some stuff`
Do something for each element in a list	`for item in my_list:` `  #do some stuff to item`
Repeat until some condition (logical statement with true false value) is no longer met	`while cond_A:` `  #do some stuff`

3 Lists

So you want to...	Code
Build a list of integers from n to m (not including m) counting by k numbers	`range(n, m, k)`
Get the length of a list	`len(my_list)`

Build a list from scratch	`my_list = []`
	`my_list.append(new_item_1)`
	`...`

Getting first, last, or which ever in between items	*(exactly like working with strings)*
Modify the n-th item in the list	`my_list[n] = new_item`
Concatenate two lists	`new_list = list_1 + list_2`

Filter the list for items that satisfy some condition	Ex:
	`sub_list = []`
	`for item in my_list:`
	`  if item < 1:`
	`  sub_list.append(item)`

Filter the list for items that satisfy some condition (list comprehension style)	Ex: `sub_list = [item for item in my_list if item < 1]`

4 numpy

	So you want to...	Code
Make an array	Make an (2D) array from a list of (list of) values	`np.array(my_list)`
	Load tabular data from a file, whose values are separated by some character delimiter, into any array	`np.loadtxt(file_name, delimiter=delim_char, [optional things, like skip rows or columns])`
Compute with Arrays (1D)	Sum up the values in a array	`my_array.sum()`
	Sum up the values in a array	`my_array.sum()`
	Return the minimum value in an array	`my_array.min()`
	Return the mean of values in an array	`my_array.mean()`
Get Array Values (2D)	Get the value at the n-th row and m-th column	`my_array[n, m]`
	Get all values between rows n_1 and n_2 (not including n_2) and columns m_1 and m_2 (not including m_2 *as a 2D array*	`my_array[n_1:n_2, m_1:m_2]`
	Get the values in the n-row *as an 1D array*	`my_array[n, :]`
	Get the values in the m-column *as an 1D array*	`my_array[:, m]`
Filter the Array (2D)	Get values satisfying some condition	Ex: `my_array[ my_array < 1]`
	Get rows with entries that satisfy some condition	Ex: `my_array[ my_array[:, 0] < 1]`
	Get values that satisfy multiple conditions	Ex: `my_array[ (my_array[:, 0] < 1) & (my_array[:, 0] > 0) ]`
	Get values that satisfy *one of* multiple conditions	Ex: `my_array[ (my_array[:, 0] < 1) | (my_array[:, 0] > 0) ]`

5 matplotlib

So you want to...	Code

Make a grid with n rows and m columns of subplots	`fig = plt.figure()`
	
	`axes_1 = fig.add_subplot(n, m, 1)`
	`#do some plotting on the first axes...`
	
	`axes_2 = fig.add_subplot(n, m, 2)`
	`#do some plotting on the second axes...`
	`...`

Add a 3D subplot	`axes_k = fig.add_subplot(n, m, k, projection='3d')`
Adding a scatter plot to a set of axes	`axes.scatter(x_array, y_array)`
Adding a curve (given by a set of coordinates on the curve) to a set of axes	`axes.plot(x_array, y_array)`
Adding a histogram of an 1D array of values to a set of axes	`axes.hist(value_array)`
Adding color to your graph	`axes.scatter(x_array, y_array, c='red')`

6 beautifulsoup

So you want to...	Code
Read an html file at some URL	`page = urllib.urlopen("some_url").read()`
Parse an html file (turn it into parse tree)	`soup = BeautifulSoup(page, "lxml")`
Turn the parse tree into a formatted string	`soup.prettify()`
Get the first instance of a tag in the parse tree	`soup.tag`
Get all the text displayed on a page	`soup.get_text()`
Get the tag called "child_tag" from its parent	`parent.child_tag`
Get all the content from a tag	`tag.contents`
Get the string from a tag	`tag.string`
Get a "list" of all child tags of a tag	`tag.children`
Get all tags named "tag"	`soup.find_all('tag')`