Python List Comprehension

prompt	command	result
Concept	variable = [* * * result * * * for element in entity if condition]	* * entity could be range(), list, any iterable value...
Multiply each number	lib = [4,8,2,4,0,3] double_nums = [num * 2 for num in lib]	>>> print(double_nums) [8, 16, 4, 8, 0, 6]
POW each number	lib = [4,8,2,4,0,3] pow_nums = [pow (x, 2) for x in lib]	>>> print(pow_nums) [16, 64, 4, 16, 0, 9]

Reverse a list	one = ['a', 'b', 'c', 'd', 'e']
	two = one[::-1]
	or
	three = ['a', 'b', 'c', 'd', 'e'][::-1]
		>>> print(two)
		['e', 'd', 'c', 'b', 'a']
		>>> print(three)
		['e', 'd', 'c', 'b', 'a']

Traverse a list: every second value from indexes 1 to 6	one = ['a', 'b', 'c', 'd', 'e', 'f', 'g'] result = one[2:6:2] or result = [x for x in values[2:6:2]]	>>> print(result) ['c', 'e']
Operations with strings	names = ['Bob', 'Mike', 'John'] new_list = ["Hi, " + name for name in names] or new_list = [f 'Hi, {name}' for name in names]	>>> print(new_list) ['Hi, Bob', 'Hi, Mike', 'Hi, John']
String call of the first char	names = ['Bob', 'Mike', 'John', 'Jerry'] new_list = [x[0] for x in names]	>>> print(new_list) ['B', 'M', 'J', 'J']
Length of a string	names = ['Bob', 'Mike', 'John', 'Jerry'] lengths = [len(x) for x in names]	>>> print(lengths) [3, 4, 4, 5]

Unique values only	values = ['h',1,'b','b',4,'1','a',4]
	option_1 = list({x for x in values})
	or
	option_2 = list(set(values))
	or
	option_3 = [x for x in set(values)]
	or
	option_4 = []
	[option_4.append(x) for x in values if x not in option_4]
		>>> print(option_1)
		[1, 'h', 4, 'a', 'b', '1']
		>>> print(option_2)
		[1, 'h', 4, 'a', 'b', '1']
		>>> print(option_3)
		[1, 'h', 4, 'a', 'b', '1']
		>>> print(option_4)
		['h', 1, 'b', 4, '1', 'a']

Common values	one = ['a', 1, 'b', 'b', 4, '1'] two = ['h', '1', 1, 'a', 'j', '1'] common = [x for x in one if x in two]	>>> print(common) ['a', 1, '1']
Unite two lists	a = [5,1,6] b = [3,2,4] united = [x for y in [a, b] for x in y] or united = [x for x in a + b]	>>> print(united) [5, 1, 6, 3, 2, 4]

Create nested list	`one = ['Jack', 'Brit', 'Lucas', 'Ben']`
	`two = [10, 15, 4, 6]`
	`nl = [[name, age] for name, age in zip(one, two)]`
	or
	`nl = [[one[i], two[i]] for i in range(len(one))]`
		`>>> print(nl)`
		`[['Jack', 10], ['Brit', 15], ['Lucas', 4], ['Ben', 6]]`

Nested list sum	`nl = [[4, 8], [15, 2], [23, 42]]`
	`sum = [x + y for x, y in nl]`
	or
	`sum = [x + y for (x, y) in nl]`
		`>>> print(sum)`
		`[12, 31, 65]`

Nested list check	`nl = [[4, 8], [15, 2], [23, 42]]`
	`check = [x > y for x, y in nl]`
	or
	`check = [x > y for (x, y) in nl]`
		`>>> print(check)`
		`[False, True, False]`

Sum integers two lists	`a = [5, 1, 6]`
	`b = [3, 2, 4]`
	`new = [x + y for x, y in zip(a, b)]`
		`>>> print(new)`
		`[8, 3, 10]`

Conditional comprehension I, ternery operator	`one = [1, 2, 3, 4, 5, 6, 7]` `new = [x if x % 2 == 0 else x * 2 for x in one]`	`>>> print(new)` `[2, 2, 6, 4, 10, 6, 14]`
Conditional comprehension II	`a = [1, 2, 3, 4, 5, 6, 7, 8, 9]` `b = [x for x in a if x > 5 and x % 2 == 0]`	`>>> print(b)` `[6, 8]`
Multiple Condition comprehension I	`sent = 'it is I, Kai, Jack, and Brit'` `c = [x for x in sent.split() if x[0].isupper() and len(x) > 1 if ',' not in x]`	`>>> print(c)` `['Brit']`
Multiple Condition comprehension II	`all_clients = [{'name': 'Jack', 'age': 10, 'balance': 100}, {'name': 'Brit', 'age': 15, 'balance': 200}, {'name': 'Lucas', 'age': 4, 'balance': 300}, {'name': 'Ben', 'age': 6, 'balance': 400}]` `checked = [x['name'] for x in all_clients if x['balance'] >= 300 or x['age'] > 20]`	`>>> print(checked)` `['Lucas', 'Ben']`
Opposite boolean	`booleans = [True, False, True]` `result = [not x for x in booleans]`	`>>> print(result)` `[False, True, False]`
Check for value I	`names = ['Bob', 'Mike', 'John', 'Jerry']` `check = [x == 'John' for x in names]`	`>>> print(check)` `[False, False, True, False]`
Check for value II	`lib = [4, 8, 2, 4]` `check = [x > 3 for x in lib]`	`>>> print(check)` `[True, True, False, True]`
Search for value index	`names = ['Bob', 'Mike', 'John', 'Jerry', 'John']` `check = [i for i, x in enumerate(names) if x == 'John']`	`>>> print(check)` `[2, 4]`