⌨️

Lists, Conditionals, Indexing, For Loops, and List Comprehensions Cheat Sheet

Some Definitions:

- packages: a set of functions and tools
- objects: a collection of data
- functions: tools or routines to do a specific action
- methods: a function attached to an object
- attributes: properties of an object

Lists:

- a list in python is a type of object, meaning that it has methods associated with it (.append, .insert)
- to make a list, use square brackets

```
my_list = [1, 2, 3, 4, 5]
```

- elements in a list are separated by commas and the contents of a list can be either ints, floats, or strings

Indexing with Lists:

- in python, counting starts at 0
- the first element in a list (my_list) is at index 0 (the number 1 in this case)
- in order to grab different sections of a list, use a colon

```
my_list[0:2]
-> 1, 2
```

- the number after the colon (the stopping place) is NOT inclusive
- indexing: [start:stop:step]
- if you add a second color after your stopping point, that's how many you should count by (python automatically assumes it's 1)

```
my_list[0:4:2]
-> 2, 4
```

- you can also index from the end (starts at -1)

Conditionals:

- these are conditional statements (if, elif, else)
- conditionals act like little gates in python

```
x = 2
if x > 0:
  print("hello")
-> hello
```

- you can use "and" and "or" with conditionals as well
  - the conditional will run if at least 1 condition is met when using "or"
  - the conditional will run if both conditions are met when using "and"
- when there are only two conditions (see below), you use if and else

```
x = 2
if x < 1:
  print("hi")
else:
  print("not less than 1")
-> not less than 1
```

- if there are multiple conditions that you are testing, use elifs

```
x = 5
if x == 1:
  print("hi")
elif x > 8:
  print("greater than 8")
elif x < 3:
  print("less than 3")
elif x > 2 and x < 4:
  print("greater than 2 and less than 4")
else:
  print("helloooo")
-> greater than 8
```

For Loops:

- for loops are used to cycle through all of the elements in a list or object

```
my_list = [1, 2, 3]
for x in my_list:
  print("hi")
-> hi
hi
hi
```

- depending what you put under the for loop, the code will do that thing (say +2 if it's a number), to each item

List Comprehensions:

- this condenses a for loop into all one line

```
flow = [flow_list[day] for day in range(len(day_list)) if day_list[day] > 5]
```

- the "flow_list[day]" is what you want at the end of the for loop
- "for day in range(len(day_list))" is the for loop part
- "if day_list[day] > 5" is the if statement that is in the for loop
- the list comprehension above would look like the following below

```
for day in range(len(day_list)):
  if day_list[day] > 5:
    print(flow_list[day])
```

- it just condenses a for loop, helping to minimize the number of lines of code
    - however, it can be more difficult to understand