List Comprehensions and Lambda Functions: Takeaways

Syntax

WORKING WITH JSON FILES

- Open a JSON data set from a file to Python objects:

```
f = open('filename.json')

json.load(f)
```

- Convert JSON data from a string to Python objects:

```
json.loads(json_string)
```

- Convert JSON data stored in Python objects to string form:

```
json.dumps(json_obj)
```

LIST COMPREHENSIONS

- Converting a for loop to a list comprehension:
    - Using a for loop:

```
letters=['a', 'b', 'c', 'd']

caps=[]

for l in letters:

    caps.append(l.upper())
```

    - Using a List comprehension:

```
caps = [l.upper() for l in letters]
```

- Common list comprehension patterns:
    - Transforming a list

```
ints = [25, 14, 13, 84, 43, 6, 77, 56]

doubled_ints = [i * 2 for i in ints]
```

    - Creating test data

```
tenths = [i/10 for i in range(5)]
```

    - Reducing a list

```
big_ints = [i for i in ints if i >= 50]
```

LAMBDA FUNCTIONS

- Converting a definition to a lambda function:
    - Defining a function:

```
def double(x):

    return x * 2
```

    - Defining a lambda function:

```
run_function(function=lambda x: x * 2)
```

THE TERNARY OPERATOR

- Create a one-line version of an if/else statement:

```
"val_1 is bigger" if val_1 > val_2 else "val_1 is not bigger"
```

Concepts

- JSON is a language independent format for storying structured data.
    - In Python, it can be represented by a series of nested lists, dictionaries, strings, and numeric objects.
- A list comprehension provides a concise way of creating lists using a single line of code, where:
    - You start with an iterable object
    - Optionally Transform the items in the iterable object
    - Optionally reduce the items in the iterable object using an if statement
    - Create a new list

- Lambda functions can be defined in a single line, which lets you define a function at the time you need it.
- The ternary operator can be used to replace an if/else statement with a single line.

Resources

- Official JSON specification
- Python Documentation: JSON Module
- Python Documentation: List Comprehensions
- Python Documentation: Lambda Functions