Lambda Expressions, a.k.a. Anonymous Functions

Overview:

In Python, `lambda` is a keyword that allows you two write a one-off function (i.e. a function you know you will only need to use once) on a single line of code. This allows you to avoid the `def()` structure for defining and naming a function. It can also help you to avoid using a `for` loop. It is particularly handy to use with lists and with the Python keywords `filter` and `map`.

Syntax:

```
 lambda <0 or more inputs>: <expression for output>, <defined object to modify>
```

Quick Examples:

```
small_list = [1,2,3,4,5,6,7,8,9,10]
```

With `map`:

```
cubed_list = list(map(lambda x: x ** 3, small_list))
```

```
output = [1, 8, 27, 64, 125, 216, 343, 512, 729, 1000]
```

With `filter`:

```
odd_list = list(filter(lambda x: x % 2 != 0, small_list))
```

```
output = [1, 3, 5, 7, 9]
```

Common Uses:

- sorting lists
- modifying items in a list
- applying a formula to a column of data in a dataframe

References:

- Python.org documentation: https://docs.python.org/3/tutorial/controlflow.html?highlight=lambda
- Lutz, Mark. *Learning Python: Powerful Object-Oriented Programming*. (See chapter 19 – "Advanced Functions: lambda")
- Deitel, Harvey & Paul. *Intro to Python for Computer & Data Science.* (See chapter 5 – "Sequences: Lists and Tuples," section 5.14)
- Pandas.pydata.org documentation (review this if you're interested in incorporating lambdas in your Pandas usage) https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.apply.html