Monroe Township Library Coding Bootcamp

6/11 Class Notes

* Zen of Python & Pythonic coding
* Exception handling (errors)
* Try, except, finally
* Docstrings
* List Comprehensions
* Unpacking

**Pythonic Coding:**

* There is a little ‘poem’ called the Zen of Python that you can access by including import this somewhere in your program
  + Essentially, the idea is that you don’t simply want to write code that works, you should write code that is clear, readable, and understandable to other people
* Pythonic coding stresses writing code that is organized and compartmentalized (like breaking up individual tasks into functions), handling potential exceptions, and clearly documenting your code
  + Writing less lines of code is not always the best option if it sacrifices clarity
* You can check the PEP8 at <pep8.org> which is a set of style guidelines for Python
  + This includes guidelines for naming variables, functions, etc. to be used to keep Python code uniform and more readable for other Python programmers

**Exception Handling:**

* Python has some built-in ways to anticipate and handle potential errors in your code
  + Usually these are used in cases involving user input, accessing files, or any instance where you’re working with unknown data
* Normally, if an exception (error) occurs in your program, it will stop running completely—using the try and except keywords, you can specify something else to happen instead
  + Code located within a try block will be attempted but if an exception is encountered, the code with the except block will run instead
* It is good practice to specify which exceptions you are handling, you can do this by including the name of the exception next to the except keyword
  + Using except by itself will catch *all* exceptions
  + You can use multiple except statements to handle different types of exceptions in different ways
* The finally keyword allows you to add a block of code that will run at the end of your try block regardless of if any exceptions were encountered or not

**Docstrings:**

* Docstrings are used in Python to document functions so that others’ (or yourself) who are reading your code will be able to understand how they work
  + These go at the top of your function (under the def life) and use multiline comments
* Typically, docstrings should explain something about your function’s parameters (or arguments) including the expected types, and a little info about how they will work
* Your docstring should also include information about what your function will return, including the type (str, int, list, etc.)

**List Comprehensions:**

* List comprehensions are a more ‘Pythonic’ way of creating new lists and can condense something that would normally require a for loop into a single line of code
* When creating a new list variable, instead of directly passing values between the square brackets you can create a mini for loop—the values will be appended to the new list at each stage of the loop
  + You can also include a conditional statement at the end of the if your list comprehension to append values to your list *only* if it meets the condition
  + You can include an else statement to append something to your list if the conditional statement is False
    - In this case, the conditional statement should go at the start of your list comprehension before the for loop
* List comprehensions are useful, but can become confusing if they are too long—even if you can condense your code into less lines, you shouldn’t do that if it sacrifices clarity

**Unpacking:**

* Another Python ‘trick’ is that you can unpack values from a list or tuple and store them in new variables all on the same line
* Simply define new named variables separated by commas and set them equal to a list or tuple
* The variables will then take the of each index in the list/tuple
  + For example, the first variable would take the value of the 0 index of the list/tuple

**Project:** Secure Password Generator/Validator

**Check class files at** [**github.com/monroecoding**](https://github.com/monroecoding)

Chris DiFazio

Adult Services Librarian

cdifazio@monroetpl.org