Zelle 3e Chapter 6 Highlights

- The naïve structure for a program is to put all code into main(). That is a good practice only for very small programs.
- The better practice for larger programs is to break the code up into main() and an appropriate number of sub-functions.
- main() becomes the **main** or orchestrating function.
- **Refactoring** is changing a program that has already been written to make it more readable, testable, or maintainable without changing its function.
- **Factoring** is organizing your code **as you write it** so that it is easy to read, easy to test, and easy to maintain. Then, less need for refactoring.
- The functions in well-factored programs have **high cohesion** and **low coupling**.
- Functions that have high cohesion contain code that is all about the same thing.
- Functions that have low coupling know as little as possible about the details of other functions. They know the call interface of functions that they call. They know what values those functions return. These details make up the *APIs* of the called functions.

Zelle 3e Chapter 6 Demos

- Demonstrate how to suppress trivial duplicate code warnings in PyCharm.
- Demonstrate use cases for factoring code out of higher-level functions and into sub-functions:
  - Functions as **bags of code**
  - Functions as **reusable parts** (DRY)
  - Functions as **parameterized tools**
  - Functions as **hiding places for assumptions** (even DRY-er)
- Demonstrate function that returns more than one value.
- Demonstrate function that uses a keyword parameter.