Working with Multiple Files in Python

- ✦ As your project grows, you will need to split your code into modules.
- ✦ A **module** is a Python file that contains definitions and statements.
- ✦ Each Python file is considered a module.
- ✦ You can import these modules into your main program and into other modules to have access to the elements defined within them, while still keeping your program organized and compartmentalized.
- ✦ When you import a module, Python creates an object representing that module in memory.
- ✦ There are multiple ways to import a module. You can import all the elements it defines or import specific elements (e.g. variables, functions, and classes).
- ✦ Import statement are typically written at the top of the file.

Import Statements (General Syntax)

These are different alternatives for writing import statements. The name of the module is represented as <module> and the imported element is represented as <element>.

Import Statement	To Access the Element
import <module>	<module>.<element>
from <module> import <element>	<element>
from <module> import *	<element>
import <module> as **<name>**	**<name>**.<element>

Import Statements (Examples)

If you need to import the foo() function from a custom module named my_module, these are some alternatives:

Import Statement	To Call foo()
import my_module	my_module.foo()
from my_module import foo()	foo()
from my_module import *	foo()
import my_module as **module_alias**	**module_alias**.foo()