The interactive Python interpreter provides the most basic way to execute Python code. However, all of the defined variables, functions, classes, etc., are lost when a programmer closes the interpreter. Thus, a programmer will typically write Python code in a file, and then pass that file as input to the interpreter. Such a file is called a ***script***.

A solution is to use a ***module***, which is a file containing Python code that can be imported and used by scripts, other modules, or the interactive interpreter. To ***import*** a module means to execute the code contained by the module and make the definitions within that module available for use by the importing program.

A module's filename should end with ".py"; otherwise, the interpreter will not be able to import the module.

*Good practice is to place import statements at the top of a file.*

A module being required by another program is often called a ***dependency***.

A ***built-in module*** comes pre-installed with Python

For simple programs, a module might be placed in the same directory. Larger projects might contain tens or hundreds of modules or use third-party modules located in different directories. In such cases, a programmer might set the environment variable PYTHONPATH in the operating system.

 a user can set the value of PYTHONPATH permanently through the control panel, or temporarily on a single instance of a command terminal (cmd.exe) using the command set PYTHONPATH="c:\dir1;c:\other\directory".