```
"""
Python Modules
Summary By: Varpe K.M.
Reference: https://www.programiz.com/python-programming/modules

"""
"""
What are modules in Python?

Modules refer to a file containing Python statements and definitions.

A file containing Python code,
for e.g.: example.py, is called a module and
its module name would be example.

We use modules to break down large programs into small manageable
and organized files.
Furthermore, modules provide reusability of code.

We can define our most used functions in a module
and import it, instead of copying their definitions into different programs.

Let us create a module.
Type the following and save it as example.py.
"""
# Python Module example

# This will be Module File 1 to be calledandusedin another File

def add(a, b):
   """This program adds two
   numbers and return the result"""
   result = a + b
   return result
"""
Here, we have defined a function add() inside a module named example.
The function takes in two numbers and returns their sum.
"""

#############################################################


"""
How to import modules in Python?

We can import the definitions inside a module to another module or
the interactive interpreter in Python.

We use the import keyword to do this.
To import our previously defined module example
we type the following in the Python prompt.
"""

# This willbe Module Test File 2 which will test by calling module
 # through import

import AddExample
"""
This does not enter the names of the functions defined
in example directly in the current symbol table.
It only enters the module name example there.
```

```
Using the module name we can access the function using the dot . operator.
For example:
"""
sum=AddExample.add(4,5.5)
print("Addition using Module is=>",sum)

"""

Python has a ton of standard modules available.

You can check out the full list of
Python standard modules and what they are for.
These files are in the Lib directory inside the location
where you installed Python.

Standard modules can be imported the same way as we import
our user-defined modules.

There are various ways to import modules. They are listed as follows.
Python import statement

We can import a module using import statement
and access the definitions inside it using
the dot operator as described above. Here is an example.
"""

################################################################################

"""
There are various ways to import modules.
They are listed as follows.

Python import statement

We can import a module using import statement
and access the definitions inside it
using the dot operator as described above.
Here is an example.
"""

# import statement example
# to import standard module math

import math
print("The value of pi is", math.pi)

"""
Import with renaming

We can import a module by renaming it as follows. """
# import module by renaming it

import math as m

print("The value of pi is", m.pi)

"""
We have renamed the math module as m.
This can save us typing time in some cases.

Note that the name math is not recognized in our scope.
```

```
Hence, math.pi is invalid, m.pi is the correct implementation.
"""
print("The value of pi is", math.pi) #NameError: name 'math' is not defined

"""
Python from...import statement

We can import specific names from a module without
importing the module as a whole.
Here is an example. """

# import only pi from math module

from math import pi
print("The value of pi is", pi)
"""
We imported only the attribute pi from the module.

In such case we don't use the dot operator.
We could have imported multiple attributes as follows.
"""
from math import pi, e
print("pi=",pi)
print("e=",e)

"""
Import all names

We can import all names(definitions)
from a module using the following construct.
"""
# import all names from the standard module math

from math import *
print("The value of pi is", pi)

"""
Python Module Search Path

While importing a module,
Python looks at several places.
Interpreter first looks for a built-in module then (if not found)
 into a list of directories defined in sys.path.
 The search is in this order.

    The current directory.
    PYTHONPATH (an environment variable with a list of directory).
    The installation-dependent default directory.

"""
import sys
print(sys.path)
```