SWCON104
Web & Python Programming

Modules

Department of Software Convergence

경희대학교
KYUNG HEE UNIVERSITY

Today

- Modules

[Textbook]
Practical Programming
(An Introduction to Computer Science Using Python),
by Paul Gries, Jennifer Campbell, Jason Montojo,
The Pragmatic Bookshelf, 2017

Practice

- Practice_08_Modules

Modules

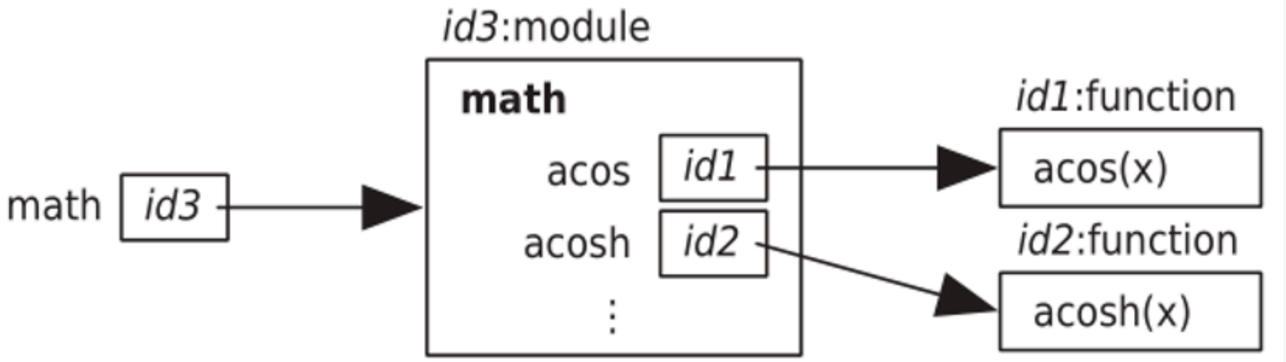
- Mathematicians don't prove every theorem from scratch.
- They build their proofs on the truths their predecessors have already established.

- Programmers don't write all of a program alone.
- They make use of the many lines of code that other programmers have written before.
- It's very common and more productive.

Import modules

```
>>> type(math)
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module
>
    type(math)
NameError: name 'math' is not defined
```

```
>>> import math
>>> type(math)
<class 'module'>
```




```
>>> help(math)
Help on built-in module math:

NAME
    math

DESCRIPTION
    This module is always available.  It
provides access to the
    mathematical functions defined by th
e C standard.

FUNCTIONS
    acos(...)
        acos(x)

        Return the arc cosine (measured
in radians) of x.

    acosh(...)
        acosh(x)

        Return the inverse hyperbolic co
sine of x.
```

How to use those functions?

- The dot(.) is an operator, just like + and **

1) Look up the object that the variable to the left of the dot refers to.

2) In that object, find the name that occurs to the right of the dot.

```
>>> sqrt(9)
Traceback (most recent call last):
  File "<pyshell#5>", line 1, in <module
>
    sqrt(9)
NameError: name 'sqrt' is not defined
>>> math.sqrt(9)
3.0
```

Variables imported from modules

- It is a bad idea to change the value of a variable defined within the module (usually meant to be a constant value)However, it is possible in Python.

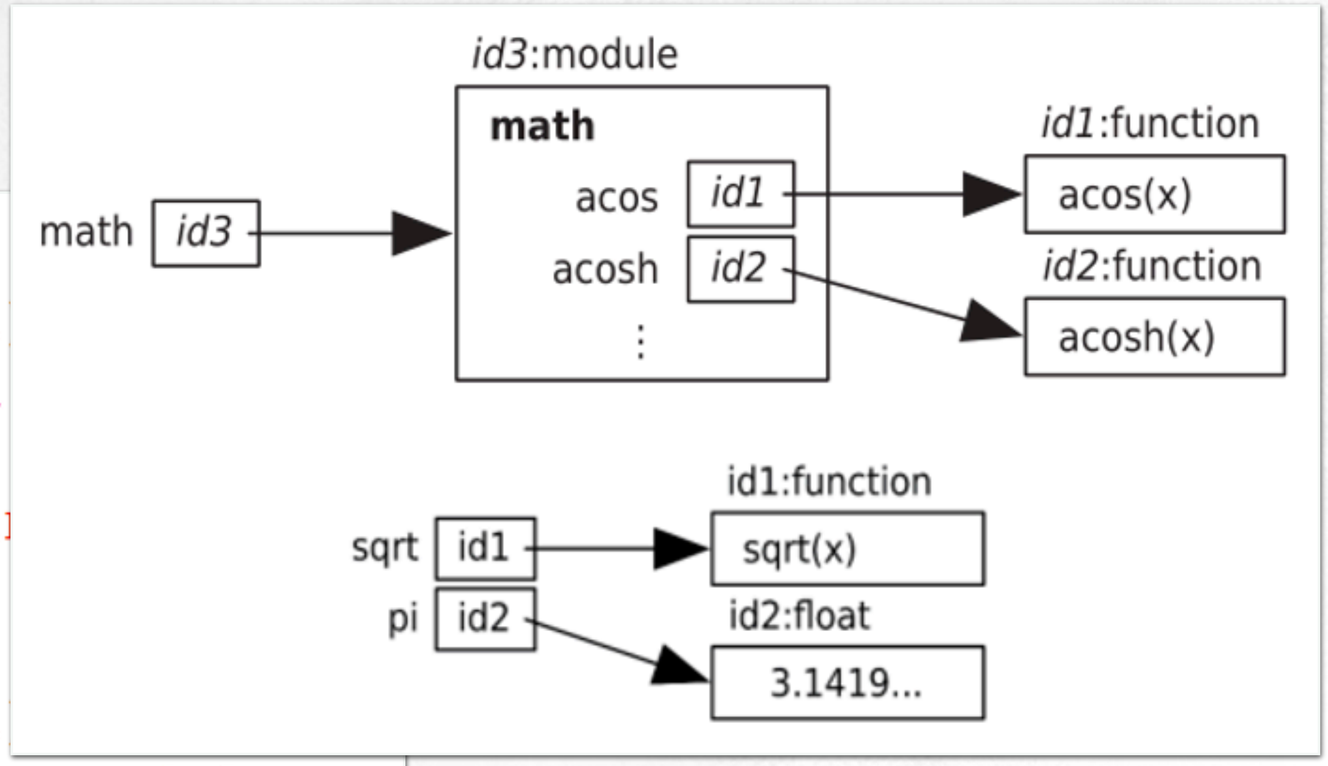
```
>>> import math
>>> math.pi
3.141592653589793
>>> radius = 5
>>> area = math.pi * radius **2
>>> area
78.53981633974483
>>> math.pi = 3
>>> area = math.pi * radius **2
>>> area
75
```

Don’t do this!!

To avoid using the dot

```
>>> pi
Traceback (most recent call last
  File "<pyshell#0>", line 1, in
    pi
NameError: name 'pi' is not defi
>>> import math
>>> pi
Traceback (most recent call last
  File "<pyshell#2>", line 1, in <module>
    pi
NameError: name 'pi' is not defined
>>> math.pi
3.141592653589793
>>> from math import pi, sqrt
>>> pi
3.141592653589793
>>> sqrt(9)
3.0
>>> from math import *
>>>
```

Usually not a good idea



Defining your own modules

temperature.py

```
def convert_to_celsius(fahrenheit):
    """ (number) -> float

    Return the number of celsius degree
    equivalent to fahrenheit degrees

    >>> convert_to_celsius(212)
    100.0
    """

    return (fahrenheit - 32)*5/9

def convert_to_fahrenheit(celsius):
    """ (number) -> float

    Return the number of fahrenheit de
    equivalent to celsius degrees

    >>> convert_to_celsius(100)
    212.0
    """

    return celsius*1.8 + 32
```

```
>>> import temperature
>>> convert_to_celsius(212)
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    convert_to_celsius(212)
NameError: name 'convert_to_celsius' is not defined
>>> temperature.convert_to_celsius(212)
100.0
>>> temperature.convert_to_fahrenheit(100)
212.0
```

Be careful with the directory of temperature.py (location)

imp.reload

exp.py

```
print("this is experiment")
```

`imp` — Access the import internals

Source code: Lib/imp.py

Deprecated since version 3.4, will be removed in version 3.12: The `imp` module is deprecated in favor of `importlib`.

`importlib.reload(module)`

Reload a previously imported *module*. The argument must be a module object, so it must have been successfully imported before. This is useful if you have edited the module source file using an external editor and want to try out the new version without leaving the Python interpreter. The return value is the module object (which can be different if re-importing causes a different object to be placed in `sys.modules`).

```
>>> import exp
this is experiment
>>> import exp
>>>
>>> import imp
>>> imp.reload(exp)
this is experiment
<module 'exp' from 'C:\\Users\\jiyoung\\AppData
\\Local\\Programs\\Python\\Python35\\exp.py'>
>>>
```

__name__

exp.py

```
print("this is experiment")
print("Name is", __name__)
```

```
>>> __name__
'__main__'
>>> exp.__name__
'exp'
>>>
== RESTART: C:/Users/jiyoung/AppData/Loc
al/Programs/Python/Python35/exp.py ==
this is experiment
Name is __main__
>>> import exp
this is experiment
Name is exp
>>>
```

Import math

```
>>> import math
>>> a = 0
>>> b = 30
>>> c = 45
>>> d = 60
>>> e = 90
>>> math.sin(b)
>>> math.sin(math.radians(b))
>>> math.degrees(math.asin(0.5))
>>> math.pi
>>> math.inf
>>> math.e
>>> math.exp(1)
>>> math.log(math.e)
>>> math.log(math.exp(2))
>>> math.log10(10.0)
>>> math.log10(100.0)
```

Import random

```
>>> import random
>>> x1 = random.random()
# generates a random floating-point number in range [0,1)
>>> x2 = random.uniform(a,b)
# generates a random floating-point number in range [a,b)
>>> x3 = random.randrange(stop)
# chooses an integer in the range [0,stop)
>>> x4 = random.randrange(start,stop)
# chooses an integer in the range [start,stop)
>>> x5 = random.randrange(start,stop,step)
# chooses an integer in the range [start,start+step, start+2*step,…,stop)
>>> x5 = random.randint(start,stop)
# chooses an integer in the range [start, stop] including both end points
```

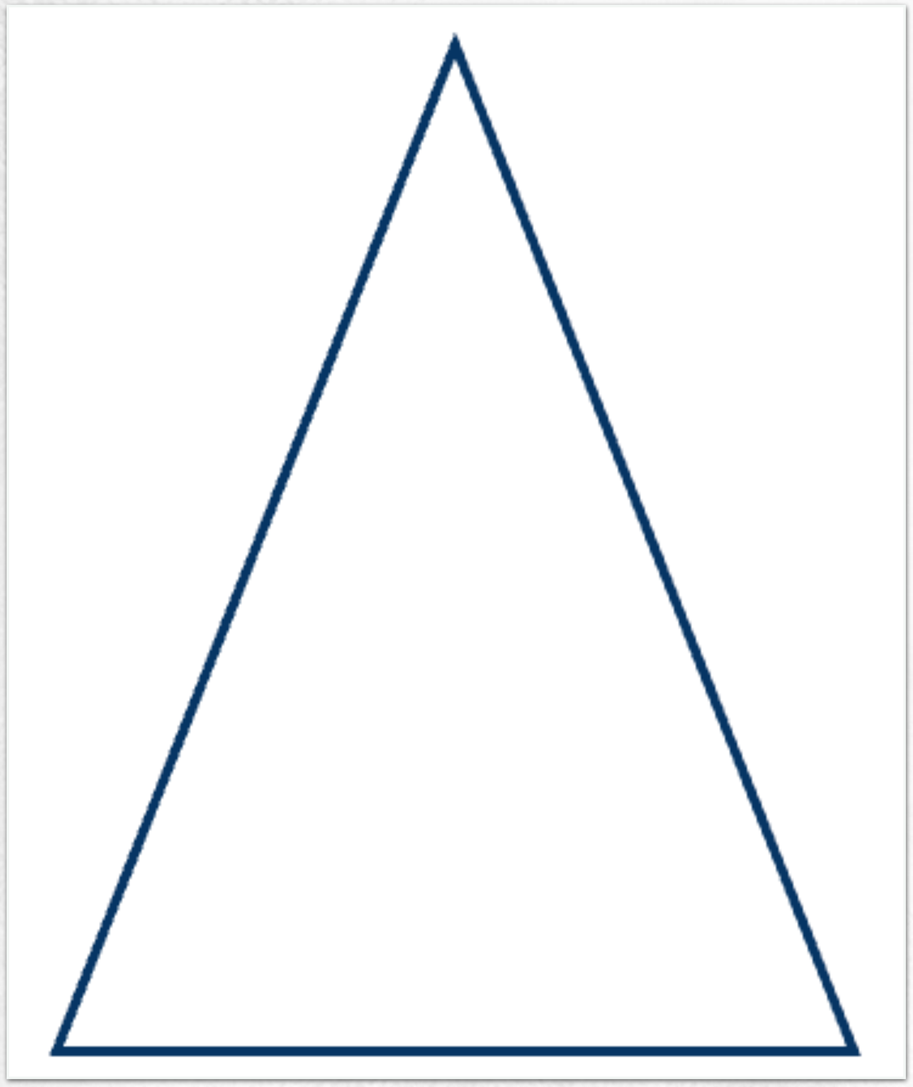
Graphic exercise

```
>>> import turtle
>>> t = turtle.Pen()
>>> t.forward(50)
>>> t.left(90)
>>> t.forward(50)
>>> t.left(90)
>>> t.forward(50)
>>> t.left(90)
>>> t.forward(50)
>>> t.left(90)
```

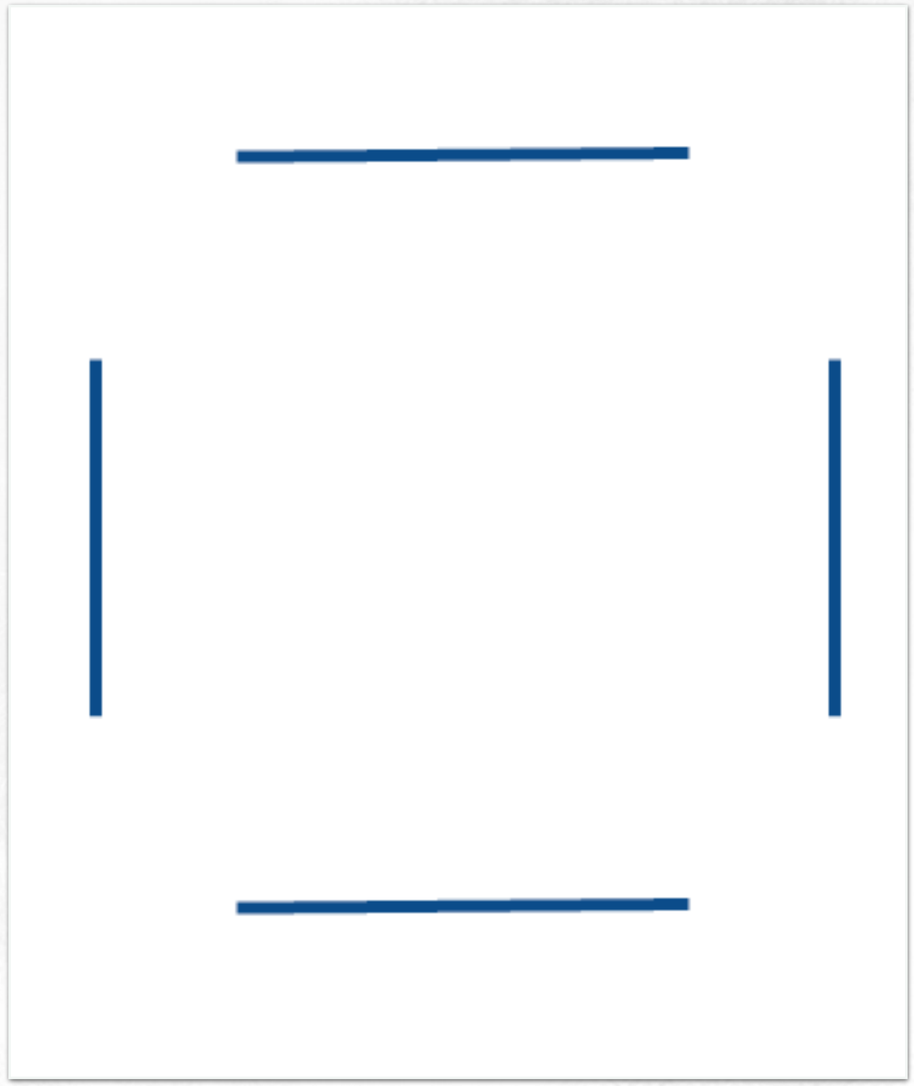
```
>>> t.reset()
>>> t.backward(100)
>>> t.up()
>>> t.right(90)
>>> t.forward(20)
>>> t.left(90)
>>> t.down()
>>> t.forward(100)

>>> t.clear()
```

Graphic exercise

- Draw a isosceles triangle
  (이등변삼각형)



- Draw a box with open corners
  (size is not important)



Summary

- A module is a kind of object, which can contain functions and other variables.
- A module is a collection of functions and variables defined within a single file.
- Math module and random module are useful
- You can make your own module

Thank you

경희대학교
KYUNG HEE UNIVERSITY