Python Essentials 2: Module 1 - Modules, Packages, and PIP

Jumat, 4 Juni 2021

In this module, you will learn about:

- importing and using Python modules;
- using some of the most useful Python standard library modules;
- constructing and using Python packages;
- PIP (Python Installation Package) and how to use it to install and uninstall ready-to-use packages from PyPI.

So what is module? The Python Tutorial defines it as **a file containing Python definitions and statements**, which can be later imported and used when necessary.

Importing module

To make module usable, you must **import** it (think of it like of taking a book off the shelf). Importing a module is done by an instruction named `import`. Note: `import` is also a keyword (with all the consequences of this fact).

In [1]:
```
import math
```

Kita bisa meng-import lebih dari satu modul sekaligus.

In [2]:
```
import math
import sys
```

In [3]:
```
import math, sys
```

Contoh penggunaan entitas pada modul.

In [4]:
```
import math
print(math.sin(math.pi/2)) # pi/2 rad = 90 derajat; sin(90 deg) = 1
```

```
1.0
```

In [10]:
```
import math

def sin(x):
    if 2 * x == pi: # 2 * pi / 2 = pi -> pi == pi -> True
        return 0.99999999
    else:
        return None

pi = 3.14

print(sin(pi/2))
print(math.sin(math.pi/2))
print(math.cos(math.pi/2))
```

```
0.99999999
1.0
6.123233995736766e-17
```

Cara untuk meng-import **sebagian** entitas pada suatu modul (tidak meng-import seluruh entitas pada modul).

In [11]:
```
from math import sin, pi

print(sin(pi/2))
print(math.cos(pi/2))
```

```
1.0
6.123233995736766e-17
```

In [13]:
```
from math import sin, pi

print(sin(pi / 2))

pi = 3.14

def sin(x):
    if 2 * x == pi:
        return 0.99999999
    else:
        return None

print(sin(pi / 2))
```

```
1.0
0.99999999
```

Cara meng-import seluruh entitas pada modul.

In [15]:
```
from math import *

sin(pi/2)

e = 10
```

Out[15]:
```
1.0
```

Cara meng-import dengan penggunaan alias.

In [18]:
```
import math as m

print(m.sin(m.pi/2))
```

```
1.0
```

In [19]:
```
from math import pi as PI, sin as sine

print(sine(PI/2))
```

```
1.0
```

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Working with standard modules (math, random, platform)

The `dir()` function returns an **alphabetically sorted list** containing all entities' names available in the module indentified by a name passed to the function as an argument.

In [21]:
```
import math

for name in dir(math):
    print(name, end="\t")
```

```
__doc__ __loader__      __name__        __package__     __spec__        acos    acosh
asin    asinh   atan    atan2   atanh   ceil    comb    copysign        cos     cosh
degrees dist    e       erf     erfc    exp     expm1   fabs    factorial       floor
fmod    frexp   fsum    gamma   gcd     hypot   inf     isclose isfinite        isinf
isnan   isqrt   ldexp   lgamma  log     log10   log1p   log2    modf    nan     perm
pi      pow     prod    radians remainder       sin     sinh    sqrt    tan     tanh
tau     trunc
```

In [20]:
```
import math
dir(math)
```

Out[20]:
```
['__doc__',
 '__loader__',
 '__name__',
 '__package__',
 '__spec__',
 'acos',
 'acosh',
 'asin',
 'asinh',
 'atan',
 'atan2',
 'atanh',
 'ceil',
 'comb',
 'copysign',
 'cos',
 'cosh',
 'degrees',
 'dist',
 'e',
 'erf',
 'erfc',
 'exp',
 'expm1',
 'fabs',
 'factorial',
 'floor',
 'fmod',
 'frexp',
 'fsum',
 'gamma',
 'gcd',
 'hypot',
 'inf',
 'isclose',
 'isfinite',
 'isinf',
 'isnan',
 'isqrt',
 'ldexp',
 'lgamma',
 'log',
 'log10',
 'log1p',
 'log2',
 'modf',
 'nan',
 'perm',
 'pi',
 'pow',
 'prod',
 'radians',
 'remainder',
 'sin',
 'sinh',
 'sqrt',
 'tan',
 'tanh',
 'tau',
 'trunc']
```

Selected functions from the $math$ module

In [22]:
```
from math import pi, radians, degrees, sin, cos, tan, asin
# pi itu konstanta nilainya 3,14....
# radians itu function untuk mengkonversi sudut dari satuan degrees (derajat) menjadi
# degrees sebaliknya

ad = 90
ar = radians(ad) # 90 derajat dikonversi ke rad -> ar = pi/2 rad
ad = degrees(ar) # pi/2 rad dikonversi ke derajat -> ad = 90 derajat

print(ad == 90.) # True
print(ar == pi / 2.) # True
print(sin(ar) / cos(ar) == tan(ar)) # True
print(asin(sin(ar)) == ar) # True
```

```
True
True
True
True
```

In [23]:
```
from math import e, exp, log
# e itu natural number (konstanta); e = 2,73...
# exp itu function eksponensial (e pangkat)
# log itu function untuk logaritma -> log dengan basis e, atau nama lainnya ln

print(pow(e, 1) == exp(log(e)))
print(pow(2, 2) == exp(2 * log(2)))
print(log(e, e) == exp(0))
```

```
True
True
True
```

In [24]:
```
from math import ceil, floor, trunc

x = 1.4
y = 2.6

print(floor(x), floor(y)) # 1.4, 2.6 -> 1, 2
print(floor(-x), floor(-y)) # -1.4, -2.6 -> -2, -3
print(ceil(x), ceil(y))
print(ceil(-x), ceil(-y))
print(trunc(x), trunc(y))
print(trunc(-x), trunc(-y))
```

```
1 2
-2 -3
2 3
-1 -2
1 2
-1 -2
```

Selected functions from the r and om module

In [28]:
```
from random import random

for i in range(5):
    print(random())
```

```
0.9386324362192736
0.859345449377014
0.9175625735961483
0.573963401034516
0.054288105649778595
```

In [45]:
```
from random import random, seed

seed(5)

for i in range(5):
    print(random())
```

```
0.5739411879281008
0.013114189588902203
0.21672980046384815
0.2794823660111103
0.9163453718085519
```

In [120…
```
from random import randrange, randint

# print(randrange(10000), end=' ')
# print(randrange(0, 1), end=' ')
# print(randrange(1, 10, 2), end=' ')
print(randint(0, 2))
```

```
1
```

In [123…
```
from random import randint

for i in range(10):
    print(randint(1, 10), end=',')
```

```
8,3,1,2,10,9,7,1,4,10,
```

In [127…
```
from random import choice, sample

my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

print(choice(my_list))
print(sample(my_list, 5))
print(sample(my_list, 10))
```

```
9
[9, 10, 6, 1, 3]
[10, 1, 5, 3, 8, 7, 4, 2, 6, 9]
```

Selected functions from $platf$ or m module

In [133…
```
from platform import platform

print(platform())
print(platform(1))
print(platform(0, 1))
```

```
Windows-10-10.0.19041-SP0
Windows-10-10.0.19041-SP0
Windows-10
```

In [134…
```
from platform import machine

print(machine())
```

```
AMD64
```

In [135…
```
from platform import processor

print(processor())
```

```
Intel64 Family 6 Model 142 Stepping 9, GenuineIntel
```

In [136…
```
from platform import system

print(system())
```

```
Windows
```

In [137…
```
from platform import version

print(version())
```

```
10.0.19041
```

In [138…
```
from platform import python_implementation, python_version_tuple

print(python_implementation())

for atr in python_version_tuple():
    print(atr)
```

```
CPython
3
8
5
```

Python Module Index

We have only covered the basics of Python modules here. Python's modules make up their own universe, in which Python itself is only a galaxy, and we would venture to say that exploring the depths of these modules can take significantly more time than getting acquainted with "pure" Python.

You can read about all standard Python modules here: https://docs.python.org/3/py-modindex.html.

Modules and Packages

Let's summarize some important issues:

- a **module is a kind of container filled with functions** - you can pack as many functions as you want into one module and distribute it across the world;
- of course, it's generally a good idea not to mix functions with different application areas within one module (just like in a library - nobody expects scientific works to be put among comic books), so group your functions carefully and name the module containing them in a clear and intuitive way (e.g., don't give the name `arcade_games` to a module containing functions intended to partition and format hard disks)
- making many modules may cause a little mess - sooner or later you'll want to **group your modules** exactly in the same way as you've previously grouped functions - is there a more general container than a module?
- yes, there is - it's a **package**; in the world of modules, a package plays a similar role to a folder/directory in the world of files.

Membuat Modul

Pertama, buat 2 file dengan nama `aritmatika.py` dan `main.py`, kemudian simpan pada folder yang sama. Lalu tuliskan potongan kode berikut pada file `aritmatika.py`.

In [ ]:
```
def tambah(a, b):
    return a + b

def kurang(a, b):
    return a - b

def kali(a, b):
    return a * b

def bagi(a, b):
    return a / b
```

Tuliskan potongan kode berikut pada `main.py`.

In [ ]:
```
import aritmatika as ar

a = ar.tambah(4, 5)
b = ar.kurang(4, 5)
c = ar.kali(4, 5)
d = ar.bagi(4, 5)

print(a)
print(b)
print(c)
print(d)
```

Membuat Package

1. Buatlah folder dengan nama `LATIHAN_PYTHON`.
2. Di dalam folder `LATIHAN_PYTHON` buatlah folder dengan nama `latihan_package`, file `main.py`, dan file `__init__.py`.
3. Di dalam folder `latihan_package` buatlah file `alpha.py` dan `beta.py`.

In [ ]:
```
# alpha.py

def alphaSatu():
    print("alphaSatu")

def alphaDua():
    print("alphaDua")
```

In [ ]:
```
# beta.py

def betaSatu():
    print("betaSatu")

def betaDua():
    print("betaDua")
```

In [ ]:
```
# main.py

import latihan_package.alpha as a
import latihan_package.beta as b

a.alphaSatu()
b.betaDua()
```

Mengakses package dari folder yang berbeda

In [ ]:
```
import os

os.chdir("C:\\Program Files (x86)") # directory bentuknya string
os.chdir(r"C:\Program Files (x86)")

import latihan_package.alpha as a, latihan_package.beta as b

a.alphaSatu()
b.alphaDua()
```