Python Basic

모듈과 패키지(Modules and Packages)

모듈의 생성

- 모듈의 생성
  - 변수, 함수, 클래스 등을 스크립트 파일로 작성한 후 다른 스크립트에서 불러와 해당 요소를 사용할 수 있음.
  - 자주 사용하는 기능을 미리 만들어 편의성 및 재사용성 추구.
  - 불러올 모듈 스크립트 파일은 기본적으로 실행할 스크립트 파일과 같은 디렉토리에 존재해야 함.

```
# util.py
INCH = 2.54 #1 inch = 2.54 cm

def calcsum(n):
    sum = 0
    for num in range(n + 1):
        sum += sum
    return sum
```

```
# utiltest.py
import util

print("1inch =", util.INCH)
print("~10 =", util.calcsum(10))
```

모듈의 경로

- 모듈의 경로
  - sys 모듈의 path 리스트에 입력되어 있는 디렉토리에 존재하는 모듈만 불러올 수 있음.
  - 스크립트 파일을 실행할 때 실행 파일이 위치한 디렉토리가 자동으로 path에 추가됨.
  - 다른 디렉토리에 존재하는 모듈을 불러오기 위해서 path에 디렉토리를 추가하여 사용 가능.

```
import sys
print(sys.path)

sys.path.append("불러올 모듈이 존재하는 경로")
print(sys.path)
```

__name__ 변수

- __name__ 변수
  - 실행 중인 모듈의 이름을 담는 특수 변수.
  - 파이썬은 따로 실행 함수가 없고 모든 모듈이 실행 가능.
  - 어떤 모듈이 직접 실행될 경우 __name__의 값은 "__main__".
  - 어떤 모듈이 다른 모듈에 의해 실행될 경우는 해당 모듈의 이름.

```
# util.py
INCH = 2.54

def calcsum(n):
    sum = 0
    for num in range(n + 1):
        sum += num
    return sum

if __name__ == "__main__":
    print("인치:", INCH)
    print("10 이하 정수의 합:", calcsum(10))
print("__name__의 값:", __name__)
```

패키지

- 패키지
  - 모듈의 집합.
  - 디렉토리로 계층을 구성함.
  - 기본적으로 *을 이용해 모든 모듈을 가져오는 것은 안 됨.

```
C:
└Python Workspace
   └testpack
      └dir1
         └mod1.py
      └dir2
         └mod2.py
      └dir3
         └mod3.py
```

```
import sys
sys.path.append(r"C:\python workspace")

import testpack.dir1.mod1
testpack.dir1.mod1.func1()

from testpack.dir2 import mod2
mod2.func2()

from testpack.dir3 import *
# mod3.func3()
```

__init__.py

- __init__.py
  - __init__.py 파일이 존재하는 디렉토리는 해당 디렉토리가 패키지의 일부임을 알려줌.
  - python 3.3 버전부터 __init__.py 파일을 생략 가능.
  - __all__ 변수를 이용해 'from 디렉토리 import *' 조작 가능.

```
C:
└Python Workspace
  └testpack
    └dir1
      └mod1.py
    └dir2
      └mod2.py
    └dir3
      └__init__.py
      └mod3.py
```

```
# __init__.py
__all__ = ["mod3"]

print(__all__ , "has been imported")
```

```
import sys
sys.path.append(r"C:\python workspace")

from testpack.dir3 import *
mod3.func3()
```

__pycache__

- __pycache__
  - 실행된 모듈을 컴파일하여 저장하는 디렉토리.
  - 컴파일한 파일을 .pyc 파일로 저장.
  - 파이썬에서 어떤 스크립트가 모듈을 불러올 때 컴파일한 파일을 캐시에 저장하고 코드를 미리 해석해 두어 로드 속도를 높임.

상대 경로

- 상대 경로
  - 패키지 내 어떤 모듈에서 다른 모듈을 부를 때 쓰는 문법.
  - 패키지명.디렉토리명.모듈명으로 불러오기 가능.
  - '..': 상위 디렉토리를 지정.
  - '.': 현재 디렉토리를 지정.

```
# mod3.py
from testpack.dir2 import mod2
from ..dir1 import mod1

def func3():
    print("mod3_func3()")
    mod2.func2()
    mod1.func1()
```

dir() 함수

- dir() 함수
  - 모듈을 전달받아 해당 모듈이 가진 요소들의 목록을 출력.

```
import math
dir(math)
```