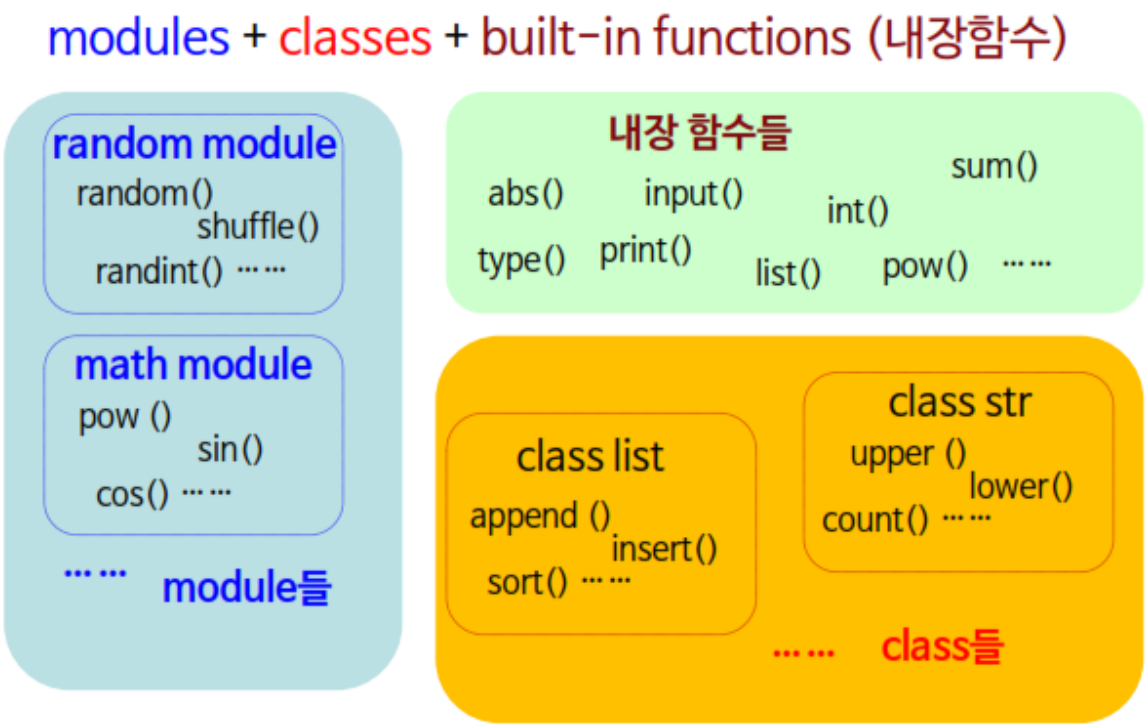
단국대학교
SW중심대학

창의적 사고와 코딩

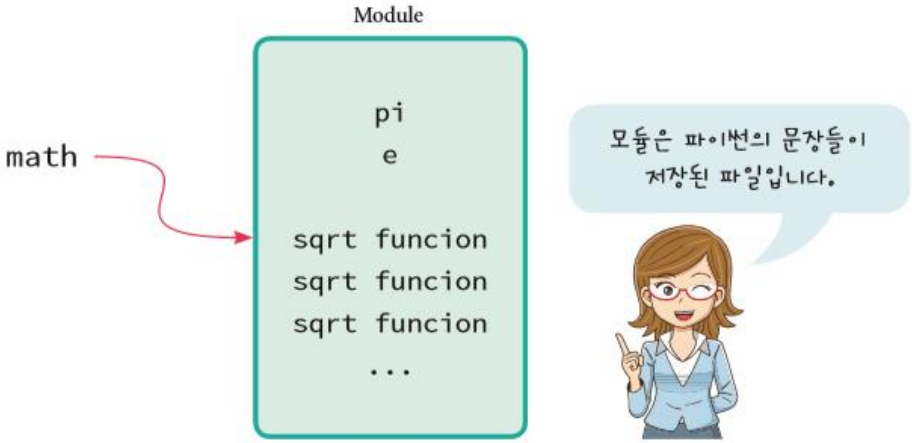
Lecture 5-3장. 모듈

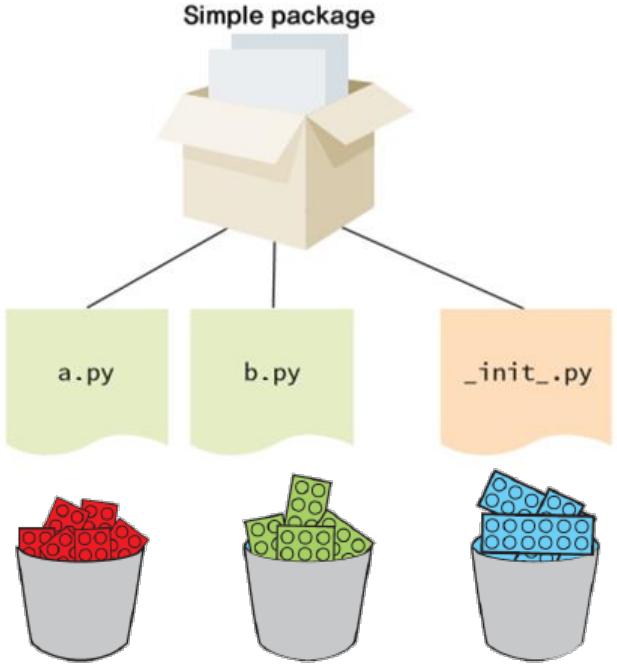
파이썬 구성 요소


modules + classes + built-in functions (내장함수)
random module
random()
shuffle()
randint() ……
math module
pow ()
sin()
cos() ……
…… module들
내장 함수들
abs()
input()
int()
sum()
type()
print()
list()
pow()
……
class list
append ()
insert()
sort() ……
class str
upper ()
lower()
count() ……
…… class들

모듈

- **파이썬에서 모듈(module)이란 함수나 변수들을 모아 놓은 파일이다.**
  - 함수가 집 짓기 블록과 같다면, 모듈은 집 짓기 블록을 담는 통과 같음
  - 원하는 만큼 통에다 블록을 담을 수 있고, 그런 통을 여러 개 만들 수도 있음 ➔ 비슷한 종류의 함수를 한 모듈 안에 모아서 사용





모듈(module)

- 코드들을 한 단위로 묶어 사용할 수 있게 하는 하나의 단위
- 각 모듈은 하드디스크상에 별도의 파일로 되어 구성되어 있음
- 모듈의 종류
  - 표준 모듈: 파이썬 패키지 안에 포함된 모듈
    - https://docs.python.org/ko/3/py-modindex.html
  - 사용자 모듈: 사용자가 만드는 모듈
  - 써드 파티(third party ,3rd Party) 모듈: 개인이 만들어서 제공하는 모듈
- 모듈 사용의 장점
  - 코드의 재사용성
  - 서로 다른 모듈에 같은 이름의 메소드가 있어도 충돌이 생기지 않음
  - 파일의 크기가 더 작아져서 코드에서 원하는 것을 찾기가 쉬어짐

모듈 사용하기

- 모듈은 반드시 import 후에 사용할 수 있다.

```
>>> import math
>>> math.pow(2,3) # math내의 pow 함수
8.0
>>> math.pi
3.141592653589793
>>> dir(math)
['__doc__', '__loader__', '__name__', '__package__', '__spec__', 'acos', 'acosh', 'asi
n', 'asinh', 'atan', 'atan2', 'atanh', 'ceil', 'copysign', 'cos', 'cosh', 'degrees', 'e', 'e
rf', 'erfc', 'exp', 'expm1', 'fabs', 'factorial', 'floor', 'fmod', 'frexp', 'fsum', 'gamma
', 'gcd', 'hypot', 'inf', 'isclose', 'isfinite', 'isinf', 'isnan', 'ldexp', 'lgamma', 'log', 'lo
g10', 'log1p', 'log2', 'modf', 'nan', 'pi', 'pow', 'radians', 'remainder', 'sin', 'sinh',
'sqrt', 'tan', 'tanh', 'tau', 'trunc']
>>>
```

모듈 사용하기

모듈 import 방법

① import <모듈명>

```
>>> import math
>>> math.pi   # math 모듈에 있는 pi 속성
3.141592653589793
```

② from <모듈명> import <함수명>

```
>>> from math import pow
>>> pow(2,3)  #모듈이름 없이 함수명만으로 사용가능
8.0
>>> from math import pow, sqrt, trunc  # 여러함수 사용하는 경우
>>> sqrt(25)
5.0
>>> trunc(1.2345)
1
>>> pow(2,3)
8.0
```

```
>>> from math import *  # math 모듈에 있는 모든 함수를 함수이름만으로 사용하는 경우
>>> sin(pi/2)
1.0
```

모듈 사용하기

모듈 import 방법

③ import <모듈명> as <alias>
모듈명 대신 별칭(alias or nickname)을 모듈명으로 사용

```
>>> import math as m
>>> m.sqrt(25)
5.0
>>> m.sin(m.pi)
1.2246467991473532e-16
```

```
>>> import random
>>> print(random) # 모듈이 정상적으로 적재되었는지 확인
<module 'random' from 'C:₩₩Users₩₩yuni₩₩AppData₩₩Local₩₩Programs
₩₩Python₩₩Python38-32₩₩lib₩₩random.py'>
```

모듈 작성

- 데이터, 함수들로 구성된 파일을 만든다.
- 파일명이 모듈명이 된다.
  - 파일명: circle.py

circle.py - C:₩Users₩yuni₩Desktop₩circle.py (3.8.3)

File Edit Format Run Options Window Help

```
PI = 3.14159265358979   # 전역 상수

def Area(radius):
    return PI*radius*radius

def Circumference(radius):
    return 2*PI*radius
```

파일명: test.py

test.py - C:₩Users₩yuni₩Desktop₩test.py (3.8.3)

File Edit Format Run Options Window Help

```
import circle

radius = float(input('원의 반지름을 입력하시오: '))

print('원의 면적:', circle.Area(radius))
print('원의 둘레:', circle.Circumference(radius))
```

```
원의 반지름을 입력하시오: 10
원의 면적: 314.159265358979
원의 둘레: 62.8318530717958
```

- import 할 때 해당 py파일을 어디서 찾을까?
  - 먼저 현재 디렉토리에서 검색한 후 발견되지 않으면 sys.path 변수에 저장된 디렉토리에서 찾는다.

함수를 사용한 프로그램 설계

1. 문제를 한 번에 해결하려고 하지 말고 더 작은 크기의 문제들로 분해한다. 문제가 충분히 작아질 때까지 계속해서 분해한다.
2. 문제가 충분히 작아졌으면 각각의 문제를 함수로 작성한다.
3. 이들 함수들을 조립하면 최종 프로그램이 완성된다.

yunyi96@gmail.com

예제

```
def readList():
    nlist = []
    flag = True;
    while flag :
        number = int(input("숫자를 입력하시오: "))
        if number < 0:
            flag = False
        else :
            nlist.append(number)
    return nlist

def processList(nlist):
    nlist.sort()
    return nlist
```

```
def printList(nlist):
    for i in nlist:
        print("성적=", i)

def main():
    nlist = readList()
    processList(nlist)
    printList(nlist)

if __name__ == "__main__":
    main()
```

```
숫자를 입력하시오: 30
숫자를 입력하시오: 50
숫자를 입력하시오: 10
숫자를 입력하시오: 90
숫자를 입력하시오: 60
숫자를 입력하시오: -1
성적= 10
성적= 30
성적= 50
성적= 60
성적= 90
```

yunyi96@gmail.com

예제

- 모듈을 가져올 때 일반적으로는 출력을 생성하지 않음 ➔ 파일이 모듈로 로드 될 때와 독립형 스크립트로 실행 될 때를 구별
- if __name__ == "__main__": 문을 사용하면 모듈이나 스크립트로 사용가능하지만 스크립트로 실행되는 경우에만 fib 함수가 실행되어 출력된다.

fibo.py - C:₩test₩fibo.py (3.8.3)

File Edit Format Run Options Window Help

```
# 피보나치 수열 모듈
def fib(n):    # 피보나치 수열을 화면에 출력한다.
    a, b = 0, 1
    while b < n:
        print(b, end=' ')
        a, b = b, a+b
    print()

if __name__ == "__main__":
    import sys
    fib(int(sys.argv[1]))
```

파이썬 스크립트 실행시 인자값을 전달받아 실행

Windows Console에서 스크립트로 실행

```
C:₩Users₩yuni>python c:₩test₩fibo.py 50
1 1 2 3 5 8 13 21 34

C:₩Users₩yuni>python c:₩test₩fibo.py 100
1 1 2 3 5 8 13 21 34 55 89
```

모듈로 실행

fibo_ex.py - C:/test/fibo_ex.py (3.8.3)

File Edit Format Run Options Window Help

```
import fibo
```

모듈 안의 코드는 실행되지 않는다.

프로젝트 : 거북이 경주 게임

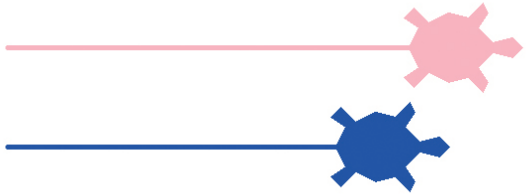
거북이 경주 게임 #1

- 거북이 2마리를 만들려면 다음과 같이 한다.

```
import turtle

t1 = turtle.Turtle() # 첫 번째 거북이
t2 = turtle.Turtle() # 두 번째 거북이
```

거북이 경주 게임 #2

- 거북이들을 구별하기 위하여 색상을 다르게 하고 모양도 다르게 하자.

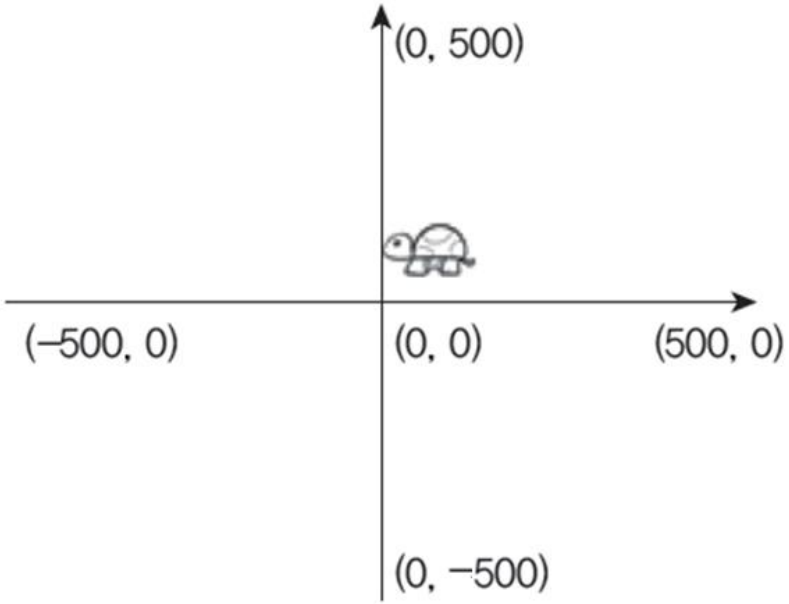


```
t1.color("pink")
t1.shape("turtle")
t1.shapesize(5)
t1.pensize(5)

t2.color("blue")
t2.shape("turtle")
t1.shapesize(5)
t2.pensize(5)
```

거북이 경주 게임 #3

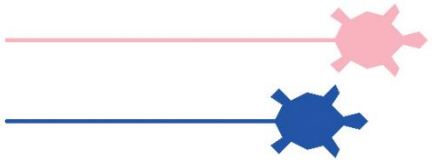
- 출발점에 세우기

```
t1.penup()
t1.goto(-300, 0)

t2.penup()
t2.goto(-300, -100)
```



거북이 경주 게임 #4

- 100번 정도 반복하면서 한 번 반복할 때마다 난수만큼 이동하도록 하자.



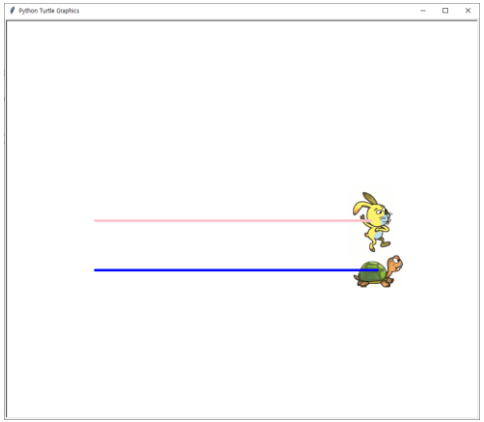
```
for i in range(100):                  # 100번 반복한다.
    t1.fd(random.randint(1, 10))      # 난수만큼 이동한다.
    t2.fd(random.randint(1, 10))      # 난수만큼 이동한다.
```

외부 이미지를 사용하자

```
import turtle # 터틀 그래픽 모듈을 불러온다.
import random # 난수 모듈을 불러온다.

screen = turtle.Screen()
image1 = "rabbit.gif"
image2 = "turtle.gif"
screen.addshape(image1)
screen.addshape(image2)

t1 = turtle.Turtle() # 첫 번째 거북이를 생성한다.
t1.shape(image1)

t2 = turtle.Turtle() # 두 번째 거북이를 생성한다.
t2.shape(image2)
```

Solution

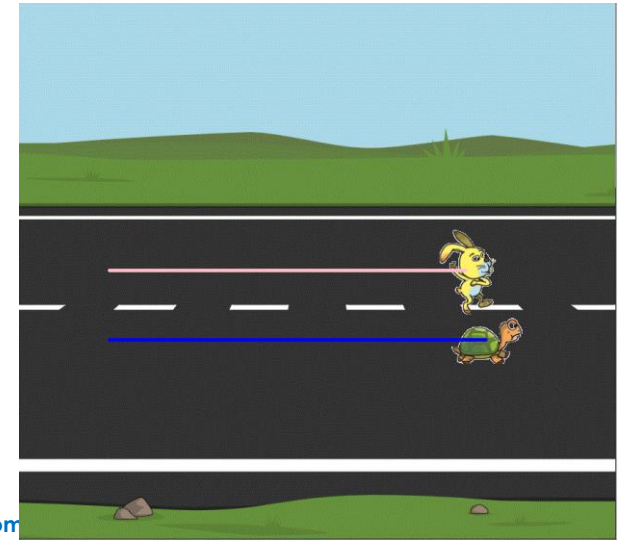
```
import random
import turtle

screen = turtle.Screen()
image1 = "rabbit.gif"
image2 = "turtle.gif"
screen.addshape(image1)
screen.addshape(image2)

t1 = turtle.Turtle() # 첫 번째 거북이
t1.shape(image1)
t1.color("pink")
t1.pensize(5)

t2 = turtle.Turtle() # 두 번째 거북이
t2.shape(image2)
t2.color("blue")
t2.pensize(5)

t1.up()
t1.goto(-300, 0)
t1.down()

t2.up()
t2.goto(-300, -100)
t2.down()

for i in range(100):                # 100번 반복한다.
    t1.fd(random.randint(1, 10))    # 난수만큼 이동한다.
    t2.fd(random.randint(1, 10))    # 난수만큼 이동한다.
```



배경 이미지 표시하기

- **배경 이미지 : gif 만 표시 가능**
- **screen.bgpic("파일명")** : 배경 이미지를 설정하거나 현재 배경 이미지의 이름을 반환

```
screen = turtle.Screen()
screen.bgpic("bg.gif")
```



```
>>> screen.bgpic("landscape.gif")
>>> screen.bgpic()
"landscape.gif"
```

yunyi96@gmail.com