모듈 활용 기초

python에는 기본적으로 제공되는 모듈들이 있습니다.

표준 라이브러리에서 제공되는 모듈을 확인해보세요!

여기 있는 모든 내용을 외울 필요도 없고, 이런 것이 있다만 확인해보세요 :)

우리가 사용했던 `random` 역시도 표준라이브러리에서 제공되고 있는 모듈이며, 난수를 발생시키는 모듈입니다.

In [2]:

```
# 로또 번호 추천을 해보세요!
import random
lotto = random.sample(range(1, 46), 6)
print(lotto)
```

```
[5, 36, 25, 34, 44, 30]
```

`import`

- 모듈을 활용하기 위해서는 반드시 `import`문을 통해 내장 모듈을 이름 공간으로 가져와야합니다.

In [3]:

```
import random
print(dir(random))
```

```
['BPF', 'LOG4', 'NV_MAGICCONST', 'RECIP_BPF', 'Random', 'SG_MAGICCONST', 'SystemRandom', 'TWOPI',
'_BuiltinMethodType', '_MethodType', '_Sequence', '_Set', '__all__', '__builtins__', '__cached__',
'__doc__', '__file__', '__loader__', '__name__', '__package__', '__spec__', '_acos', '_bisect', '_
ceil', '_cos', '_e', '_exp', '_inst', '_itertools', '_log', '_pi', '_random', '_sha512', '_sin',
'_sqrt', '_test', '_test_generator', '_urandom', '_warn', 'betavariate', 'choice', 'choices', 'exp
ovariate', 'gammavariate', 'gauss', 'getrandbits', 'getstate', 'lognormvariate', 'normalvariate',
'paretovariate', 'randint', 'random', 'randrange', 'sample', 'seed', 'setstate', 'shuffle',
'triangular', 'uniform', 'vonmisesvariate', 'weibullvariate']
```

- `import`는 다양한 방법으로 할 수 있습니다.

`from` *모듈명* `import` *어트리뷰트*

특정한 함수 혹은 어트리뷰트만 활용하고 싶을 때, 위와 같이 작성합니다.

In [4]:

```
# 우리가 beautifulsoup을 사용할 때 활용했던 코드를 작성해봅시다.
from bs4 import BeautifulSoup    # import bs4.beautifulSoup
# 바로 bs4라는 모듈에서 BeautifulSoup만 가져온 것이었습니다.
```

In [ ]:

```
# random 모듈 중에 sample을 바로 활용해봅시다.
```

In [7]:

```
# 이름공간에 현재 sample이 없습니다.
sample([1,2,3], 2)
```

Out[7]:

```
[3, 2]
```

In [6]:

```
from random import sample
```

`from` *모듈명* `import` `*`

해당하는 모듈 내의 모든 변수, 함수, 클래스를 가져옵니다.

In [8]:

```
from random import *
choice([1, 2, 3])
```

Out[8]:

```
3
```

`from` *모듈명* `import` *어트리뷰트* `as`

내가 지정하는 이름을 붙여 가져올 수 있습니다.

In [ ]:

```
from bs4 import beautifulSoup as bs
```

(번외) 모듈과 시작점

- vscode로 가서 실험 해보자.

In [ ]:

```
if __name__ == '__main__':
    print('This is main!')
```

숫자 관련 함수

이외에도 분수(frctions), 십진(decimal), 통계(statistics)등이 있습니다.

수학 관련 함수(math)

다음의 기본 함수는 `import` 없이 활용하였습니다.

`sum` , `max` , `min` , `abs` , `pow` , `round` , `divmod`

In [9]:

```
import math
```

- 활용할 수 있는 상수는 다음과 같습니다.

In [11]:

```
# 원주율(pi)
math.pi
```

Out[11]:

```
3.141592653589793
```

In [12]:

```
# 자연 상수(e)
math.e
```

Out[12]:

```
2.718281828459045
```

- 활용할 수 있는 연산 관련 함수는 다음과 같습니다.

함수	비고
math.ceil(x)	소수점 올림
math.floor(x)	소수점 내림
math.trunc(x)	소수점 버림
math.copysign(x, y)	y의 부호를 x에 적용한 값
math.fabs(x)	float 절대값 - 복소수 오류 발생
math.factorial(x)	팩토리얼 계산 값
math.fmod(x, y)	float 나머지 계산
math.fsum(iterable)	float 합
math.modf(x)	소수부 정수부 분리

In [15]:

```
# 올림
pi = 3.141592
math.ceil(pi)
```

Out[15]:

```
4
```

In [16]:

```
# 내림
math.floor(pi)
```

Out[16]:

```
3
```

In [17]:

```
# 버림
math.trunc(pi)
```

Out[17]:

```
3
```

In [18]:

```
# 내림과 버림은 음수에서 처리가 다르다.
math.floor(-pi)
```

Out[18]:

```
-4
```

In [19]:

```
math.trunc(-pi)
```

Out[19]:

```
-3
```

In [ ]:

```
# 프로그래밍에서 나눗셈은 음수로 하거나 양수로 하거나 두가지 상황이 있음.
# %는 정수를 fmod는 float
# 부호가 다른 경우 서로 다르게 출력함.
```

In [20]:

```
math.fmod(-5, 2)
```

Out[20]:

```
-1.0
```

In [21]:

```
-5 % 2
```

Out[21]:

```
1
```

- 로그, 지수 연산은 다음과 같습니다.

함수	비고
math.pow(x,y)	x의 y승 결과
math.sqrt(x)	x의 제곱근의 결과
math.exp(x)	e^x 결과
math.log(x[, base])	밑을 base로 하는 logx

In [22]:

```
# 제곱
math.pow(2, 5)
```

Out[22]:

```
32.0
```

In [23]:

```
# 제곱근
math.sqrt(8)
```

Out[23]:

```
2.8284271247461903
```

In [24]:

```
# e
math.exp(1)
```

Out[24]:

```
2.718281828459045
```

In [25]:

```
# 로그 계산
math.log(math.e)
```

Out[25]:

```
1.0
```

In [26]:

```
math.log(10, 10)
```

Out[26]:

```
1.0
```

- 삼각함수는 다음과 같습니다.

  sin, cos, tan
  asin, acos, atan,
  sinh, cosh, tanh,
  ashinh, acosh, atanh

In [27]:

```
math.sin(0)
```

Out[27]:

```
0.0
```

In [28]:

```
math.cos(0)
```

Out[28]:

```
1.0
```

난수 발생관련 함수(random)

In [29]:

```
import random
```

In [30]:

```
# sample과 choice를 각각 활용해봅시다.
random.choice(range(1,6))
```

Out[30]:

```
1
```

In [31]:

```
random.sample(range(1,46), 6)
```

Out[31]:

```
[34, 29, 32, 19, 33, 9]
```

In [32]:

```
# 난수 생성
random.random()
```

Out[32]:

```
0.36124220173740273
```

In [34]:

```
# 임의의 정수 반환
random.randint(1, 5)
```

Out[34]:

```
4
```

In [140]:

```
# 시드 설정 - 시드 설정을 하지 않으면 현재 시간을 기반으로 만든다.
random.seed(1)
```

In [143]:

```
# 시드 설정 후에 첫번째 값을 확인해보자
random.random()
```

Out[143]:

```
0.763774618976614
```

In [144]:

```
# 시퀀스 객체를 섞는다.
a = ['kim', 'kang', 'yu', 'choi', 'hwang']
# .shuffle()
random.shuffle(a)
print(a)
```

```
['choi', 'hwang', 'kang', 'kim', 'yu']
```

날짜 관련 모듈

datetime

In [176]:

```
# 1970년 1월 1일부터 1초씩 증가합니다.
# 오늘을 출력해봅시다.
import datetime
now = datetime.datetime.now()
print(now)
```

```
2019-01-10 13:41:11.934314
```

In [177]:

```
# 오늘을 출력하는 다른 방법도 있습니다.
now_2 = datetime.datetime.today()
print(now_2)
```

```
2019-01-10 13:41:40.014931
```

In [209]:

```
# UTC기준시도 출력가능합니다.
print(datetime.datetime.utcnow())
```

```
2019-01-10 04:43:13.313133
```

- 시간 형식지정

형식 지시자(directive)	의미
%y	연도표기(00~99)
%Y	연도표기(전체)
%b	월 이름(축약)
%B	월 이름(전체)
%m	월 숫자(01~12)
%d	일(01~31)
%H	24시간 기준(00~23)
%I	12시간 기준(01~12)
%M	분(00~59)
%S	초(00~61)
%p	오전/오후
%a	요일(축약)
%A	요일(전체)
%w	요일(숫자 : 일요일(0))
%j	1월 1일부터 누적 날짜

In [210]:

```
# 내가 원하는대로 예쁘게 출력해봅시다.
now.strftime('%Y' '%M' '%D' '%A')
```

Out[210]:

```
'20194101/10/19Thursday'
```

속성/메소드	내용
.year	년
.month	월
.day	일
.hour	시
.minute	분
.second	초
.weekday()	월요일을 0부터 6까지

In [211]:

```
# 년도
now.year
```

Out[211]:

```
2019
```

In [213]:

```
# 월요일 0부터
now.weekday()
```

Out[213]:

```
3
```

- 특정한 날짜 만들기

```
datetime.date(year, month, day, hour, minute, second, microsecond)
```

In [222]:

```
# 크리스마스를 만들어봅시다.
import datetime
christmas =datetime.datetime(2018, 12, 25)
print(christmas)
```

```
2018-12-25 00:00:00
```

In [215]:

```
# 예쁘게 출력해봅시다.
christmas.strftime('%Y %m %A %H:%M')
```

Out[215]:

```
'2018 12 Tuesday 00:00'
```

timedelta

```
from datetime import timedelta
```

In [216]:

```
from datetime import timedelta
```

In [217]:

```
# 활용해봅시다.
ago = timedelta(days=-3)
```

In [218]:

```
# 비교 및 연산이 가능합니다.
now + ago
```

Out[218]:

```
datetime.datetime(2019, 1, 7, 13, 41, 11, 934314)
```

In [219]:

```
# 오늘부터 1일일때, 100일 뒤는?
now + timedelta(days=100)
```

Out[219]:

```
datetime.datetime(2019, 4, 20, 13, 41, 11, 934314)
```

In [224]:

```
# 크리스마스부터 지금까지 얼마나 지났을까?
diff = christmas - now
str(diff)
```

Out[224]:

```
'-17 days, 10:18:48.065686'
```

In [225]:

```
# 초로 만들어봅시다.
diff_seconds = diff.total_seconds()
print(diff_seconds)
```

```
-1431671.934314
```

In [ ]:

```
# [실습] 아래에 초를 예쁘게 출력하는 함수를 만들어봅시다.
# 예) '10일 1시간 18분 51초 전'
def print_time_delta(seconds):
    # 여기에 코드를 작성하세요.

    return None
```

In [ ]:

```
print_time_delta(diff_seconds)
```

In [ ]:

```

```