6장. 모듈과 패키지

목 차

모듈(Module)

- **모듈(module)**
  - 함수나 변수 또는 클래스를 모아 놓은 **소스파일**로써, 이를 사용하는 다른 파일에서는 첫 부분에 [**import 모듈이름**]으로 선언한다.

모 듈	설 명
math	수학 계산과 관련된 모듈
datetime	날짜 및 시간과 관련된 모듈
time	시간과 관련된 모듈
calendar	달력을 보여주는 모듈
random	난수를 발생시키는 모듈
sys	변수와 함수를 직접 제어할 수 있게 해주는 모듈
threading	일정 주기마다 함수를 실행하는 기능을 가진 모듈이다.

math 모듈

◎ **math** 모듈

import math

math.ceil(2.54)	올림
math.floor(2.54)	내림
math.sqrt(4)	제곱근
math.factorial(5)	팩토리얼
math.pi	원주율

```
import math

print(math.pi)

# 올림
print(math.ceil(2.54)) #3

# 반올림
print(round(2.6))  #3

# 버림
print(math.floor(4.9))  #4

# 제곱근
print(math.sqrt(25))  #5
```

math 모듈

- 절대값 함수 구현하기

```
import math

# 절대값 알고리즘1 - 부호판단
def abs_sign(x):
    if x >= 0:
        return x
    else:
        return -x
```

```
# 절대값 알고리즘2 - (제곱 -> 제곱근)
def abs_square(a):
    b = a * a
    return math.sqrt(b)

print(abs_sign(3))
print(abs_sign(-4))
print()
print(abs_square(5))
print(abs_square(-4))
```

datetime 모듈

◎ **datetime** 모듈

- datetime.datetime.today() – 년, 월, 일, 시, 분, 초

```
2021-07-05 05:39:40.051344
2021년
7월
5일
5시
39분
40초
2021.07.05 05:39:40
2021년 7월 5일, 5시 39분 40초
```

```
import datetime

today = datetime.datetime.today()
print(today)
print("{}년".format(today.year))
print("{}월".format(today.month))
print("{}일".format(today.day))
print("{}시".format(today.hour))
print("{}분".format(today.minute))
print("{}초".format(today.second))
```

```
now = datetime.datetime.now()
print(now.strftime("%Y. %m. %d %H:%M:%S"))
print("{}년 {}월 {}일, {}시 {}분 {}초".format(now.year,
    now.month, now.day, now.hour, now.minute, now.second))
```

datetime 모듈

◎ 나이가 100세 되는 해의 연도 계산하기 프로그램

```
# 나이가 100살 되는 해의 연도 계산하기 프로그램

import datetime
today = datetime.datetime.today()
print(today.year)

age = int(input('나이 입력: '))

years_left = 100 - age
years_100 = today.year + years_left
print("100세 되는 해 :" , years_100)
```

datetime 모듈

◎ 지나온 날짜 계산하기

datetime.date(2021, 10, 26) – 날짜만 가져옴

```
♠ 지금까지 몇 일? ♠
개강 이후 30일이 지났습니다.
```

```python
import datetime

# date()함수는 날짜만 출력
print("♠ 지금까지 몇 일? ♠")
day1 = datetime.date(2021, 10, 26)
print(day1)

today = datetime.date.today()
print(today)
passedtime = today - day1
print("개강 이후 {0}일이 지났습니다.".format(passedtime.days))
```

calendar 모듈

◎ **calendar** 모듈

- **calendar.prcal(2021)** : 2021년의 달력을 표시
- **calendar.prmonth(2021, 6)** : 2021년 6월의 달력을 표시
- **calendar.weekday(2021, 10, 26)** : 날짜에 해당하는 요일 정보

```
>>> import calendar
>>> calendar.prcal(2021)
                                  2021

      January                   February
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
             1  2  3       1  2  3  4  5  6  7
 4  5  6  7  8  9 10       8  9 10 11 12 13 14
11 12 13 14 15 16 17      15 16 17 18 19 20 21
18 19 20 21 22 23 24      22 23 24 25 26 27 28
25 26 27 28 29 30 31

       April                     May
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
          1  2  3  4                     1  2
 5  6  7  8  9 10 11       3  4  5  6  7  8  9
12 13 14 15 16 17 18      10 11 12 13 14 15 16
19 20 21 22 23 24 25      17 18 19 20 21 22 23
26 27 28 29 30            24 25 26 27 28 29 30
                          31
```

6월1일은 화요일(1)이고, 30일까지 있음.

```
>>> calendar.monthrange(2021, 6)
(1, 30)
>>> calendar.weekday(2021, 6, 21)
0
```

6월 21일의 요일 0(월요일) – 0(일), 1(월), 2(화)…

datetime 모듈

◎ 날짜로 요일 알아내기

```
import datetime
import calendar

days = ["월", "화", "수", "목", "금", "토", "일"]

# 오늘의 요일
weekday = datetime.datetime.today().weekday()
print(weekday)          #3
print(days[weekday])    #목

# 특정한 날의 요일
theday = datetime.datetime(2021, 10, 26).weekday()
print(days[theday])

# 달력 출력하기
calendar.prmonth(2021, 10)
```

```
3
목
화
     October 2021
Mo Tu We Th Fr Sa Su
                 1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31
```

datetime 모듈

◎ 날짜로 요일 알아내기 – 함수로 정의하기

```
import datetime

def get_days(yyyy, mm, dd):
    days = ['월요일', '화요일', '수요일', '목요일', '금요일', '토요일', '일요일']
    x = datetime.date(yyyy, mm, dd).weekday()
    return days[x]

# 특정한 날의 요일
yyyy = 2021
mm = 10
dd = 21
weekday = get_days(yyyy, mm, dd)
print("{0}년 {1}월 {2}일은 {3}입니다.".format(yyyy, mm, dd, weekday))
```

time

◎ **time** 모듈

- time.time() 현재 시간을 실수 형태로 돌려주는 함수
- time.sleep(2) 일정한 시간 간격을 두고 루프를 실행할 수 있다.
- time.localtime() 연도, 월, 일, 시, 분, 초.. 형태

```
>>> import time
>>> time.time()
1586032691.8111842
>>> time.localtime()
time.struct_time(tm_year=2020, tm_mon=4, tm_mday=5,
tm_hour=5, tm_min=38, tm_sec=19, tm_wday=6, tm_yday
=96, tm_isdst=0)
>>> time.ctime()
'Sun Apr  5 05:38:47 2020'
>>> time.strftime('%x', time.localtime())
'04/05/20'
```

time 모듈

◎ **time** 모듈

- time.sleep(1) 일정한 시간 간격을 두고 루프를 실행할 수 있다.

```
>>> for i in range(10):
        print(i)
        time.sleep(1)
```

1초 기다리기(대기)

```
0
1
2
3
4
5
```

다른 언어에서는 1초를 1000으로 표기하고, 파이썬은 1초를 1로 표기함

time 모듈

- **속으로 20초를 세어 맞히는 프로그램**

1. "엔터를 누르고 20초를 셉니다 " 라는 문장이 뜨면 Enter를 누르고 속으로 20초를 센다.
2. 20초가 지났다고 생각되면 Enter를 누른다.
3. 실제 시간과 차이가 화면에 표시된다.

```
엔터를 누르고 20초를 셉니다.
20초 후에 다시 엔터를 누릅니다.
실제 시간 : 19.65초
차이 : 0.35초
```

```python
import time

input("엔터를 누르고 20초를 셉니다.")
start = time.time()

input("20초 후에 다시 엔터를 누릅니다.")
end = time.time()

et = end - start
print("실제 시간 : %.2f초" % et)
print("차이 : %.2f" % abs(et-20))
```

random 모듈

◎ **random** 모듈

- random.random() : 0.0에서 1.0사이의 실수 값 중에서 난수 값 발생
- random.randint(1, 10) : 1과 10사이의 정수중에서 난수 값 발생
- random.choice(a) : 리스트에서 무작위로 하나를 선택

```
>>> import random
>>> random.randint(1, 6)
1
>>> random.randint(1, 6)
5
>>> random.randint(1, 6)
5
>>> random.randint(1, 6)
3
>>> random.randint(1, 6)
5
>>> random.random()
0.4140764418205686
>>> random.random()
0.06228136629206793
```

```
>>> lis = ['딸기', '수박', '참외', '사과']
>>> random.choice(lis)
'사과'
>>> random.choice(lis)
'참외'
>>> random.choice(lis)
'수박'
>>> random.shuffle(lis)
>>> lis
['참외', '수박', '딸기', '사과']
>>> random.shuffle(lis)
>>> lis
['참외', '사과', '딸기', '수박']
>>> random.shuffle(lis)
>>> lis
['딸기', '참외', '수박', '사과']
```

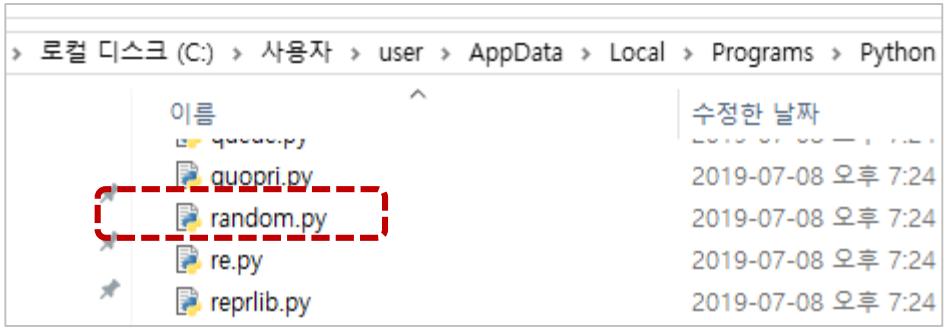
random 모듈

◎ **random 모듈**

```
import random
import math

print(random.random()) # 0 <= x < 1

# 주사위
dice = math.floor(random.random() * 6) + 1
print(dice)

dice2 = random.randint(1, 6) # 주사위
print(dice2)

lis = ["강아지", "고양이", "돼지", "말", "소"]

# 리스트에서 무작위로 1개 선택
print(random.choice(lis))

# 리스트의 항목을 무작위로 섞기
random.shuffle(lis)
print(lis)
```

random 모듈

- random 모듈(파일) 위치




```
def randint(self, a, b):
    """Return random integer in range [a, b], in
    """

    return self.randrange(a, b+1)


## -------------------- sequence methods  ------

def choice(self, seq):
    """Choose a random element from a non-empty
    # raises IndexError if seq is empty
    return seq[self._randbelow(len(seq))]
```

ramdom 모듈

- 활용 예제

```
# 랜덤하게 단어 추출하기
words = ['sky', 'earth', 'moon', 'flower', 'tree',
         'strawberry', 'grape', 'garlic','onion', 'potato']
word = random.choice(words)
print(word)

'''
rand = math.floor(random.random() * 6)
print(rand)
print(words[rand])
'''
```

```
# 주사위 10번 던지기
import random
import math

for x in range(1, 11):
    dice = random.randint(1, 6)
    print(dice, end=' ')
print()
```

ramdom 모듈

- **주사위 2개를 10번 던지기**
  - 두 눈의 합이 7이면 "Seven Thrown" 출력
  - 두 눈의 합이 11이면 "Eleven Thrown" 출력
  - 두 눈의 수가 같으면 "Double Thrown" 출력

☞ 실행 결과

```
6
3
10
10
7
Seven Thrown
7
Seven Thrown
11
Eleven Thrown
10
Double Thrown!!
3
9
```

```
import random

for i in range(10):
    dice1 = random.randint(1, 6)
    dice2 = random.randint(1, 6)
    total = dice1 + dice2
    print(total)
    if total == 7:
        print("Seven Thrown")
    if total == 11:
        print("Eleven Thrown")
    if dice1 == dice2:
        print("Double Thrown!!")
```

up_and_down 게임

❖ up_and_down 게임 (숫자를 추측해서 맞히는 게임)

- 게임이 시작되면 컴퓨터가 난수(정답)를 생성한다.
- 사용자의 추측값이 정답과 같으면 '정답!', 추측값이 정답보다 크면 "너무 커요!", 추측값이 정답보다 작으면 '너무 작아요!' 출력
- 총 횟수는 10회이며, 점수는 남은 횟수 x 10으로 한다.

```
맞혀보세요([1회] 1 ~ 100) -> 50
너무 작아요!
맞혀보세요([2회] 50 ~ 100) -> 70
너무 작아요!
맞혀보세요([3회] 70 ~ 100) -> 80
너무 커요!
맞혀보세요([4회] 70 ~ 80) -> 75
너무 커요!
맞혀보세요([5회] 70 ~ 75) -> 72
너무 작아요!
맞혀보세요([6회] 72 ~ 75) -> 73
정답!!
정답 : 73
점수 : 50점
```

숫자 추측 게임

```
import random

com = random.randint(1, 100)  # 난수
min_v = 1     # 범위 - 최소값
max_v = 100   # 범위 - 최대값
for i in range(0, 10):
    try:  #숫자가 아닌 문자 입력시 에러 처리
        guess = int(input("맞혀보세요([%d회] %d ~ %d) -> " % (i+1, min_v, max_v)))
```

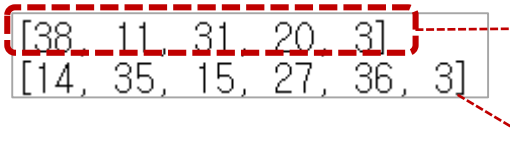
숫자 추측 게임

```
        if guess > 100 or guess < 1 :
            print("입력 범위가 아니에요. 다시 입력해 주세요: ")
        elif guess == com:
            print("정답!!")
            break
        elif guess > com:
            print("너무 커요!")
            max_v = guess
        else:
            print("너무 작아요!")
            min_v = guess

    except ValueError:
        print("유효한 숫자가 아닙니다. 다시 입력해 주세요")

print("정답 : %d" % guess)
print("점수 : %d점" % ((10 - i) * 10))
```

로또(lotto) 복권

- **로또(lotto) 복권 추첨 프로그램**

  로또 번호를 중복되지 않도록 생성하는 프로그램 만들기




```
import random

lotto = []
for i in range(6):
    n = random.randint(1, 45)
    if n not in lotto:
        lotto.append(n)

print(lotto)
```

```
import random

lotto = []
while len(lotto) < 6:
    print(len(lotto))
    n = random.randint(1, 45)
    if n not in lotto:
        lotto.append(n)

print(lotto)
```

튜플(tuple) 예제

- 튜플(tuple) 예제

```
import random

say1 = "자! 해보세요!"
say2 = "됐네요, 이 사람아"
say3 = "뭐라고? 다시 생각해보세요"
say4 = "모르니 두려운 것입니다."
say5 = "칠푼인가요?? 제 정신이 아니군요!"
say6 = "당신이라면 할 수 있어요!"
say7 = "해도 그만, 안 해도 그만, 아니겠어요?"
say8 = "맞아요, 당신은 올바른 선택을 했어요"

print("안녕하세요~ Magic 상담소입니다.");
question = input(
"""조언을 구하고 싶으면 질문을 입력하고
엔터 키를 누르세요""")
print("고민 중...\n" * 4)
```

```
choice = random.randint(1, 8)
if choice == 1:
    answer = say1
elif choice == 2:
    answer = say2
elif choice == 3:
    answer = say3
elif choice == 4:
    answer = say4
elif choice == 5:
    answer = say5
elif choice == 6:
    answer = say6
elif choice == 7:
    answer = say7
else:
    answer = say8

print(answer)
input("마치려면 엔터 키를 누르세요")
```

튜플(tuple)

- 튜플(tuple) 로 바꾸기

```
import random

answers = (
    "자! 해보세요!",
    "됐네요, 이 사람아",
    "뭐라고? 다시 생각해보세요",
    "모르니 두려운 것입니다.",
    "칠푼인가요?? 제 정신이 아니군요!",
    "당신이라면 할 수 있어요!",
    "해도 그만, 안 해도 그만, 아니겠어요?",
    "맞아요, 당신은 올바른 선택을 했어요"
    )

print("안녕하세요~ Magic 상담소입니다.");

question = input("조언을 구하고 싶으면 질문을 입력하고\n엔터 키를 누르세요.\n")

print("고민 중...\n" * 4)

choice = random.randint(0, 7)
print(answers[choice])
```

타자 연습 게임

- **영어 타자 연습 프로그램**

  게임 방법

  - 게임이 시작되면 영어 단어가 화면에 표시된다.
  - 사용자는 최대한 빠르고 정확하게 입력해야 한다.
  - 바르게 입력했으면 다음 문제로 넘어가고 "통과"를 출력한다.
  - 오타가 있으면 같은 단어가 한 번 더 나온다.
  - 타자 게임 시간을 측정한다.

타자 연습 게임

```
[타자 게임]준비되면 엔터!

-문제 1
sky
sky
통과!
-문제 2
garlic
garlic
통과!
-문제 3
grape
grape
통과!
-문제 4
garlic
ga
오타! 다시 도전!
-문제 4
garlic
garlic
통과!
-문제 5
tree
tree
통과!
-문제 6
garlic
garlic
통과!
-문제 7
earth
earth
통과!
-문제 8
earth
earth
통과!
-문제 9
flower
flower
통과!
-문제 10
onion
onion
통과!
타자 시간 : 33.62초
```

타자 연습 게임

```
import random
import time

word = ['sky', 'earth', 'moon', 'flower', 'tree',
        'strawberry', 'grape', 'garlic','onion', 'potato']
n = 1   # 문제 번호

print("[타자 게임]준비되면 엔터!")
input()
start = time.time()
```

```
while n < 11:
    print('-문제', n)
    q = random.choice(word)
    print(q)
    u = input()  # 사용자 입력
    if q == u:
        print('통과!')
        n += 1
    else:
        print('오타! 다시 도전!')

end = time.time()
et = end - start
print('타자 시간 : %.2f초' % et)
```

sys 모듈(Module)

◎ **sys** 모듈

- **sys.argv :** 명령 행에서 인수 전달하기

```
(pyweb) C:\pyworks\ch06>python sys_import.py cat 100 dog
['cat', '100', 'dog']
cat
100
dog
```

argv_test.py cat 100 dog

argv[0] argv[1] argv[2] argv[3]

```python
import sys

# 명령 행에서 인수를 전달함
args = sys.argv[1:]
print(args)

# 전체 요소 출력
for i in args:
    print(i)
```

sys 모듈(Module)

◎ sys 모듈

- sys.exit(O) : 파이썬 프로그램 종료

```
import sys

x = input('수를 입력하세요 : ')
n = int(x)

if n == 0:
    print('0으로 나눌 수 없습니다.')
    sys.exit(0)

num = 3 / n

print(num)
```

sys 모듈(Module)

◎ sys 모듈 예제

- 입력값 더하기

```
(pyweb) C:\pyworks\ch06>sys_import.py 1 2 3 4
['1', '2', '3', '4']
10
```

```
args = sys.argv[1:]
print(args)

total = 0
for i in args:
    total += int(i)

print(total)
```

threading 모듈

- **threading(쓰레딩)**
  - **threading.Timer**는 일정 주기마다 함수를 실행하는 기능을 가진 클래스.
  - repeat()처럼 함수의 매개변수로 사용되는 함수를 **콜백 함수**라 한다.

```
3초 간격으로 반복 실행
3초 간격으로 반복 실행
3초 간격으로 반복 실행
3초 간격으로 반복 실행
3초 간격으로 반복 실행
```

```
import threading

def repeat():
    print("3초 간격으로 반복 실행")
    timer = threading.Timer(3, repeat)
    timer.start()

repeat()
```

주기

Timer가 실행할 함수(괄호 생략)

threading 모듈

- **threading(쓰레딩)**

  일정 시간 후에 타이머 종료

```
2021-09-08 06:43:39.448949
Timer expired
2021-09-08 06:43:49.451556
```

```python
import threading
import datetime

def call():
    print("Timer expired")
    print(datetime.datetime.now())

print(datetime.datetime.now())
threading.Timer(10, call).start()
```

os 모듈(Module)

◎ **os** 모듈

- 환경변수나 디렉터리, 파일 등의 OS 자원을 제어할 수 있게 해주는 모듈이다.

```
import os

os.chdir('c:/pyworks')  # pydb 디렉터리로 이동

dir = os.popen('dir')   # dir 명령을 실행

print(dir.read())       # dir 결과 출력
```

```
import cx_Oracle
import os

def getconn():
    LOCATION = r'c:/instantclient_19_12'
    os.environ["PATH"] = LOCATION + ":" + os.environ["PATH"] # 환경 설정
    conn = cx_Oracle.connect("system", "12345", "localhost:1522/xe")
    return conn
```

glob 모듈(Module)

◎ **glob** 모듈

- 파일 입출력 관련 프로그램을 만들때 특정 디렉터리에 있는 파일 이름을 모두 알아야 할때 사용.
- 파일 이름을 리스트로 반환해 준다.

```
import glob

data = glob.glob("c:/pyworks/ch06/*.py")
print(data)
```

```
['C:/pyworks/ch06\\argv_test.py', 'C:/pyworks/ch06\\calendar_ex.py', 'C:/pyworks/ch06\\dice.py', 'C:/pyworks/ch06\\dice2.py',
```

연습 문제

---

c 드라이브에서 모든 폴더를 읽어서 pyfile을 찾고, pyfile 폴더에서 모든 txt 파일을 읽어와서 결과를 출력해 보시오.

```python
import os
import glob

# pyfile 찾기
os.chdir('c:/')
dir = os.popen('dir')
print(dir.read())

# 모든 txt파일 읽기
file = glob.glob('c:/pyfile/*.txt')
print(file)

# 특정 파일의 내용 읽기
with open('c:/pyfile/kbo2021.txt', 'r') as f:
    data = f.read()
    print(data)
```

pickle 모듈

◎ **pickle** 모듈

- 객체의 형태를 그대로 유지하면서 파일에 저장하고 불러올 수 있는 모듈이다.
- 이때 객체란, 리스트나 딕셔너리등의 자료구조도 포함한다.

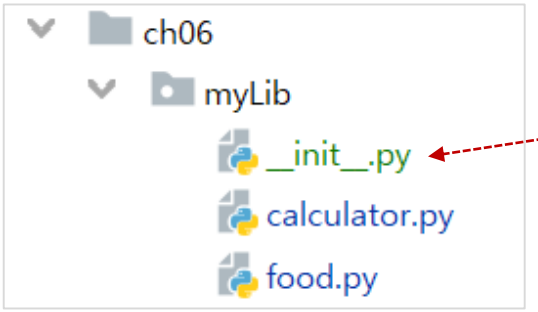
```
import pickle

with open('data.txt', 'wb') as f:
    li = ['강아지', '고양이', '닭']
    dic = {1:'강아지', 2:'고양이', 3:'닭'}
    pickle.dump(li, f)
    pickle.dump(dic, f)

with open('data.txt', 'rb') as f:
    li = pickle.load(f)
    dic = pickle.load(f)
    print(li)
    print(dic)
```

모드	설명
pickle.dump	쓰기
pickle.load	읽기

패키지(package)

◆ 패키지(package)

- 모듈(파일)을 모아 놓은 디렉터리를 **패키지**라 한다.
- 파이썬의 패키지로 인식되려면 __init__.py을 포함해야 한다.



__init__py엔 아무 내용이 없지만, 삭제하면 작동하지 않음

모듈(Module) ^사용 방법

◆ **패키지의 모듈 사용방법**

- **import** 모듈 이름

- **from** 패키지 이름 **import** 파일(모듈)이름

- **from** 패키지이름.파일이름 **import** 함수, 클래스

모듈(Module) 가져오기

calculator.py

```
def add(x, y):
    return x + y

def sub(x, y):
    return x - y

def mul(x, y):
    return x * y

def div(x, y):
    return x / y
```

food.py

```
name = "장금이"

def cook():
    print('요리하다')

def eat():
    print('먹다')
```

모듈(Module) 가져오기

libtest1.py

```
from ch06.myLib.food import cook, eat, name
from ch06.myLib.calculator import add, sub, mul, div

# import food
print(name)
cook()
eat()

# import calculator
print(add(10, 4))
print(sub(10, 4))
print(mul(10, 4))
print(div(10, 4))
```

모듈(Module) 가져오기

libtest2.py

```
from ch06.myLib import food
from ch06.myLib import calculator

# import food
print(food.name)
food.cook()
food.eat()

# import calculator
print(calculator.add(10, 4))
print(calculator.sub(10, 4))
print(calculator.mul(10, 4))
print(calculator.div(10, 4))
```