

PROGRAMMING WITH PYTHON

A decorative wavy blue line runs vertically along the left side of the page, starting from the top and extending to the bottom. It has a slightly irregular, hand-drawn appearance.

PART I

PYTHON 프로그래밍

BY ANECE SO

2

PYTHON 모듈 & 패키지

PYTHON 모듈 & 패키지

◆ 모듈(Module)

- 기능 수행 위해 **변수, 함수, 클래스를 모아둔 파일**
- python에 **다양한 기능을 확장**시켜 줄 수 있는 것
- **다른 Python 프로그램에서 불러서 사용**가능하게 만든 것

분류	설명
표준 모듈	파이썬에서 기본 제공 모듈
사용자 생성 모듈	개발자가 만든 모듈
써드 파티션 모듈	다른 사람들이 만들어서 공개하는 모듈

PYTHON 모듈 & 패키지

◆ 모듈(Module) 사용

➤ 전체 사용

import 모듈명

<= 모듈 내의 변수, 함수 호출 사용

import 모듈명 **as** 별칭

<= 모듈 내의 변수, 함수 호출 사용

```
import math
```

```
print(" 원주율 = ", math.pi)  
print(" 2의 3승 = ", math.pow(2,3) )  
print(" 3! = ", math.factorial(3) )
```

```
import math as m
```

```
print(" 원주율 = ", m.pi)  
print(" 2의 3승 = ", m.pow(2,3) )  
print(" 3! = ", m.factorial(3) )
```

PYTHON 모듈 & 패키지

◆ 모듈(Module) 사용

➤ 모듈 일부만 사용

from 모듈명 **import** 함수명1, 함수명2 <= 특정 함수만 사용

```
from math import pow
print("제곱 = ", pow(2,3))
```

```
from math import pow as p
print(" 원주율 = ", p(2,3))
```

```
from math import pow, pi
print(" 제곱 = ", pow(2,3))
print(" 원주율 = ", pi )
```

```
from math import *
print(" 제곱 = ", pow(2,3))
print(" 원주율 = ", pi )
```

PYTHON 모듈 & 패키지

◆ 모듈(Module) & 스크립트(Script)

➤ 2가지 동작 모드 제어

`__name__` <= 현재 실행 모듈 이름 저장하고 있는 내장 변수

- 현재 실행 중일 경우 변수 저장 값 : `__main__`
- 다른 파일에 포함된 경우 변수 저장 값 : **파일명**

`__name__.py` <= 패키지의 경우 동일 효과

PYTHON 모듈 & 패키지

◆ 모듈(Module) & 스크립트(Script)

➤ 2가지 동작 모드 제어

```
print( __name__ )

if __name__ == "__main__":
    print("나는 현재 실행 중입니다.")
else:
    print("나의 이름은 {0}입니다.".format(__name__))
```


PYTHON 모듈 & 패키지

◆ 모듈(Module) & 스크립트(Script)

➤ 2가지 동작 모드 제어

```
def get_sum(a, b):  
    return a+b
```

```
def main():  
    data_list=[[1,1], [2,2], [3,3], [4,4]]  
    sum =0  
    for i in range(0,len(data_list)):  
        sum += get_sum(data_list[i][0], data_list[i][1])  
  
    print("sum = %d" %sum)
```

PYTHON 모듈 & 패키지

◆ 모듈(Module) & 스크립트(Script)

➤ 2가지 동작 모드 제어

```
if __name__ == "__main__":  
    print("나는 현재 실행 중입니다.")  
    print("aaa.py 시작합니다.")  
    main()  
else:  
    print("나는 {0}입니다.".format(__name__))
```

PYTHON 모듈 & 패키지

◆ 패키지(Package)

- 특정 기능과 관련된 여러 모듈을 묶은 것
- python에 다양한 기능을 확장시켜 줄 수 있는 것
- 다른 Python 프로그램에서 불러서 사용가능하게 만든 것
- 설치 작업이 추가로 필요한 경우도 있음
 - 파이썬 패키지 인덱스(Python Package Index, PyPI)
 - www.pypi.org

PYTHON 모듈 & 패키지

◆ 패키지(Package)

- 파이썬 패키지 인덱스(Python Package Index, PyPI)

www.pypi.org



PYTHON 모듈

◆ 패키지(Package)

➤ 전체 사용

```
import 패키지명.모듈명
```

```
import 패키지명.모듈명1, 모듈명2
```

```
import 패키지명.모듈명 as 별칭
```

```
import urllib.request
```

```
import urllib.request as r
```

PYTHON 모듈

◆ 패키지(Package)

➤ 전체 사용

```
import urllib                # urllib 전체 패키지

req = urllib.request.Request('http://www.google.co.kr')
response = urllib.request.urlopen(req)
```

```
import urllib.request as request    # urllib패키지 request 모듈

req = request.Request('http://www.google.co.kr')
response = request.urlopen(req)
```

PYTHON 모듈

◆ 패키지(Package)

➤ 일부만 사용

from 패키지명.모듈명 **import** 변수명

from 패키지명.모듈명 **import** 함수명

from 패키지명.모듈명 **import** 클래스명

클래스, 함수만 사용

from urllib.request **import** Request, urlopen

PYTHON 모듈

◆ 패키지(Package)

➤ 일부만 사용

```
from urllib.request import Request, urlopen      # urlopen 함수, Request 클래스 가져옴  
  
req = Request('http://www.google.co.kr')        # Request 클래스를 사용하여 req 생성  
response = urlopen(req)                         # urlopen 함수 사용
```

```
from urllib.request import *                    # urllib의 request 모듈의 모든 변수, 함수, 클래스  
  
req = Request('http://www.google.co.kr')  
response = urlopen(req)
```


PYTHON 모듈

◆ 패키지(Package)

➤ 설치 명령어

`pip install 패키지명`

`pip --version`

➤ 버전 2가지 존재

Terminal: Local x +

(c) 2020 Microsoft Corporation. All rights reserved.

(EV_PY37) D:\BEGINNER_AI_2ND\EXAM_PY>pip --version

pip 20.2.2 from C:\Users\anece\anaconda3\envs\EV_PY37\lib\site-packages\pip (python 3.7)

(EV_PY37) D:\BEGINNER_AI_2ND\EXAM_PY>

PYTHON 모듈

◆ 패키지(Package)

➤ 명령어 사용법 확인

`pip --help`

```
Terminal: Local x +  
  
(EV_PY37) D:\BEGINNER_AI_2ND\EXAM_PY>pip --help  
  
Usage:  
  pip <command> [options]  
  
Commands:  
  install          Install packages.  
  download         Download packages.  
  uninstall        Uninstall packages.  
  freeze           Output installed packages in requirements  
  list             List installed packages.  
  show            Show information about installed packages.  
  check            Verify installed packages have compatible
```

PYTHON 모듈

◆ 패키지(Package)

➤ pip 명령어 옵션

- **pip search** 패키지 패키지 검색
- **pip install** 패키지==버전 특정 버전 패키지 설치
(예: pip install requests==2.9.0)
- **pip list** 패키지 목록 출력
- **pip freeze** 패키지 목록 출력
- **pip uninstall** 패키지 패키지 삭제

PYTHON 파일 입출력

FILE I/O

PYTHON 파일 입출력

◆ 바이너리 & 텍스트

[데이터 분류]

데이터 타입	장점	단점
텍스트	<ul style="list-style-type: none">- 텍스트 편집기로 편집 가능- 데이터 설명 추가 가능	<ul style="list-style-type: none">- 바이너리에 비해 크기가 큼- 문자 인코딩 주의 (대부분 UTF-8)
바이너리	<ul style="list-style-type: none">- 텍스트 데이터에 비해 크기 작음- WEB에서 사용되는 데이터	<ul style="list-style-type: none">- 텍스트 편집기로 편집 불가- 데이터 설명 추가 불가

[텍스트 데이터 파일]

파일	특징
XML 파일	<ul style="list-style-type: none">- 범용적인 형식, 웹 API 활용 형식
JSON 파일	<ul style="list-style-type: none">- 자바스크립트 객체 표기 방법 기반 형식 파일- 데이터 교환에 활용
YAML	<ul style="list-style-type: none">- JSON 대용으로 사용, 어플리케이션 설정 파일에 많이 사용되는 파일
CSV/TSV	<ul style="list-style-type: none">- WEB상에서 많이 사용되는 파일

PYTHON 파일 입출력

◆ 인코딩 & 디코딩

[사람 중심]

UNICODE

가을입니다.

인코딩(Encoding)
코드화/암호화

디코딩(Decoding)
역코드화/복호화

[기계 중심]

UTF-8, ASCII 등등 형식의 BYTE

\xea\xb0\x80\xec\x9d\x84\
\xec\x9e\x85\xeb\x8b\x88\
\xeb\x8b\xa4

0b10000000b11101100
0b10011101b10000100
0b11101100b10011110
0b10000101b11101011
0b10001011b10001000
0b11101011b10001011
0b10100100b10111000

PYTHON 파일 입출력

◆ 파일 읽기& 쓰기

쓰기 → file_Object = `open(file , mode='w', encoding=None)`

읽기 → file_Object = `open(file , mode='r', encoding=None)`

racter	Meaning
'r'	open for reading (default)
'w'	open for writing, truncating the file first
'x'	open for exclusive creation, failing if the file already exists
'a'	open for writing, appending to the end of the file if it exists
'b'	binary mode
't'	text mode (default)
'+'	open a disk file for updating (reading and writing)

PYTHON 파일 입출력

◆ 파일 데이터 쓰기

파일 전체 데이터 쓰기

➔ `alldata=f.write(str)`

파일 줄 단위 데이터 쓰기

➔ `line = f.writeline(list)` ** 자동 개행 안됨

PYTHON 파일 입출력

◆ 파일 데이터 읽기

파일 전체 데이터

→ `alldata=f.read()`

파일 `n`만큼 데이터

→ `alldata=f.read(n)`

파일 줄 단위 데이터

→ `line = f.readline()`

파일 줄 단위 전체 리스트 반환

→ `lines = f.readlines()`

PYTHON 파일 입출력

◆ 파일 포인터 위치

파일 위치 설정/이동 ➔ `f.seek(offset)` # `f.seek(0)` 파일 처음

파일 위치 읽기 ➔ `f.tell()`

파일 닫힘 여부 반환 ➔ `f.closed`

파일모드 반환 ➔ `f.mode`

파일이름 반환 ➔ `f.name`

PYTHON 파일 입출력

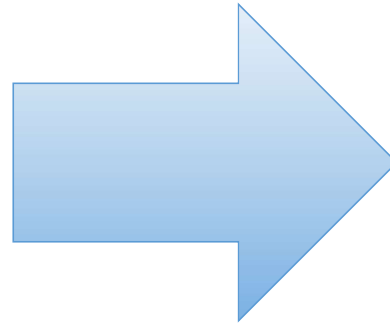
◆ 파일 데이터 읽기

```
alldata=f.read( )
```

```
alldata=f.read( n )
```

```
line = f.readline( )
```

```
lines = f.readlines( )
```



1바이트 크기
배열 bytes 객체

PYTHON 파일 입출력

◆ with 파일 as 구문

```
with open(file , mode='w', encoding=None) as file_obj:  
    file_obj.write( ' data ' )
```

file_obj.close() 생략 가능

파일의 close() 자동으로 처리됨!!!

바이너리 FILE I/O



PYTHON 파일 입출력

◆ 바이너리 파일

- 이진 파일 또는 바이너리 파일(binary file)
- 컴퓨터 저장과 처리 목적 위해 이진 형식으로 인코딩된 데이터 파일
- 문서 편집기로 열었을 경우 알아볼수 없는 문자들

[바이너리 데이터 파일]

분류	파일 종류
이미지 파일	- jpg, png,
오디오 파일	- mp3, mp4, ...
실행 파일	- exe, bin,

PYTHON 파일 입출력

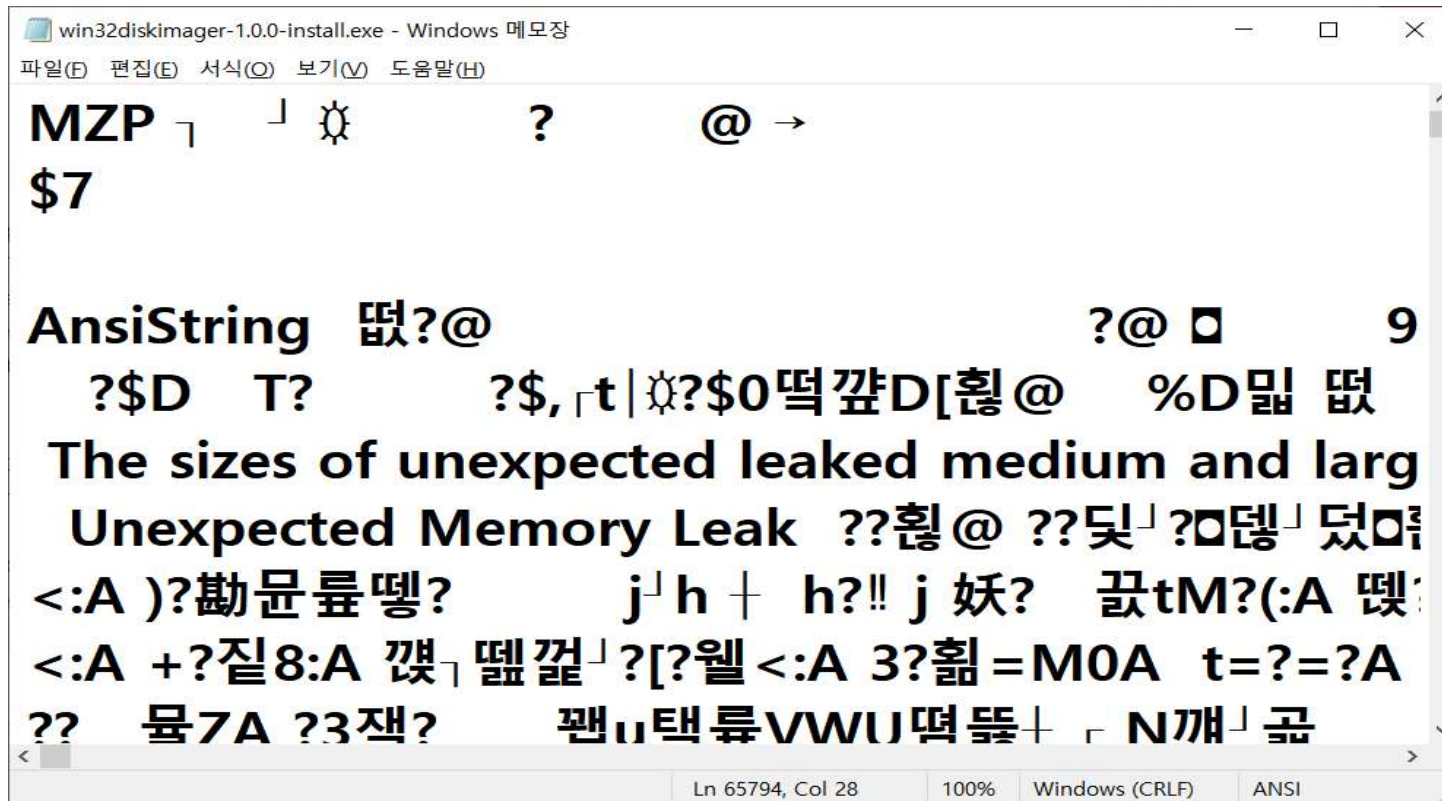
◆ 바이너리 데이터 타입

bytes 데이터 타입

- 1바이트 단위 값을 연속적으로 저장하는 시퀀스 자료형
- 1바이트 => 8비트
- 0~255(0x00~0xFF)까지 정수 사용

PYTHON 파일 입출력

◆ 바이너리 파일



win32diskimager-1.0.0-install.exe - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

MZP ǀ ǂ ? @ →
\$7

AnsiString 덫?@ ?@ □ 9
?\$D T? ?\$, ǂt|ǂ? \$0떡깟D[훤@ %D밧 덫
The sizes of unexpected leaked medium and larg
Unexpected Memory Leak ??훤@ ??딛 ǂ ǂ덫 ǂ덫
<:A)?勘문릏덫? j ǂh + h?!! j 妖? ٱtM?(:A 덫
<:A +?질8:A ٱ 덫 ٱ ǂ[?웰<:A 3?훤=M0A t=?=?A
?? 릏7A ?3잭? ٱ ǂ ٱ ٱ VWU떡뽳+ ǂ N개 ٱ

Ln 65794, Col 28 100% Windows (CRLF) ANSI

PYTHON 파일 입출력

◆ 바이너리 데이터 타입

bytes 객체 생성

bytes(숫자):	숫자만큼 0으로 채워진 바이트 객체 생성
bytes(반복가능한객체)	반복 가능한 객체로 바이트 객체 생성
bytes(b'바이트객체')	바이트 객체로 바이트 객체 생성

PYTHON 파일 입출력

◆ 바이너리 모듈

import struct

- C언어의 구조체를 구현한 모듈
- 파일이나 네트워크 연결에 사용하는 이진 데이터 다루는 모듈

Format	C Type	Python type	Standard size	Notes
x	pad byte	no value		
c	char	bytes of length 1	1	
b	signed char	integer	1	(1), (2)
B	unsigned char	integer	1	(2)
?	_Bool	bool	1	(1)
h	short	integer	2	(2)
H	unsigned short	integer	2	(2)
i	int	integer	4	(2)
I	unsigned int	integer	4	(2)
l	long	integer	4	(2)
L	unsigned long	integer	4	(2)
q	long long	integer	8	(2)