

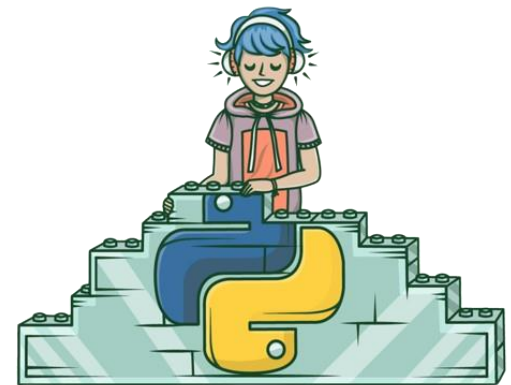


파이썬 데이터 구조

인공지능 플랫폼 설계 - 5주차

# 학습 내용

1. 권한 설정과 프로세스
2. 리스트
3. 딕셔너리



---

# IT 플랫폼 실습

---

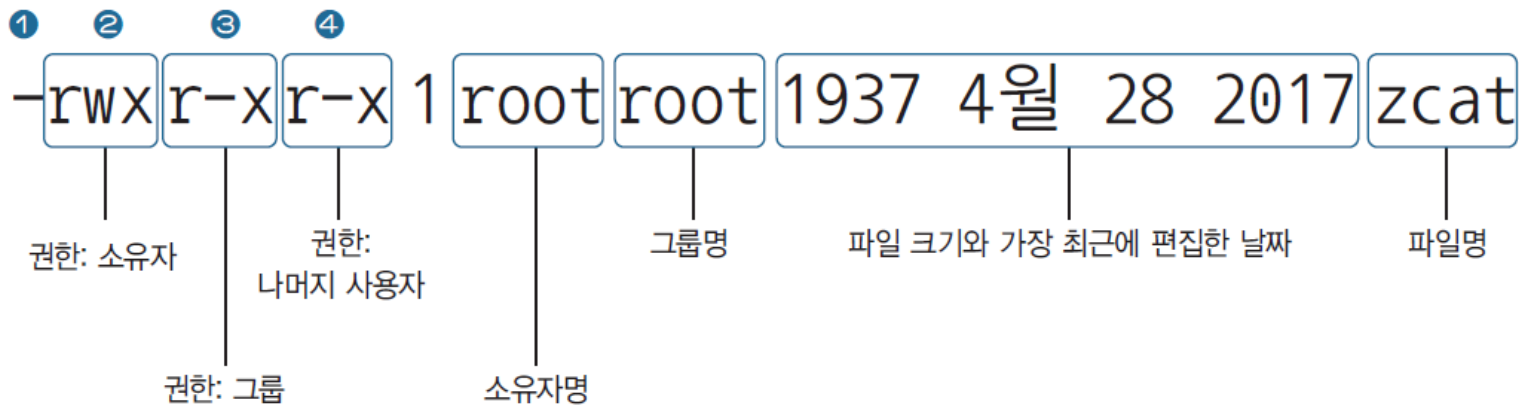
# \_리눅스 파일 시스템



# \_리눅스 파일 시스템

## 권한 및 소유권

- 파일에 지정할 수 있는 권한 3가지  
: 읽기(r), 쓰기(w), 실행(x)
- 권한 대상  
: 소유자 (u), 그룹소속 (g), 나머지 (o)



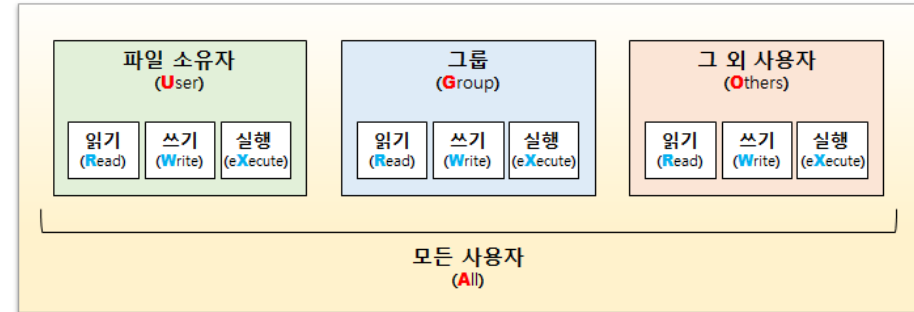
# \_리눅스 파일 시스템

## chmod : 권한 모드 변경(change mode) 도구

- 상대 변경법 : 필요한 권한만 설정(+)하거나 해제(-)  
파일에 대한 접근 권한을 변경할 때

# chmod o-r /bin/zcat

# chmod g+w /bin/zcat



- 절대 변경법 : 숫자로 계산해서 지정  
: 읽기(4), 쓰기(2), 실행(1) → 읽기+쓰기+실행 권한 = 4+2+1 = 7  
# chmod 755 /bin/zcat

### 권한 코드

권한	문자	숫자
읽기	r	4
쓰기	w	2
실행	x	1



## \_리눅스 파일 시스템

```
./myls.sh  
chmod a+x myls.sh  
./myls.sh
```

```
cd /home/soxuser  
mkdir testdir  
→ 에러 메시지 확인
```

```
sudo mkdir testdir  
→ 비번 입력
```

# \_리눅스 파일 시스템

## ps 명령으로 프로세스 살펴보기

- `$ for i in {1..10}; do sleep 1; done &`  
`[1] 19829` ← 백그라운드에서 실행 중인 명령의 PID  
`$ ps`

PID	TTY	TIME	CMD
19522	pts/17	00:00:00	bash
19829	pts/17	00:00:00	bash
19832	pts/17	00:00:00	sleep
19833	pts/17	00:00:00	ps

- 특정 프로세스 찾기  
`ps ax | grep java`

- top





# \_리눅스 파일 시스템

## 프로세스 종료시키기

- kill : PID 기반으로 프로세스 하나만 종료  
# kill -9 1367
- killall : 프로세스를 생성한 프로그램 이름을 이용해 그 프로그램이 생성한 프로세스를 모두 종료  
# killall -9 mysqld



# \_리눅스 파일 시스템

## SSH로 리눅스로 접속하기

- ssh : 리눅스에 커맨드로 접속하는 명령

- 형식

ssh [접속 ID]@접속주소 -p [포트번호]

ssh osboxes@192.168.100.2 -p 22

- 접속 주소를 확인하는 명령

ifconfig



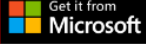
# \_SSH 접속 환경

## Winscp 다운로드

<https://winscp.net/eng/download.php>

WinSCP 6.3 is a major application update. New features and enhancements include:

- Single large file can be downloaded using multiple SFTP connections.
- Support for OpenSSH certificates for host verification.
- File hash can be used as criterion for synchronization.
- Improved behavior when duplicating and moving remote files.
- SSH core upgraded to PuTTY 0.80. That includes support for HMAC-SHA-512 and mitigation of "Terrapin" vulnerability
- TLS/SSL core upgraded to OpenSSL 3.
- [List of all changes.](#)

**DOWNLOAD WINSCP 6.3 (11 MB)**  **50% OFF** [OTHER DOWNLOADS](#)

308,138 downloads since 2024-02-14 [What is this?](#)

## Putty 다운로드

<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>

**Package files**

You probably want one of these. They include versions of all the PuTTY utilities (except the WinSCP client). (Not sure whether you want the 32-bit or the 64-bit version? Read the [FAQ entry](#).)

We also publish the latest PuTTY installers for all Windows architectures as a free-of-charge service.

**MSI ("Windows Installer")**

64-bit x86:	<a href="#">putty-64bit-0.80-installer.msi</a>	<a href="#">(signature)</a>
64-bit Arm:	<a href="#">putty-arm64-0.80-installer.msi</a>	<a href="#">(signature)</a>
32-bit x86:	<a href="#">putty-0.80-installer.msi</a>	<a href="#">(signature)</a>

**Unix source archive**

.tar.gz:	<a href="#">putty-0.80.tar.gz</a>	<a href="#">(signature)</a>
----------	-----------------------------------	-----------------------------



# \_SSH 접속 환경

## 1. 주소 확인

ifconfig

```
Ubuntu18 Server [실행 중] - Oracle VM VirtualBox
파일  마신  보기  입력  장치  도움말

Swap usage: 0%

* Strictly confined Kubernetes makes edge and IoT secure. Learn how Micro
just raised the bar for easy, resilient and secure K8s cluster deployme

https://ubuntu.com/engage/secure-kubernetes-at-the-edge

63 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

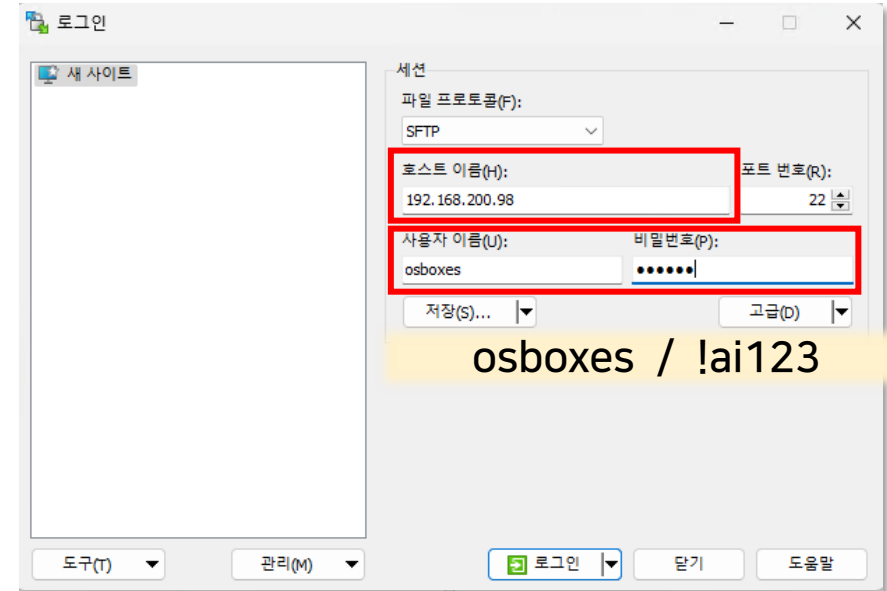
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

New release '20.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

osboxes@osboxes:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.200.98 netmask 255.255.255.0 broadcast 192.168.200.
    inet6 fe80::a00:27ff:fe6e:32ca prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:6e:32:ca txqueuelen 1000 (Ethernet)
    RX packets 128 bytes 8883 (8.8 KB)
    RX errors 0 dropped 11 overruns 0 frame 0
    TX packets 12 bytes 1450 (1.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

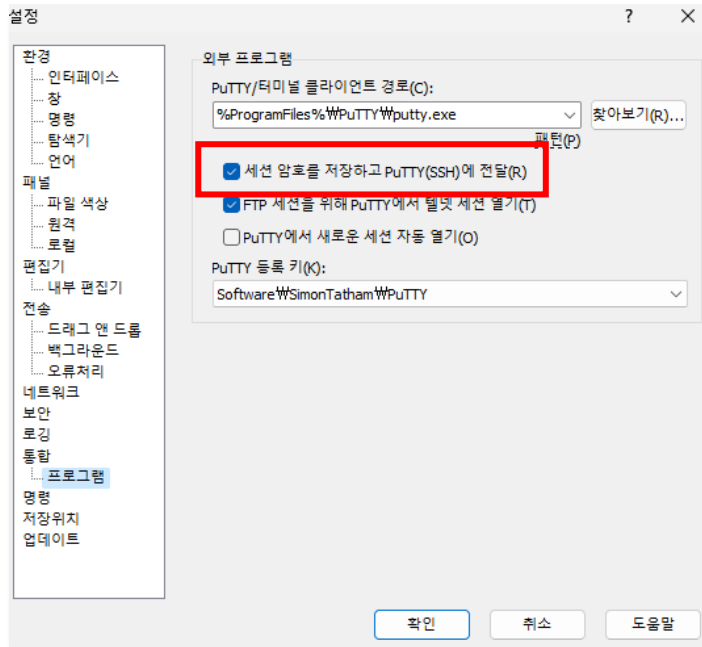
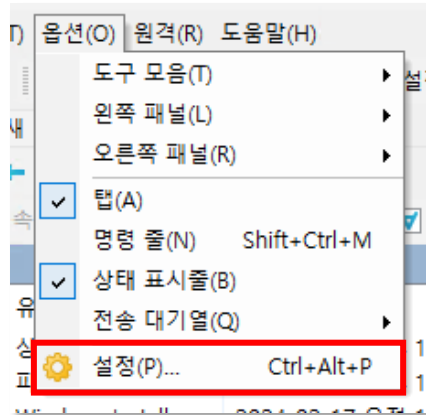
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 80 bytes 5920 (5.9 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 80 bytes 5920 (5.9 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

## 2. 접속 정보 입력

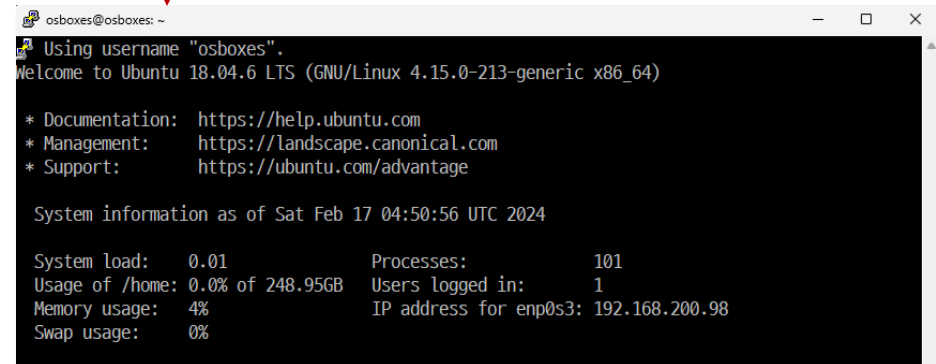
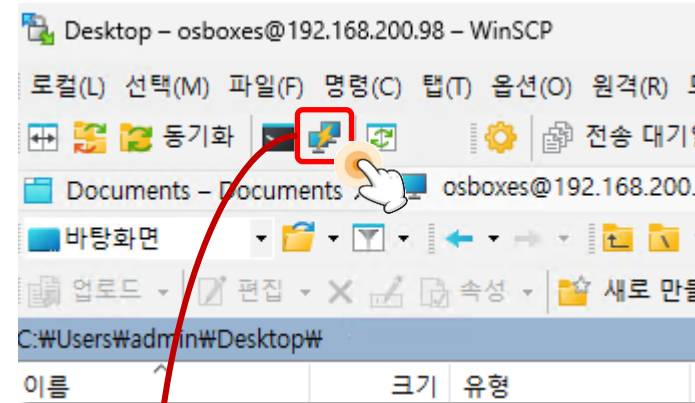


# \_SSH 접속

## 3. 연결 설정



## 4. 터미널 열기



---

# Python 프로그래밍

---

# \_컬렉션

## ■ 리스트

→리스트(List)는 데이터를 다루기 편리하여 매우 자주 활용되는 컬렉션 자료형

→파이썬의 내장함수로 리스트 데이터를 다루는 함수가 있음

```
list_1 = [1, 2, 3, 4, 5, 1, 3]
list_2 = [ ]
print(list_1)
print(list_2)
```

```
print(len(list_1))
```

```
list_1[3] = 9999
print(list_1)
```

```
list_1.append(100)
print(list_1)
```

```
[1, 2, 3, 4, 5, 1, 3]
```

```
[ ]
```

```
7
```

```
[1, 2, 3, 9999, 5, 1, 3]
```

```
[1, 2, 3, 9999, 5, 1, 3, 100]
```

# \_컬렉션

## ■ 리스트

→ 데이터 삽입(Insert), 삭제(Remove), 복사(Copy)

```
list_1 = [1, 2, 3, 9999, 5, 1, 3, 100]  
list_2 = [ ]  
print(list_1)
```

```
list_1.remove(9999)  
print(list_1)
```

```
list_1.insert(0,777)  
print(list_1)
```

```
list_2 = list_1.copy( )  
print(list_2)
```

```
[1, 2, 3, 9999, 5, 1, 3, 100]
```

```
[1, 2, 3, 5, 1, 3, 100]
```

```
[777, 1, 2, 3, 5, 1, 3, 100]
```

```
[777, 1, 2, 3, 5, 1, 3, 100]
```



## \_컬렉션

### ■ 리스트

→ 데이터 루프 ( for - in )

```
list = [1, 2, 3, 5, 1, 3]
```

```
for num in list :  
    print(num)
```

## Python 실습

실습문제 2 : 다음의 배열을 모두 출력하는 프로그램을 작성하시오

`['kim' , 'lee' , 'park']`

kim

lee

park



## Python 실습

실습문제 2 : 다음의 배열을 모두 출력하는 프로그램을 작성하시오

`['kim' , 'lee' , 'park']`

```
data = ['kim' , 'lee' , 'park']
```

```
for str in data :
```

```
    print(str)
```



# \_컬렉션

## ■ 딕셔너리

→ 딕셔너리(Dictionary)는 단어 그대로 사전과 같은 자료형으로,  
값(value)과 키(key)가 한 쌍을 이루어 요소가 되는 자료구조.

→ 키를 이용하여 쌍을 이루는 값에 접근할 수 있으므로 신속하게 값을 찾아내야 할 때 딕셔너리 사용.

```
dict_1 = {'name': '홍길동', 'birth': 1990, 'addr': 'KR'}  
print(dict_1)  
print(dict_1['birth'])
```

```
{'name': '홍길동', 'birth': 1990, 'addr': 'KR'}  
1990
```

```
dict_1['weight'] = 60.5  
dict_1['family'] = ['아빠', '엄마', '여동생']  
print(dict_1)
```

```
{'name': '홍길동', 'birth': 1990, 'addr': 'KR',  
'weight': 60.5, 'family': ['아빠', '엄마', '여동생']}
```

```
dict_1.update({'weight': 67.8, 'hobby': ['게임', '독서']})  
print(dict_1)
```

```
{'name': '홍길동', 'birth': 1990, 'addr': 'KR',  
'weight': 67.8, 'family': ['아빠', '엄마', '여동생'],  
'hobby': ['게임', '독서']}
```

# \_컬렉션

## ■ 딕셔너리

```
dict_1['hobby'] = ['축구', '등산']  
print(dict_1)
```

```
{'name': '홍길동', 'birth': 1990, 'addr': 'KR',  
 'weight': 67.8, 'family': ['아빠', '엄마', '여동생'],  
 'hobby': ['축구', '등산']}
```

```
del dict_1['weight']  
del dict_1['birth']  
del dict_1['addr']  
print(dict_1)
```

```
{'name': '홍길동', 'family': ['아빠', '엄마',  
 '여동생'], 'hobby': ['축구', '등산']}
```