

오픈소스SW기초 (2024-2)

7. 리눅스 기초 명령어 III

국립금오공과대학교

컴퓨터공학과 / 인공지능공학과

김 경 수

목차

- ① grep 명령과 정규 표현식
- ② 명령어 파이프
- ③ 아카이브와 압축 명령

학습 목표

- ① 정규 표현식의 개념을 이해하고 정규 표현식을 이용하여 다양한 텍스트 패턴을 작성하는 방법을 숙달할 수 있다.
- ② 리눅스의 grep 명령과 정규 표현식을 이용하여 텍스트로부터 다양한 정보를 추출하는 방법을 이해하고 실제 활용할 수 있다.
- ③ 리눅스에서 사용되는 다양한 명령어들을 순차적으로 조합하여 활용하는 명령어 파이프의 개념을 이해하고 간단한 명령들을 조합하여 실행할 수 있다.
- ④ 리눅스에서 제공하는 다양한 아카이브 및 압축 명령의 개념을 이해하고, 이들의 특성을 논리적으로 설명하며, 실제 활용하는 방법을 숙달할 수 있다.

실습 준비

- LMS에 업로드되어있는 “**reg_exp.txt**” 파일을 다운로드한 후 실습을 수행할 폴더인 “**C:/LinuxStudy/**” 경로에 붙여넣기한다.
- Git bash에서 “**cd C:/LinuxStudy/**” 명령어를 이용하여 실습 폴더로 이동한다.

grep 명령과 정규 표현식

grep 명령과 정규 표현식

- “grep”명령어는 무엇인가?
 - 리눅스에서 제공하는 텍스트 패턴 매칭 및 추출을 위한 명령어
 - 정규 표현식을 이용하여 텍스트 내 특정 패턴을 검색하는 기능 제공
- 정규 표현식(Regular Expression; RE)이란?
 - 텍스트에 존재하는 다양한 패턴을 표현하기 위한 도구
 - 정규 표현식으로 작성된 텍스트의 특정 패턴과 매칭되는 모든 텍스트를 문서에서 검색하여 추출하는 데 널리 사용됨

grep 명령과 정규 표현식

- 정규 표현식의 사용 사례

- HTML 문서 내 존재하는 웹 링크(Web Link) 추출하기
- 웹 크롤러(Web Crawler) 제작
- 문서 내 존재하는 개인정보(e.g., 주민등록번호, 전화번호)를 검색하여 삭제하기
- 문서 내 동사, 명사, 관사와 같은 특정 문법을 갖는 단어/문자 추출

이 외에도, 문서로부터 각종 정보를 추출하기 위한 다양한 응용 분야에서 가장 기본적인 도구로 광범위하게 활용됨.

grep 명령어의 기본 패턴

- 기본 사용 패턴: **grep [옵션] <검색패턴> <파일이름>**

➤ (예제) 현재 디렉토리 내의 파일 중에서 “txt”를 글자를 포함하는 파일들의 리스트를 화면에 출력하시오.

```
ls -al | grep 'txt'
```

- “grep” 명령에서 자주 사용되는 옵션
 - **-n**: 행 번호를 함께 출력
 - **-i**: 대소문자를 구별하지 않고 검색
 - **-v**: 검색할 문자열이 나타나지 않는 행을 출력

정규 표현식을 활용한 텍스트 검색

- grep 에서 정규표현식을 사용하는 경우, 검색 패턴 입력 시 반드시 따옴표 ‘ ’ 안에 정규 표현식을 기술해야 함.

➤ (예제 1) `ls -al | grep '^d'`

➤ (예제 2) `ls -al | grep '.py$'`

➤ (예제 3) `ls -al | grep '.txt$'`

임의의 문자를 지정하는 메타 문자

- 임의의 문자를 나타내는 메타 문자: `.`
- 임의의 두 문자를 나타내는 메타 문자: `..`
- 이 때, 메타 문자가 아닌 “.” 글자를 검색하는 경우: `\.`으로 기입함.
 - 이스케이프(Escape): 문자 앞에 `\`를 붙여서 메타 문자로 인식하지 않게 함.

메타 문자	의 미
<code>.</code>	임의의 문자 하나
<code>[]</code>	<code>[]</code> 안에 포함된 임의의 문자
<code>[^]</code>	<code>[]</code> 안에 포함되지 않는 문자
<code>\</code>	<code>\</code> 다음의 문자는 메타 문자로 인식하지 않음.

임의의 문자를 지정하는 메타 문자

- 정규 표현식 작성 및 grep에서의 활용 예
 - (예제 1) t와 f 사이에 임의의 문자 하나를 포함하는 문자열을 행 단위로 검색하시오.

```
grep 't.f' reg_exp.txt
```

```
KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ grep 't.f' reg_exp.txt
tefst
tefefefst
tefefefefefefefefst example135978
example4 tefst
```

임의의 문자를 지정하는 메타 문자

- 정규 표현식 작성 및 grep에서의 활용 예

➤(예제 2) “example” 바로 뒤에 1부터 4까지의 숫자 중 하나가 나오는 문자열을 행 단위로 검색하시오.

```
grep 'example[1-4]' reg_exp.txt
```

```
KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ grep 'example[1-4]' reg_exp.txt
example1
example11111
example2
example156
tefefefefefefefst example135978
example4 tefst
example1344
example13
```

임의의 문자를 지정하는 메타 문자

- 정규 표현식 작성 및 grep에서의 활용 예

➤(예제 3) “example” 바로 뒤에 1,3을 포함하지 않는 모든 문자열을 행 단위로 검색하시오.

```
grep 'example[^13]' reg_exp.txt
```

```
KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ grep 'example[^13]' reg_exp.txt
example2
example758
example4 tefst
example513
example95
```

위치를 지정하는 메타 문자

- 다른 메타 문자와 조합하여 위치를 지정할 때 사용함.

➤ #으로 시작하는 문자열을 검색 → **^#**

➤ #으로 끝나는 문자열을 검색 → **#\$**

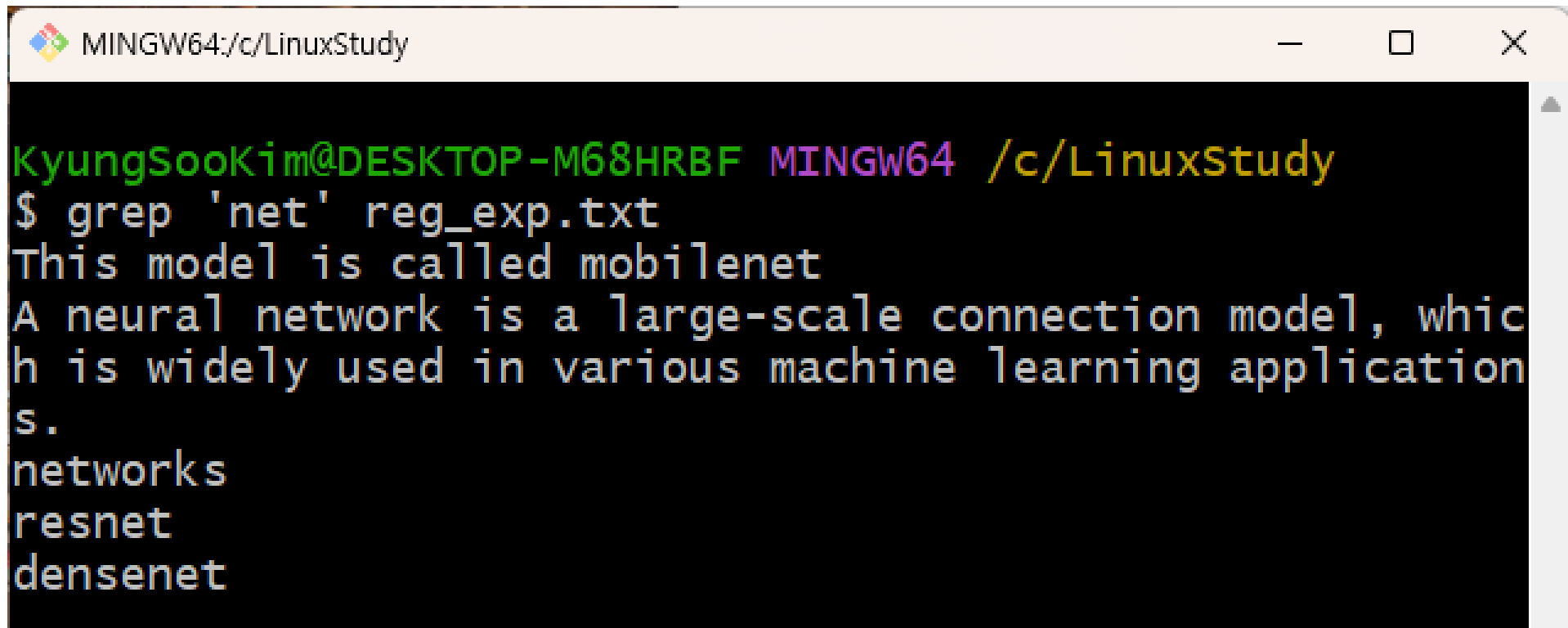
➤ 공백 행(empty row) → **^\$**

여기서 #은 임의의 문자열
또는 메타 문자를 의미함.

위치를 지정하는 메타 문자

- (예제 1) “net”을 포함하는 모든 문자열을 행 단위로 검색하시오.

➤ `grep 'net' reg_exp.txt`



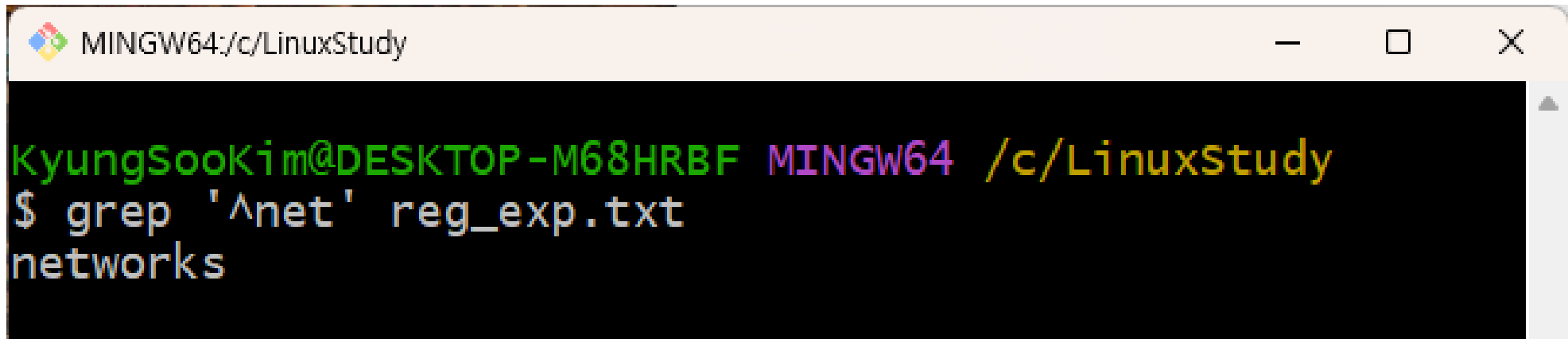
```
MINGW64:/c/LinuxStudy

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ grep 'net' reg_exp.txt
This model is called mobilenet
A neural network is a large-scale connection model, which
is widely used in various machine learning applications.
networks
resnet
densenet
```

위치를 지정하는 메타 문자

- (예제 2) “net”으로 시작하는 모든 문자열을 행 단위로 검색하시오.

➤ `grep '^net' reg_exp.txt`

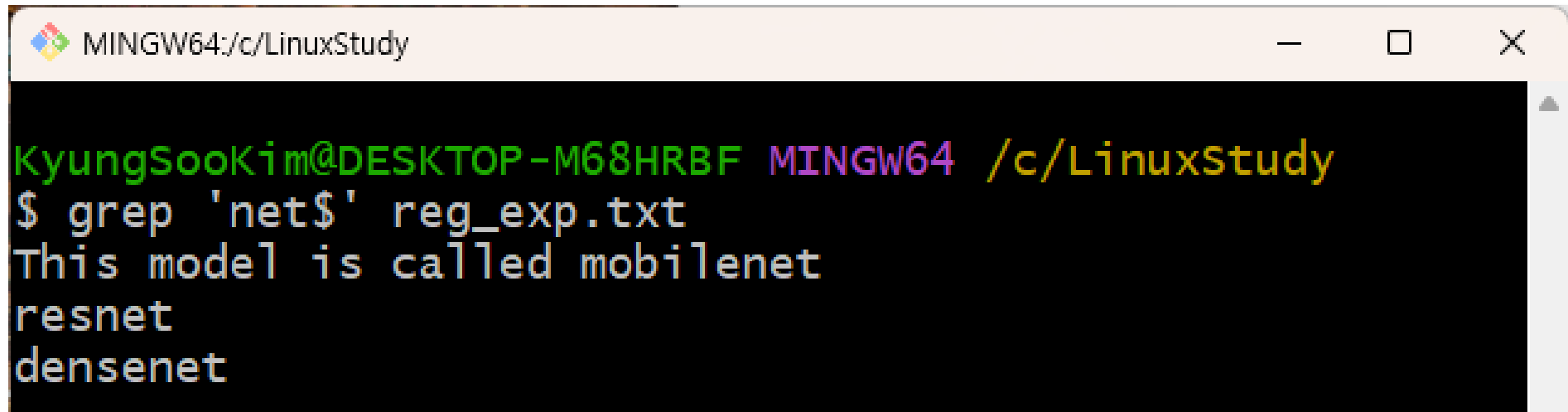


```
MINGW64:/c/LinuxStudy  
  
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy  
$ grep '^net' reg_exp.txt  
networks
```


위치를 지정하는 메타 문자

- (예제 3) “net”으로 끝나는 모든 문자열을 행 단위로 검색하시오.


➤ `grep 'net$' reg_exp.txt`



```
MINGW64:/c/LinuxStudy

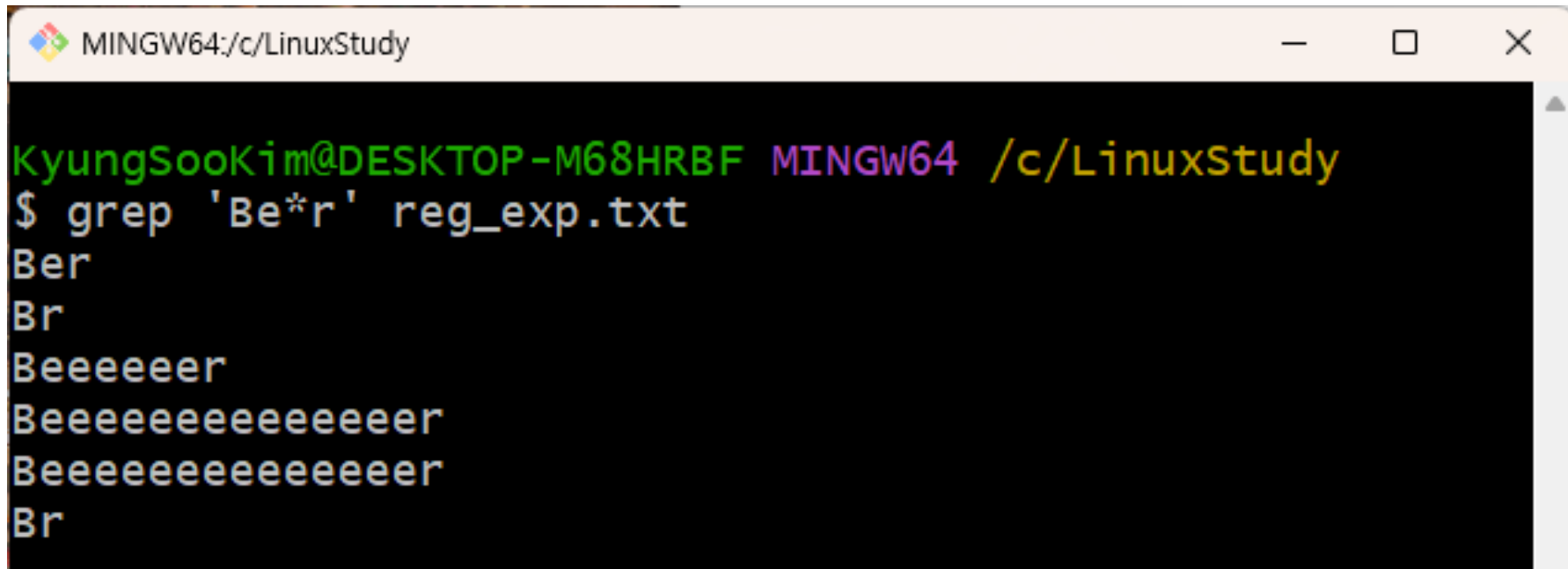
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ grep 'net$' reg_exp.txt
This model is called mobilenet
resnet
densenet
```

반복을 지정하는 메타 문자

- 다른 정규 표현식의 뒤에서 사용되어 “직전의 정규 표현식이 일정 횟수만큼 반복됨”을 의미함.
- 반복을 나타내는 메타 문자 →  *
- 앞의 문자열이 0회 이상 반복됨을 의미함.

반복을 지정하는 메타 문자

- (예제 1) B와 r사이에 문자 'e'가 0회 이상 반복되는 모든 문장을 검색하시오.
 - 즉, Br, Ber, Beer, Beeer, Beeeer, BeeerBeeeer과 같은 패턴을 갖는 문장 검색
 - **grep 'Be*r' reg_exp.txt**



```
MINGW64:/c/LinuxStudy  
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy  
$ grep 'Be*r' reg_exp.txt  
Ber  
Br  
Beeeeeer  
Beeeeeeeeeeeeeeer  
Beeeeeeeeeeeeeeer  
Br
```

확장 정규 표현식

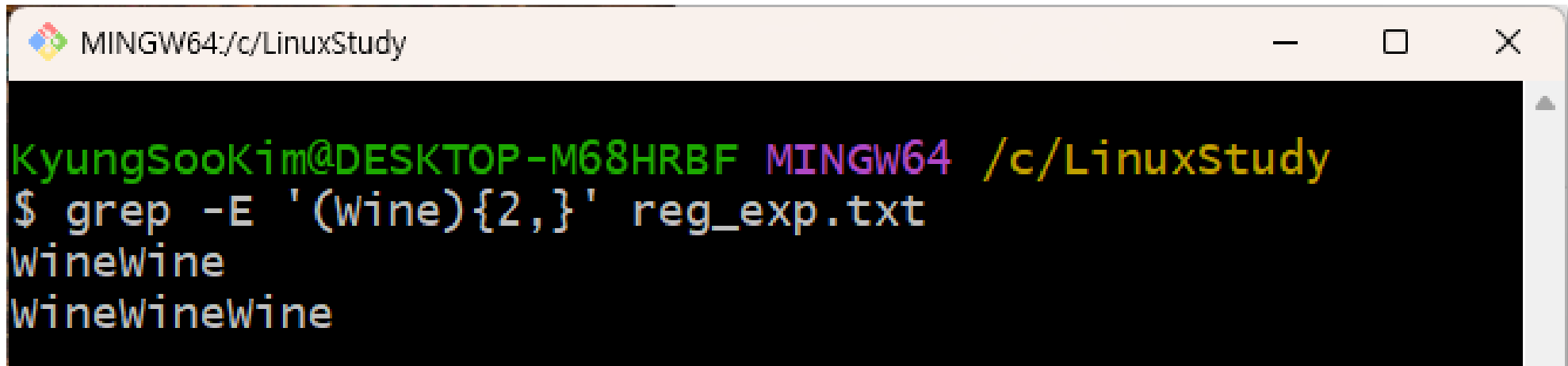
- grep 명령에서 옵션으로 “-E”를 지정하면 확장 정규 표현식으로 해석함.
- 반복을 나타내는 확장 정규 표현식

기본 정규 표현식	확장 정규 표현식	의 미
*	*	<u>0회 이상</u> 반복되는 문자열을 검색
	+	<u>1회 이상</u> 반복되는 문자열을 검색
	?	<u>0회 또는 1회</u> 반복되는 문자열을 검색
\{m,n\}	{m,n}	m회 이상 n회 이하 반복되는 문자열을 검색
\{m\}	{m}	m회 반복되는 문자열을 검색
\{m,\}	{m,}	m회 이상 반복되는 문자열을 검색

확장 정규 표현식

- (예제 1) 단어 “Wine”이 2번 이상 반복되는 모든 문자열을 검색하시오.

➤ `grep -E '(Wine){2,}' reg_exp.txt`



```
MINGW64:/c/LinuxStudy  
  
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy  
$ grep -E '(Wine){2,}' reg_exp.txt  
WineWine  
WineWineWine
```

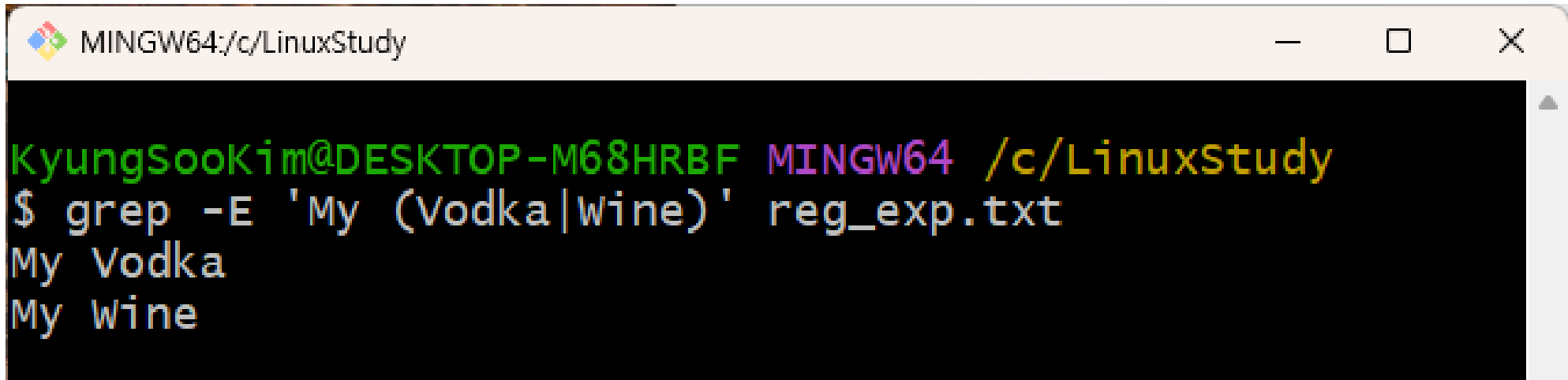
기타 메타 문자

기본 정규 표현식	확장 정규 표현식	의 미
\(\)	()	그룹화할 때 사용
		여러 정규 표현식을 OR 조건으로 연결할 때 사용

확장 정규 표현식

- (예제 1) “My”로 시작하고 뒤에 “Vodka” 또는 “Wine”이 나타나는 패턴을 갖는 문자열을 검색하시오.

➤ `grep -E 'My (Vodka|Wine)' reg_exp.txt`

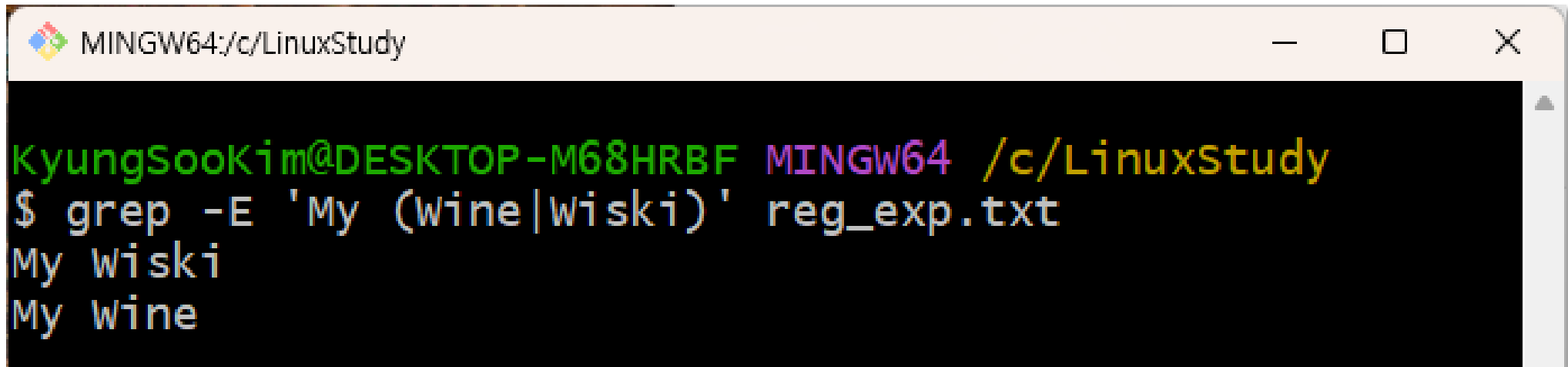


```
MINGW64:/c/LinuxStudy  
  
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy  
$ grep -E 'My (Vodka|Wine)' reg_exp.txt  
My Vodka  
My Wine
```

확장 정규 표현식


- (예제 2) “My”로 시작하고 뒤에 “Wine” 또는 “Wiski”가 나타나는 패턴을 갖는 문자열을 검색하시오.

➤ `grep -E 'My (Wine|Wiski)' reg_exp.txt`



```
MINGW64:/c/LinuxStudy  
  
KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy  
$ grep -E 'My (Wine|Wiski)' reg_exp.txt  
My wiski  
My wine
```

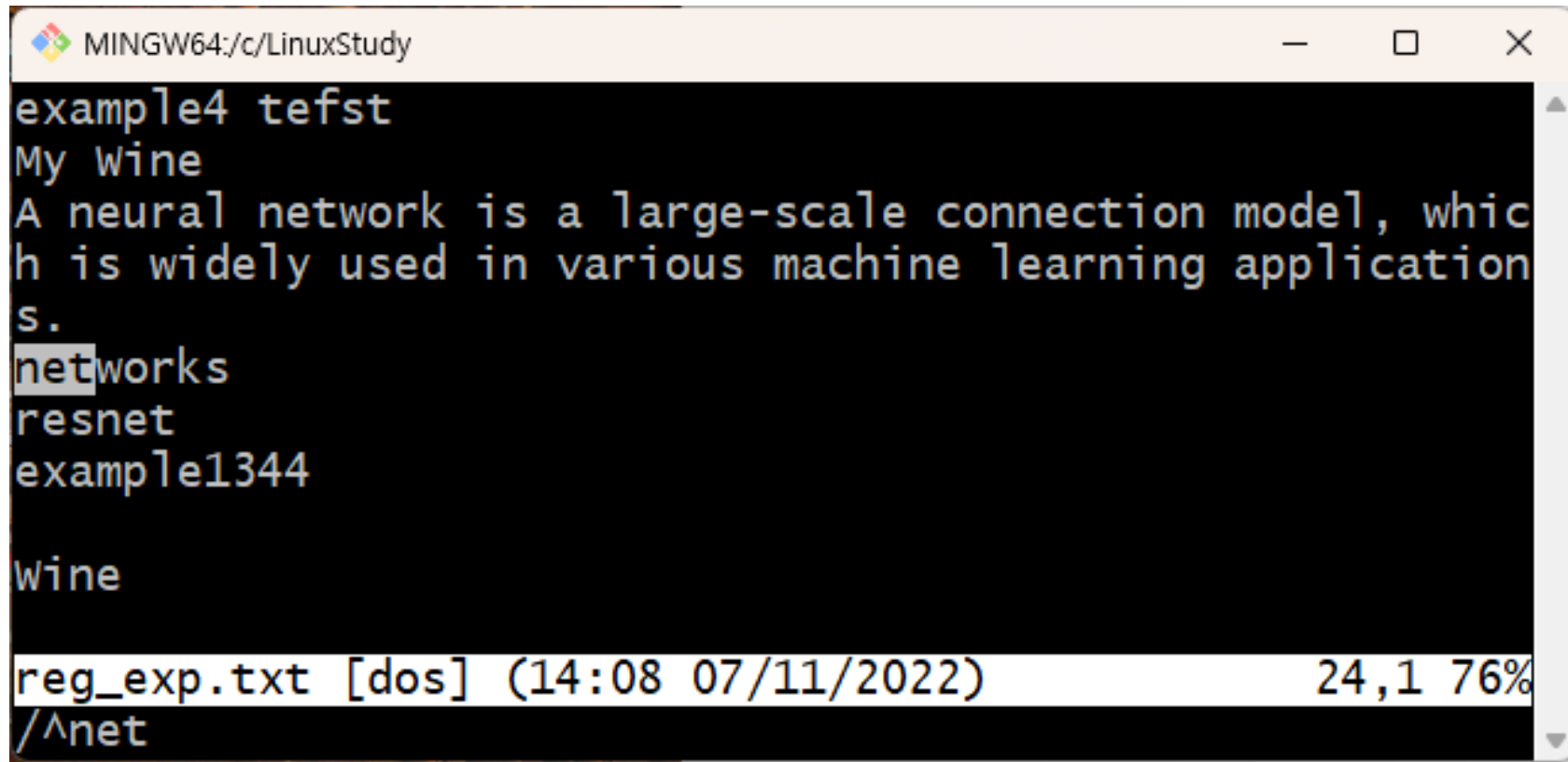

Vim에서의 정규 표현식 활용 방법

- Vim에서 텍스트 검색을 수행할 때, 정규 표현식의 일부 기능을 사용할 수 있다.
- Vim 에서의 텍스트 검색 방법
 - ① “Esc” 키로 명령 모드 진입
 - ②  / 뒤에 검색을 원하는 문자열(정규 표현식 포함) 입력

Vim에서의 정규 표현식 활용 방법

- (예제 1) “net”으로 시작하는 모든 문자열을 검색하시오.

➤ 명령 모드 → `/^net`



```
MINGW64:/c/LinuxStudy
example4 tefst
My Wine
A neural network is a large-scale connection model, which
is widely used in various machine learning applications.
networks
resnet
example1344

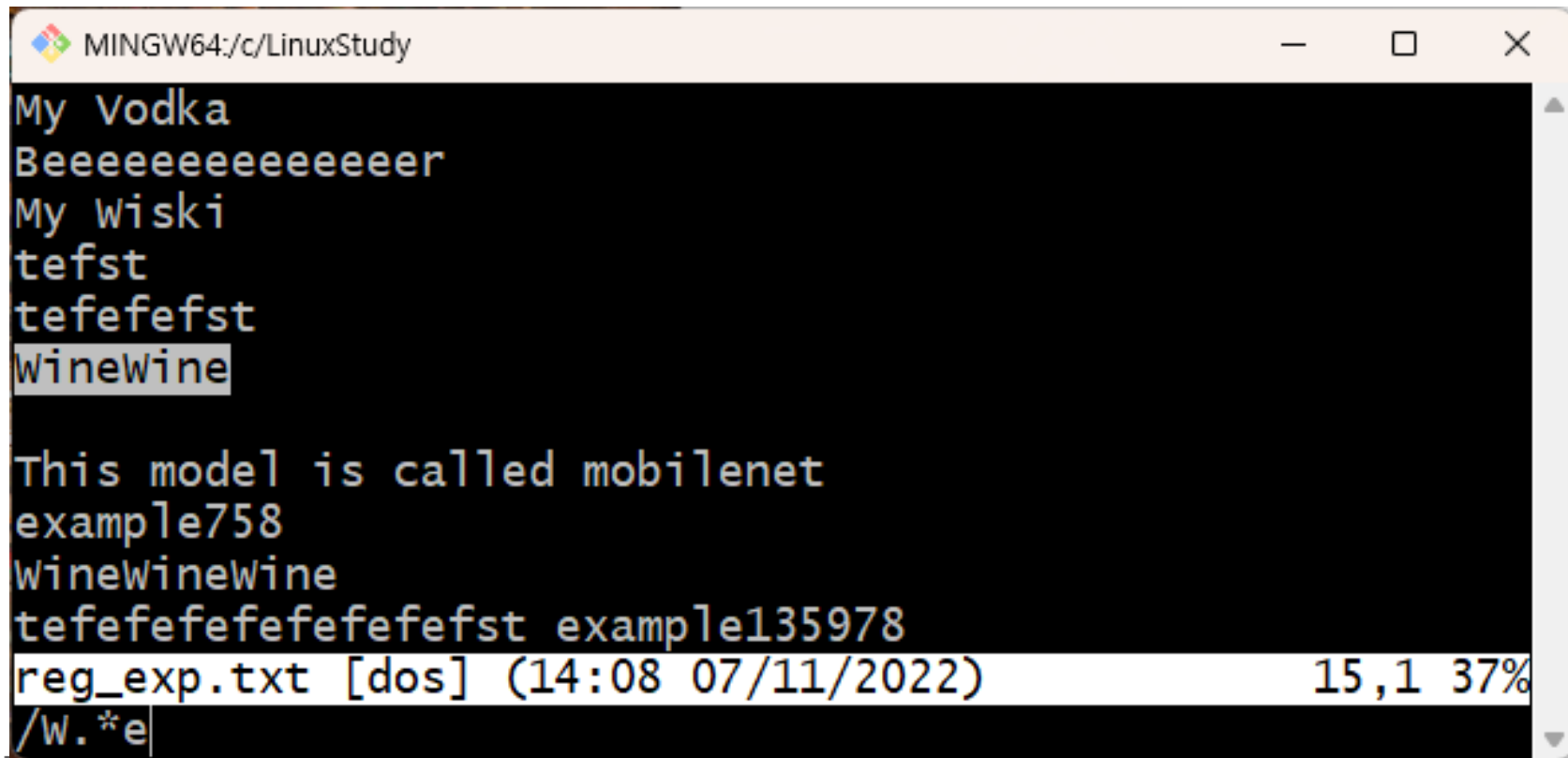
Wine

reg_exp.txt [dos] (14:08 07/11/2022) 24,1 76%
/^net
```

Vim에서의 정규 표현식 활용 방법

- (예제 2) “W”와 “e”사이에 0개 이상의 문자를 포함하는 문자열을 검색하시오.

➤ 명령 모드 → `/W.*e`



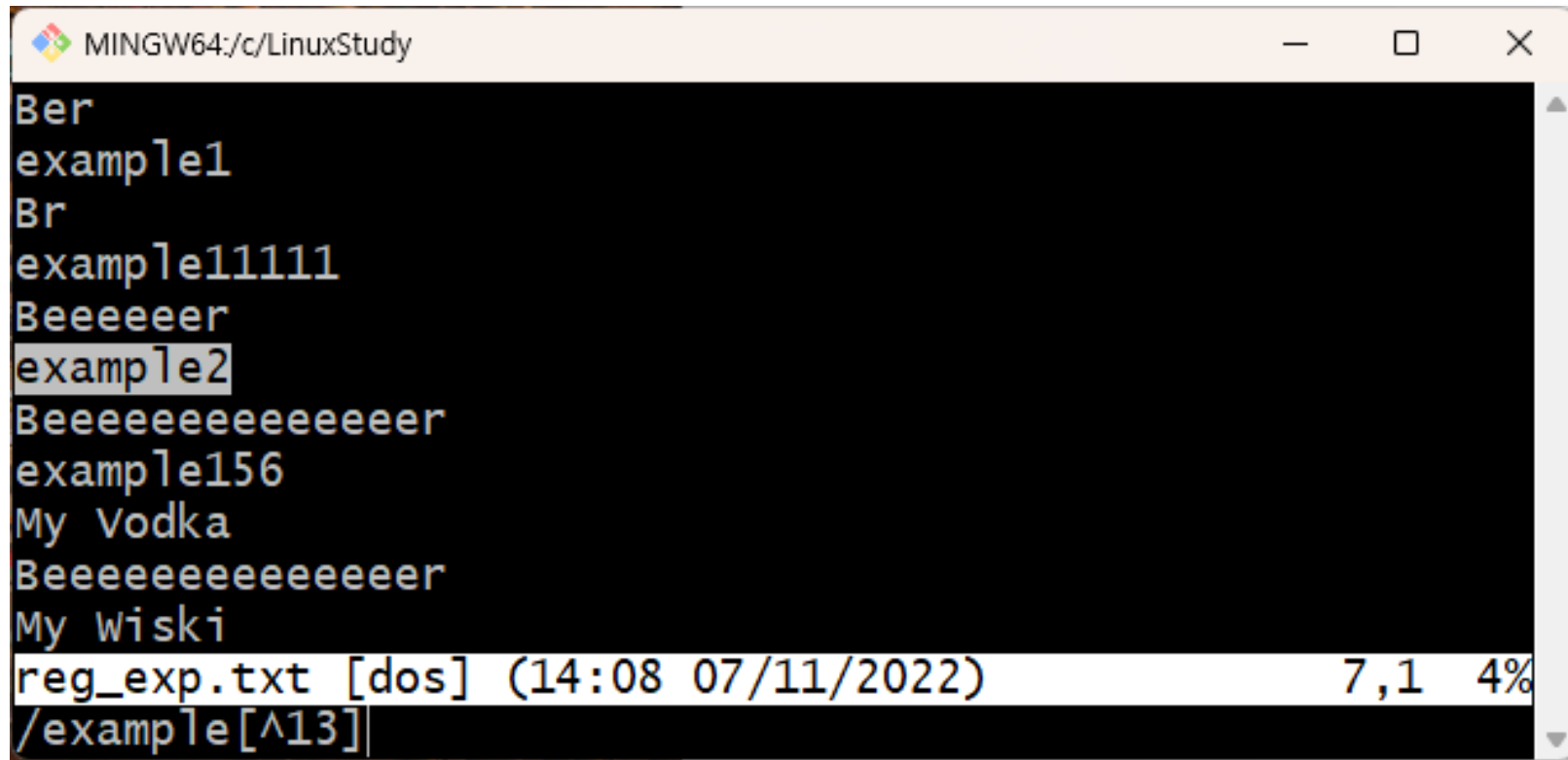
```
MINGW64:/c/LinuxStudy
My Vodka
Beeeeeeeeeeeeer
My Wiski
tefst
tefefefst
WineWine

This model is called mobilenet
example758
WineWineWine
tefefefefefefefst example135978
reg_exp.txt [dos] (14:08 07/11/2022) 15,1 37%
/W.*e
```

Vim에서의 정규 표현식 활용 방법

- (예제 3) “example” 뒤에 1,3이 등장하지 않는 문자열을 검색하시오.

➤ 명령 모드 → `/example[^13]`

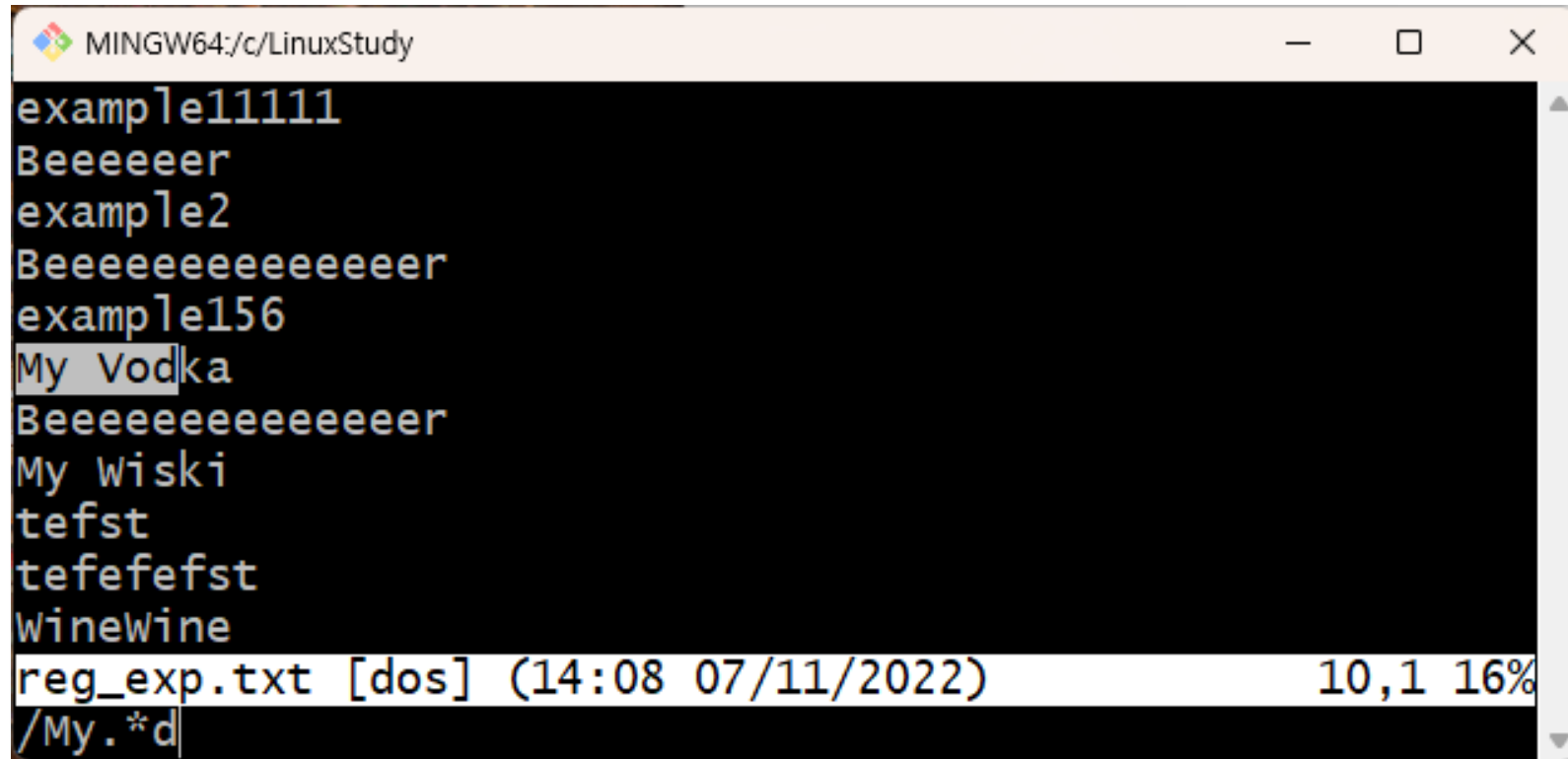


```
MINGW64:/c/LinuxStudy
Ber
example1
Br
example11111
Beeeeeer
example2
Beeeeeeeeeeeeeeer
example156
My Vodka
Beeeeeeeeeeeeeeer
My Wiski
reg_exp.txt [dos] (14:08 07/11/2022) 7,1 4%
/example[^13]
```

Vim에서의 정규 표현식 활용 방법

- (예제 4) “My”와 “d” 사이에 0개 이상의 문자를 갖는 문자열을 검색하시오.

➤ 명령 모드 → `/My.*d`

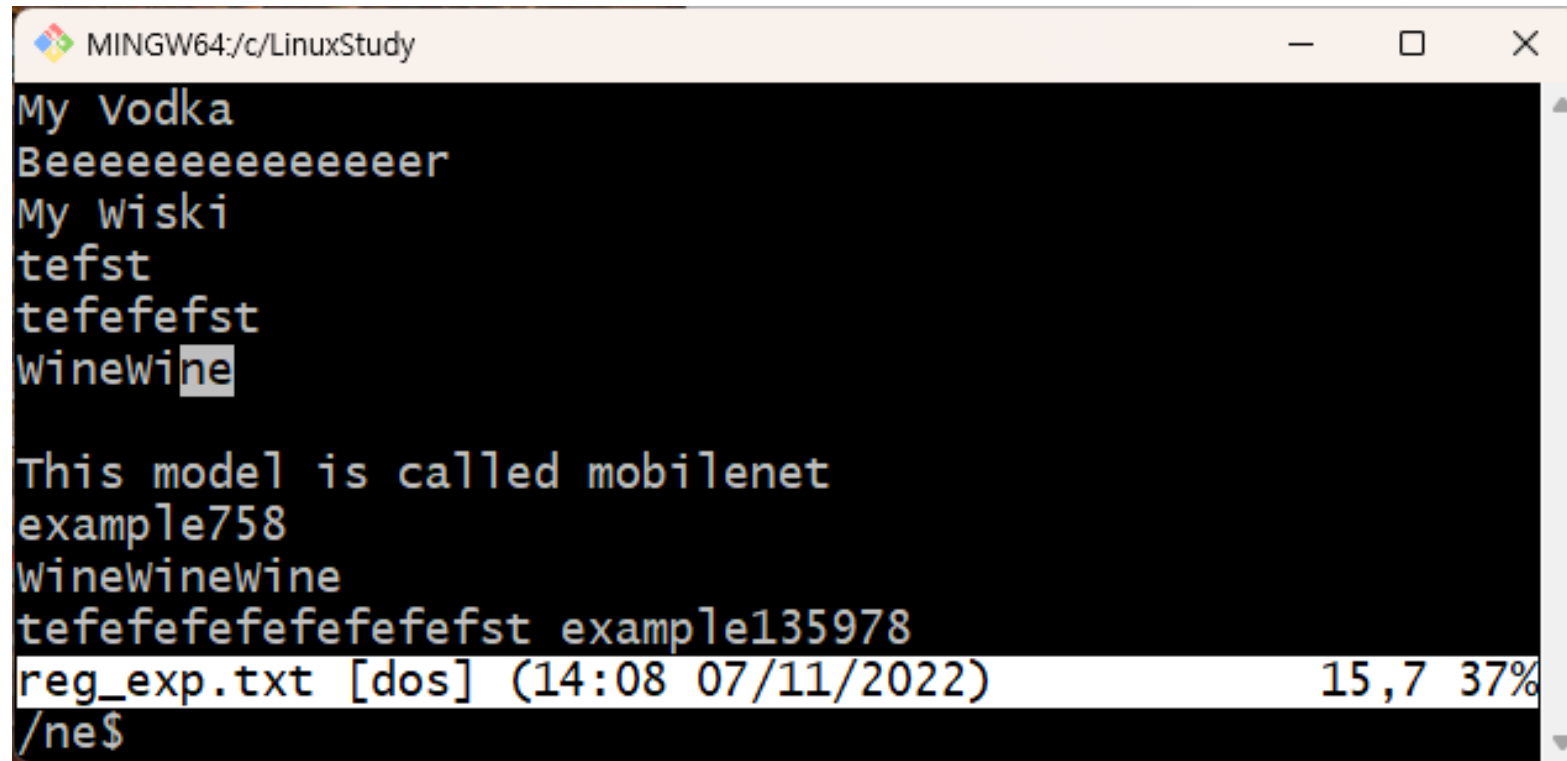


```
MINGW64:/c/LinuxStudy
example11111
Beeeeer
example2
Beeeeeeeeeeeeer
example156
My Vodka
Beeeeeeeeeeeeer
My wiski
tefst
tefefefst
winewine
reg_exp.txt [dos] (14:08 07/11/2022) 10,1 16%
/My.*d
```

Vim에서의 정규 표현식 활용 방법

- (예제 5) “ne”로 끝나는 문자열을 검색하시오.

➤ 명령 모드 → `/ne$`



```
MINGW64:/c/LinuxStudy
My Vodka
Beeeeeeeeeeer
My Wiski
tefst
tefefefst
wineWine

This model is called mobilenet
example758
wineWineWine
tefefefefefefefst example135978
reg_exp.txt [dos] (14:08 07/11/2022) 15,7 37%
/ne$
```

명령어 파이프

명령어 파이프(pipe)

- 여러 개의 명령어를 동시에 수행할 때, 이전 명령어의 결과를 다음 명령어의 입력으로 사용하는 기능
- 파이프 기호(|)를 사용하여 여러 명령어를 일렬로 연결하여 사용할 수 있음.
- 사용법: **명령어 1 | 명령어 2 | ... | 명령어 N**
 - 명령어 1의 출력 결과가 명령어 2의 인자(입력) 값으로 사용된다.
 - 명령어 2의 출력 결과가 명령어 3의 인자(입력) 값으로 사용된다.
 - ...
 - 명령어 N-1의 출력 결과가 명령어 N의 인자(입력) 값으로 사용된다.

명령어 파이프 연습 (1)

• 예제: “C:/LinuxStudy/” 디렉토리의 세부 정보를 출력하시오.

➤ 명령어: **ls -al**

```

MINGW64/c/LinuxStudy
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ ls -al
total 266
drwxr-xr-x 1 KyungSooKim 197121  0 Oct 12 16:06 ./
drwxr-xr-x 1 KyungSooKim 197121  0 Oct 17 13:19 ../
-rw-r--r-- 1 KyungSooKim 197121 20480 Oct 12 16:06 aaaaa.tar
-rw-r--r-- 1 KyungSooKim 197121 20480 Oct 11 18:23 archive1.tar
-rw-r--r-- 1 KyungSooKim 197121 102400 Oct 11 18:14 archive2.tar
-rw-r--r-- 1 KyungSooKim 197121 51200 Oct 11 18:15 archive3.tar
-rw-r--r-- 1 KyungSooKim 197121  497 Oct 11 19:00 compression1.tar.gz
-rw-r--r-- 1 KyungSooKim 197121  2186 Oct 11 19:03 compression2.tar.bz2
-rw-r--r-- 1 KyungSooKim 197121 11072 Oct 11 19:05 compression3.tar.xz
drwxr-xr-x 1 KyungSooKim 197121  0 Oct 17 19:57 documents/
-rw-r--r-- 1 KyungSooKim 197121  156 Oct 12 14:59 hello.c
-rw-r--r-- 1 KyungSooKim 197121  208 Oct 12 14:41 hello2.py
-rw-r--r-- 1 KyungSooKim 197121  542 Oct  5 16:20 main2.txt
-rw-r--r-- 1 KyungSooKim 197121  208 Oct  2 18:26 max_profit.txt
-rw-r--r-- 1 KyungSooKim 197121 9546 Oct 11 20:19 mobilenet.txt
  
```

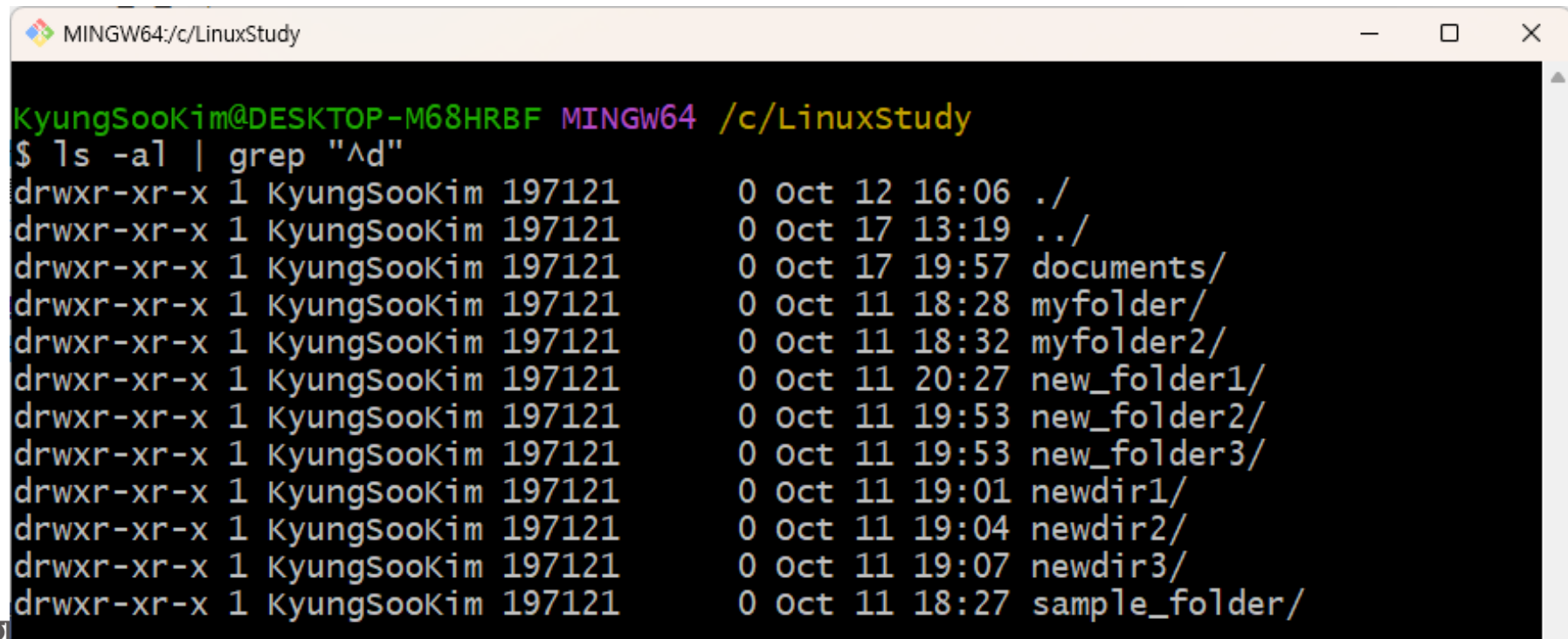
<참고> 각 열의 의미

- 첫 번째: 파일 종류와 허가권
 - ✓ d는 디렉토리, -는 일반파일,
 - ✓ 뒤의 rwxr-r-는 파일 허가권 의미
- 두 번째: 링크의 수
- 세 번째: 사용자 ID
- 네 번째: 파일을 소유한 그룹의 이름
- 다섯 번째: 파일 크기
- 여섯 번째: 파일의 수정 월
- 일곱 번째: 파일의 수정 일
- 여덟 번째: 파일의 수정 시간
- 아홉 번째: 파일의 이름

명령어 파이프 연습 (2)

- 예제 1: “C:/LinuxStudy/” 디렉토리의 세부 정보에서 “d”로 시작하는 모든 행을 출력하시오.

➤ 명령어: `ls -al | grep '^d'`



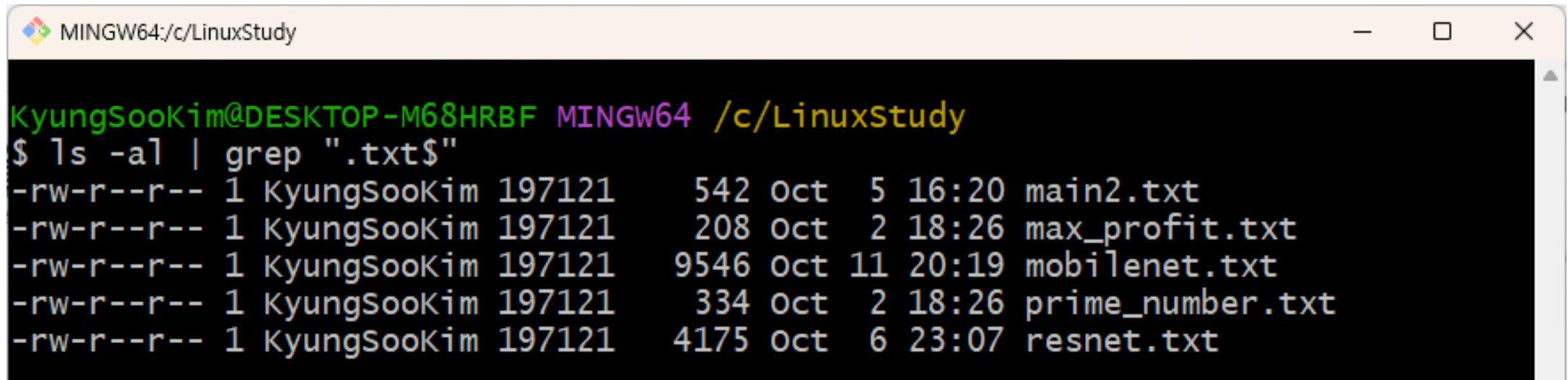
```
MINGW64:/c/LinuxStudy

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ ls -al | grep "^d"
drwxr-xr-x 1 KyungSooKim 197121 0 Oct 12 16:06 ./
drwxr-xr-x 1 KyungSooKim 197121 0 Oct 17 13:19 ../
drwxr-xr-x 1 KyungSooKim 197121 0 Oct 17 19:57 documents/
drwxr-xr-x 1 KyungSooKim 197121 0 Oct 11 18:28 myfolder/
drwxr-xr-x 1 KyungSooKim 197121 0 Oct 11 18:32 myfolder2/
drwxr-xr-x 1 KyungSooKim 197121 0 Oct 11 20:27 new_folder1/
drwxr-xr-x 1 KyungSooKim 197121 0 Oct 11 19:53 new_folder2/
drwxr-xr-x 1 KyungSooKim 197121 0 Oct 11 19:53 new_folder3/
drwxr-xr-x 1 KyungSooKim 197121 0 Oct 11 19:01 newdir1/
drwxr-xr-x 1 KyungSooKim 197121 0 Oct 11 19:04 newdir2/
drwxr-xr-x 1 KyungSooKim 197121 0 Oct 11 19:07 newdir3/
drwxr-xr-x 1 KyungSooKim 197121 0 Oct 11 18:27 sample_folder/
```

명령어 파이프 연습 (2)

- 예제 2: “C:/LinuxStudy/” 디렉토리의 세부 정보에서 확장자가 “txt”인 모든 파일들을 출력하시오.

➤ 명령어: `ls -al | grep '.txt$'`



```
MINGW64:/c/LinuxStudy

KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ ls -al | grep ".txt$"
-rw-r--r-- 1 KyungSookKim 197121    542 Oct  5 16:20 main2.txt
-rw-r--r-- 1 KyungSookKim 197121    208 Oct  2 18:26 max_profit.txt
-rw-r--r-- 1 KyungSookKim 197121   9546 Oct 11 20:19 mobilenet.txt
-rw-r--r-- 1 KyungSookKim 197121    334 Oct  2 18:26 prime_number.txt
-rw-r--r-- 1 KyungSookKim 197121   4175 Oct  6 23:07 resnet.txt
```

명령어 파이프 연습 (3)

- 예제 3: 현재 디렉토리의 전체 파일 목록을 파일 이름순으로 정렬하여 화면에 출력하시오.

➤ 명령어: `ls -al | sort -k 9`

```

MINGW64:/c/LinuxStudy
KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ ls -al | sort -k 9
total 266
drwxr-xr-x 1 KyungSookKim 197121      0 Oct 17 13:19 ../
drwxr-xr-x 1 KyungSookKim 197121      0 Oct 12 16:06 ./
-rw-r--r-- 1 KyungSookKim 197121    20480 Oct 12 16:06 aaaaa.tar
-rw-r--r-- 1 KyungSookKim 197121    20480 Oct 11 18:23 archive1.tar
-rw-r--r-- 1 KyungSookKim 197121   102400 Oct 11 18:14 archive2.tar
-rw-r--r-- 1 KyungSookKim 197121    51200 Oct 11 18:15 archive3.tar
-rw-r--r-- 1 KyungSookKim 197121     497 Oct 11 19:00 compression1.tar.gz
-rw-r--r-- 1 KyungSookKim 197121    2186 Oct 11 19:03 compression2.tar.bz2
-rw-r--r-- 1 KyungSookKim 197121    11072 Oct 11 19:05 compression3.tar.xz
drwxr-xr-x 1 KyungSookKim 197121      0 Oct 17 19:57 documents/
-rw-r--r-- 1 KyungSookKim 197121     156 Oct 12 14:59 hello.c
-rw-r--r-- 1 KyungSookKim 197121     208 Oct 12 14:41 hello2.py
-rw-r--r-- 1 KyungSookKim 197121     542 Oct  5 16:20 main2.txt
-rw-r--r-- 1 KyungSookKim 197121     208 Oct  2 18:26 max_profit.txt
-rw-r--r-- 1 KyungSookKim 197121    9546 Oct 11 20:19 mobilenet.txt
  
```

명령어 파이프 연습 (3)

- 예제 4: 현재 디렉토리의 전체 파일 목록을 파일 이름의 내림차순으로 정렬하여 화면에 출력하시오.

➤ 명령어: `ls -al | sort -rk 9`

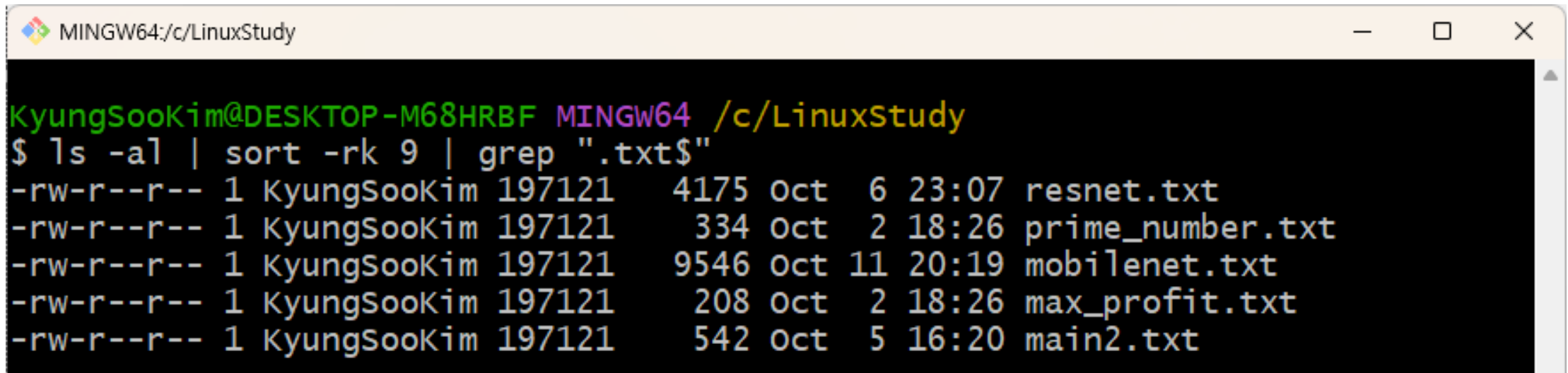
```
MINGW64:/c/LinuxStudy

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ ls -al | sort -rk 9
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 18:27 sample_folder/
-rw-r--r-- 1 KyungSooKim 197121    4175 Oct  6 23:07 resnet.txt
-rw-r--r-- 1 KyungSooKim 197121     334 Oct  2 18:26 prime_number.txt
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 19:07 newdir3/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 19:04 newdir2/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 19:01 newdir1/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 19:53 new_folder3/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 19:53 new_folder2/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 20:27 new_folder1/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 18:32 myfolder2/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 18:28 myfolder/
-rw-r--r-- 1 KyungSooKim 197121   9546 Oct 11 20:19 mobilenet.txt
-rw-r--r-- 1 KyungSooKim 197121    208 Oct  2 18:26 max_profit.txt
-rw-r--r-- 1 KyungSooKim 197121    542 Oct  5 16:20 main2.txt
-rw-r--r-- 1 KyungSooKim 197121    208 Oct 12 14:41 hello2.py
-rw-r--r-- 1 KyungSooKim 197121    156 Oct 12 14:59 hello.c
```

명령어 파이프 연습 (4)

- 예제 5: 현재 디렉토리의 전체 파일 목록을 파일 이름의 내림차순으로 정렬한 결과에서 텍스트 파일만을 화면에 출력하시오.

➤ 명령어: `ls -al | sort -rk 9 | grep '.txt$'`



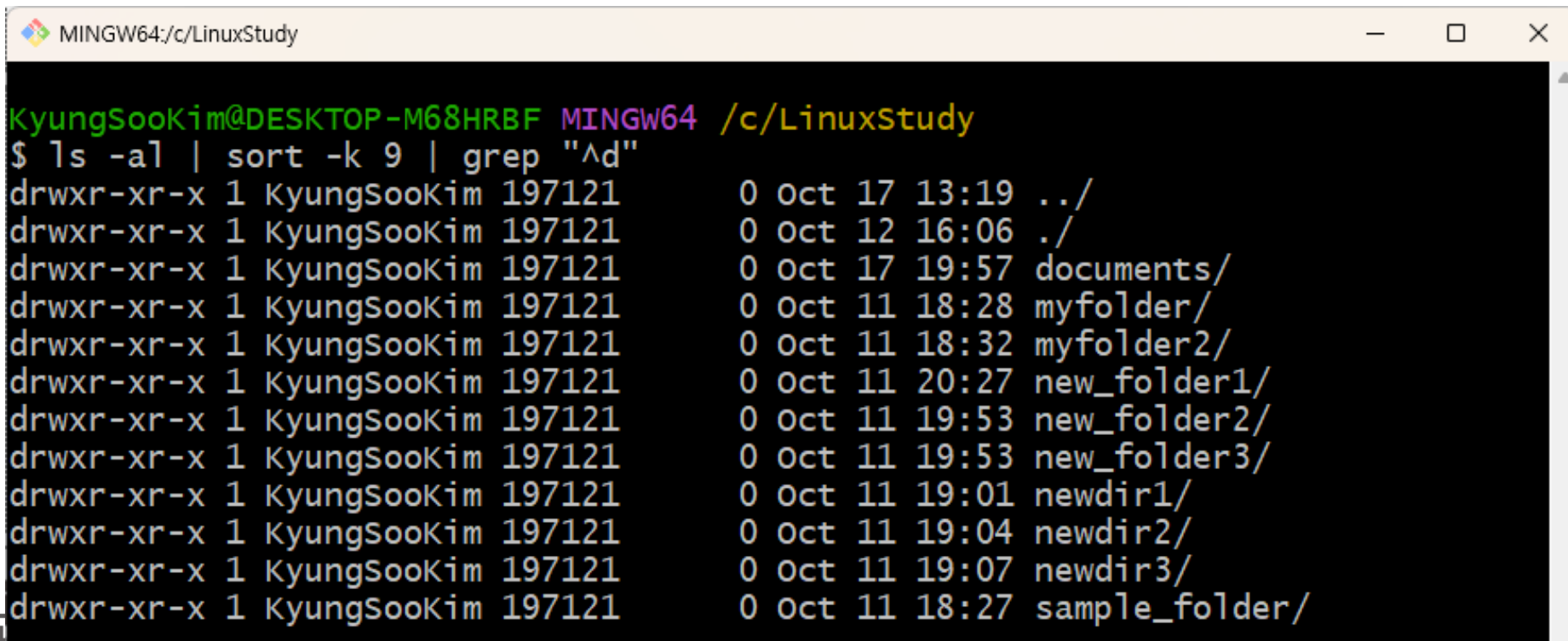
```
MINGW64:/c/LinuxStudy

KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ ls -al | sort -rk 9 | grep ".txt$"
-rw-r--r-- 1 KyungSookKim 197121 4175 Oct 6 23:07 resnet.txt
-rw-r--r-- 1 KyungSookKim 197121 334 Oct 2 18:26 prime_number.txt
-rw-r--r-- 1 KyungSookKim 197121 9546 Oct 11 20:19 mobilenet.txt
-rw-r--r-- 1 KyungSookKim 197121 208 Oct 2 18:26 max_profit.txt
-rw-r--r-- 1 KyungSookKim 197121 542 Oct 5 16:20 main2.txt
```


명령어 파이프 연습 (4)

- 예제 6: 현재 디렉토리의 전체 파일 목록을 파일 이름의 오름차순으로 정렬한 결과에서 디렉토리만을 화면에 출력하시오.

➤ 명령어: `ls -al | sort -k 9 | grep '^d'`



```
MINGW64:/c/LinuxStudy

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ ls -al | sort -k 9 | grep "^d"
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 17 13:19 ../
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 12 16:06 ./
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 17 19:57 documents/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 18:28 myfolder/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 18:32 myfolder2/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 20:27 new_folder1/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 19:53 new_folder2/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 19:53 new_folder3/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 19:01 newdir1/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 19:04 newdir2/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 19:07 newdir3/
drwxr-xr-x 1 KyungSooKim 197121      0 Oct 11 18:27 sample_folder/
```

명령어 파이프 연습 (4)

- 예제 7: 현재 수행중인 프로세스 목록을 프로세스 경로명의 내림차순으로 정렬한 결과에서 bash와 관련된 행을 화면에 출력하시오.

➤ 명령어: `ps | sort -rk 8 | grep '/bash$'`

```

MINGW64:/c/LinuxStudy

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ ps
  PID   PPID   PGID   WINPID   TTY      UID     STIME  COMMAND
   927     1    927    13740    ?        197609  17:26:56 /usr/bin/mintty
  1571   928   1571   10992   pty0      197609  21:23:28 /usr/bin/ps
   928   927    928   15196   pty0      197609  17:26:57 /usr/bin/bash

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ ps | sort -rk 8 | grep "/bash$"
   928   927    928   15196   pty0      197609  17:26:57 /usr/bin/bash
  
```


아카이브와 압축 명령

아카이브와 압축

- 리눅스에서는 아카이브와 압축을 용도에 따라 구분하여 사용함.
- 아카이브(archive)
 - Tape ARchive
 - Tape 저장 장치에 데이터를 기록하기 위해서 사용되는 방법으로 UNIX 계열에서 널리 사용되는 표준 형식
 - 여러 개의 파일을 하나의 파일로 아카이빙(archiving), 즉 묶는 기능을 수행
 - 이 때, 압축률은 0% → 단순히 여러 파일을 하나로 묶는 기능만 수행.
 - 단, 추가 옵션을 통해 실제 압축을 수행하는 기능 또한 제공함.
 - 종류: tar, dd, rsync, ... 등

아카이브와 압축

- 리눅스에서는 아카이브와 압축을 용도에 따라 구분하여 사용함.
- 압축(compression)
 - 여러 개의 파일을 하나로 묶음과 동시에 파일의 크기를 최소화한 것.
 - 아카이브 방법인 tar와는 달리 실제 압축을 수행.
 - 대표적인 압축 파일 형식: zip, rar, ... 등

tar

- 리눅스에서 널리 활용되는 파일 아카이브 유틸리티
- 기본 사용법

➤ **tar [옵션] 아카이브 파일명.tar 파일 이름**

(예제 1) **tar -cvf file_name.tar learning.txt**

(예제 2) **tar -cvf file_name.tar ./documents/**

(예제 3) **tar -cvf file_name.tar ***

tar - 주요 옵션

tar [옵션] [아카이브 파일명] 대상 파일명

옵션	설명
-c	새로운 tar 파일을 생성한다. (= create)
-t	tar 파일의 내용을 출력한다. (=list)
-x	현재 아카이브 파일을 해제한다. (=extract)
-r	현재 아카이브 파일에 새로운 파일을 추가한다.
-u	현재 아카이브 파일에서 수정된 내용을 업데이트한다.

tar - 주요 옵션

tar [옵션] [아카이브 파일명] 대상 파일명

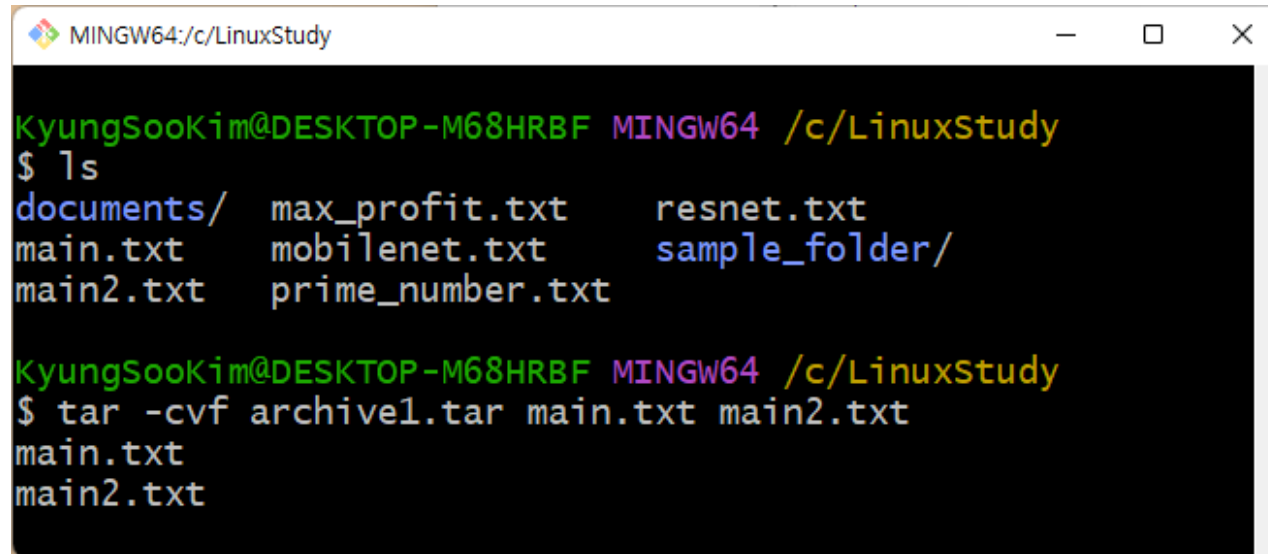
옵션	설명
-f	파일을 읽어들인다. (=file)
-v	현재 처리되는 파일의 정보를 상세하게 출력한다. (=visual)
-p	파일 복구 시 원래의 접근 권한을 유지한다.
-j	bzip2 형식으로 압축 또는 해제한다.
-J	xz 형식으로 압축 또는 해제한다.
-z	gzip 형식으로 압축 또는 해제한다.

tar 사용 방법 1 - 아카이브 생성하기

- **tar -cvf 아카이브 파일명.tar 압축할 대상파일명**

- -c : 새로운 tar 파일을 생성함.
- -v : 현재 처리하는 파일의 정보를 상세히 출력함.
- 압축파일명에는 확장자 “*.tar”를 함께 기재할 것을 권장함.

(예제 1) **tar -cvf archive1.tar main.txt main2.txt**



```
MINGW64:/c/LinuxStudy

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ ls
documents/  max_profit.txt  resnet.txt
main.txt    mobilenet.txt   sample_folder/
main2.txt   prime_number.txt

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -cvf archive1.tar main.txt main2.txt
main.txt
main2.txt
```

tar 사용 방법 1 - 아카이브 생성하기

- **tar -cvf 아카이브 파일명.tar 압축할 대상파일명**

- -c : 새로운 tar 파일을 생성함.
- -v : 현재 처리하는 파일의 정보를 상세히 출력함.
- 압축파일명에는 확장자 "*.tar"를 함께 기재할 것을 권장함.

(예제 2) **tar -cvf archive2.tar ***

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ ls
archive1.tar  main2.txt      prime_number.txt
documents/    max_profit.txt resnet.txt
main.txt      mobilenet.txt  sample_folder/

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -cvf archive2.tar *
archive1.tar
documents/
documents/.hello.c.swp
documents/hello.c
documents/learning.txt
```


tar 사용 방법 1 – 아카이브 생성하기

- **tar -cvf 아카이브 파일명.tar 압축할 대상파일명**

- -c : 새로운 tar 파일을 생성함.
- -v : 현재 처리하는 파일의 정보를 상세히 출력함.
- 압축파일명에는 확장자 “*.tar”를 함께 기재할 것을 권장함.

(예제 3) **tar -cvf archive3.tar documents**

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ ls
archive1.tar  main2.txt      resnet.txt
archive2.tar  max_profit.txt sample_folder/
documents/    mobilenet.txt
main.txt      prime_number.txt

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -cvf archive3.tar documents
documents/
documents/.hello.c.swp
documents/hello.c
documents/learning.txt
```

tar 사용 방법 2 - 아카이브 내용 보기

- **tar -tvf 아카이브 파일명.tar**

- -t : 아카이브의 내용을 확인할 시에 내부 파일들을 화면에 테이블(table) 형태로 출력하기 위해 옵션 “t”를 사용함.
- -v : 현재 처리중인 파일의 정보를 상세히 출력함.

(예제 1) tar -tvf archive1.tar

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -tvf archive1.tar
-rw-r--r-- KyungSooKim/197121 9688 2022-10-11 18:11 main.txt
-rw-r--r-- KyungSooKim/197121 542 2022-10-05 16:20 main2.txt

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$
```

tar 사용 방법 2 - 아카이브 내용 보기

- **tar -tvf 아카이브 파일명.tar**

- -t : 아카이브의 내용을 확인할 시에 내부 파일들을 화면에 테이블(table) 형태로 출력하기 위해 옵션 “t”를 사용함.
- -v : 현재 처리중인 파일의 정보를 상세히 출력함.

(예제 2) tar -tvf archive3.tar

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -tvf archive3.tar
drwxr-xr-x KyungSooKim/197121 0 2022-10-06 23:04 documents/
-rw-r--r-- KyungSooKim/197121 12288 2022-10-06 23:37 documents/.hello.c.swp
-rw-r--r-- KyungSooKim/197121 137 2022-10-06 23:04 documents/hello.c
-rw-r--r-- KyungSooKim/197121 4055 2022-10-05 16:39 documents/learning.txt
-rw-r--r-- KyungSooKim/197121 13046 2022-10-02 20:25 documents/neuralnet.txt
-rw-r--r-- KyungSooKim/197121 5508 2022-10-02 20:28 documents/neural_types.txt
-rw-r--r-- KyungSooKim/197121 54 2022-10-05 22:03 documents/resnet.txt
-rw-r--r-- KyungSooKim/197121 2984 2022-10-02 20:28 documents/theory.txt
```

tar 사용 방법 3 - 아카이브에 파일 또는 디렉토리 추가 (1)

- 현재 아카이브 파일에 아카이브에 파일 또는 디렉토리를 추가
- **tar -rvf 아카이브 파일명.tar 새롭게 추가되는 파일명**
 - -r : 새로운 파일을 아카이브 파일에 추가하는 옵션
 - 새롭게 추가하려는 파일이 기존 아카이브에 존재하는 경우, 덮어쓰지 않고, 새로운 버전 이름을 붙여서 추가시킴.
 - 아카이브 해제 시에는 최신 파일로 해제됨.

(예제) `tar -rvf archive1.tar max_profit.txt`

```
KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -rvf archive1.tar max_profit.txt
max_profit.txt

KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -tvf archive1.tar
-rw-r--r-- KyungSookKim/197121 9688 2022-10-11 18:11 main.txt
-rw-r--r-- KyungSookKim/197121 542 2022-10-05 16:20 main2.txt
-rw-r--r-- KyungSookKim/197121 208 2022-10-02 18:26 max_profit.txt
```

tar 사용 방법 4 - 아카이브에 파일 또는 디렉토리 추가 (2)

- **tar -uvf 아카이브 파일명.tar 새롭게 추가되는 파일명**

➤ “-r” 대신에 “-u” 옵션을 사용하면, 새롭게 추가하려는 파일이 기존 아카이브에 존재하는 경우, 버전을 비교하여 추가하는 파일이 최신 버전인 경우에만 업데이트를 수행함.

(예제) `tar -uvf archive1.tar max_profit.txt`

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -uvf archive1.tar max_profit.txt

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -uvf archive1.tar resnet.txt
resnet.txt

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -tvf archive1.tar
-rw-r--r-- KyungSooKim/197121 9688 2022-10-11 18:11 main.txt
-rw-r--r-- KyungSooKim/197121 542 2022-10-05 16:20 main2.txt
-rw-r--r-- KyungSooKim/197121 208 2022-10-02 18:26 max_profit.txt
-rw-r--r-- KyungSooKim/197121 4175 2022-10-06 23:07 resnet.txt
```

tar 사용 방법 5 – 아카이브 내 일부 파일의 삭제

- 아카이브 내에서 일부 파일만 삭제
- **tar --delete --file=아카이브 파일명.tar 삭제할 파일명**
 - 옵션 사용시 “-” 가 아닌 “--”임에 유의할 것.

(예제) `tar --delete --file=archive1.tar resnet.txt max_profit.txt`

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -tvf archive1.tar
-rw-r--r-- KyungSooKim/197121 9688 2022-10-11 18:11 main.txt
-rw-r--r-- KyungSooKim/197121 542 2022-10-05 16:20 main2.txt
-rw-r--r-- KyungSooKim/197121 208 2022-10-02 18:26 max_profit.txt
-rw-r--r-- KyungSooKim/197121 4175 2022-10-06 23:07 resnet.txt

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar --delete --file=archive1.tar resnet.txt max_profit.txt

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -tvf archive1.tar
-rw-r--r-- KyungSooKim/197121 9688 2022-10-11 18:11 main.txt
-rw-r--r-- KyungSooKim/197121 542 2022-10-05 16:20 main2.txt
```

tar 사용 방법 6 – 아카이브 해제

- 현재 디렉토리에 아카이브를 해제 (= 아카이브 파일 풀림)
- **tar -xvf 아카이브 파일명.tar**
 - 아카이브 파일이 풀릴 디렉토리를 지정하지 않으므로, 아카이브가 위치해 있는 디렉토리에 아카이브 파일이 풀림.

(예제)

LinuxStudy 디렉토리 내에
myfolder 디렉토리 생성 후

archive1.tar 파일을 myfolder 디렉토리로 복사

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ mkdir myfolder

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ cp archive1.tar myfolder

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ cd myfolder

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy/myfolder
$ ls
archive1.tar
```


tar 사용 방법 6 – 아카이브 해제

- 현재 디렉토리에 아카이브를 해제 (= 아카이브 파일 풀림)
- **tar -xvf 아카이브 파일명.tar**
 - 아카이브 파일이 풀릴 디렉토리를 지정하지 않으므로, 아카이브가 위치해 있는 디렉토리에 아카이브 파일이 풀림.

(예제 – 계속)

myfolder 디렉토리로 이동 후
tar -xvf archive1.tar 명령 실행

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy/myfolder
$ tar -xvf archive1.tar
main.txt
main2.txt

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy/myfolder
$ ls
archive1.tar  main.txt  main2.txt

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy/myfolder
$
```


tar 사용 방법 7 – 특정 디렉토리에 아카이브 해제

- 특정 디렉토리에 아카이브 파일을 해제
- **tar -xvf 아카이브 파일명.tar -C 디렉토리 경로**
 - 디렉토리 경로 옵션에서 “C”는 대문자임에 유의할 것.

(예제)

LinuxStudy 디렉토리 내에 myfolder2 디렉토리 생성 후
tar -xvf archive2.tar -C ./myfolder2/ 명령 실행

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ mkdir myfolder2

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -xvf archive2.tar -C ./myfolder2/
archive1.tar
documents/
documents/.hello.c.swp
documents/hello.c
documents/learning.txt
documents/neuralnet.txt
documents/neural_types.txt
documents/resnet.txt
documents/theory.txt
main.txt
main2.txt
max_profit.txt
```

tar 사용 방법 7 – 특정 디렉토리에 아카이브 해제

- 특정 디렉토리에 아카이브 파일을 해제
- **tar -xvf 아카이브 파일명.tar -C 디렉토리 경로**
 - 디렉토리 경로 옵션에서 “C”는 대문자임에 유의할 것.

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ cd myfolder2

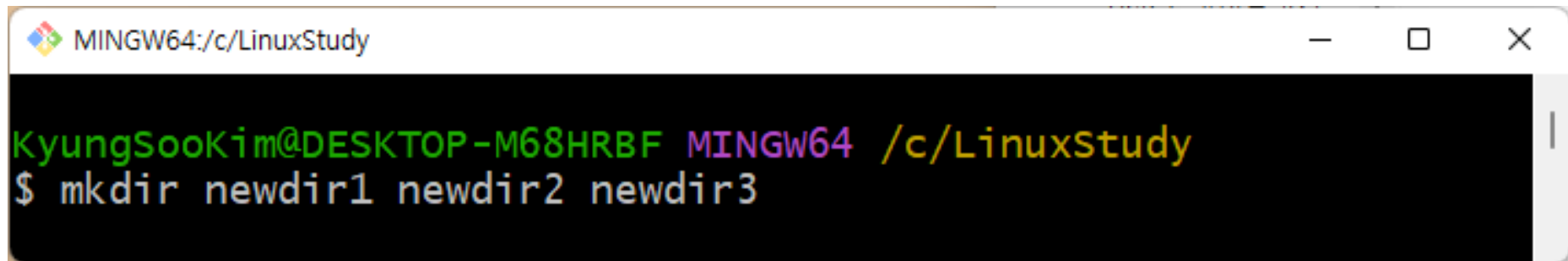
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy/myfolder2
$ ls
archive1.tar  main2.txt      prime_number.txt
documents/   max_profit.txt resnet.txt
main.txt      mobilenet.txt  sample_folder/

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy/myfolder2
$ |
```

myfolder2 디렉토리로 이동하여 “ls” 명령을 실행하여
archive2.tar 파일의 내용이 정상적으로 풀렸는지 확인

tar를 사용한 압축 방법

- 리눅스에서 사용되는 주요 압축 방식
 - ✓ gzip, bzip2, xz
 - ✓ “gzip → bzip2 → xz” 순서대로 압축률 향상, 속도는 감소
- [실습 준비] tar를 사용한 압축 실습을 위해 “C:/LinuxStudy/” 경로에 아래와 같이 3개의 새로운 폴더를 생성하시오.
 - **mkdir newdir1 newdir2 newdir3**



```
MINGW64:/c/LinuxStudy


KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ mkdir newdir1 newdir2 newdir3
```

tar를 활용한 압축과 해제 1 - gzip

- gzip: 가장 오래되고 속도가 빠른 압축 방법

➤ 압축 방법: `tar -cvzf [아카이브명.tar.gz] 압축 대상 파일명`

(예제) `tar -cvzf compression1.tar.gz main.txt main2.txt`



```
MINGW64:/c/LinuxStudy

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -cvzf compression1.tar.gz main.txt main2.txt
main.txt
main2.txt

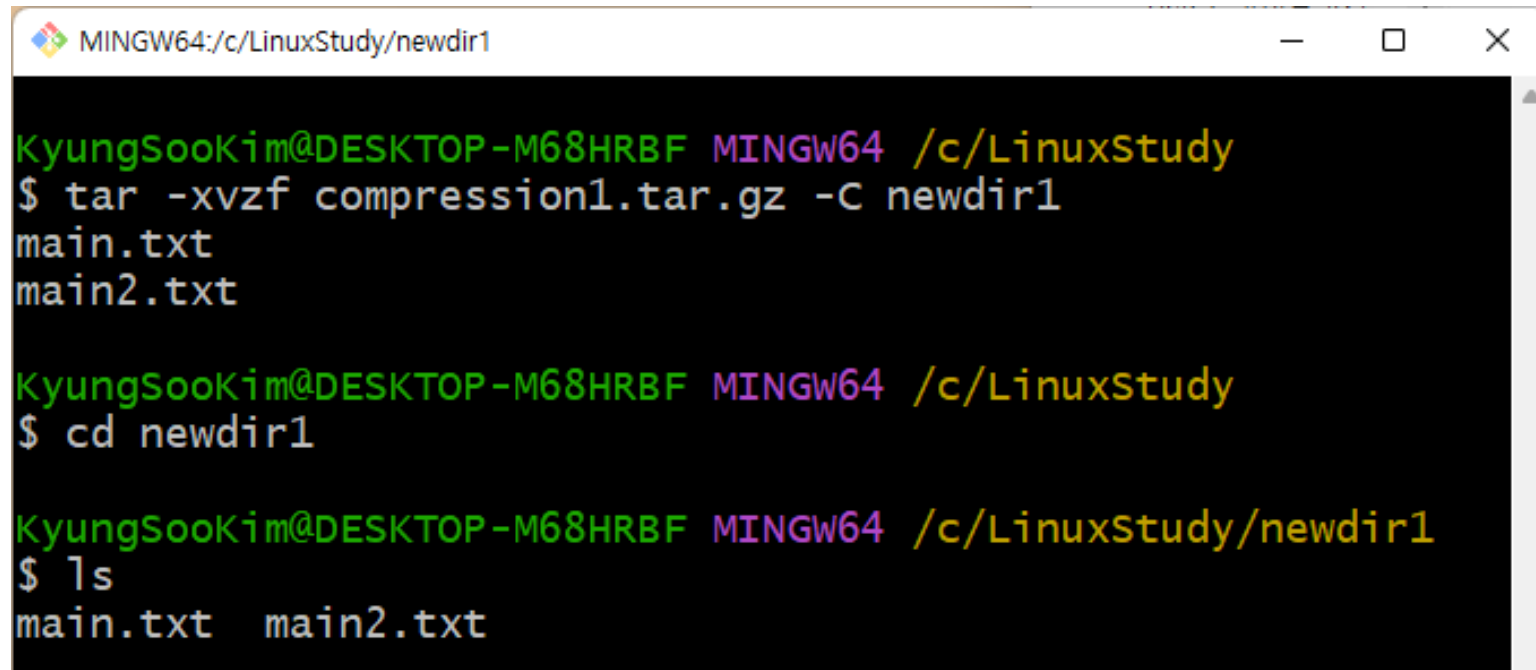
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$
```

tar를 활용한 압축과 해제 1 - gzip

- gzip: 가장 오래되고 속도가 빠른 압축 방법

➤ 해제 방법: `tar -xvzf 아카이브명.tar.gz [-C 압축 해제할 디렉토리]`

(예제) `tar -xvzf compression1.tar.gz -C newdir1`



```
MINGW64:/c/LinuxStudy/newdir1

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -xvzf compression1.tar.gz -C newdir1
main.txt
main2.txt

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ cd newdir1

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy/newdir1
$ ls
main.txt  main2.txt
```

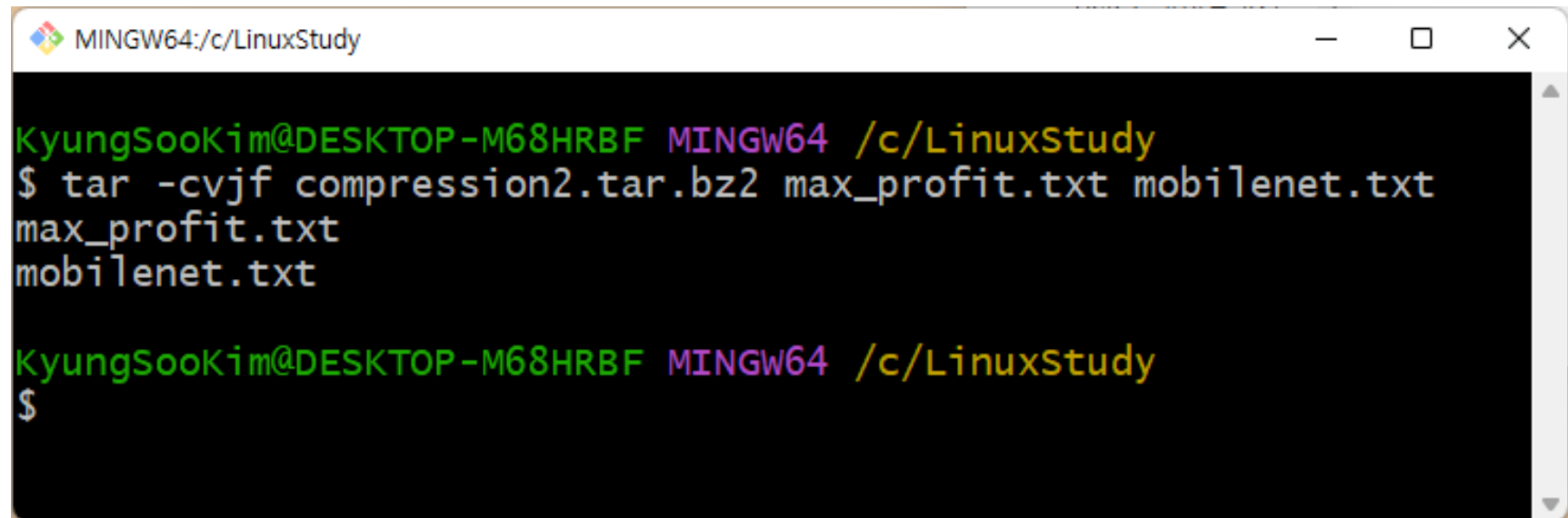
tar를 활용한 압축과 해제 2 - bzip2

- bzip2: 압축률이 gzip 보다 높으며, 고용량 처리가 가능함.

➤ 압축 방법: `tar -cvjf [아카이브명.tar.bz2] 압축 대상 파일명`

➤ 상기 옵션에서 “j”는 소문자임에 유의할 것.

(예제) `tar -cvjf compression2.tar.bz2 max_profit.txt mobilenet.txt`



```
MINGW64:/c/LinuxStudy
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -cvjf compression2.tar.bz2 max_profit.txt mobilenet.txt
max_profit.txt
mobilenet.txt
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$
```

tar를 활용한 압축과 해제 2 - bzip2

- bzip2: 압축률이 gzip 보다 높으며, 고용량 처리가 가능함.

- 해제 방법: `tar -xvjf 아카이브명.tar.bz2 [-C 압축 해제할 디렉토리]`
- 상기 옵션에서 “j”는 소문자임에 유의할 것.

(예제) `tar -xvjf compression2.tar.bz2 -C newdir2`

```
MINGW64:/c/LinuxStudy/newdir2

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -xvjf compression2.tar.bz2 -C newdir2
max_profit.txt
mobilenet.txt

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ cd newdir2

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy/newdir2
$ ls
max_profit.txt  mobilenet.txt
```

tar를 활용한 압축과 해제 3 - xz

- xz: 가장 최근에 개발된 압축 방식으로 압축률이 제일 우수함.

➤ 압축 방법: **tar -cvJf [아카이브명.tar.xz] 압축 대상 파일명**

➤ 상기 옵션에서 “J”는 대문자임에 유의할 것.

(예제) **tar -cvJf compression3.tar.xz
resnet.txt prime_number.txt documents**



```
MINGW64:/c/LinuxStudy

KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -cvJf compression3.tar.xz resnet.txt prime_number.txt documents
resnet.txt
prime_number.txt
documents/
documents/.hello.c.swp
documents/hello.c
documents/learning.txt
documents/neuralnet.txt
documents/neural_types.txt
```


tar를 활용한 압축과 해제 3 - xz

- xz: 가장 최근에 개발된 압축 방식으로 압축률이 제일 우수함.

➤ 해제 방법: `tar -xvJf 아카이브명.tar.xz [-C 압축 해제할 디렉토리]`

➤ 상기 옵션에서 “J”는 대문자임에 유의할 것.

(예제) `tar -xvJf compression3.tar.xz -C newdir3`

```
MINGW64:/c/LinuxStudy
KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ tar -xvJf compression3.tar.xz -C newdir3
resnet.txt
prime_number.txt
documents/
documents/.hello.c.swp
documents/hello.c
documents/learning.txt
documents/neuralnet.txt
documents/neural_types.txt
documents/resnet.txt
```

```
MINGW64:/c/LinuxStudy/newdir3
KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ cd newdir3

KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy/newdir3
$ ls
documents/ prime_number.txt resnet.txt
```

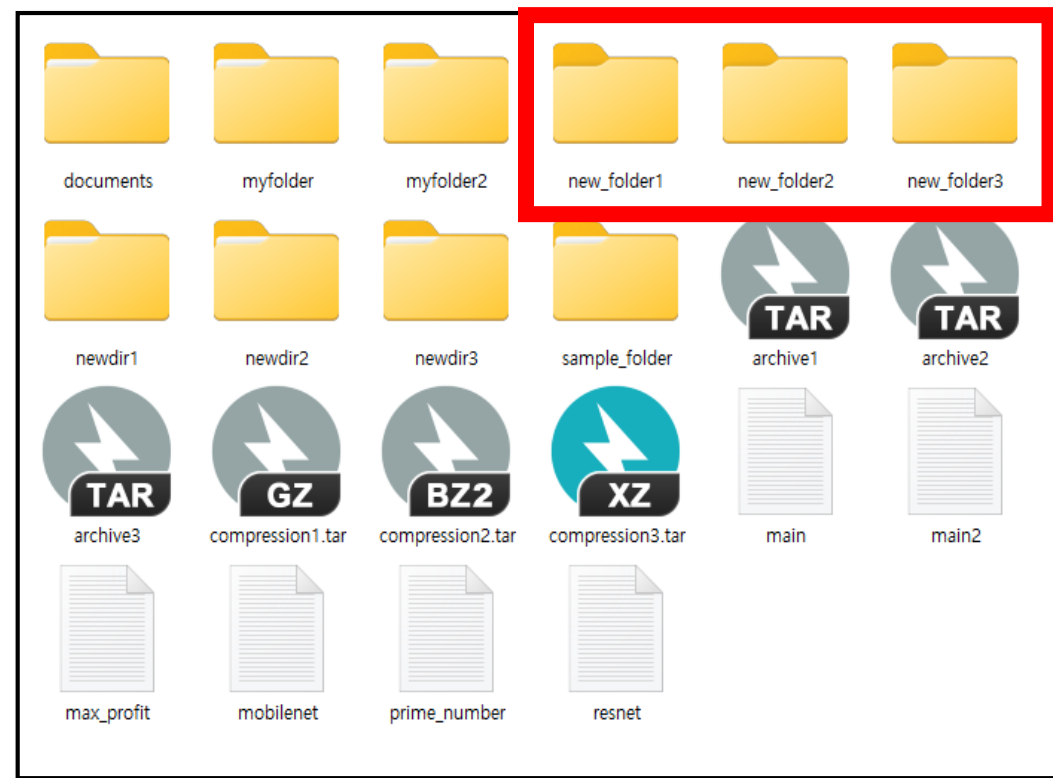
압축 명령

- tar를 사용하지 않고 바로 압축하는 명령: `gzip`, `bzip2`, `xz`
- 실습을 위해 “C:/LinuxStudy/” 경로에 아래와 같이 세 개의 새로운 폴더 “new_folder1”, “new_folder2”, “new_folder3”를 생성함.

```
MINGW64:/c/LinuxStudy

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ mkdir new_folder1 new_folder2 new_folder3

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ |
```



압축 명령 gzip

- gzip 형식으로 파일을 압축하는 기능으로 여러 파일을 하나로 묶는 기능은 없음.
 - 즉, 하나의 파일에 대해서만 압축을 수행함.
 - 압축 명령: **gzip [옵션] [압축 대상 파일명]**
 - 해제 명령: **gunzip [옵션] [압축파일명.gz]**
 - 보기 명령: **zcat [압축파일명]**
- 주요 옵션
 - **-d**: 압축 해제 → “gunzip” 명령어 대신 “gzip” 명령어를 사용하여 압축 해제할 때 사용
 - **-1**: 압축시간 줄이기(※ 단, 압축률이 떨어짐)
 - **-6**: 압축 속도와 압축률 사이의 절충된 설정으로 기본 설정(default option)
 - **-9**: 파일을 최대한으로 압축 (※ 단, 압축 시간이 오래 걸림)
 - **-l**: 압축파일에 대한 정보 출력
 - **-r**: 압축대상이 디렉토리인 경우 하위 디렉토리까지 모두 압축
 - **-v**: 압축 진행과정을 화면에 자세히 출력함.

압축 명령 gzip

- (예제 1) 파일 압축: **gzip -9v mobilenet.txt**

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ gzip -9v mobilenet.txt
mobilenet.txt: 80.7% -- replaced with mobilenet.txt.gz

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ |
```



- (예제 2) 압축 해제: **gunzip mobilenet.txt.gz**

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ gunzip mobilenet.txt.gz

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ |
```



압축 명령 bzip2

- tar를 사용하지 않고 bzip2 형식으로 바로 압축하는 기능
 - “gzip”과 마찬가지로, 하나의 파일에 대해서만 압축을 수행함.
 - 압축 명령: **bzip2 [옵션] [압축파일명] [압축 대상 파일명]**
 - 해제 명령: **bunzip2 [옵션] [압축파일명]**
- 주요 옵션
 - **-d**: 압축 해제 → “bunzip2” 명령어 대신 “bzip2” 명령어를 사용하여 압축 해제할 때 사용
 - **-1**: 압축시간 줄이기 (단, 압축률이 떨어짐)
 - **-6**: 압축 속도와 압축률 사이의 절충된 설정으로 기본 설정(default option)
 - **-9**: 파일을 최대로 압축 (단, 압축 시간이 오래 걸림)
 - **-r**: 압축대상이 디렉토리인 경우 하위 디렉토리까지 모두 압축
 - **-v**: 압축 진행과정을 화면에 자세히 출력함.

압축 명령 bzip2

- (예제 1) 파일 압축: **bzip2 -k9v archive.tar**

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ bzip2 -k9v archive2.tar
archive2.tar: 7.728:1, 1.035 bits/byte, 87.06% saved, 102400 in, 13250 out.
```

- (예제 2) 압축 해제: **bunzip2 archive2.tar.bz2**

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ mv archive2.tar.bz2 new_folder1

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ cd new_folder1

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy/new_folder1
$ bunzip2 archive2.tar.bz2
```

압축 명령 xz

- tar를 사용하지 않고 xz 형식으로 바로 압축하는 기능
 - “gzip”, “bzip2”와 마찬가지로, 하나의 파일에 대해서만 압축을 수행함.
 - 압축 명령: **xz [옵션] [압축파일명] [압축 대상 파일명]**
 - 해제 명령: **unxz [옵션] [압축파일명]**
- 주요 옵션
 - -d: 압축 해제 → “unxz” 명령어 대신 “xz” 명령어를 사용하여 압축 해제할 때 사용
 - -1: 압축시간 줄이기 (단, 압축률이 떨어짐)
 - -6: 압축 속도와 압축률 사이의 절충된 설정으로 기본 설정(default option)
 - -9: 파일을 최대한으로 압축 (단, 압축 시간이 오래 걸림)

압축 명령 xz

- (예제 1) 파일 압축: **xz neuralnet.txt**

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy
$ cd documents

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy/documents
$ xz neuralnet.txt
```

- (예제 2) unxz 명령어를 활용한 압축 해제: **unxz neuralnet.txt.xz**

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy/documents
$ unxz neuralnet.txt.xz
```

- (예제 3) xz 명령어를 활용한 압축 해제: **xz -d neuralnet.txt.xz**

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/LinuxStudy/documents
$ xz -d neuralnet.txt.xz
```


Q & A