

2022 A Basic CS skill: ABC Winter School

Shell & Linux 2

Team 8

2022 / 01 / 19



ULSAN NATIONAL INSTITUTE OF
SCIENCE AND TECHNOLOGY

VPN

- <https://vpn.unist.ac.kr>
- 위 링크에서 VPN User Guide 파일을 참고하여 VPN 설치
- 설치 된 PulseSecure를 실행하고 사용자 이름/암호에 Unist Portal ID / 비밀번호 입력
- 사용에 문제가 있는 경우 톡방에 문의하기!!

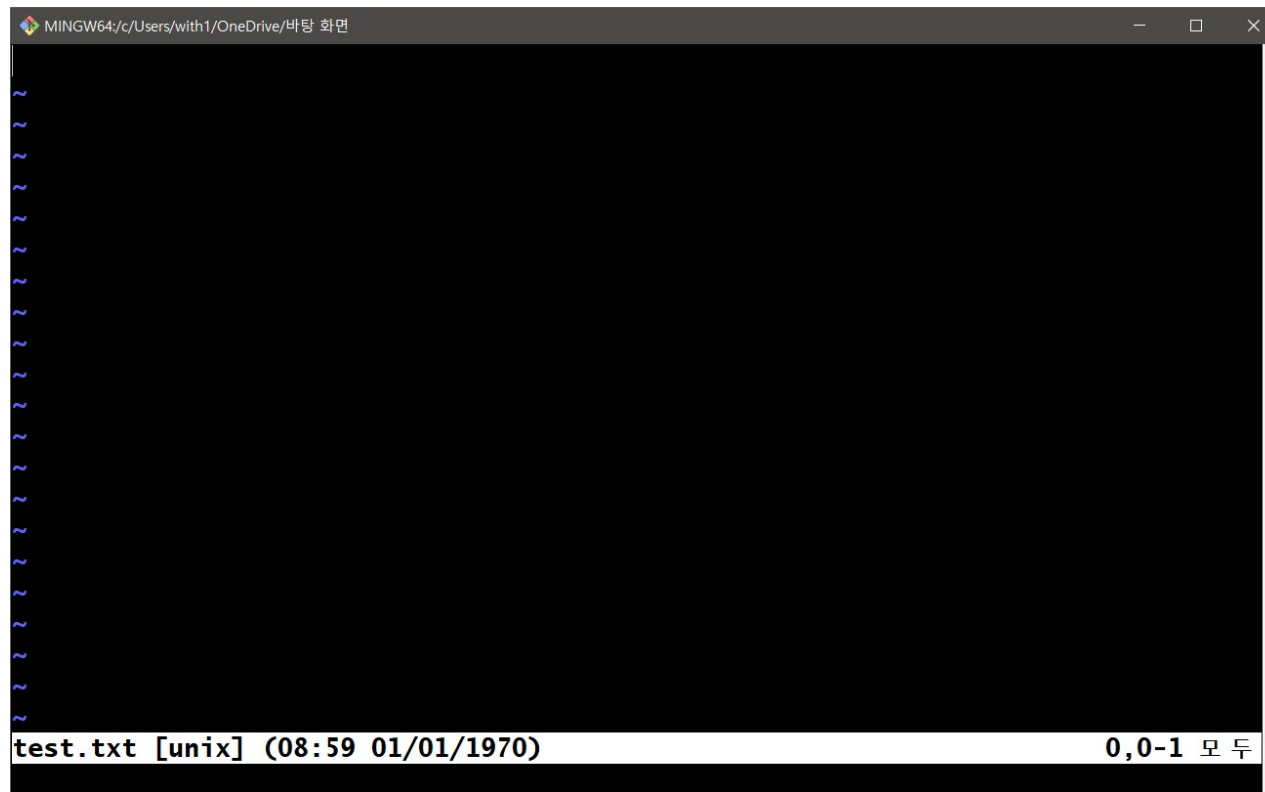
UNI07 Server

- 서버 접속 방법
 - PulseSecure VPN 실행 후, 터미널에서 아래와 같이 입력
 - **ssh -p 12345 [ID]@10.0.2.96**
- 서버 접속 ID / Passwd
 - ID : cs + [학번] (ex, cs20211234)
 - Passwd : (개인톡으로 알려드리겠습니다.)

```
cs20215350@uni07:~  
with1015@with1015-p01:~$ ssh -p 12345 cs20215350@10.0.2.96  
cs20215350@10.0.2.96's password:  
Last login: Wed Jan 19 13:16:28 2022 from 10.20.26.250  
[cs20215350@uni07 ~]$
```

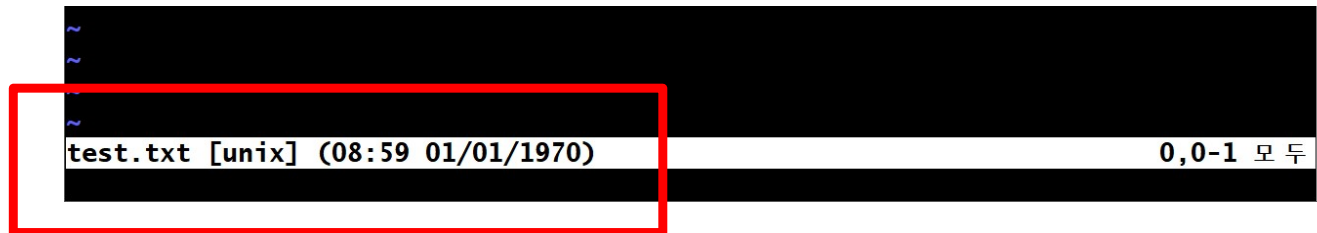
Vim basic

- vi [존재하는 파일 이름] 또는 vi [생성할 파일 이름]



Vim basic

Command 모드



A screenshot of the Vim editor interface in Command mode. The status bar at the bottom shows 'test.txt [unix] (08:59 01/01/1970)' on the left and '0,0-1 모두' on the right. A red rectangular box highlights the status bar area.

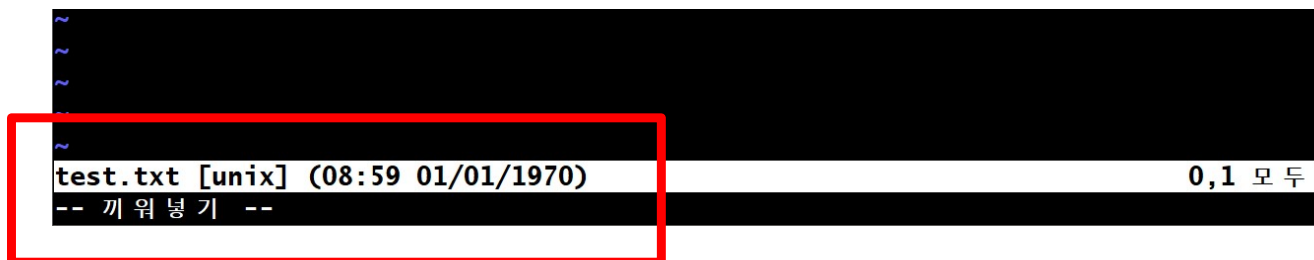
i 입력



Esc 입력



입력 모드



A screenshot of the Vim editor interface in Insert mode. The status bar at the bottom shows 'test.txt [unix] (08:59 01/01/1970)' on the left and '0,1 모두' on the right. Below the status bar, the text '-- 끼워 넣 기 --' is visible. A red rectangular box highlights the status bar area.

Vim basic

- Command 모드에서 저장/닫기/저장 후 닫기 방법

```
~
~
~
test.txt [unix] (0
:w
```

저장

```
~
~
~
test.txt [unix] (08:5
:q
```

닫기 (저장 x)

```
~
~
~
test.txt [unix] (08:5
:wq
```

저장 후 닫기

```
~
~
~
test.txt [unix] (0
:wq!
```

명령에 ! 붙이면 강제로 명령 실행

Bash Script

- 파일 상단에 `#!/bin/bash`로 시작합니다.
- Bash shell을 사용한다는 것을 의미합니다.
- .sh 확장자를 사용합니다 (ex: file.sh)
- 실행 할 때는 `source [sh 파일]`을 입력하여 사용합니다.

Variable

- Bash script의 변수는 type이 따로 존재하지 않습니다.
- [변수 이름]=[데이터] 로 선언합니다.
- = 사이에 공백이 반드시 존재해선 안됩니다.
- [변수 이름]='[명령어]' 또는 \$(명령어) 를 사용하면 명령어의 실행 결과가 변수에 들어가게 됩니다.
- 변수의 참조를 위해서는 \${변수 이름}를 사용합니다.

```
1 #!/bin/bash
2
3 variable=10
4 file_list='ls'
5 echo $file_list
```

변수 선언

명령어 실행 결과를 변수로 선언

변수 참조

Arithmetic operation

- 괄호 2개 안에서 연산을 합니다.
- `Val1=$((1+2*3))`
- `((Val1++))`
- `Val2=$((${Val1}/2))`
- / 연산자는 소수점 나눗셈을 할 수 없습니다.
- bc 명령어와 pipeline을 이용하면 소수점 연산이 가능합니다.
- `echo "55/3" | bc -l` (Linux basic calculator)

Array

- `<array 이름>=(값1 값2 값3 ... 값N)`
- Ex) `array=("hello" 5 123 "ABC")`
- 각 element 사이에 **space**를 통해 구분합니다.
- `${array[index]}`를 통해 특정 index의 값 참조가 가능합니다.
- `${array[@]}`를 통해 모든 index의 값을 참조할 수 있습니다.
- `+=(추가할 값들)` 로 array에 값 추가가 가능합니다.
- Ex) `array+=("hello2" "ABC2" 123456)`

if

```
if [ 조건 1 ]  
then  
    ..실행..  
elif [ 조건 2 ]  
then  
    .. 실행 ..  
else  
    .. 실행 ..  
fi
```

연산자	설명 (True가 되는 경우)
! 명제	명제가 거짓일 때
-n 문자열	문자열의 길이가 0보다 클 때
-z 문자열	문자열의 길이가 0일 때
문자열1 = 문자열2	두 문자열이 서로 같을 때
문자열 != 문자열2	두 문자열이 서로 다를 때
정수1 -eq 정수2	두 정수가 서로 같을 때
정수1 -gt 정수2	정수1이 정수2보다 클 때
정수1 -lt 정수2	정수1이 정수2보다 작을 때
-d 디렉토리	해당 디렉토리가 존재할 때
-e 파일	해당 파일이 존재할 때

for

- Java나 Python처럼 foreach 형식으로 많이 사용합니다.

```
for <변수> in <sequence/array>
do
    .. 실행 ..
done
```



foreach 스타일

```
for (( c=1; c<=5; c++))
do
    .. 실행 ..
done
```



Three-expression
스타일

while

- 주로 redirection과 함께 파일을 읽을 때 사용합니다.

```
while [ 조건 ]  
do  
    .. 실행 ..  
done
```

```
while read line  
do  
    echo $line  
done < [파일 이름]
```

case

- 주로 bash script 파일의 실행 argument를 처리할 때 사용합니다.

```
case <변수> in
    case1)
        .. 실행 ..
    ;;
    case2)
        .. 실행 ..
    ;;
    ...
    *)
        .. 실행 ..
    ;;
esac
```

Case가 끝나면 ;;를
반드시 써준다.

모든 case에 만족하지
않는 경우를 처리

function

- 함수를 정의할 때 parameter는 따로 쓰지 않습니다.

```
function <함수 이름> (){  
    ... 실행 ...  
    return <return 할 것>  
}
```



```
<함수 이름> <parameters>
```

Argument

- \$0 : script의 이름
- \$1 ~ \$9 (그 이상은 \${10}, \${11} ...) : script로 들어오는 파라미터
- @\$: 모든 argument (배열 형태)
- \$# : argument의 개수
- \$? : 이전에 사용한 커맨드의 return code
- \$\$: 현재 script가 사용하고 있는 PID 값
- !! : 마지막으로 사용한 커맨드 전체
- \$_ : 마지막으로 사용한 커맨드의 argument들

다른 파일의 함수 사용하기

- Bash script 파일 안에서 source 명령어를 사용하면 다른 파일에 있는 함수 사용 가능.
- Python의 import나 C언어의 include와 같은 역할을 합니다.
- **source <sh 파일 이름>** 으로 사용.

Activity

- **Bash script로 Python 자동화 파일을 만들고 argument parsing을 적용 해봅시다.**

Thank you

