



Linux/Ubuntu



OMiLAB KOREA

# INDEX

---

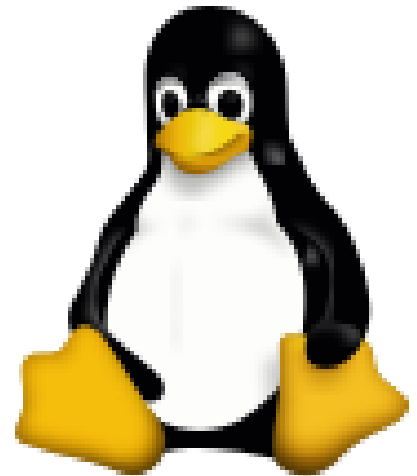
- ▶ Linux
- ▶ Ubuntu
- ▶ Linux/Ubuntu 명령어
- ▶ 텍스트 편집기
- ▶ gcc 컴파일러
- ▶ 실습: C 프로그래밍
- ▶ Homework: output
- ▶ Remote-server 접속
- ▶ Remote-FTP 접속

# Linux

---

## ▶ Linux

- ▶ 리누스 토르발스가 커뮤니티 주체로 개발한 운영체제
  - ▶ 유닉스 계열의 운영체제
  - ▶ 다중사용자, 다중작업(멀티태스킹), 다중 스레드 지원
- ▶ 사용 범위
  - ▶ 개인용 컴퓨터
  - ▶ 슈퍼컴퓨터
  - ▶ 스마트폰
  - ▶ 스마트 TV
  - ▶ 임베디드 시스템 등
- ▶ 오픈 소스
  - ▶ 리눅스 기반 운영체제: 페도라, 우분투, 안드로이드



# Ubuntu

---

## ▶ Ubuntu

- ▶ 캐노니컬의 CEO인 마크 셔틀워스가 시작
  - ▶ 개발 주체: 우분투 재단
  - ▶ 유닉스 계열 운영체제
- ▶ 리눅스 배포판
  - ▶ 오픈소스
  - ▶ GUI 제공
  - ▶ 다중사용자
  - ▶ 다중작업(멀티태스킹)
  - ▶ 다중 스레드 지원

ubuntu 



# Linux/Ubuntu 명령어

---

## ▶ man

- ▶ 기능 : 리눅스 명령어나 함수, 유틸리티 사용법 등에 관한 매뉴얼을 보여주는 명령
- ▶ 사용법 : man [옵션] 명령어

# Linux/Ubuntu 명령어

---

## ▶ mkdir

### ▶ 기능

- ▶ 새로운 디렉토리를 신규로 생성하는 명령어

### ▶ 사용법

- ▶ mkdir [옵션] 디렉토리명
- ▶ mkdir [옵션] /디렉토리명
  - ex) mkdir hw1

# Linux/Ubuntu 명령어

---

## ▶ ls

- ▶ 기능 : 파일 목록 출력 명령어

- ▶ 사용법 : ls [옵션]

  - ▶ ex) ls

  - ▶ ex) ls -l

- ▶ 옵션

  - ▶ -a : 디렉토리 내의 모든 파일 출력

  - ▶ -l : 파일 허용 여부, 소유자, 그룹, 크기, 날짜 등을 출력

# Linux/Ubuntu 명령어

---

## ▶ cd

- ▶ 기능 : 현재의 디렉토리에서 다른 디렉토리로 경로를 변경하는 명령어
- ▶ 사용법
  - ▶ cd 디렉토리
  - ▶ cd /디렉토리
    - ex) cd hw1
    - ex) cd /hw1/test



# Linux/Ubuntu 명령어

---

- ▶ pwd

- ▶ 기능

- ▶ 현재의 디렉토리 위치를 알려주는 명령어

- ▶ 사용법

- ▶ pwd

# Linux/Ubuntu 명령어

---

## ▶ cp

- ▶ 기능 : 파일을 복사하는 명령어

### ▶ 사용법

- ▶ cp [옵션] [원본] [복사파일명]

- ▶ cp [옵션] [원본] [복사위치]

- ex) cp test.c test2.c

- ex) cp test.c /hw1/test2.c

### ▶ 옵션

- ▶ -r : 하위 디렉토리를 포함한 모든 파일 복사

# Linux/Ubuntu 명령어

---

## ▶ mv

### ▶ 기능

- ▶ 파일을 다른 디렉토리로 이동할 때 사용되는 명령어

### ▶ 사용법

- ▶ mv [옵션] [원본] [이동위치]

- ex) mv test.c ./hw1

### ▶ 옵션

- ▶ -i : 옮길 디렉토리에 존재하는 파일이 있으면 확인

# Linux/Ubuntu 명령어

---

## ▶ rm

### ▶ 기능

- ▶ 파일을 삭제하는 명령어

### ▶ 사용법

- ▶ `rm [옵션] 파일명`

- ex) `rm test.c`

- ex) `rm -r hw1`

### ▶ 옵션

- ▶ `-i` : 지우기 전에 확인
- ▶ `-r` : 하위 디렉토리의 파일까지 삭제

# Linux/Ubuntu 명령어

---

## ▶ cat

### ▶ 기능

- ▶ 텍스트 파일을 만들거나 파일 내용을 출력하는 명령어

### ▶ 사용법

- ▶ cat [옵션] 파일명

□ ex) cat test.c

### ▶ 옵션

- ▶ -b : 비어있는 행을 제외한 모든 행에 번호를 붙임
- ▶ -E : 각 행 끝에 '\$' 표시
- ▶ -T : '^\t'로 TAB 문자 표시

# Linux/Ubuntu 명령어

---

## ▶ chmod

### ▶ 기능

- ▶ 파일의 권한과 디렉토리의 권한을 바꾸는 명령어

### ▶ 사용법

- ▶ `chmod [옵션] [권한 파일|디렉토리]`

- ex) `chmod 744 test.c`

- ex) `chmod g+x test.c` or `chmod g-x test.c`

- u: user, g: group, o: other

- r: read, w: write, x: execute

### ▶ 옵션

- ▶ `-R` : 하위 파일과 디렉토리까지 변경

# Linux/Ubuntu 명령어

---

## ▶ 권한

▶ **drwxrwxrwx**

- ▶ d : 디렉토리임을 나타냄
  - ▶ r : 읽기 권한
  - ▶ w : 쓰기 권한
  - ▶ x : 실행 권한
- 
- ▶ 2~4번째 : 소유자 권한
  - ▶ 5~7번째 : 그룹 권한
  - ▶ 8~10번째 : 실행 권한

# Linux/Ubuntu 명령어

---

## ▶ 권한 설정

### ▶ 권한은 이진수를 사용하여 관리

- ▶ 1은 권한 있음, 0은 권한 없음을 나타냄

### ▶ `rw-rw-rw- : 111111111`

- ▶ 사용자 구분에 따라 세자리씩 분할 : 111 111 111
- ▶ 각각을 십진수로 변환 : 777
- ▶ `chmod 777 파일명` 실행 시 모든 사용자에게 모든 권한이 부여됨

### ▶ `rw-r--r-- : 111100100`

- ▶ 사용자 구분에 따라 세자리씩 분할 : 111 100 100
- ▶ 각각을 십진수로 변환 : 744
- ▶ `chmod 744 파일명` 실행 시 소유자는 모든 권한, 다른 사용자는 읽기 권한만 부여됨

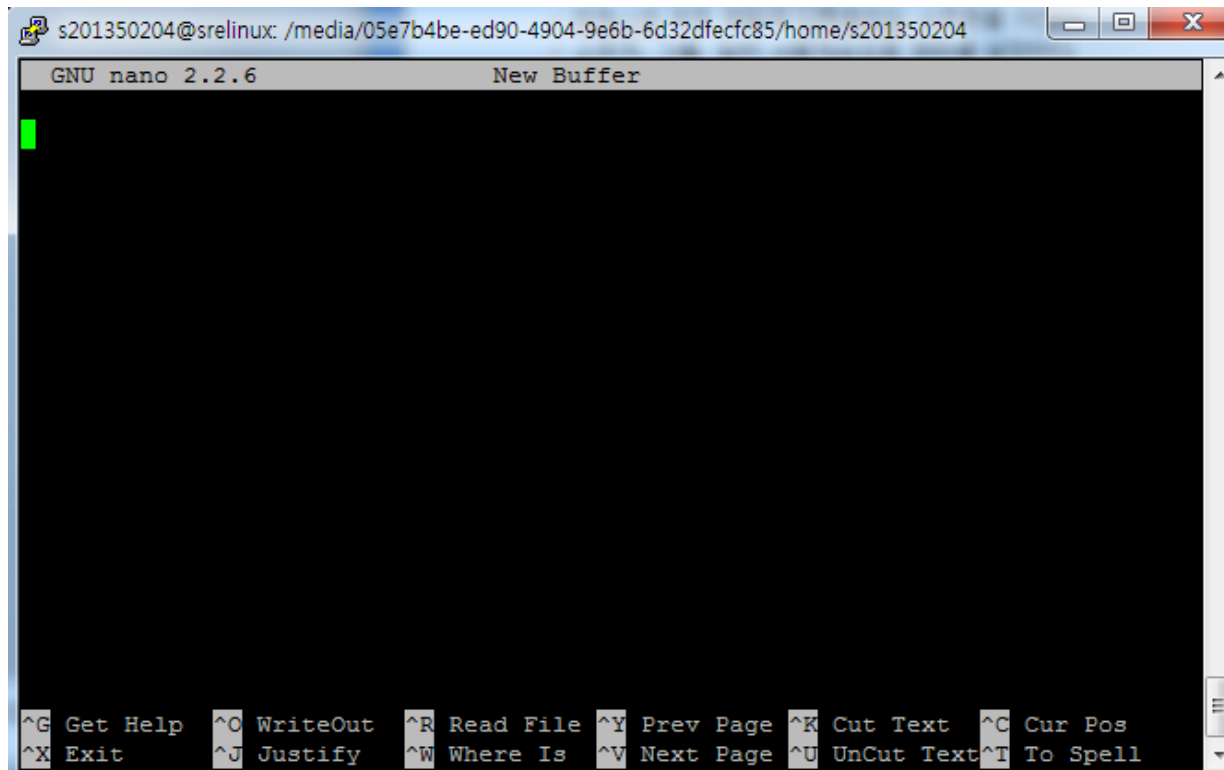


# 텍스트 편집기

---

## ▶ nano 편집기

- ▶ “nano” 또는 “nano 파일명” 명령어 사용
  - ▶ 파일명 부분에 이미 존재하는 파일명을 입력 시 해당 파일 열림



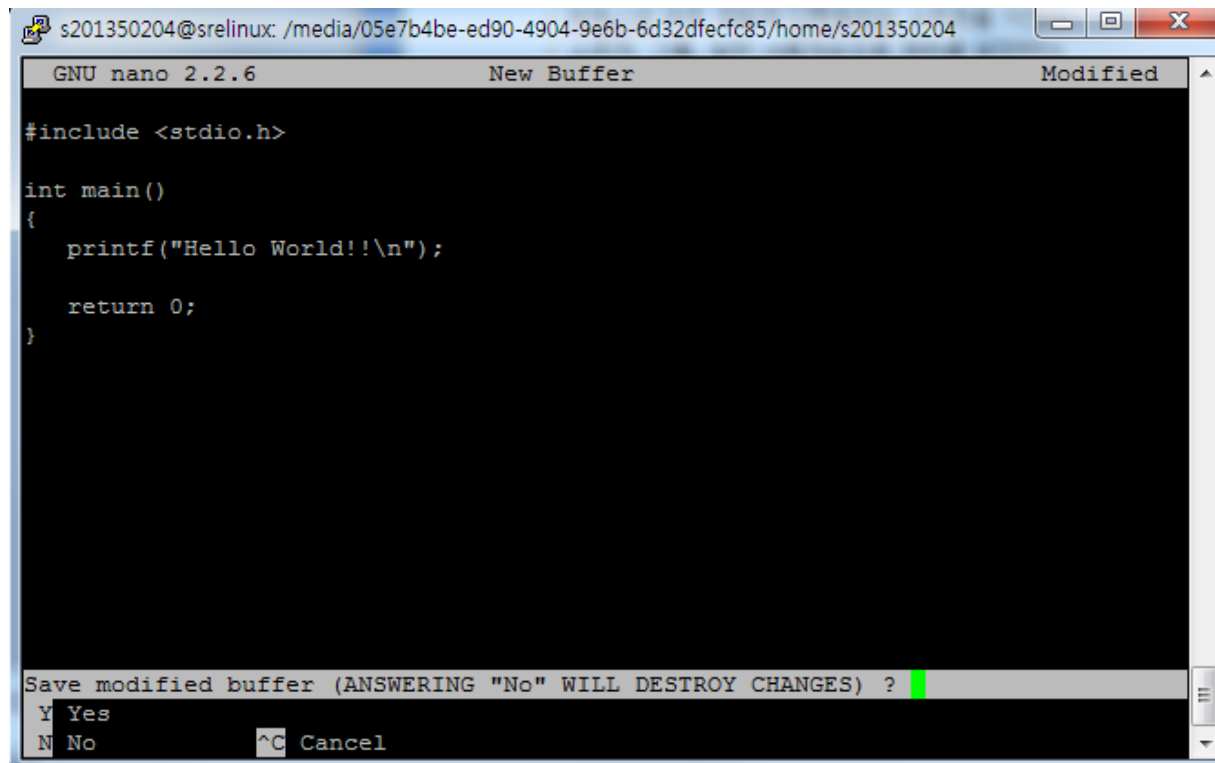
```
s201350204@srelinux: /media/05e7b4be-ed90-4904-9e6b-6d32dfecfc85/home/s201350204
GNU nano 2.2.6 New Buffer

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

# 텍스트 편집기

---

- ▶ nano 편집기
  - ▶ Ctrl + x로 종료
    - ▶ 파일에 변화가 있을 시 아래의 선택문 등장



The screenshot shows a terminal window with the GNU nano 2.2.6 text editor. The editor is displaying a C program that prints "Hello World!!\n". The program code is as follows:

```
#include <stdio.h>

int main()
{
    printf("Hello World!!\n");

    return 0;
}
```

At the bottom of the editor, a confirmation dialog is displayed, asking to save the modified buffer. The dialog text is: "Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES) ?". The dialog offers three options: "Y Yes", "N No", and "^C Cancel". The "Y Yes" option is currently selected, indicated by a green cursor.

# 텍스트 편집기

## ▶ vi 편집기

- ▶ “vi” 또는 “vi 파일명” 명령어 사용
  - ▶ 파일명 부분에 이미 존재하는 파일명을 입력 시 해당 파일 열림

```
s201855077@SRElab: ~  
~  
~  
~  
~  
VIM - Vi IMproved  
~  
version 7.4.1689  
by Bram Moolenaar et al.  
Modified by pkg-vim-maintainers@lists.alioth.debian.org  
Vim is open source and freely distributable  
  
Help poor children in Uganda!  
type :help iccf<Enter>      for information  
  
type :q<Enter>              to exit  
type :help<Enter> or <F1>   for on-line help  
type :help version7<Enter>  for version info  
  
0,0-1 All
```

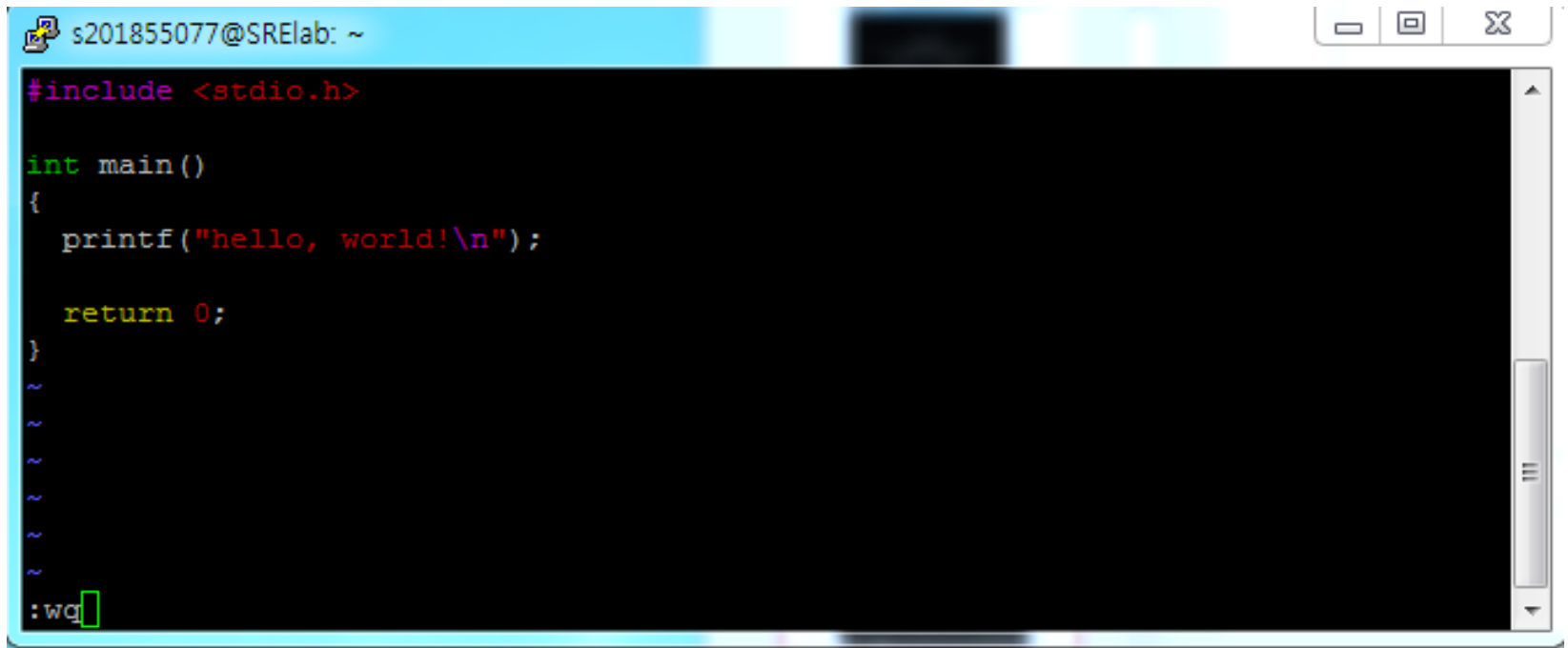
# 텍스트 편집기

---

## ▶ vi 편집기

### ▶ 저장 및 종료

- ▶ “ESC” 누른 후 ‘:wq’ 키 또는 ‘:wq 파일명’ 으로 저장 및 종료



```
s201855077@SRElab: ~  
#include <stdio.h>  
  
int main()  
{  
    printf("hello, world!\n");  
  
    return 0;  
}  
~  
~  
~  
~  
~  
~  
~  
:wq
```

# gcc 컴파일러

---

- ▶ 컴파일 : gcc 명령어 이용
  - ▶ gcc 파일명 ex) gcc hello.c
- ▶ gcc 소스코드
  - ▶ 실행파일이 a.out 이라는 이름으로 생성됨
  - ▶ 실행파일 실행 : ./a.out
- ▶ gcc -o 실행파일명 소스코드
  - ▶ 실행파일이 입력한 실행파일명으로 생성됨
  - ▶ ex) gcc -o **hello** hello.c

# 실습

---

## ▶ C 프로그래밍

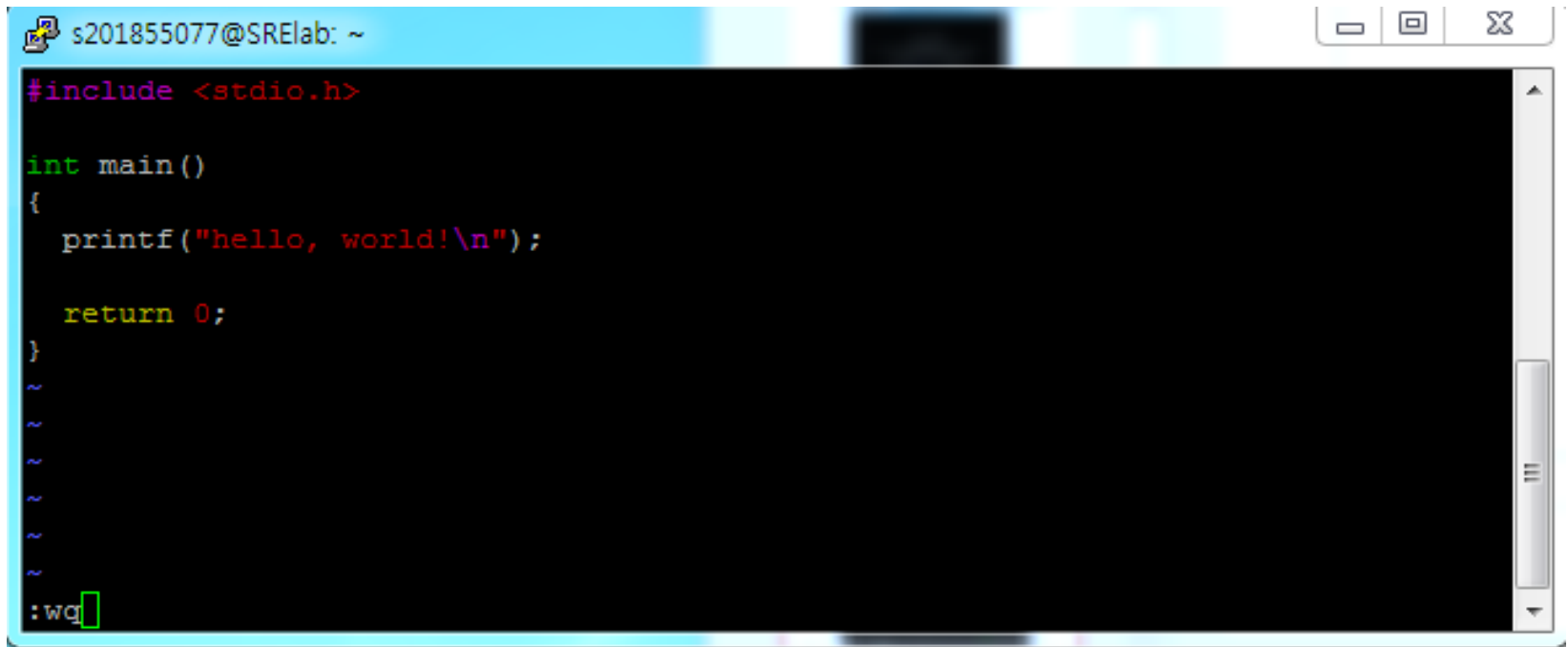
- ▶ nano 또는 vi 편집기를 활용하여 “Hellow, world!” 를 출력하는 C 프로그램을 작성하시오.

# 실습

---

## ▶ C 프로그래밍

- ▶ nano 또는 vi 편집기를 활용하여 “Hello, world!” 를 출력하는 C 프로그램을 작성하시오.



```
s201855077@SRElab: ~  
#include <stdio.h>  
  
int main()  
{  
    printf("hello, world!\n");  
  
    return 0;  
}  
~  
~  
~  
~  
~  
~  
:wq
```

# Homework: output

---

- ▶ script p.output (파일명) ex) hw1.output
- ▶ env
- ▶ whoami
- ▶ date
- ▶ pwd
- ▶ w
- ▶ ls -l
- ▶ cat p.c (파일명) ex) cat hello.c
- ▶ gcc p.c (파일명) ex) gcc hello.c
- ▶ ls -l
- ▶ ./a.out
- ▶ ^D (Ctrl + D)



# Remote-server 접속

## ▶ Putty 다운로드

▶ <http://www.chiark.greenend.org.uk/~sgtatham/putty/>

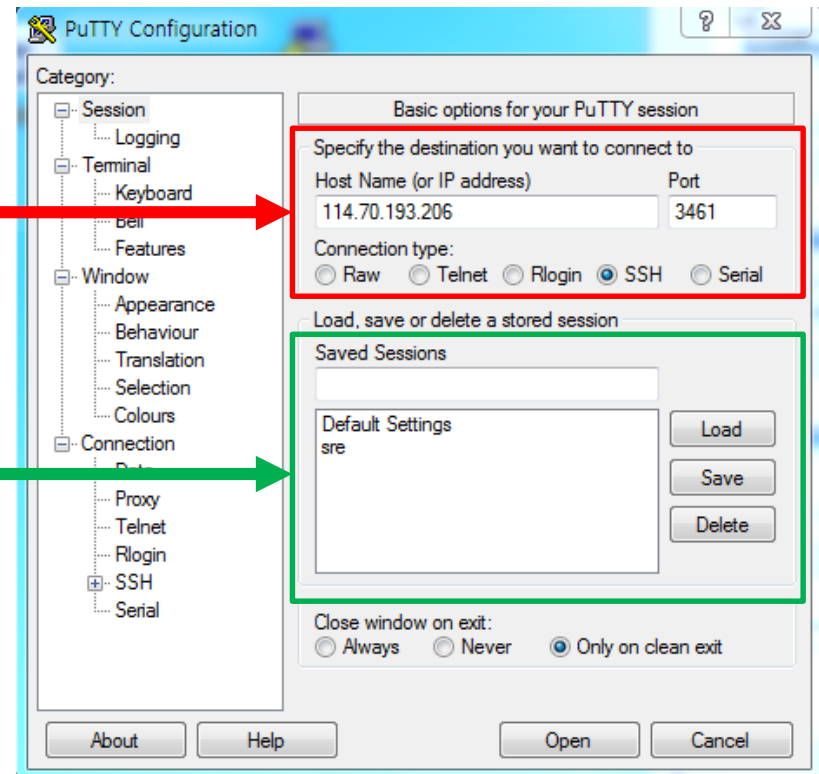
## ▶ Linux 서버 연결

▶ IP: 203.254.143.132

▶ Port: 7777

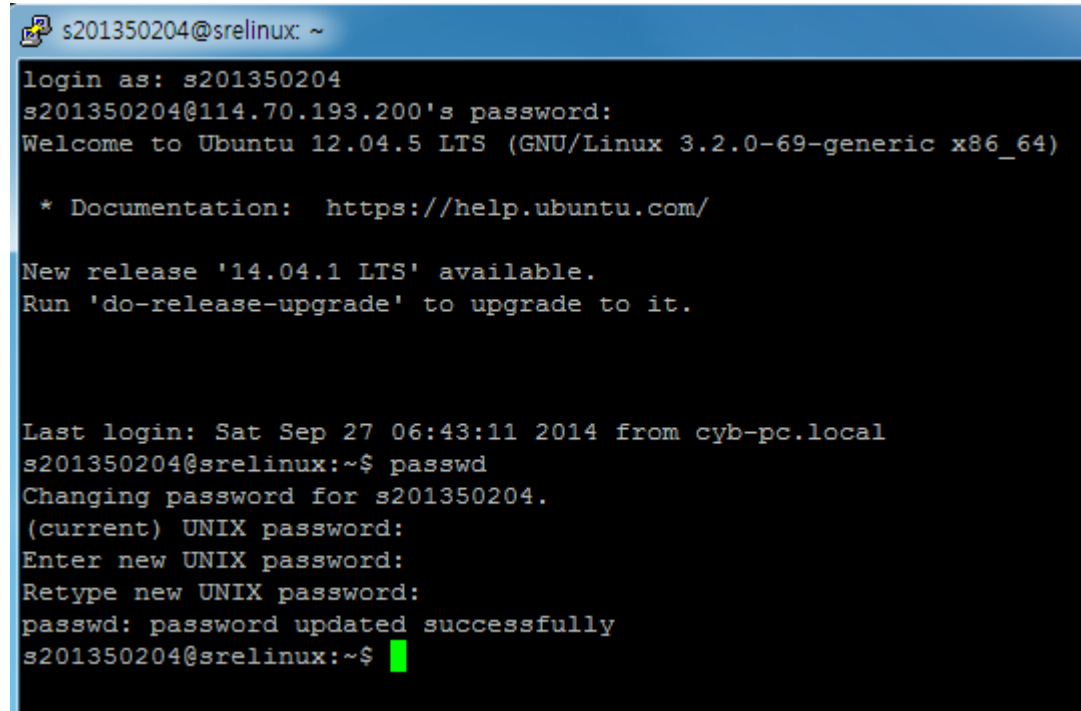
## ▶ Saved Sessions 저장

▶ 추후 접속 시 ip입력없이 바로 접속



# Remote-server 접속

- ▶ ID: s학번
  - ▶ 예) s201855077
- ▶ Password: 학번
- ▶ 비밀번호 변경
  - ▶ passwd 명령어

A terminal window with a blue title bar showing the user 's201350204@srelinux'. The terminal text shows a successful login, system information for Ubuntu 12.04.5 LTS, a notification for a new release, and the execution of the 'passwd' command to change the password. The password was updated successfully, and the prompt returns to the user.

```
s201350204@srelinux: ~
login as: s201350204
s201350204@114.70.193.200's password:
Welcome to Ubuntu 12.04.5 LTS (GNU/Linux 3.2.0-69-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

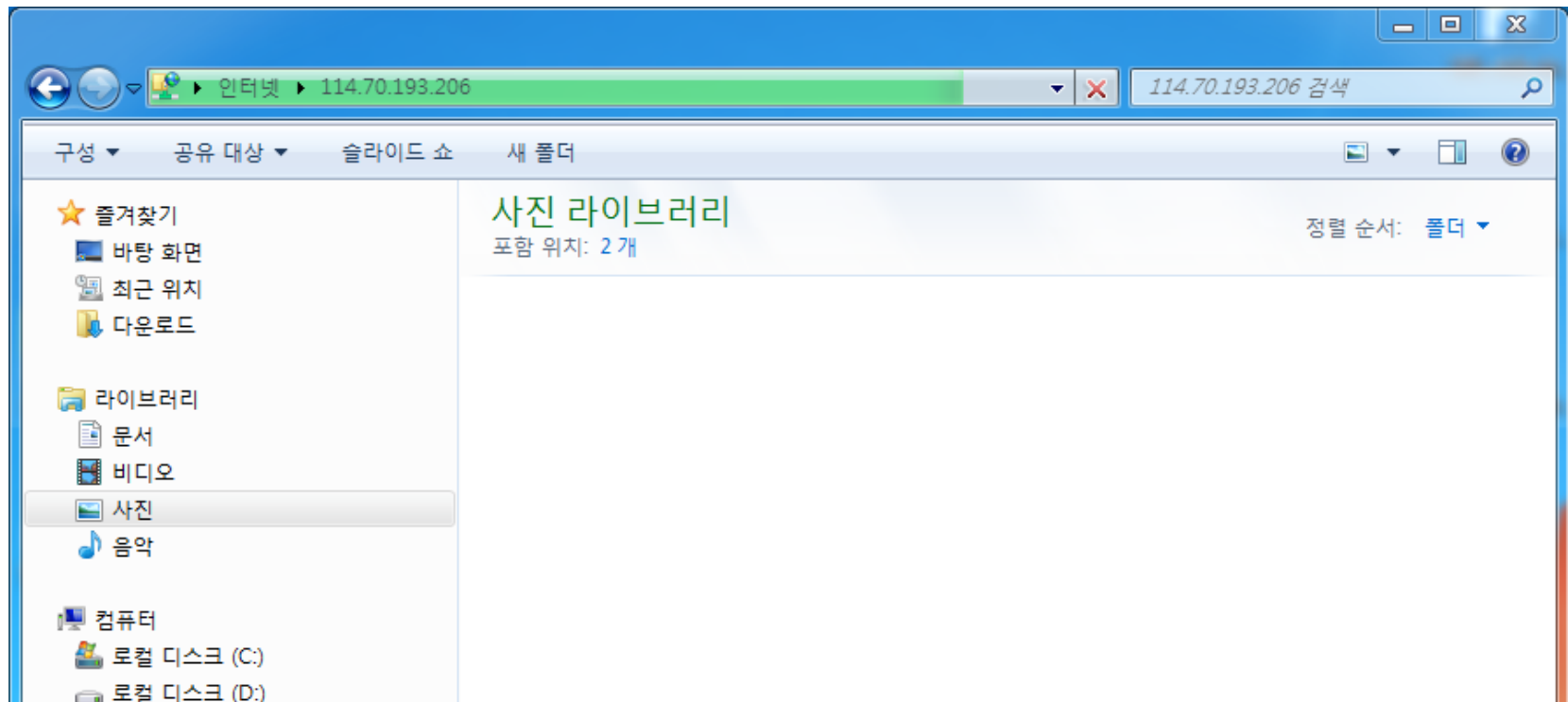
New release '14.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sat Sep 27 06:43:11 2014 from cyb-pc.local
s201350204@srelinux:~$ passwd
Changing password for s201350204.
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
s201350204@srelinux:~$
```

# Remote-FTP 접속

## ▶ ftp 접속

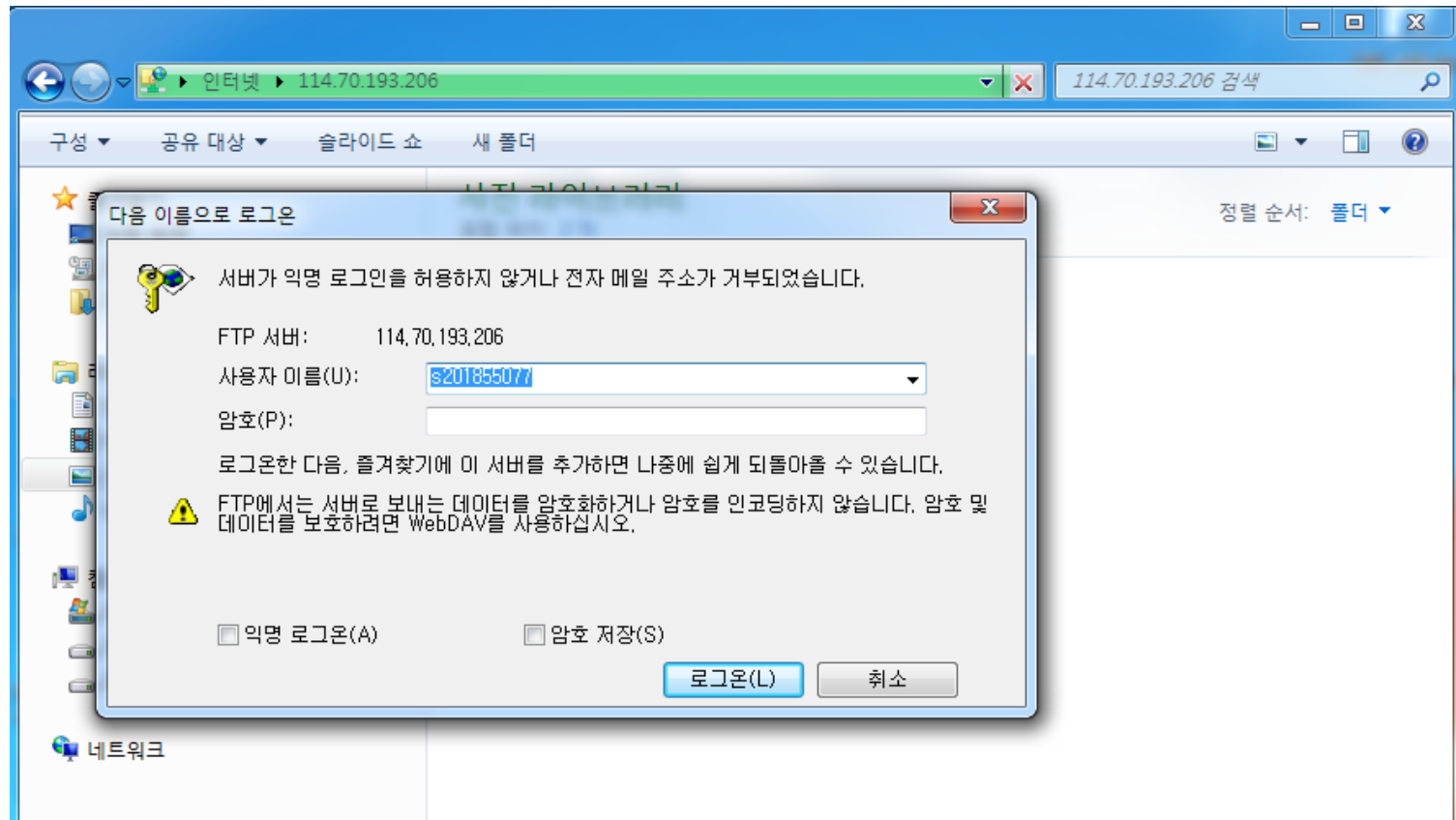
- ▶ Windows와의 파일 전송을 위하여 ftp 접속
  - ▶ ftp://203.254.143.132 입력



# Remote-FTP 접속

## ▶ ftp 접속

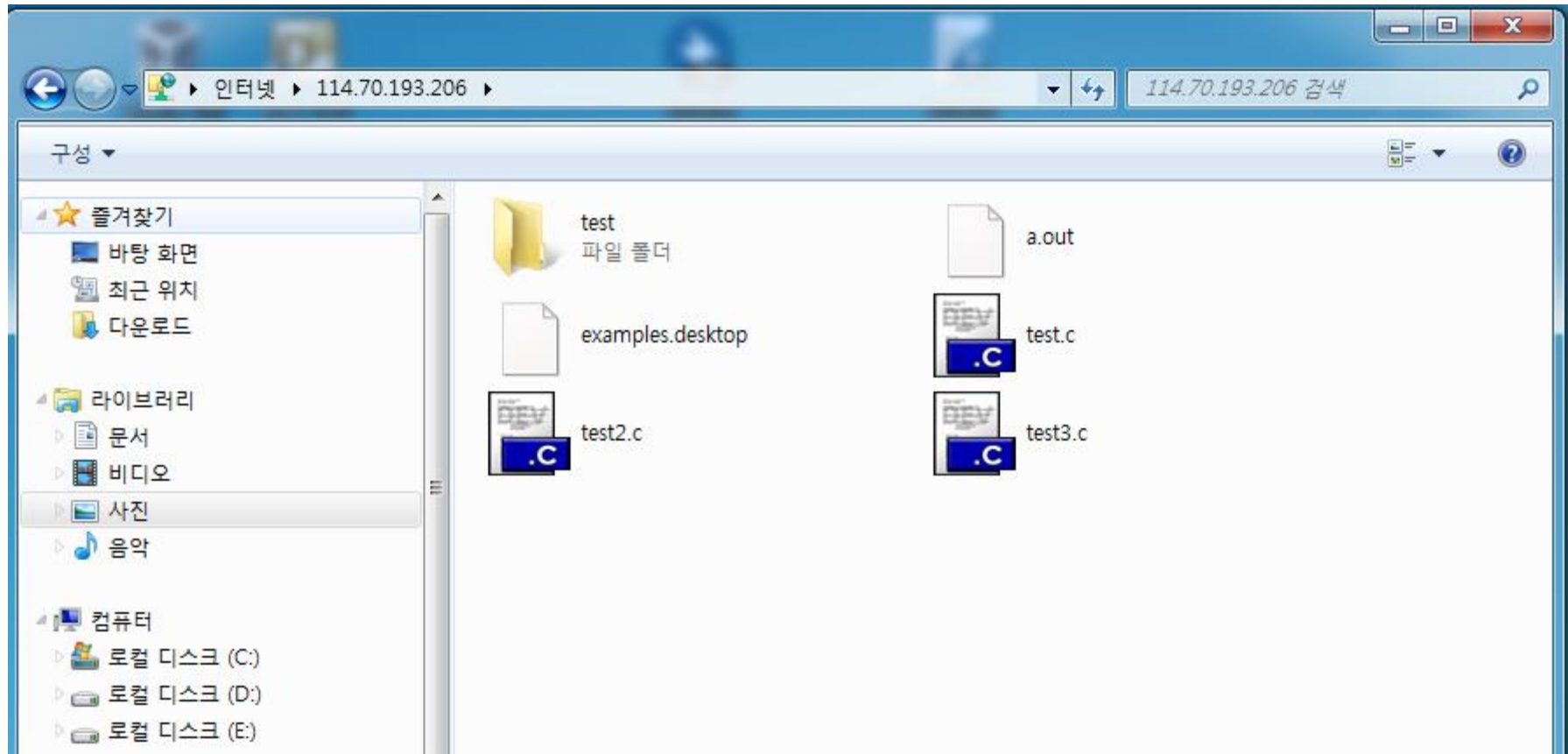
### ▶ Windows와의 파일 전송을 위하여 ftp 접속



# Remote-FTP 접속

## ▶ ftp 접속

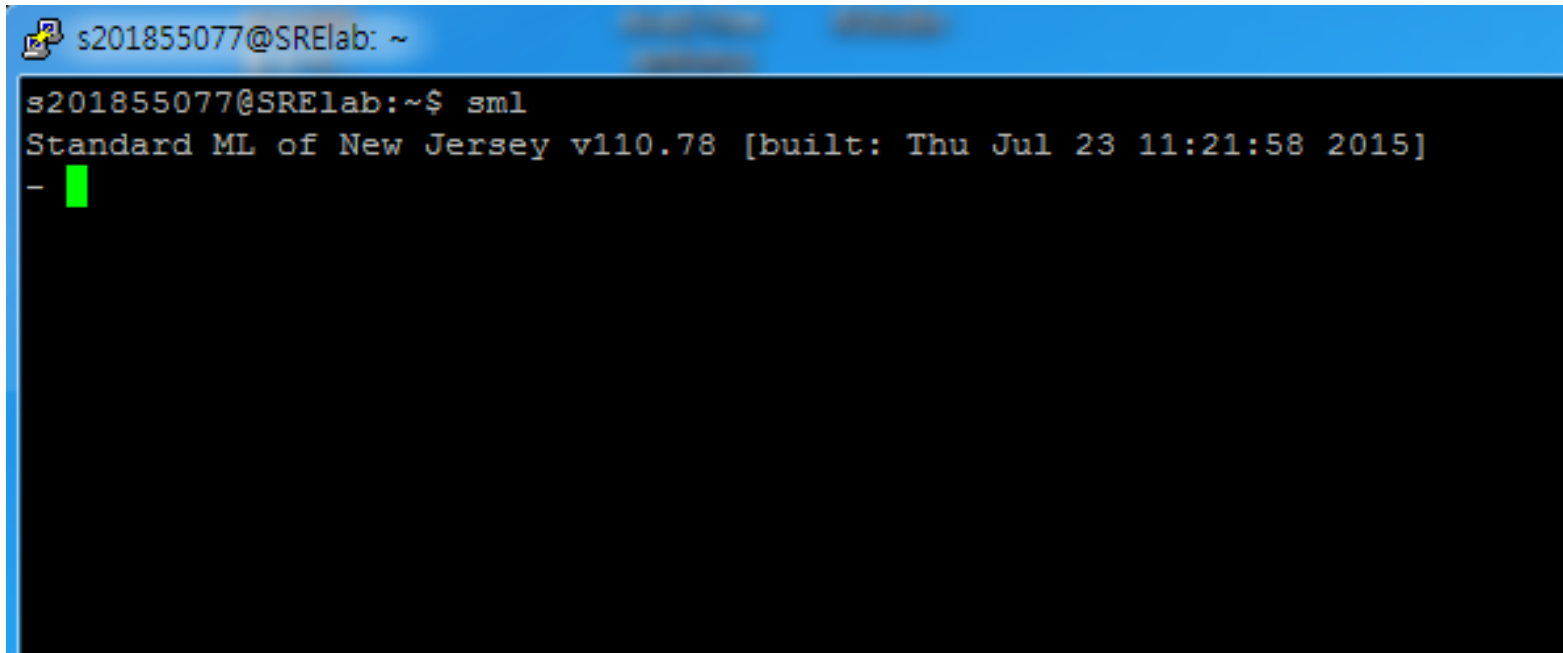
### ▶ Windows와의 파일 전송을 위하여 ftp 접속



# ML Interpreter

---

## ▶ 실행 명령어: sml



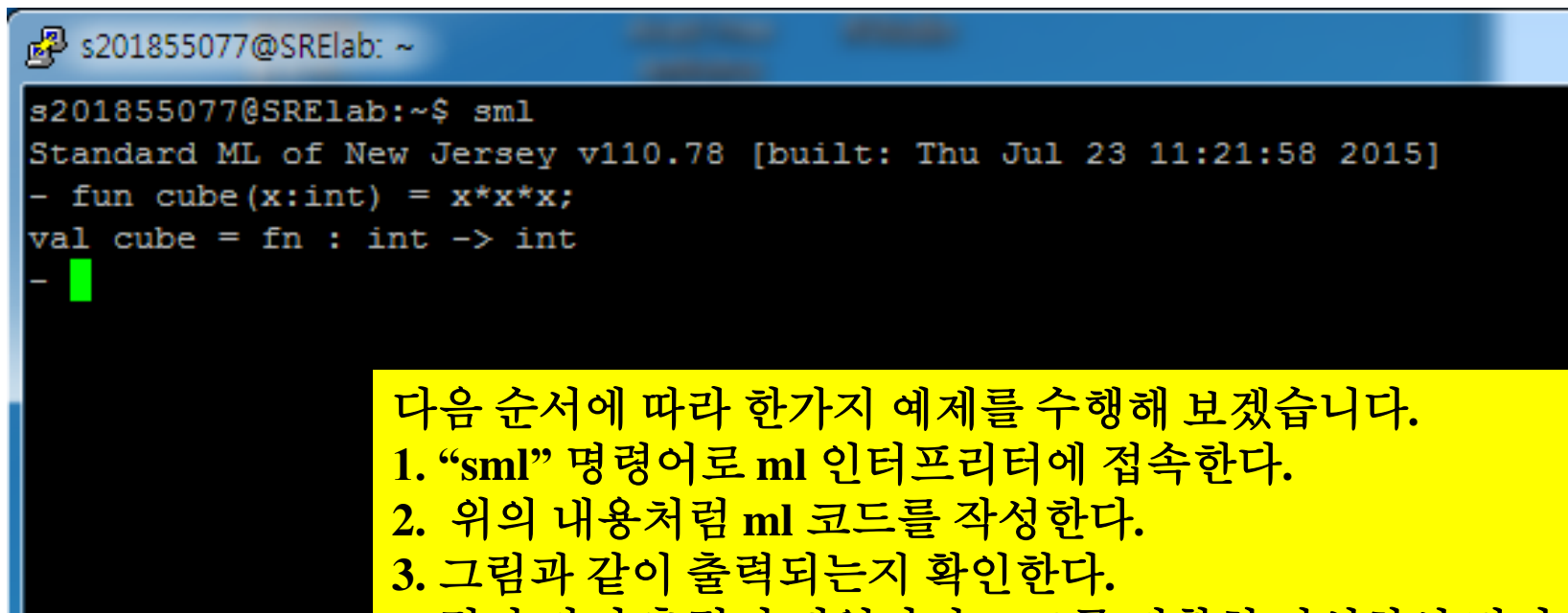
```
s201855077@SRElab: ~  
s201855077@SRElab:~$ sml  
Standard ML of New Jersey v110.78 [built: Thu Jul 23 11:21:58 2015]  
- 
```

셸 창에서 “sml” 명령어를 통해 sml interpreter로 접속할 수 있습니다.  
모든 ml 과제는 여기서 수행하시면 됩니다.

# ML Interpreter

## ▶ 실습

▶ `fun cube(x:int) = x*x*x;`

A terminal window with a blue title bar showing the user 's201855077@SRElab: ~'. The terminal text shows the command 'sml' being executed, which starts the 'Standard ML of New Jersey v110.78' interpreter. The user then enters the function definition 'fun cube(x:int) = x\*x\*x;' and the prompt changes to 'val cube = fn : int -> int'. A green cursor is visible on the line following the definition.

```
s201855077@SRElab: ~  
s201855077@SRElab:~$ sml  
Standard ML of New Jersey v110.78 [built: Thu Jul 23 11:21:58 2015]  
- fun cube(x:int) = x*x*x;  
val cube = fn : int -> int  
- 
```

다음 순서에 따라 한가지 예제를 수행해 보겠습니다.

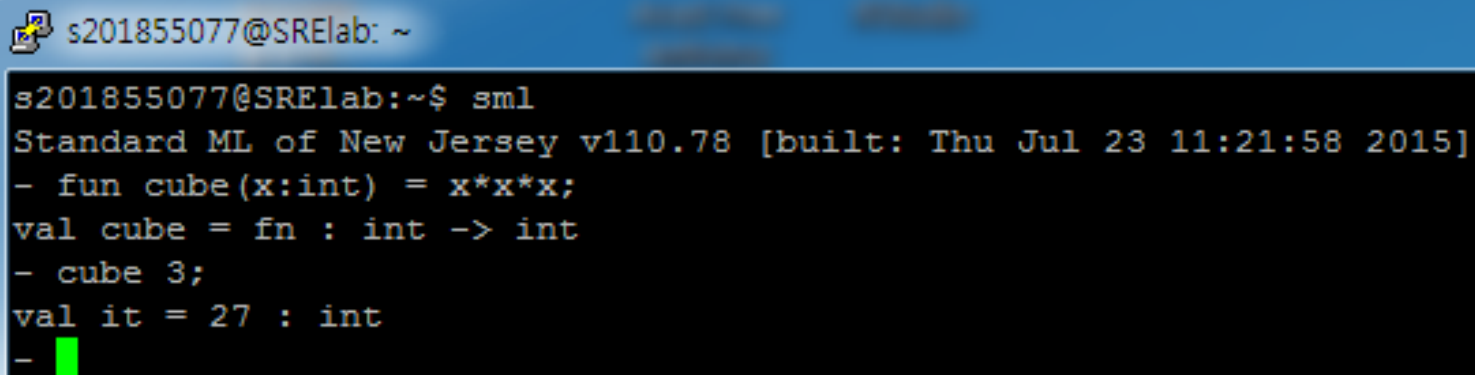
1. “sml” 명령어로 ml 인터프리터에 접속한다.
2. 위의 내용처럼 ml 코드를 작성한다.
3. 그림과 같이 출력되는지 확인한다.

그림과 같이 출력이 되었다면 코드를 정확히 작성하신 겁니다.

# ML Interpreter

## ▶ 함수 호출

▶ ex) cube 3;

A terminal window with a blue title bar showing the user 's201855077@SRElab: ~'. The terminal content shows the execution of the 'sml' command, which starts the 'Standard ML of New Jersey v110.78' interpreter. The user enters several lines of ML code: '- fun cube(x:int) = x\*x\*x;', 'val cube = fn : int -> int', '- cube 3;', and 'val it = 27 : int'. The prompt returns to '- ' with a green cursor.

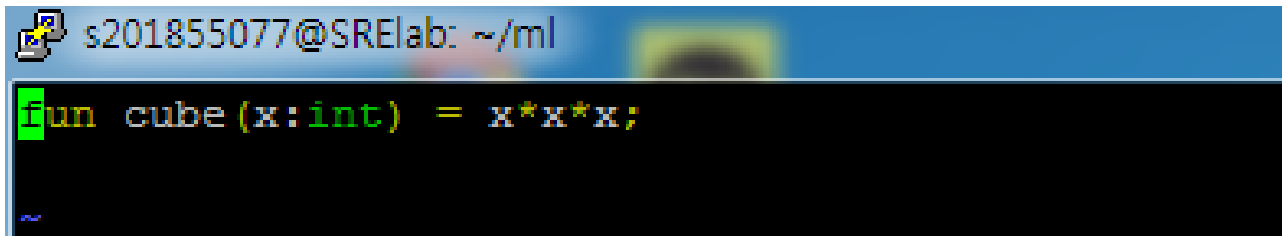
```
s201855077@SRElab: ~  
s201855077@SRElab:~$ sml  
Standard ML of New Jersey v110.78 [built: Thu Jul 23 11:21:58 2015]  
- fun cube(x:int) = x*x*x;  
val cube = fn : int -> int  
- cube 3;  
val it = 27 : int  
- 
```

다음은 앞에서 작성한 코드, 즉 `cube`라는 함수를 호출해 보겠습니다.  
함수 호출은 “함수이름”를 사용하여 호출할 수 있습니다.  
이 예제에서는 “`cube`”라는 함수이름으로 함수를 호출하였으며,  
3은 매개변수를 의미합니다.  
`cube 3;` 이라고 입력 했을 때, 그림과 같이 출력이 나온다면 잘 수행하신 겁니다.



# ML Interpreter

- ▶ 코드 파일 생성/작성
  - ▶ 확장자: .sml 파일로 생성
  - ▶ 파일 생성: vi (or nano) 파일명.sml

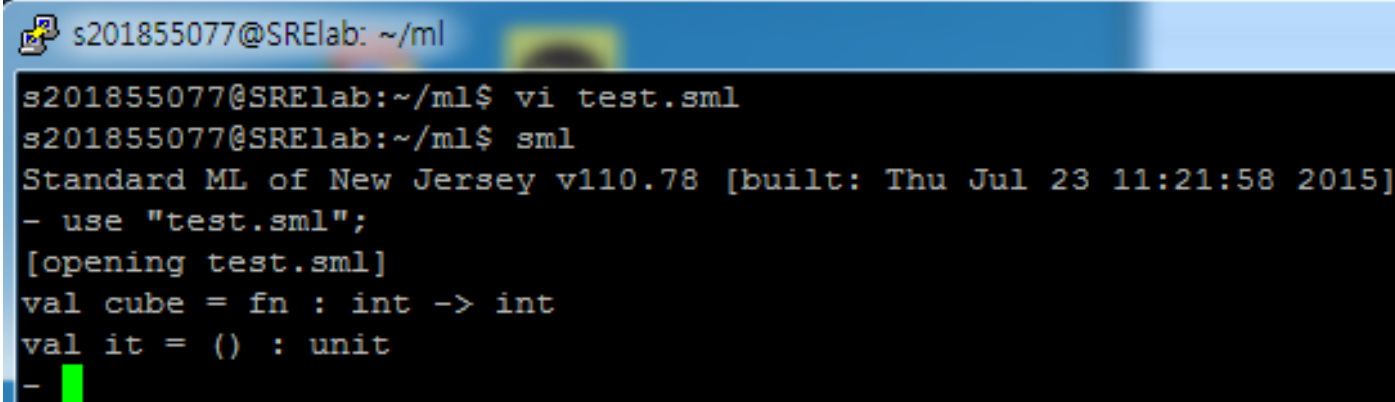
A terminal window with a blue title bar showing the user 's201855077@SRElab' in the directory '~/ml'. The terminal has a black background with green text. The code 'fun cube(x:int) = x\*x\*x;' is entered, with the 'fun' keyword highlighted in green. A cursor is visible at the end of the line.

```
s201855077@SRElab: ~/ml  
fun cube(x:int) = x*x*x;
```

앞에서 확인했듯이 ML은 인터프리터 방식으로 코드가 수행되므로 `sml interpreter` 내에서 계속 코드를 작성 해야 하는 불편함이 있었을 겁니다. 하지만 그림과 같이 `vi` 또는 `nano` 편집기를 통해 `ml` 코드를 작성 및 저장하고, `Sml interpreter` 에서 불러오는 식으로 이용하면 과제하시는데 도움이 될 겁니다. **ml 코드가 작성된 파일의 확장자는 “.sml”로 해야 합니다. ex) hw2.sml** 그림은 앞에서 작성했던 예제 코드를 .sml 확장자로 파일을 생성한 것을 보여줍니다.

# ML Interpreter

- ▶ 파일 불러오기
  - ▶ ML 인터프리터 내에서 use 명령어 사용
  - ▶ ex) use “파일명.sml”;

A terminal window with a blue title bar showing the user 's201855077@SRElab' in the directory '~/ml'. The command prompt shows the user running 'vi test.sml' and then 'sml'. The ML interpreter output shows the version 'Standard ML of New Jersey v110.78 [built: Thu Jul 23 11:21:58 2015]', followed by the command '- use "test.sml";'. The interpreter then shows '[opening test.sml]' and the contents of the file: 'val cube = fn : int -> int' and 'val it = () : unit'. The prompt returns to '- ' with a green cursor.

```
s201855077@SRElab: ~/ml
s201855077@SRElab:~/ml$ vi test.sml
s201855077@SRElab:~/ml$ sml
Standard ML of New Jersey v110.78 [built: Thu Jul 23 11:21:58 2015]
- use "test.sml";
[opening test.sml]
val cube = fn : int -> int
val it = () : unit
- 
```

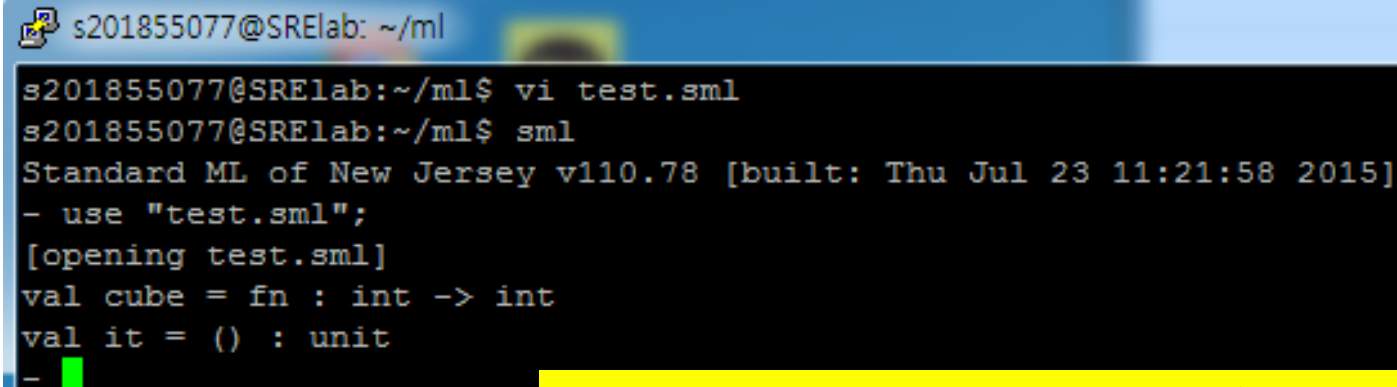
다음 그림은 앞에서 작성한 “.sml” 확장자 파일을 sml interpreter에서 불러오는 방법을 보여줍니다.

sml interpreter 내에서 “use” 명령어를 통해 작성한 파일을 불러올 수 있습니다. use 명령어는 use “파일명.sml”; 처럼 사용합니다. ex) use “hw2.sml”; 그림과 같이 파일을 불러오고 함수 호출을 수행하면 같은 결과가 나오는 것을 확인할 수 있습니다.

# ML Interpreter

---

- ▶ 종료: Ctrl + D

A terminal window with a blue title bar showing the user 's201855077@SRElab' in the directory '~/ml'. The terminal text shows the user running 'vi test.sml' and 'sml'. The ML interpreter version 'Standard ML of New Jersey v110.78' is displayed, followed by the command '- use "test.sml";'. The file 'test.sml' is opened, and its contents are shown: 'val cube = fn : int -> int' and 'val it = () : unit'. The prompt '-' is followed by a green cursor.

```
s201855077@SRElab: ~/ml
s201855077@SRElab:~/ml$ vi test.sml
s201855077@SRElab:~/ml$ sml
Standard ML of New Jersey v110.78 [built: Thu Jul 23 11:21:58 2015]
- use "test.sml";
[opening test.sml]
val cube = fn : int -> int
val it = () : unit
- █
```

sml interpreter 종료는 Ctrl + d 키를 입력하면 됩니다.

# Homework: output

▶ script p.output

▶ env

▶ whoami

▶ date

▶ pwd

▶ w

▶ ls -l

▶ cat p.sml

▶ sml

▶ use "p.sml";

▶ ML 코드 실행 ex) cube 3;

▶ ^ D

▶ ^ D

다음은 **script**를 통해 과제 **output** 파일을 만드는 명령어를 나열한 겁니다. 첫번째 c 과제에서 만든 것 처럼 수행하시면 되는데, 차이점은 빨간 테두리로 표시된 부분입니다. 빨간 테두리의 4줄의 내용은 다음과 같습니다.

1. ML 코드를 수행하기 위해 **sml interpreter**에 접속한다.

2. 작성한 **ml** 코드를 불러온다.

3. 작성한 코드를 실행한다.

(실행까지 **.sml** 코드에 삽입했다면, 바로 결과가 출력됩니다.)

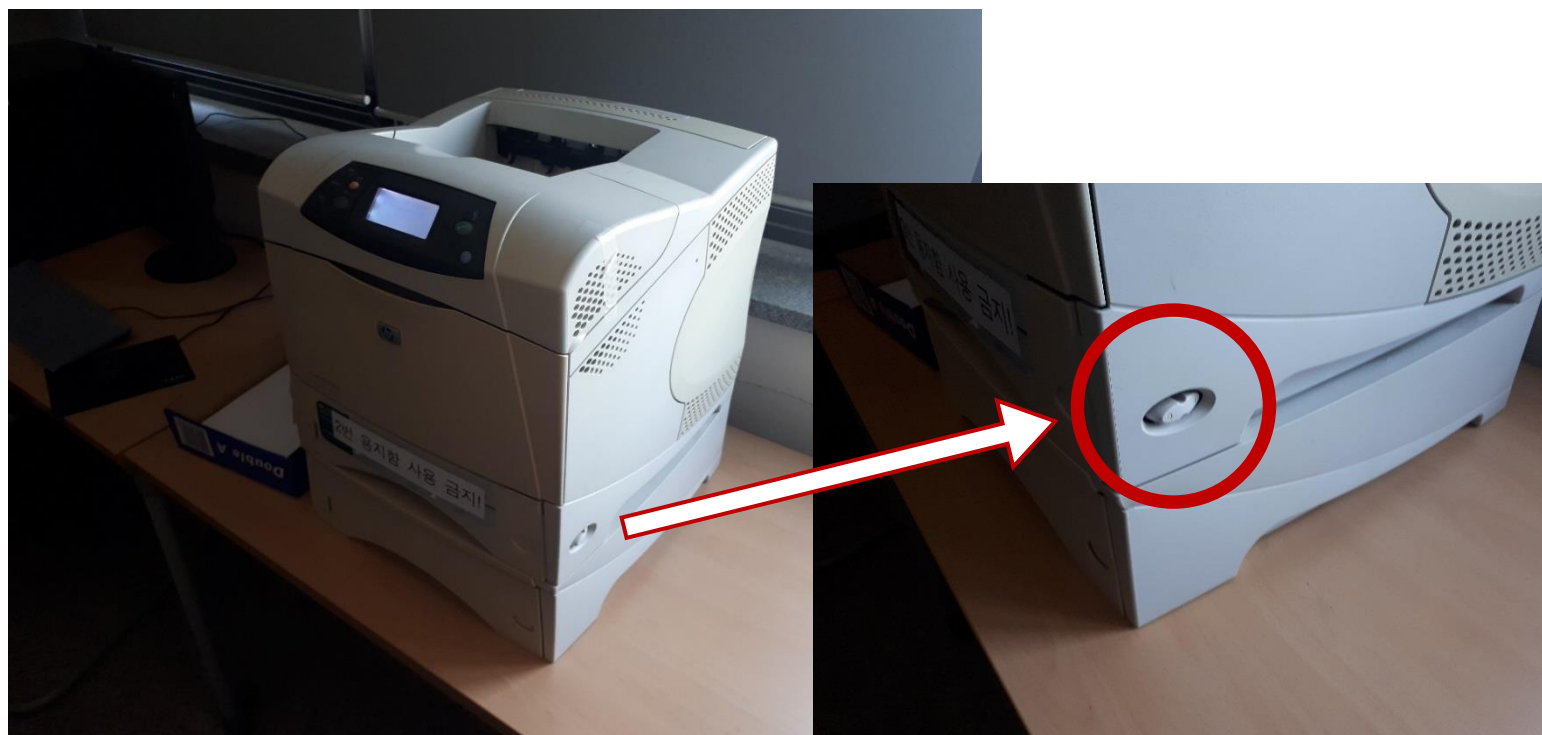
4. **Interpreter**를 종료합니다.

// ml interpreter 종료

// script 종료

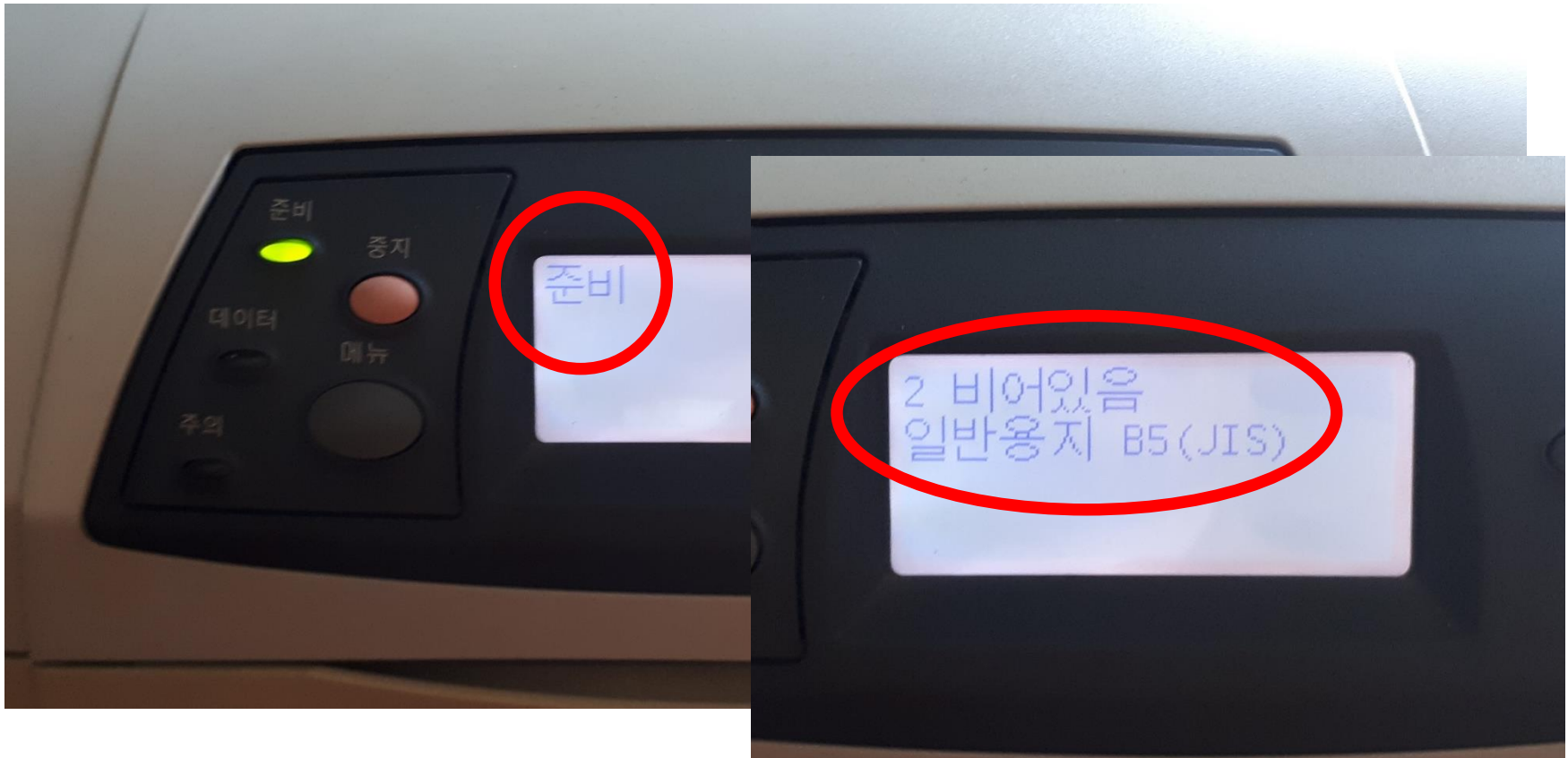
# 프린터 준비

- ▶ 위치: 공대 7호관 622호
  - ▶ Linux 실습실
  - ▶ **사용후 반드시 꺼주세요!!**



# 프린터 상태확인

- ▶ 준비 & 2 비어있음 일반용지 B5(JIS)



# 리눅스 프린터 명령어 (lpq)

- ▶ lpq
  - ▶ 프린터의 현재 상태 확인
  - ▶ 현재 프린트 대기 중인 작업 확인
  - ▶ ex) lpq



s201855077@pl-2019: ~

```
s201855077@pl-2019:~$ lpq
no entries
s201855077@pl-2019:~$
```

```
s201855077@pl-2019:~$ lpq
Warning: no daemon present
Rank  Owner      Job   Files      Total Size
1st   s201855077  13    pow.sml    120 bytes
2nd   s201855077  14    pow.sml    120 bytes
3rd   s201855077  15    pow.sml    120 bytes
s201855077@pl-2019:~$
```

# 리눅스 프린터 명령어 (**lp**, lpr)

- ▶ **lp**, lpr

- ▶ 출력 명령어

- ▶ lp “파일 명”

- ▶ ex) lp p.output

- lp powerset.sml



s201855077@SRElab: ~/ml2

```
s201855077@SRElab:~/ml2$ ls
```

```
powerset.sml
```

```
s201855077@SRElab:~/ml2$ lp powerset.sml
```

```
s201855077@SRElab:~/ml2$
```



# 리눅스 프린터 명령어 (lprm)

## ▶ lprm

- ▶ 프린터 작업 스택에 쌓인 자신의 스택 제거
- ▶ lprm “작업 번호”
- ▶ ex) lprm 17

```
s201855077@pl-2019:~$ lpq
Warning: no daemon present
Rank  Owner      Job  Files      Total Size
1st   s201855077  17   pow.sml    120 bytes
2nd   s201855077  18   pow.sml    120 bytes
3rd   s201855077  19   pow.sml    120 bytes
s201855077@pl-2019:~$ lprm 17
dfA017pl-2019 dequeued
cfA017pl-2019 dequeued
s201855077@pl-2019:~$ lpq
Warning: no daemon present
Rank  Owner      Job  Files      Total Size
1st   s201855077  18   pow.sml    120 bytes
2nd   s201855077  19   pow.sml    120 bytes
s201855077@pl-2019:~$
```

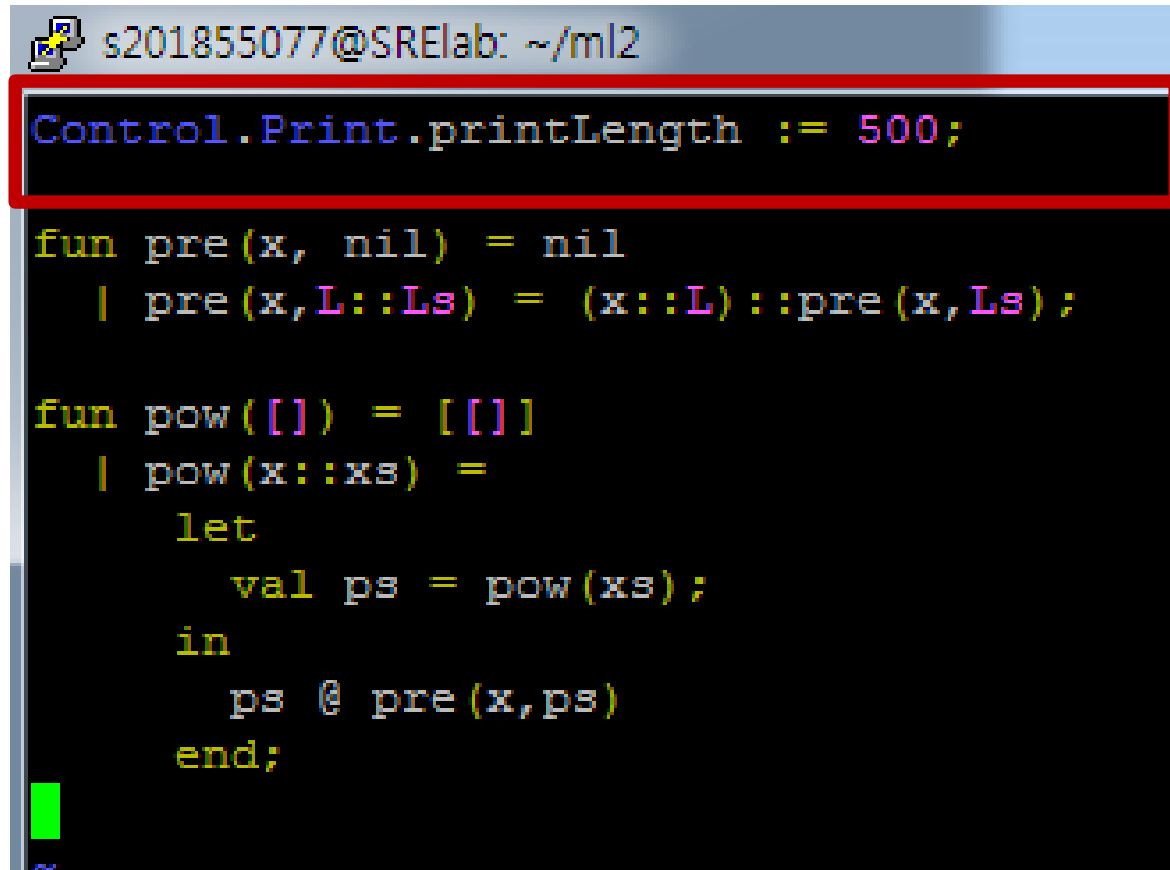
# ML – 모든 리스트 출력

---

```
s201855077@SRElab:~/ml2$ ls
powerset.sml  pow.sml
s201855077@SRElab:~/ml2$ sml
Standard ML of New Jersey v110.78 [built: Thu Jul 23 11:21:58 2015]
- use "pow.sml";
[opening pow.sml]
val pre = fn : 'a * 'a list list -> 'a list list
val pow = fn : 'a list -> 'a list list
val it = () : unit
- pow([1,2,3,4,5]);
val it =
  [[], [5], [4], [4, 5], [3], [3, 5], [3, 4], [3, 4, 5], [2], [2, 5], [2, 4], [2, 4, 5], ...
  : int list list
- 
```

# ML – 모든 리스트 출력

- ▶ `Control.Print.printLength := 500;`

A terminal window with a dark background and light-colored text. The title bar at the top shows a user icon and the text 's201855077@SRElab: ~/ml2'. The first line of code, 'Control.Print.printLength := 500;', is highlighted with a red rectangular box. Below it, there are two function definitions: 'fun pre(x, nil) = nil' followed by a pattern match '| pre(x, L::Ls) = (x::L)::pre(x, Ls);', and 'fun pow([]) = [[]]' followed by a pattern match '| pow(x::xs) = let val ps = pow(xs); in ps @ pre(x, ps) end;'. A green cursor is visible at the end of the last line of code.

```
s201855077@SRElab: ~/ml2
Control.Print.printLength := 500;

fun pre(x, nil) = nil
  | pre(x, L::Ls) = (x::L)::pre(x, Ls);

fun pow([]) = [[]]
  | pow(x::xs) =
    let
      val ps = pow(xs);
    in
      ps @ pre(x, ps)
    end;
```

# ML – 모든 리스트 출력

```
[opening powerset.sml]
[autoloading]
[library $smlnj/compiler/current.cm is stable]
[library $smlnj/compiler/x86.cm is stable]
[library $smlnj/viscomp/core.cm is stable]
[library $smlnj/viscomp/basics.cm is stable]
[library $smlnj/viscomp/elabdata.cm is stable]
[library $smlnj/viscomp/elaborate.cm is stable]
[library $SMLNJ-BASIS/basis.cm is stable]
[library $smlnj/MLRISC/Lib.cm is stable]
[library $SMLNJ-MLRISC/Lib.cm is stable]
[library $smlnj/viscomp/debugprof.cm is stable]
[library $SMLNJ-LIB/Util/smlnj-lib.cm is stable]
[library $smlnj/smlnj-lib/pp-lib.cm is stable]
[library $SMLNJ-LIB/PP/pp-lib.cm is stable]
[library $html-lib.cm(=$SMLNJ-LIB/HTML)/html-lib.cm is stable]
[library $smlnj/MLRISC/Control.cm is stable]
[library $SMLNJ-MLRISC/Control.cm is stable]
[library $controls-lib.cm(=$SMLNJ-LIB/Controls)/controls-lib.cm is stable]
[library $smlnj/smlnj-lib/controls-lib.cm is stable]
[autoloading done]
val it = () : unit

val pre = fn : 'a * 'a list list -> 'a list list
val pow = fn : 'a list -> 'a list list
val it = () : unit
- pow([1,2,3,4,5]);
val it =
  [[], [5], [4], [4, 5], [3], [3, 5], [3, 4], [3, 4, 5], [2], [2, 5], [2, 4], [2, 4, 5], [2, 3],
   [2, 3, 5], [2, 3, 4], [2, 3, 4, 5], [1], [1, 5], [1, 4], [1, 4, 5], [1, 3], [1, 3, 5], [1, 3, 4],
   [1, 3, 4, 5], [1, 2], [1, 2, 5], [1, 2, 4], [1, 2, 4, 5], [1, 2, 3], [1, 2, 3, 5], [1, 2, 3, 4],
   [1, 2, 3, 4, 5]] : int list list
```