

Programming on UNIX/Linux, Vi editor & C++ Sample Program



PROGRAMMING ON UNIX/LINUX, VI EDITOR

목차

- UNIX/Linux 소개
- UNIX/Linux 사용환경
- Vi 편집기
- C++ 프로그래밍

UNIX 소개: 역사 (1/2)

- 1965년 Bell 연구소는 General Electric사와 MIT의 MAC 프로젝트와 공동으로 MULTICS라는 새로운 운영 체제 개발에 참여
 - ◎ UNIX의 밑바탕이 되는 운영체제
 - ◎ 개발 목적
 - ⊙ 많은 수의 사용자들이 동시에 컴퓨터에 접근
 - ⊙ 충분한 계산 능력과 저장 장소를 제공
 - ⊙ 사용자가 원할 경우 자신의 데이터를 서로 공유
 - ◎ 운영 실태
 - ⊙ 1969년에 그 초판이 GE645 컴퓨터에 설치되어 실행
 - ⊙ 원래 의도했던 일반적인 서비스를 제공해 주지 못함
 - ⊙ 개발 목표를 언제 달성할 수 있을지 명확하게 예측 불가
 - ⊙ Bell 연구소는 이 프로젝트에서 손을 땀

UNIX 소개: 역사 (2/2)

- Ken Thompson

- ◎ “UNIX 시스템의 위대한 아버지”
- ◎ Thompson은 Ritchie와 함께 UNIX 파일 시스템의 초기 버전과 프로세스 서브시스템, 그리고 몇 개의 유틸리티 프로그램들을 포함하는 시스템을 PDP-7에서 구현
- ◎ 이 새로운 시스템에 UNIX라는 이름이 붙여짐

- Dennis Ritchie

- ◎ C언어로 UNIX 시스템 재 작성 : 1973년

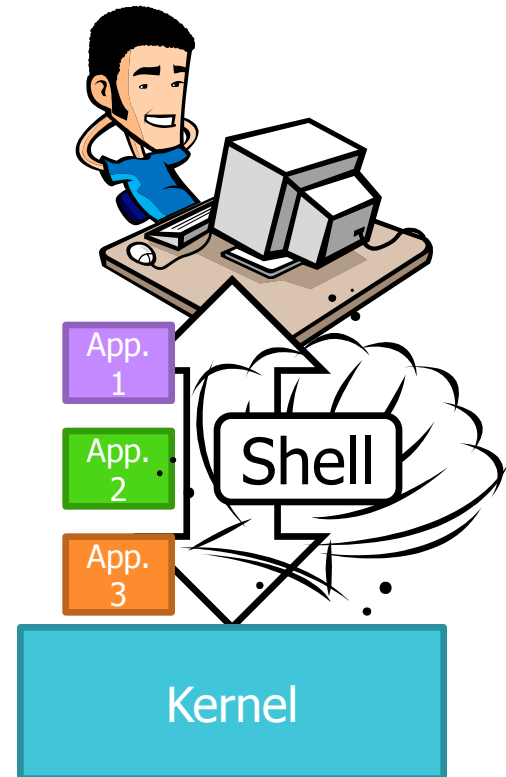
- 대학에 보급되기 시작: 1974년

- BSD 계열과 System V 계열로 양분됨: 1978년

- ◎ BSD 계열은 주로 대학과 관련됨 – SunOS의 근간
- ◎ System V 계열은 주로 기업과 관련됨 – 각종 대기업에서 자체 도입한 UNIX들의 근간

UNIX 소개: 특징

- 대화형 시스템
 - ⊙ 사용자가 명령을 내리면 시스템은 그 명령을 수행하고 결과를 알려준다.
- 높은 이식성
 - ⊙ C언어로 구성되어 있어 타 기종에 이식이 용이
- Multi-tasking, Multi-user 시스템
 - ⊙ 여러 명의 사용자가 동시에 사용가능
 - ⊙ 동시에 여러 개의 프로세서 수행이 가능
- 셸 프로그래밍
 - ⊙ 반복적이고 복잡한 명령어를 대화식으로 간단히 처리



Linux 소개

- Andrew S. Tanenbaum: 1987년

- ◎ 운영체제 강의를 위해 개발한 유닉스의 PC 버전 소스코드
- ◎ MINIX라는 이름으로 공개

- Linus Torvalds: 1991년

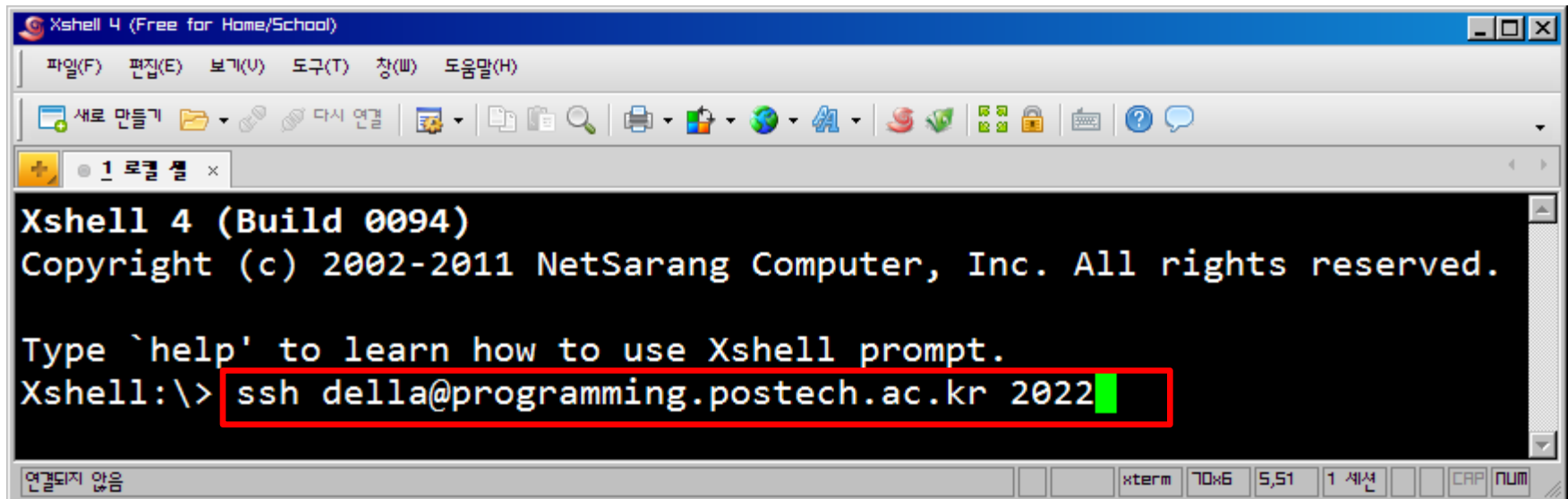
- ◎ 핀란드 헬싱키 대학의 학생
- ◎ MINIX를 기반유닉스 표준화 규격인 POSIX(Portable Operating System Interface for Unix)를 도입하여 PC버전의 유닉스 개발

- 특징

- ◎ 소스 코드를 공개하여 수백만의 개발자가 지속적으로 업그레이드 되고 있음
- ◎ UNIX와 유사한 환경을 제공
- ◎ 무료로 배포되는 특징 때문에 주로 학교, 기업, 공공기관 등에 널리 보급

UNIX/Linux 사용 환경

- 원격 Shell 접속 프로그램
 - ◎ **Putty**, Xshell, ...
- **Xshell**을 이용한 접속 방법
 - ◎ **ssh** user_id@programming.postech.ac.kr 2022
- 접속 과정



The screenshot shows the Xshell 4 (Free for Home/School) application window. The title bar reads "Xshell 4 (Free for Home/School)". The menu bar includes "파일(F)", "편집(E)", "보기(V)", "도구(T)", "창(W)", and "도움말(H)". The toolbar contains icons for "새로 만들기", "열기", "다시 연결", "찾기", "인쇄", "복사", "붙여넣기", "자동 저장", "종료", "도움말", and "대화". The main terminal area displays the following text:

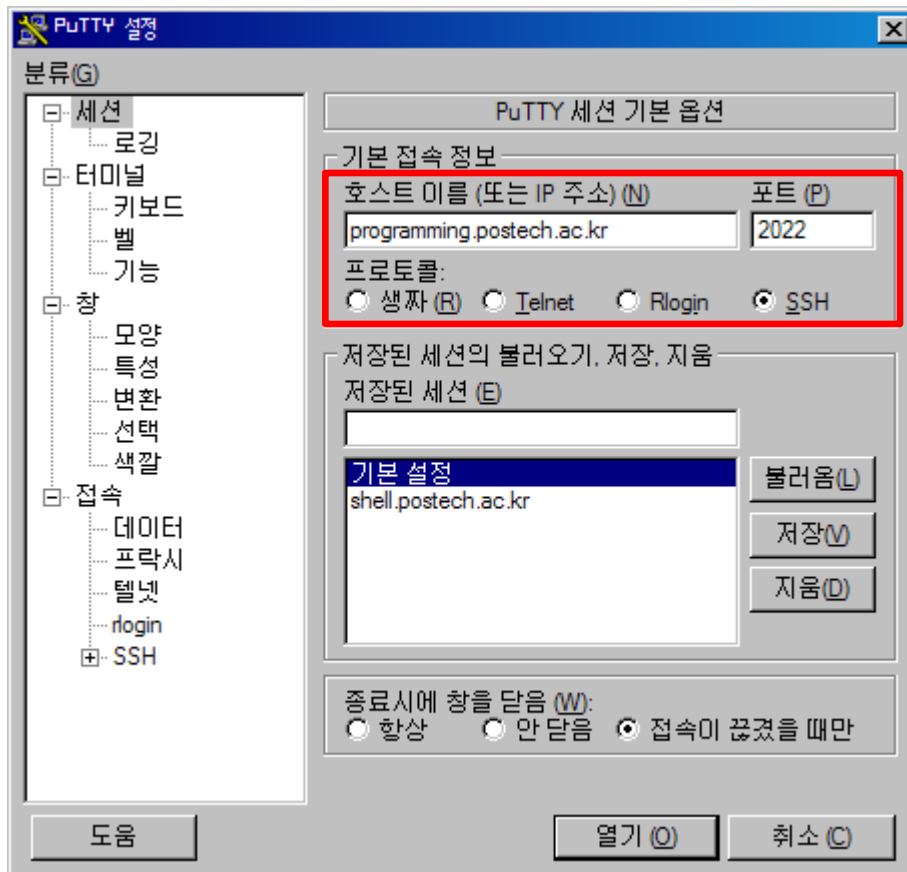
```
Xshell 4 (Build 0094)
Copyright (c) 2002-2011 NetSarang Computer, Inc. All rights reserved.

Type `help' to learn how to use Xshell prompt.
Xshell:\> ssh della@programming.postech.ac.kr 2022
```

The command "ssh della@programming.postech.ac.kr 2022" is highlighted with a red rectangle. The status bar at the bottom shows "연결되지 않음", "xterm", "70x6", "5,51", "1 세션", and "CAP NUM".

UNIX/Linux 사용 환경

- 원격 Shell 접속 프로그램
 - Putty, Xshell, ...
- Putty를 이용한 접속 방법
- 접속 과정



기본접속정보

- 호스트이름
programming.postech.ac.kr
- 포트
2222
- 프로토콜
SSH

UNIX/Linux 명령어: 정보 관리

- whoami: 자신의 계정 이름 보기

```
$ whoami  
della
```

- passwd: 패스워드 변경 명령

```
$ passwd  
Changing password for 'della' in cell 'postech.ac.kr'.  
Old password:  
New password (RETURN to abort):  
Retype new password:  
$
```

- who: 현재 시스템에 로그인한 사용자 보기
- finger <username>: 해당 사용자에게 대한 정보 보기

UNIX/Linux 명령어: 디렉토리

- **pwd**: 현재 디렉토리가 어디인지 표시

```
$ pwd  
/home/std/user_id
```

- **cd <directory>**: 해당 디렉토리로 위치를 변경

- ⦿ **cd ..**: 이전 상위 디렉토리로 위치 변경
- ⦿ **cd .**: 현재 디렉토리로 이동 (무의미함)
- ⦿ **cd ../abc**: 상위 디렉토리의 abc 디렉토리로 이동
- ⦿ **cd /**: 최상위(root) 디렉토리로 이동

```
$ cd ..  
$ pwd  
/home/std
```

- **ls**: DOS의 dir 명령

- ⦿ **ls**: 파일 리스트만 나열 (숨겨진 파일 표시 안함)
- ⦿ **ls -l**: 각 파일에 대한 속성, 크기, 날짜 등을 상세히 보여줌
- ⦿ **ls -a**: 파일 리스트 나열, 숨겨진 파일 포함
- ⦿ **ls -al**: 숨겨진 파일 포함, 각 파일의 상세한 정보 출력
- ⦿ **ls -R**: 서브디렉토리 내의 파일도 표시

UNIX/Linux 명령어: 파일 (1/3)

○ cp: 파일/디렉토리 복사 명령

- ⊙ cp srcfile destfile: srcfile을 destfile이란 이름으로 복사
- ⊙ cp srcfile destdir: srcfile을 destdir 디렉토리로 복사
- ⊙ cp -R srcdir destdir: srcdir 디렉토리 및 그 안의 내용을 (디렉토리가 안에 있어도 상관없음) destdir 디렉토리로 복사
- ⊙ cp -f srcfile dest: srcfile을 dest(파일 혹은 디렉토리)로 복사하는데, 이미 존재하는 경우 강제로 덮어씀
- ⊙ cp -a srcfile dest: srcfile을 dest로 복사하는데, 원본 파일의 속성을 유지함

○ mv: 파일/디렉토리 이동 명령

- ⊙ mv src dest: src의 내용을 dest로 옮김
- ⊙ src는 파일 혹은 디렉토리
- ⊙ dest는 src가 파일이면 파일 혹은 디렉토리, src가 디렉토리면 디렉토리
- ⊙ mv -f src dest: src의 내용을 dest로 옮기는데, 이미 존재하는 파일을 무시하고 덮어씀

○ rm: 파일(+디렉토리) 삭제 명령

- ⊙ rm filename: filename이라는 파일을 삭제
- ⊙ rm -f filename: filename이라는 파일을 강제로 삭제
- ⊙ rm -rf dirname: dirname 디렉토리 및 그 안의 내용을 강제로 삭제
- ⊙ 주의: 현재 경로를 반드시 확인하고 명령 수행-복구 불가능

UNIX/Linux 명령어: 파일 (2/3)

- **mkdir** dir: dir이라는 디렉토리를 새로 생성
- **rmdir** dir: dir이라는 디렉토리를 삭제 (단, 디렉토리 안에 파일이나 디렉토리가 없어야 함)
- **find**: 특정 패턴을 가지는 파일을 찾는 명령
 - ⊙ **find dir -name filename -print**: dir이라는 디렉토리 내에 filename이라는 패턴을 가지는 파일을 찾아내서 출력함
- **grep**: 특정 패턴의 텍스트를 찾는 명령
 - ⊙ **grep <pattern> filename**: filename이라는 파일 내에 지정한 패턴의 텍스트가 있는 줄을 표시
 - ⊙ E.g., **grep include test.c**: test.c 내에서 include라는 문자열이 있는 줄을 전부 표시

UNIX/Linux 명령어: 파일 (3/3)

- `cat <filename>`: text file을 표준 출력으로 출력, 도스의 `type`과 비슷
- `less/more`: 긴 텍스트를 효과적으로 보여주는 프로그램
 - 다른 커맨드들과 조합해서 사용할 때 편리
 - `less <filename>`
 - `cat <filename> | less`
- `exit`: shell 종료

UNIX/Linux 명령어: 도움말

- <명령어 이름> --help (-h)

```
$ ls --help
```

- ⦿ ls에 대한 도움말이 출력

- man (번호) <명령어 이름>

- ⦿ 해당 명령어(항목)에 대한 매우 자세한 정보(매뉴얼)를 표시
- ⦿ 같은 이름의 명령어 혹은 항목이 여러 개인 경우 명령어 이름 앞에다가 번호를 붙여서 정확히 표시하도록 지정하는 경우가 있음

```
$ man ls
```

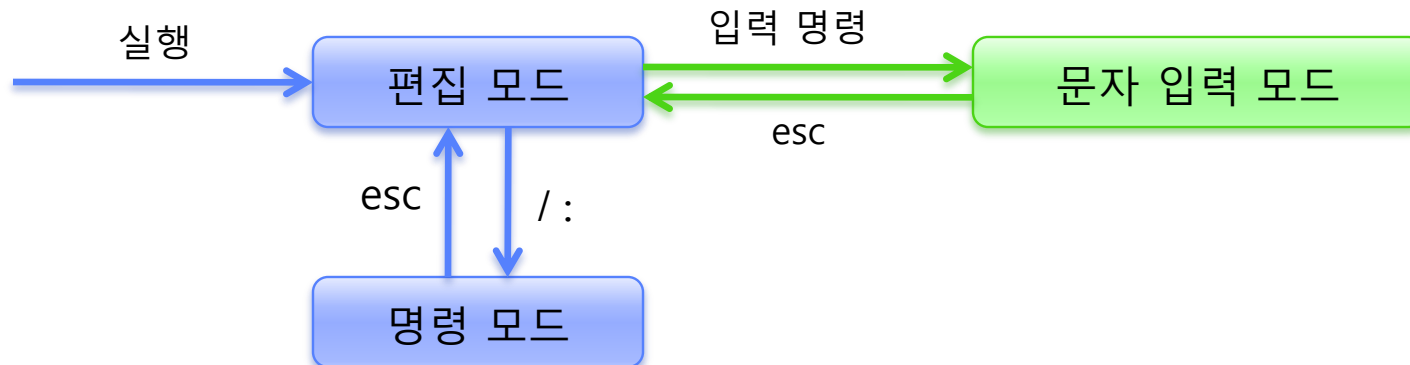
- ⦿ ls 명령에 대한 상세한 매뉴얼을 표시

Vi 편집기: 소개

- UNIX/Linux에서 가장 광범위하게 사용되는 문서편집기
 - ◎ 프로그래밍 환경에 최적화
- Vim: Vi Improved의 약자로 Vi의 보강된 버전
- 실행 방법
 - ◎ vi / vim filename

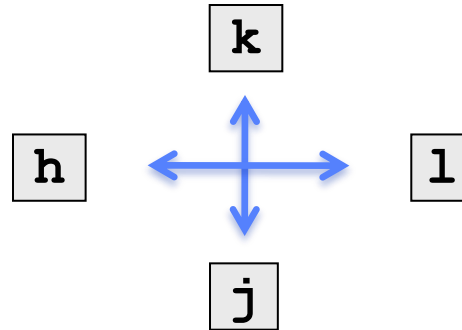
```
$ vi myfile
```

- 편집 모드와 명령 모드로 나누어짐



VI 편집기: 이동

커서(Cursor) 이동:



w	다음 단어로
e	현재 단어의 끝으로
b	이전 단어로
0	현재 줄의 맨 처음으로
\$	현재 커서가 있는 줄의 끝
G	현재 문서의 마지막 줄
ctrl + f	현재 화면을 한 화면 뒤로 이동
ctrl + b	현재 화면을 한 화면 앞으로 이동

* 숫자 조합 : 반복 또는 정량 설정을 의미함. ex) 3w

* vi 의 대부분 명령어는 숫자 조합이 가능

VI 편집기: 입력

i	Insert, 현재 커서의 위치에서 입력
I	Insert, 커서가 있는 줄(line)의 맨 앞에서 입력
a	Append, 현재 커서 위치의 다음 칸에서 입력
A	Append, 커서가 있는 줄(line)의 맨 뒤에서 입력
o	Open line, 현재 커서가 있는 줄 밑에 한 줄 추가하면서 그 줄 첫 칸에서 입력
O	Open line, 현재 커서가 있는 줄 위에 한 줄 추가하면서 그 줄 첫 칸에서 입력
s	Substitute, 현재 커서의 위치에서 한 글자를 지우면서 입력
S	Substitute, 현재 줄을 모두 지우면서 입력
CW	Change word, 현재 커서가 위치한 위치부터 해당 단어를 지우고 커서가 시작했던 위치에다가 입력

VI 편집기: 편집 기능 (1/2)

x	현재 글자를 잘라냄
dd	현재 줄을 잘라냄
d + 이동명령	'커서를 이동하면서 잘라낸다'는 느낌으로 쓸 수 있음
dw	단어 단위 잘라냄. dw, de는 커서 부터. db는 커서 앞으로
de	
db	
dj	줄 단위 잘라냄
dk	
dh	글자 단위 잘라냄
dl	

p	커서 뒤에 붙여 넣기 (줄 붙여 넣기의 경우 아래쪽으로)
P	커서 앞에 붙여 넣기 (줄 붙여 넣기의 경우 위쪽으로)

VI 편집기: 편집 기능 (2/2)

yy	현재 줄을 복사
y + 이동명령	'커서를 이동하면서 복사한다'는 느낌으로 쓸 수 있음
yw	단어 단위 복사
ye	
yb	
yj	줄 단위 복사
yk	
yh	글자 단위 복사
yl	

VI 편집기: 기타 명령어 (1/2)

○ Visual Block

<code>v + 이동</code>	Visual 설정
<code>ctrl + v + 이동</code>	Visual Block 설정
<code>esc</code>	Visual (or Visual Block) 선택 취소

○ Indentation

<code>>></code>	줄의 앞쪽에 Tab 추가
<code><<</code>	줄의 앞쪽에서 Tab 삭제
<code>Visual + 숫자 + ></code>	복수 개 TAB 추가
<code>Visual + 숫자 + <</code>	복수 개 TAB 삭제
<code>J</code>	아랫줄 당겨 붙이기

VI 편집기: 기타 명령어 (2/2)

○ 명령 반복, 취소 및 재실행

.	이전 명령 반복
u	이전 명령 취소 (다수의 명령 취소 가능)
ctrl + r	취소한 명령 재실행 (다수의 명령 재실행 가능)

○ 기타

%	{ }, (), [] 등에서 실행하면 짝을 찾아줌
~	대 소문자 바꾸기

VI 편집기: 저장 & 종료

:w	저장
:w 파일이름	다른 이름으로 저장
:w!	(읽기 전용 파일의 경우) 강제 저장
:q	종료
:q!	(변경된 파일의 경우 저장 안하고) 강제 종료
:wq	저장하고 종료

VI 편집기: 명령모드 (찾기, 바꾸기)

/ + 문자열 + enter	커서 위치에서 아래 방향으로 검색
? + 문자열 + enter	커서 위치에서 위 방향으로 검색
n	계속 검색 (편집 모드 명령)

:범위s/찾을문자열/바꿀문자열/옵션		
범위	2	Line 2
	.	현재 줄
	2,10	Line 1 ~ Line 10
	2,\$	Line 2 ~ 파일 끝
	%	파일 전체
옵션	g	한 줄에서 여러 개를 찾았을 경우 이 옵션이 없으면 처음 것만 바꿈
	c	바꿀 때마다 확인하기
	gc	←이거 쓰세요



VI 편집기: 명령모드 (기타)

:help	도움말 파일을 열어줌
:숫자	해당 줄로 이동
:set	환경 설정 표시

:set 옵션	
:set number	열 번호 표시
:set nonumber	열 번호 표시 안함

다양한 옵션은 **.vimrc** 파일에
미리 설정할 수 있습니다.

.vimrc : vi 설정 파일
본인의 홈 디렉토리에 저장

```
set nocompatible
set nu
set cin
set nowrap
set listchars=extends:,>,precedes:<
set t_kb=^?
set showmatch
syntax on
colorscheme elflord
set bg=dark
```

실습

① \$ vi test.c

② i (편집모드로) 누른 후 아래와 같이 입력

```
#include<iostream>

using namespace std;

int main()
{
    cout << "Hello C++~!!\n";

    return 0;

}
```

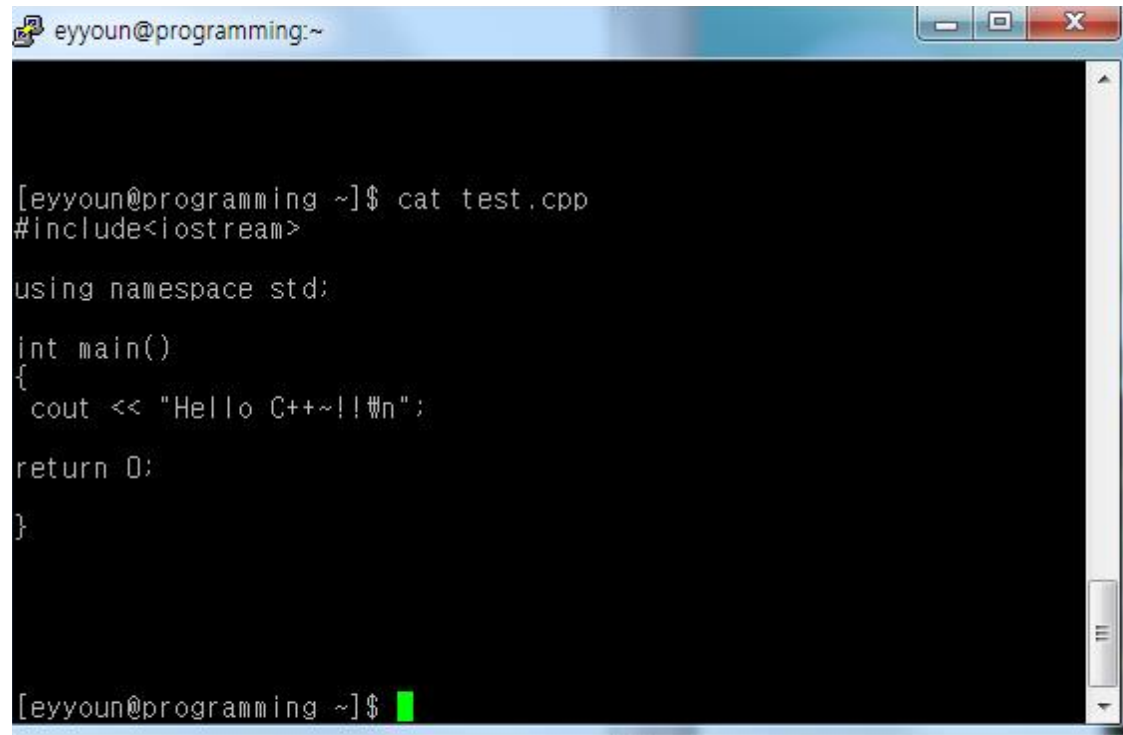
③ Esc 명령 모드에서 : w 눌러서 저장 : q 눌러서 종료

또는 : w q 눌러서 저장하고 종료

변경된 파일의 경우 저장 안하고) 강제 종료 : q !

실습

```
$ cat test.cpp
```

A terminal window titled 'eyyoun@programming:~' with standard window controls. The terminal displays the output of the 'cat test.cpp' command, showing C++ code for a program that prints 'Hello C++!!\n'. The code includes the iostream header, uses the std namespace, and has a main function that outputs the string and returns 0. A green cursor is visible at the end of the prompt line.

```
eyyoun@programming:~  
[eyyoun@programming ~]$ cat test.cpp  
#include<iostream>  
  
using namespace std;  
  
int main()  
{  
    cout << "Hello C++!!\n";  
    return 0;  
}
```

```
[eyyoun@programming ~]$
```

C++ 프로그래밍: 소개

- Windows Visual Studio에서는...
 - ⊙ Solution이란 이름으로 컴파일, 디버깅 등을 모두 함께 관리
- 그러나, Unix/Linux 상에서는 각각의 역할이 분리
- C-컴파일러
 - ⊙ cc: 표준 C 컴파일러
 - ⊙ gcc: GNU C 컴파일러
 - ⊙ g++: GNU C++ 컴파일러
- 컴파일 작업 관리기: make
- 디버깅: gdb

C++ 프로그래밍: g++ 컴파일

- 기본적인 gcc 명령어 포맷

gcc [option list] source.c [-o source.out] [option list]

```
$ g++ source.cpp
```

- ⦿ source.c 를 컴파일
- ⦿ 실행파일은 지정해주지 않으면 default로 a.out으로 명명

```
$ g++ source.cpp -o source.out
```

- ⦿ 실행파일 이름을 source.out 으로 지정

```
$ g++ source.cpp -o source.out -g
```

- ⦿ -g 옵션은 실행파일에 디버깅 정보를 추가해서 gdb로 디버깅하는데 사용

실습

```
$ g++ test.cpp -o test.out  
$ ls  
...  
$ ./test.out
```

```
[eyyoun@programming ~]$ g++ test.cpp -o test.out  
[eyyoun@programming ~]$ ./test.out  
Hello C++~!!  
[eyyoun@programming ~]$ █
```

**** 리눅스에서 실행파일 실행하는 방법 ****

./실행파일명

의미 : 현재 디렉터리에 있는 “실행파일명” 실행
(실행 시킬 때, 파일 경로를 명확히 해 주어야 합니다.)

참고 자료

- Vi/VIM의 활용

- ◎ <http://wiki.kldp.org/wiki.php/DocbookSgml/Vim-KLDP>