

# Computer Programming

## Lecture 3

이윤진  
서울대학교  
2007.12.27.

# Slide Credits

- 엄현상 교수님
  - 서울대학교 컴퓨터공학부
  - Computer Programming, 2007 봄학기

# Editors

# 순서

- Editors
  - vi
  - emacs
- Q&A

# Editors

- Vi (VIsual)
  - Text Editor
    - Interactive Computer Program That the User Controls to Enter or Change Text in a File
  - <http://www.vim.org/>
- Emacs (Editor MACroS)
  - Text Editor
  - <http://www.gnu.org/software/emacs/>

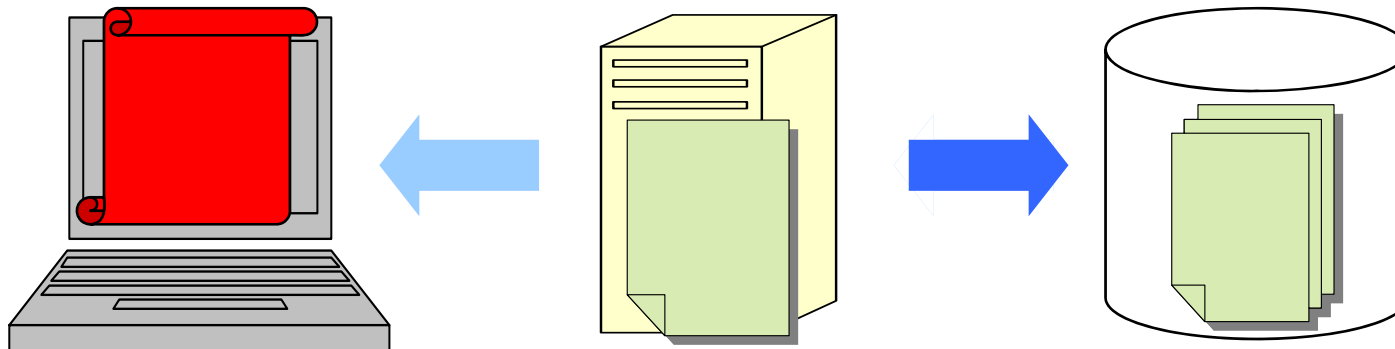
# Editors (계속)

- Vi vs Emacs
  - Vi
    - Simple; Easy to Learn
    - Screen-Oriented
  - Emacs
    - Suitable for Handling Collections of Files
    - Window-Oriented
    - Allowing the User to Do Everything the System Does
    - Customizable

# vi

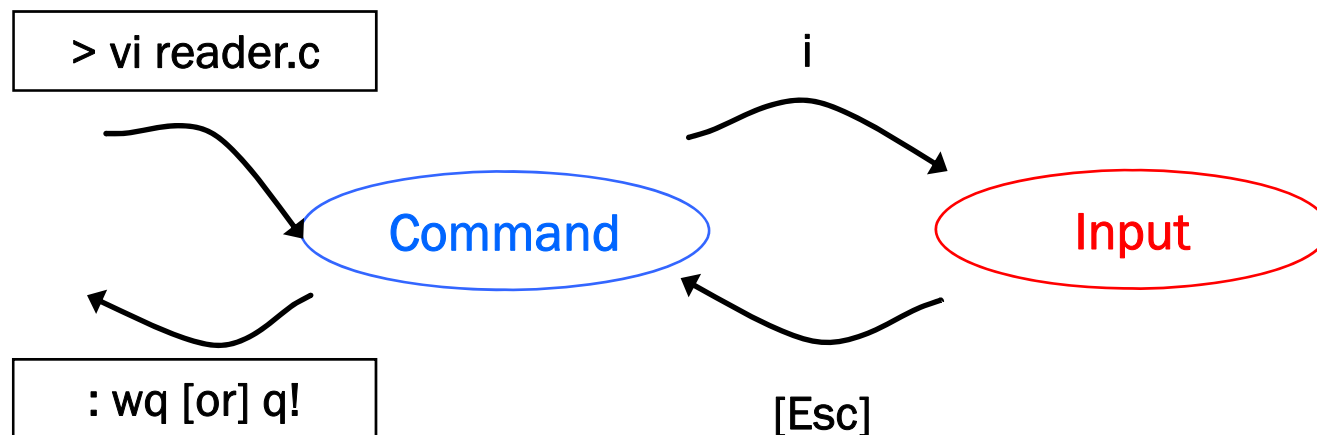
```
martini:~>vi reader.c
```

- vi Command
  - Get the File for Editing
  - Copy the File to a Buffer
  - Have Editing Done in the Buffer
  - Have the Original File Changed When the Buffer Is Saved



# vi Modes

- Command Mode
  - Keystrokes Treated as vi Commands
  - Started When Entering vi
- Input Mode
  - Keystrokes Interpreted Literally
  - Keystrokes Added to the Buffer
  - Permitting Editing Text



# vi Summary

General Administration	Move Cursor	Add Text	Delete Text
:wq, ZZ - write, quit :q! - quit u - undo	h - left l - right j - down k - up G - EOF or last line nG - line <i>n</i> w, W - word	a - append i - insert o, O - open line	x, X - character dd - line ndd - <i>n</i> lines dw, dW - word

Modify Text	Search for Text	Copy/Move Text	Global Changes
cw, cW - change word R - replace rX - replace character s - substitute	/ <b>text</b> , ? <b>text</b> - locate n, N - next	ny - yank text p, P - put text back ndd - delete lines	:start line, endline s/ <b>old</b> / <b>new</b> /g - change old to new between startline and endline  :g/ <b>old</b> /s// <b>new</b> /g - change old to new everywhere in the file

# vim (Vi IMproved)

- Improved version of the vi editor
  - Unlimited undo
  - Portability
  - Syntax highlighting
  - GUI
  - vi compatibility mode
  - Split windows
  - Visual mode

# Syntax Highlighting

- C, C++, HTML, Matlab, Make file, ...
- Customizing syntax highlighting
- See `.vimrc`



The screenshot shows a PuTTY terminal window titled "vision.postech.ac.kr - PuTTY". The terminal displays the following Vim configuration for syntax highlighting:

```
highlight Comment ctermfg=6
highlight Include ctermfg=6 cterm=bold
highlight Statement ctermfg=3
highlight String ctermfg=4 cterm=bold
highlight Type ctermfg=2
highlight Function cterm=bold
highlight PreCondit ctermfg=6 cterm=bold
highlight Error ctermfg=1 cterm=bold
```

Below the configuration, a double quote character is shown. The terminal then displays the contents of the `.vimrc` file, with a status bar at the bottom showing ".vimrc 17,1 19%". The code in `.vimrc` is:

```
/* comment! */
// comment 2
int main(int argc, char *argv[])
{
    int fdold, fdnew, count;

    (argc !=3){
        exit(1);
    }

    ((fdold = open(argv[1], O_RDONLY)) == -1){
```

At the bottom of the terminal, the status bar shows "tmp.c [RO] 5,1 25%".

# Multi-Windows

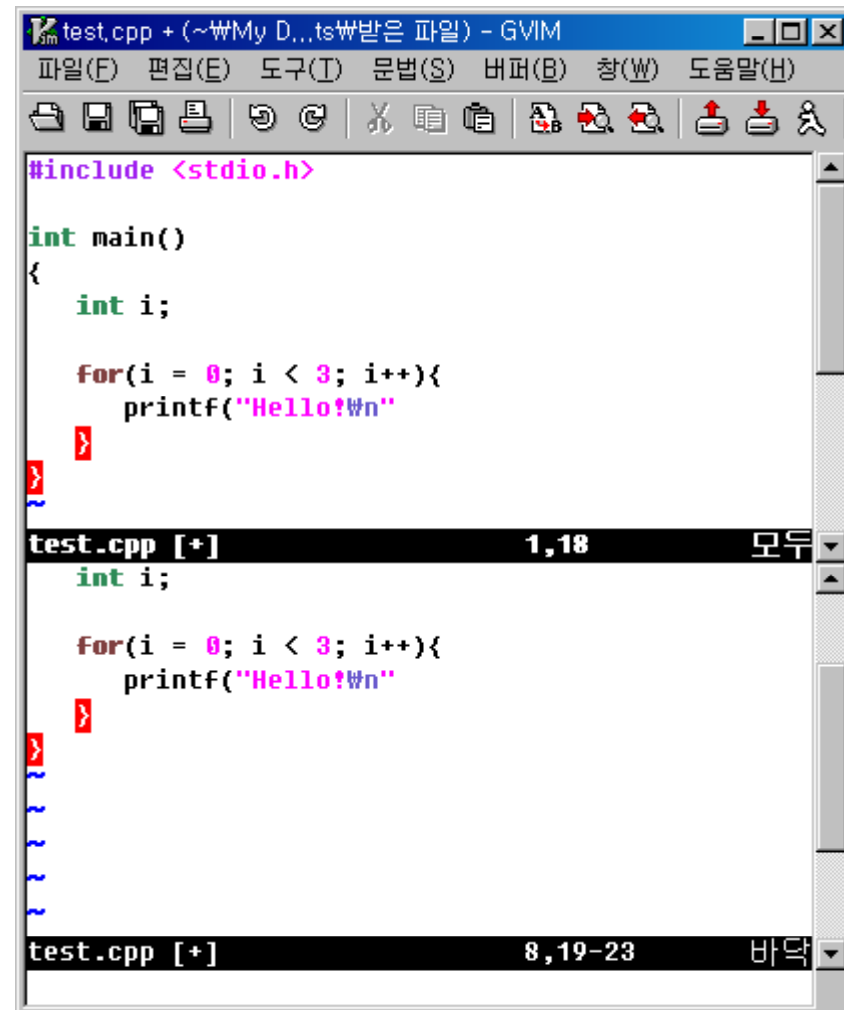
- Window split

sp [file], C-wn

- Window switch

C-ww, C-wk, C-wj

C- : [CtrlI]



```
test.cpp + (~\My D...ts\받은 파일) - GVIM
파일(F) 편집(E) 도구(T) 문법(S) 버퍼(B) 창(W) 도움말(H)
#include <stdio.h>

int main()
{
    int i;

    for(i = 0; i < 3; i++){
        printf("Hello!\n")
    }
}

test.cpp [+] 1,18 모두
int i;

for(i = 0; i < 3; i++){
    printf("Hello!\n")
}

test.cpp [+] 8,19-23 바닥
```

# Visual Mode

- Start visual mode

v: character-by character

V: line-by line

- End visual mode

ESC

- Editing a highlighted region

d: delete

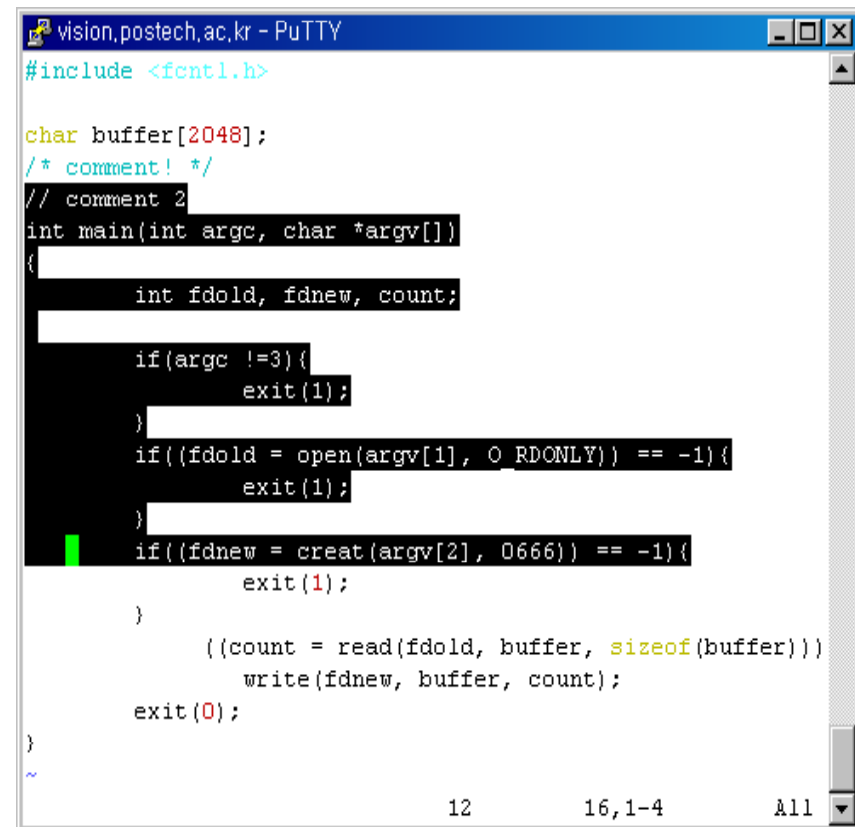
c: change

y: yank

>: shifts the region one shift width to the right

<: shifts the region one shift width to the left

J: join lines



```
vision.postech.ac.kr - PuTTY
#include <fcntl.h>

char buffer[2048];
/* comment! */
// comment 2
int main(int argc, char *argv[])
{
    int fdold, fdnew, count;

    if(argc !=3){
        exit(1);
    }
    if((fdold = open(argv[1], O_RDONLY) == -1){
        exit(1);
    }
    if((fdnew = creat(argv[2], 0666) == -1){
        exit(1);
    }

    ((count = read(fdold, buffer, sizeof(buffer)))
    write(fdnew, buffer, count);
    exit(0);
}
~
12      16,1-4      All
```

# Miscellaneous

- Running a shell command within vi

```
!command
```

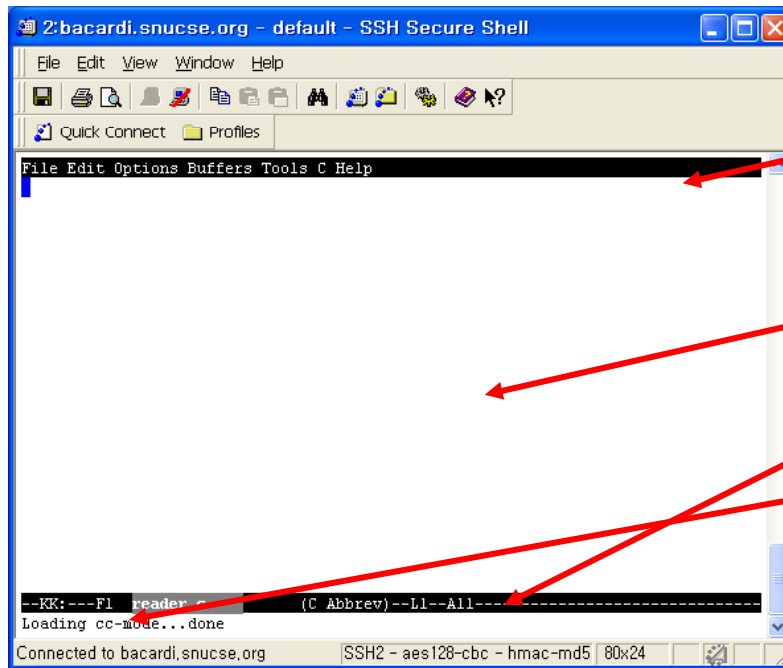
- Adding Customizations to `~/ .vimrc`

```
set ts=4
```

- ...

# Emacs

```
martini:~>emacs reader.c
```

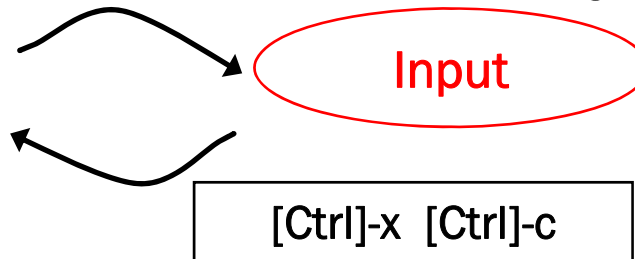


- (Initial) Screen
  - Menu Bar (X Window System)
  - Window
  - Status Line
  - Minibuffer
    - Typing names
    - Typing commands
    - Printing messages

```
> emacs reader.c
```

**Input**

```
[Ctrl]-x [Ctrl]-c
```



# Emacs Status Line

The screenshot shows the Emacs status line in a terminal window titled "bacardi.snucse.org - bacardi - SSH Secure Shell". The status line is: `--KK:++-F1 hello.txt (Text)--L2--All-----`. Annotations explain the parts:

- coding system:** `--KK`
- Frame's name:** `++`
- name of the buffer:** `hello.txt`
- major mode:** `(Text)`
- line number:** `--L2`
- location:** `--All`

Legend for buffer status symbols:

- `**` : if the buffer is modified
- `--` : if the buffer is not modified
- `%%` : read-only text

Bottom status bar: Connected to bacardi.snucse.org | SSH2 - aes128-cbc - hmac-md5 | 66x18

# Emacs Summary

C- : [Ctrl]

ESC : [ESC]

General Administration	Move Cursor	Add Text	Delete Text
<p>C-x C-s – save                      C-x C-c – quit                      C-_ – undo                      C-x u – undo</p>	<p>C-p – previous line                      C-n – next line                      C-f – forward char                      C-b – backward char                      C-a – beginning of the line                      C-e – end of the line</p>	<p>Just type text – No input/edit mode in vi</p>	<p>C-k – line from cursor                      C-a C-k – entire line                      C-backspace – ← key                      C-d – DEL key                      C-@ – select region as mark set                      C-w – delete mark set</p>

Modify Text	Search for Text	Copy/Move Text	Global Changes
<p>ESC x overwrite-mode – turn on/off overwrite mode</p>	<p>C-s – start incremental search forward (isearch-forward)                       C-r – start incremental search backward (isearch-backward)</p>	<p>C-@, ESC w – copy text                      C-y – paste                       C-@, C-w – cut text                      C-y – paste</p>	<p>ESC x replace-string – type old and new</p>

# Emacs Multiple Windows

- Horizontal Split (into 2 Windows)

C-x 2

- Vertical Split (into 2 Window)

C-x 3

- Window Switch

C-x o

- Current Window Deletion

C-x 0

- All the Other Windows Deletion

C-x 1

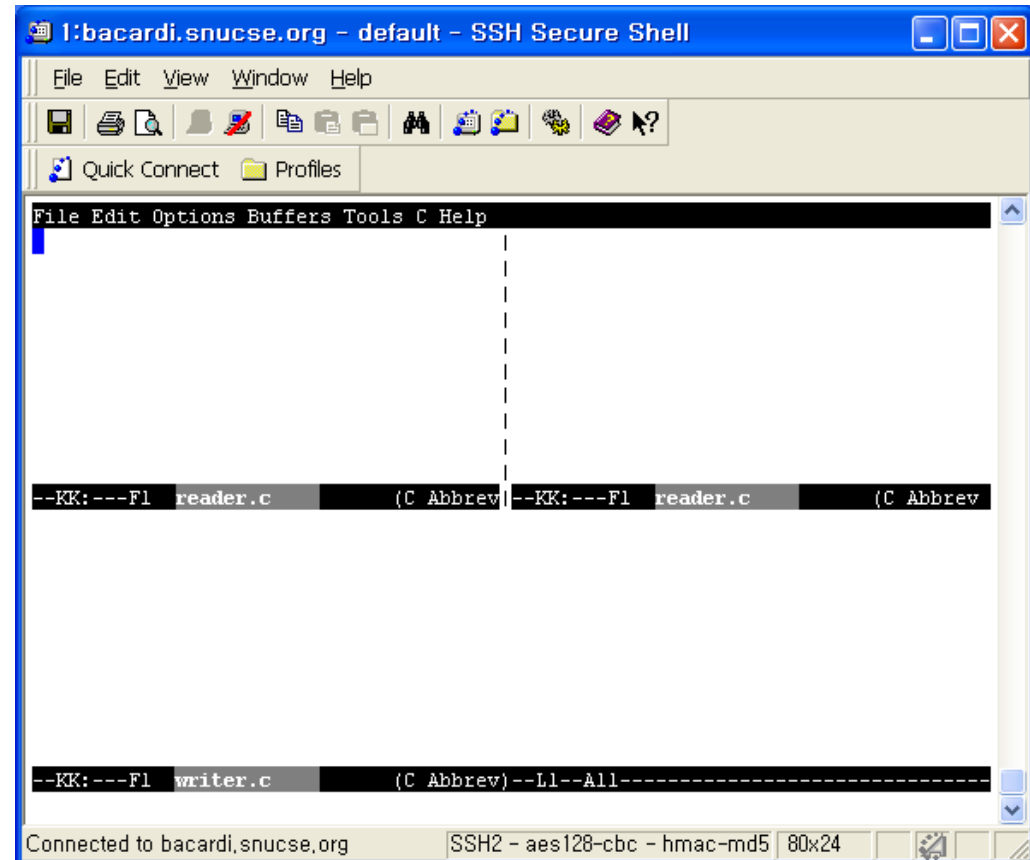
# Emacs Multiple Windows (계속)

- Getting another File into the Current Window

C-x C-f

```
> touch writer.c  
> emacs reader.c  
C-x 2  
C-x 3  
C-x o  
C-x o  
C-x C-f  
Find file: ~/writer.c  
C-x o
```

Result



# Miscellaneous

- Running a Unix Shell within Emacs

```
ESC x shell
```

- Adding Customizations to `~/.emacs`

```
(setq c-indent-level 5)
```

- ...