# Lecture Note 2.
# Programming Environment

September 15, 2025

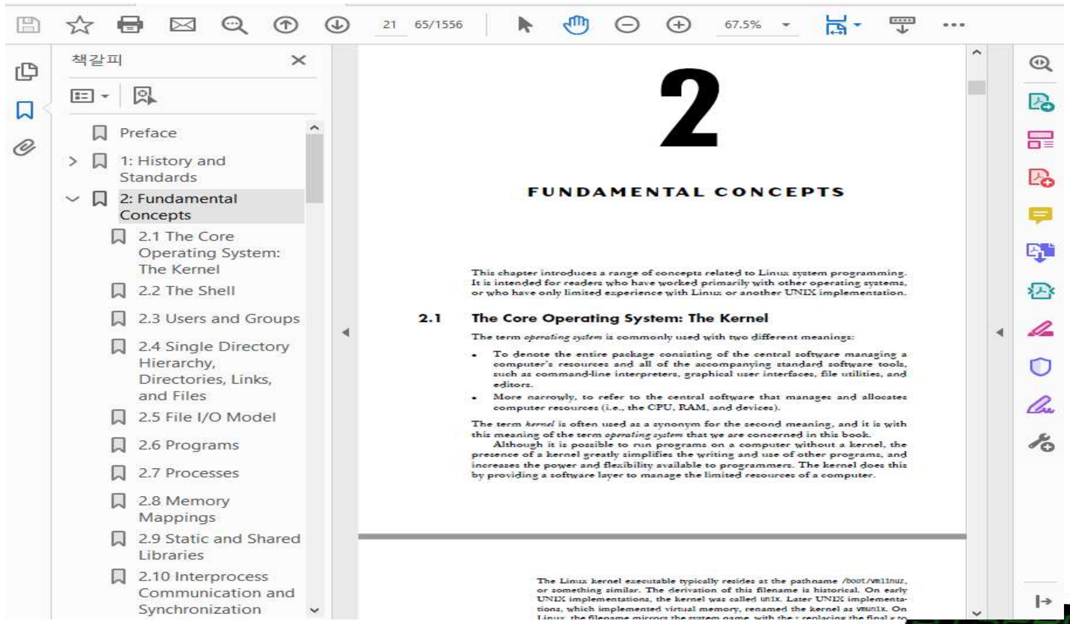
Jongmoo Choi
Dept. of Software
Dankook University
http://embedded.dankook.ac.kr/~choijm

(본 교재는 **2025**년도 과학기술정보통신부 및 정보통신기획평가원의 '**SW**중심대학사업' 지원을 받아 제작 되었습니다.)

# Objectives

- Discuss the history of Linux
- Understand key concepts of Linux
- Learn how to access Linux
- Learn how to use commands in Linux
- Learn how to make programs in Linux

- Refer to Chapter 1, 2 in the LPI



2

FUNDAMENTAL CONCEPTS

This chapter introduces a range of concepts related to Linux system programming. It is intended for readers who have worked primarily with other operating systems, or who have only limited experience with Linux or another UNIX implementation.

2.1 The Core Operating System: The Kernel

The term *operating system* is commonly used with two different meanings:

- To denote the entire package consisting of the central software managing a computer's resources and all of the accompanying standard software tools, such as command-line interpreters, graphical user interfaces, file utilities, and editors.
- More narrowly, to refer to the central software that manages and allocates computer resources (i.e., the CPU, RAM, and devices).

The term *kernel* is often used as a synonym for the second meaning, and it is with this meaning of the term *operating system* that we are concerned in this book.

Although it is possible to run programs on a computer without a kernel, the presence of a kernel greatly simplifies the writing and use of other programs, and increases the power and flexibility available to programmers. The kernel does this by providing a software layer to manage the limited resources of a computer.

The Linux kernel executable typically resides at the pathname `/boot/vmlinuz`, or something similar. The derivation of this filename is historical. On early UNIX implementations, the kernel was called `unix`. Later UNIX implementations, which implemented virtual memory, renamed the kernel as `vmunix`. On Linux, the filename mirrors the system name, with the *z* replacing the final *x* to

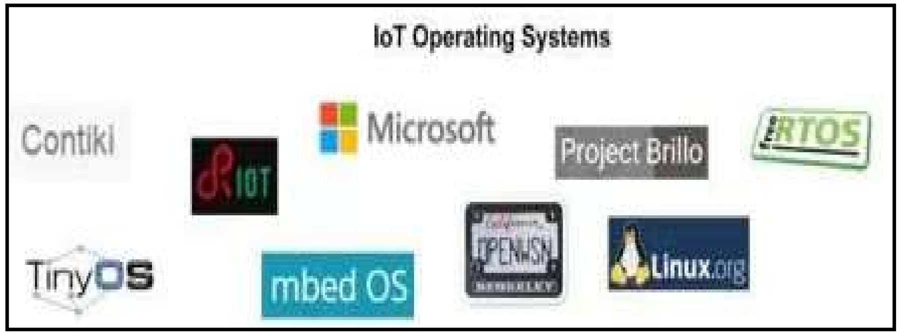
## Linux Introduction (1/7)

- Operating System
  - ✓ Definition: Resource Manager
  - ✓ Examples: Linux, Windows, OS X and so on.

Battle Royale
Microsoft Talisker [center] prepares to slam the Linux penguin, while fending off the FreeBSD devil [lower left]. Wind River Tornado [upper left] and Red Hat [lower right] join the fray.

(Source: IEEE Spectrum, 2001)

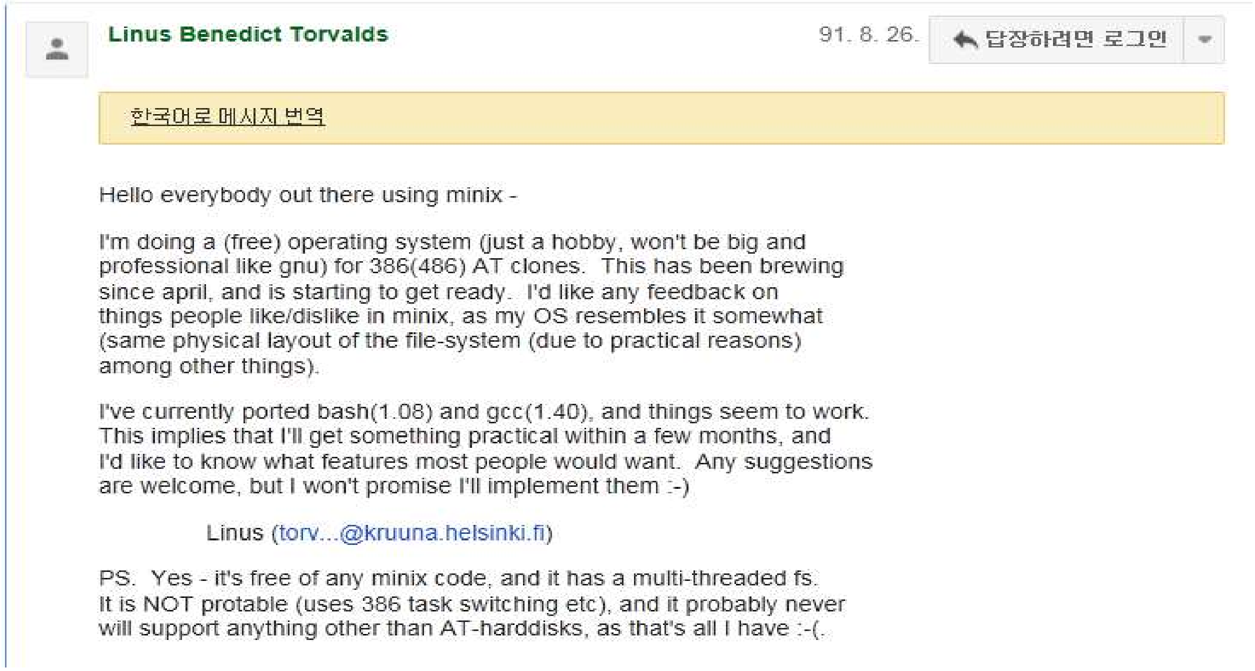
**(source: https://www.deviantart.com/nick-os/art/Os-war-choose-your-poison-110510677)**



**(source: https://maxhemingway.com/2015/10/21/iot-device-security-considerations-and-security-layers-operating-system/)**

# Linux Introduction (2/7)

- Linux Definition
  - ✓ Linux is a clone of the UNIX Operating System
  - ✓ Written from scratch by Linus B. Torvalds, with assistance from a loosely-knit team of Developers across the Network



Hello everybody out there using minix -

I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since april, and is starting to get ready. I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things).

I've currently ported bash(1.08) and gcc(1.40), and things seem to work. This implies that I'll get something practical within a few months, and I'd like to know what features most people would want. Any suggestions are welcome, but I won't promise I'll implement them :-)

Linus (torv...@kruuna.helsinki.fi)

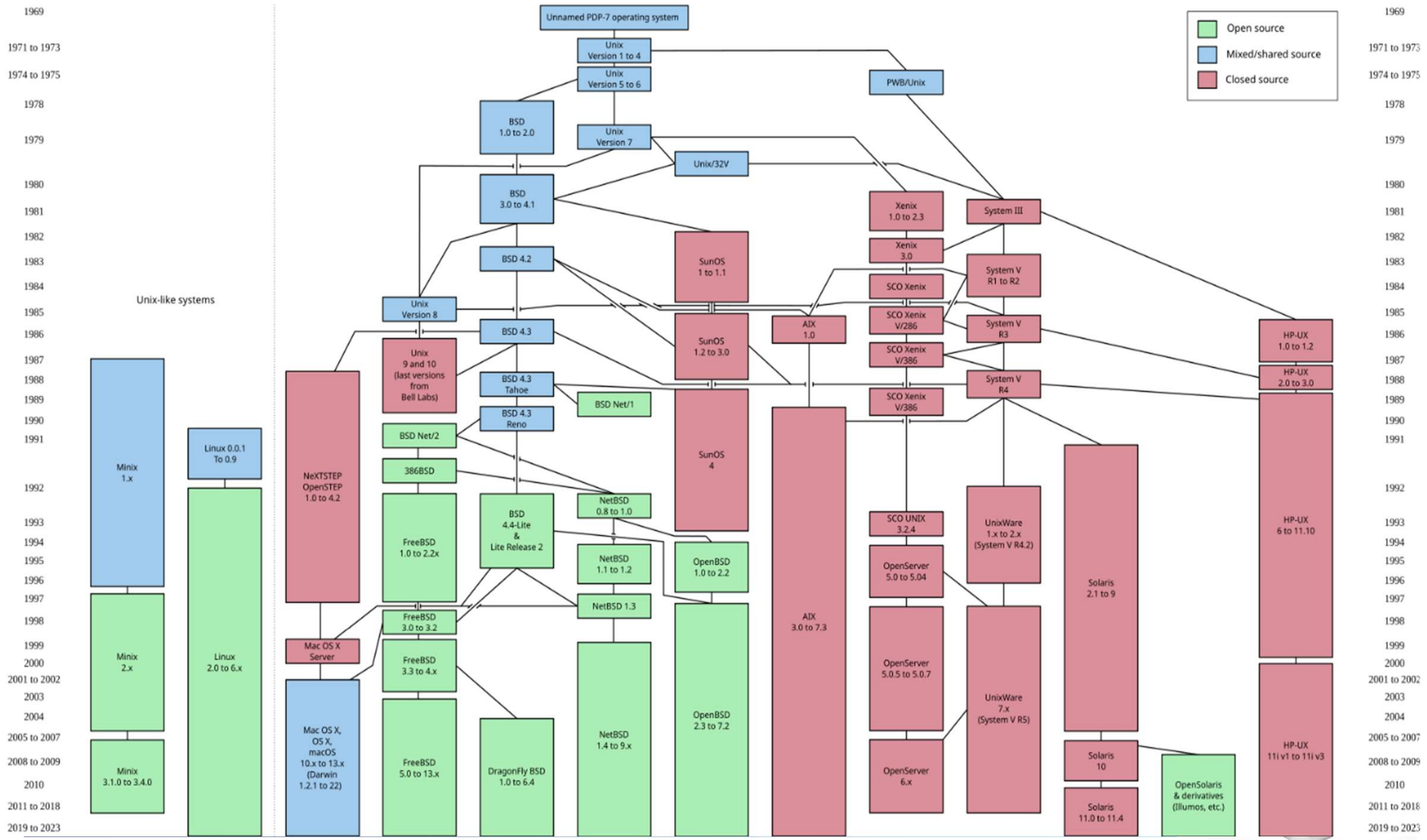
PS. Yes - it's free of any minix code, and it has a multi-threaded fs. It is NOT protable (uses 386 task switching etc), and it probably never will support anything other than AT-harddisks, as that's all I have :-(.

  - ✓ Univ. of Helsinki in Finland
  - ✓ May, 1991: Release 0.0.1 version
  - ✓ 2025: Release 6.14.1 (refer to https://www.kernel.org/)

# Linux Introduction (3/7)

- Unix-like OSes

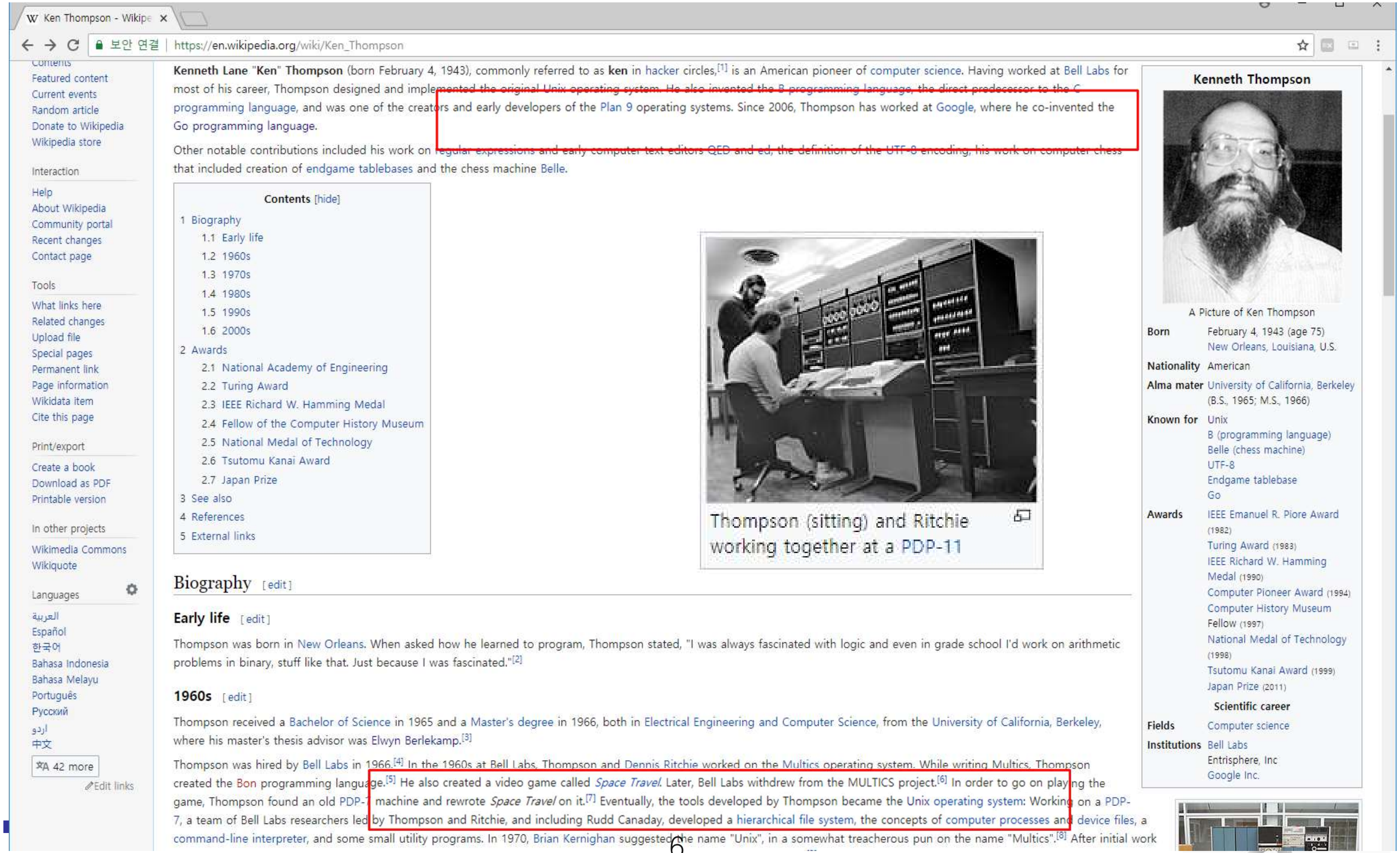
**(Source: wikipedia.org)**



☞ **POSIX (Portable Operating Systems Interface for UNIX)**

# Linux Introduction (4/7)

- Ken and Dennis



W Ken Thompson - Wikipe ×

보안 연결 | https://en.wikipedia.org/wiki/Ken_Thompson

Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Create a book
Download as PDF
Printable version

In other projects
Wikimedia Commons
Wikiquote

Languages
العربية
Español
한국어
Bahasa Indonesia
Bahasa Melayu
Português
Русский
اردو
中文
42 more
Edit links

**Kenneth Lane "Ken" Thompson** (born February 4, 1943), commonly referred to as **ken** in hacker circles,[1] is an American pioneer of computer science. Having worked at Bell Labs for most of his career, Thompson designed and implemented the original Unix operating system. He also invented the B programming language, the direct predecessor to the C programming language, and was one of the creators and early developers of the Plan 9 operating systems. Since 2006, Thompson has worked at Google, where he co-invented the Go programming language.

Other notable contributions included his work on regular expressions and early computer text editors QED and ed, the definition of the UTF-8 encoding, his work on computer chess that included creation of endgame tablebases and the chess machine Belle.

Contents [hide]

Thompson (sitting) and Ritchie working together at a PDP-11

**Kenneth Thompson**

A Picture of Ken Thompson

| | |
|---|---|
| **Born** | February 4, 1943 (age 75) New Orleans, Louisiana, U.S. |
| **Nationality** | American |
| **Alma mater** | University of California, Berkeley (B.S., 1965; M.S., 1966) |
| **Known for** | Unix<br>B (programming language)<br>Belle (chess machine)<br>UTF-8<br>Endgame tablebase<br>Go |
| **Awards** | IEEE Emanuel R. Piore Award (1982)<br>Turing Award (1983)<br>IEEE Richard W. Hamming Medal (1990)<br>Computer Pioneer Award (1994)<br>Computer History Museum Fellow (1997)<br>National Medal of Technology (1998)<br>Tsutomu Kanai Award (1999)<br>Japan Prize (2011) |
| | **Scientific career** |
| **Fields** | Computer science |
| **Institutions** | Bell Labs<br>Entrisphere, Inc<br>Google Inc. |

## Biography [edit]

### Early life [edit]

Thompson was born in New Orleans. When asked how he learned to program, Thompson stated, "I was always fascinated with logic and even in grade school I'd work on arithmetic problems in binary, stuff like that. Just because I was fascinated."[2]

### 1960s [edit]

Thompson received a Bachelor of Science in 1965 and a Master's degree in 1966, both in Electrical Engineering and Computer Science, from the University of California, Berkeley, where his master's thesis advisor was Elwyn Berlekamp.[3]

Thompson was hired by Bell Labs in 1966.[4] In the 1960s at Bell Labs, Thompson and Dennis Ritchie worked on the Multics operating system. While writing Multics, Thompson created the Bon programming language.[5] He also created a video game called *Space Travel*. Later, Bell Labs withdrew from the MULTICS project.[6] In order to go on playing the game, Thompson found an old PDP-7 machine and rewrote *Space Travel* on it.[7] Eventually, the tools developed by Thompson became the Unix operating system: Working on a PDP-7, a team of Bell Labs researchers led by Thompson and Ritchie, and including Rudd Canaday, developed a hierarchical file system, the concepts of computer processes and device files, a command-line interpreter, and some small utility programs. In 1970, Brian Kernighan suggested the name "Unix", in a somewhat treacherous pun on the name "Multics".[8] After initial work

# Linux Introduction (5/7)

- Contributors
  - ✓ GNU (www.gnu.org)
    - Richard M. Stallman (rms)
    - Free software
  - ✓ Minix
    - Andrew Tanenbaum
  - ✓ BSD
    - Bill Joy (cofounder of Sun Microsystems), FFS, TCP/IP, …
    - Linus Torvalds has said that if 386BSD had been available at the time, he probably would not have created Linux

GNU Operating System

Sponsored by the Free Software Foundation

What is GNU?

GNU is an operating system that is free software—that is, it respects users' freedom. The development of GNU made it possible to use a computer without software that would trample your freedom.

We recommend installable versions of GNU (more precisely, GNU/Linux distributions) which are entirely free software. More about GNU below.

What is free software?

The Free Software Definition

The free software definition presents the criteria for whether a particular software program qualifies as free software. From time to time we

What is the Free Software Movement?

Planet GNU

Friday Free Software Directory IRC meetup: August 28: Join the FSF and friends Friday, August 28, from 12pm to 3pm EDT (16:00 to 19:00 UTC) to help improve the Free Software Directory by addin... more

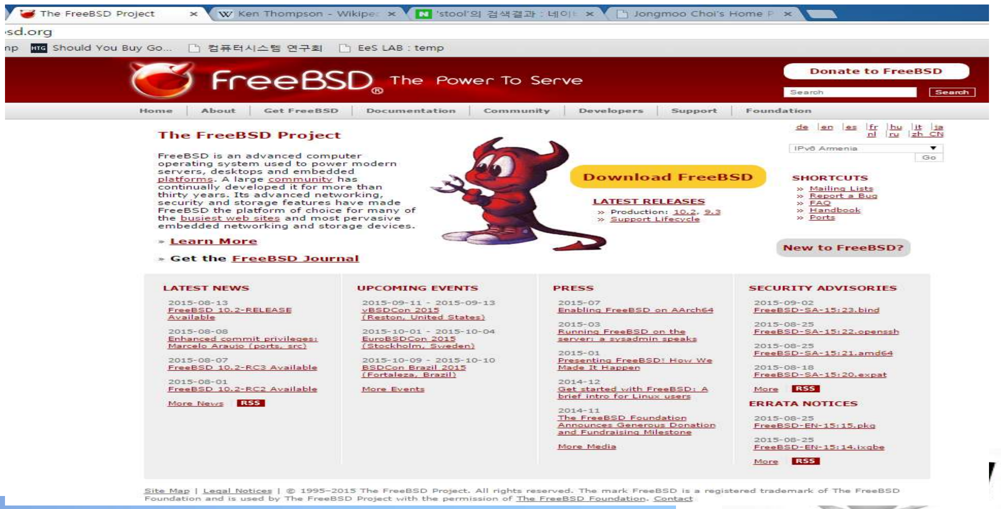
Richard Stallman - "Copyright vs. Community" (Gent, Belgium): Copyright developed in the age of the printing press, and was designed to fit with the system of centralized copying imp... more

Join us at IWEEE 2015 : Social Medicine: Dear GNU Health and Free Software community: In a month from now, GNU Solidario will celebrate it's 8th International Workshop on eHealth in Emerging E... more

For more news, see Planet GNU and the list of recent GNU releases.

Unifont

GNU Unifont is a bitmap font covering essentially all of Unicode's Basic Multilingual Plane. The package also includes utilities to ease adding new glyphs to the font. (doc)



The FreeBSD Project

FreeBSD is an advanced computer operating system used to power modern servers, desktops and embedded platforms. A large community has continually developed it for more than thirty years. Its advanced networking, security and storage features have made FreeBSD the platform of choice for many of the busiest web sites and most pervasive embedded networking and storage devices.

» Learn More

» Get the FreeBSD Journal

LATEST RELEASES
» Production: 10.2, 9.3
» Support Lifecycle

SHORTCUTS
» Mailing Lists
» Report a Bug
» FAQ
» Handbook
» Ports

LATEST NEWS
2015-08-13 FreeBSD 10.2-RELEASE Available
2015-08-08 Enhanced commit privileges: Marcelo Araujo (ports, src)
2015-08-07 FreeBSD 10.2-RC3 Available
2015-08-01 FreeBSD 10.2-RC2 Available
More News RSS

UPCOMING EVENTS
2015-09-11 - 2015-09-13 vBSDCon 2015 (Reston, United States)
2015-10-01 - 2015-10-04 EuroBSDCon 2015 (Stockholm, Sweden)
2015-10-09 - 2015-10-10 BSDCon Brazil 2015 (Fortaleza, Brazil)
More Events

PRESS
2015-07 Enabling FreeBSD on AArch64
2015-03 Running FreeBSD on the server: a sysadmin speaks
2015-01 Presenting FreeBSD! How We Made It Happen
2014-12 Get started with FreeBSD: A brief intro for Linux users
2014-11 The FreeBSD Foundation Announces Generous Donation and Fundraising Milestone
More Media

SECURITY ADVISORIES
2015-09-02 FreeBSD-SA-15:23.bind
2015-08-25 FreeBSD-SA-15:22.openssh
2015-08-25 FreeBSD-SA-15:21.amd64
2015-08-18 FreeBSD-SA-15:20.expat
More RSS

ERRATA NOTICES
2015-08-25 FreeBSD-EN-15:15.pkg
2015-08-25 FreeBSD-EN-15:14.ixgbe
More RSS

Site Map | Legal Notices | © 1995–2015 The FreeBSD Project. All rights reserved. The mark FreeBSD is a registered trademark of The FreeBSD Foundation and is used by The FreeBSD Project with the permission of The FreeBSD Foundation. Contact

## Linux Introduction (6/7)

- Applications

(Source: images at google)

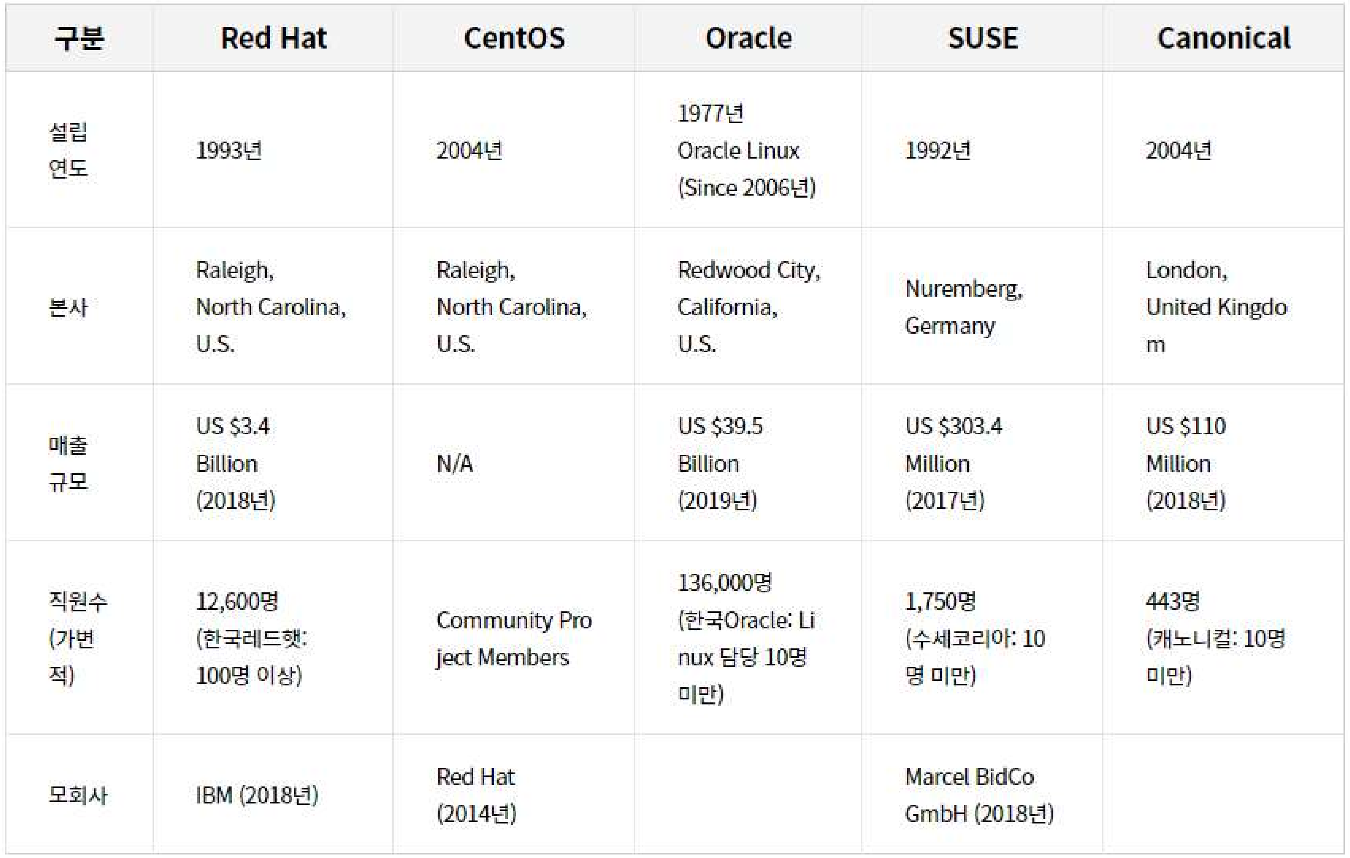
## Linux Introduction (7/7)

- Some notes about UNIX and Linux (From LPI Chapter 1)
  - ✓ Linux is a member of the UNIX family
  - ✓ History
    - 1969~ : UNIX Invented by Ken and Dennis, UNIX 1~7 edition at AT&T
    - 1975~ : popularly used at universities include Berkeley, MIT and CMU.
    - 1979~ : BSD and new features (FFS, TCP/IP, C shell, ...)
    - 1981~ : System III and System V from AT&T
    - 1985~ : UNIX golden ages (IBM, HP, Sun, NeXTStep, SCO, ...) ➔ UNIX War
    - 1990~ : Standardization (POSIX, FIPS, X/Open, SUS (Single UNIX Spec.)
    - 2021: Three representative OSes + Vendor proprietary OSes + New OSes

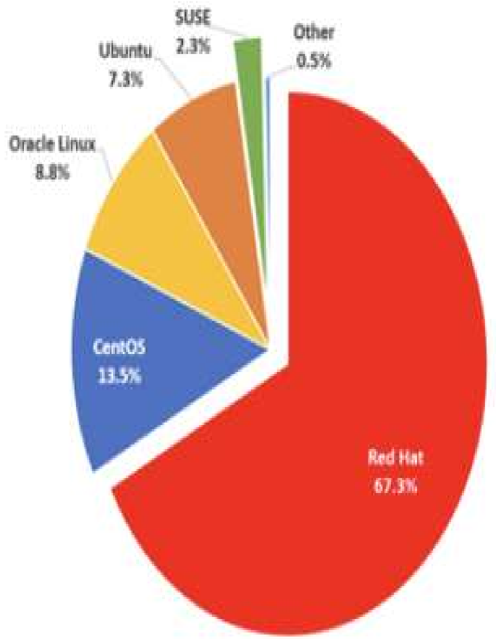
    - 1984~ : GNU by R. Stallman (gcc, Emacs, bash, …), GPL (General Public License)
    - 1991~ : Linux by L. Torvalds, Minix + Intel optimization, GNU incorporation
    - 2025: Linux kernel version 6.14.1
  - ✓ Linux version number
    - x.y.z: Major.Minor.Revision
    - Even minor: stable, odd minor: development (but NOT strict today)

## Aside (Optional)

- Linux vs. Distribution
  - ✓ Linux: Kernel
  - ✓ Distribution: Kernel + Packages + Frameworks + …



| 구분 | Red Hat | CentOS | Oracle | SUSE | Canonical |
|---|---|---|---|---|---|
| 설립 연도 | 1993년 | 2004년 | 1977년 Oracle Linux (Since 2006년) | 1992년 | 2004년 |
| 본사 | Raleigh, North Carolina, U.S. | Raleigh, North Carolina, U.S. | Redwood City, California, U.S. | Nuremberg, Germany | London, United Kingdo m |
| 매출 규모 | US $3.4 Billion (2018년) | N/A | US $39.5 Billion (2019년) | US $303.4 Million (2017년) | US $110 Million (2018년) |
| 직원수 (가변 적) | 12,600명 (한국레드햇: 100명 이상) | Community Pro ject Members | 136,000명 (한국Oracle: Li nux 담당 10명 미만) | 1,750명 (수세코리아: 10 명 미만) | 443명 (캐노니컬: 10명 미만) |
| 모회사 | IBM (2018년) | Red Hat (2014년) | | Marcel BidCo GmbH (2018년) | |

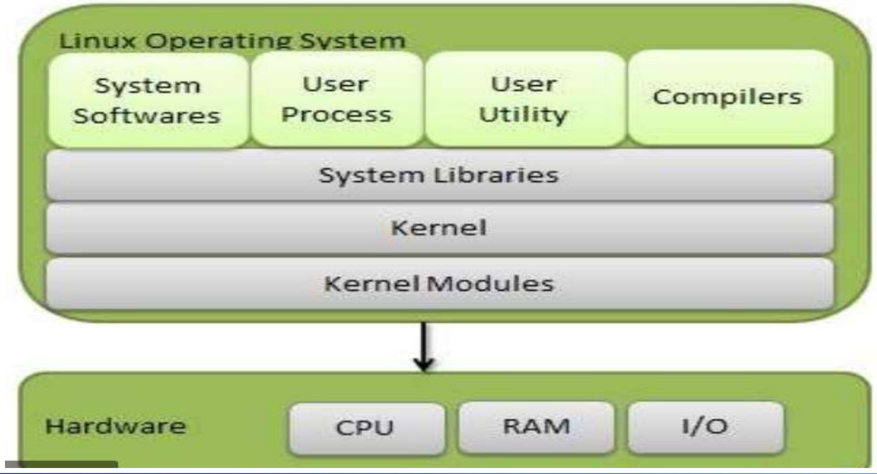


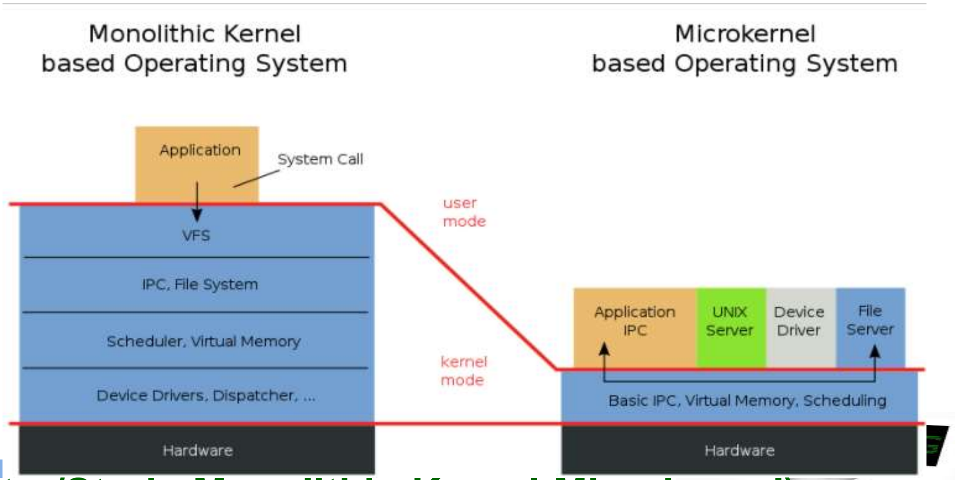
[그림 3] 'A' 기업 내 Linux 배포본 사용 현황

**(Source: https://www.samsungsds.com/kr/insights/linux_distribution.html)**

# Fundamental Concepts of Linux (1/7)

- From LPI Chapter 2
- 2.1 The Core of Operating System: kernel
  - ✓ OS: Computing environments vs. Kernel: Central part of OS
    - OS = Kernel + Other System Programs (GUI, Shell, GCC, Packages, …)
    - Kernel's role: 1) Process mgmt., 2) VM, 3) FS, 4) Device access, 5) Networking, 6) system call, 7) multi-user support
    - Kernel module: dynamic loadable SW runs in kernel mode
  - ✓ User mode vs kernel mode (also called as supervisor mode)
    - To protect kernel from applications
    - Monolithic kernel vs. Microkernel (u-kernel)
  - ✓ System: process's viewpoint vs. kernel's viewpoint

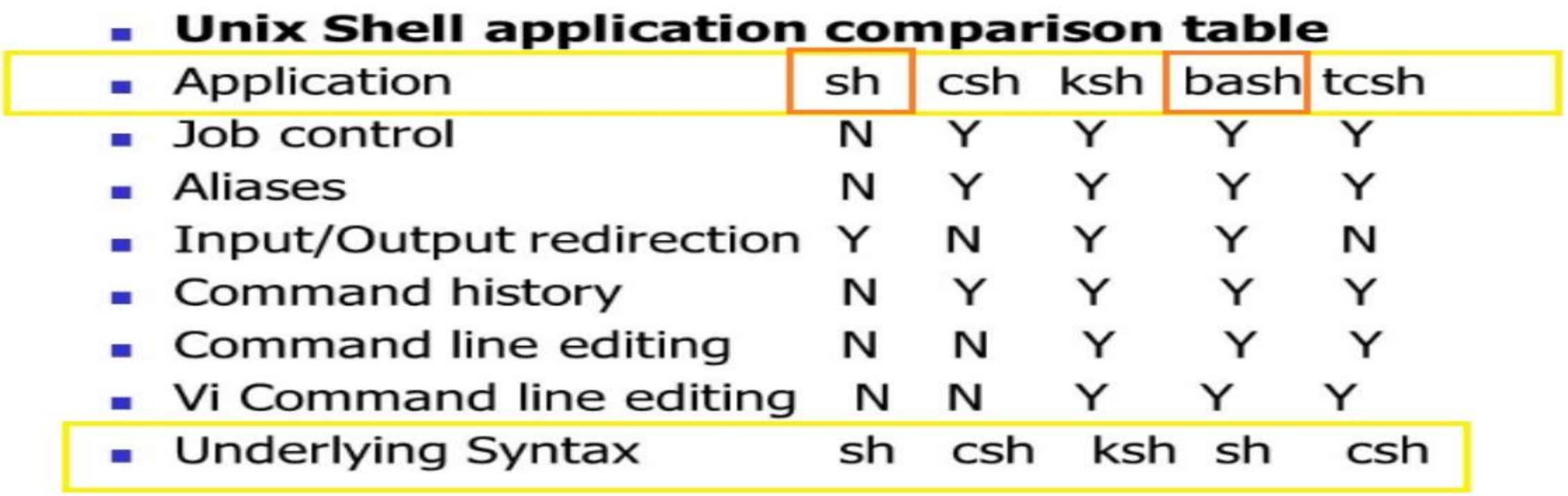



Monolithic Kernel based Operating System
Microkernel based Operating System
Application
System Call
user mode
VFS
IPC, File System
Scheduler, Virtual Memory
Device Drivers, Dispatcher, ...
kernel mode
Hardware
Application IPC
UNIX Server
Device Driver
File Server
Basic IPC, Virtual Memory, Scheduling
Hardware

(Source: https://talkingaboutme.tistory.com/entry/Study-Monolithic-Kernel-Microkernel)

# Fundamental Concepts of Linux (2/7)

- 2.2 The shell
  - ✓ Special-purpose program designed to read commands typed by a user and execute them ➔ command interpreter
  - ✓ Examples: Bourne shell (Bell Lab.), C shell (BSD), Korn Shell (AT&T), bash (GNU)
- 2.3 Users and Groups
  - ✓ 3 categories: user, group, others
  - ✓ Superuser: has special privileges (User ID: 0, login name: root)



**Unix Shell application comparison table**

| Application | sh | csh | ksh | bash | tcsh |
|---|---|---|---|---|---|
| Job control | N | Y | Y | Y | Y |
| Aliases | N | Y | Y | Y | Y |
| Input/Output redirection | Y | N | Y | Y | N |
| Command history | N | Y | Y | Y | Y |
| Command line editing | N | N | Y | Y | Y |
| Vi Command line editing | N | N | Y | Y | Y |
| Underlying Syntax | sh | csh | ksh | sh | csh |

(Source: https://stackoverflow.com/questions/5725296/difference-between-sh-and-bash)

# Fundamental Concepts of Linux (3/7)

- 2.4 Directory and Links
  - ✓ file types: regular, directory, link, device, ... (almost everything is file)
  - ✓ directory: a set of related file, support hierarchical structure
  - ✓ Home directory, root directory, current directory
- 2.5 File I/O Model
  - ✓ stdio library: fopen(), fread(), fwrite(), fclose(), printf(), scanf(), …
  - ✓ system call: open(), read(), write(), close(), … ➔ LN3
  - ✓ After open(): file name → file descriptor

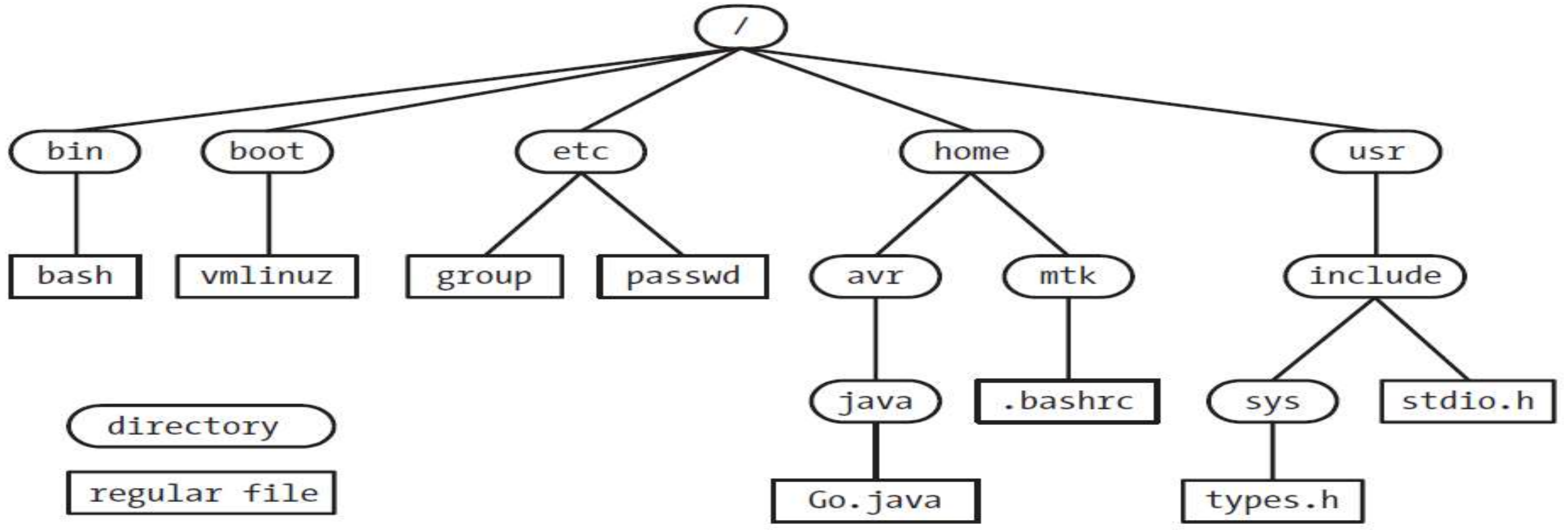
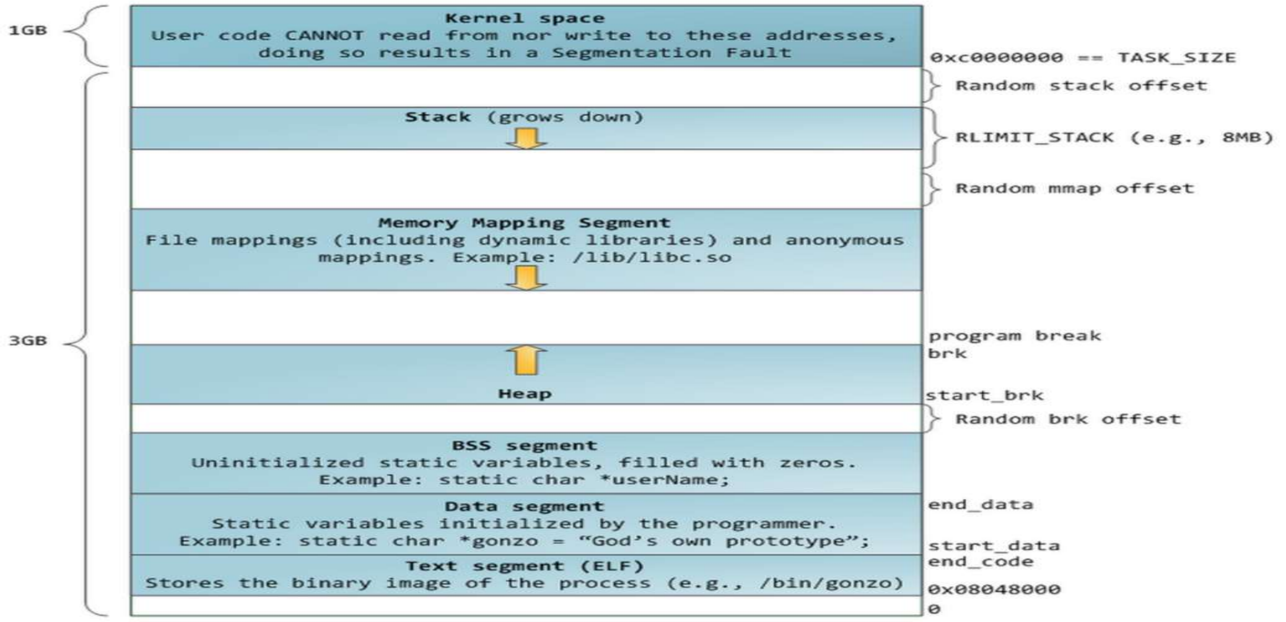


Figure 2-1: Subset of the Linux single directory hierarchy

## Fundamental Concepts of Linux (4/7)

- 2.6 Programs
  - ✓ A set of instructions that describes how to perform a specific task
  - ✓ Two forms: source code, binary (machine language)
- 2.7 Processes
  - ✓ An instance of an executing program ➔ LN4, 5
  - ✓ Has its own virtual memory (layout: text, data, heap, stack, map)
- 2.8 Memory Mappings
  - ✓ mmap(): maps a file into the calling process's virtual memory
  - ✓ Access file using a pointer instead of open()/read()/write()

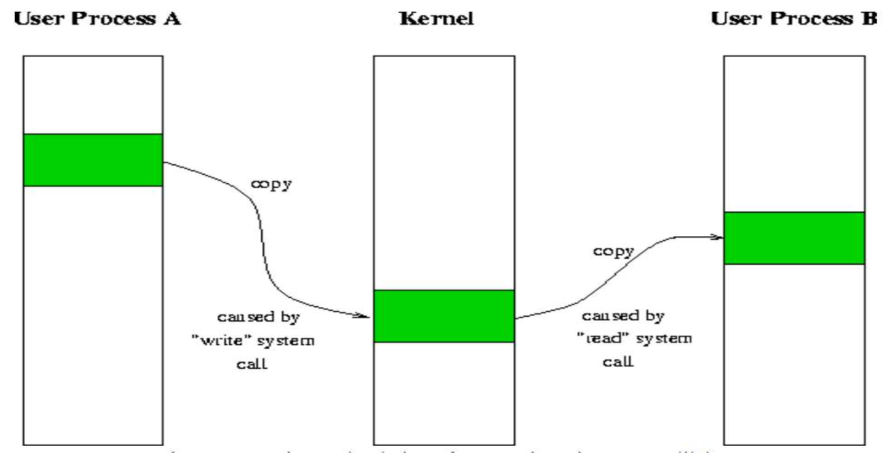


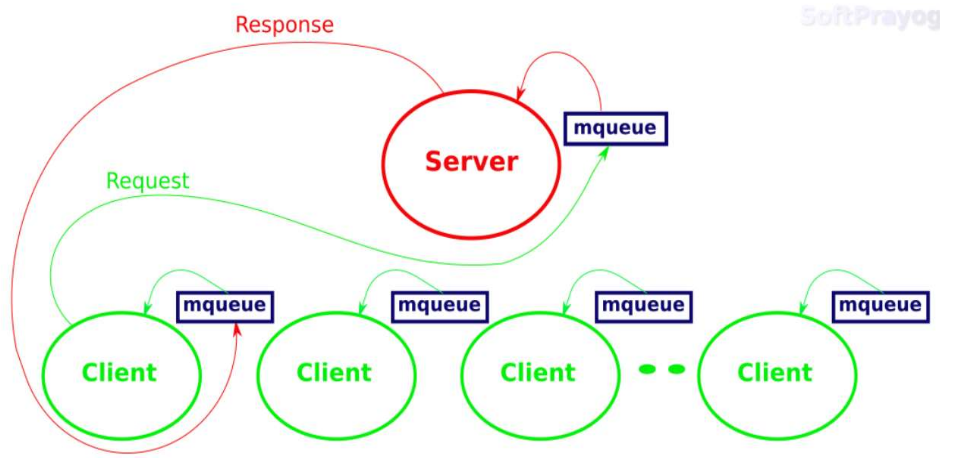
1GB
3GB
Kernel space
User code CANNOT read from nor write to these addresses, doing so results in a Segmentation Fault
0xc0000000 == TASK_SIZE
Random stack offset
Stack (grows down)
RLIMIT_STACK (e.g., 8MB)
Random mmap offset
Memory Mapping Segment
File mappings (including dynamic libraries) and anonymous mappings. Example: /lib/libc.so
program break
brk
Heap
start_brk
Random brk offset
BSS segment
Uninitialized static variables, filled with zeros. Example: static char *userName;
end_data
Data segment
Static variables initialized by the programmer. Example: static char *gonzo = "God's own prototype";
start_data
end_code
Text segment (ELF)
Stores the binary image of the process (e.g., /bin/gonzo)
0x08048000
0

(Source: brunch.co.kr/@alden/13)

## Fundamental Concepts of Linux (5/7)

- 2.9 Static and Shared Libraries
  - ✓ Compiled objects (relocatable and logically related)
  - ✓ Static libraries (also called as archive): compile-time linking
    - extracts copies of the required object modules from the library and copies these into an executable file
  - ✓ Shared libraries: run-time linking
    - instead of copying object modules from library into executable, just write a record, which allows shared libraries to be linked on-demand
- 2.10 IPC and Synchronization
  - ✓ Inter Process Communication and Process orchestration
  - ✓ Examples: signal, pipe, socket, message queue, shared memory, semaphore, …





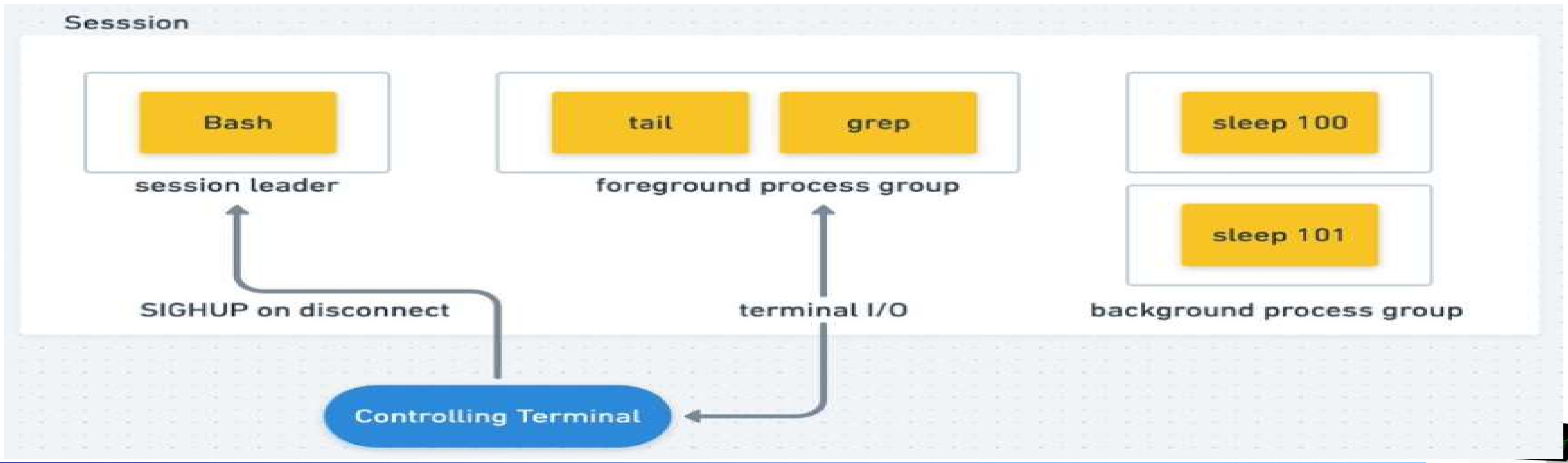
(Source: http://www.gerhardmueller.de/docs/UnixCommunicationFacilities/ip/node6.html, https://www.softprayog.in/programming/interprocess-communication-using-system-v-message-queues-in-linux)

# Fundamental Concepts of Linux (6/7)

- 2.11 Signal
  - ✓ User-level interrupt: inform to a process (^C)
  - ✓ c.f.) Interrupt: a mechanism to inform an event to kernel
- 2.12 Thread
  - ✓ A flow control in a process (threads share virtual memory) ➔ LN5
- 2.13 Job control (Process group)
  - ✓ Allows the user to simultaneously execute and manipulate multiple commands or pipelines.

```
$ ls -l | sort -k5n | less
```

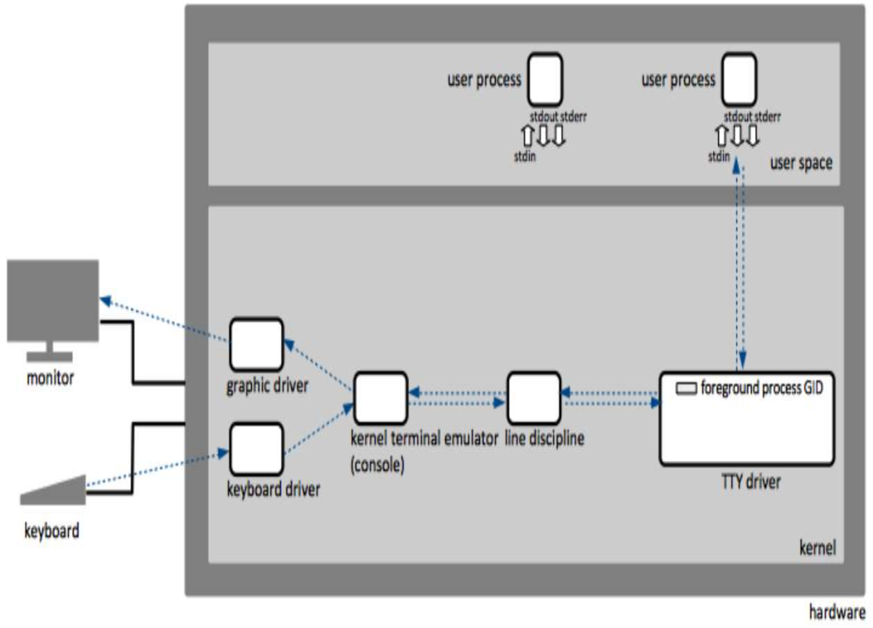
- 2.14 Session
  - ✓ A session is a collection of process groups (jobs).
  - ✓ Related with a terminal (controlling terminal, usually login terminal)
    - One foreground job and multiple background jobs

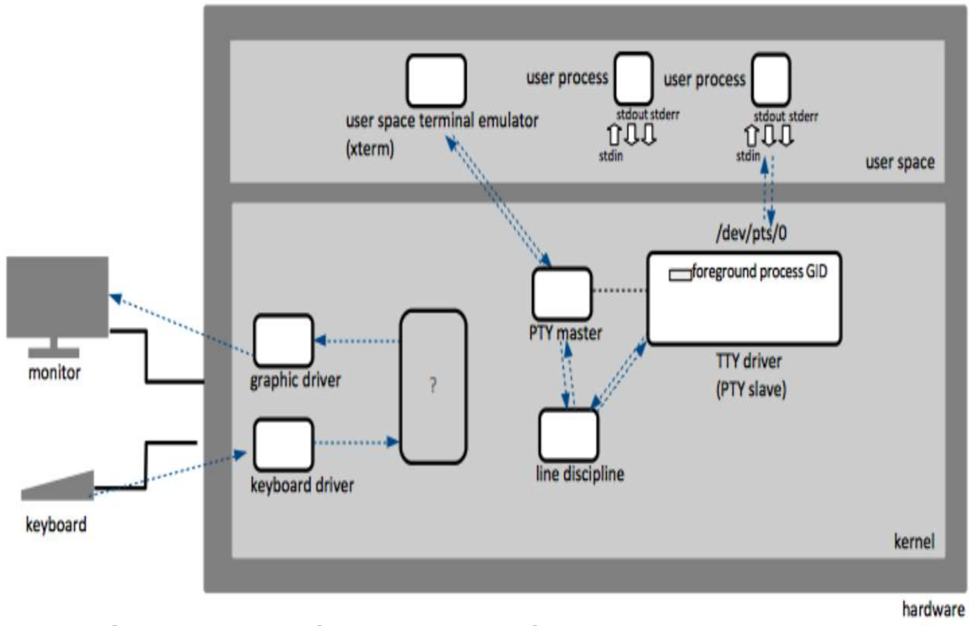


(Source: https://twitter.com/igor_sarcevic/status/1157349076809191425)

## Fundamental Concepts of Linux (7/7)

- 2.15 Pseudo-terminal
  - ✓ Connected virtual devices (e.g. terminal emulator)
- 2.16 Date and time
  - ✓ Real time (also called as epoch time): Since 1st January, 1970.
  - ✓ Process time (also called as CPU time)
    - Total amount of CPU time that a process has used since starting
    - system CPU time, user CPU time
- Others
  - ✓ Client-Server architecture, Realtime, /proc file system

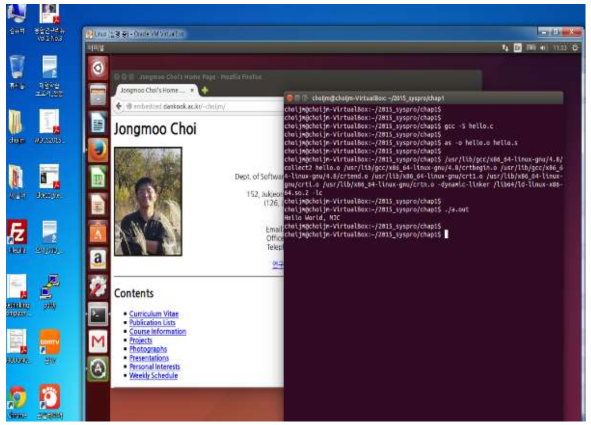




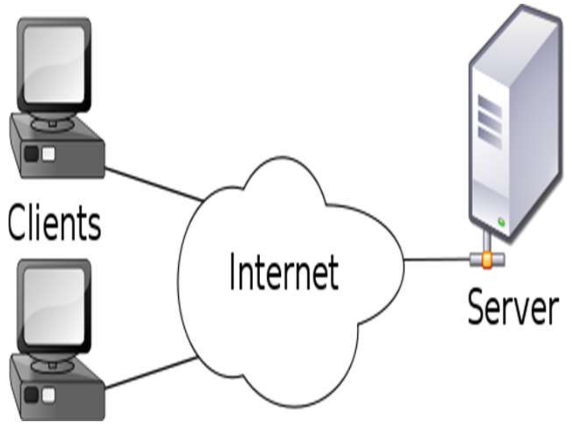
(Source: https://kb.novaordis.com/index.php/Linux_TTY)

## How to access Linux (1/4)

- 1) Standalone (usually with multi-boot)
- 2) Virtualization (or WSL)
- 3) Client-Server



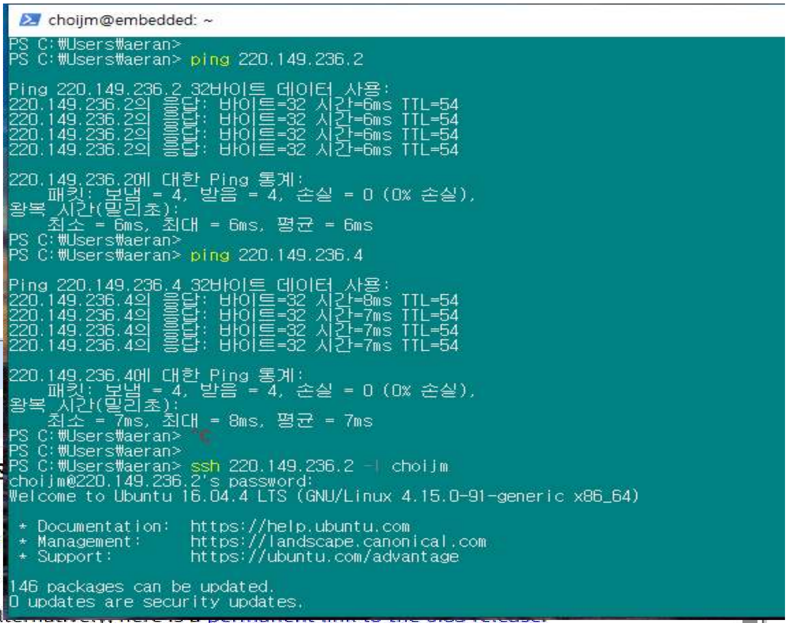
```
choijm@choijm-VirtualBox:~/2015_syspro/chap1$
choijm@choijm-VirtualBox:~/2015_syspro/chap1$
choijm@choijm-VirtualBox:~/2015_syspro/chap1$ gcc -S hello.c
choijm@choijm-VirtualBox:~/2015_syspro/chap1$
choijm@choijm-VirtualBox:~/2015_syspro/chap1$ as -o hello.o hello.s
choijm@choijm-VirtualBox:~/2015_syspro/chap1$
choijm@choijm-VirtualBox:~/2015_syspro/chap1$ /usr/lib/gcc/x86_64-linux-gnu/4.8/
collect2 hello.o /usr/lib/gcc/x86_64-linux-gnu/4.8/crtbegin.o /usr/lib/gcc/x86_6
4-linux-gnu/4.8/crtend.o /usr/lib/x86_64-linux-gnu/crt1.o /usr/lib/x86_64-linux-
gnu/crti.o /usr/lib/x86_64-linux-gnu/crtn.o -dynamic-linker /lib64/ld-linux-x86-
64.so.2 -lc
choijm@choijm-VirtualBox:~/2015_syspro/chap1$
choijm@choijm-VirtualBox:~/2015_syspro/chap1$ ./a.out
Hello World, MJC
choijm@choijm-VirtualBox:~/2015_syspro/chap1$
choijm@choijm-VirtualBox:~/2015_syspro/chap1$
```



- ✓ In our course
  - Client: terminal emulator (telnet/ssh client, putty, …)
  - Server: Linux system (PC)
    - IP: 220.149.236.2 (primary), 220.149.236.4 (secondary)
  - Alternative: Amazon EC2, Google Cloud, MS Azure or Solid Cloud

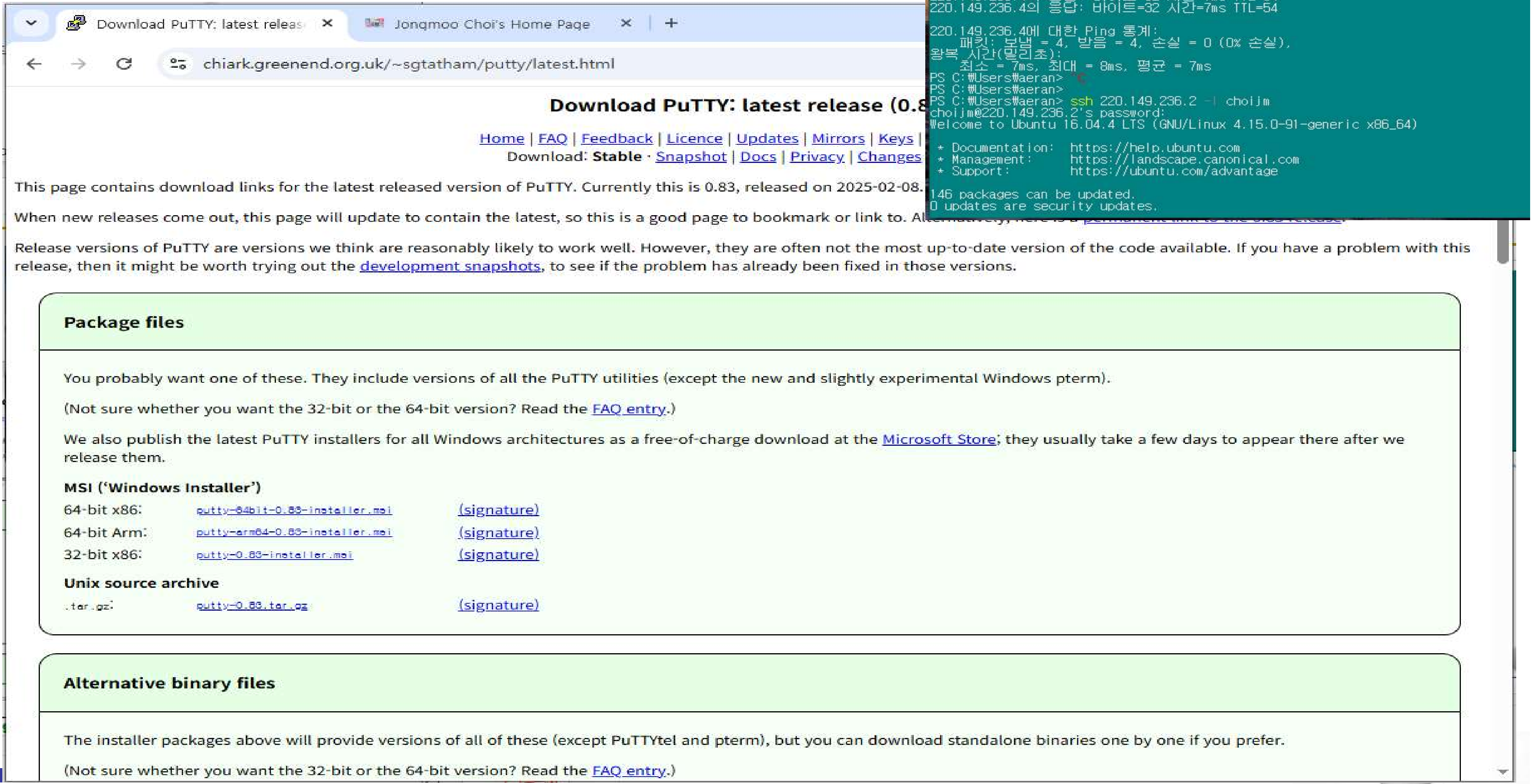
# How to access Linux (2/4)

- Client
  - ✓ telnet, ssh, ping, …
  - ✓ putty, SecureCRT, powershell, …



choijm@embedded: ~

```
PS C:￦Users￦aeran>
PS C:￦Users￦aeran> ping 220.149.236.2

Ping 220.149.236.2 32바이트 데이터 사용:
220.149.236.2의 응답: 바이트=32 시간=6ms TTL=54
220.149.236.2의 응답: 바이트=32 시간=6ms TTL=54
220.149.236.2의 응답: 바이트=32 시간=6ms TTL=54
220.149.236.2의 응답: 바이트=32 시간=6ms TTL=54

220.149.236.2에 대한 Ping 통계:
    패킷: 보냄 = 4, 받음 = 4, 손실 = 0 (0% 손실),
왕복 시간(밀리초):
    최소 = 6ms, 최대 = 6ms, 평균 = 6ms
PS C:￦Users￦aeran>
PS C:￦Users￦aeran> ping 220.149.236.4

Ping 220.149.236.4 32바이트 데이터 사용:
220.149.236.4의 응답: 바이트=32 시간=8ms TTL=54
220.149.236.4의 응답: 바이트=32 시간=7ms TTL=54
220.149.236.4의 응답: 바이트=32 시간=7ms TTL=54
220.149.236.4의 응답: 바이트=32 시간=7ms TTL=54

220.149.236.4에 대한 Ping 통계:
    패킷: 보냄 = 4, 받음 = 4, 손실 = 0 (0% 손실),
왕복 시간(밀리초):
    최소 = 7ms, 최대 = 8ms, 평균 = 7ms
PS C:￦Users￦aeran> ^C
PS C:￦Users￦aeran>
PS C:￦Users￦aeran> ssh 220.149.236.2 -l choijm
choijm@220.149.236.2's password:
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.15.0-91-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

146 packages can be updated.
0 updates are security updates.
```


Download PuTTY: latest releas
Jongmoo Choi's Home Page
chiark.greenend.org.uk/~sgtatham/putty/latest.html

**Download PuTTY: latest release (0.8**

Home | FAQ | Feedback | Licence | Updates | Mirrors | Keys |
Download: **Stable** · Snapshot | Docs | Privacy | Changes

This page contains download links for the latest released version of PuTTY. Currently this is 0.83, released on 2025-02-08.

When new releases come out, this page will update to contain the latest, so this is a good page to bookmark or link to. Al

Release versions of PuTTY are versions we think are reasonably likely to work well. However, they are often not the most up-to-date version of the code available. If you have a problem with this release, then it might be worth trying out the development snapshots, to see if the problem has already been fixed in those versions.

**Package files**

You probably want one of these. They include versions of all the PuTTY utilities (except the new and slightly experimental Windows pterm).

(Not sure whether you want the 32-bit or the 64-bit version? Read the FAQ entry.)

We also publish the latest PuTTY installers for all Windows architectures as a free-of-charge download at the Microsoft Store; they usually take a few days to appear there after we release them.

**MSI ('Windows Installer')**

| | | |
|---|---|---|
| 64-bit x86: | putty-64bit-0.83-installer.msi | (signature) |
| 64-bit Arm: | putty-arm64-0.83-installer.msi | (signature) |
| 32-bit x86: | putty-0.83-installer.msi | (signature) |

**Unix source archive**

| | | |
|---|---|---|
| .tar.gz: | putty-0.83.tar.gz | (signature) |

**Alternative binary files**

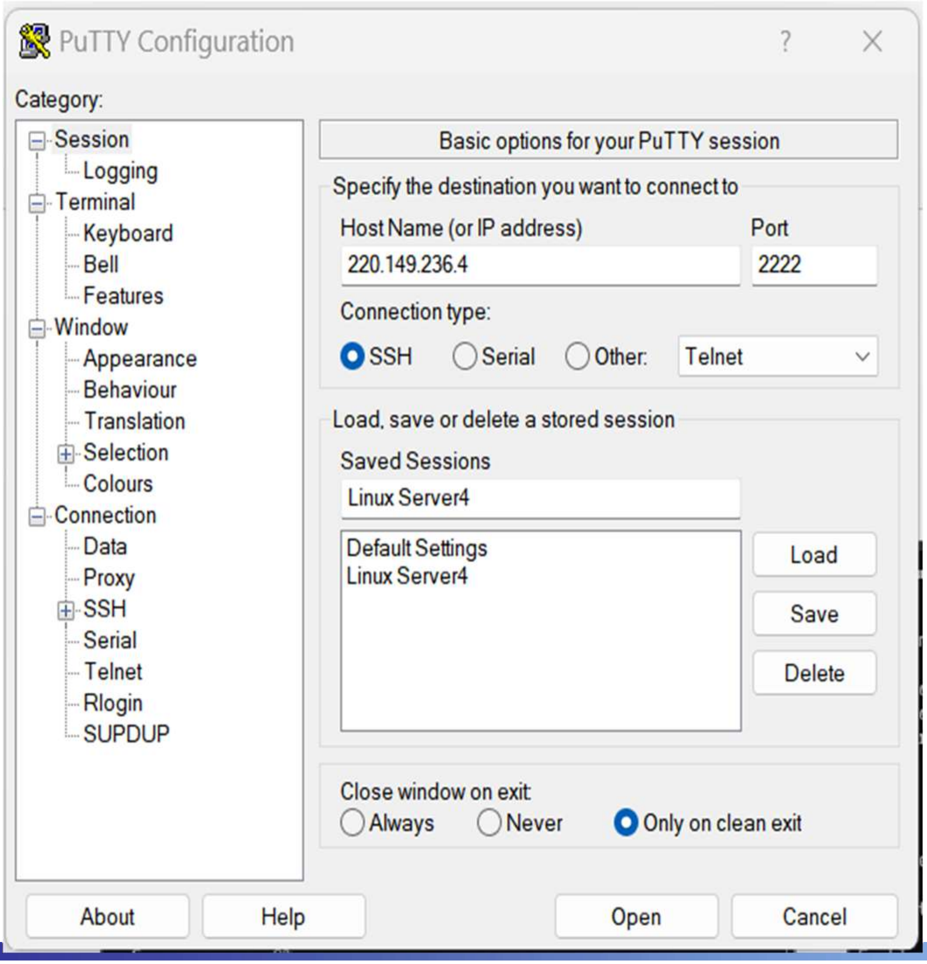
The installer packages above will provide versions of all of these (except PuTTYtel and pterm), but you can download standalone binaries one by one if you prefer.

(Not sure whether you want the 32-bit or the 64-bit version? Read the FAQ entry.)

**(Source: https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html)**

# How to access Linux (3/4)

- Putty with ssh
  - ✓ IP: 220.149.236.2 (check that “type is ssh” and “port is 22” or “2222”)
  - ✓ Colours: click “Use system colours
  - ✓ Translation: choose “UTF-8”



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): 220.149.236.4
Port: 2222

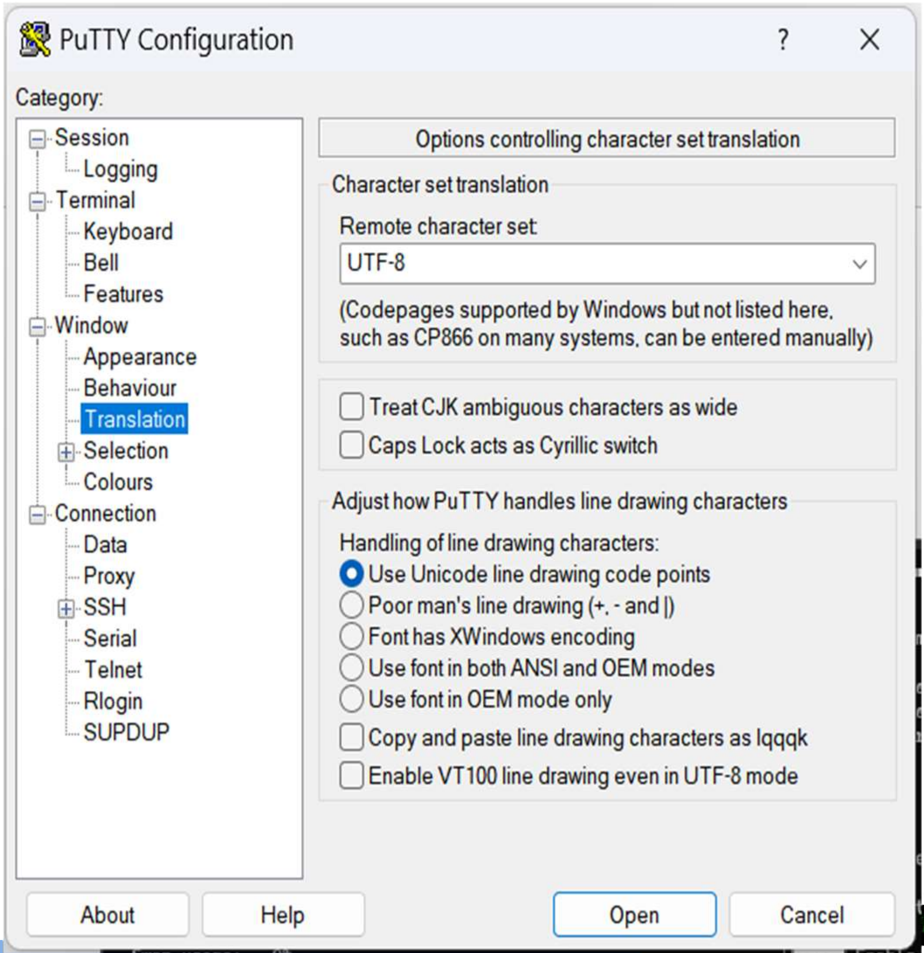
Connection type: [✓ SSH] Serial Other: Telnet

Load, save or delete a stored session

Saved Sessions: Linux Server4

Default Settings
Linux Server4

Close window on exit: Always Never [✓ Only on clean exit]



Options controlling character set translation

Character set translation

Remote character set: UTF-8

(Codepages supported by Windows but not listed here, such as CP866 on many systems, can be entered manually)

- [ ] Treat CJK ambiguous characters as wide
- [ ] Caps Lock acts as Cyrillic switch

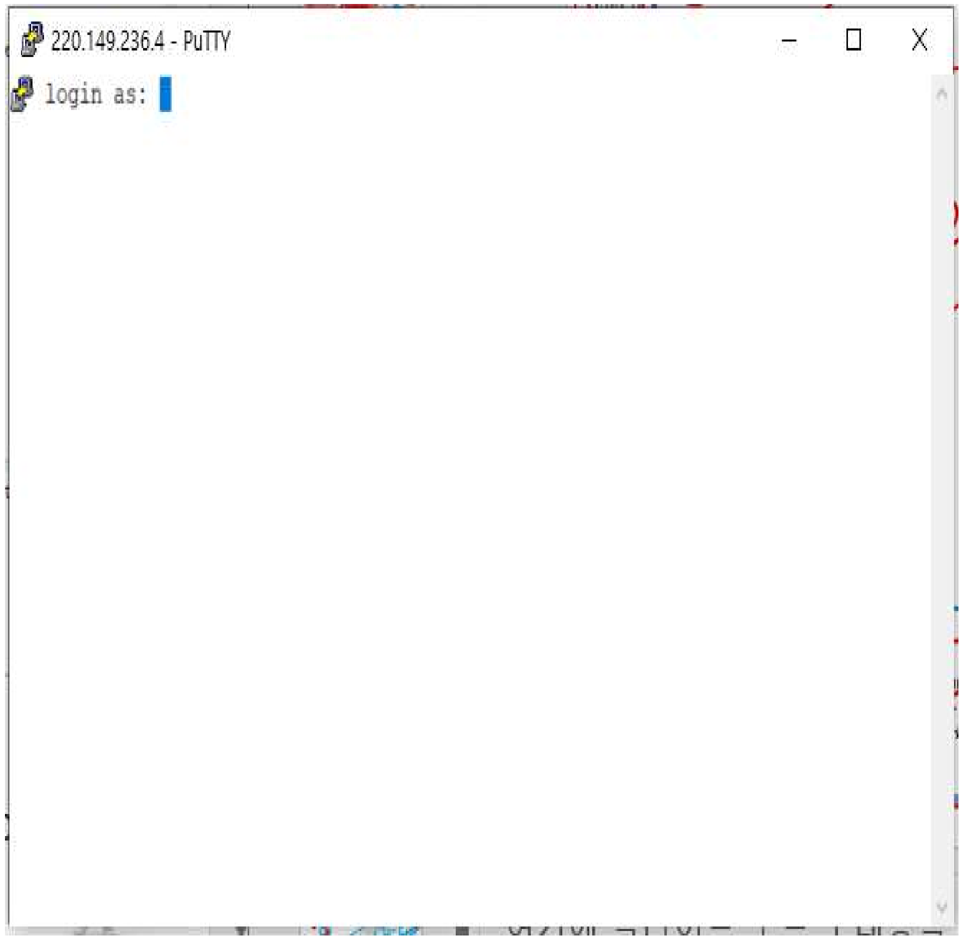
Adjust how PuTTY handles line drawing characters

Handling of line drawing characters:
- [✓ Use Unicode line drawing code points]
- Poor man's line drawing (+, - and |)
- Font has XWindows encoding
- Use font in both ANSI and OEM modes
- Use font in OEM mode only

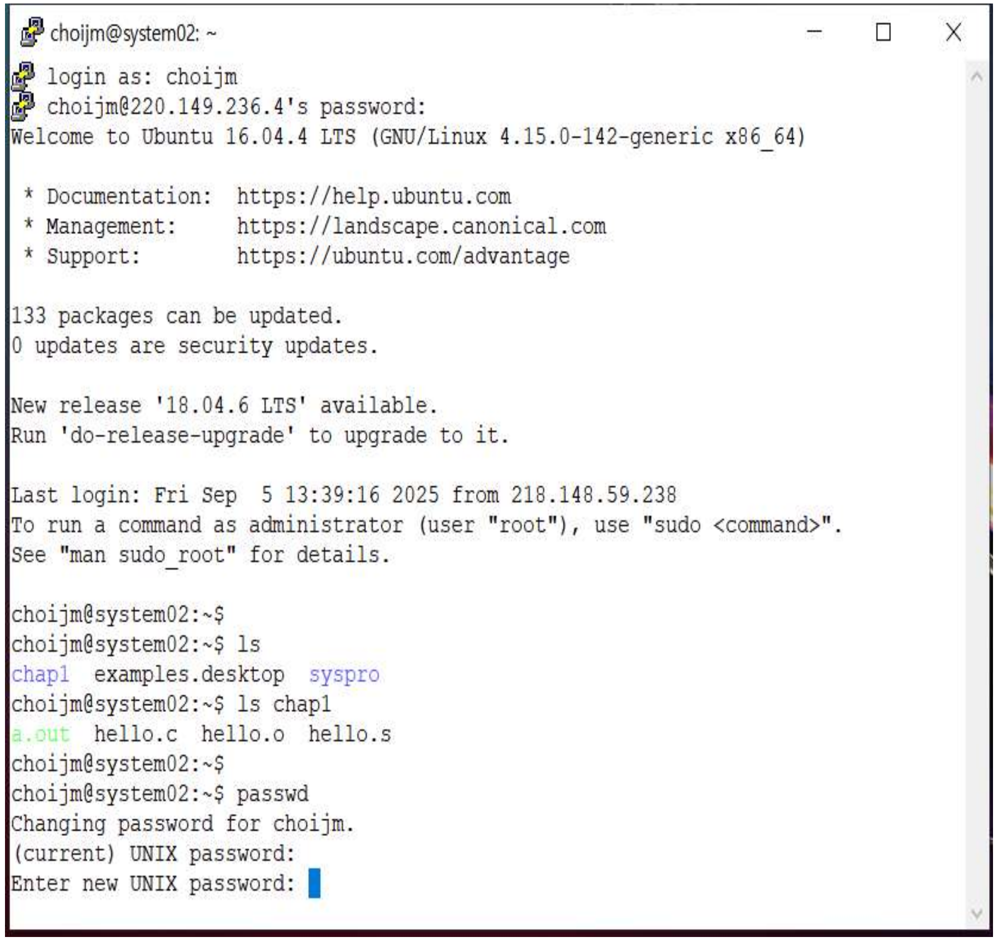
- [ ] Copy and paste line drawing characters as lqqqk
- [ ] Enable VT100 line drawing even in UTF-8 mode

## How to access Linux (4/4)

- Login and shell



```
login as:
```



```
login as: choijm
choijm@220.149.236.4's password:
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.15.0-142-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

133 packages can be updated.
0 updates are security updates.

New release '18.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Fri Sep  5 13:39:16 2025 from 218.148.59.238
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

choijm@system02:~$
choijm@system02:~$ ls
chap1  examples.desktop  syspro
choijm@system02:~$ ls chap1
a.out  hello.c  hello.o  hello.s
choijm@system02:~$
choijm@system02:~$ passwd
Changing password for choijm.
(current) UNIX password:
Enter new UNIX password:
```

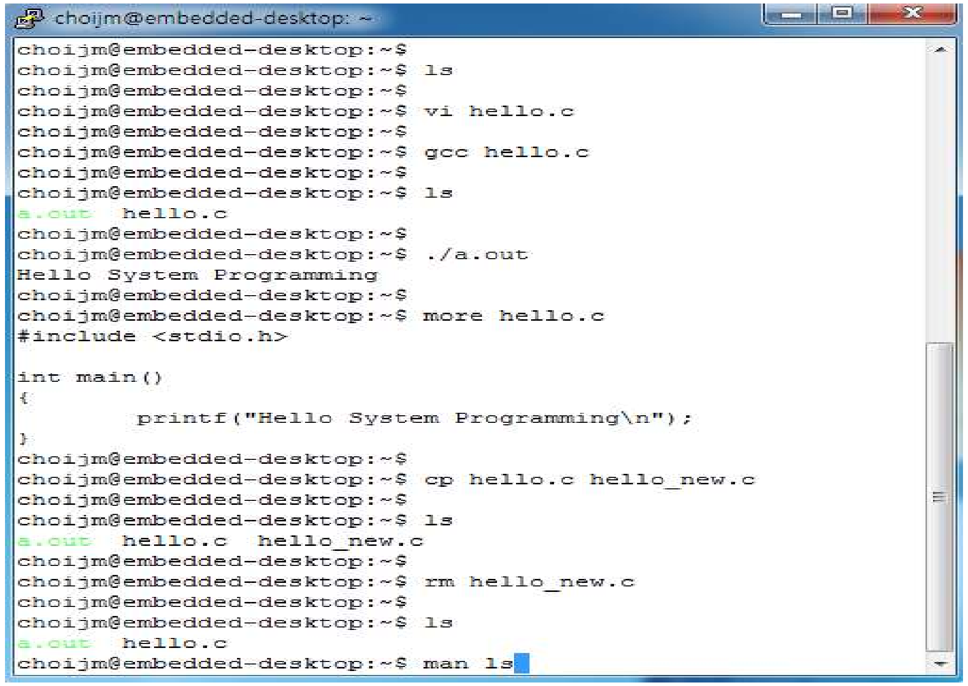
- ✓ ID: sys학번 (8 numbers of Student ID)
- ✓ Default passwd: sys****** (change using the “passwd” command)

# How to use commands in Linux (1/12)

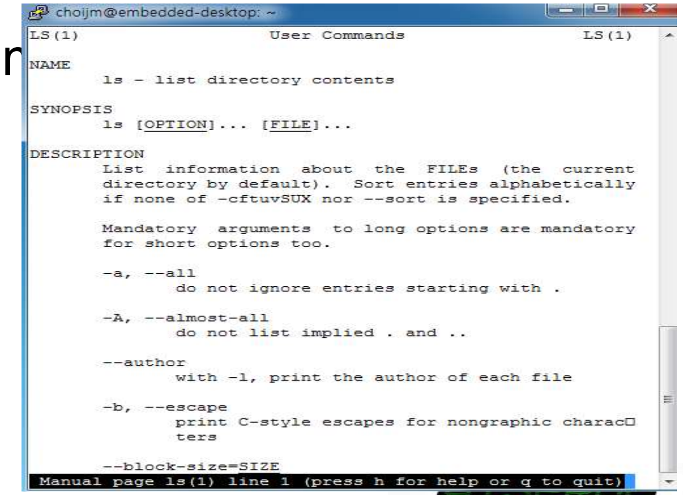
- UNIX
  - ✓ Two key objects in UNIX: file as a “place” and process (task) as a “life” (by M. Bach, The Design of the UNIX Operating Systems)
- File
  - ✓ Array of bytes, stream of character (attributes: start, size, current offset)
  - ✓ Associated with disk blocks
  - ✓ Supports a variety of objects using file concept (eg. device, network, memory, and even process)
- Process (Task)
  - ✓ Program in execution
  - ✓ Associated with CPUs (Scheduling entity)
  - ✓ Having context such as memory space and CPU registers

## How to use commands in Linux (2/12)

- file related command
  - ✓ create
    - vi, gcc, mknod, …
  - ✓ copy/move
    - cp, mv, ln, …
  - ✓ delete
    - rm
  - ✓ listing
    - ls
  - ✓ file content view
    - cat, more, less, head, tail, objdump, hexdur
  - ✓ file attributes manipulation
    - chmod, chown, chgrp, touch
  - ✓ redirection
    - >



```
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ls
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ vi hello.c
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ gcc hello.c
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ls
a.out  hello.c
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ./a.out
Hello System Programming
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ more hello.c
#include <stdio.h>

int main()
{
        printf("Hello System Programming\n");
}
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ cp hello.c hello_new.c
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ls
a.out  hello.c  hello_new.c
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ rm hello_new.c
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ls
a.out  hello.c
choijm@embedded-desktop:~$ man ls
```

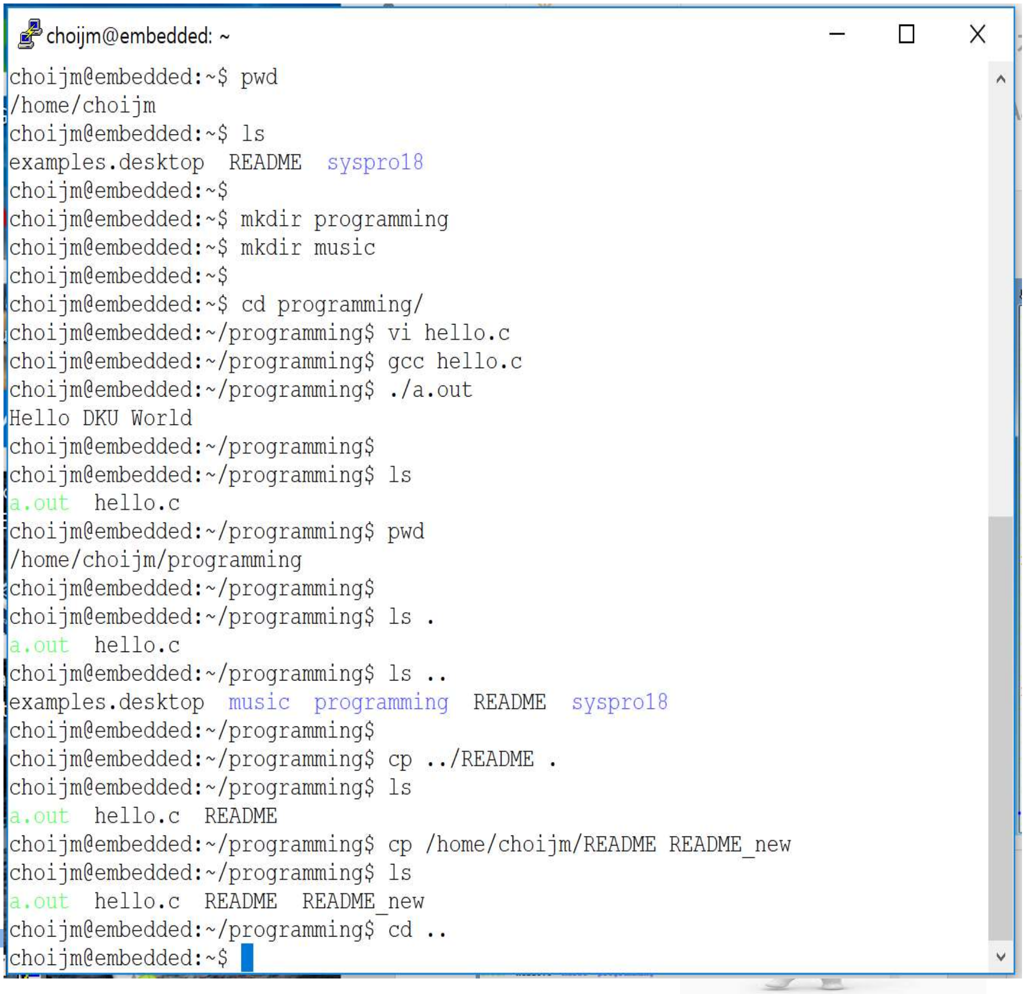


```
LS(1)                      User Commands                      LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about  the  FILEs  (the  current
       directory by default).  Sort entries alphabetically
       if none of -cftuvSUX nor --sort is specified.

       Mandatory  arguments  to long options are mandatory
       for short options too.

       -a, --all
              do not ignore entries starting with .

       -A, --almost-all
              do not list implied . and ..

       --author
              with -l, print the author of each file

       -b, --escape
              print C-style escapes for nongraphic charac□
              ters

       --block-size=SIZE
 Manual page ls(1) line 1 (press h for help or q to quit)
```

## How to use commands in Linux (3/12)

- directory
  - ✓ a set of files
  - ✓ provide hierarchical structure of files
  - ✓ home directory, root directory, current directory
  - ✓ relative path, absolute path

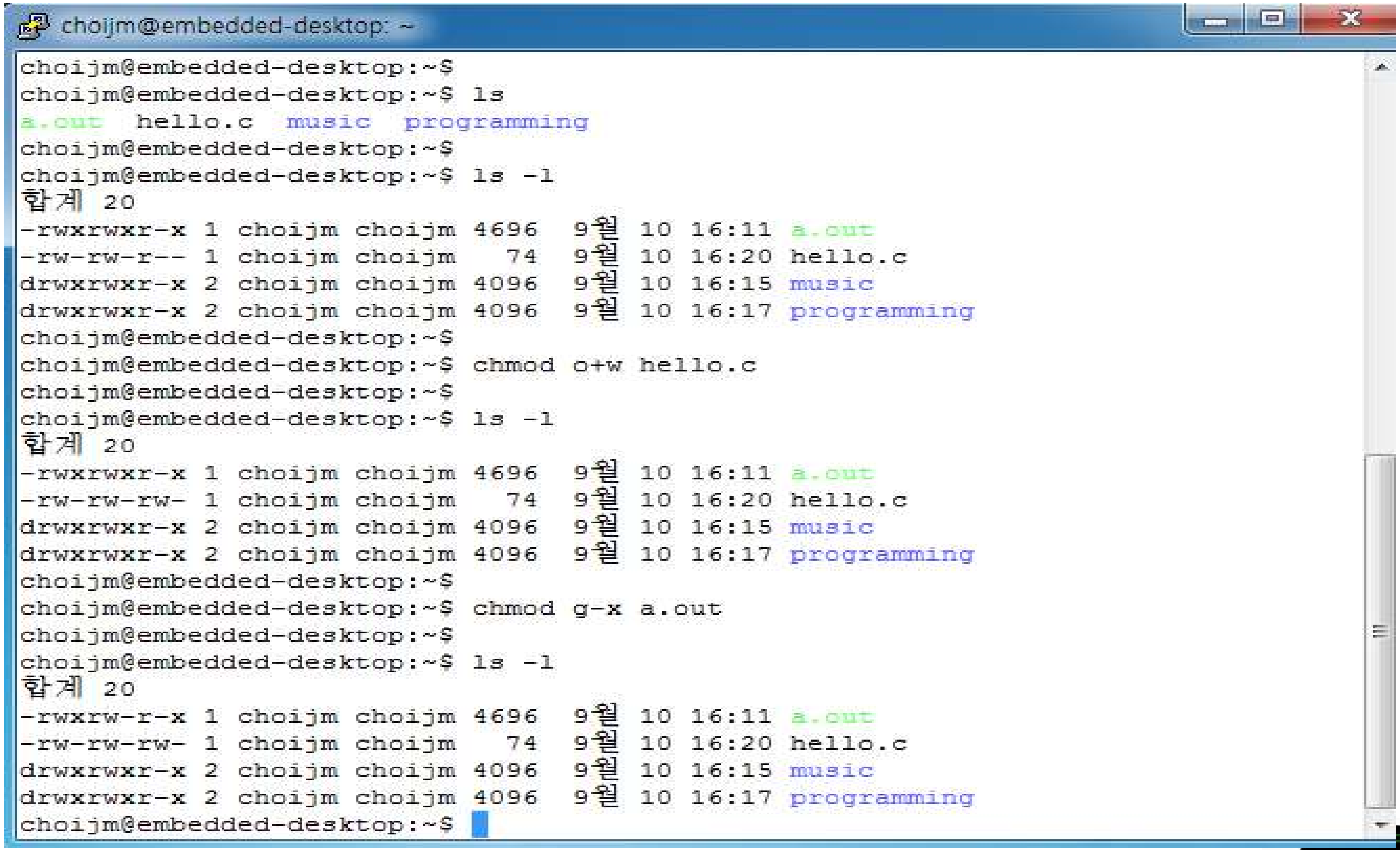
- directory related command
  - ✓ create
    - mkdir
  - ✓ change
    - cd
  - ✓ delete
    - rmdir
  - ✓ current position
    - pwd



```
choijm@embedded:~$ pwd
/home/choijm
choijm@embedded:~$ ls
examples.desktop  README  syspro18
choijm@embedded:~$
choijm@embedded:~$ mkdir programming
choijm@embedded:~$ mkdir music
choijm@embedded:~$
choijm@embedded:~$ cd programming/
choijm@embedded:~/programming$ vi hello.c
choijm@embedded:~/programming$ gcc hello.c
choijm@embedded:~/programming$ ./a.out
Hello DKU World
choijm@embedded:~/programming$
choijm@embedded:~/programming$ ls
a.out  hello.c
choijm@embedded:~/programming$ pwd
/home/choijm/programming
choijm@embedded:~/programming$
choijm@embedded:~/programming$ ls .
a.out  hello.c
choijm@embedded:~/programming$ ls ..
examples.desktop  music  programming  README  syspro18
choijm@embedded:~/programming$
choijm@embedded:~/programming$ cp ../README .
choijm@embedded:~/programming$ ls
a.out  hello.c  README
choijm@embedded:~/programming$ cp /home/choijm/README README_new
choijm@embedded:~/programming$ ls
a.out  hello.c  README  README_new
choijm@embedded:~/programming$ cd ..
choijm@embedded:~$
```

## How to use commands in Linux (4/12)

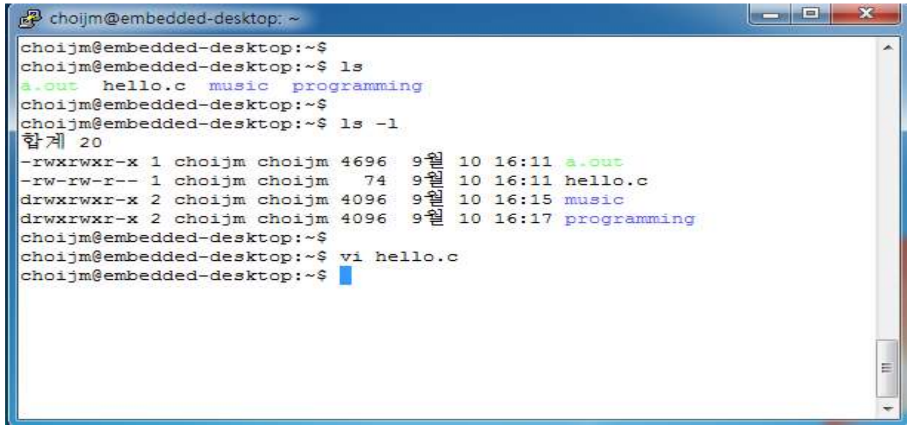
- file attribute manipulation
  - ✓ Permission and owner
  - ✓ cf. Command format: 1) command, 2) option, 3) argument



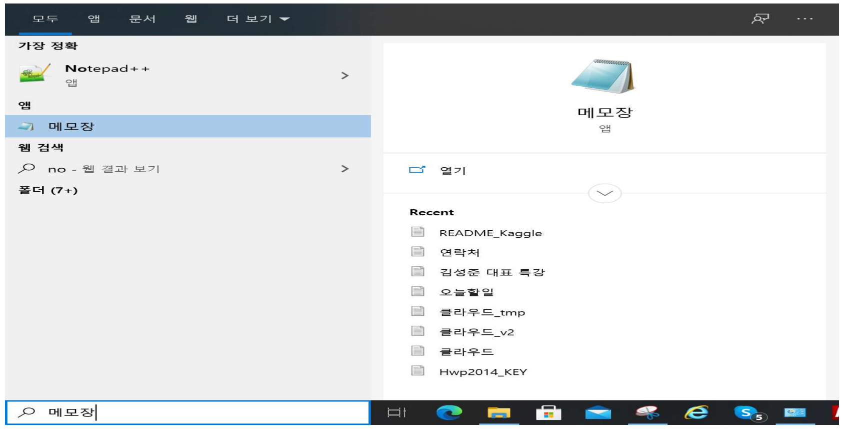
```
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ls
a.out   hello.c   music   programming
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ls -l
합계 20
-rwxrwxr-x 1 choijm choijm 4696  9월 10 16:11 a.out
-rw-rw-r-- 1 choijm choijm   74  9월 10 16:20 hello.c
drwxrwxr-x 2 choijm choijm 4096  9월 10 16:15 music
drwxrwxr-x 2 choijm choijm 4096  9월 10 16:17 programming
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ chmod o+w hello.c
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ls -l
합계 20
-rwxrwxr-x 1 choijm choijm 4696  9월 10 16:11 a.out
-rw-rw-rw- 1 choijm choijm   74  9월 10 16:20 hello.c
drwxrwxr-x 2 choijm choijm 4096  9월 10 16:15 music
drwxrwxr-x 2 choijm choijm 4096  9월 10 16:17 programming
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ chmod g-x a.out
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ls -l
합계 20
-rwxrw-r-x 1 choijm choijm 4696  9월 10 16:11 a.out
-rw-rw-rw- 1 choijm choijm   74  9월 10 16:20 hello.c
drwxrwxr-x 2 choijm choijm 4096  9월 10 16:15 music
drwxrwxr-x 2 choijm choijm 4096  9월 10 16:17 programming
choijm@embedded-desktop:~$
```

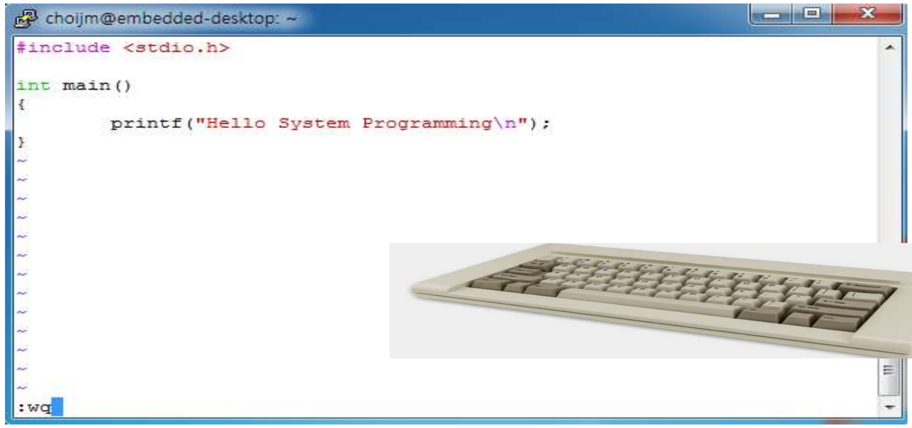
# How to use commands in Linux (5/12)

- vi editor (vim)
  - ✓ What are the differences between vi and notepad (or VS code)
    - Explicit input mode vs. Instant editable
    - No “파일” or “편집” button ➔ need line command mode

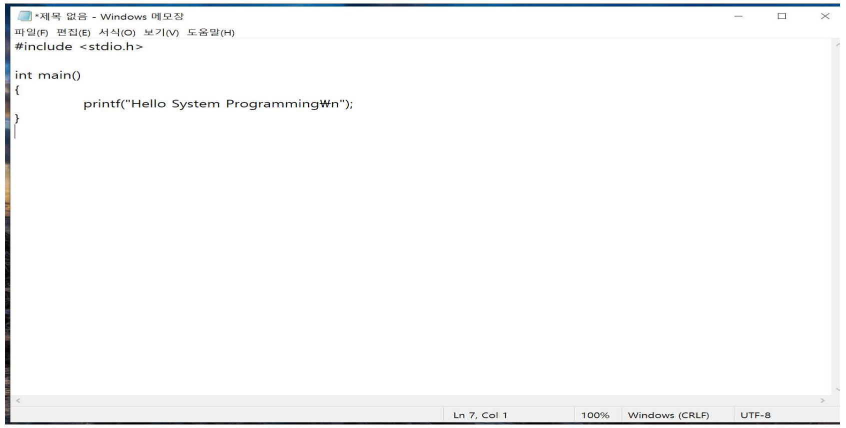


```
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ls
a.out   hello.c  music  programming
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ls -l
합계 20
-rwxrwxr-x 1 choijm choijm 4696  9월  10 16:11 a.out
-rw-rw-r-- 1 choijm choijm   74  9월  10 16:11 hello.c
drwxrwxr-x 2 choijm choijm 4096  9월  10 16:15 music
drwxrwxr-x 2 choijm choijm 4096  9월  10 16:17 programming
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ vi hello.c
choijm@embedded-desktop:~$
```





```
#include <stdio.h>

int main()
{
        printf("Hello System Programming\n");
}
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq
```


*제목 없음 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

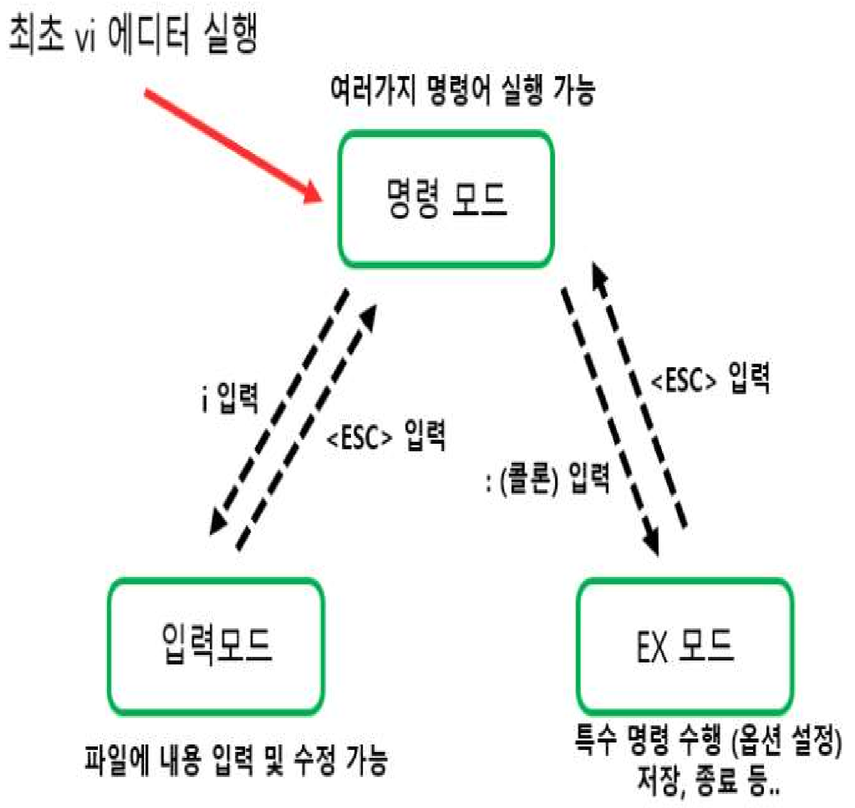
```
#include <stdio.h>

int main()
{
        printf("Hello System Programming₩n");
}
```

Ln 7, Col 1
100%
Windows (CRLF)
UTF-8

**(vi vs. notepad)**

# How to use commands in Linux (6/12)

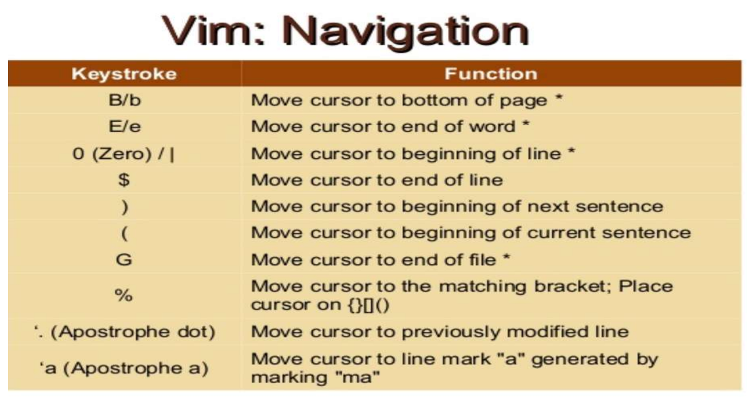
- vi editor (vim)
  - ✓ 3 modes
    - command/input/line command(a.k.a. execution mode)
  - ✓ At first (just before loading vi): command mode
  - ✓ Switch to the input mode
    - a (append), i (insert), o, r, …
  - ✓ Switch to the command mode
    - ESC
  - ✓ Switch to the line command mode
    - : at command mode
  - ✓ Switch to the command mode
    - Enter or ESC



(Source: https://dololak.tistory.com/379)

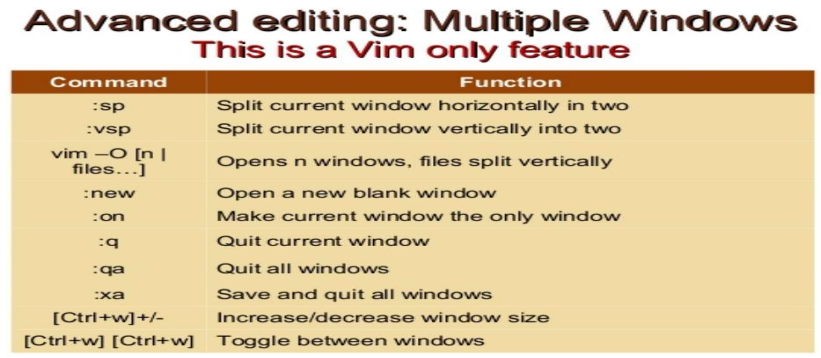
# How to use commands in Linux (7/12)

- vi editor (vim)
  - ✓ Actions allowed at the command/line command mode
    - Navigation (cursor movement): up/down, begin/end of word/line, …
    - File management: save, quit (e.g. :wq or :q), open, …
    - Editing: delete, change, substitute, transpose, …
    - Multiple windows, files, shell interaction, …



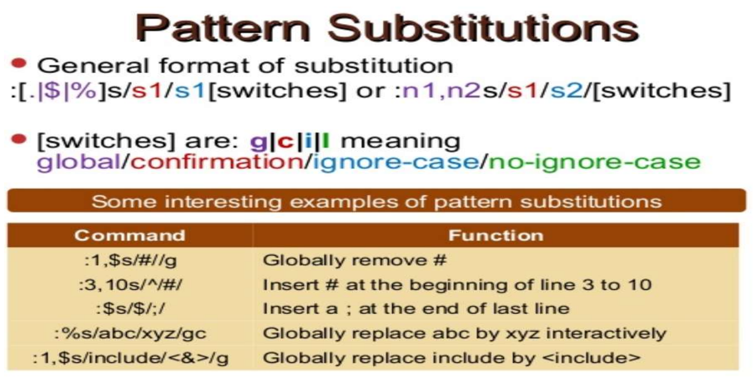
### Vim: Navigation

| Keystroke | Function |
|---|---|
| B/b | Move cursor to bottom of page * |
| E/e | Move cursor to end of word * |
| 0 (Zero) / \| | Move cursor to beginning of line * |
| $ | Move cursor to end of line |
| ) | Move cursor to beginning of next sentence |
| ( | Move cursor to beginning of current sentence |
| G | Move cursor to end of file * |
| % | Move cursor to the matching bracket; Place cursor on {}[]() |
| '. (Apostrophe dot) | Move cursor to previously modified line |
| 'a (Apostrophe a) | Move cursor to line mark "a" generated by marking "ma" |



### Advanced editing: Multiple Windows
This is a Vim only feature

| Command | Function |
|---|---|
| :sp | Split current window horizontally in two |
| :vsp | Split current window vertically into two |
| vim –O [n \| files…] | Opens n windows, files split vertically |
| :new | Open a new blank window |
| :on | Make current window the only window |
| :q | Quit current window |
| :qa | Quit all windows |
| :xa | Save and quit all windows |
| [Ctrl+w]+/- | Increase/decrease window size |
| [Ctrl+w] [Ctrl+w] | Toggle between windows |

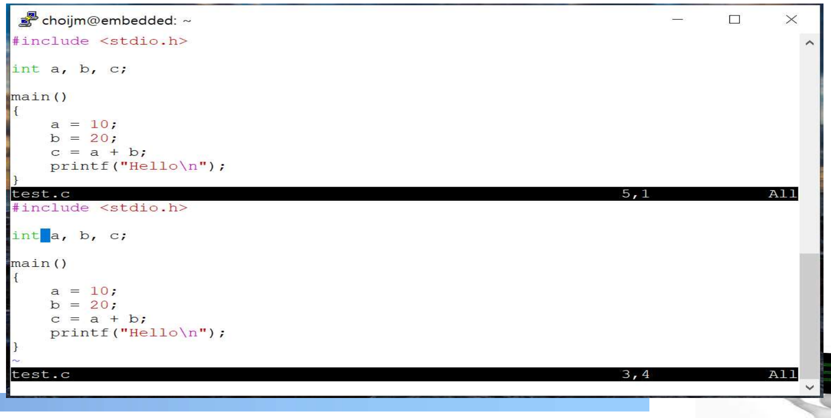


### Pattern Substitutions

- General format of substitution
  :[.|$|%]s/s1/s1[switches] or :n1,n2s/s1/s2/[switches]
- [switches] are: g|c|i|I meaning
  global/confirmation/ignore-case/no-ignore-case

Some interesting examples of pattern substitutions

| Command | Function |
|---|---|
| :1,$s/#//g | Globally remove # |
| :3,10s/^/#/ | Insert # at the beginning of line 3 to 10 |
| :$s/$/;/ | Insert a ; at the end of last line |
| :%s/abc/xyz/gc | Globally replace abc by xyz interactively |
| :1,$s/include/<&>/g | Globally replace include by <include> |



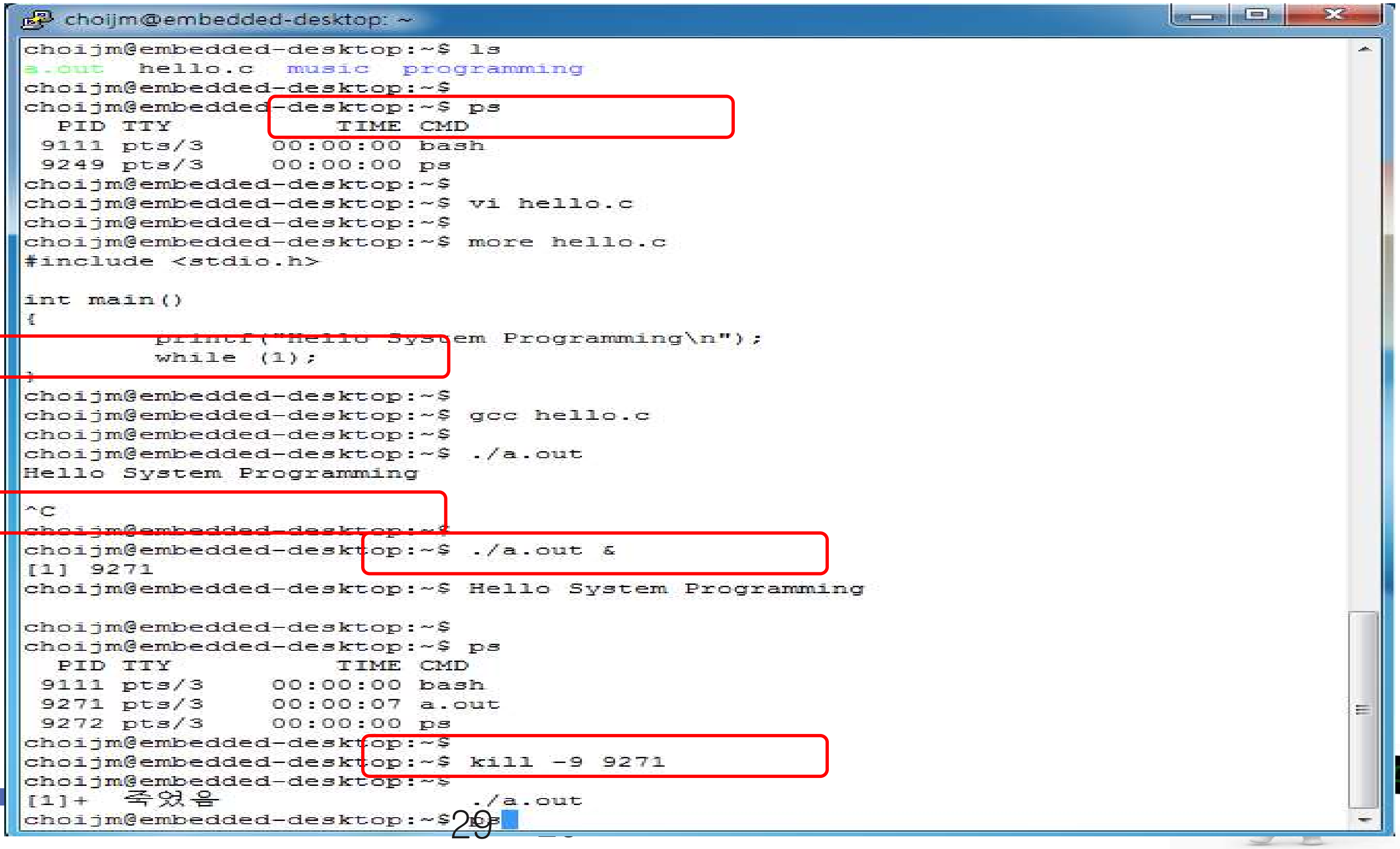
choijm@embedded: ~

```
#include <stdio.h>

int a, b, c;

main()
{
    a = 10;
    b = 20;
    c = a + b;
    printf("Hello\n");
}
test.c                                           5,1            All
#include <stdio.h>

int a, b, c;

main()
{
    a = 10;
    b = 20;
    c = a + b;
    printf("Hello\n");
}
~
test.c                                           3,4            All
```

(Source: https://www.slideshare.net/TusharadriSarkar/vim-vi-improved-23917134)

# How to use commands in Linux (8/12)

- process related commands
  - ✓ process status
    - ps, pstree, top, /proc
  - ✓ Creation and deletion
    - Implicitly: using shell (fork(), execve() and exit() internally)
    - Explicitly: signal, kill command

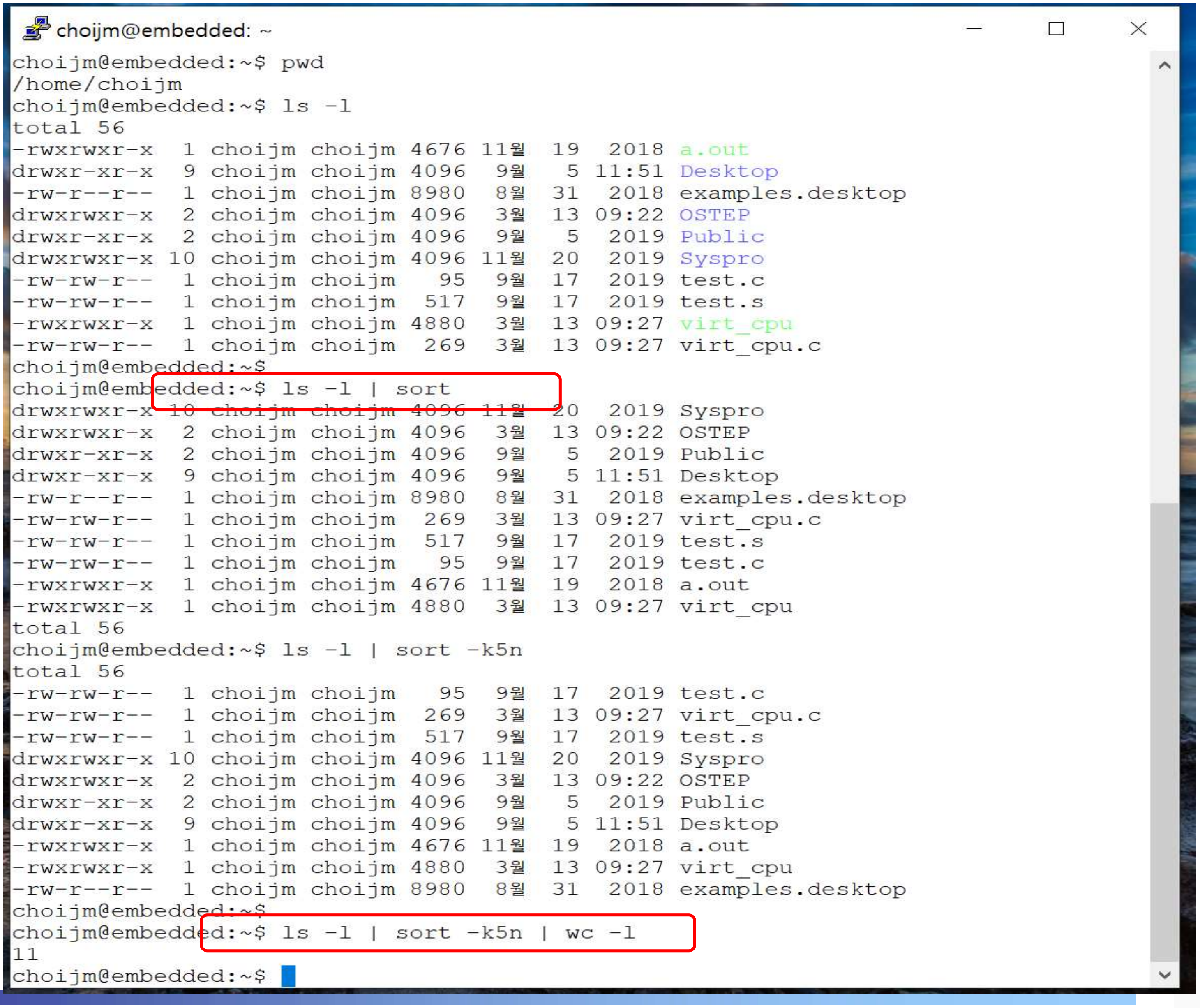


choijm@embedded-desktop: ~

```
choijm@embedded-desktop:~$ ls
a.out   hello.c   music   programming
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ps
  PID TTY          TIME CMD
 9111 pts/3    00:00:00 bash
 9249 pts/3    00:00:00 ps
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ vi hello.c
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ more hello.c
#include <stdio.h>

int main()
{
        printf("Hello System Programming\n");
        while (1);
}
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ gcc hello.c
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ./a.out
Hello System Programming

^C
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ./a.out &
[1] 9271
choijm@embedded-desktop:~$ Hello System Programming

choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ ps
  PID TTY          TIME CMD
 9111 pts/3    00:00:00 bash
 9271 pts/3    00:00:07 a.out
 9272 pts/3    00:00:00 ps
choijm@embedded-desktop:~$
choijm@embedded-desktop:~$ kill -9 9271
choijm@embedded-desktop:~$
[1]+  죽었음                 ./a.out
choijm@embedded-desktop:~$ ps
```

# How to use commands in Linux (9/12)

- Advanced commands: pipe

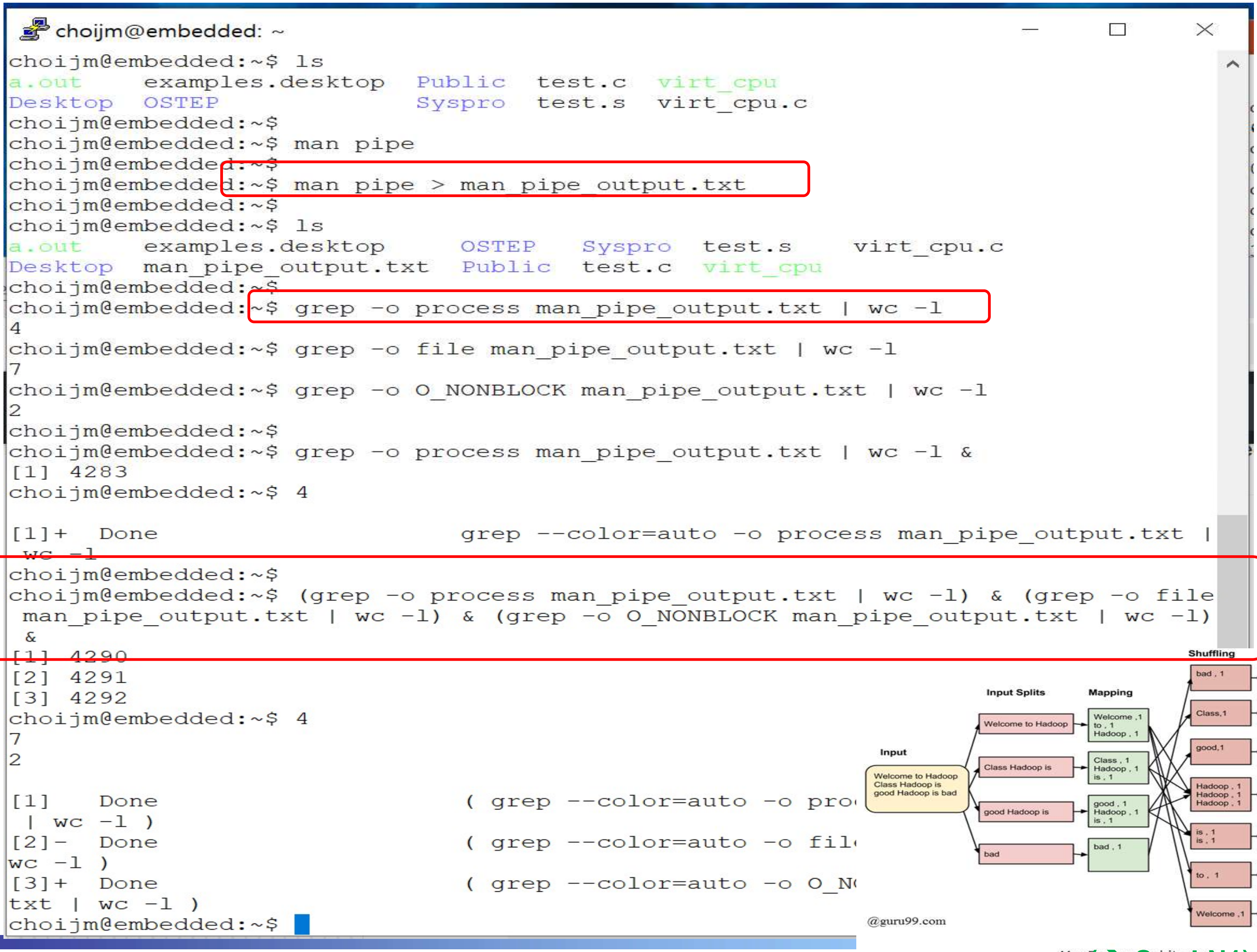


choijm@embedded: ~

```
choijm@embedded:~$ pwd
/home/choijm
choijm@embedded:~$ ls -l
total 56
-rwxrwxr-x  1 choijm choijm 4676 11월  19  2018 a.out
drwxr-xr-x  9 choijm choijm 4096  9월   5 11:51 Desktop
-rw-r--r--  1 choijm choijm 8980  8월  31  2018 examples.desktop
drwxrwxr-x  2 choijm choijm 4096  3월  13 09:22 OSTEP
drwxr-xr-x  2 choijm choijm 4096  9월   5  2019 Public
drwxrwxr-x 10 choijm choijm 4096 11월  20  2019 Syspro
-rw-rw-r--  1 choijm choijm   95  9월  17  2019 test.c
-rw-rw-r--  1 choijm choijm  517  9월  17  2019 test.s
-rwxrwxr-x  1 choijm choijm 4880  3월  13 09:27 virt_cpu
-rw-rw-r--  1 choijm choijm  269  3월  13 09:27 virt_cpu.c
choijm@embedded:~$
choijm@embedded:~$ ls -l | sort
drwxrwxr-x 10 choijm choijm 4096 11월  20  2019 Syspro
drwxrwxr-x  2 choijm choijm 4096  3월  13 09:22 OSTEP
drwxr-xr-x  2 choijm choijm 4096  9월   5  2019 Public
drwxr-xr-x  9 choijm choijm 4096  9월   5 11:51 Desktop
-rw-r--r--  1 choijm choijm 8980  8월  31  2018 examples.desktop
-rw-rw-r--  1 choijm choijm  269  3월  13 09:27 virt_cpu.c
-rw-rw-r--  1 choijm choijm  517  9월  17  2019 test.s
-rw-rw-r--  1 choijm choijm   95  9월  17  2019 test.c
-rwxrwxr-x  1 choijm choijm 4676 11월  19  2018 a.out
-rwxrwxr-x  1 choijm choijm 4880  3월  13 09:27 virt_cpu
total 56
choijm@embedded:~$ ls -l | sort -k5n
total 56
-rw-rw-r--  1 choijm choijm   95  9월  17  2019 test.c
-rw-rw-r--  1 choijm choijm  269  3월  13 09:27 virt_cpu.c
-rw-rw-r--  1 choijm choijm  517  9월  17  2019 test.s
drwxrwxr-x 10 choijm choijm 4096 11월  20  2019 Syspro
drwxrwxr-x  2 choijm choijm 4096  3월  13 09:22 OSTEP
drwxr-xr-x  2 choijm choijm 4096  9월   5  2019 Public
drwxr-xr-x  9 choijm choijm 4096  9월   5 11:51 Desktop
-rwxrwxr-x  1 choijm choijm 4676 11월  19  2018 a.out
-rwxrwxr-x  1 choijm choijm 4880  3월  13 09:27 virt_cpu
-rw-r--r--  1 choijm choijm 8980  8월  31  2018 examples.desktop
choijm@embedded:~$
choijm@embedded:~$ ls -l | sort -k5n | wc -l
11
choijm@embedded:~$
```

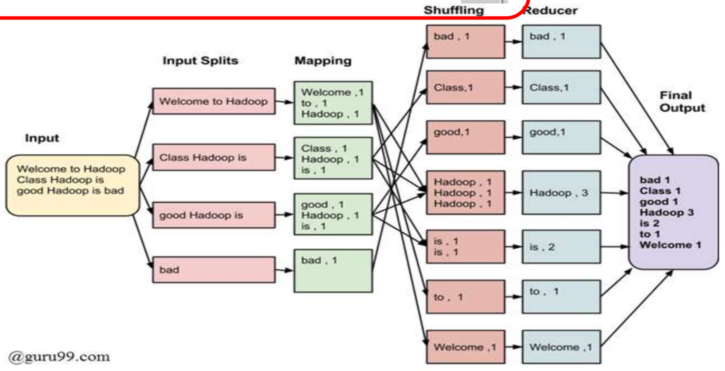
# How to use commands in Linux (10/12)

- Advanced commands: pipe, redirection and background



choijm@embedded: ~

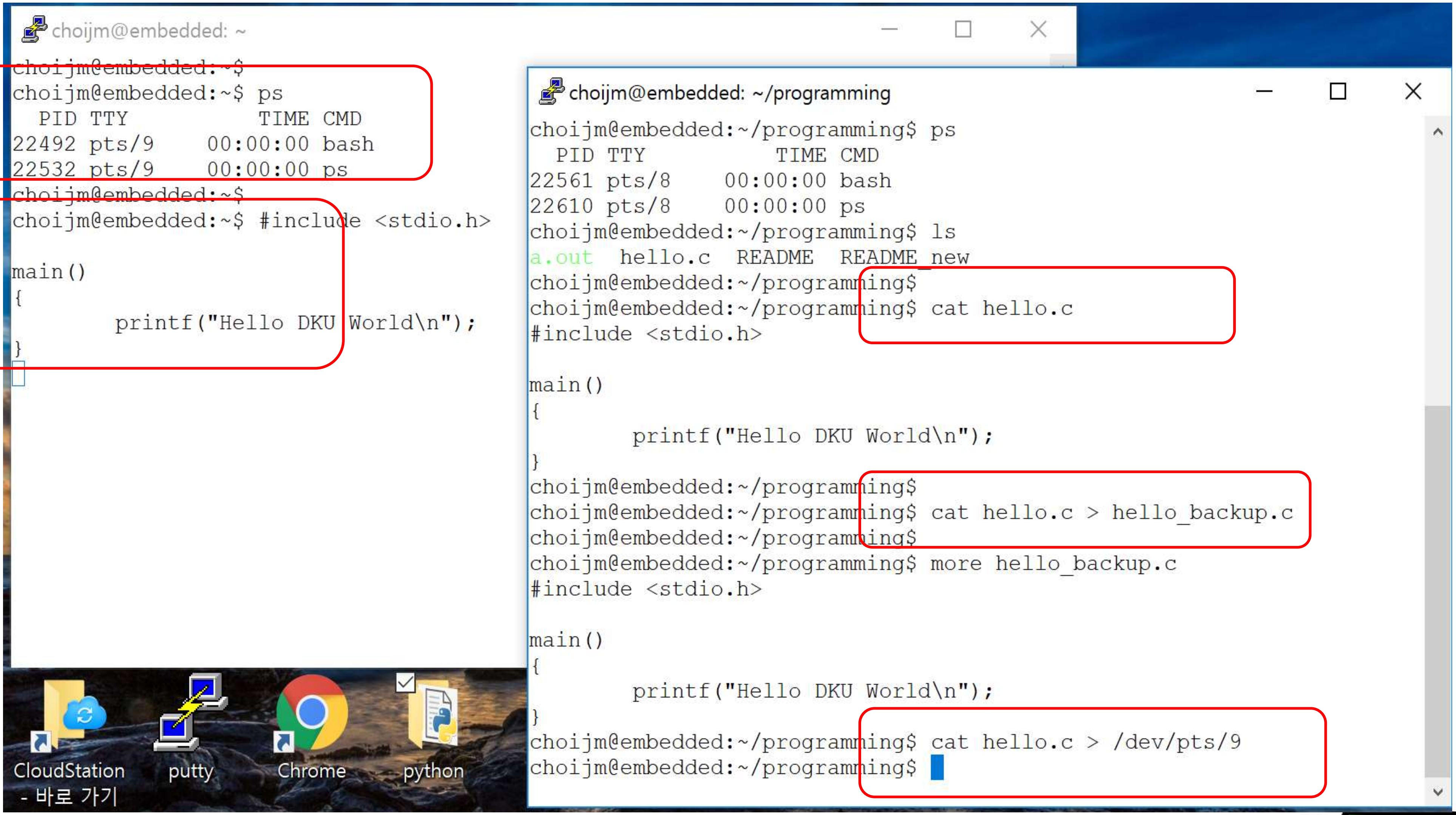
```
choijm@embedded:~$ ls
a.out     examples.desktop   Public   test.c   virt_cpu
Desktop   OSTEP              Syspro   test.s   virt_cpu.c
choijm@embedded:~$
choijm@embedded:~$ man pipe
choijm@embedded:~$
choijm@embedded:~$ man pipe > man_pipe_output.txt
choijm@embedded:~$
choijm@embedded:~$ ls
a.out     examples.desktop      OSTEP    Syspro   test.s     virt_cpu.c
Desktop   man_pipe_output.txt   Public   test.c   virt_cpu
choijm@embedded:~$
choijm@embedded:~$ grep -o process man_pipe_output.txt | wc -l
4
choijm@embedded:~$ grep -o file man_pipe_output.txt | wc -l
7
choijm@embedded:~$ grep -o O_NONBLOCK man_pipe_output.txt | wc -l
2
choijm@embedded:~$
choijm@embedded:~$ grep -o process man_pipe_output.txt | wc -l &
[1] 4283
choijm@embedded:~$ 4

[1]+  Done                    grep --color=auto -o process man_pipe_output.txt |
 wc -l
choijm@embedded:~$
choijm@embedded:~$ (grep -o process man_pipe_output.txt | wc -l) & (grep -o file
 man_pipe_output.txt | wc -l) & (grep -o O_NONBLOCK man_pipe_output.txt | wc -l)
 &
[1] 4290
[2] 4291
[3] 4292
choijm@embedded:~$ 4
7
2

[1]   Done                    ( grep --color=auto -o pro
 | wc -l )
[2]-  Done                    ( grep --color=auto -o fil
wc -l )
[3]+  Done                    ( grep --color=auto -o O_N
txt | wc -l )
choijm@embedded:~$
```



MapReduce Architecture (➔ See LN1)

# How to use commands in Linux (11/12)

- Generalization of file concept
  - ✓ Treat device, socket, IPC as a file

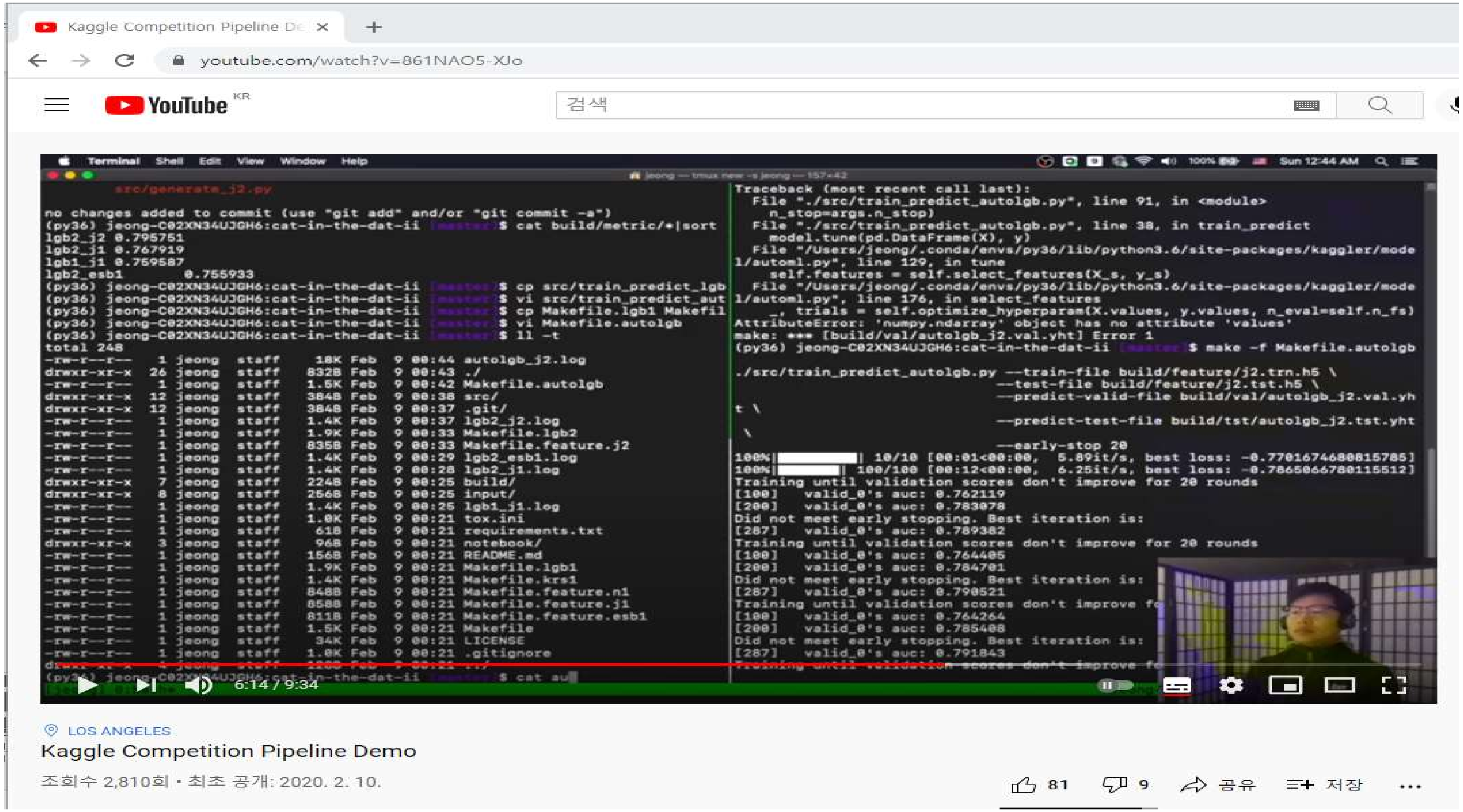


```
choijm@embedded:~$
choijm@embedded:~$ ps
  PID TTY          TIME CMD
22492 pts/9    00:00:00 bash
22532 pts/9    00:00:00 ps
choijm@embedded:~$
choijm@embedded:~$ #include <stdio.h>

main()
{
        printf("Hello DKU World\n");
}
```

```
choijm@embedded:~/programming$ ps
  PID TTY          TIME CMD
22561 pts/8    00:00:00 bash
22610 pts/8    00:00:00 ps
choijm@embedded:~/programming$ ls
a.out  hello.c  README  README_new
choijm@embedded:~/programming$
choijm@embedded:~/programming$ cat hello.c
#include <stdio.h>

main()
{
        printf("Hello DKU World\n");
}
choijm@embedded:~/programming$
choijm@embedded:~/programming$ cat hello.c > hello_backup.c
choijm@embedded:~/programming$
choijm@embedded:~/programming$ more hello_backup.c
#include <stdio.h>

main()
{
        printf("Hello DKU World\n");
}
choijm@embedded:~/programming$ cat hello.c > /dev/pts/9
choijm@embedded:~/programming$
```

## How to use commands in Linux (12/12)

- Reference: Dr. Jeong-Yoon Lee's Kaggle demo (terminal mode)



Kaggle Competition Pipeline De
youtube.com/watch?v=861NAO5-XJo
YouTube KR
검색
Terminal Shell Edit View Window Help
Sun 12:44 AM
6:14 / 9:34

```
        src/generate_j2.py

no changes added to commit (use "git add" and/or "git commit -a")
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii [master] $ cat build/metric/*|sort
lgb2_j2 0.795751
lgb2_j1 0.767919
lgb1_j1 0.759587
lgb2_esb1       0.755933
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii [master] $ cp src/train_predict_lgb
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii [master] $ vi src/train_predict_aut
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii [master] $ cp Makefile.lgb1 Makefil
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii [master] $ vi Makefile.autolgb
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii [master] $ ll -t
total 248
-rw-r--r--   1 jeong  staff   18K Feb  9 00:44 autolgb_j2.log
drwxr-xr-x  26 jeong  staff  832B Feb  9 00:43 ./
-rw-r--r--   1 jeong  staff  1.5K Feb  9 00:42 Makefile.autolgb
drwxr-xr-x  12 jeong  staff  384B Feb  9 00:38 src/
drwxr-xr-x  12 jeong  staff  384B Feb  9 00:37 .git/
-rw-r--r--   1 jeong  staff  1.4K Feb  9 00:37 lgb2_j2.log
-rw-r--r--   1 jeong  staff  1.9K Feb  9 00:33 Makefile.lgb2
-rw-r--r--   1 jeong  staff  835B Feb  9 00:33 Makefile.feature.j2
-rw-r--r--   1 jeong  staff  1.4K Feb  9 00:29 lgb2_esb1.log
-rw-r--r--   1 jeong  staff  1.4K Feb  9 00:28 lgb2_j1.log
drwxr-xr-x   7 jeong  staff  224B Feb  9 00:25 build/
drwxr-xr-x   8 jeong  staff  256B Feb  9 00:25 input/
-rw-r--r--   1 jeong  staff  1.4K Feb  9 00:25 lgb1_j1.log
-rw-r--r--   1 jeong  staff  1.0K Feb  9 00:21 tox.ini
-rw-r--r--   1 jeong  staff   61B Feb  9 00:21 requirements.txt
drwxr-xr-x   3 jeong  staff   96B Feb  9 00:21 notebook/
-rw-r--r--   1 jeong  staff  156B Feb  9 00:21 README.md
-rw-r--r--   1 jeong  staff  1.9K Feb  9 00:21 Makefile.lgb1
-rw-r--r--   1 jeong  staff  1.4K Feb  9 00:21 Makefile.krs1
-rw-r--r--   1 jeong  staff  848B Feb  9 00:21 Makefile.feature.n1
-rw-r--r--   1 jeong  staff  858B Feb  9 00:21 Makefile.feature.j1
-rw-r--r--   1 jeong  staff  811B Feb  9 00:21 Makefile.feature.esb1
-rw-r--r--   1 jeong  staff  1.5K Feb  9 00:21 Makefile
-rw-r--r--   1 jeong  staff   34K Feb  9 00:21 LICENSE
-rw-r--r--   1 jeong  staff  1.0K Feb  9 00:21 .gitignore
drwxr-xr-x   4 jeong  staff  128B Feb  9 00:21 ../
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii [master] $ cat au
```

```
Traceback (most recent call last):
  File "./src/train_predict_autolgb.py", line 91, in <module>
    n_stop=args.n_stop)
  File "./src/train_predict_autolgb.py", line 38, in train_predict
    model.tune(pd.DataFrame(X), y)
  File "/Users/jeong/.conda/envs/py36/lib/python3.6/site-packages/kaggler/mode
l/automl.py", line 129, in tune
    self.features = self.select_features(X_s, y_s)
  File "/Users/jeong/.conda/envs/py36/lib/python3.6/site-packages/kaggler/mode
l/automl.py", line 176, in select_features
    _, trials = self.optimize_hyperparam(X.values, y.values, n_eval=self.n_fs)
AttributeError: 'numpy.ndarray' object has no attribute 'values'
make: *** [build/val/autolgb_j2.val.yht] Error 1
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii [master] $ make -f Makefile.autolgb

./src/train_predict_autolgb.py --train-file build/feature/j2.trn.h5 \
                               --test-file build/feature/j2.tst.h5 \
                               --predict-valid-file build/val/autolgb_j2.val.yh
t \

                               --predict-test-file build/tst/autolgb_j2.tst.yht
 \

                               --early-stop 20
100%|██████████| 10/10 [00:01<00:00,  5.89it/s, best loss: -0.7701674680815785]
100%|████████| 100/100 [00:12<00:00,  6.25it/s, best loss: -0.7865066780115512]
Training until validation scores don't improve for 20 rounds
[100]   valid_0's auc: 0.762119
[200]   valid_0's auc: 0.783078
Did not meet early stopping. Best iteration is:
[287]   valid_0's auc: 0.789382
Training until validation scores don't improve for 20 rounds
[100]   valid_0's auc: 0.764405
[200]   valid_0's auc: 0.784701
Did not meet early stopping. Best iteration is:
[287]   valid_0's auc: 0.790521
Training until validation scores don't improve fo
[100]   valid_0's auc: 0.764264
[200]   valid_0's auc: 0.785408
Did not meet early stopping. Best iteration is:
[287]   valid_0's auc: 0.791843
Training until validation scores don't improve fo
```

LOS ANGELES

Kaggle Competition Pipeline Demo

조회수 2,810회 • 최초 공개: 2020. 2. 10.

81  9  공유  저장

(Source: https://www.youtube.com/watch?v=861NAO5-XJo)

# How to make and run a program in Linux (1/7)

- Overall

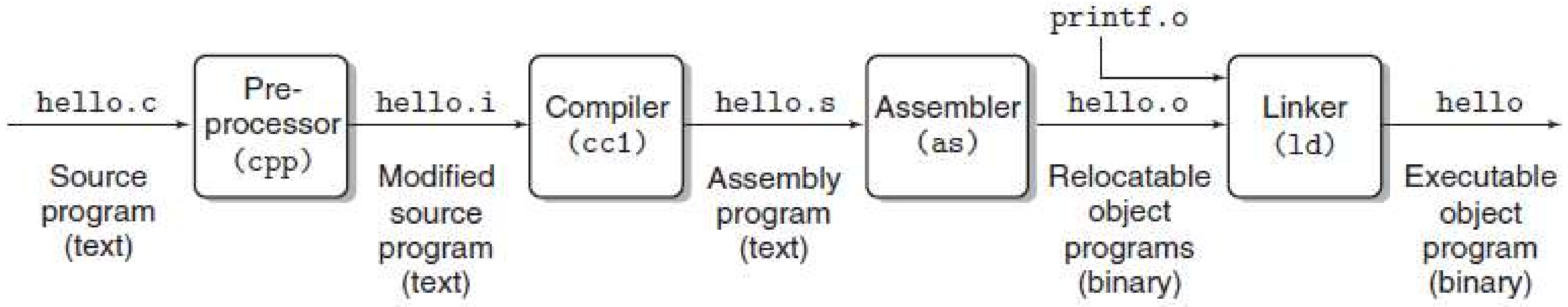
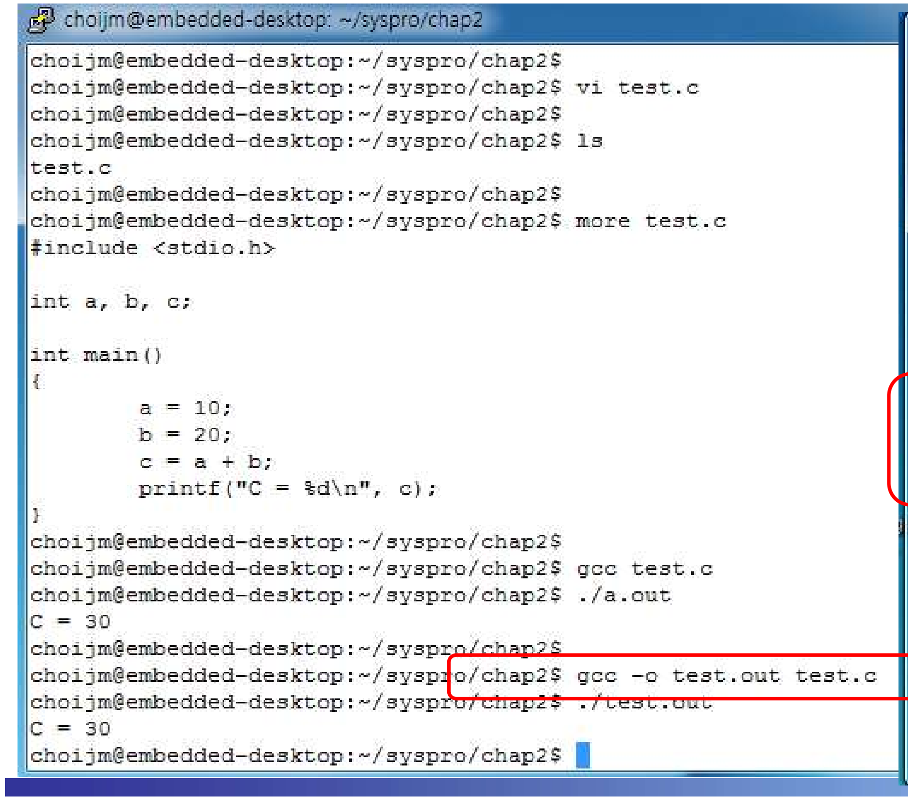
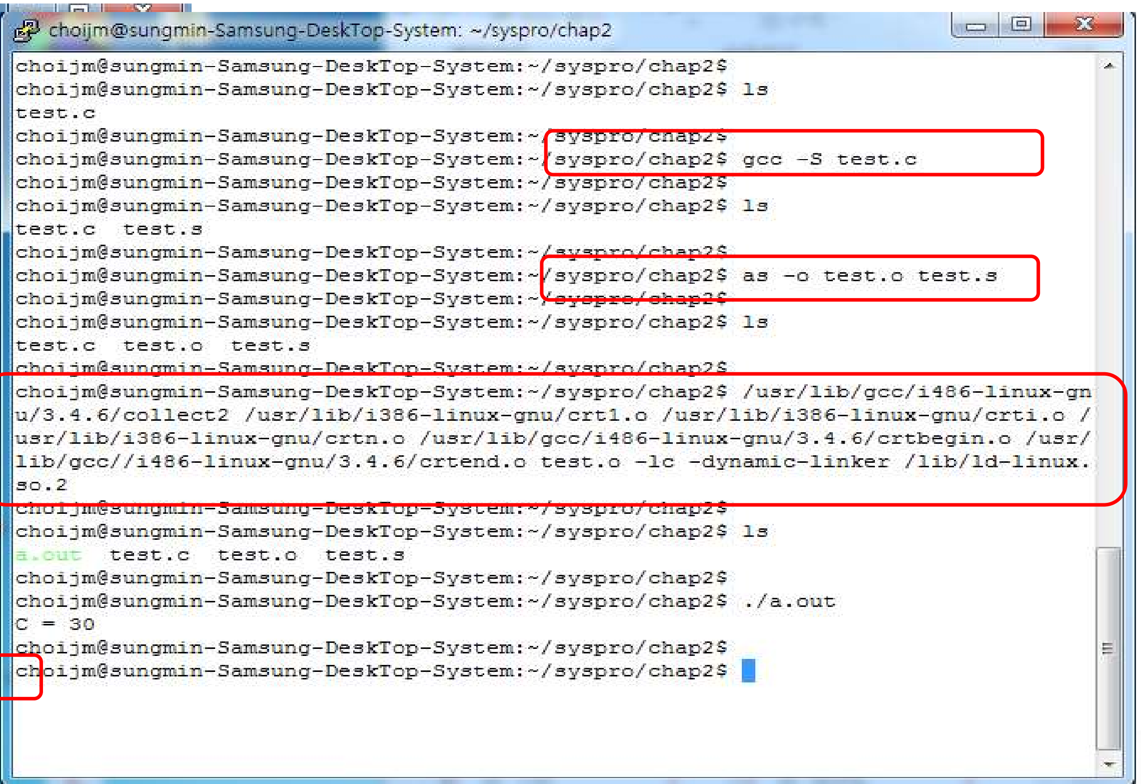


Figure 1.3 The compilation system.

**(Source: computer systems: a programmer perspective, Figure 1.3)**


choijm@embedded-desktop: ~/syspro/chap2

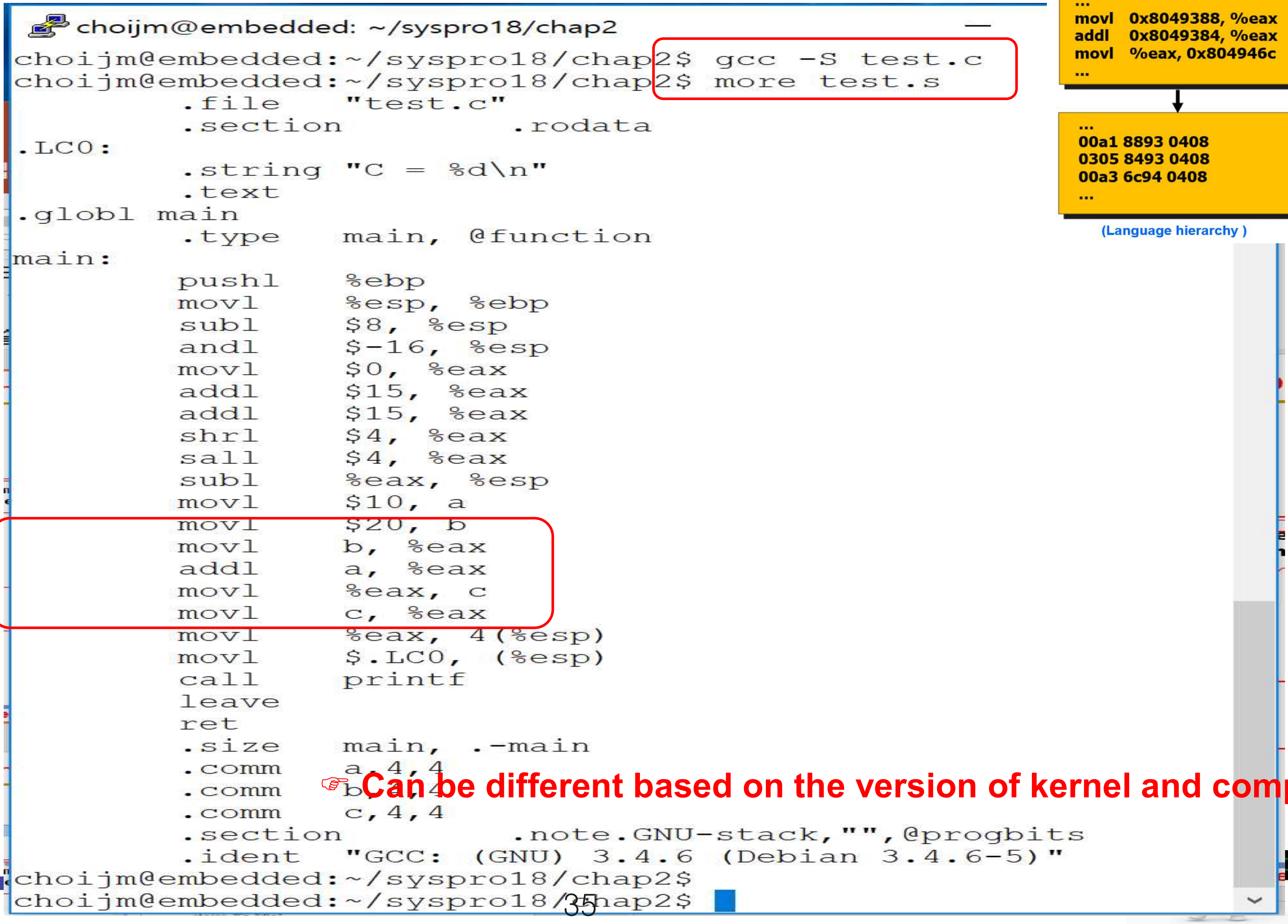
```
choijm@embedded-desktop:~/syspro/chap2$
choijm@embedded-desktop:~/syspro/chap2$ vi test.c
choijm@embedded-desktop:~/syspro/chap2$
choijm@embedded-desktop:~/syspro/chap2$ ls
test.c
choijm@embedded-desktop:~/syspro/chap2$
choijm@embedded-desktop:~/syspro/chap2$ more test.c
#include <stdio.h>

int a, b, c;

int main()
{
        a = 10;
        b = 20;
        c = a + b;
        printf("C = %d\n", c);
}
choijm@embedded-desktop:~/syspro/chap2$
choijm@embedded-desktop:~/syspro/chap2$ gcc test.c
choijm@embedded-desktop:~/syspro/chap2$ ./a.out
C = 30
choijm@embedded-desktop:~/syspro/chap2$
choijm@embedded-desktop:~/syspro/chap2$ gcc -o test.out test.c
choijm@embedded-desktop:~/syspro/chap2$ ./test.out
C = 30
choijm@embedded-desktop:~/syspro/chap2$
```


choijm@sungmin-Samsung-DeskTop-System: ~/syspro/chap2

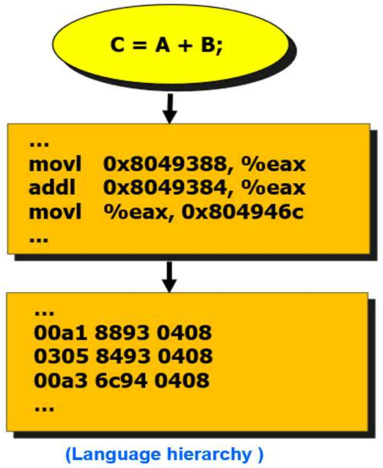
```
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$ ls
test.c
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$ gcc -S test.c
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$ ls
test.c  test.s
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$ as -o test.o test.s
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$ ls
test.c  test.o  test.s
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$ /usr/lib/gcc/i486-linux-gn
u/3.4.6/collect2 /usr/lib/i386-linux-gnu/crt1.o /usr/lib/i386-linux-gnu/crti.o /
usr/lib/i386-linux-gnu/crtn.o /usr/lib/gcc/i486-linux-gnu/3.4.6/crtbegin.o /usr/
lib/gcc//i486-linux-gnu/3.4.6/crtend.o test.o -lc -dynamic-linker /lib/ld-linux.
so.2
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$ ls
a.out  test.c  test.o  test.s
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$ ./a.out
C = 30
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$
choijm@sungmin-Samsung-DeskTop-System:~/syspro/chap2$
```

## How to make and run a program in Linux (2/7)

- Assembly code



```
choijm@embedded: ~/syspro18/chap2
choijm@embedded:~/syspro18/chap2$ gcc -S test.c
choijm@embedded:~/syspro18/chap2$ more test.s
        .file   "test.c"
        .section        .rodata
.LC0:
        .string "C = %d\n"
        .text
.globl main
        .type   main, @function
main:
        pushl   %ebp
        movl    %esp, %ebp
        subl    $8, %esp
        andl    $-16, %esp
        movl    $0, %eax
        addl    $15, %eax
        addl    $15, %eax
        shrl    $4, %eax
        sall    $4, %eax
        subl    %eax, %esp
        movl    $10, a
        movl    $20, b
        movl    b, %eax
        addl    a, %eax
        movl    %eax, c
        movl    c, %eax
        movl    %eax, 4(%esp)
        movl    $.LC0, (%esp)
        call    printf
        leave
        ret
        .size   main, .-main
        .comm   a,4,4
        .comm   b,4,4
        .comm   c,4,4
        .section        .note.GNU-stack,"",@progbits
        .ident  "GCC: (GNU) 3.4.6 (Debian 3.4.6-5)"
choijm@embedded:~/syspro18/chap2$
choijm@embedded:~/syspro18/chap2$
```



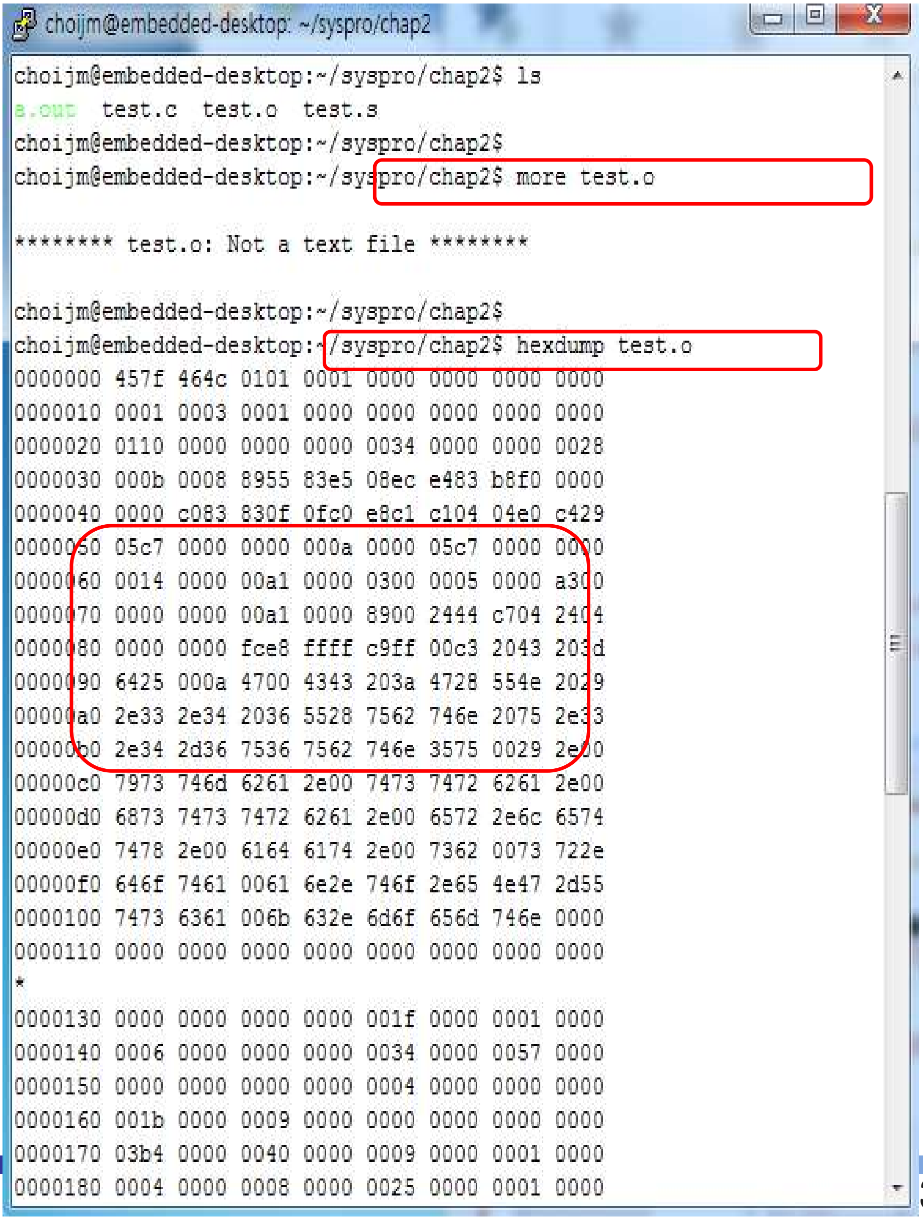
```
C = A + B;

...
movl   0x8049388, %eax
addl   0x8049384, %eax
movl   %eax, 0x804946c
...

...
00a1 8893 0408
0305 8493 0408
00a3 6c94 0408
...
```

(Language hierarchy )

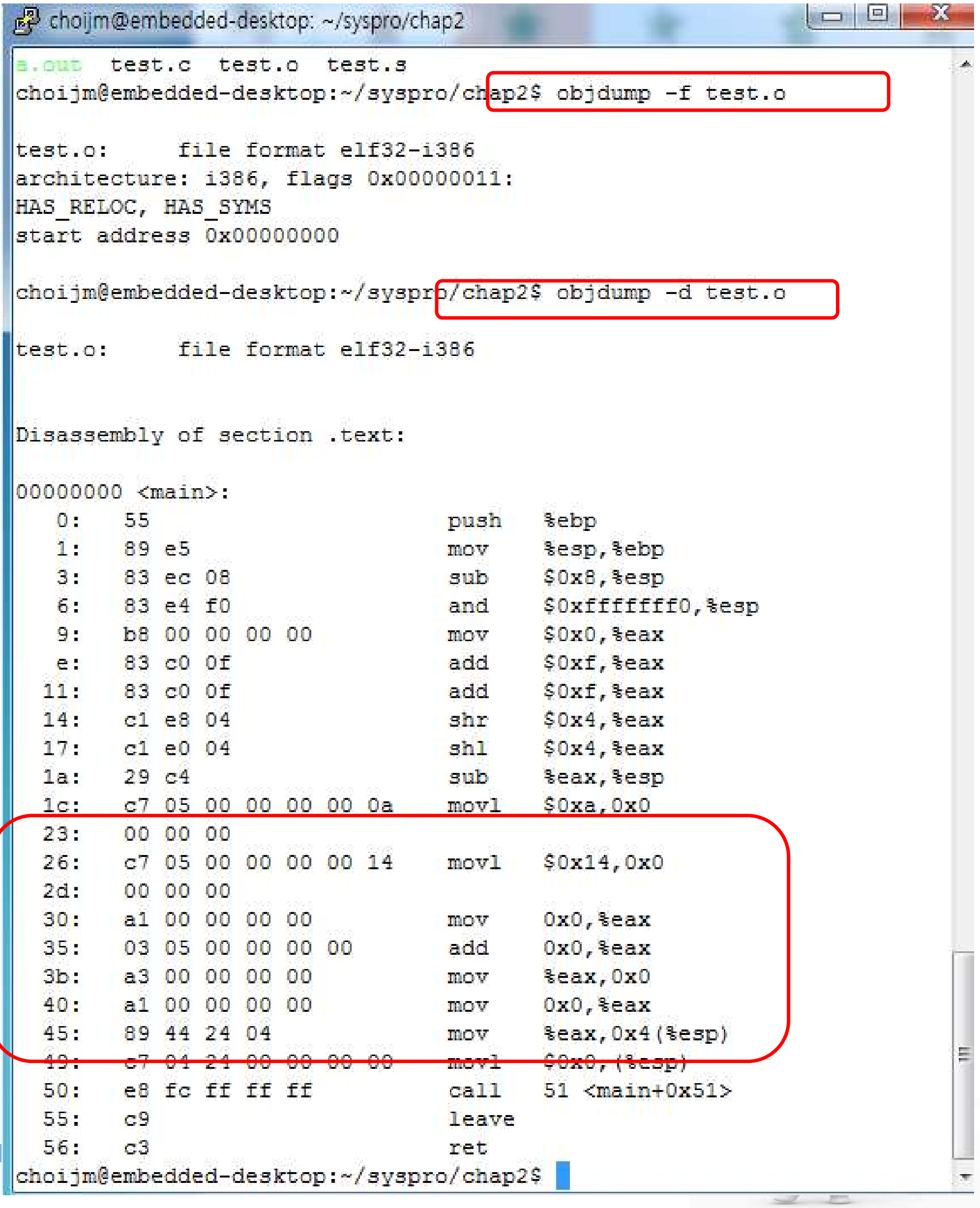
☞ **Can be different based on the version of kernel and compiler**

# How to make and run a program in Linux (3/7)

- Relocatable code
  - ✓ Hexdump (or xxd), objdump



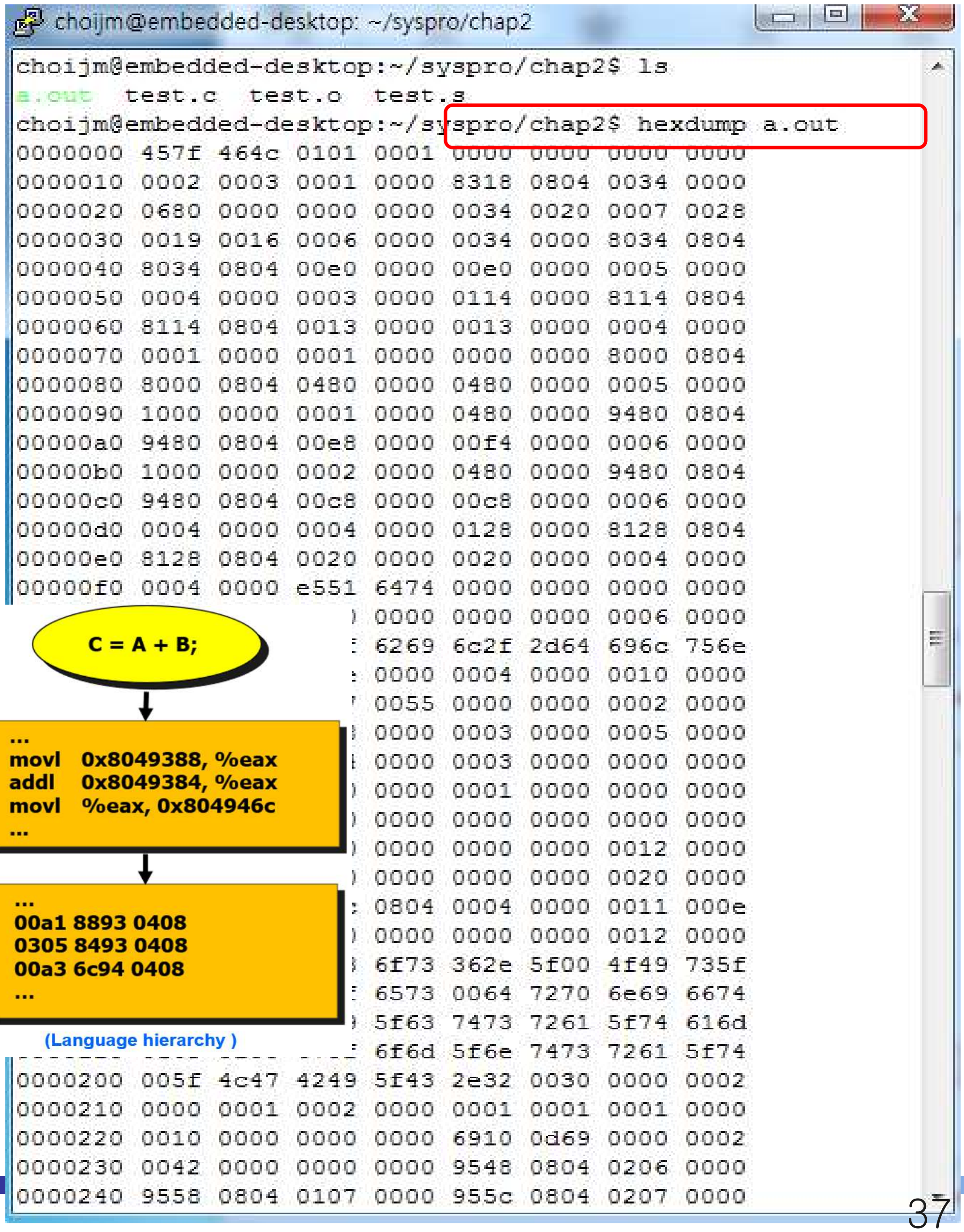
```
choijm@embedded-desktop:~/syspro/chap2$ ls
a.out  test.c  test.o  test.s
choijm@embedded-desktop:~/syspro/chap2$
choijm@embedded-desktop:~/syspro/chap2$ more test.o

******** test.o: Not a text file ********

choijm@embedded-desktop:~/syspro/chap2$
choijm@embedded-desktop:~/syspro/chap2$ hexdump test.o
0000000 457f 464c 0101 0001 0000 0000 0000 0000
0000010 0001 0003 0001 0000 0000 0000 0000 0000
0000020 0110 0000 0000 0000 0034 0000 0000 0028
0000030 000b 0008 8955 83e5 08ec e483 b8f0 0000
0000040 0000 c083 830f 0fc0 e8c1 c104 04e0 c429
0000050 05c7 0000 0000 000a 0000 05c7 0000 0000
0000060 0014 0000 00a1 0000 0300 0005 0000 a300
0000070 0000 0000 00a1 0000 8900 2444 c704 2404
0000080 0000 0000 fce8 ffff c9ff 00c3 2043 203d
0000090 6425 000a 4700 4343 203a 4728 554e 2029
00000a0 2e33 2e34 2036 5528 7562 746e 2075 2e33
00000b0 2e34 2d36 7536 7562 746e 3575 0029 2e00
00000c0 7973 746d 6261 2e00 7473 7472 6261 2e00
00000d0 6873 7473 7472 6261 2e00 6572 2e6c 6574
00000e0 7478 2e00 6164 6174 2e00 7362 0073 722e
00000f0 646f 7461 0061 6e2e 746f 2e65 4e47 2d55
0000100 7473 6361 006b 632e 6d6f 656d 746e 0000
0000110 0000 0000 0000 0000 0000 0000 0000 0000
*
0000130 0000 0000 0000 0000 001f 0000 0001 0000
0000140 0006 0000 0000 0000 0034 0000 0057 0000
0000150 0000 0000 0000 0000 0004 0000 0000 0000
0000160 001b 0000 0009 0000 0000 0000 0000 0000
0000170 03b4 0000 0040 0000 0009 0000 0001 0000
0000180 0004 0000 0008 0000 0025 0000 0001 0000
```



```
a.out  test.c  test.o  test.s
choijm@embedded-desktop:~/syspro/chap2$ objdump -f test.o

test.o:     file format elf32-i386
architecture: i386, flags 0x00000011:
HAS_RELOC, HAS_SYMS
start address 0x00000000

choijm@embedded-desktop:~/syspro/chap2$ objdump -d test.o

test.o:     file format elf32-i386


Disassembly of section .text:

00000000 <main>:
   0:   55                      push   %ebp
   1:   89 e5                   mov    %esp,%ebp
   3:   83 ec 08                sub    $0x8,%esp
   6:   83 e4 f0                and    $0xfffffff0,%esp
   9:   b8 00 00 00 00          mov    $0x0,%eax
   e:   83 c0 0f                add    $0xf,%eax
  11:   83 c0 0f                add    $0xf,%eax
  14:   c1 e8 04                shr    $0x4,%eax
  17:   c1 e0 04                shl    $0x4,%eax
  1a:   29 c4                   sub    %eax,%esp
  1c:   c7 05 00 00 00 00 0a    movl   $0xa,0x0
  23:   00 00 00
  26:   c7 05 00 00 00 00 14    movl   $0x14,0x0
  2d:   00 00 00
  30:   a1 00 00 00 00          mov    0x0,%eax
  35:   03 05 00 00 00 00       add    0x0,%eax
  3b:   a3 00 00 00 00          mov    %eax,0x0
  40:   a1 00 00 00 00          mov    0x0,%eax
  45:   89 44 24 04             mov    %eax,0x4(%esp)
  49:   c7 04 24 00 00 00 00    movl   $0x0,(%esp)
  50:   e8 fc ff ff ff          call   51 <main+0x51>
  55:   c9                      leave
  56:   c3                      ret
choijm@embedded-desktop:~/syspro/chap2$
```

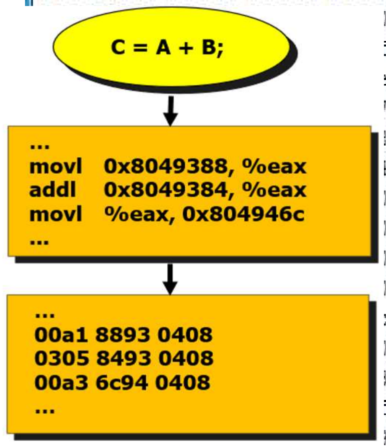
# How to make and run a program in Linux (4/7)

- Executable code

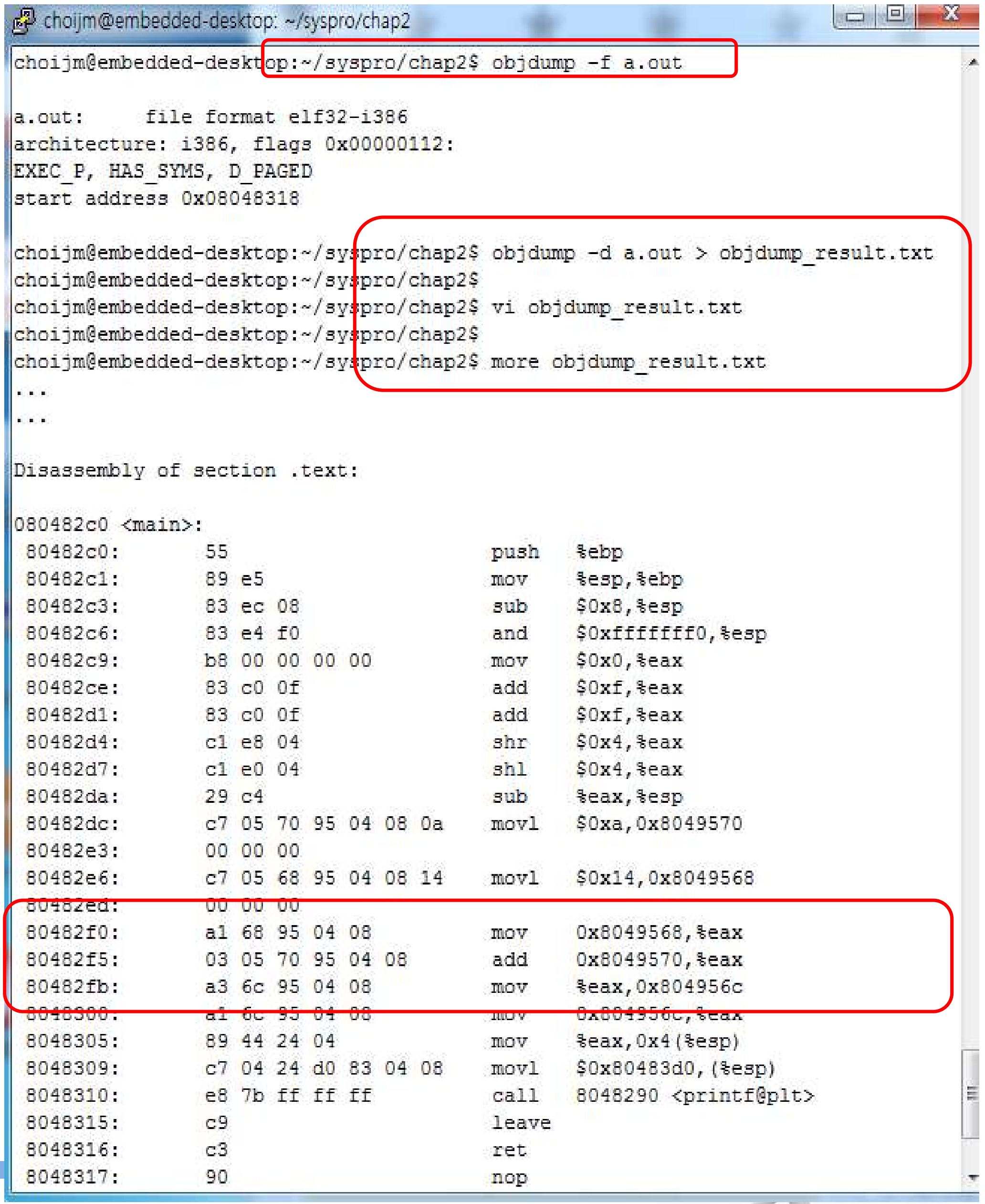


choijm@embedded-desktop: ~/syspro/chap2

```
choijm@embedded-desktop:~/syspro/chap2$ ls
a.out  test.c  test.o  test.s
choijm@embedded-desktop:~/syspro/chap2$ hexdump a.out
0000000 457f 464c 0101 0001 0000 0000 0000 0000
0000010 0002 0003 0001 0000 8318 0804 0034 0000
0000020 0680 0000 0000 0000 0034 0020 0007 0028
0000030 0019 0016 0006 0000 0034 0000 8034 0804
0000040 8034 0804 00e0 0000 00e0 0000 0005 0000
0000050 0004 0000 0003 0000 0114 0000 8114 0804
0000060 8114 0804 0013 0000 0013 0000 0004 0000
0000070 0001 0000 0001 0000 0000 0000 8000 0804
0000080 8000 0804 0480 0000 0480 0000 0005 0000
0000090 1000 0000 0001 0000 0480 0000 9480 0804
00000a0 9480 0804 00e8 0000 00f4 0000 0006 0000
00000b0 1000 0000 0002 0000 0480 0000 9480 0804
00000c0 9480 0804 00c8 0000 00c8 0000 0006 0000
00000d0 0004 0000 0004 0000 0128 0000 8128 0804
00000e0 8128 0804 0020 0000 0020 0000 0004 0000
00000f0 0004 0000 e551 6474 0000 0000 0000 0000
                          0000 0000 0000 0006 0000
                          6269 6c2f 2d64 696c 756e
                          0000 0004 0000 0010 0000
                          0055 0000 0000 0002 0000
                          0000 0003 0000 0005 0000
                          0000 0003 0000 0000 0000
                          0000 0001 0000 0000 0000
                          0000 0000 0000 0000 0000
                          0000 0000 0000 0012 0000
                          0000 0000 0000 0020 0000
                          0804 0004 0000 0011 000e
                          0000 0000 0000 0012 0000
                          6f73 362e 5f00 4f49 735f
                          6573 0064 7270 6e69 6674
                          5f63 7473 7261 5f74 616d
                          6f6d 5f6e 7473 7261 5f74
0000200 005f 4c47 4249 5f43 2e32 0030 0000 0002
0000210 0000 0001 0002 0000 0001 0001 0001 0000
0000220 0010 0000 0000 0000 6910 0d69 0000 0002
0000230 0042 0000 0000 0000 9548 0804 0206 0000
0000240 9558 0804 0107 0000 955c 0804 0207 0000
```



(Language hierarchy )

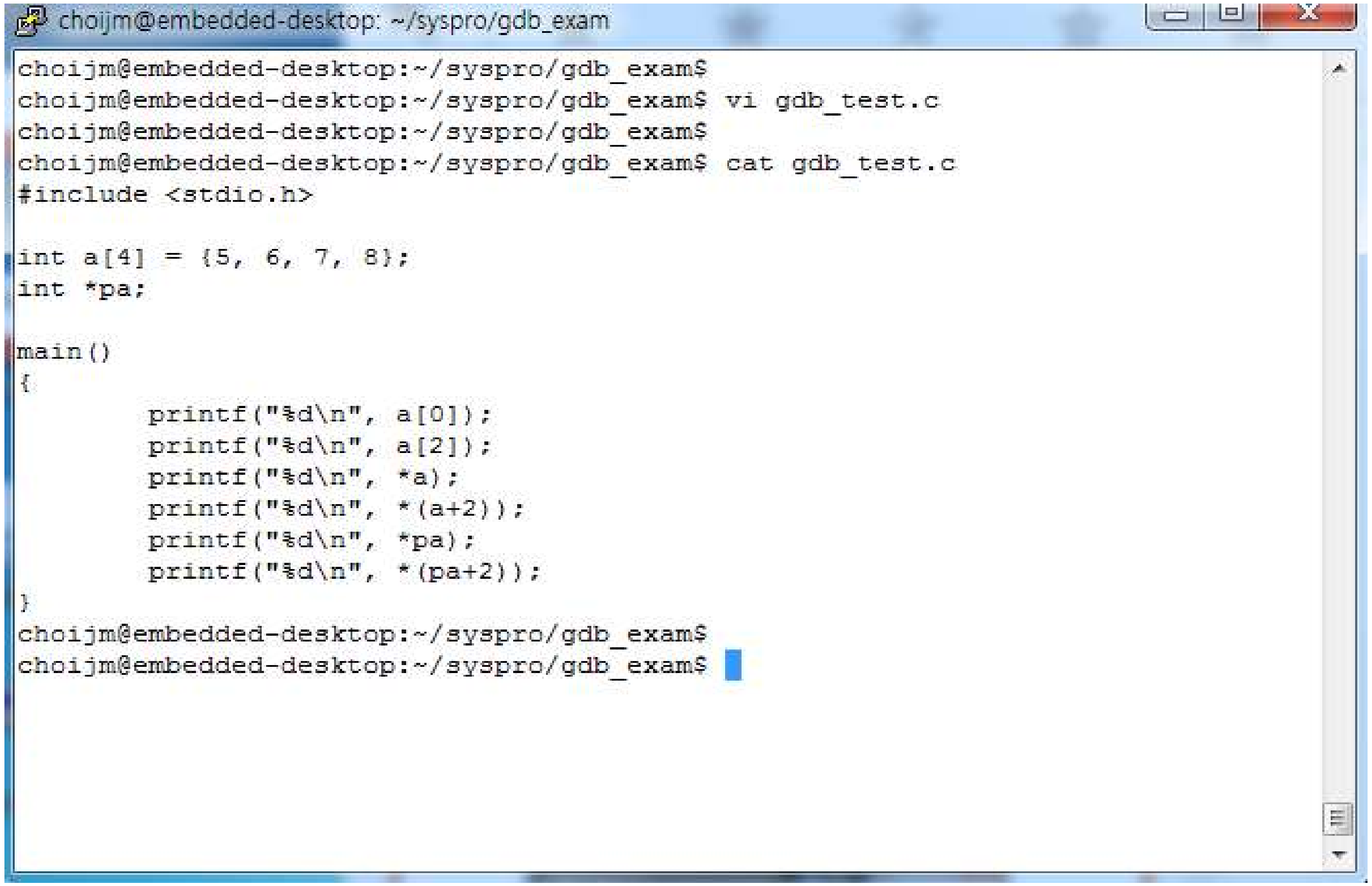


choijm@embedded-desktop: ~/syspro/chap2

```
choijm@embedded-desktop:~/syspro/chap2$ objdump -f a.out

a.out:     file format elf32-i386
architecture: i386, flags 0x00000112:
EXEC_P, HAS_SYMS, D_PAGED
start address 0x08048318

choijm@embedded-desktop:~/syspro/chap2$ objdump -d a.out > objdump_result.txt
choijm@embedded-desktop:~/syspro/chap2$
choijm@embedded-desktop:~/syspro/chap2$ vi objdump_result.txt
choijm@embedded-desktop:~/syspro/chap2$
choijm@embedded-desktop:~/syspro/chap2$ more objdump_result.txt
...
...

Disassembly of section .text:

080482c0 <main>:
 80482c0:       55                      push   %ebp
 80482c1:       89 e5                   mov    %esp,%ebp
 80482c3:       83 ec 08                sub    $0x8,%esp
 80482c6:       83 e4 f0                and    $0xfffffff0,%esp
 80482c9:       b8 00 00 00 00          mov    $0x0,%eax
 80482ce:       83 c0 0f                add    $0xf,%eax
 80482d1:       83 c0 0f                add    $0xf,%eax
 80482d4:       c1 e8 04                shr    $0x4,%eax
 80482d7:       c1 e0 04                shl    $0x4,%eax
 80482da:       29 c4                   sub    %eax,%esp
 80482dc:       c7 05 70 95 04 08 0a    movl   $0xa,0x8049570
 80482e3:       00 00 00
 80482e6:       c7 05 68 95 04 08 14    movl   $0x14,0x8049568
 80482ed:       00 00 00
 80482f0:       a1 68 95 04 08          mov    0x8049568,%eax
 80482f5:       03 05 70 95 04 08       add    0x8049570,%eax
 80482fb:       a3 6c 95 04 08          mov    %eax,0x804956c
 8048300:       a1 6c 95 04 08          mov    0x804956c,%eax
 8048305:       89 44 24 04             mov    %eax,0x4(%esp)
 8048309:       c7 04 24 d0 83 04 08    movl   $0x80483d0,(%esp)
 8048310:       e8 7b ff ff ff          call   8048290 <printf@plt>
 8048315:       c9                      leave
 8048316:       c3                      ret
 8048317:       90                      nop
```

# How to make and run a program in Linux (5/7)

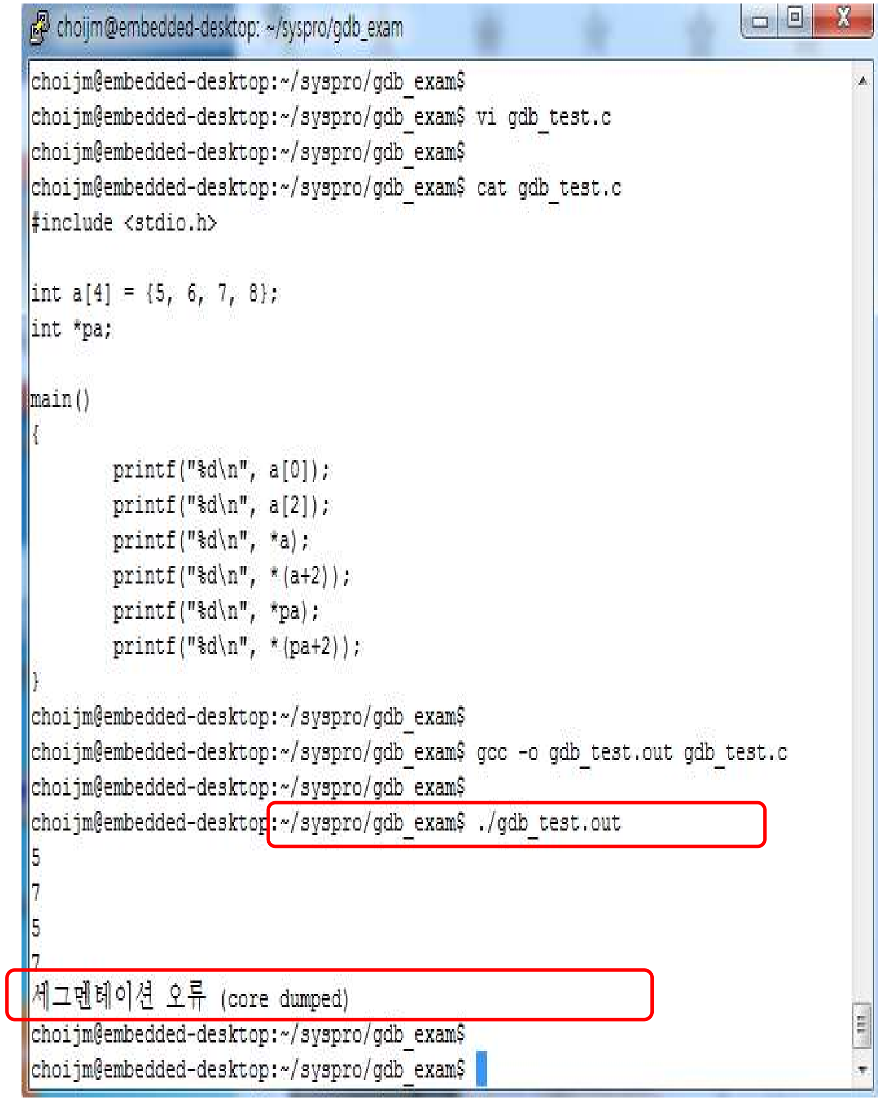
- What are the execution results of this program?



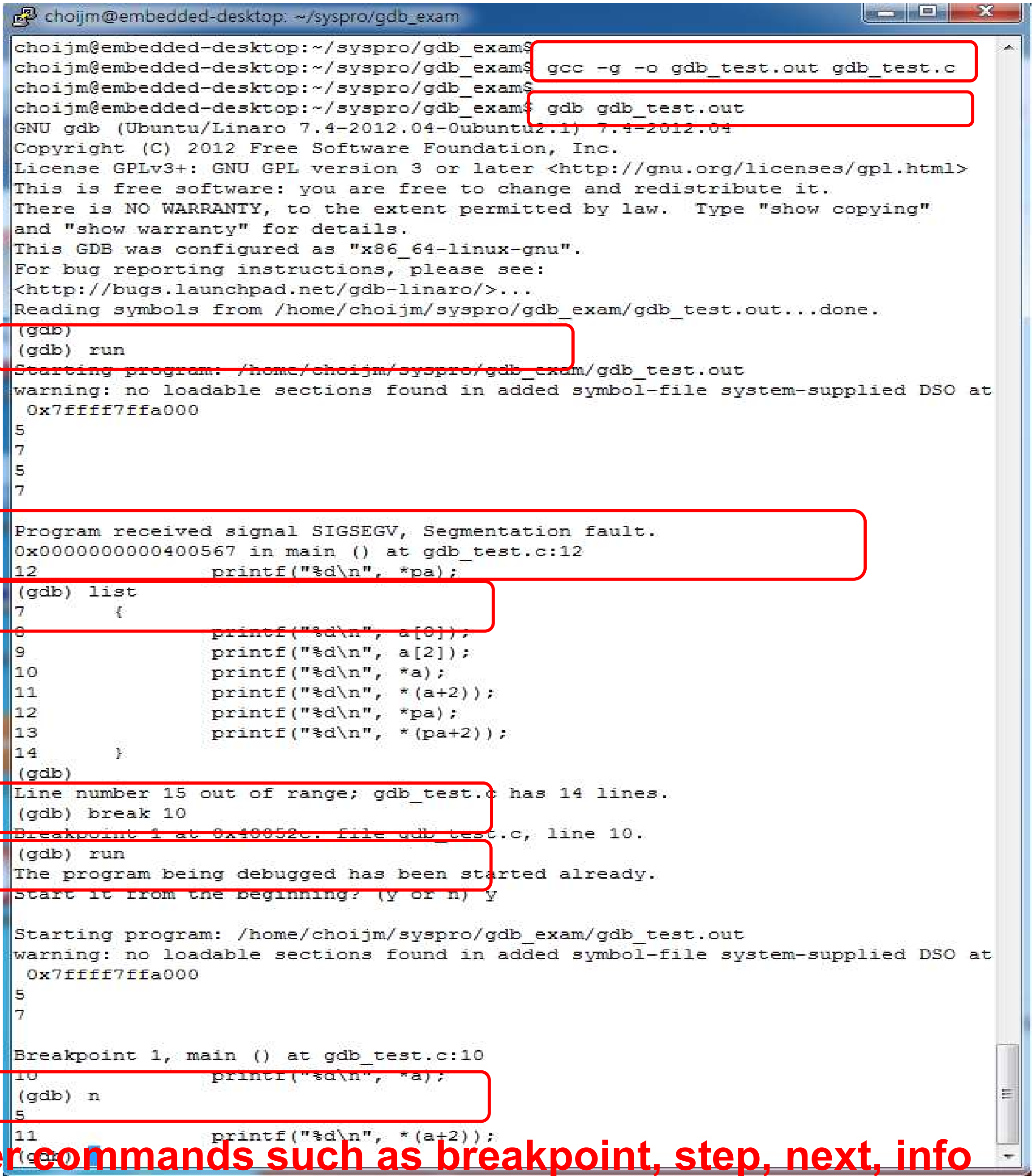
```
choijm@embedded-desktop:~/syspro/gdb_exam$
choijm@embedded-desktop:~/syspro/gdb_exam$ vi gdb_test.c
choijm@embedded-desktop:~/syspro/gdb_exam$
choijm@embedded-desktop:~/syspro/gdb_exam$ cat gdb_test.c
#include <stdio.h>

int a[4] = {5, 6, 7, 8};
int *pa;

main()
{
        printf("%d\n", a[0]);
        printf("%d\n", a[2]);
        printf("%d\n", *a);
        printf("%d\n", *(a+2));
        printf("%d\n", *pa);
        printf("%d\n", *(pa+2));
}
choijm@embedded-desktop:~/syspro/gdb_exam$
choijm@embedded-desktop:~/syspro/gdb_exam$
```

# How to make and run a program in Linux (6/7)

- debugger



```
choijm@embedded-desktop:~/syspro/gdb_exam$
choijm@embedded-desktop:~/syspro/gdb_exam$ vi gdb_test.c
choijm@embedded-desktop:~/syspro/gdb_exam$
choijm@embedded-desktop:~/syspro/gdb_exam$ cat gdb_test.c
#include <stdio.h>

int a[4] = {5, 6, 7, 8};
int *pa;

main()
{
        printf("%d\n", a[0]);
        printf("%d\n", a[2]);
        printf("%d\n", *a);
        printf("%d\n", *(a+2));
        printf("%d\n", *pa);
        printf("%d\n", *(pa+2));
}
choijm@embedded-desktop:~/syspro/gdb_exam$
choijm@embedded-desktop:~/syspro/gdb_exam$ gcc -o gdb_test.out gdb_test.c
choijm@embedded-desktop:~/syspro/gdb_exam$
choijm@embedded-desktop:~/syspro/gdb_exam$ ./gdb_test.out
5
7
5
7
세그멘테이션 오류 (core dumped)
choijm@embedded-desktop:~/syspro/gdb_exam$
choijm@embedded-desktop:~/syspro/gdb_exam$
```



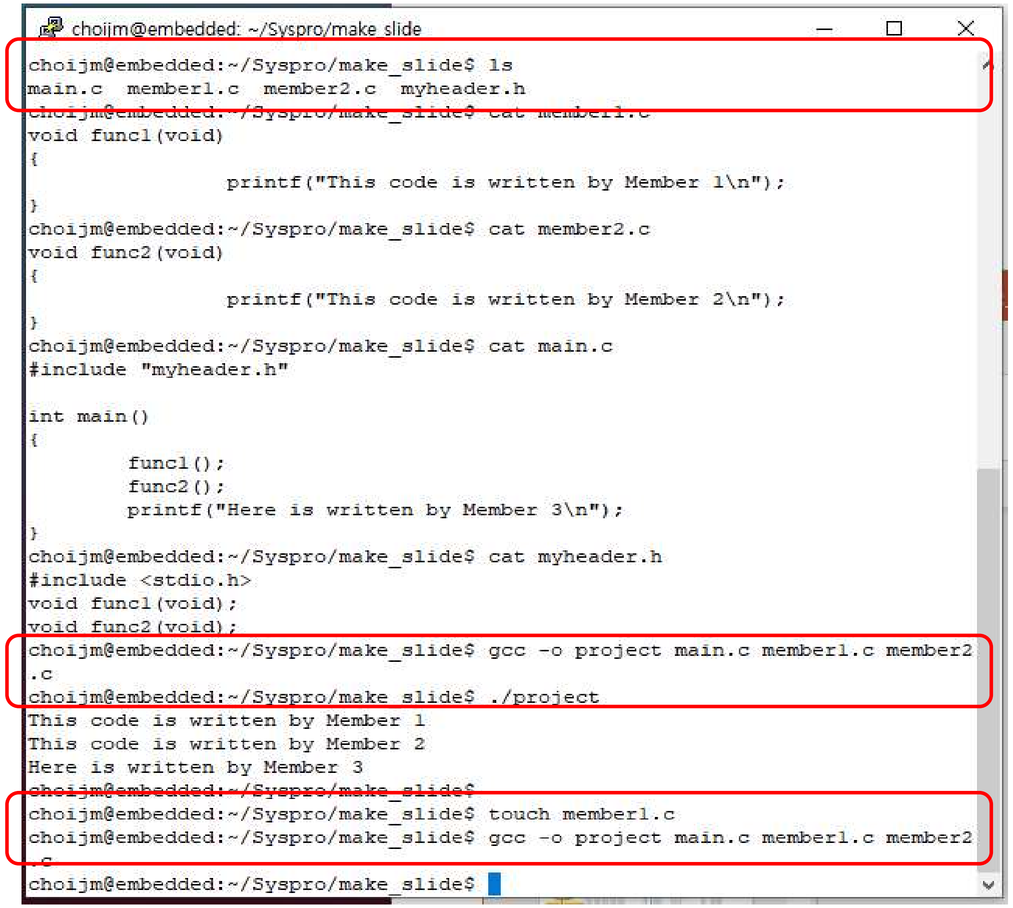
```
choijm@embedded-desktop:~/syspro/gdb_exam$
choijm@embedded-desktop:~/syspro/gdb_exam$ gcc -g -o gdb_test.out gdb_test.c
choijm@embedded-desktop:~/syspro/gdb_exam$
choijm@embedded-desktop:~/syspro/gdb_exam$ gdb gdb_test.out
GNU gdb (Ubuntu/Linaro 7.4-2012.04-0ubuntu2.1) 7.4-2012.04
Copyright (C) 2012 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
For bug reporting instructions, please see:
<http://bugs.launchpad.net/gdb-linaro/>...
Reading symbols from /home/choijm/syspro/gdb_exam/gdb_test.out...done.
(gdb)
(gdb) run
Starting program: /home/choijm/syspro/gdb_exam/gdb_test.out
warning: no loadable sections found in added symbol-file system-supplied DSO at
 0x7ffff7ffa000
5
7
5
7

Program received signal SIGSEGV, Segmentation fault.
0x0000000000400567 in main () at gdb_test.c:12
12              printf("%d\n", *pa);
(gdb) list
7       {
8               printf("%d\n", a[0]);
9               printf("%d\n", a[2]);
10              printf("%d\n", *a);
11              printf("%d\n", *(a+2));
12              printf("%d\n", *pa);
13              printf("%d\n", *(pa+2));
14      }
(gdb)
Line number 15 out of range; gdb_test.c has 14 lines.
(gdb) break 10
Breakpoint 1 at 0x40052c: file gdb_test.c, line 10.
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y

Starting program: /home/choijm/syspro/gdb_exam/gdb_test.out
warning: no loadable sections found in added symbol-file system-supplied DSO at
 0x7ffff7ffa000
5
7

Breakpoint 1, main () at gdb_test.c:10
10              printf("%d\n", *a);
(gdb) n
5
11              printf("%d\n", *(a+2));
(gdb)
```

☞ **There are various valuable debugger commands such as breakpoint, step, next, info reg, … ➔ See http://beej.us/guide/bggdb/**

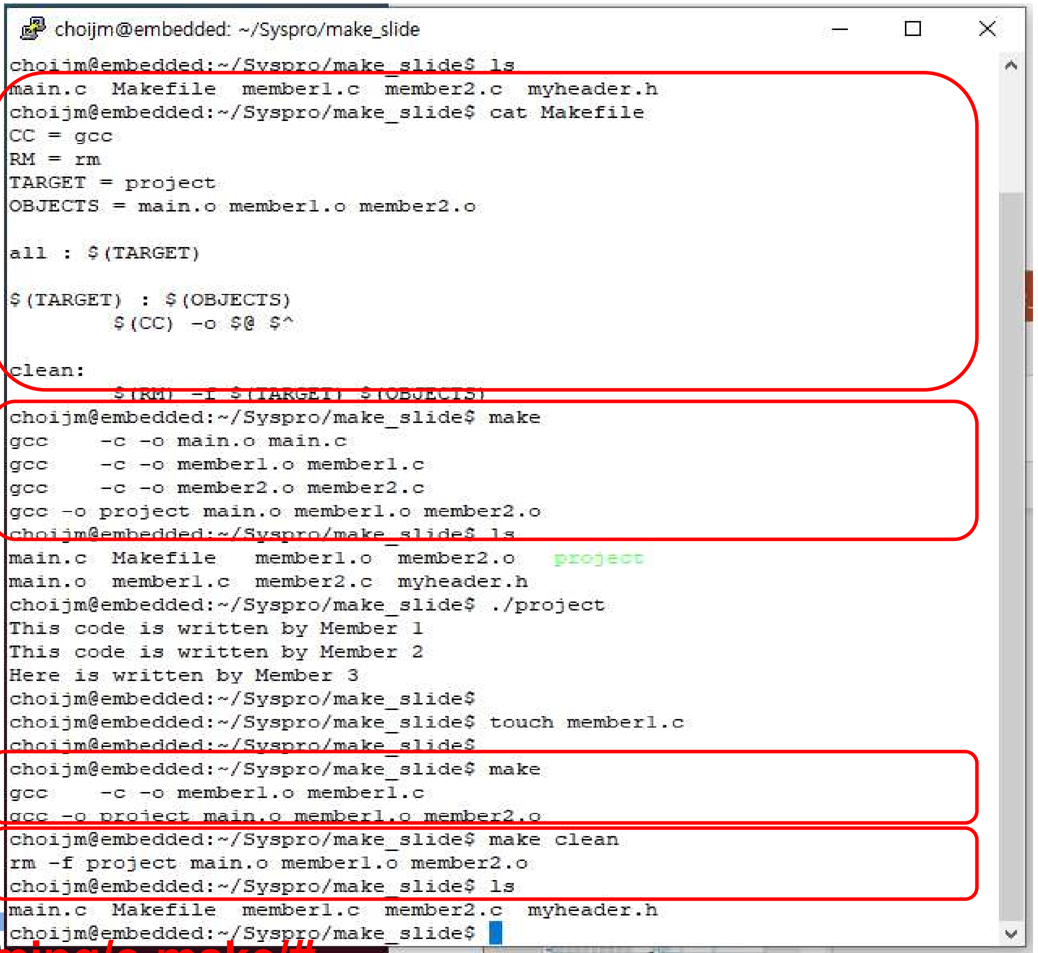
# How to make and run a program in Linux (7/7)

- Make utility
  - ✓ Why? Using multiple files ➔ 1) complex gcc command, 2) dependency
  - ✓ Makefile format
  - ✓ Makefile example

```
target : dependency1 dependency2
         command1
         command2
```



```
choijm@embedded:~/Syspro/make_slide$ ls
main.c  member1.c  member2.c  myheader.h
choijm@embedded:~/Syspro/make_slide$ cat member1.c
void func1(void)
{
                printf("This code is written by Member 1\n");
}
choijm@embedded:~/Syspro/make_slide$ cat member2.c
void func2(void)
{
                printf("This code is written by Member 2\n");
}
choijm@embedded:~/Syspro/make_slide$ cat main.c
#include "myheader.h"

int main()
{
        func1();
        func2();
        printf("Here is written by Member 3\n");
}
choijm@embedded:~/Syspro/make_slide$ cat myheader.h
#include <stdio.h>
void func1(void);
void func2(void);
choijm@embedded:~/Syspro/make_slide$ gcc -o project main.c member1.c member2
.c
choijm@embedded:~/Syspro/make_slide$ ./project
This code is written by Member 1
This code is written by Member 2
Here is written by Member 3
choijm@embedded:~/Syspro/make_slide$
choijm@embedded:~/Syspro/make_slide$ touch member1.c
choijm@embedded:~/Syspro/make_slide$ gcc -o project main.c member1.c member2
.c
choijm@embedded:~/Syspro/make_slide$
```



```
choijm@embedded:~/Syspro/make_slide$ ls
main.c  Makefile  member1.c  member2.c  myheader.h
choijm@embedded:~/Syspro/make_slide$ cat Makefile
CC = gcc
RM = rm
TARGET = project
OBJECTS = main.o member1.o member2.o

all : $(TARGET)

$(TARGET) : $(OBJECTS)
        $(CC) -o $@ $^

clean:
        $(RM) -f $(TARGET) $(OBJECTS)
choijm@embedded:~/Syspro/make_slide$ make
gcc    -c -o main.o main.c
gcc    -c -o member1.o member1.c
gcc    -c -o member2.o member2.c
gcc -o project main.o member1.o member2.o
choijm@embedded:~/Syspro/make_slide$ ls
main.c  Makefile   member1.o  member2.o   project
main.o  member1.c  member2.c  myheader.h
choijm@embedded:~/Syspro/make_slide$ ./project
This code is written by Member 1
This code is written by Member 2
Here is written by Member 3
choijm@embedded:~/Syspro/make_slide$
choijm@embedded:~/Syspro/make_slide$ touch member1.c
choijm@embedded:~/Syspro/make_slide$
choijm@embedded:~/Syspro/make_slide$ make
gcc    -c -o member1.o member1.c
gcc -o project main.o member1.o member2.o
choijm@embedded:~/Syspro/make_slide$ make clean
rm -f project main.o member1.o member2.o
choijm@embedded:~/Syspro/make_slide$ ls
main.c  Makefile  member1.c  member2.c  myheader.h
choijm@embedded:~/Syspro/make_slide$
```

☞See https://losskatsu.github.io/programming/c-make/#

## Summary

- Discuss the features of Linux
- Understand the commands related to file and process
- Explore the language hierarchy in Linux (UNIX)

☞ **Homework 2.**

**1.1 Make a file using vi editor that contains your favorite poem**

**1.2 Make a snapshot that**

**- has at least 10 commands (e.g. ls –l, ps, pipe, redirection, …)**
**including compilation practice (e.g. gcc, as, gdb, …)**

**- shows student's ID and date (using whoami and date)**

**- Server IP: 220.149.236.4 (recommended) or 220.149.236.2**

**1.3 Write a report**

**- 1) Introduction: What to do, How, …**

**- 2) Snapshot for 1.1,**

**- 3) Snapshot for 1.2,**

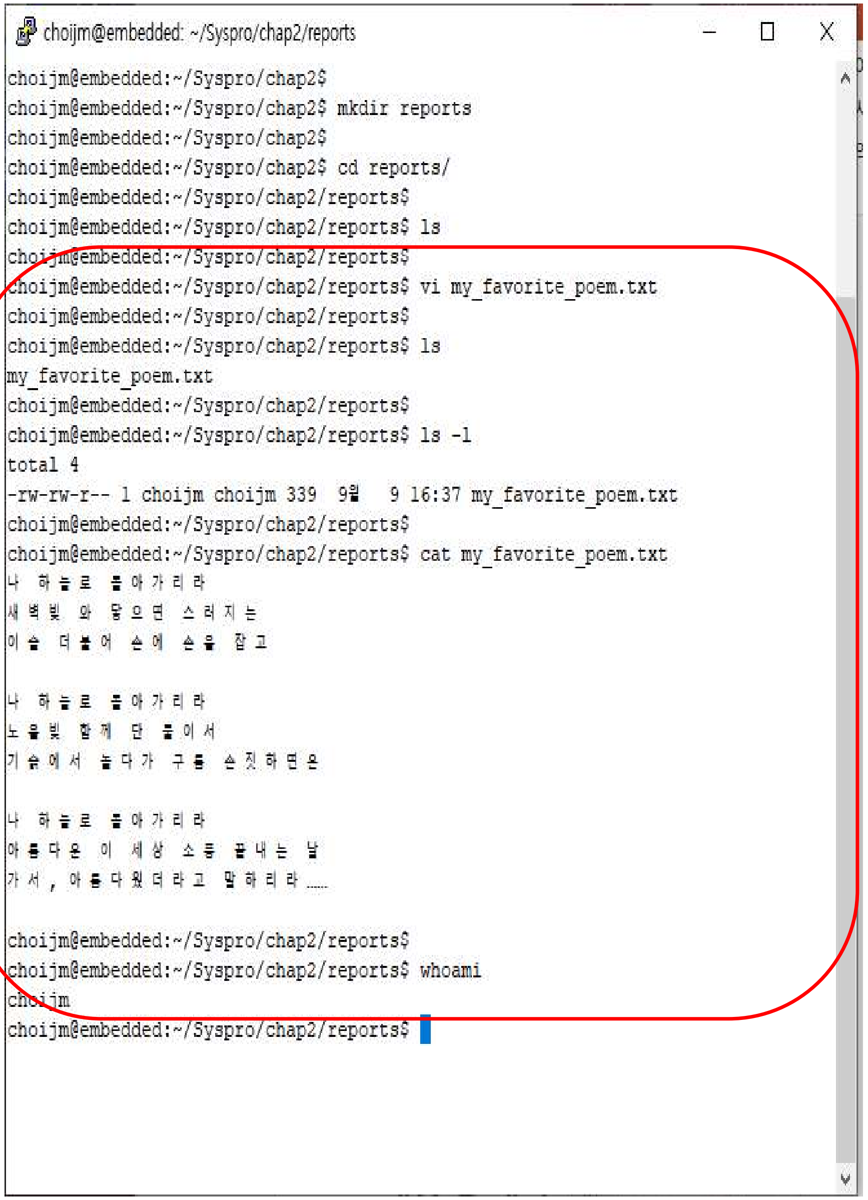
**- 4) Discussion: what you learn, issues, …**

**1.4 Deadline: Next week (same time)**

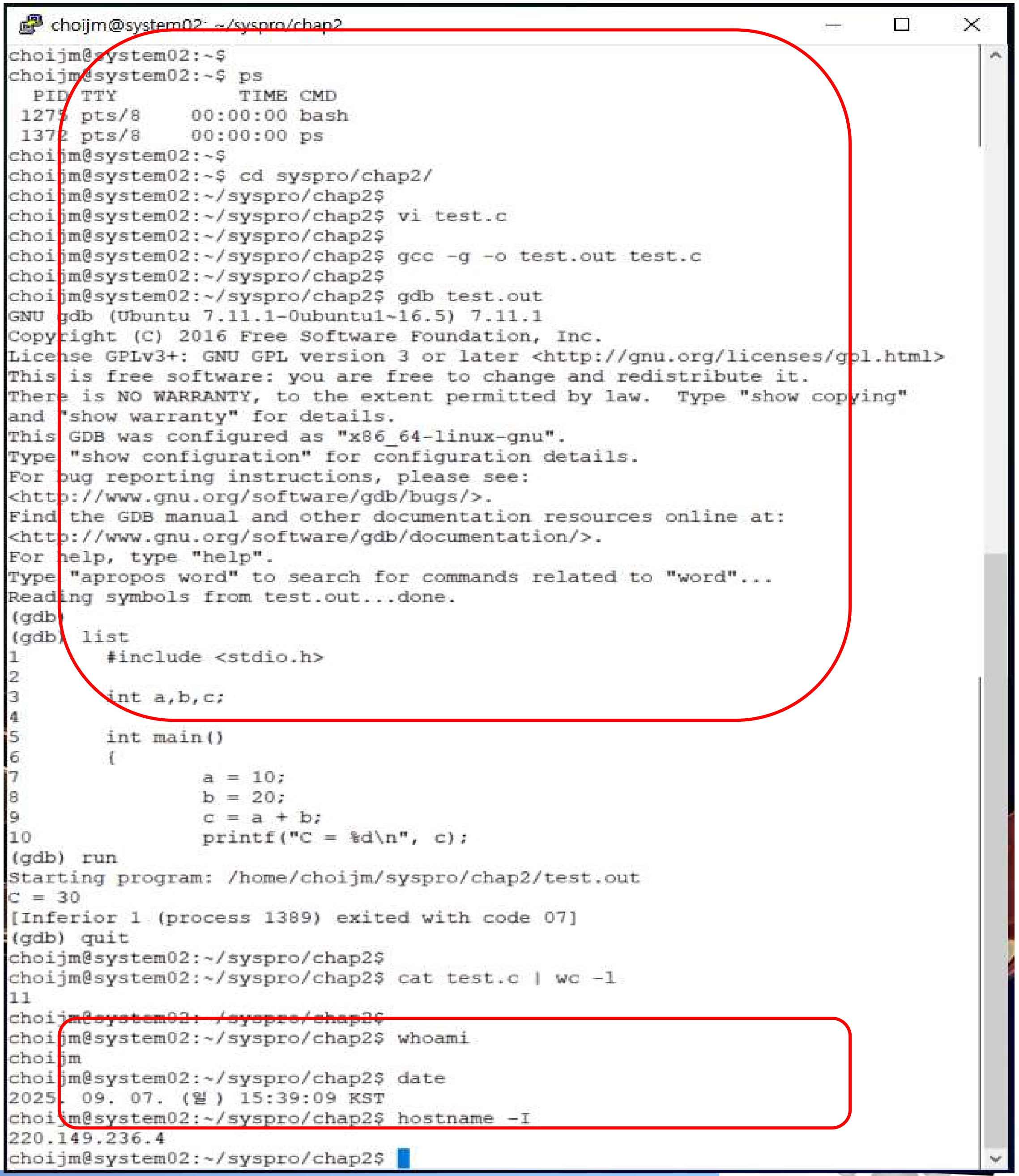
**1.5 How to submit? Email to choiyg@dankook.ac.kr**

# Appendix 1. Snapshot Example

- Example


choijm@embedded: ~/Syspro/chap2/reports

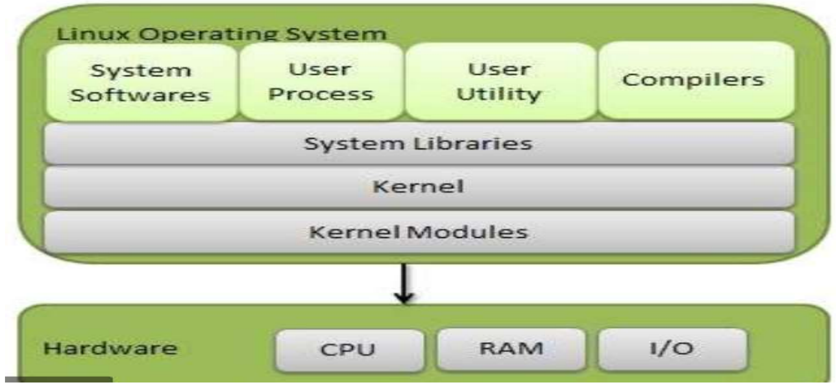
```
choijm@embedded:~/Syspro/chap2$
choijm@embedded:~/Syspro/chap2$ mkdir reports
choijm@embedded:~/Syspro/chap2$
choijm@embedded:~/Syspro/chap2$ cd reports/
choijm@embedded:~/Syspro/chap2/reports$
choijm@embedded:~/Syspro/chap2/reports$ ls
choijm@embedded:~/Syspro/chap2/reports$
choijm@embedded:~/Syspro/chap2/reports$ vi my_favorite_poem.txt
choijm@embedded:~/Syspro/chap2/reports$
choijm@embedded:~/Syspro/chap2/reports$ ls
my_favorite_poem.txt
choijm@embedded:~/Syspro/chap2/reports$
choijm@embedded:~/Syspro/chap2/reports$ ls -l
total 4
-rw-rw-r-- 1 choijm choijm 339  9월   9 16:37 my_favorite_poem.txt
choijm@embedded:~/Syspro/chap2/reports$
choijm@embedded:~/Syspro/chap2/reports$ cat my_favorite_poem.txt
나 하늘로 돌아가리라
새벽빛 와 닿으면 스러지는
이슬 더불어 손에 손을 잡고

나 하늘로 돌아가리라
노을빛 함께 단 둘이서
기슭에서 놀다가 구름 손짓하면은

나 하늘로 돌아가리라
아름다운 이 세상 소풍 끝내는 날
가서, 아름다웠더라고 말하리라.....

choijm@embedded:~/Syspro/chap2/reports$
choijm@embedded:~/Syspro/chap2/reports$ whoami
choijm
choijm@embedded:~/Syspro/chap2/reports$
```

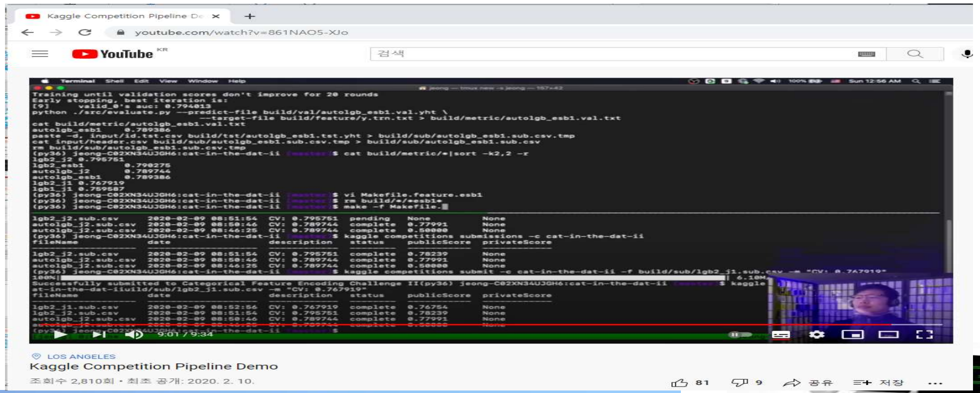

choijm@system02: ~/syspro/chap2

```
choijm@system02:~$
choijm@system02:~$ ps
  PID TTY          TIME CMD
 1275 pts/8    00:00:00 bash
 1372 pts/8    00:00:00 ps
choijm@system02:~$
choijm@system02:~$ cd syspro/chap2/
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ vi test.c
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ gcc -g -o test.out test.c
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ gdb test.out
GNU gdb (Ubuntu 7.11.1-0ubuntu1~16.5) 7.11.1
Copyright (C) 2016 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from test.out...done.
(gdb)
(gdb) list
1       #include <stdio.h>
2
3       int a,b,c;
4
5       int main()
6       {
7               a = 10;
8               b = 20;
9               c = a + b;
10              printf("C = %d\n", c);
(gdb) run
Starting program: /home/choijm/syspro/chap2/test.out
C = 30
[Inferior 1 (process 1389) exited with code 07]
(gdb) quit
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ cat test.c | wc -l
11
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ whoami
choijm
choijm@system02:~/syspro/chap2$ date
2025. 09. 07. (월) 15:39:09 KST
choijm@system02:~/syspro/chap2$ hostname -I
220.149.236.4
choijm@system02:~/syspro/chap2$
```

## Quiz for this Lecture

- Quiz
  - ✓ 1. Classify UNIX-like OSes into 5 categories.
  - ✓ 2. Discuss the difference between OS (Operating System) and Kernel using the below left figure.
  - ✓ 3. Explain differences between “$ls .” and “$ls ..”. Also, explain differences between “ls” and “ls –l”.
  - ✓ 4. What is the background music in “Dr Jeong-Joon Lee’s Kaggle Demo”? What commands can you find in the Demo? (at least 5 that you have learned in the LN2.)
  - ✓ 5. Discuss three different modes in the vi editor.
  - ✓ 6. What are the roles of “break” and “step” command in gdb?



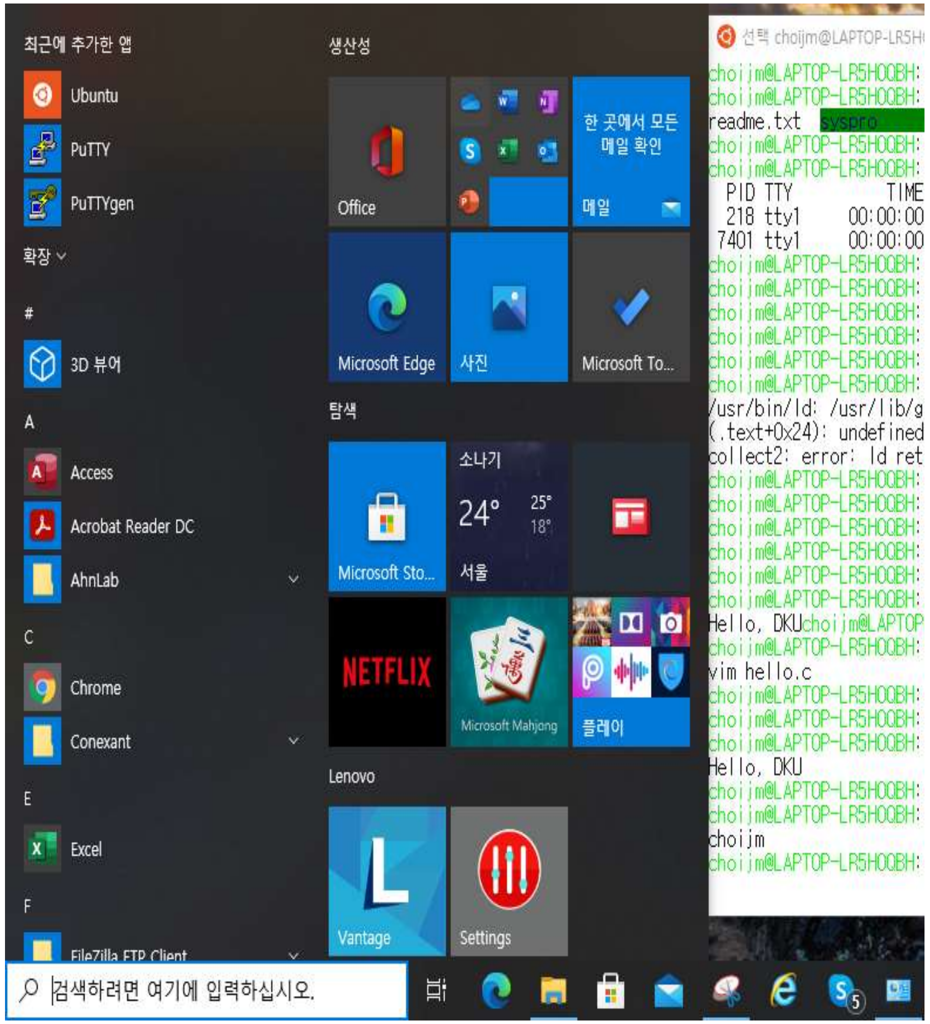

Kaggle Competition Pipeline De
youtube.com/watch?v=861NAO5-XJo
YouTube KR
검색
Terminal Shell Edit View Window Help

```
Training until validation scores don't improve for 20 rounds
Early stopping, best iteration is:
[9]     valid_0's auc: 0.794013
python ./src/evaluate.py --predict-file build/val/autolgb_esb1.val.yht \
                         --target-file build/feature/y.trn.txt > build/metric/autolgb_esb1.val.txt
cat build/metric/autolgb_esb1.val.txt
autolgb_esb1     0.789386
paste -d, input/id.tst.csv build/tst/autolgb_esb1.tst.yht > build/sub/autolgb_esb1.sub.csv.tmp
cat input/header.csv build/sub/autolgb_esb1.sub.csv.tmp > build/sub/autolgb_esb1.sub.csv
rm build/sub/autolgb_esb1.sub.csv.tmp
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii (master)$ cat build/metric/*|sort -k2,2 -r
lgb2_j2 0.795751
lgb2_esb1        0.790275
autolgb_j2       0.789744
autolgb_esb1     0.789386
lgb2_j1 0.767919
lgb1_j1 0.759587
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii (master)$ vi Makefile.feature.esb1
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii (master)$ rm build/*/*esb1*
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii (master)$ make -f Makefile.

lgb2_j2.sub.csv      2020-02-09 08:51:54  CV: 0.795751  pending   None     None
autolgb_j2.sub.csv   2020-02-09 08:50:46  CV: 0.789744  complete  0.77991  None
autolgb_j2.sub.csv   2020-02-09 08:46:25  CV: 0.789744  complete  0.50000  None
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii (master)$ kaggle competitions submissions -c cat-in-the-dat-ii
fileName             date                 description   status    publicScore  privateScore
-------------------  -------------------  ------------  --------  -----------  ------------
lgb2_j2.sub.csv      2020-02-09 08:51:54  CV: 0.795751  complete  0.78239      None
autolgb_j2.sub.csv   2020-02-09 08:50:46  CV: 0.789744  complete  0.77991      None
autolgb_j2.sub.csv   2020-02-09 08:46:25  CV: 0.789744  complete  0.50000      None
(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii (master)$ kaggle competitions submit -c cat-in-the-dat-ii -f build/sub/lgb2_j1.sub.csv -m "CV: 0.767919"
100%|████████████████████████████████████████| 6.10M
Successfully submitted to Categorical Feature Encoding Challenge II(py36) jeong-C02XN34UJGH6:cat-in-the-dat-ii (master)$ kaggle
at-in-the-dat-iiuild/sub/lgb2_j1.sub.csv -m "CV: 0.767919"
fileName             date                 description   status    publicScore  privateScore
-------------------  -------------------  ------------  --------  -----------  ------------
lgb2_j1.sub.csv      2020-02-09 08:52:56  CV: 0.767919  complete  0.76754      None
lgb2_j2.sub.csv      2020-02-09 08:51:54  CV: 0.795751  complete  0.78239      None
autolgb_j2.sub.csv   2020-02-09 08:50:46  CV: 0.789744  complete  0.77991      None
```

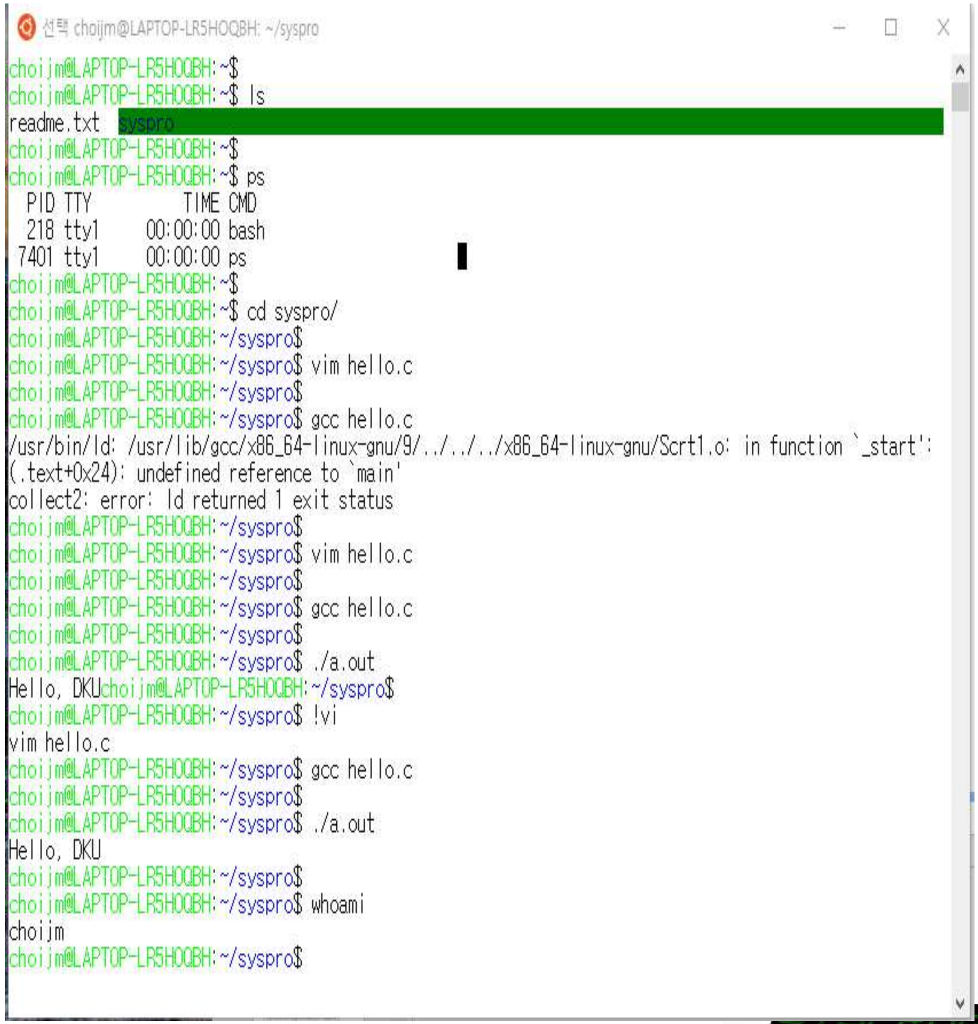
LOS ANGELES
Kaggle Competition Pipeline Demo
조회수 2,810회 • 최초 공개: 2020. 2. 10.

# Appendix 1: How to access Linux: Alternative

- WSL (Windows Subsystem for Linux)
  - ✓ A compatibility layer for running Linux binary executables (in ELF format) natively on Windows OS



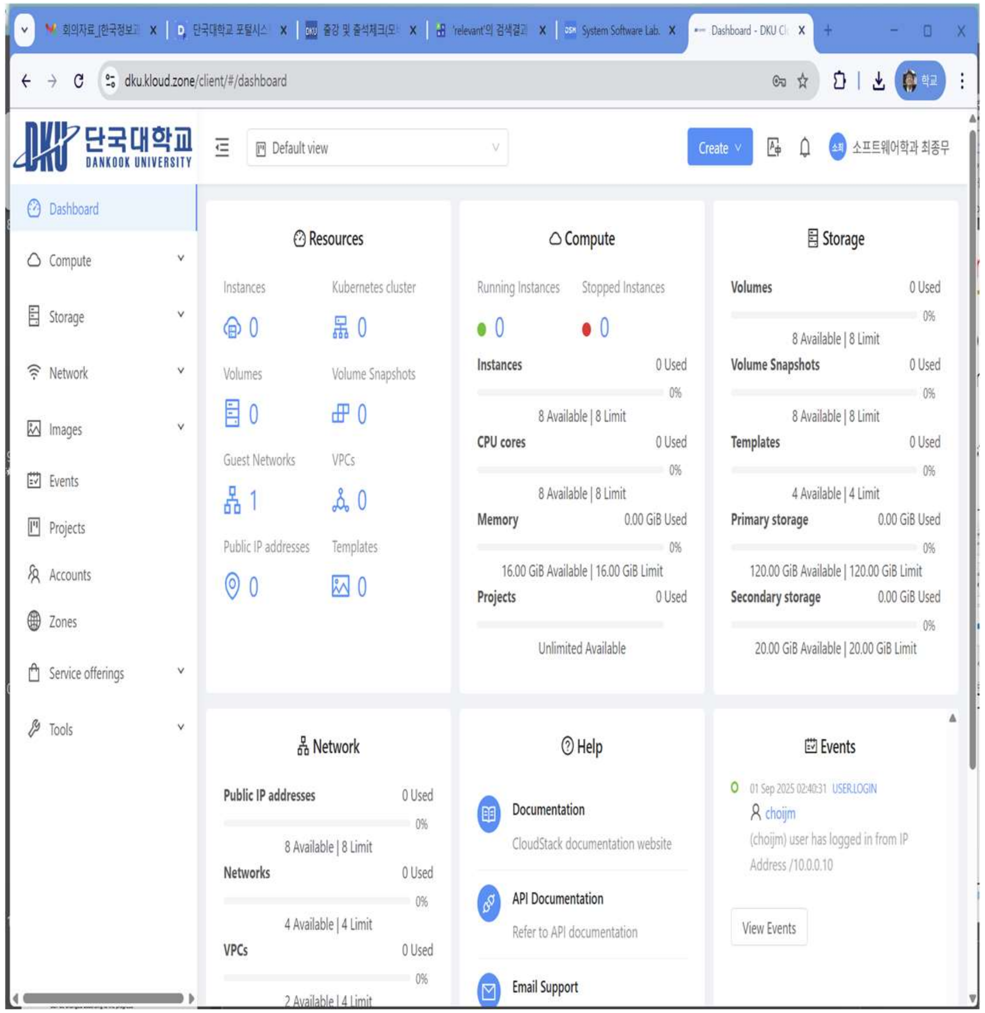
```
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
readme.txt  syspro
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
  PID TTY          TIME
  218 tty1     00:00:00
 7401 tty1     00:00:00
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
/usr/bin/ld: /usr/lib/g
(.text+0x24): undefined
collect2: error: ld ret
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
Hello, DKUchoijm@LAPTOP
choijm@LAPTOP-LR5HOQBH:
vim hello.c
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
Hello, DKU
choijm@LAPTOP-LR5HOQBH:
choijm@LAPTOP-LR5HOQBH:
choijm
choijm@LAPTOP-LR5HOQBH:
```



```
choijm@LAPTOP-LR5HOQBH:~$
choijm@LAPTOP-LR5HOQBH:~$ ls
readme.txt  syspro
choijm@LAPTOP-LR5HOQBH:~$
choijm@LAPTOP-LR5HOQBH:~$ ps
  PID TTY          TIME CMD
  218 tty1     00:00:00 bash
 7401 tty1     00:00:00 ps
choijm@LAPTOP-LR5HOQBH:~$
choijm@LAPTOP-LR5HOQBH:~$ cd syspro/
choijm@LAPTOP-LR5HOQBH:~/syspro$
choijm@LAPTOP-LR5HOQBH:~/syspro$ vim hello.c
choijm@LAPTOP-LR5HOQBH:~/syspro$
choijm@LAPTOP-LR5HOQBH:~/syspro$ gcc hello.c
/usr/bin/ld: /usr/lib/gcc/x86_64-linux-gnu/9/../../../x86_64-linux-gnu/Scrt1.o: in function `_start':
(.text+0x24): undefined reference to `main'
collect2: error: ld returned 1 exit status
choijm@LAPTOP-LR5HOQBH:~/syspro$
choijm@LAPTOP-LR5HOQBH:~/syspro$ vim hello.c
choijm@LAPTOP-LR5HOQBH:~/syspro$
choijm@LAPTOP-LR5HOQBH:~/syspro$ gcc hello.c
choijm@LAPTOP-LR5HOQBH:~/syspro$
choijm@LAPTOP-LR5HOQBH:~/syspro$ ./a.out
Hello, DKUchoijm@LAPTOP-LR5HOQBH:~/syspro$
choijm@LAPTOP-LR5HOQBH:~/syspro$ !vi
vim hello.c
choijm@LAPTOP-LR5HOQBH:~/syspro$ gcc hello.c
choijm@LAPTOP-LR5HOQBH:~/syspro$
choijm@LAPTOP-LR5HOQBH:~/syspro$ ./a.out
Hello, DKU
choijm@LAPTOP-LR5HOQBH:~/syspro$
choijm@LAPTOP-LR5HOQBH:~/syspro$ whoami
choijm
choijm@LAPTOP-LR5HOQBH:~/syspro$
```

## Appendix 1: How to access Linux: Alternative

- SOLID Cloud (or Amazon EC2 or Google or Toast Cloud)
  - ✓ Supported by Dankook Univ. (like Amazon EC2 or NHN Toast)

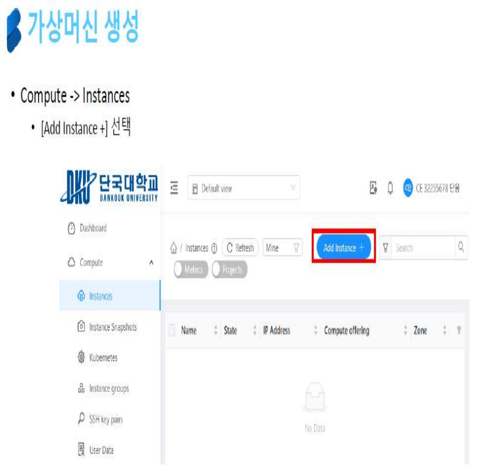

dku.kloud.zone/client/#/dashboard

| Resources | | Compute | | Storage | |
|---|---|---|---|---|---|
| Instances 0 | Kubernetes cluster 0 | Running Instances 0 | Stopped Instances 0 | Volumes | 0 Used, 0%, 8 Available \| 8 Limit |
| Volumes 0 | Volume Snapshots 0 | Instances | 0 Used, 0%, 8 Available \| 8 Limit | Volume Snapshots | 0 Used, 0%, 8 Available \| 8 Limit |
| Guest Networks 1 | VPCs 0 | CPU cores | 0 Used, 0%, 8 Available \| 8 Limit | Templates | 0 Used, 0%, 4 Available \| 4 Limit |
| Public IP addresses 0 | Templates 0 | Memory | 0.00 GiB Used, 0%, 16.00 GiB Available \| 16.00 GiB Limit | Primary storage | 0.00 GiB Used, 0%, 120.00 GiB Available \| 120.00 GiB Limit |
| | | Projects | 0 Used, Unlimited Available | Secondary storage | 0.00 GiB Used, 0%, 20.00 GiB Available \| 20.00 GiB Limit |

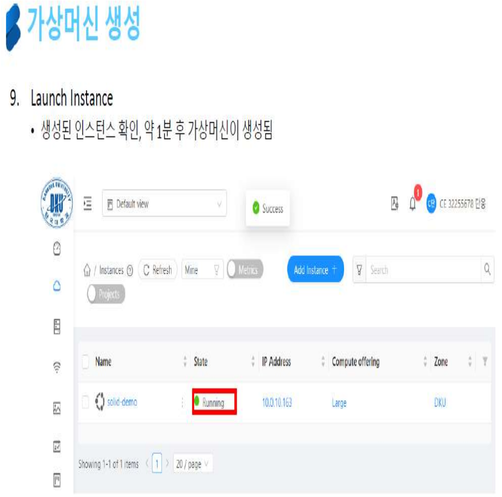
Network: Public IP addresses 0 Used, 0%, 8 Available | 8 Limit; Networks 0 Used, 0%, 4 Available | 4 Limit; VPCs 0 Used, 0%, 2 Available | 4 Limit

Help: Documentation — CloudStack documentation website; API Documentation — Refer to API documentation; Email Support

Events: 01 Sep 2025 02:40:31 USER.LOGIN choijm (choijm) user has logged in from IP Address /10.0.0.10 — View Events

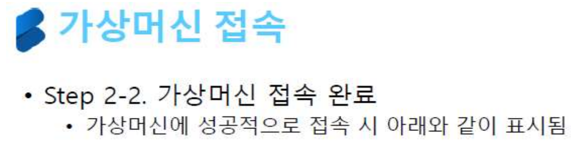

가상머신 생성

- Compute -> Instances
  - [Add Instance +] 선택

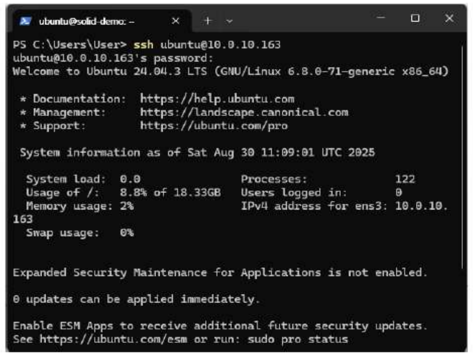

가상머신 생성

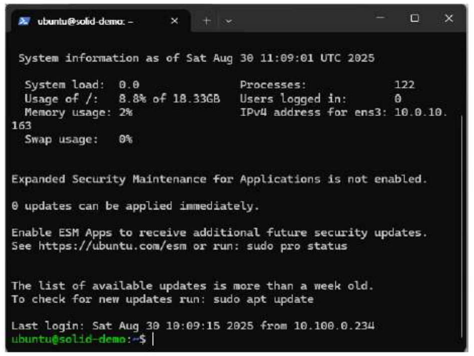
9. Launch Instance
   - 생성된 인스턴스 확인, 약 1분 후 가상머신이 생성됨

| Name | State | IP Address | Compute offering | Zone |
|---|---|---|---|---|
| solid-demo | Running | 10.0.10.163 | Large | DKU |


가상머신 접속

- Step 2-2. 가상머신 접속 완료
  - 가상머신에 성공적으로 접속 시 아래와 같이 표시됨

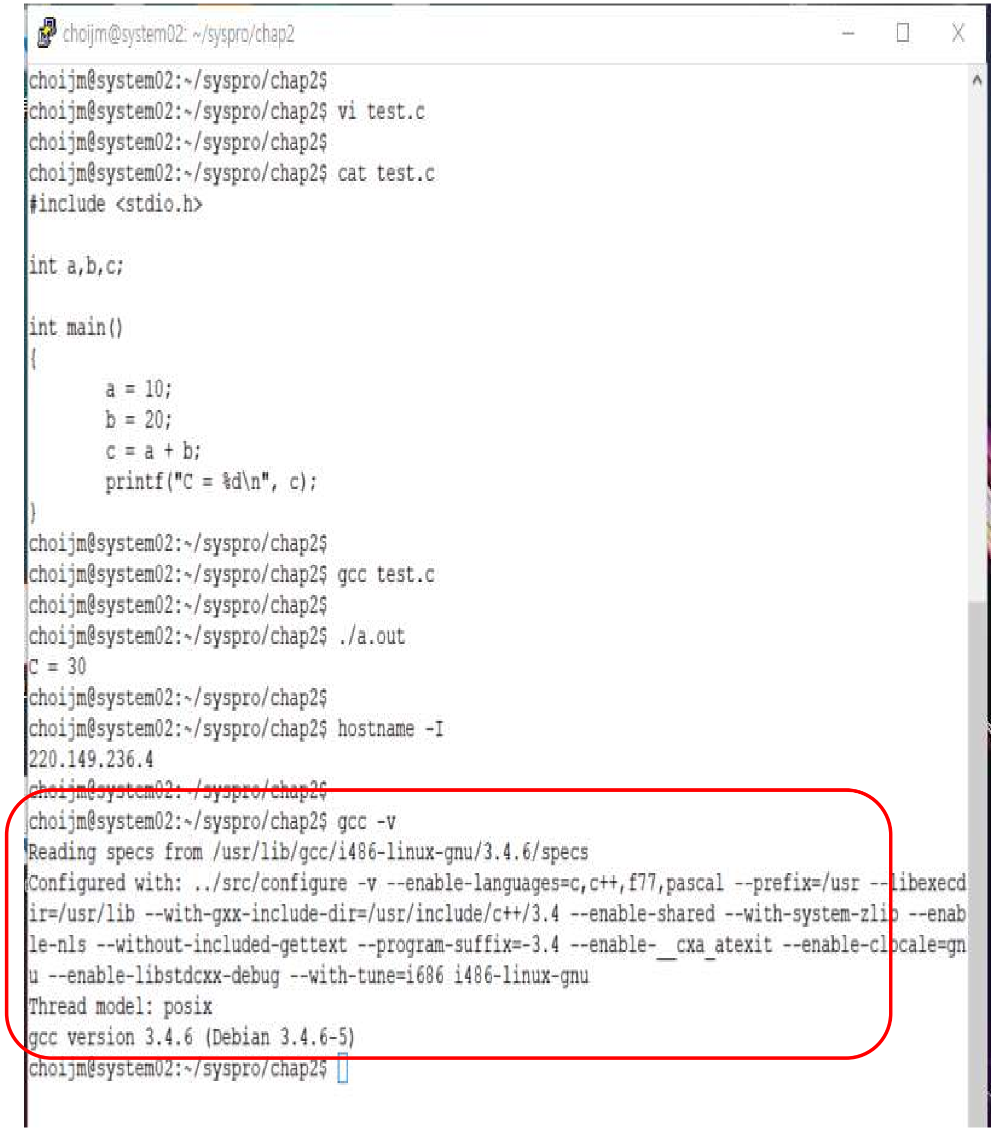

```
PS C:\Users\User> ssh ubuntu@10.0.10.163
ubuntu@10.0.10.163's password:
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-71-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Sat Aug 30 11:09:01 UTC 2025

  System load:  0.0               Processes:             122
  Usage of /:   8.8% of 18.33GB   Users logged in:       0
  Memory usage: 2%                IPv4 address for ens3: 10.0.10.
163
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status
```


```
 System information as of Sat Aug 30 11:09:01 UTC 2025

  System load:  0.0               Processes:             122
  Usage of /:   8.8% of 18.33GB   Users logged in:       0
  Memory usage: 2%                IPv4 address for ens3: 10.0.10.
163
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sat Aug 30 10:09:15 2025 from 10.100.0.234
ubuntu@solid-demo:~$
```

**(Source: 남재현 교수님, SOLID CLOUD 사용자 설명서 및 접속 가이드)**

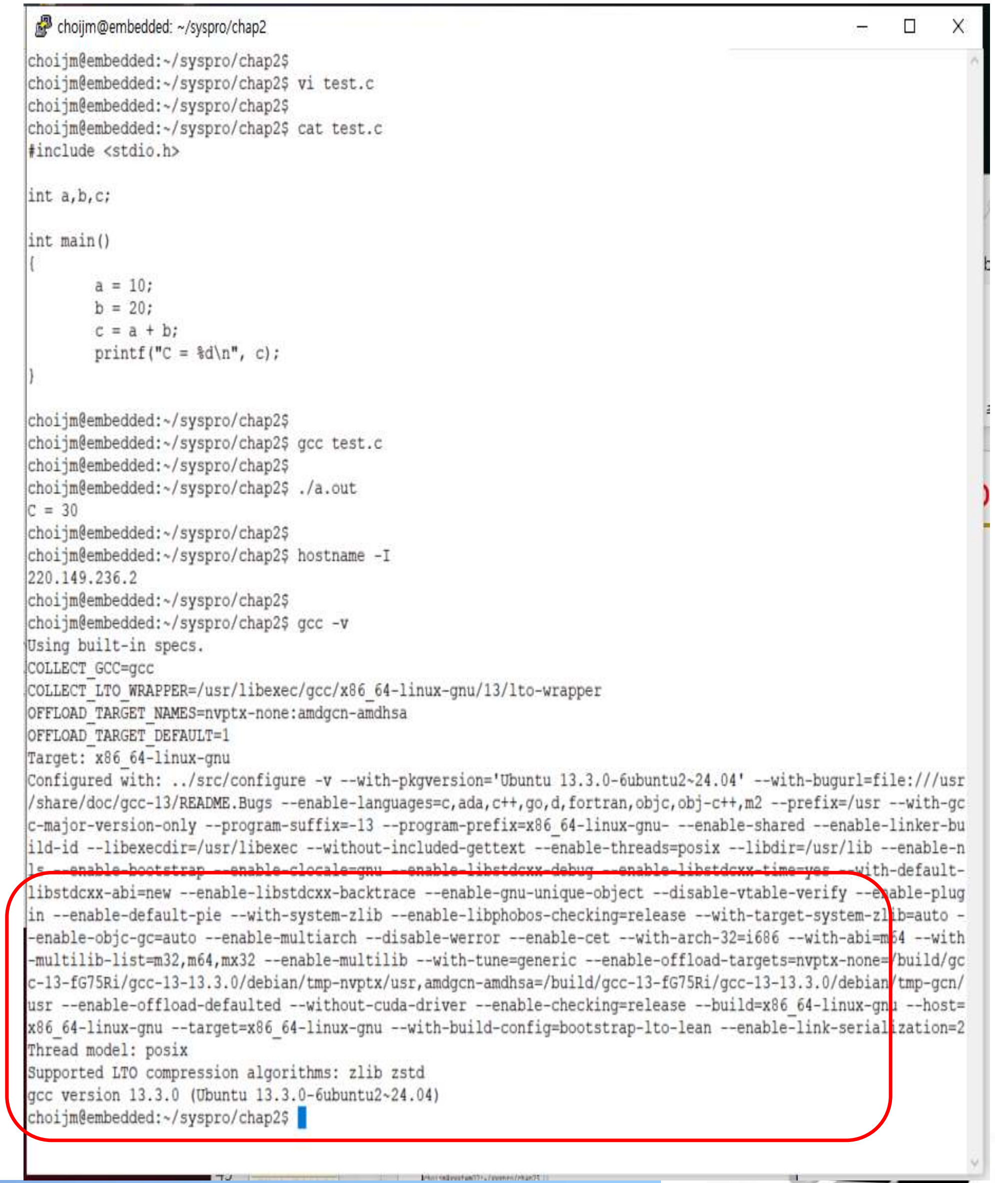
## Appendix 2: Effect of different compilers

- Application programmer's viewpoint



choijm@system02: ~/syspro/chap2

```
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ vi test.c
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ cat test.c
#include <stdio.h>

int a,b,c;

int main()
{
        a = 10;
        b = 20;
        c = a + b;
        printf("C = %d\n", c);
}
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ gcc test.c
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ ./a.out
C = 30
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ hostname -I
220.149.236.4
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ gcc -v
Reading specs from /usr/lib/gcc/i486-linux-gnu/3.4.6/specs
Configured with: ../src/configure -v --enable-languages=c,c++,f77,pascal --prefix=/usr --libexecd
ir=/usr/lib --with-gxx-include-dir=/usr/include/c++/3.4 --enable-shared --with-system-zlib --enab
le-nls --without-included-gettext --program-suffix=-3.4 --enable-__cxa_atexit --enable-clocale=gn
u --enable-libstdcxx-debug --with-tune=i686 i486-linux-gnu
Thread model: posix
gcc version 3.4.6 (Debian 3.4.6-5)
choijm@system02:~/syspro/chap2$
```

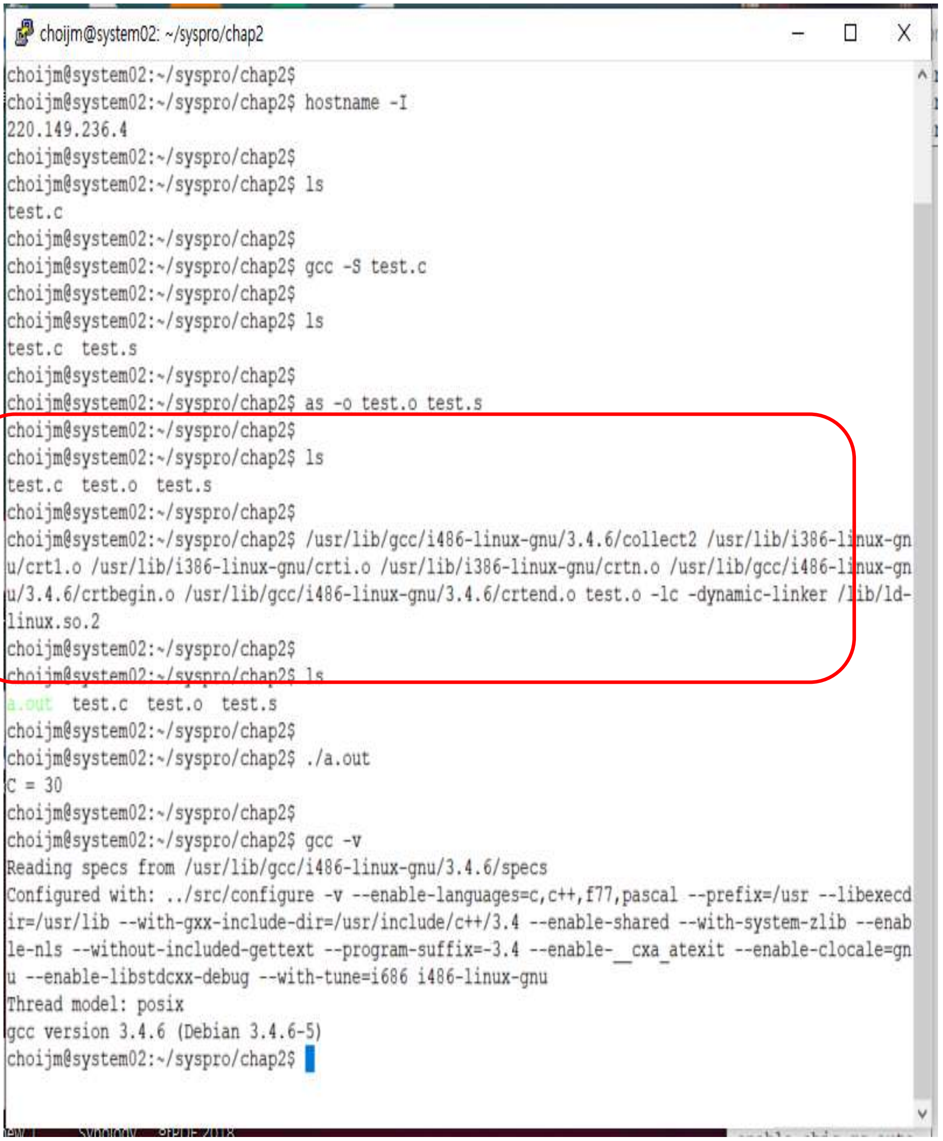


choijm@embedded: ~/syspro/chap2

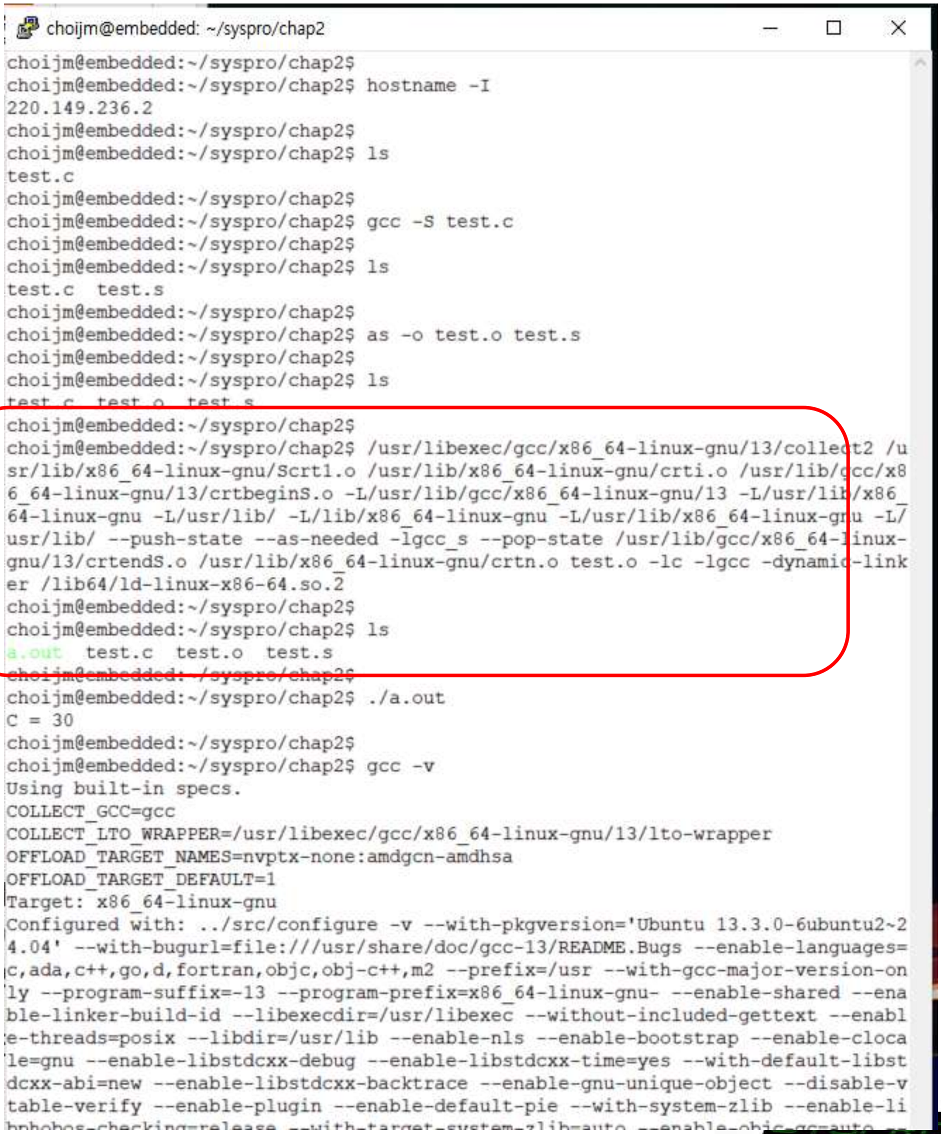
```
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ vi test.c
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ cat test.c
#include <stdio.h>

int a,b,c;

int main()
{
        a = 10;
        b = 20;
        c = a + b;
        printf("C = %d\n", c);
}

choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ gcc test.c
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ ./a.out
C = 30
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ hostname -I
220.149.236.2
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/x86_64-linux-gnu/13/lto-wrapper
OFFLOAD_TARGET_NAMES=nvptx-none:amdgcn-amdhsa
OFFLOAD_TARGET_DEFAULT=1
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 13.3.0-6ubuntu2~24.04' --with-bugurl=file:///usr
/share/doc/gcc-13/README.Bugs --enable-languages=c,ada,c++,go,d,fortran,objc,obj-c++,m2 --prefix=/usr --with-gc
c-major-version-only --program-suffix=-13 --program-prefix=x86_64-linux-gnu- --enable-shared --enable-linker-bu
ild-id --libexecdir=/usr/libexec --without-included-gettext --enable-threads=posix --libdir=/usr/lib --enable-n
ls --enable-bootstrap --enable-clocale=gnu --enable-libstdcxx-debug --enable-libstdcxx-time=yes --with-default-
libstdcxx-abi=new --enable-libstdcxx-backtrace --enable-gnu-unique-object --disable-vtable-verify --enable-plug
in --enable-default-pie --with-system-zlib --enable-libphobos-checking=release --with-target-system-zlib=auto -
-enable-objc-gc=auto --enable-multiarch --disable-werror --enable-cet --with-arch-32=i686 --with-abi=m64 --with
-multilib-list=m32,m64,mx32 --enable-multilib --with-tune=generic --enable-offload-targets=nvptx-none=/build/gc
c-13-fG75Ri/gcc-13-13.3.0/debian/tmp-nvptx/usr,amdgcn-amdhsa=/build/gcc-13-fG75Ri/gcc-13-13.3.0/debian/tmp-gcn/
usr --enable-offload-defaulted --without-cuda-driver --enable-checking=release --build=x86_64-linux-gnu --host=
x86_64-linux-gnu --target=x86_64-linux-gnu --with-build-config=bootstrap-lto-lean --enable-link-serialization=2
Thread model: posix
Supported LTO compression algorithms: zlib zstd
gcc version 13.3.0 (Ubuntu 13.3.0-6ubuntu2~24.04)
choijm@embedded:~/syspro/chap2$
```

# Appendix 2: Effect of different compilers

- System programmer's viewpoint



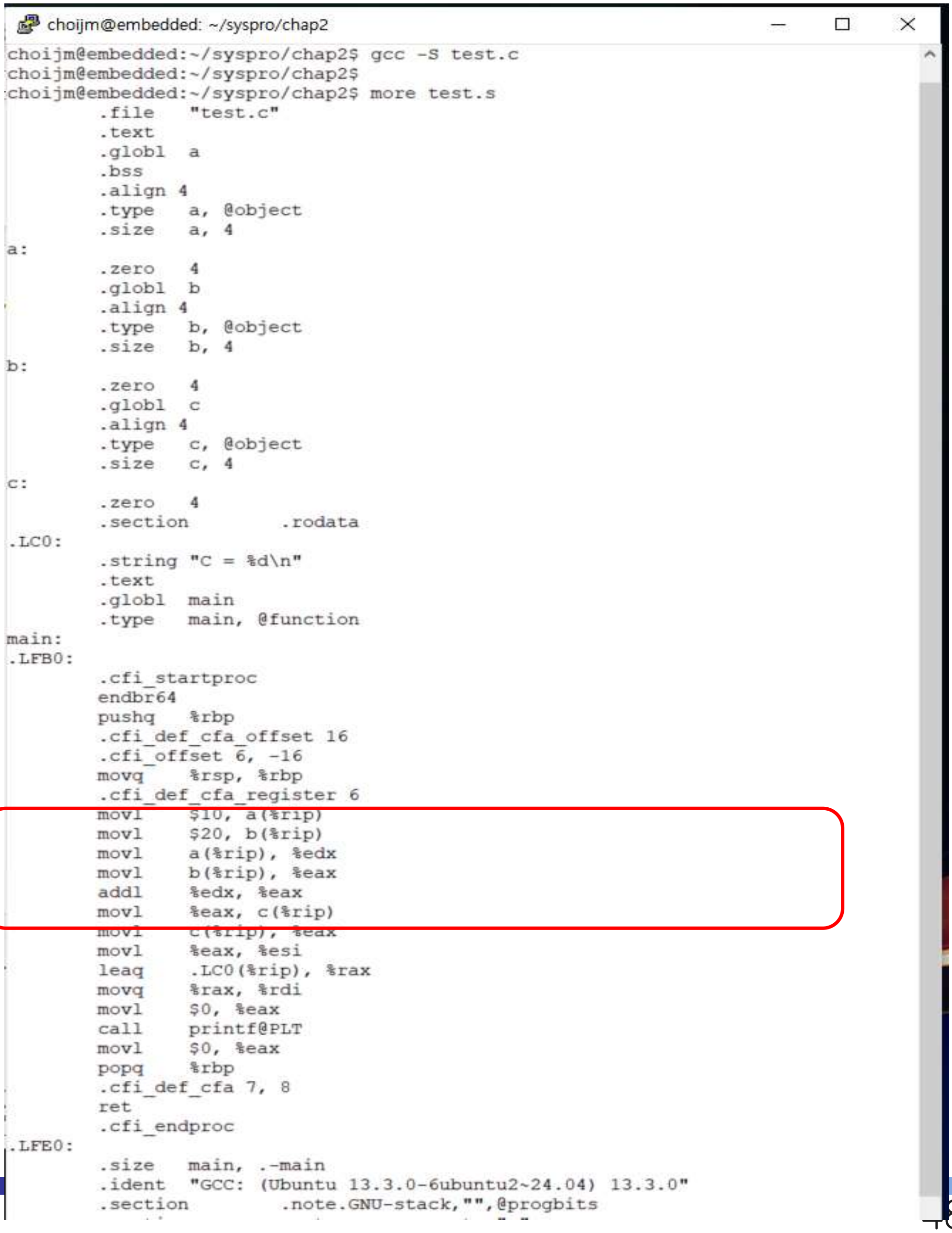
```
choijm@system02: ~/syspro/chap2
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ hostname -I
220.149.236.4
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ ls
test.c
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ gcc -S test.c
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ ls
test.c  test.s
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ as -o test.o test.s
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ ls
test.c  test.o  test.s
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ /usr/lib/gcc/i486-linux-gnu/3.4.6/collect2 /usr/lib/i386-linux-gn
u/crt1.o /usr/lib/i386-linux-gnu/crti.o /usr/lib/i386-linux-gnu/crtn.o /usr/lib/gcc/i486-linux-gn
u/3.4.6/crtbegin.o /usr/lib/gcc/i486-linux-gnu/3.4.6/crtend.o test.o -lc -dynamic-linker /lib/ld-
linux.so.2
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ ls
a.out  test.c  test.o  test.s
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ ./a.out
C = 30
choijm@system02:~/syspro/chap2$
choijm@system02:~/syspro/chap2$ gcc -v
Reading specs from /usr/lib/gcc/i486-linux-gnu/3.4.6/specs
Configured with: ../src/configure -v --enable-languages=c,c++,f77,pascal --prefix=/usr --libexecd
ir=/usr/lib --with-gxx-include-dir=/usr/include/c++/3.4 --enable-shared --with-system-zlib --enab
le-nls --without-included-gettext --program-suffix=-3.4 --enable-__cxa_atexit --enable-clocale=gn
u --enable-libstdcxx-debug --with-tune=i686 i486-linux-gnu
Thread model: posix
gcc version 3.4.6 (Debian 3.4.6-5)
choijm@system02:~/syspro/chap2$
```



```
choijm@embedded: ~/syspro/chap2
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ hostname -I
220.149.236.2
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ ls
test.c
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ gcc -S test.c
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ ls
test.c  test.s
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ as -o test.o test.s
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ ls
test.c  test.o  test.s
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ /usr/libexec/gcc/x86_64-linux-gnu/13/collect2 /u
sr/lib/x86_64-linux-gnu/Scrt1.o /usr/lib/x86_64-linux-gnu/crti.o /usr/lib/gcc/x8
6_64-linux-gnu/13/crtbeginS.o -L/usr/lib/gcc/x86_64-linux-gnu/13 -L/usr/lib/x86_
64-linux-gnu -L/usr/lib/ -L/lib/x86_64-linux-gnu -L/usr/lib/x86_64-linux-gnu -L/
usr/lib/ --push-state --as-needed -lgcc_s --pop-state /usr/lib/gcc/x86_64-linux-
gnu/13/crtendS.o /usr/lib/x86_64-linux-gnu/crtn.o test.o -lc -lgcc -dynamic-link
er /lib64/ld-linux-x86-64.so.2
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ ls
a.out  test.c  test.o  test.s
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ ./a.out
C = 30
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/x86_64-linux-gnu/13/lto-wrapper
OFFLOAD_TARGET_NAMES=nvptx-none:amdgcn-amdhsa
OFFLOAD_TARGET_DEFAULT=1
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 13.3.0-6ubuntu2~2
4.04' --with-bugurl=file:///usr/share/doc/gcc-13/README.Bugs --enable-languages=
c,ada,c++,go,d,fortran,objc,obj-c++,m2 --prefix=/usr --with-gcc-major-version-on
ly --program-suffix=-13 --program-prefix=x86_64-linux-gnu- --enable-shared --ena
ble-linker-build-id --libexecdir=/usr/libexec --without-included-gettext --enabl
e-threads=posix --libdir=/usr/lib --enable-nls --enable-bootstrap --enable-cloca
le=gnu --enable-libstdcxx-debug --enable-libstdcxx-time=yes --with-default-libst
dcxx-abi=new --enable-libstdcxx-backtrace --enable-gnu-unique-object --disable-v
table-verify --enable-plugin --enable-default-pie --with-system-zlib --enable-li
bphobos-checking=release --with-target-system-zlib=auto --enable-objc-gc=auto
```

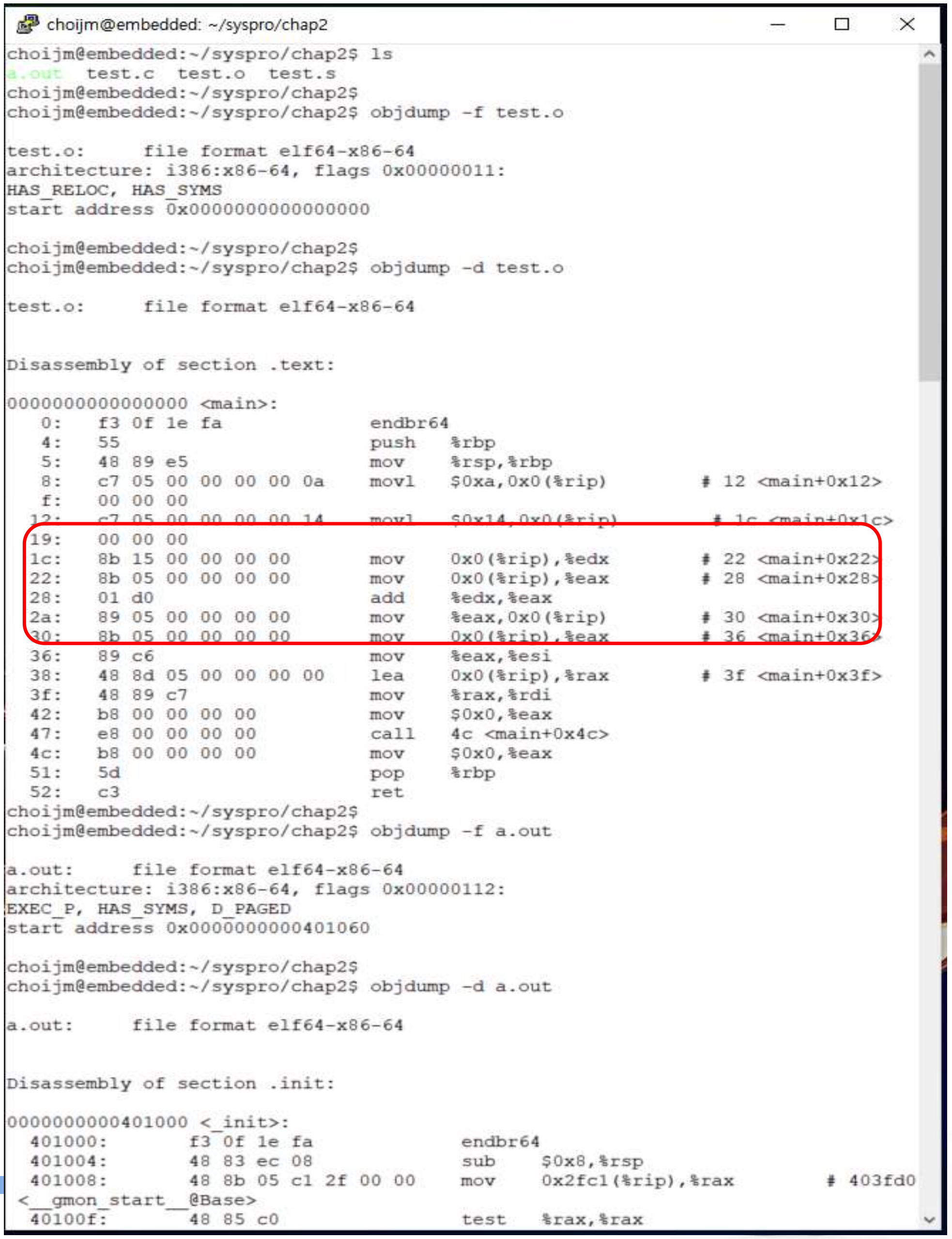
# Appendix 2: Effect of different compilers

- System programmer's viewpoint



choijm@embedded: ~/syspro/chap2

```
choijm@embedded:~/syspro/chap2$ gcc -S test.c
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ more test.s
        .file   "test.c"
        .text
        .globl  a
        .bss
        .align 4
        .type   a, @object
        .size   a, 4
a:
        .zero   4
        .globl  b
        .align 4
        .type   b, @object
        .size   b, 4
b:
        .zero   4
        .globl  c
        .align 4
        .type   c, @object
        .size   c, 4
c:
        .zero   4
        .section        .rodata
.LC0:
        .string "C = %d\n"
        .text
        .globl  main
        .type   main, @function
main:
.LFB0:
        .cfi_startproc
        endbr64
        pushq   %rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
        movq    %rsp, %rbp
        .cfi_def_cfa_register 6
        movl    $10, a(%rip)
        movl    $20, b(%rip)
        movl    a(%rip), %edx
        movl    b(%rip), %eax
        addl    %edx, %eax
        movl    %eax, c(%rip)
        movl    c(%rip), %eax
        movl    %eax, %esi
        leaq    .LC0(%rip), %rax
        movq    %rax, %rdi
        movl    $0, %eax
        call    printf@PLT
        movl    $0, %eax
        popq    %rbp
        .cfi_def_cfa 7, 8
        ret
        .cfi_endproc
.LFE0:
        .size   main, .-main
        .ident  "GCC: (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0"
        .section        .note.GNU-stack,"",@progbits
```



choijm@embedded: ~/syspro/chap2

```
choijm@embedded:~/syspro/chap2$ ls
a.out  test.c  test.o  test.s
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ objdump -f test.o

test.o:     file format elf64-x86-64
architecture: i386:x86-64, flags 0x00000011:
HAS_RELOC, HAS_SYMS
start address 0x0000000000000000

choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ objdump -d test.o

test.o:     file format elf64-x86-64


Disassembly of section .text:

0000000000000000 <main>:
   0:   f3 0f 1e fa             endbr64
   4:   55                      push   %rbp
   5:   48 89 e5                mov    %rsp,%rbp
   8:   c7 05 00 00 00 00 0a    movl   $0xa,0x0(%rip)        # 12 <main+0x12>
   f:   00 00 00
  12:   c7 05 00 00 00 00 14    movl   $0x14,0x0(%rip)       # 1c <main+0x1c>
  19:   00 00 00
  1c:   8b 15 00 00 00 00       mov    0x0(%rip),%edx        # 22 <main+0x22>
  22:   8b 05 00 00 00 00       mov    0x0(%rip),%eax        # 28 <main+0x28>
  28:   01 d0                   add    %edx,%eax
  2a:   89 05 00 00 00 00       mov    %eax,0x0(%rip)        # 30 <main+0x30>
  30:   8b 05 00 00 00 00       mov    0x0(%rip),%eax        # 36 <main+0x36>
  36:   89 c6                   mov    %eax,%esi
  38:   48 8d 05 00 00 00 00    lea    0x0(%rip),%rax        # 3f <main+0x3f>
  3f:   48 89 c7                mov    %rax,%rdi
  42:   b8 00 00 00 00          mov    $0x0,%eax
  47:   e8 00 00 00 00          call   4c <main+0x4c>
  4c:   b8 00 00 00 00          mov    $0x0,%eax
  51:   5d                      pop    %rbp
  52:   c3                      ret
choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ objdump -f a.out

a.out:     file format elf64-x86-64
architecture: i386:x86-64, flags 0x00000112:
EXEC_P, HAS_SYMS, D_PAGED
start address 0x0000000000401060

choijm@embedded:~/syspro/chap2$
choijm@embedded:~/syspro/chap2$ objdump -d a.out

a.out:     file format elf64-x86-64


Disassembly of section .init:

0000000000401000 <_init>:
  401000:       f3 0f 1e fa             endbr64
  401004:       48 83 ec 08             sub    $0x8,%rsp
  401008:       48 8b 05 c1 2f 00 00    mov    0x2fc1(%rip),%rax        # 403fd0
 <__gmon_start__@Base>
  40100f:       48 85 c0                test   %rax,%rax
```

# 사사

- 본 교재는 2025년도 과학기술정보통신부 및 정보통신기획평가원의 ‘SW중심대학사업’ 지원을 받아 제작 되었습니다.
- 본 결과물의 내용을 전재할 수 없으며, 인용(재사용)할 때에는 반드시 과학기술정보통신부와 정보통신기획평가원이 지원한 ‘SW중심대학’의 결과물이라는 출처를 밝혀야 합니다.

IITP 정보통신기획평가원

디지털인재양성단 SW인재팀

SW중심대학이 무엇인가요?

SW중심대학은
대학교육을 SW중심으로 혁신함으로써,
SW전문인력을 양성하고
학생·기업·사회의 SW경쟁력을 강화해
진정한 SW가치 확산을 실현하는 대학을 말합니다.