

Lecture Note 0.

Lecture Overview

September 1, 2025

Jongmoo Choi
Dept. of Software
Dankook University

<http://embedded.dankook.ac.kr/~choijm>

(본 교재는 2025년도 과학기술정보통신부 및 정보통신기획평가원의 ‘SW중심대학사업’ 지원을 받아 제작 되었습니다.)

Contents

- Course objectives
 - ✓ What can we learn in this semester?
- Course contents
 - ✓ Text book, Lecture notes, ...
- Course methods
 - ✓ Assignment, Grade, ...

DKU

단국대학교 웹정보시스템

강의계획서 [2025년도 2 학기]

I 교과목 기본정보(Course Information)

교과목명 Course Title	시스템프로그래밍	학점 Credits	3
교과목 코드 Course Code	366850-1	이수영역	전공필수
주수강대상	SW융합대학 소프트웨어학과	언어 Language	
강의형태		강의실	월4,5,6/수4,5,6(소프트332)
시간구분	이론(3) 실형(0) 실습(0) 실기(0) 설계(0)	사이버강의	웹보조수업
강의유형	대면수업		
원격수업미리보기			

II 담당교수

담당교수	성명	최종우	직급	교수	최종학위	공학박사
	소속	SW융합대학		연구실	소프트웨어 ICT관 504	
	전화번호	031-8005-3140		e-mail	choijm@dankook.ac.kr	
	관심분야					

III 교과목 설명(Course Summary)

교과목 개요	본 강의는 컴퓨터 하드웨어와 소프트웨어가 어떻게 통합되어 동작하는지 배운다. 이를 위해 대표적인 시스템 소프트웨어인 운영체제, 컴파일러, 어셈블러, 링커/로더, 라이브러리, 디버거 등에 대하여 논의한다. 또한 386, 펜티엄, i3/5/7 등 IA(Intel Architecture) 구조와 IA 어셈블리 언어를 배우고, 프로그램이 CPU 상에서 어떻게 수행되는지 이해한다. 이러한 과정을 통해 컴퓨터 시스템을 여러 층에서 추상화할 수 있는 능력을 키우며, 각 추상화 간에 인터페이스를 이해하는 것이 본 강의의 주요 목표가 된다. 강의 내용을 구체적으로 이해하기 위해 Linux 운영체제 상에서 태스크의 생성과 파일 입출력, 셸, 어셈블리 프로그래밍, 디버깅 등의 프로그램을 직접 작성해 보게 된다. 또한 HW/SW co-design 기반 최적화를 주제로 하는 설계과정을 실시한다. 특히 올해는 SW중심대학사업의 지원으로 솔리드 클라우드 2.0 환경을 사용하여 실습을 진행하며, 산업체에서 요구하는 주제에 대한 PBL (Project Based Learning)을 수행해 본다.
연계교과목 정보	C 프로그래밍 컴퓨터 구조: 시스템 프로그램의 관리 대상이 되는 컴퓨터 하드웨어, 특히 CPU 구조
역량기반 학습목표	지식활용역량: 시스템 프로그램의 종류와 기능을 이해한다. 창의적문제해결역량: 프로그램이 CPU에서 어떻게 동작하는지 이해한다.



Course Objectives (1/2)

■ What is System Programming?

- ✓ Application program vs. System program

```
#include <stdio.h>

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

- ☞ How to run this program on CPU?
- ☞ What is the role of printf()?
- ☞ How the string is displayed on Monitor?
- ☞ How this program can be executed with other programs concurrently?
- ☞ What are the differences between local and global variables?
- ☞ What if we split the string "Hello, World\n" into two strings with two printf()s?



Course Objectives (2/2)

- Understand how software runs on hardware (or how software and hardware are connected)
 - ✓ High-level program for human vs. **Binary** for CPU
 - ✓ Compiler, Assembler, Linker, Loader, Debugger, **Library** (dll), ...
 - ✓ File system, **Device driver**
 - ✓ Concept of Process, **Scheduling** for multiple processes
 - ✓ Memory management (data/stack/heap, **virtual memory**)
 - ✓ Software-level **optimizations**: code motion, loop unrolling, ...
 - ✓ Hardware-level **optimizations**: pipeline, cache, ...
 - ✓ Recent technologies in Intel CPU

- Grasp the concept of **abstraction**
 - ✓ Information hiding (expose relevant information only)
 - ✓ Interface vs. Implementation
 - ✓ Layered architecture



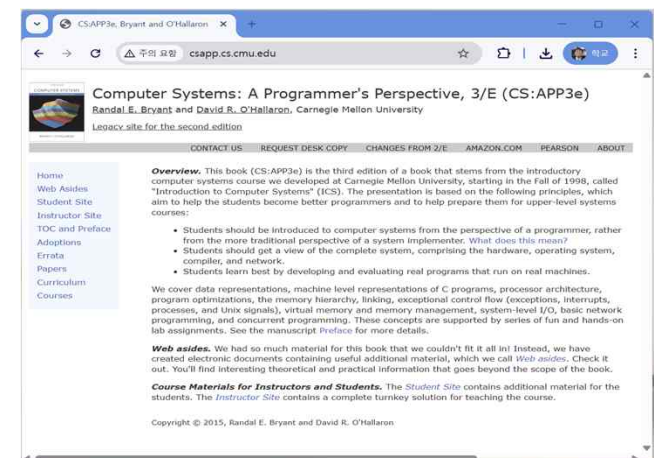
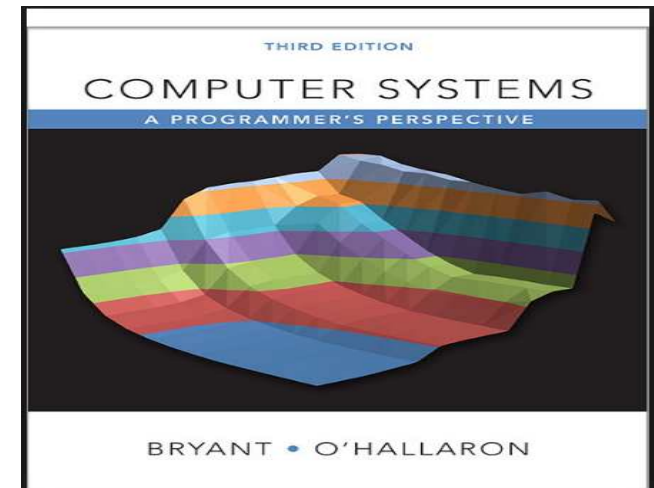
Course Contents (1/4)

■ Textbook 1: CSAPP

✓ Computer Systems: A Programmer's Perspective, by R. Bryant and D. O'Hallaron

✓ Contents

1. A Tour of Computer Systems
2. Representing and Manipulating Information
3. Machine-level Representation of Programs
4. Processor Architecture
5. Optimizing Program Performance
6. The Memory Hierarchy
7. Linking
8. Exceptional Control Flow
9. Virtual Memory
10. System-Level I/O
11. Network Programming
12. Concurrent Programming



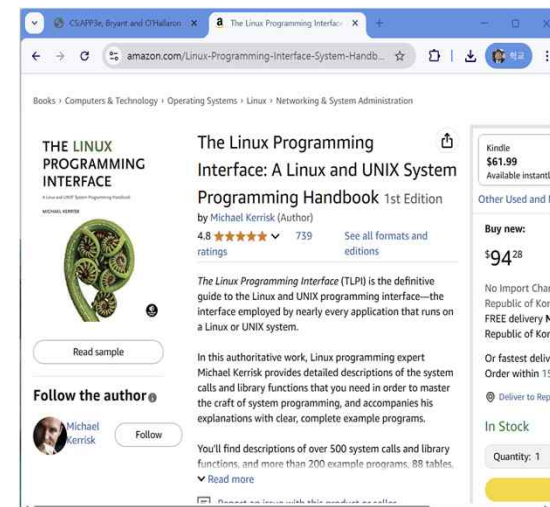
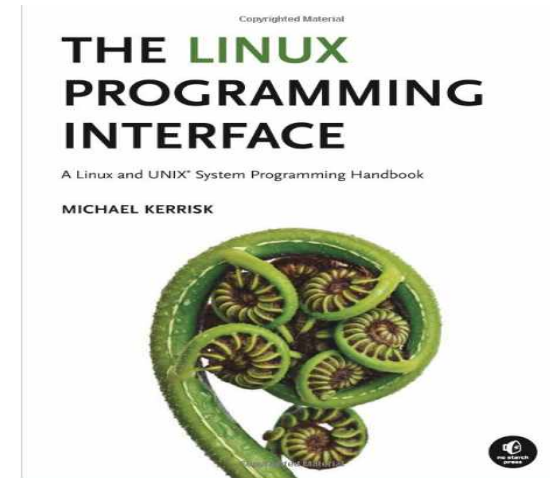
(<http://csapp.cs.cmu.edu/>)

Course Contents (2/4)

■ Textbook 2: LPI

✓ The Linux Programming Interface: A Linux and UNIX System Programming Handbook

1. History and Standards
2. Fundamental Concepts
3. System programming concepts
4. File I/O: The Universal I/O Model
5. File I/O: Further Details
6. Process
7. Memory Allocation
8. Users and Groups
- ...
24. Process Creation
25. Process Termination
26. Monitoring Child Processes
27. Program Execution
- ... /* total 64 chapters */



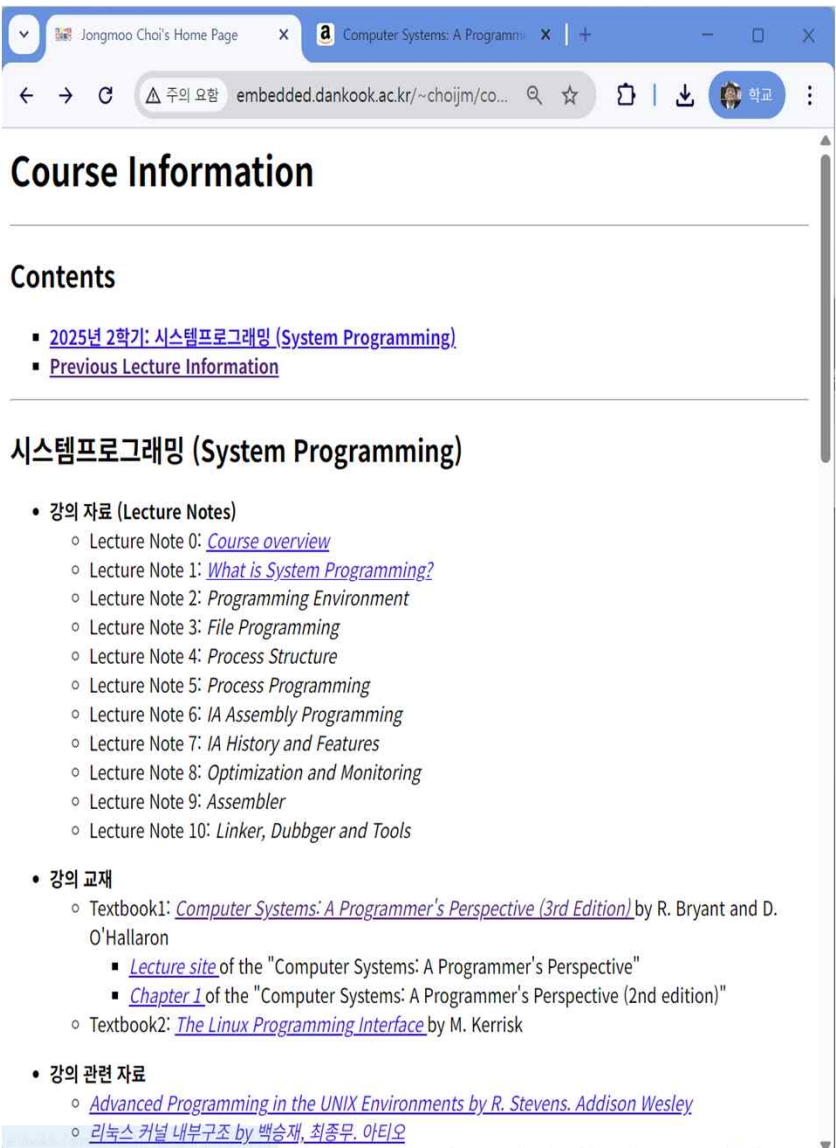
(<https://www.amazon.com/>)

Course Contents (3/4)

■ Lecture Notes

- ✓ LN0: Course Overview
- ✓ LN1: What is System Programming?
- ✓ LN2: Programming Environments
- ✓ LN3: File Programming
- ✓ LN4: Process Structure
- ✓ LN5: Process Programming

- ✓ LN6: IA Assembly Programming
- ✓ LN7: IA History and Features
- ✓ LN8: Optimization Practice
- ✓ LN9: Assembler
- ✓ LN10: Linker, Debugger and Tools



The screenshot shows a web browser with two tabs: 'Jongmoo Choi's Home Page' and 'Computer Systems: A Programmer's Perspective'. The address bar shows the URL 'embedded.dankook.ac.kr/~choijm/co...'. The page title is 'Course Information'. The main content area is titled 'Contents' and lists two items: '2025년 2학기: 시스템프로그래밍 (System Programming)' and 'Previous Lecture Information'. Below this, there is a section titled '시스템프로그래밍 (System Programming)' which contains a list of lecture notes (LN0 to LN10) and a list of textbooks. The lecture notes are: LN0: Course overview, LN1: What is System Programming?, LN2: Programming Environment, LN3: File Programming, LN4: Process Structure, LN5: Process Programming, LN6: IA Assembly Programming, LN7: IA History and Features, LN8: Optimization and Monitoring, LN9: Assembler, and LN10: Linker, Debugger and Tools. The textbooks are: 'Computer Systems: A Programmer's Perspective (3rd Edition)' by R. Bryant and D. O'Hallaron, and 'The Linux Programming Interface' by M. Kerrisk.

Course Information

Contents

- 2025년 2학기: 시스템프로그래밍 (System Programming)
- Previous Lecture Information

시스템프로그래밍 (System Programming)

- 강의 자료 (Lecture Notes)
 - Lecture Note 0: [Course overview](#)
 - Lecture Note 1: [What is System Programming?](#)
 - Lecture Note 2: [Programming Environment](#)
 - Lecture Note 3: [File Programming](#)
 - Lecture Note 4: [Process Structure](#)
 - Lecture Note 5: [Process Programming](#)
 - Lecture Note 6: [IA Assembly Programming](#)
 - Lecture Note 7: [IA History and Features](#)
 - Lecture Note 8: [Optimization and Monitoring](#)
 - Lecture Note 9: [Assembler](#)
 - Lecture Note 10: [Linker, Debugger and Tools](#)
- 강의 교재
 - Textbook1: [Computer Systems: A Programmer's Perspective \(3rd Edition\)](#) by R. Bryant and D. O'Hallaron
 - [Lecture site](#) of the "Computer Systems: A Programmer's Perspective"
 - [Chapter 1](#) of the "Computer Systems: A Programmer's Perspective (2nd edition)"
 - Textbook2: [The Linux Programming Interface](#) by M. Kerrisk
- 강의 관련 자료
 - [Advanced Programming in the UNIX Environments by R. Stevens. Addison Wesley](#)
 - [리눅스 커널 내부구조 by 백승재, 최종무, 아티오](#)

(<http://embedded.dankook.ac.kr/~choijm/>)

Course Contents (4/4)

■ Suggestion

- ✓ Lecture notes are sufficient for this class
- ✓ But, text books are powerful tools to make your knowledge deeper

■ Relation btw Lecture Notes and Textbooks

- ✓ LN1. What is System Programming? : [CSAPP Chap. 1](#)
- ✓ LN2. Programming Environment: [LPI Chap. 1, 2, 3](#)
- ✓ LN3. File Programming: [LPI Chap. 4, 5](#) / CSAPP Chap. 10
- ✓ LN4. Process Structure: [LPI Chap. 6](#) / CSAPP Chap. 8, 9
- ✓ LN5. Process Programming: [LPI Chap. 24, 25, 27, 29](#) / CSAPP Chap. 8, 12
- ✓ LN6. IA assembly Programming: [CSAPP Chap. 2, 3](#) / Intel Dev. Manual
- ✓ LN7. IA History and Features: [CSAPP Chap. 4](#) / Intel Dev. Manual
- ✓ LN8. Optimization Practice: [CSAPP Chap. 5, 6](#) / LPI Chap. 23
- ✓ LN9. Assembler: [CSAPP Chap. 3](#)
- ✓ LN10. Linker, Debugger and Tools: [CSAPP Chap. 7](#)



Course Methods (1/3)

■ Class hour

✓ Lecturing and Discussion (Q&A)

- Using ppt from lecture site
- Q&A is quite important (especially I like questions from students)

Jongmoo Choi's Home Page x Computer Systems: A Programmer's Perspective x +

주요요항 embedded.dankook.ac.kr/~choijm/co... Q ☆ | | 학교

시스템프로그래밍 (System Programming)

- 강의 자료 (Lecture Notes)
 - Lecture Note 0: [Course overview](#)
 - Lecture Note 1: [What is System Programming?](#)
 - Lecture Note 2: [Programming Environment](#)
 - Lecture Note 3: [File Programming](#)
 - Lecture Note 4: [Process Structure](#)
 - Lecture Note 5: [Process Programming](#)
 - Lecture Note 6: [IA Assembly Programming](#)
 - Lecture Note 7: [IA History and Features](#)
 - Lecture Note 8: [Optimization and Monitoring](#)
 - Lecture Note 9: [Assembler](#)
 - Lecture Note 10: [Linker, Debugger and Tools](#)
- 강의 교재
 - Textbook1: [Computer Systems: A Programmer's Perspective \(3rd Edition\)](#) by R. Bryant and D. O'Hallaron
 - Lecture site of the "Computer Systems: A Programmer's Perspective"
 - Chapter 1 of the "Computer Systems: A Programmer's Perspective (2nd edition)"
 - Textbook2: [The Linux Programming Interface](#) by M. Kerrisk
- 강의 관련 자료
 - [Advanced Programming in the UNIX Environments](#) by R. Stevens, Addison-Wesley
 - [리눅스 커널 내부구조](#) by 백승재, 최준무, 아티오
 - [Linux System Programming: Talking Directly to the Kernel](#) by R. Love, O'Reilly
 - [유닉스/리눅스 프로그래밍 필수 유틸리티](#) by 백창용, 한빛
 - [Intel 64 and IA-32 Architecture Software Developer's Manual](#)
 - [ARM System-on-Chip Architecture \(2nd Edition\)](#) by S. Furber
 - The UNIX time-sharing system: [UNIX paper](#)
 - [RAG 기반 운영체제 교육](#)
 - [How do Hard Disk Drives Work?](#)
 - [Inside of a Hard Disk](#)
 - [Concept of Pipeline](#)
 - [Memory Address](#)
 - [GNU GCC](#)
 - [GNU Assembler](#)
 - [GNU Debugger](#)
 - [Quick Guide to GDB](#)
 - [GNU Make](#)

CSAPP3e, Bryant and O'Hallaron x +

주요요항 csapp.cs.cmu.edu

Computer Systems: A Programmer's Perspective, 3/E (CS:APP3e)

Randal E. Bryant and David R. O'Hallaron, Carnegie Mellon University

Leave site for the second edition

CONTACT US REQUEST DESK COPY CHANGES FROM 2/E AMAZON.COM PEARSON ABOUT

Home Web Author Student Site Instructor Site TOC and Preface Adoptions Errata Papers Curriculum Courses

Overview. This book (CS:APP3e) is the third edition of a book that stems from the introductory computer systems course we developed at Carnegie Mellon University, starting in the Fall of 1989, called "Introduction to Computer Systems" (ICS). The presentation is based on the following principles, which aim to help the students become better programmers and to help prepare them for upper-level systems courses:

- Students should be introduced to computer systems from the perspective of a programmer, rather than from the more traditional perspective of a system implementer; what does this mean?
- Students should get a view of the complete system, comprising the hardware, operating system, compiler, and network.
- Students learn best by developing and evaluating real programs that run on real machines.

We cover data representations, machine-level representations of C programs, processor architectures, program optimizations, the memory hierarchy, linking, exceptional control flow (exceptions, interrupts, processes, and Unix signals), virtual memory and memory management, system-level I/O, basic network programming, and concurrent programming. These concepts are supported by series of fun and hands-on lab assignments. See the manuscript preface for more details.

Web aides. We had so much material for this book that we couldn't fit it all in! Instead, we have created electronic documents containing useful additional material, which we call *web aides*. Check it out: You'll find interesting theoretical and practical information that goes beyond the scope of the book.

Course Materials for Instructors and Students. The Student Site contains additional material for the students. The Instructor Site contains a complete turnkey solution for teaching the course.

Copyright © 2015, Randal E. Bryant and David R. O'Hallaron

Intel® 64 and IA-32 Architectures x +

intel.com/content/www/us/en/developer/articles/technical/intel-sdm.html

PRODUCTS SUPPORT MORE +

Developers Tools Intel® 64 and IA-32 Architecture Developer Manuals

ID	Version	Latest	Public
767375			

Overview

Combined Volume Set of Intel® 64 and IA-32 Architectures Software Developer's Manuals

Four-Volume Set of Intel® 64 and IA-32 Architectures Software Developer's Manuals

Ten-Volume Set of Intel® 64 and IA-32 Architectures Software Developer's Manuals

Overview

These manuals describe the architecture and programming environment of the Intel® 64 and IA-32 architectures.

Electronic versions of these documents allow you to quickly get to the information you need and print only the pages you want. The Intel® 64 and IA-32 architectures software developer's manuals.

Beej's Quick Guide to GDB x +

beej.us/guide/bgdb/

Beej's Quick Guide to GDB

Release 2 (2009 Jun 14)

Translations

- Russian

This is a very quick and dirty guide meant to get you started with the GNU Debugger, **gdb**, from the comfort of your own terminal. **gdb** is run via an IDE, but many people out there learn IDEs for a variety of reasons, and this tutorial is for you.

Again, this is only a getting-started guide. There's much much MUCH more to learn about what the debugger does than is written in these few short paragraphs. Check out your "man" pages or the online resources listed below for more info.

This tutorial is meant to be read in order, up to, but not including, the "Misc" section.

Contents

- Compiling to use a debugger
- More Information
- Starting **gdb** and getting to **main()**
- Breaking
- Stepping Around
- Examining Variables
- Next Step
 - Quick Manipulation
 - Additional Stepping Methods
 - Changing the Section of Code
 - Changing Variables and Values at Runtime
 - Hardware Watchpoints
 - Attach to a Running Process
 - Using Core Dumps for Postmortem Analysis
 - Using Core Dumps for Postmortem Analysis
 - Display Registers and Assembly
 - Writing a Final End
- Quick Reference Cheat Sheet

Compiling

You have to tell your compiler to compile your code with symbolic debugging information included. Here's

Course Methods (2/3)

■ Assignment: personal

✓ Programming assignment: 4 or 5

■ Make programs in **Linux Environment!!**

- Linux Server: 220.149.236.2 (primary), 220.149.236.4 (secondary)
 - Option: Linux in Virtual machine, Linux in Cloud (DKU's SOLID Cloud)
 - Note: Server IP number can be altered according to university's policy.
- TA: Yeongyu Choi (Room 515, SW-ICT Bldg)

■ Program examples

- Using vi editor, file I/O, process manipulation, shell, assembly, optimization, ...

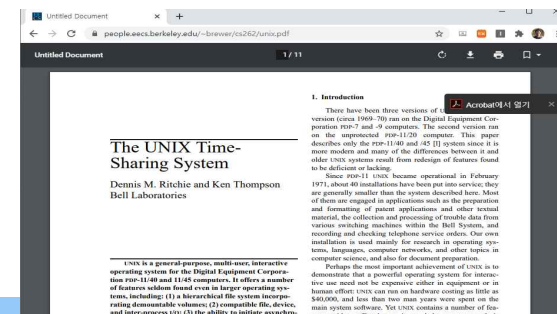
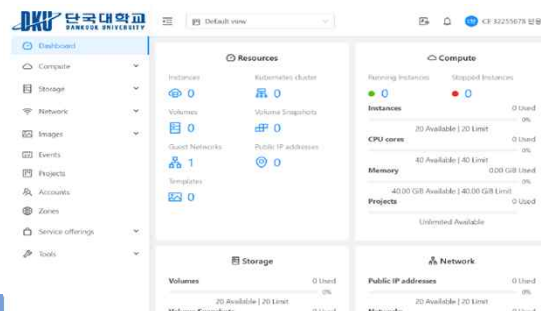
✓ Documentation assignment: 1 or 2

■ Reading a chapter in our textbooks

- E.g. Chapter 1 in CSAPP or Chapter 3 in LPI

■ Reading a well-known paper

- E.g. UNIX paper



■ Evaluation

- ✓ Mid exam.: 30%
- ✓ Final exam.: 30%
- ✓ Assignment: 30%
- ✓ Attendance/Q&A: 10%
 - Can be changed according to the progress

■ Grade

- ✓ Roughly, 20% students are expected to get the A grade.
 - 45% for B, others for C or D
- ✓ **Absence more than 5 times or Mid and Final Exam. Score below 20 or No assignment → F**



Discussion

- Q&A

- ✓ Email: choijm@dankook.ac.kr



Appendix: Intel Developer's Manual

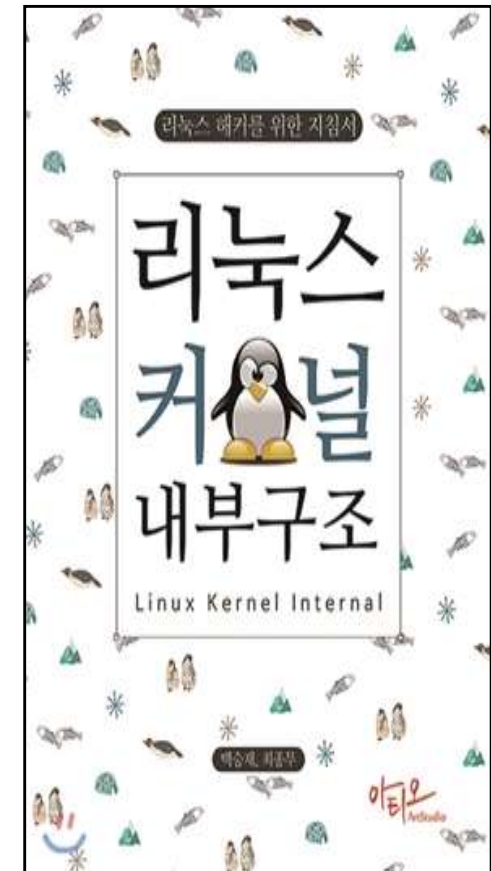
- Intel®64 & IA-32 Architectures Software Developer's Manual (Volume 1: Basic Architecture)
 1. About This Manual
 2. Intel® 64 and IA-32 Architecture
 3. Basic Execution Environment
 4. Data type
 5. Instruction Set Summary
 6. Procedure Calls, Interrupts, and Exceptions
 7. Programming with General Purpose Instructions
 8. Programming with the x87 FPU
 9. Programming with Intel MMX Technology
 10. Programming with Streaming SIMD Extensions
 11. ...



Appendix: Good book for Learning Linux

■ Linux Kernel Internal (리눅스 커널 내부 구조)

- ✓ 0장. 운영체제 이야기
- ✓ 1장. 리눅스 소개
- ✓ 2장. 리눅스 커널 구조
- ✓ 3장. 태스크 관리
- ✓ 4장. 메모리 관리
- ✓ 5장. 파일 시스템과 가상 파일 시스템
- ✓ 6장. 인터럽트와 트랩 그리고 시스템 호출
- ✓ 7장. 리눅스 모듈 프로그래밍
- ✓ 8장. 디바이스 드라이버
- ✓ 9장. 네트워킹
- ✓ 10장. 운영체제 관련 실습
- ✓ 부록 A. 리눅스와 가상화 그리고 XEN
- ✓ 부록 B. MTD와 YAFFS
- ✓ 부록 C: Map of the Linux



- 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

