# Lecture Note 0:
# Course Introduction

March 4, 2025

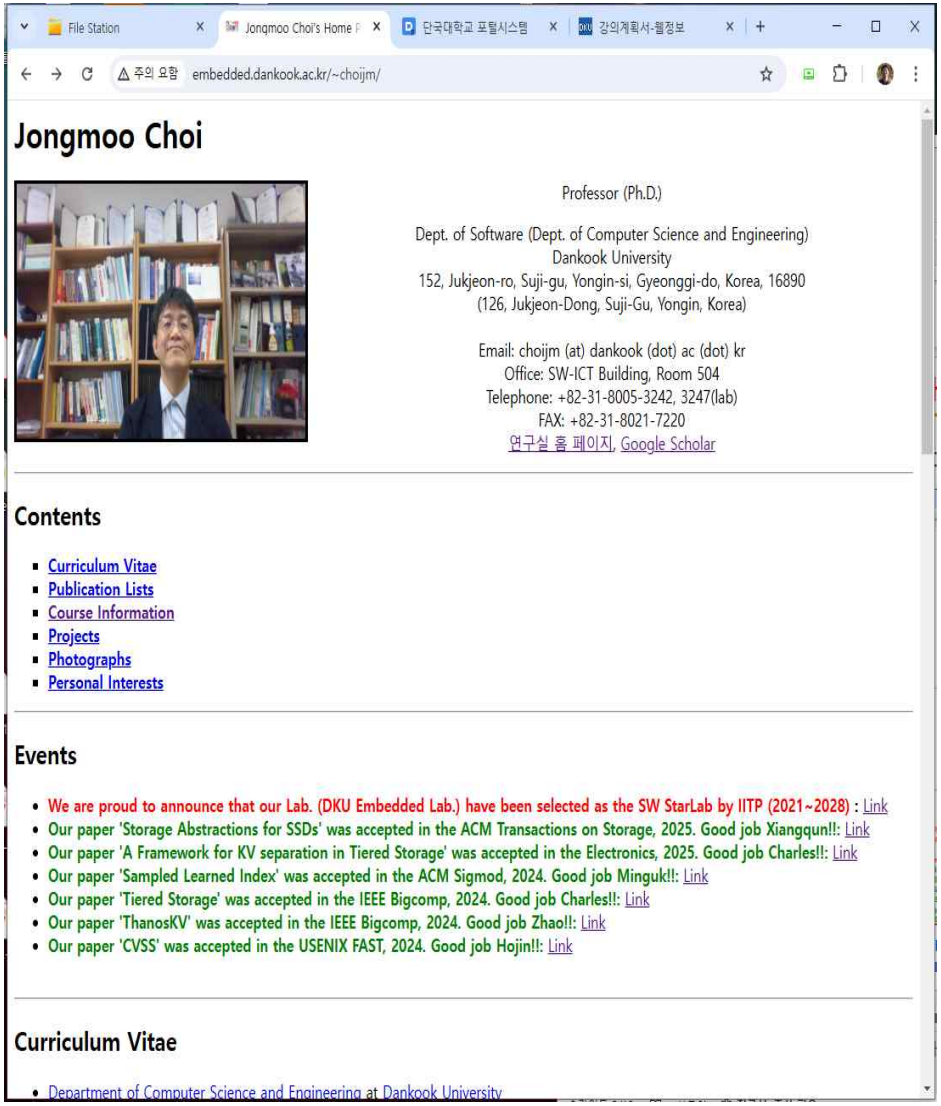
Jongmoo Choi

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

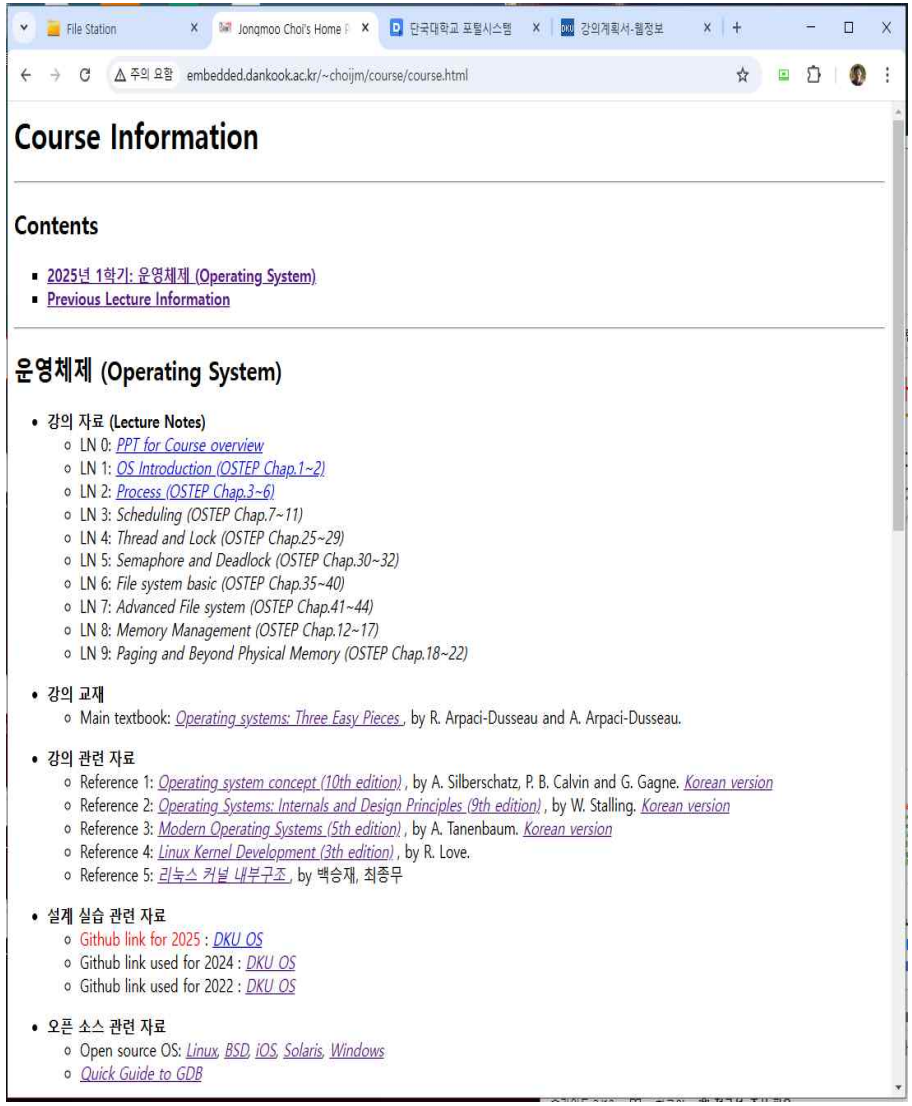
(본 교재는 **2025**년도 과학기술정보통신부 및 정보통신기획평가원의 '**SW**중심대학사업' 지원을 받아 제작 되었습니다.)

# How to access lecture contents?

- Lecture site



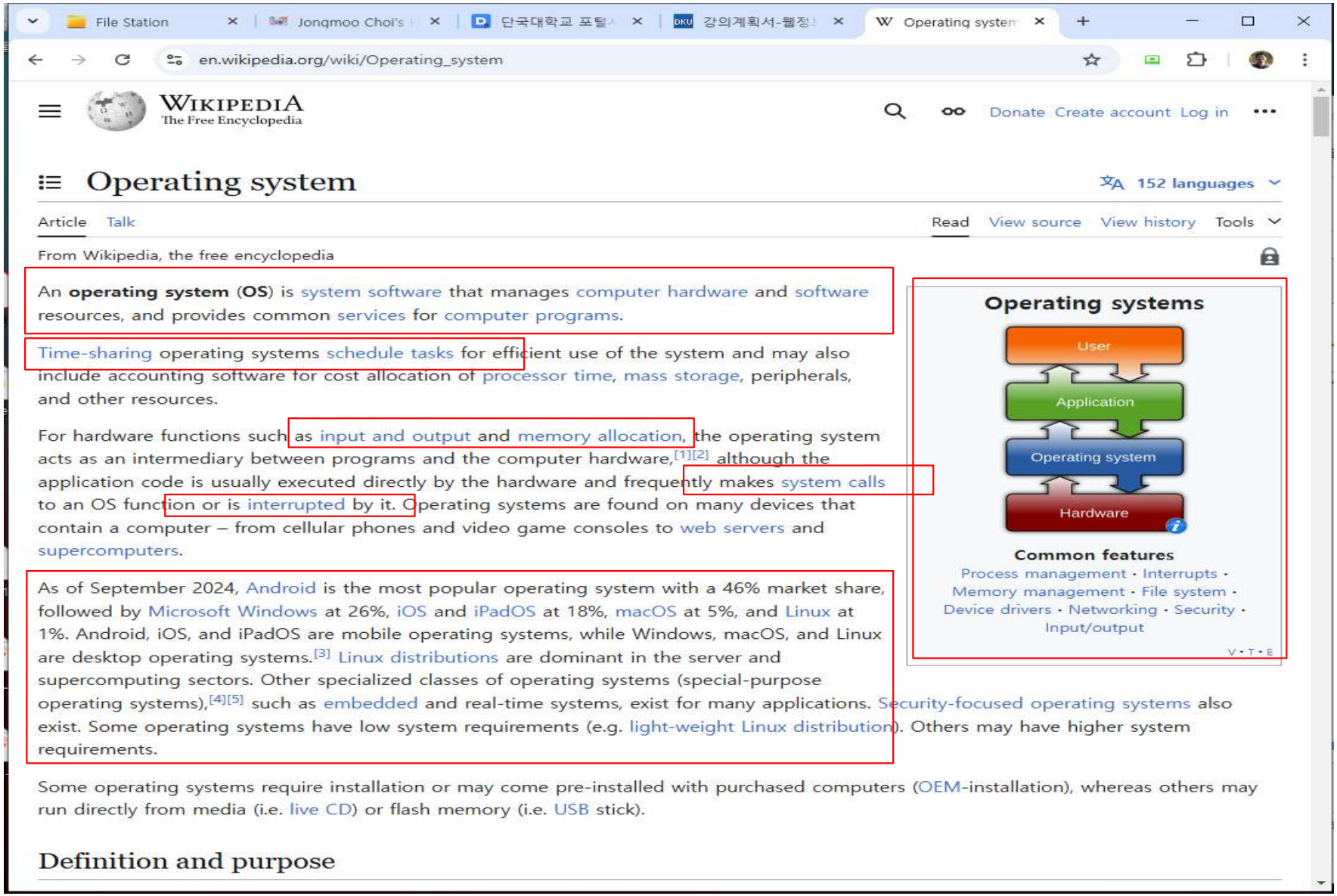

(home page)




(lecture page)

# What is Operating System?

- Definition (from wikipedia.org)



File Station | Jongmoo Choi's | 단국대학교 포털 | 강의계획서-웹정 | Operating system

en.wikipedia.org/wiki/Operating_system

WIKIPEDIA
The Free Encyclopedia

Donate Create account Log in

## Operating system

152 languages

Article Talk | Read View source View history Tools

From Wikipedia, the free encyclopedia

An **operating system** (**OS**) is system software that manages computer hardware and software resources, and provides common services for computer programs.

Time-sharing operating systems schedule tasks for efficient use of the system and may also include accounting software for cost allocation of processor time, mass storage, peripherals, and other resources.

For hardware functions such as input and output and memory allocation, the operating system acts as an intermediary between programs and the computer hardware,[1][2] although the application code is usually executed directly by the hardware and frequently makes system calls to an OS function or is interrupted by it. Operating systems are found on many devices that contain a computer – from cellular phones and video game consoles to web servers and supercomputers.

As of September 2024, Android is the most popular operating system with a 46% market share, followed by Microsoft Windows at 26%, iOS and iPadOS at 18%, macOS at 5%, and Linux at 1%. Android, iOS, and iPadOS are mobile operating systems, while Windows, macOS, and Linux are desktop operating systems.[3] Linux distributions are dominant in the server and supercomputing sectors. Other specialized classes of operating systems (special-purpose operating systems),[4][5] such as embedded and real-time systems, exist for many applications. Security-focused operating systems also exist. Some operating systems have low system requirements (e.g. light-weight Linux distribution). Others may have higher system requirements.

Some operating systems require installation or may come pre-installed with purchased computers (OEM-installation), whereas others may run directly from media (i.e. live CD) or flash memory (i.e. USB stick).

### Definition and purpose

**Operating systems**



**Common features**

Process management · Interrupts · Memory management · File system · Device drivers · Networking · Security · Input/output

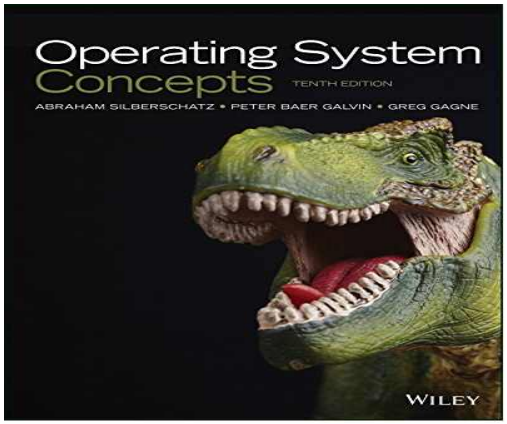
V · T · E

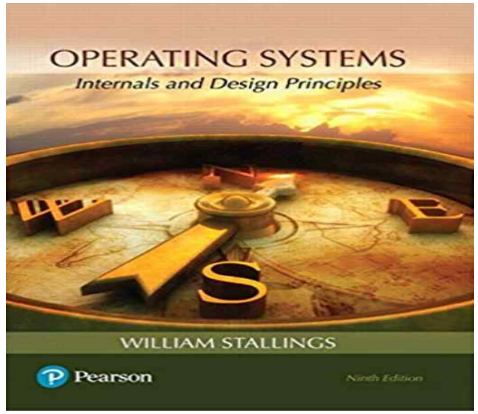
# Course Objectives

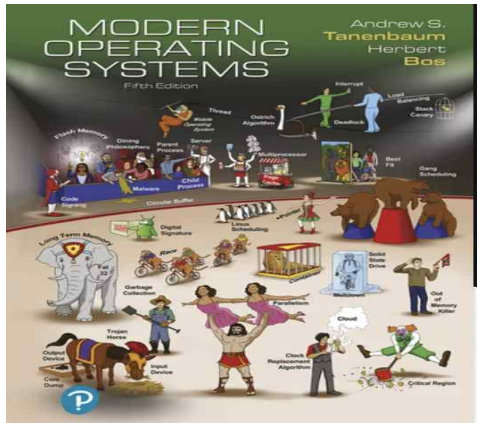
- Understand the definition, role and goal of OS
  - ✓ Resource manager, computing environments, …
- Know the existing operating systems
  - ✓ UNIX, Windows, Apple OS X, Linux, Android, iOS, WebOS, Mach, …
- Learn the internal structure of OS
  - ✓ Process, Virtual memory, File system, Driver, Protocol, Interrupt, …
- Comprehend the policies and mechanisms used by OS
  - ✓ CPU scheduling, Demand paging, LRU, inode, System call, …
- Grasp the idea of abstraction
  - ✓ Information Hiding, Illusion, Interface, Layered architecture, …
- Demonstrate what we have learned
  - ✓ Lab. project

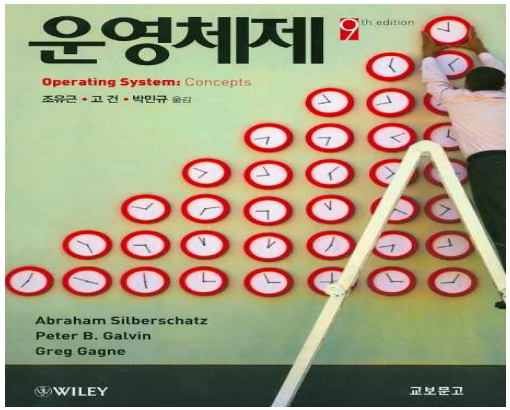
# Traditional Textbook

- Three representative textbooks for operating system course
  - ✓ Operating Systems Concepts, by A. Silberschatz, P. Galvin and G. Gagne
  - ✓ Operating Systems: Internals and Design Principles, by W. Stalling
  - ✓ Modern Operating Systems, by A. Tanenbaum and H. Bos

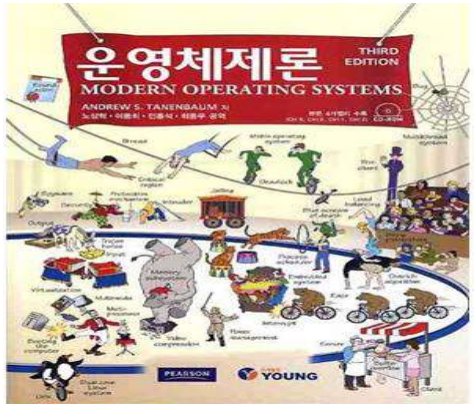


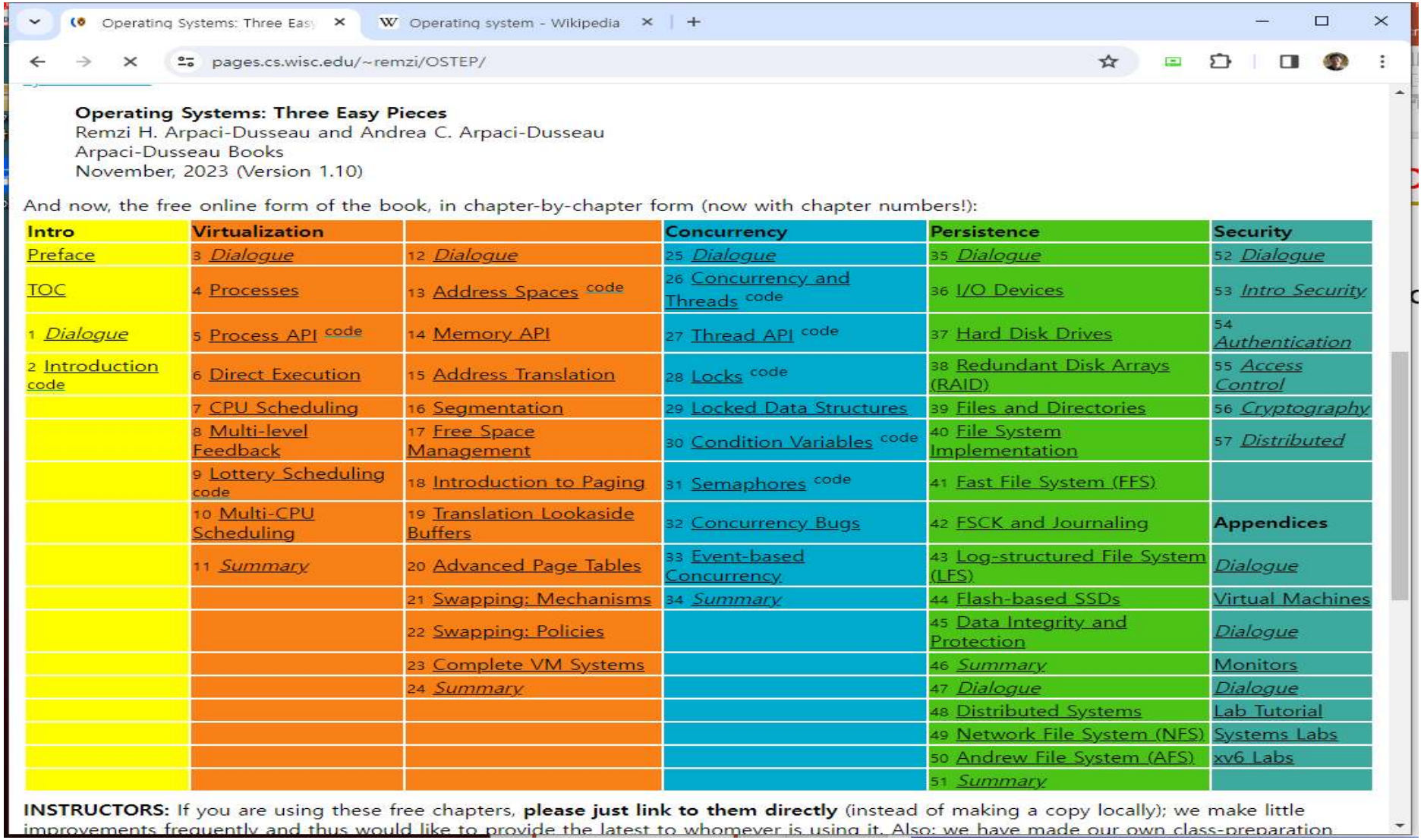







# Textbook in this course

- Remzi's OSTEP (OS Three Easy Pieces)
  - ✓ http://pages.cs.wisc.edu/~remzi/OSTEP/



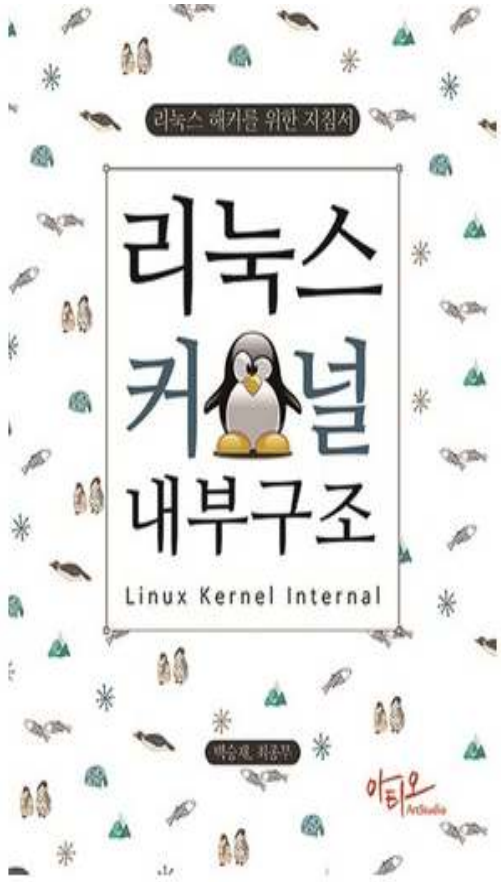
# Textbook in this course

- TOC (Table of Contents) of OSTEP

## Contents

# Reference

- Linux Kernel Internals (리눅스 커널 내부 구조)
  - ✓ 1장. 리눅스 소개
  - ✓ 2장. 리눅스 커널 구조
  - ✓ 3장. 태스크 관리
  - ✓ 4장. 메모리 관리
  - ✓ 5장. 파일시스템과 가상 파일시스템
  - ✓ 6장. 인터럽트와 트랩 그리고 시스템 호출
  - ✓ 7장. 리눅스 모듈 프로그래밍
  - ✓ 8장. 디바이스 드라이버
  - ✓ 9장. 네트워킹
  - ✓ 10장. 운영체제 관련 실습
  - ✓ 부록1. 리눅스와 가상화 그리고 XEN
  - ✓ 부록2. MTD와 YAFFS



# Teaching Method

- Mainly Lecturing
  - ✓ Discussion (Q&A) during the course is quite important
- Homework
  - ✓ Reading assignment
    - 1 or 2 times
  - ✓ Lab. Project (Programming or Analysis)
    - Lab1: scheduling
    - Lab2: concurrency
    - Lab3: file system
    - Lab4: virtual memory


- Grading
  - ✓ Exam(50%) + Lab. Project/Assignment (40%) + Attendance/Discussion (10%) ➔ can be changed later
  - ✓ Absence more than 5 times or Mid and Final Exam. score below 20 or No lab. Project ➔ F
  - ✓ Roughly, 20% students are expected to get the A grade.

# Discussion

- Any questions? Feel free to ask at our class or send an email to me: choijm@dankook.ac.kr

# Quiz for this Lecture

- Quiz
  - ✓ 1. What are the differences between Operating System (e.g. MS Windows or Linux) and Application (e.g. MS Word or Chrome)?. Explain the difference using the word “mode”.
  - ✓ 2. What are three pieces of Operating System?
  - ✓ 3. There is a Confucian philosopher, Xunzi, in Chapter 1, “A Dialog on the Book”, of the OSTEP. Explain what he said.


(Source: Google Image )

Operating Systems: Three Easy Pieces
Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau
Arpaci-Dusseau Books
November, 2023 (Version 1.10)

And now, the free online form of the book, in chapter-by-chapter form (now with chapter numbers!):

| Intro | Virtualization | | Concurrency | Persistence | Security |
|---|---|---|---|---|---|
| Preface | 3 Dialogue | 12 Dialogue | 25 Dialogue | 35 Dialogue | 52 Dialogue |
| TOC | 4 Processes | 13 Address Spaces code | 26 Concurrency and Threads code | 36 I/O Devices | 53 Intro Security |
| 1 Dialogue | 5 Process API code | 14 Memory API | 27 Thread API code | 37 Hard Disk Drives | 54 Authentication |
| 2 Introduction code | 6 Direct Execution | 15 Address Translation | 28 Locks code | 38 Redundant Disk Arrays (RAID) | 55 Access Control |
| | 7 CPU Scheduling | 16 Segmentation | 29 Locked Data Structures | 39 Files and Directories | 56 Cryptography |
| | 8 Multi-level Feedback | 17 Free Space Management | 30 Condition Variables code | 40 File System Implementation | 57 Distributed |
| | 9 Lottery Scheduling code | 18 Introduction to Paging | 31 Semaphores code | 41 Fast File System (FFS) | |
| | 10 Multi-CPU Scheduling | 19 Translation Lookaside Buffers | 32 Concurrency Bugs | 42 FSCK and Journaling | Appendices |
| | 11 Summary | 20 Advanced Page Tables | 33 Event-based Concurrency | 43 Log-structured File System (LFS) | Dialogue |
| | | 21 Swapping: Mechanisms | 34 Summary | 44 Flash-based SSDs | Virtual Machines |
| | | 22 Swapping: Policies | | 45 Data Integrity and Protection | Dialogue |
| | | 23 Complete VM Systems | | 46 Summary | Monitors |
| | | 24 Summary | | 47 Dialogue | Dialogue |
| | | | | 48 Distributed Systems | Lab Tutorial |
| | | | | 49 Network File System (NFS) | Systems Labs |
| | | | | 50 Andrew File System (AFS) | xv6 Labs |
| | | | | 51 Summary | |

INSTRUCTORS: If you are using these free chapters, **please just link to them directly** (instead of making a copy locally); we make little



**Professor:** *Excellent query! Well, each person needs to figure this out on their own, of course, but here is what I would do: go to class, to hear the professor introduce the material. Then, at the end of every week, read these notes, to help the ideas sink into your head a bit better. Of course, some time later (hint: before the exam!), read the notes again to firm up your knowledge. Of course, your professor will no doubt assign some homeworks and projects, so you should do those; in particular, doing projects where you write real code to solve real problems is the best way to put the ideas within these notes into action. As Confucius said...*

**Student:** *Oh, I know! 'I hear and I forget. I see and I remember. I do and I understand.' Or something like that.*

**Professor:** *(surprised) How did you know what I was going to say?!*

**Student:** *It seemed to follow. Also, I am a big fan of Confucius, and an even bigger fan of Xunzi, who actually is a better source for this quote*[1]*.*

**Professor:** *(stunned) Well, I think we are going to get along just fine! Just fine indeed.*

**Student:** *Professor – just one more question, if I may. What are these dialogues for? I mean, isn't this just supposed to be a book? Why not present the material directly?*

**Professor:** *Ah, good question, good question! Well, I think it is sometimes useful to pull yourself outside of a narrative and think a bit; these dialogues are those times. So you and I are going to work together to make sense of all of these pretty complex ideas. Are you up for it?*

**Student:** *So we have to think? Well, I'm up for that. I mean, what else do I have to do anyhow? It's not like I have much of a life outside of this book.*

**Professor:** *Me neither, sadly. So let's get to work!*

[1] According to http://www.baxypopik.com (on, December 19, 2012, entitled "Tell me and I forget; teach me and I may remember; involve me and I will learn") Confucian philosopher Xunzi said "Not having heard something is not as good as having heard it; having heard it is not as good as having seen it; having seen it is not as good as knowing it; knowing it is not as good as putting it into practice." Later on, the wisdom got attached to Confucius for some reason. Thanks to Jiao Dong (Rutgers) for telling us!

# 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