Jin-Soo Kim
(*jinsoo.kim@snu.ac.kr*)

Systems Software &
Architecture Lab.

Seoul National University

Spring 2026

4190.307:
Operating Systems

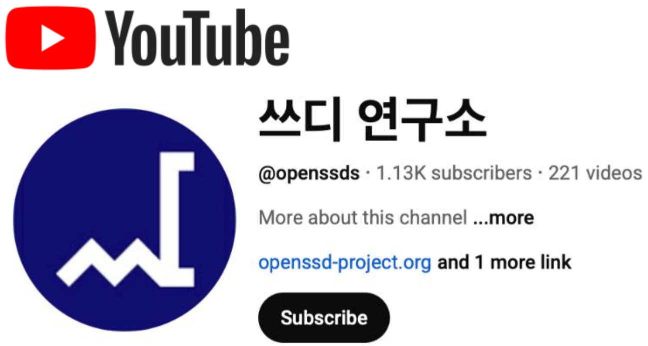
Course Information

- Schedule
  - 14:00 – 15:15 (Tuesday & Thursday)
  - Lecture room: Engineering Bldg. #301-203
  - 3 credits
  - Official language: Korean

- TA: Heejae Kim and Gahyun Lee (snucsl.ta [at] gmail.com)
- SNU eTL system for exam/project scores
- https://csl.snu.ac.kr/courses/4190.307/2026-1 for announcements and lecture slides
- https://sys.snu.ac.kr for project submissions and automatic grading

About Me

- Jin-Soo Kim (김진수)
  - Professor @ CSE Dept.
  - Systems Software & Architecture Laboratory
  - Operating systems, storage systems, parallel and distributed computing, embedded systems, …



쓰디 연구소

@openssds · 1.13K subscribers · 221 videos

More about this channel ...more

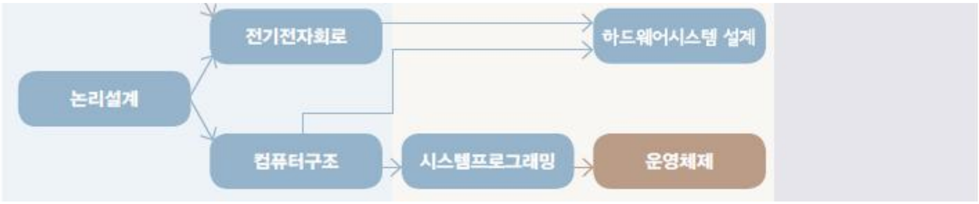
openssd-project.org and 1 more link

- E-mail: jinsoo.kim@snu.ac.kr
- Tel: 02-880-7302
- Office: Engineering Bldg. #301-504
- Office hours: Tuesday & Thursday (appointments by email)
- https://csl.snu.ac.kr/

Prerequisites

- Courses
  - Computer Architecture (4190.308) – Must!
  - System Programming (M1522.000800) – Must!

학사과정 선수 교과목 연계도

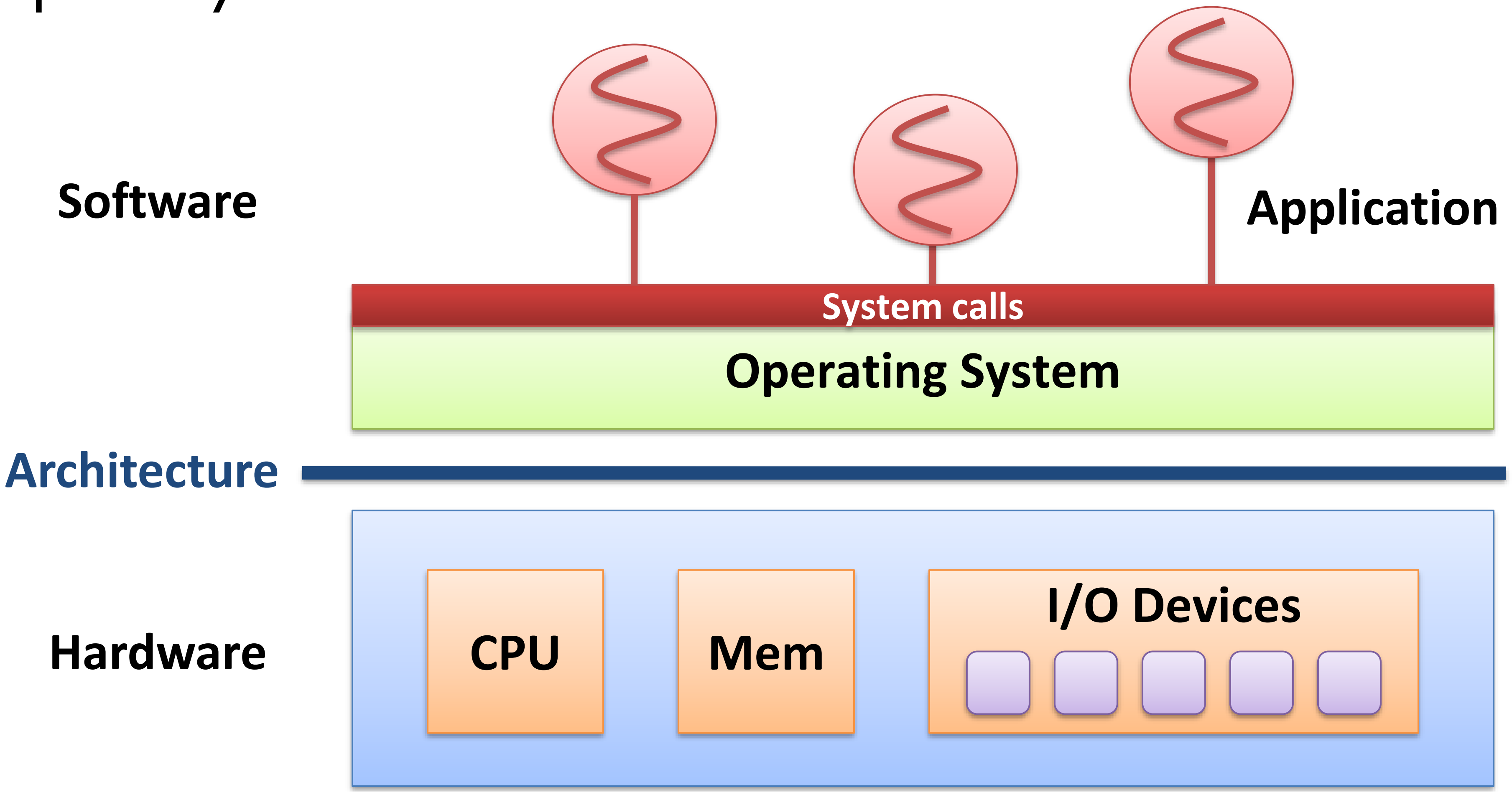


- Skills
  - Fluent C programming
  - Familiarity with Linux commands and build environment (e.g., gcc, gdb, make, …)
  - Reading a large, complex program
  - RISC-V architecture & assembly programming

- Accessible Linux (Ubuntu 24.04.1 LTS or later) or MacOS machine

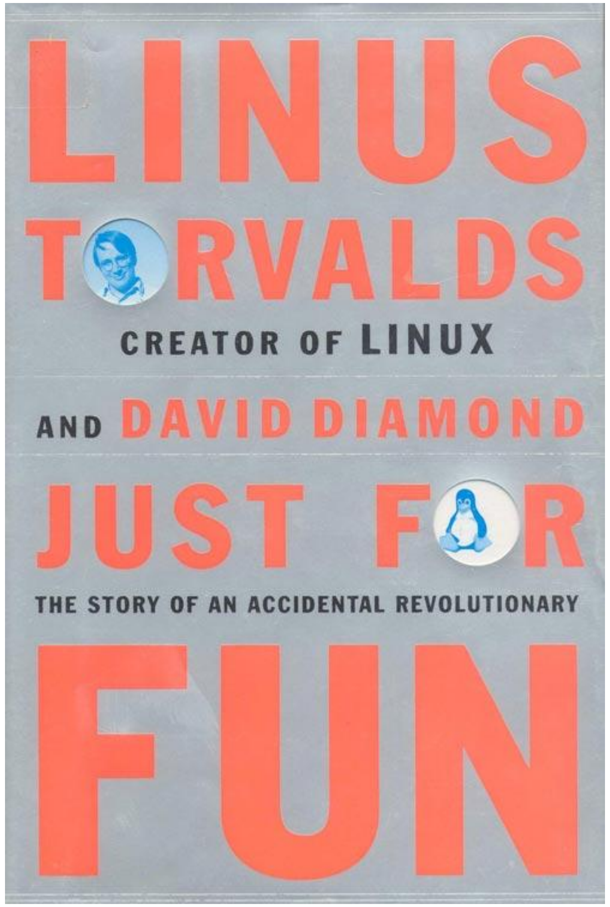
What is an OS?

- Computer systems internals



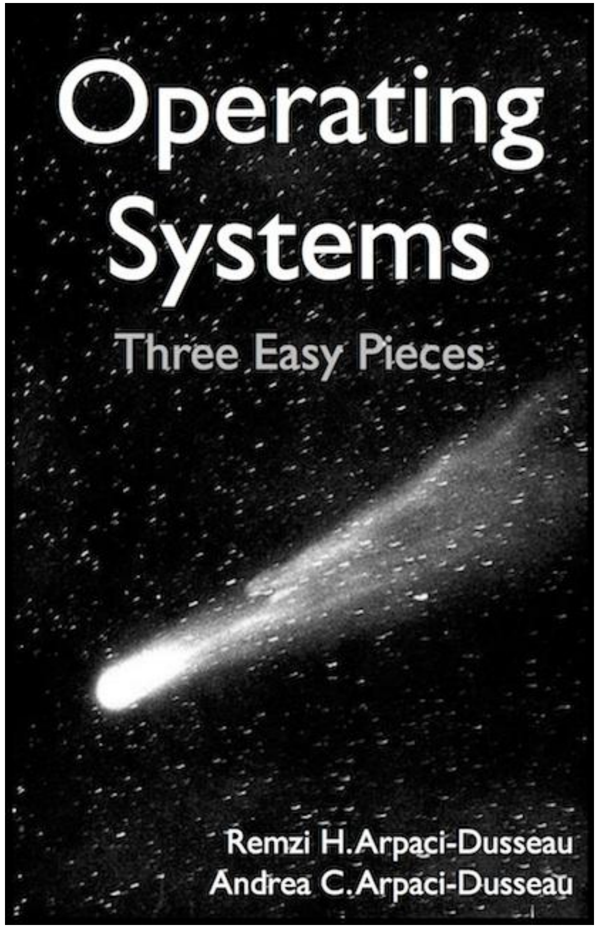
Why do we learn OS?

- ~~To graduate~~
- To understand computer systems better
- To make a better OS or system
  - Functionality
  - Performance/cost
  - Reliability
  - Energy efficiency
- To make new hardware up and running
- To design OS-aware hardware
- Just for fun!



Textbook

- Operating Systems: Three Easy Pieces
  - Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau
  - Arpaci-Dusseau Books
  - November 2023 (Version 1.10)
  - Available (with several options) at http://ostep.org
  - Korean version (based on Version 0.91) is also available at https://github.com/remzi-arpacidusseau/ostep-translations/, but I highly recommend you read the original English version
  - Read Remzi's great article at http://from-a-to-remzi.blogspot.com/2014/01/the-case-for-free-online-books-fobs.html

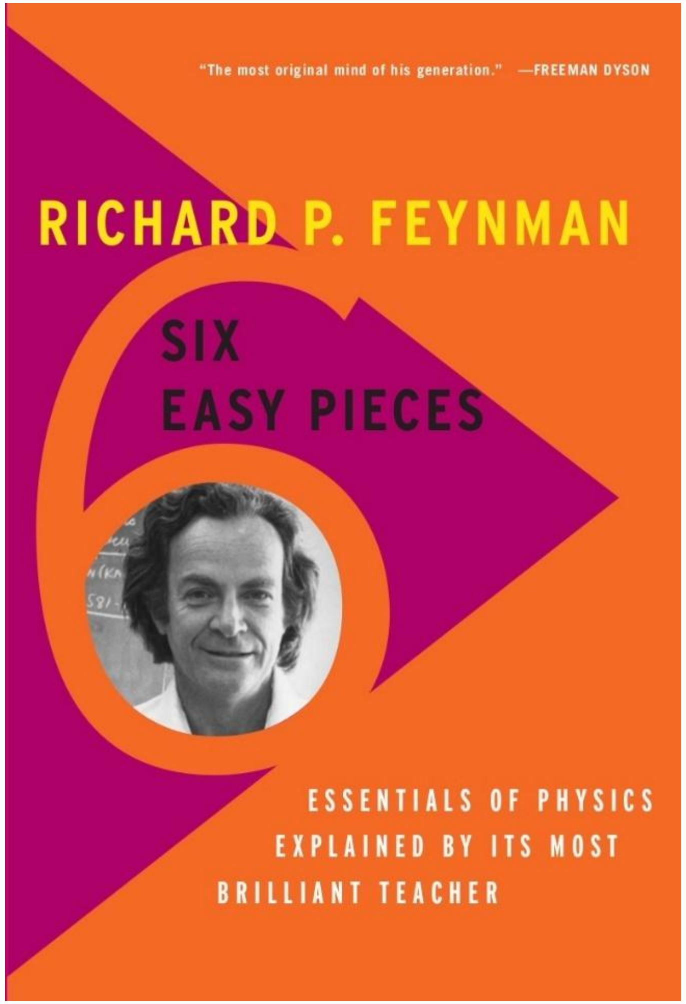


Why Three Pieces?

“… as *Operating Systems are about half as hard as Physics.*”

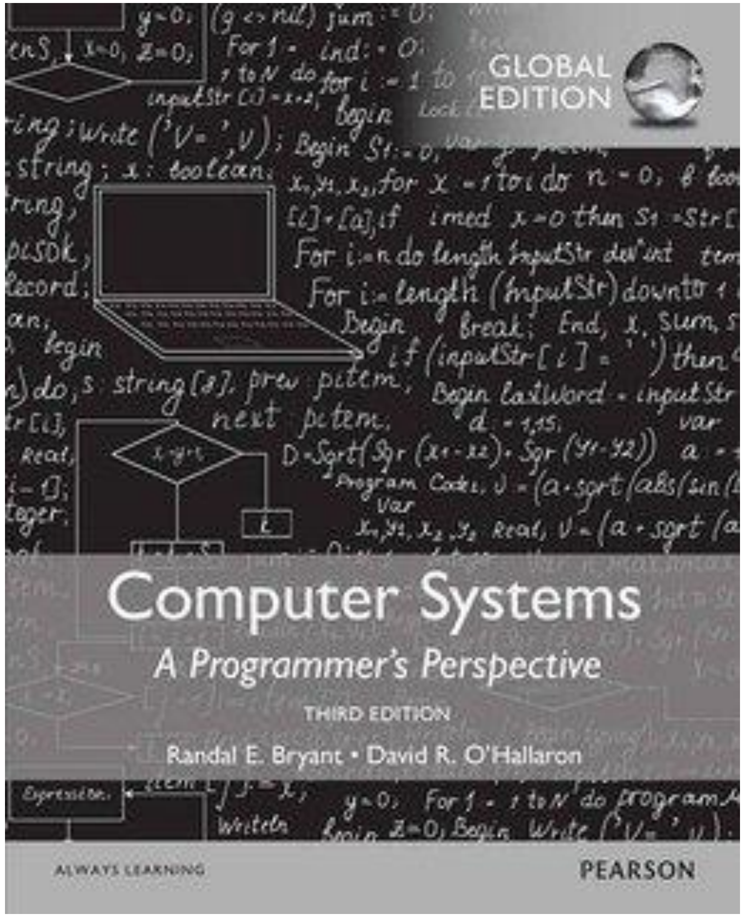
Chap. 1

A Dialogue on the Book


"The most original mind of his generation." —FREEMAN DYSON
RICHARD P. FEYNMAN
SIX EASY PIECES
ESSENTIALS OF PHYSICS EXPLAINED BY ITS MOST BRILLIANT TEACHER

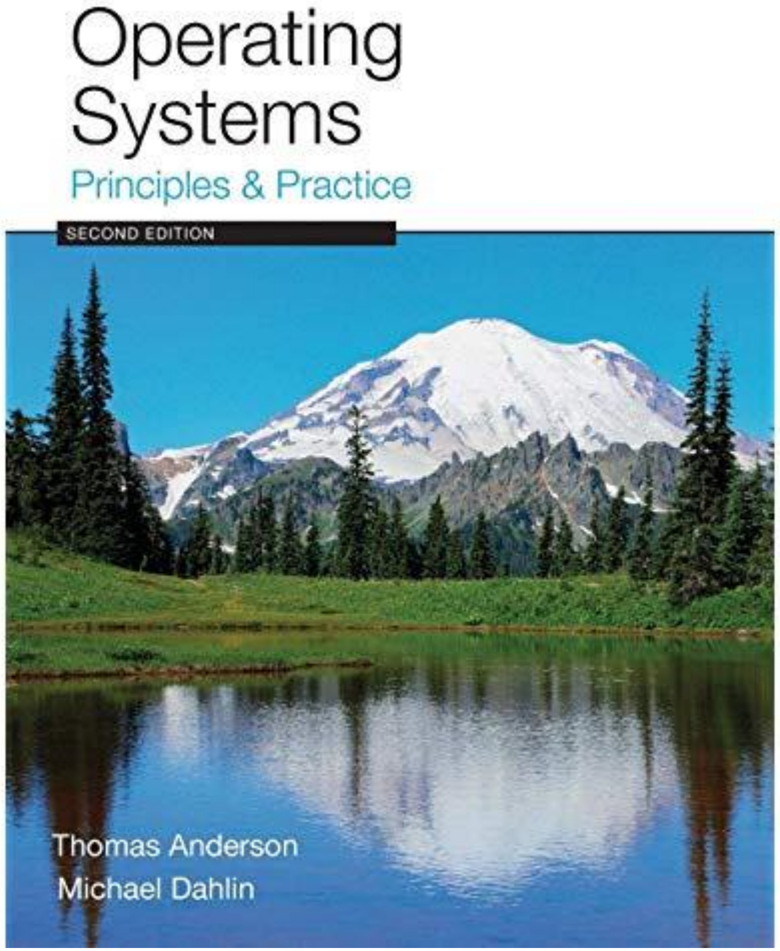
Reference: CSAPP

- Computer Systems: A Programmer’s Perspective
  - Randel E. Bryant and David R. O’Hallaron
  - Third Edition
  - Pearson
  - March 2015
  - http://csapp.cs.cmu.edu



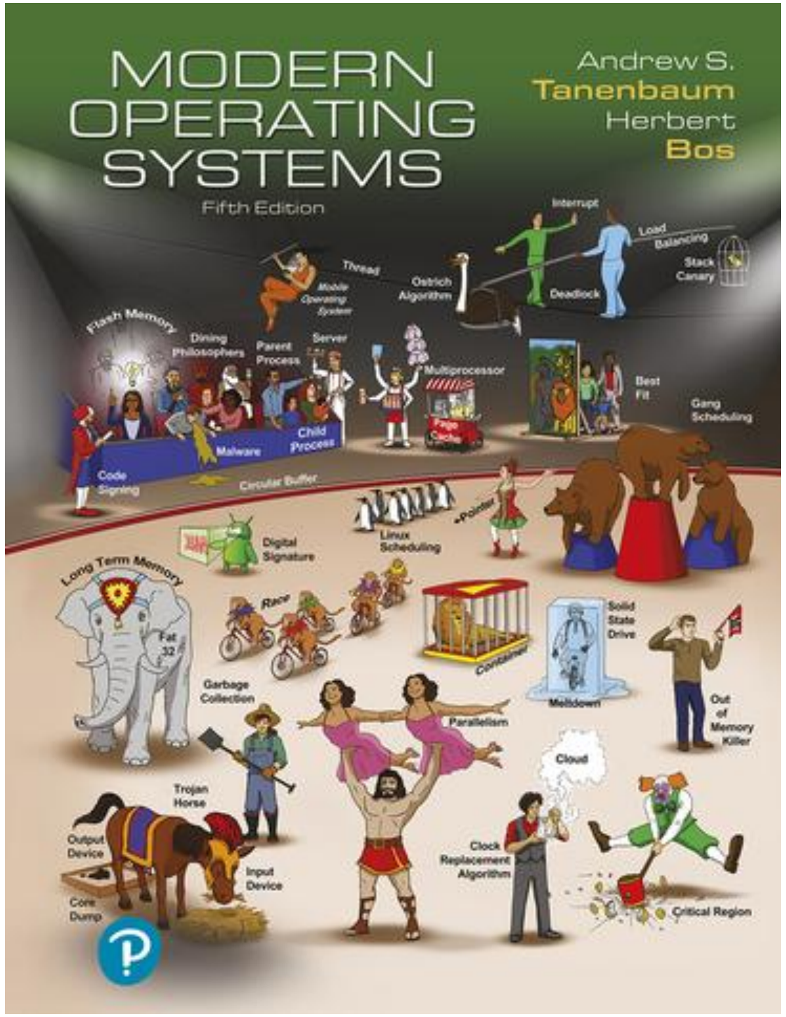
Reference: OSPP

- Operating Systems: Principles and Practice
  - Thomas Anderson and Michael Dahlin
  - Second Edition
  - Recursive Books
  - August 2014
  - http://ospp.cs.washington.edu/



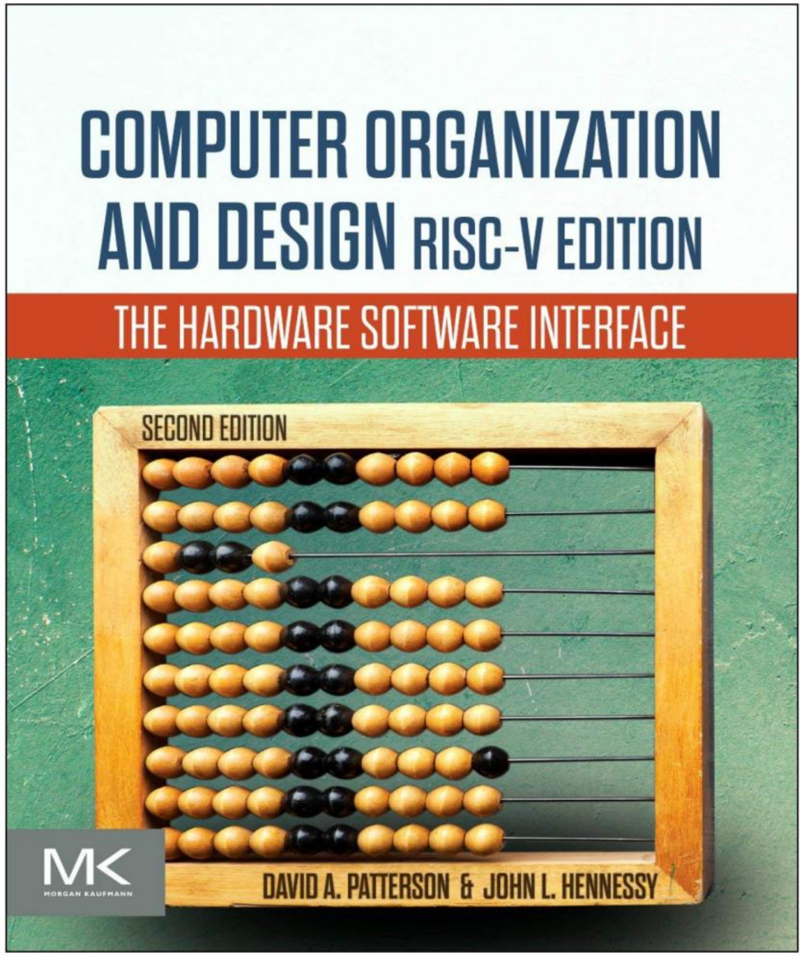
Reference: MOS

- Modern Operating Systems
  - Andrew S. Tanenbaum and Herbert Bos
  - Fifth Edition
  - Pearson
  - October 2022
  - https://www.pearson.com/en-us/subject-catalog/p/modern-operating-systems/P200000003295



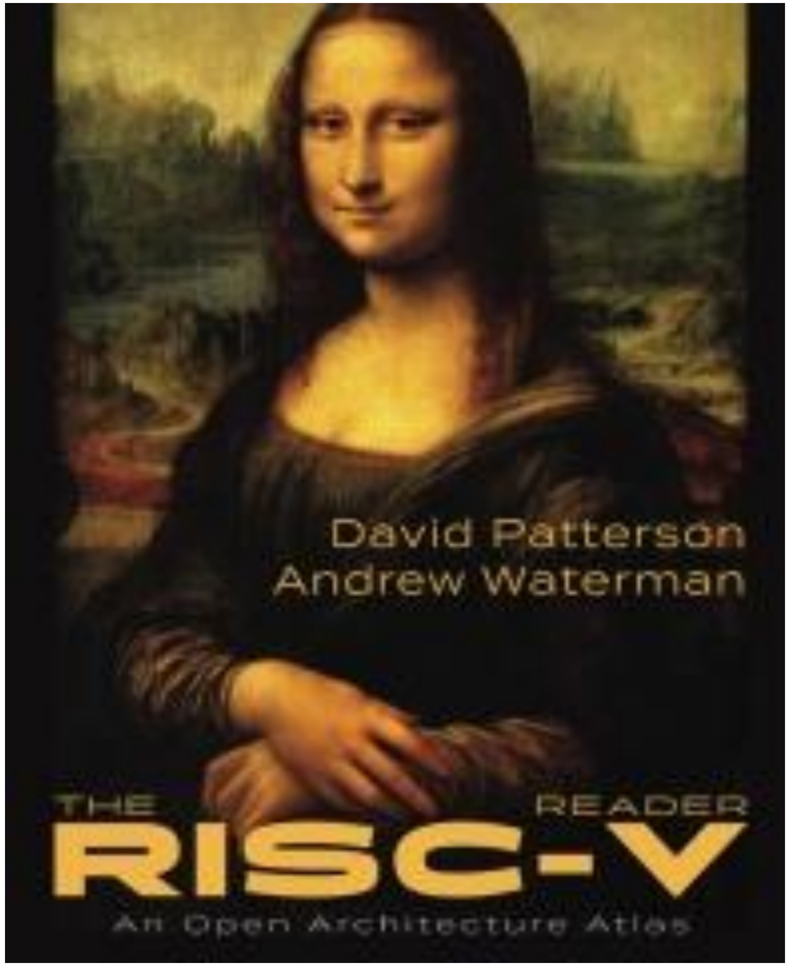
Reference: RISC-V (1)

- Computer Organization and Design: The Hardware/Software Interface (RISC-V Edition)
  - David A. Patterson and John L. Hennessy (Turing Award Recipients in 2017)
  - Second Edition
  - Morgan Kaufmann, 2020
  - http://booksite.elsevier.com/9780128203316/


COMPUTER ORGANIZATION AND DESIGN RISC-V EDITION
THE HARDWARE SOFTWARE INTERFACE
SECOND EDITION
MK MORGAN KAUFMANN
DAVID A. PATTERSON & JOHN L. HENNESSY

Reference: RISC-V (2)

- The RISC-V Reader:
  An Open Architecture Atlas
  - David A. Patterson and Andrew Waterman
  - Strawberry Canyon, 2017
  - http://riscvbook.com/
  - The free Korean (pdf) version is available



Reference: RISC-V (3)

- https://riscv.org/technical/specifications/
  - Volume I: Unprivileged Architecture (v20250508)
  - Volume II: Privileged Architecture (v20250508)

RISC-V®

The RISC-V Instruction Set Manual
Volume I

Unprivileged Architecture

Version 20250508: This document is in ratified state.

RISC-V®

The RISC-V Instruction Set Manual:
Volume II

Privileged Architecture

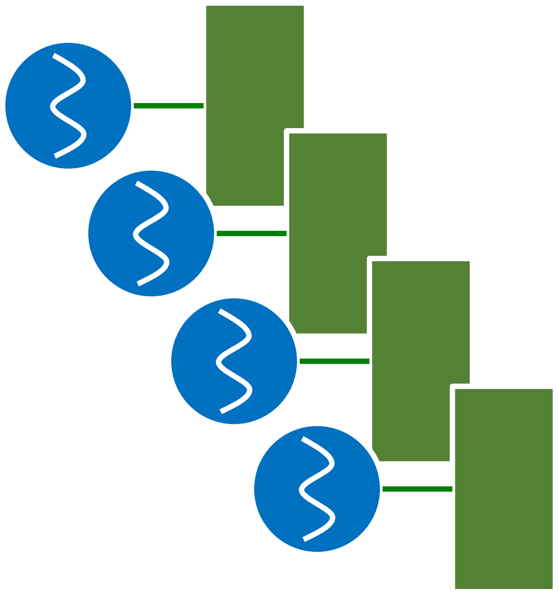
Version 20250508: This document is in Ratified state.

Course Plan

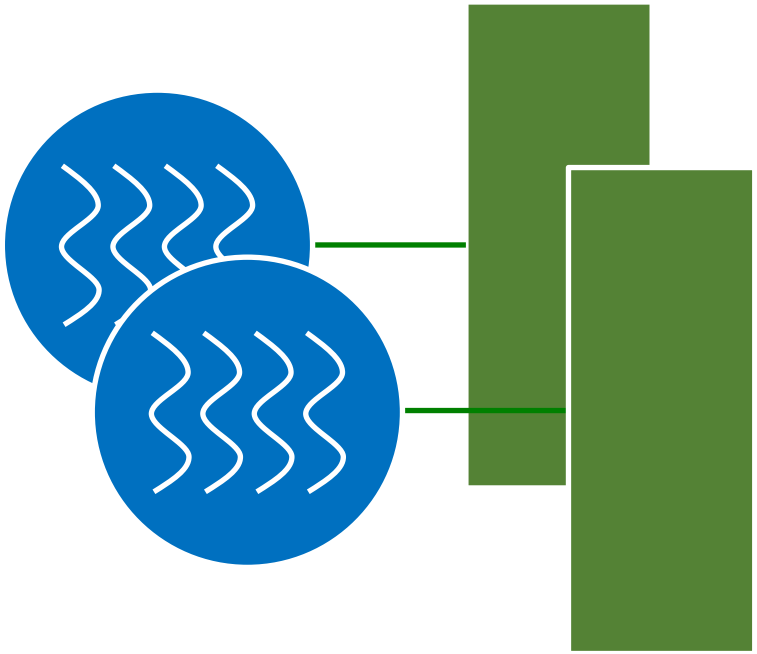
- Lectures
  - General operating system concepts
  - Case study: Linux, xv6
- Hands-on projects
  - Using xv6 instructional OS
  - Based on 64-bit RISC-V architecture

Lectures: Topics

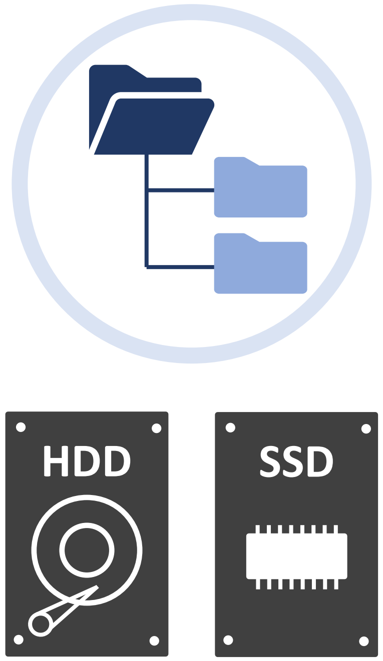
- Virtualization
  - Process
  - CPU scheduling
  - Virtual memory



- Concurrency
  - Threads
  - Synchronization



- Persistence
  - Storage
  - File systems



Projects: xv6

- A teaching OS developed by MIT
  - Port of the Sixth Edition Unix (v6) in ANSI C
  - Originally runs on multi-core x86 systems
  - We will use the version that runs on multi-core 64-bit RISC-V systems
- Why xv6?
  - Code inherited from a real, historical OS!
  - Includes working user-level programs and libraries
  - Small: *only* 6K LOCs (vs. 27+ million LOCs for Linux)
  - Easier to install on modern Linux / MacOS systems using QEMU
  - Easier to extend
  - Easier to understand modern OSes such as Linux

Projects Plan

- We are preparing five project assignments
  - The relative weight of each project can vary, typically increasing monotonically
  - Just for your reference: In the previous semester, there were five projects, and their weights were 1%, 3%, 10%, 13%, and 13% for PA1 – PA5, respectively
- These will be individual projects
- You can use up to 3 *slip* days (Used slip days are irrevocable!)
- Lab sessions
  - A separate online class with TAs
  - Project announcement and Q & A
  - Hints & helps
  - …

Schedule

3 March

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

4 April

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

midterm exam

5 May

S	M	T	W	T	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24 31	25	26	27	28	29	30

6 June

S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

final exam

final project due

Grading Policy (subject to change)

- Exams: 60% (Midterm 25%, Final 35%)
- Projects: 40%

- University policy requires students to attend at least 2/3 of the scheduled classes. Otherwise, you'll fail this course.
- We are NOT using the electronic attendance system

- If you miss any of the exams, you'll fail this course

- Course withdrawal is only allowed before the midterm exam for students who have taken the prerequisite course "System Programming (M1522.000800)"

Cheating Policy

- What is cheating?
  - Copying another student's solution (or one from the Internet) and submitting it as your own
  - Allowing another student to copy your solution (including publicly posting your solution on Github, etc.)
- What is NOT cheating?
  - Helping others use systems or tools
  - Helping others with high-level design issues
  - Helping others debug their code
- Penalty for cheating
  - Severe penalty on the grade (F) and report to dept. committee
  - Ask helps to your TA or instructor if you experience any difficulty!

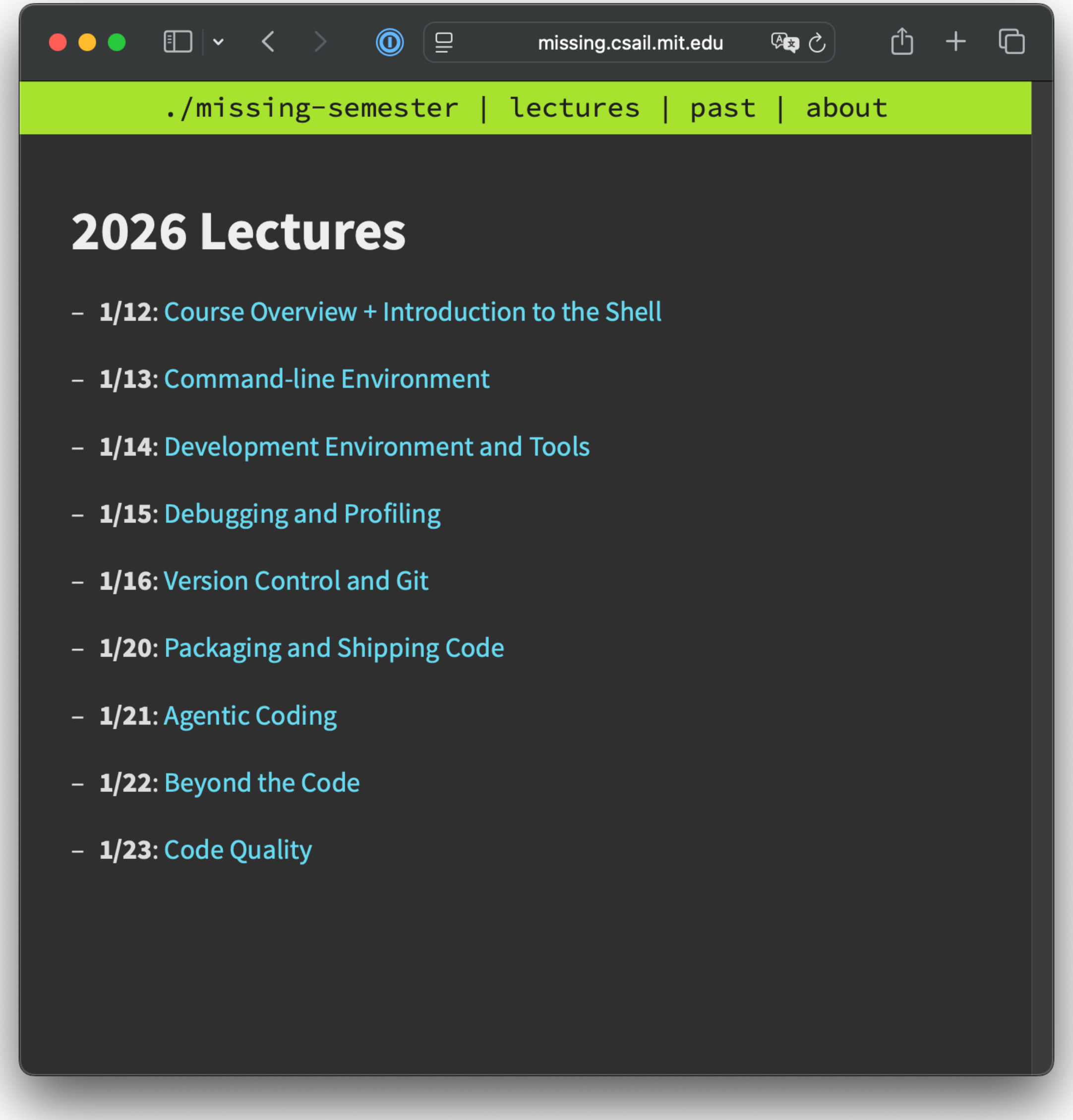
AI Tools Policy for Project Assignments

- AI tools are allowed as assistive tools
  - Brainstorming ideas, debugging hints, improving code readability/documentation
- You must understand what you submit
  - You are responsible for every line of code you submit
  - You must be able to explain your design and implementation choices
- Plagiarism policy still applies
  - Submissions with substantially similar code may be treated as plagiarism
- Exams will check your understanding
  - Midterm/final exams will include questions related to the project assignments
  - These questions are intended to assess whether you genuinely understand the concepts and implementation

Summary

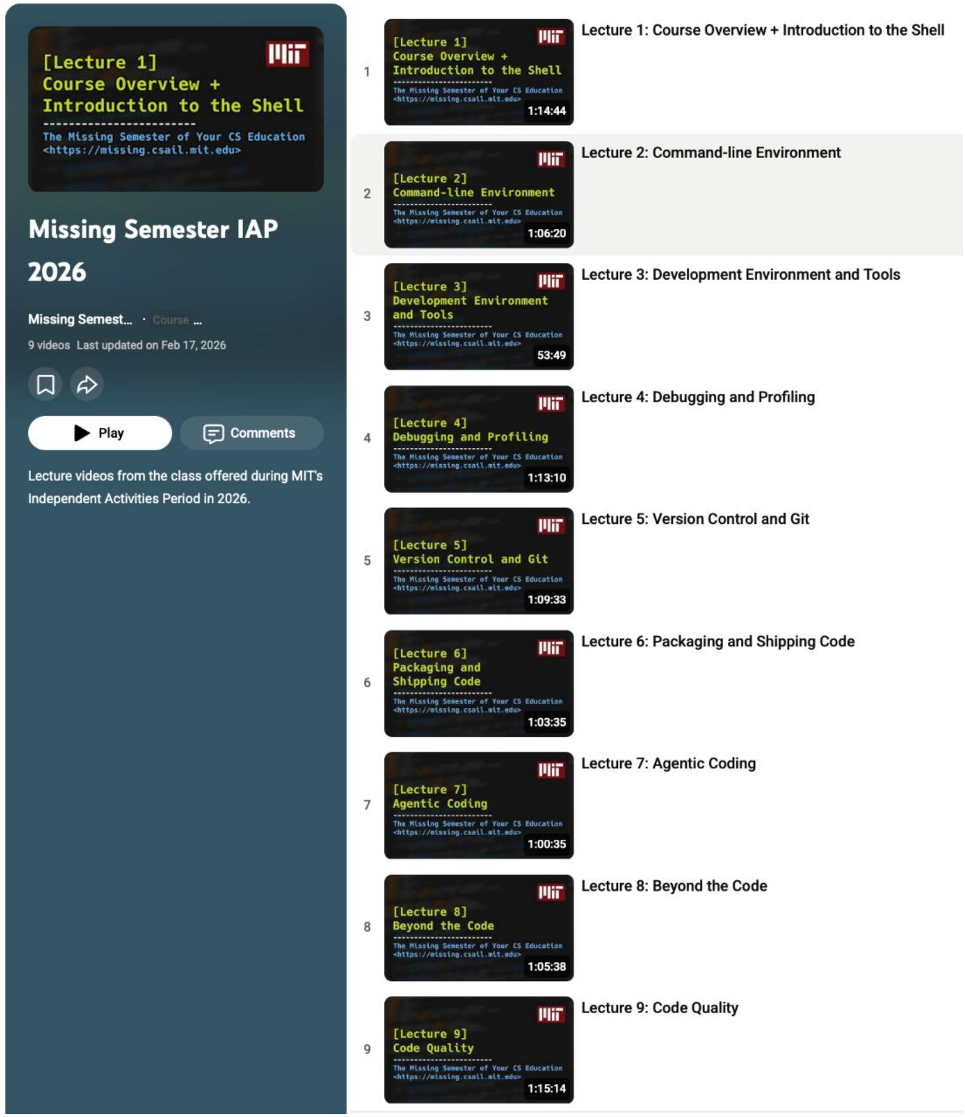
- Understanding OS is essential for a broad spectrum of computer systems research & development
  - Embedded systems
  - Cloud computing
  - Distributed systems
  - Security, …
- It has been one of the toughest courses! Use your time wisely
- Please make sure if you're ready to take this course
- Happy hacking!

Missing Semester @ MIT



2026 Lectures

- **1/12**: Course Overview + Introduction to the Shell
- **1/13**: Command-line Environment
- **1/14**: Development Environment and Tools
- **1/15**: Debugging and Profiling
- **1/16**: Version Control and Git
- **1/20**: Packaging and Shipping Code
- **1/21**: Agentic Coding
- **1/22**: Beyond the Code
- **1/23**: Code Quality



[Lecture 1]
Course Overview +
Introduction to the Shell
The Missing Semester of Your CS Education
<https://missing.csail.mit.edu>
Play
Comments

Missing Semester IAP 2026

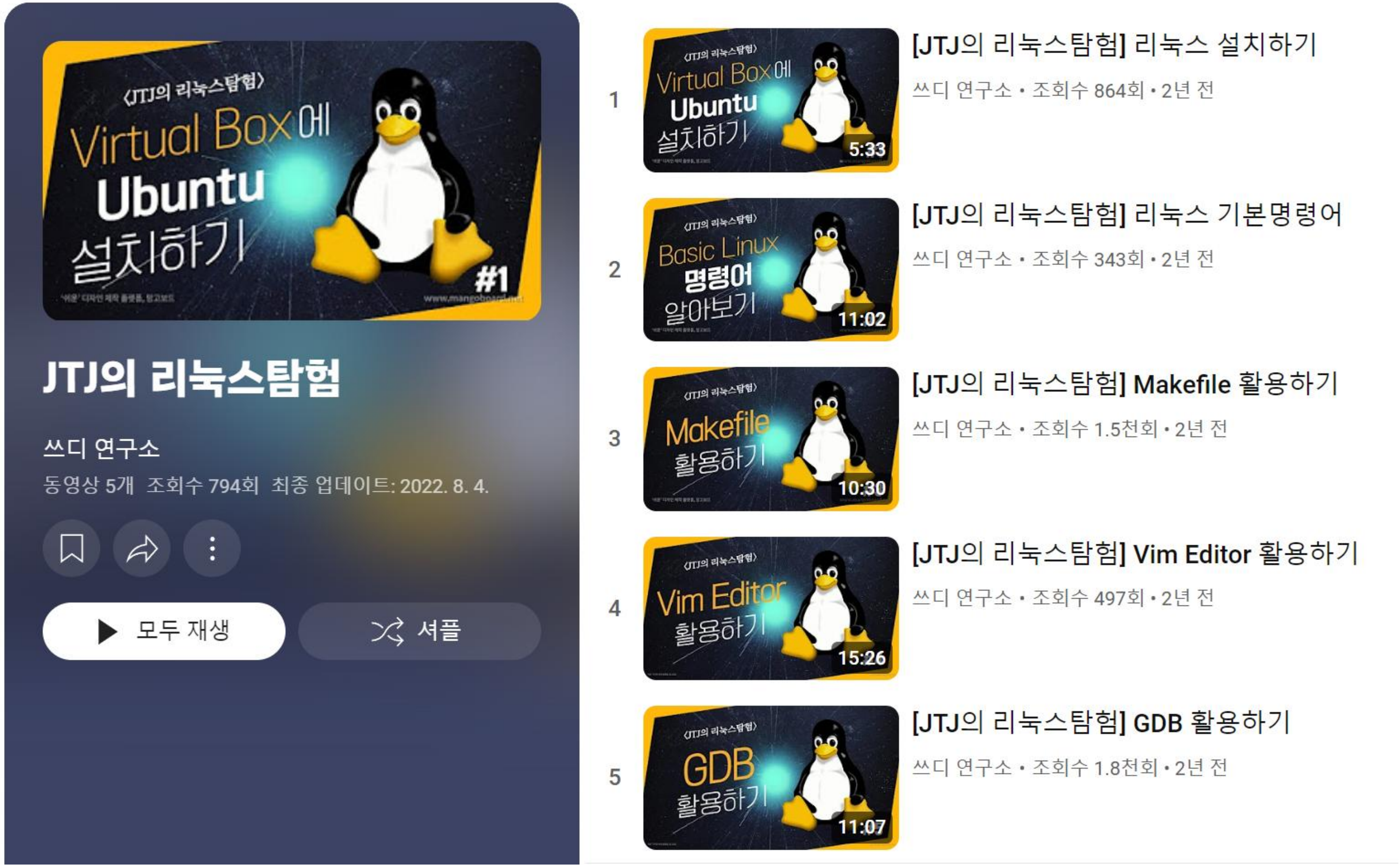
Missing Semest... · Course ...

9 videos Last updated on Feb 17, 2026

Lecture videos from the class offered during MIT's Independent Activities Period in 2026.

1	Lecture 1: Course Overview + Introduction to the Shell	1:14:44
2	Lecture 2: Command-line Environment	1:06:20
3	Lecture 3: Development Environment and Tools	53:49
4	Lecture 4: Debugging and Profiling	1:13:10
5	Lecture 5: Version Control and Git	1:09:33
6	Lecture 6: Packaging and Shipping Code	1:03:35
7	Lecture 7: Agentic Coding	1:00:35
8	Lecture 8: Beyond the Code	1:05:38
9	Lecture 9: Code Quality	1:15:14

One More Thing…



JTJ의 리눅스탐험

쓰디 연구소

동영상 5개 조회수 794회 최종 업데이트: 2022. 8. 4.

1 5:33 [JTJ의 리눅스탐험] 리눅스 설치하기
쓰디 연구소 • 조회수 864회 • 2년 전

2 11:02 [JTJ의 리눅스탐험] 리눅스 기본명령어
쓰디 연구소 • 조회수 343회 • 2년 전

3 10:30 [JTJ의 리눅스탐험] Makefile 활용하기
쓰디 연구소 • 조회수 1.5천회 • 2년 전

4 15:26 [JTJ의 리눅스탐험] Vim Editor 활용하기
쓰디 연구소 • 조회수 497회 • 2년 전

5 11:07 [JTJ의 리눅스탐험] GDB 활용하기
쓰디 연구소 • 조회수 1.8천회 • 2년 전