Tel Aviv University, Faculty of Engineering

Syllabus for the course:

0512.4402: Operating Systems

Winter 2024

Lectures: Gadi Oxman, gdaliaox@tauex.tau.ac.il

Recitations: Yoav Chachamovitz chachamovitz@mail.tau.ac.il

Credit points: 3 hours lecture, 1 hour recitation

Perquisites: 0512.1820 (Programming 2 – C), 0512.4400 (Computer Organization), 0512.2510 (Data Structures and Algorithms).

The aim of the course is to study operating systems principles and to improve programming skills in C. The material includes the following topics. The concept of operating systems. The concept of a process.The layering approach. The hardware/software interface. The application/OS interface: system calls. The interface/encapsulation approach. CPU scheduling: measures, preemption, some policies. Inter-Process communication mechanisms. Synchronization: hardware solutions; software solutions. Mutexes, semaphores, monitors. Deadlocks: detection, prevention. The memory hierarchy. Segmentation, paging. Caching algorithms. Virtual memory. Introduction to IO devices. File systems: organization and implementation on disks. Communication: TCP/IP.Client/server architecture.

The course includes extensive programming in the C programming language under the Linux and XV6 operating system.

Course requirements:

Requirement	**Details**	**Grade percentage**
Programming exercises	Submit in pairs.	20%
Final exam	Passing the exam is required to pass the course.	80%

Bibliography:

[1] Andrew S. Tanenbaum, Herbert Bos, "Modern Operating Systems", Pearson, 4th edition, 2014

[2] Avi Silberschatz, Peter Baer Galvin, Greg Gagne, “Operating System Concepts”, Wiley, 10th edition, 2018

[3] Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau, “Operating Systems: Three Easy Pieces”, available online: https://pages.cs.wisc.edu/~remzi/OSTEP/

[4] Andrew S. Tanenbaum, David J. Wetherall, “Computer Networks”, Prentice Hall, 5th edition, 2010

[5] Russ Cox, Frans Kasshoek, Robert Morris, “The xv6 teaching operating system” from MIT, available online:

xv6 source code (for x86): https://github.com/mit-pdos/xv6-public

Annotated source code: https://pdos.csail.mit.edu/6.828/2018/xv6/xv6-rev11.pdf

Book: https://pdos.csail.mit.edu/6.828/2018/xv6/book-rev11.pdf

Tentative Schedule

	Subject	Date	Book
1	Introduction	Nov 6	[1] Chapter 1 [2] Chapters 1,2
2	Processes & Threads I	Nov 13	[1] Chapter 2 [2] Chapters 3-7
3	Processes & Threads II	Nov 20	[1] Chapter 2 [2] Chapters 3-7
4	Processes & Threads III	Nov 27	[1] Chapter 2 [2] Chapters 3-7
5	Networks I	Dec 4	[1] Chapter 8 Section 3 [2] Chapter 19 [4] Chapt
6	Networks II	Dec 11	[1] Chapter 8 Section 3 [2] Chapter 19 [4] Chapt
7	File Systems	Dec 18	[1] Chapter 4 [2] Chapters 13-15
8	Memory Management I	Dec 25	[1] Chapter 3 [2] Chapters 9,10
9	Memory Management II	Jan 1	[1] Chapter 3 [2] Chapters 9,10
10	Memory Management III	Jan 8	[1] Chapter 3 [2] Chapters 9,10
11	Deadlocks	Jan 15	[1] Chapter 6 [2] Chapter 8
12	Input/Output	Jan 22	[1] Chapter 5 [2] Chapter 12
13	Virtualization	Jan 29	[1] Chapter 7 [2] Chapter 18
	Exam Moed A	Feb 21	
	Exam Moed B	March 28	

Detailed Topics

Introduction to OS

Operating Systems introduction: History of OS. Classes of OS. Concepts: Processes, Address Spaces, Files, I/O, Protection, The Shell. System Calls. OS structure.

Processes and Threads

The Process: Model, Creation, Termination, Hierarchies, States, Implementation, Modeling Multiprogramming.

The Thread: Usage, Classical Model, POSIX Threads. Implementation: User Space, Kernel Space, Hybrid. Scheduler Activations. Pop-Up Threads. Making Single-Threaded Code Multithreaded.

Interprocess Communication: Race Conditions. Critical Regions. Mutual Exclusion with Busy Waiting. Sleep and Wakeup. Semaphores. Mutexes. Monitors. Message Passing. Barriers. Avoiding Locks: Read-Copy-Update.

Scheduling: Batch Systems. Interactive Systems. Real-Time Systems. Policy vs Mechanism. Thread Scheduling.

Classical IPC Problems: The Dining Philosophers. The Readers and Writers.

Deadlocks

Resources: Preemptable. Nonpreemptable. Acquisition. Condition for Resource Deadlocks. Deadlock Modeling. The Ostrich Algorithm. Deadlock Detection. Deadlock Recovery. Deadlock Avoidance: Resource Trajectories, Safe and Unsafe States, The Banker's Algorithm. Deadlock Prevention. Other Issues: Two-Phase Locking. Communication Deadlocks. Livelock. Starvation.

Memory Management

Early computers. Address spaces abstraction: Swapping, Managing Free Memory.

Virtual Memory: Paging, Page tables, Speeding Up Paging, TLB, Page Tables for Large Memories.

Page Replacement Algorithms: Optimal, Not Recently Used, FIFO, Second-Chance, Clock Page, LRU, Working Set, WSClock.

Design Issues: Local vs Global Allocation. Load Control. Page Size. Separate Instruction and Data Spaces. Shared Pages. Shared Libraries. Mapped Files. Cleaning Policy. Virtual Memory Interface.

Implementation Issues: OS involvement. Page Fault Handling. Instruction Backup. Locking Pages. Backing Store. Separation Policy and Mechanism.

File Systems

Files: Naming. Structure. Types. Access. Attributes. Operations. File System Calls.

Directories: Single Level. Hierarchical. Path Names. Directory Operations.

Implementation: Layout. Files. Directories. Shared Files. Log Structured FS. Journaling FS. Virtual FS.

Management and Optimization: Disk Space. Backups. Consistency. Performance. Defragmenting Disks.

Examples: MS-DOS, Unix V7, CD-ROM.

Input/Output

Hardware: Devices. Controllers. Memory Mapped I/O. Direct Memory Access. Interrupts.

Software: Goals. Programmed I/O. Interrupt-Driven I/O. I/O Using DMA.

I/O Software Layers: Interrupt Handlers. Device Drivers. Device Independent I/O. User Space I/O Software.

Disks: Hardware. Formatting. Arm Scheduling Algorithms. Error Handling. Stable Storage.

Clocks: Hardware. Software. Soft Timers.

User Interfaces: Keyboard, Mouse, Monitor, Input/Output Software.

Case Study: The Linux OS

Unix History. Overview. Processes. Memory Management. Input/Output. File System. Security.