COURSE DESCRIPTION

Dept., Number	CS450	Course Title	Introduction to Operating Systems
Semester hours	3	Course Coordinator	Dr. Xian-He Sun, Professor

Current Catalog Description

Introduction to operating system concepts—including system organization for uniprocessors and multiprocessors, scheduling algorithms, process management, deadlocks, paging and segmentation, files and protection, and process coordination and communication. Prerequisites: (CS 331 and CS 350) or (CS 331 and ECE 242) or (CS 401 and CS 402) or CS 403. (3-0-3) (T)

Textbook

Silberschatz, Adam, Peter Galvin, and Greg Gagne. "Operating System Concepts, 7th Edition." John Wiley & Sons, 2004.

Lions, John. "Lions' Commentary on UNIX, 6th Edition." Annabooks, 1996.

References

Kernighan, Brian W., and Dennis M. Ritchie. "The C Programming Language", 2nd Edition. Prentice Hall, 1988.

Course Outcomes

Students should be able to:

- Explain the range of requirements that a modern operating system has to address.
- Define the functionality that a modern operating system must deliver to meet a particular need.
- Articulate design tradeoffs inherent in operating system design.
- Explain the concept of a logical layer.
- From the perspective of building operating systems, explain the benefits of building these layers in a hierarchical fashion.
- Describe how the resources of the computer system are managed by software.
- Relate system state to user protection.
- Justify the presence of concurrency within the framework of an operating system.
- Demonstrate the potential run-time problems arising from the concurrent operation of many (possibly a dynamic number of) tasks.
- Summarize the range of mechanisms (at an operating system level) that can be employed to realize concurrent systems and be able to describe the benefits of each.

- Explain the different states that a task may pass through and the data structures needed to support the management of many tasks.
- Compare and contrast the common algorithms used for both preemptive and non-preemptive scheduling of tasks in operating systems.
- Describe relationships between scheduling algorithms and application domains.
- Investigate the wider applicability of scheduling in such contexts as disk I/O, networking scheduling, and project scheduling.
- Introduce memory hierarchy and cost-performance tradeoffs.
- Explain what virtual memory is and how it is realized in hardware and software.
- Examine the wider applicability and relevance of the concepts of virtual entity and of caching.
- Evaluate the trade-offs in terms of memory size (main memory, cache memory, auxiliary memory) and processor speed.
- Defend the different ways of allocating memory to tasks on the basis of the relative merits of each.
- Summarize the features of an operating system used to provide protection and security, and describe the limitations of each of these.
- Summarize the full range of considerations that support file systems.

Relationship between Course Outcomes and Program Outcomes

The following Program Outcomes are supported by the above Course Outcomes:

- a. An ability to apply knowledge of computing and mathematics appropriate to the discipline
- h. Recognition of the need for, and an ability to engage in, continuing professional development
- j. An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices
- l. Be prepared to enter a top-ranked graduate program in Computer Science.

Prerequisites by Topic

To be successful in this course you should have substantial programming experience in a high level language (C is ideal) with direct access to the underlying operating system's system call interface. You should be, at minimum, adept at making use of the language's facilities for process control, memory management, I/O, file management, and IPC. Experience with some form of assembly language is also required.

Major Topics Covered in the Course

1. Processes, Threads, and Context Switching	5 hours
2. System Calls, Interrupts, and Exceptions	5 hours

3. Kernel and User Modes	5 hours
4. Scheduling	5 hours
5. IPC	5 hours
6. Address spaces, virtual memory and memory management	5 hours
7. I/O and device management	5 hours
8. File systems	5 hours
9. Concurrency	5 hours
Midterm Exam	
Final Exam	-
	45 hours

Assessment Plan for the Course

End of every semester Course Objective Assessments by CS department. End of semester Course Evaluations by IIT. Reviewed every Spring semester by CS Undergraduate Studies Committee for possible updates in the following Fall. Once every 4-5 years a detailed review of all materials for the course is made by the CS Undergraduate Studies Committee.

How Data in the Course is Used to Assess Program Outcomes (unless adequately covered already in the assessment discussion under Criterion 4)

See the assessment discussion under Criterion 4

For a computer science program

Estimate Curriculum Category Content (Semester hours)

Area	Core	Advanced	Area	Core	Advanced
Algorithms		1	Software design		
Data structures		1	Concepts of programming languages		1