

CS 5631: Operating Systems (4)**Catalog Description:**

Operating system as a resource manager. Processor management and scheduling, deadlocks, concurrency, memory management, protection and security, as applied in modern operating systems. Concepts are illustrated via laboratory assignments which heavily emphasize concurrency.

Textbook: Silberschatz, Galvin, and Gagne, *Operating Systems Concepts, 8th Ed.*, John Wiley & Sons, 2009.

Course Goals:

The objective of this course is to present the operating system as the resource manager of today's large, complex computing systems. The resources to be managed include memory, processors, devices, and information. The emphasis of this course is processor, memory and virtual memory management; concurrency; scheduling; deadlocks; and security and protection. The most important of these concepts are illustrated by programming assignments and case studies.

Prerequisites by Course & Topic

CS 2511: Software Analysis & Design – proficiency in object-oriented design and coding, a systematic approach to testing and debugging, ability to work successfully on a team

CS 2521: Computer Organization & Architecture - understanding how programs and data are stored and represented in a computer system

Major Topics Covered in the Course

- Processes & Scheduling
- Concurrency
- Deadlocks
- Memory Management
- Virtual Memory Management
- Protection & Security
- Case Studies
- File Systems
- Mass Storage of I/O Systems

Class/Laboratory Schedule: Lecture: 3 hours per week, Laboratory: 1

Course Outcomes

1. Understand the basic issues of concurrency (synchronization, mutual exclusion), using processes and threads as basic units.
 - a. Analyze and construct correct solutions for common concurrent programming problems.
 - b. Illustrate by implementing a Java program for each such solution.
2. Apply basic OS vocabulary and understand the evolutionary nature of operating systems.
 - a. Illustrate orally in class discussion and in written form on both homework and tests the use of appropriate OS terminology.
 - b. Demonstrate the incremental changes made to operating systems over the years which are realized in the operating systems of today.
3. Analyze under what conditions deadlock occurs, understand how to detect and prevent it.
 - a. Given a set scenario, correctly apply the Banker's algorithm to determine whether or not deadlock occurs.
4. Comprehend how various memory management systems evolved over time (culminating with current day systems, i.e., demand paging and segmentation), their designs and implementations.
 - a. Demonstrate the modifications made to operating systems over the years which correlate largely with underlying improvements in hardware.
 - b. Illustrate, by correctly solving specific problems, the mapping from physical to virtual memory under various memory management schemes.

5. Understand the measures used to enhance security in operating systems.
 - a. Understand program threats, how they are constructed, and how they may be defeated.
 - b. Demonstrate the features incorporated within hardware and operating systems to improve protection and security.
 - c. By examining case studies, see how these features are used in today's systems (Windows, Linux, etc.).
6. Apply the ability to work in a team environment to solve challenging problems.
 - a. Meet with fellow team members to design and properly implement the solutions to problems.

Relationship to Program Outcomes

Students must have completed systems design and analysis and computer organization and architecture in order to take CS 5631. This course contributes to meeting the following program outcomes:

2. *Students can design, develop, and analyze significant software systems.*
 Students develop data structures and implement complex algorithms to solve concurrent programming problems, skills that can be used to develop other software systems. Course outcomes 1-5 map to this program outcome.
3. *Students understand the fundamentals of computer organization and architecture, data structures and related algorithms, and programming languages.*
 Students increase their knowledge of algorithms and data structures by solving challenging concurrent programming problems and learning the intricacies of virtual memory management. Course outcomes 1-5 map to this program outcome.
4. *Students can apply computer science principles and practices to a variety of problems.*
 Students have the opportunity to write concurrent code using threads, providing both mutual exclusion and synchronization. Course outcomes 1-5 map to this program outcome.
5. *Students can work independently and also work effectively in teams.*
 Students gain significant experience with working in a team environment as part of the file system team project undertaken in the course. Course outcome 6 maps to this program outcome.

Assessment Plan for Course:

This course is assessed every third year by the instructor and a course assessment document covering all of the course outcomes and their effect on the program outcomes is prepared.

Estimate CSAB Category Content

	CORE	ADVANCED		CORE	ADVANCED
Data Structures			Computer Organization and Architecture Concept of Programming Languages		3
Algorithms					
Software Design		1			

Coordinator/Prepared by: C. Prince