Course Number : CO327
Course Title : Operating Systems
Credits : 3
Core/Elective : Core
Prerequisites : CO224: Computer Architecture; CO322: Data Structures and Algorithms

Aims/Objectives:

The aim of this course is to expose students to the essential and well established operating system (OS) concepts, their application and the issues related to realizing them on modern hardware.

Intended Learning Outcomes (ILOs)

Knowledge: At the end of this course, student will be able to;

- Use operating system primitives; files, threads, synchronization primitives; in user level applications.
- Able to explain how OS concepts such as threads, processors and address spaces, synchronization are implemented using modern hardware.
- Analyze the tradeoffs involved with realizing threads, processes, address spaces and file systems on modern hardware.
- Able to implement a simple user-level thread management package.
- Able to explain the UNIX file permissions and use them in system administration tasks.
- Analyze the tradeoffs in file system implementations.
- Explain the benefits of virtualization in system software.
- Able to solve classical synchronization problems and use these results to solve real world problems.
-

Skill: At the end of this course, student will be able to;

- Able to write simple assembly code for selected target architecture.
- Able to read and understand assembly code.
- Able to link assembly and C language code by adhering to the calling convention.
- Able to design user-level applications to take advantage of parallel processing.
- Use of API in software development.
-

Attitude:

- Discourage trial and error programming.
- Willingness to work with relatively large code base.

Textbooks and References:

- A. Tanenbaum, Modern Operating Systems
- Vahalia, Unix Internals: The new frontiers

Topic	Time Allocated / hours			
	L	T	P	A
Introduction: The role and the purpose of an operating system (OS), history of OS, OS structuring methods, typical OS abstractions	2			
OS abstractions: **U**sing files, POSIX threads, processors, synchronization primitives effectively in user-level applications.			2	4
Threads: Execution of a program on hardware, visible register set, calling conventions, realization of threads at user-level, realization of threads at kernel-level, mapping between user and kernel-level threads, scheduling of threads. POSIX and other selected thread API.	6		2	4
Processes: Review of hardware MMU, processes as the unit of protection, context switching, memory management policies, virtual memory address space, page tables, paged memory, cache memory, working set, thrashing.	6		2	2
Synchronization: Introduction of common sychronization primitives such as locks, semaphores etc. Use of sychronization primitives in classical synchronization problmes. Mutually exclusive execution and realization of mutual exclution using sychronization primitives.	6		4	2
File systems: File operations, access methods, security, virtual file system, implementation techniques and their tradeoffs, UNIX access control.	5			2
IO subsystem: Device access, interrupt handling, device drivers, API for device access, DMA, IO-MMUs, UNIX drivers.	4		2	
OS implementation methods: Design issues, kernel structuring methods, virtual machine monitors, small kernels.	2			
Self-study: Modern trends in OS design and implementation and their industrial applications				2
Total	31		12	16

L = Lectures, T = Tutorial classes, P = Practical classes, A – Assignments

Assessment	**Percentage Marks**
Continuous Assessments	**60**
Tutorials/ Assignments/ Practical	15
Assignment (Project)	25
Mid-semester examination	20
End of Semester Evaluation	**40**
End-Semester examination	40