Course Code: 22CS401

LINUX PROGRAMMING

Instruction	:	3 Periods/week	Continuous Internal Evaluation	:	40 Marks
Tutorial	:	-	Semester End Examination	:	60 Marks
Credits	:	3	Semester End Exam Duration	:	3 Hours

Course Objectives:

1: To develop the skills necessary for systems programming.

2: To model asynchronous event handling.

3: To establish efficient communication between two asynchronous processes.

4: To design various client & server communication models

Unit I – File System and Directory Structure

Files: Files concept, File System Structure, I-nodes, File Attributes, File types, Library functions, the standard I/O and formatted I/O in C, stream errors, kernel support for files, System calls, file descriptors, low level file access- File structure related system calls(File APIs), file and record locking, file and directory management – Directory file APIs, Symbolic links & hard links.

Unit II - Process and Signals

Process: Process concept, Kernel support for process, process attributes, process control - process creation, waiting for a process, process termination, zombie process, orphan process APIs.

Signals: Introduction to signals, Signal generation and handling, Kernel support for signal, Signal function, unreliable signal, reliable signal, kill, raise, alarm, pause, abort, sleep functions.

Inter Process Communication: Introduction to IPC, Pipes, FIFOs.

Unit III - IPC, Message Queues, Semaphores, Shared Memory and Socket Programming

Message Queues – Kernel support for messages, Unix system V APIs for messages, client/server example.

Semaphores: Kernel support for semaphores, Unix system V APIs for semaphores.
Shared Memory: Kernel support for shared memory, Unix system V APIs for shared memory, semaphore and shared memory example.

Sockets: Introduction to Sockets, Socket Addresses Structures, Byte ordering and manipulation functions, Socket related system calls for TCP sockets- Socket, connect, bind, listen, fork, exec, and close, Implementation of concurrent server, TCP Client Server programs, Normal startup, terminate and signal handling server process termination. TCP Client-Server Program and UNIX domain Sockets.

Unit IV - Socket Programming, Multithreaded Programming

Socket Options, Server and service models, Super Server, Elementary UDP Sockets: Introduction UDP Echo server function, lost datagram, summary of UDP example, Lack of flow control with UDP.

Multithreaded Programming: Differences between threads and processes, Thread structure and uses. Threads and Lightweight Processes, POSIX Thread APIs, Creating Threads, Thread Attributes, Thread Synchronization with Condition Variables and with Mutexes, Example programs.

Unit V –Advanced I/O

I/O Multiplexing and Socket options: I/O Models, Select Function, poll function.
Record Locking, Readn and Writen functions, Scatter and Gather IO, Memory Mapped IO. Asynchronous IO and Async Options. Remote login overview and RPC Transparency issues. FTP server configuration.

Course Outcomes: At the end of the course, the student should be able to

CO 1 : Make use of well-defined Korn shell utilities and develop menu driven Text processing Application.
CO 2 : Appreciate process abstraction and asynchronous event handling using signals.
CO 3 : Implement IPC Mechanisms, Messages Queues and synchronize the access patterns as a shared memory.
CO 4 : Design concurrent server programs based on various design alternatives.
CO 5 : Implement I/O multiplexing mechanisms.

Textbooks:

1. Advanced Programming in the UNIX Environment, W Richard Stevens and Stephen A Rago, 3rd Edition, Addison Wesley / Pearson Education Inc., 2013.
2. Unix System Programming using C++, T.Chan, PHI, 1999

References:

1. Unix Network Programming, W R Stevens, PHI, 2003.
2. Unix Internals: The New Frontiers, Uresh Vahalia, Pearson Education, 1995
3. Unix for Programmers and Users, Graham Glass and King Ables, 3rd Edition, Pearson Education, 2003.