OPERATING SYSTEMS LABORATORY

OBJECTIVE:

This lab complements the operating systems course. Students will gain practical experience with designing and implementing concepts of operating systems such as system calls, CPU scheduling, process management, memory management, file systems and deadlock handling using C language in Linux environment.

OUTCOMES:

Upon the completion of Operating Systems practical course, the student will be able to:

1. **Understand** and implement basic services and functionalities of the operating system using system calls.
2. **Use** modern operating system calls and synchronization libraries in software/ hardware interfaces**.**
3. **Understand** the benefits of thread over process and implement synchronized programs using multithreading concepts.
4. **Analyze** and simulate CPU Scheduling Algorithms like FCFS, Round Robin, SJF, and Priority.
5. **Implement** memory management schemes and page replacement schemes.
6. **Simulate** file allocation and organization techniques.
7. **Understand** the concepts of deadlock in operating systems and implement them in multiprogramming system.

List of Programs:

1. Write C programs to simulate the following CPU scheduling algorithms:

   a) Round Robin b) SJF

2. Write C programs to simulate the following CPU scheduling algorithms:

   a) FCFS b) Priority

3. Write C programs to simulate the following File organization techniques:

   a) Single level directory b) Two level c) Hierarchical

4. Write C programs to simulate the following File allocation methods:

   a)Contiguous b)Linked c)Indexed

5. Write a C program to copy the contents of one file to another using system calls.

6. Write a C program to simulate Bankers Algorithm for Dead Lock Avoidance

7. Write a C program to simulate Bankers Algorithm for Dead Lock Prevention

8. Write C programs to simulate the following page replacement algorithms:

a) FIFO b) LRU c) LFU

9. Write C programs to simulate the following techniques of memory management:

a) Paging b) Segmentation

10. Write a C program to implement the ls | sort command. (Use unnamed Pipe)

11. Write a C program to solve the Dining- Philosopher problem using semaphores.

12. Write C programs to implement ipc between two unrelated processes using named pipe.