|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Operating Systems Lab | L | T | P | C |
| **Version 1.0** |  | 0 | 0 | 2 | 1 |
| **Pre-requisites/Exposure** | Knowledge of data structure and algorithms with programming in C. | | | | |
| **Co-requisites** | - | | | | |

**Course Objectives:**

To implement basic functionalities of Operating system

**Course Outcomes:**

CO1. Apply system programming in process management and I/O management.

CO2. Implement various CPU scheduling algorithms and memory management techniques.

CO3. Implement various deadlock prevention, avoidance, detection and recovery techniques.

**Catalog Description**

Operating Systems mainly covers five types of programs, a) process management, b) I/O management, c) Scheduling, d) memory management and e) File systems management. The theory covers the algorithmic aspects as well as conceptual aspects of all these. The lab sessions are designed to bring in the prior knowledge of programming into implementing such algorithms in a small way. In doing so the students will appreciate the concepts and the complexities involved in implementing such algorithms. In this way the students will be able to appreciate as well as understand the inner intricacies of designing data structures and using these to get a proper implementation of OS programs and in the long run the students will use these skills in many other aspects of Coding. The lab experiments of implementing system calls in UNIX is designed so that the students learn and appreciate the tasks and intricacies of a system programmer.

**List of Experiments**

**Experiment No 1: System calls & I/O System calls**

i) To write programs to perform following operations in UNIX:

a) Process Creation

b) Executing a command

c) Sleep command

d) Sleep command using get pid

e) Signal handling using kill

f) Wait command

ii) To write programs to perform following operations in UNIX:

a) Reading from a file

b) Writing into a file

c) File Creation

d) Implementation of ls command

e) Implementation of grep command.

**Experiment no 2: CPU Scheduling**

1. To write a C program to implement the CPU scheduling algorithm for FIRST COME FIRST SERVE.
2. To write a C program to implement the CPU scheduling algorithm for Shortest Job First
3. To write a C program to implement the CPU scheduling algorithm for Round Robin
4. To write a C program to implement the CPU scheduling algorithm for Priority Scheduling.

**Experiment no 3: Inter-process Communication**

Write a program that creates a child process. Parent process writes data to pipe and child process reads the data from pipe and prints it on the screen

**Experiment no 4: Semaphore**

1. Write a program that demonstrates how two processes can share a variable using semaphore
2. To write a C program to implement the Producer & consumer Problem (Semaphore)

**Experiment no 5: Memory management-1**

To write a C program to implement memory management using paging technique

**Experiment no 6: Memory management-II**

To write a C program to implement memory management using segmentation technique

**Experiment no 7: FILE MANIPULATION**

1. Displays the file and Directory
2. Creating new Directory

**Experiment no 8: Fork Execution**

1. Simple fork execution
2. fork system call

**Experiment no 9 & 10: Deadlock avoidance**

1. To implement Banker's algorithm for a multiple resources
2. To implement dinning philosopher’s problem.

**Text Books:**

T1. SILBERSCHATZ, Galvin (2010), Operating System Concepts 8e , Wiley India.  
T2. William Stallings, “Operating systems”, Pearson Education, Fifth edition.  
T3. D.M. Dhamdhere, “Operating Systems”, 2nd Edition, Tata McGraw-Hill.

**Reference book:**

R1. Garry Nutt, “Operating Systems – A Modern perspective”, Third Edition,  
Pearson Education.

R2. Andrew S. Tanenbaum, “Modern Operating Systems”, Prentice Hall.  
R3. Bach, M.J., “Design of UNIX Operating System”, Prentice Hall.  
R4. Charles Crowley, “Operating systems – A Design Oriented Approach”, Tata  
Mc Grawhill, 1997.

R5. Michel Palmer “Guide o Operating Systems”, Vikas Thomson Learning  
Publishing, New Delhi.

**Continuous Evaluation-** There will be continuous evaluation for all practical subjects of SCS during the semester. The performance of a student in a Practical subject will be evaluated as per process given below:

* Components of evaluation
  1. Viva voce / Quiz (50%) + Performance & Records (50%).
  2. Lab performance and record evaluation shall be a continuous process throughout the semester.
  3. Minimum three Viva voce/ Quiz based on practical sessions shall be conducted during the semester.

**Relationship between the Course Outcomes (COs) and Program Outcomes (POs) and Program Specific Outcomes (PSOs)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Course Outcomes | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 | PSO 1 | PSO 2 | PSO 3 |
|
| CO1 | 2 | 1 | 1 | 2 | 1 |  |  |  |  |  |  |  | 3 |  |  |
| CO2 | 2 | 1 | 2 | 2 | 1 |  |  |  |  |  |  |  | 3 |  |  |
| CO3 | 1 | 2 | 2 | 2 | 1 |  |  |  |  |  |  |  | 3 |  |  |
| Average | 1.7 | 1.3 | 1.7 | 2 | 1 |  |  |  |  |  |  |  | 3 |  |  |

1. = weak 2 = moderate 3 = strong