|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| D:\UAAR\UIIT\courseOutlineCommittee\CourseContents_Final_V02\New folder\logo4.png | **PMAS Arid Agriculture University Rawalpindi**  **University Institute of Information Technology** | | | | C:\Users\Shahid\Downloads\IMG-20210824-WA0001.jpg |
| CS-583 Operating System | | | | | | |
| **Credit Hours:** | | **4(3-3)** | **Prerequisites:** | **None** | | |
| **Teacher:** | | **Dr. Ghulam Mustafa** |  |  | | |

|  |
| --- |
| **Course Description:** |
| Operating Systems Structures, System structures , Process Concepts Multitasking Multiprogramming, Multithreading, CPU Scheduling, Process management and scheduling policies, Inter process communication, Concurrent processing, Critical sections Mutual exclusion and synchronization, Semaphores, Monitors etc, Deadlocks, Resource allocation and Deadlock management, Memory Management policies (Contiguous i.e. partitioned / Segmented), Paged memory systems for Real memory, Address translation mechanisms, Virtual memory management, Page Removal Algorithms, Address translation mechanisms and cache management models / policies, File Systems, Access methods, Directory structure and protection, File structure, Directory implementation. Disk space allocation strategies, File allocation table (FAT), inodes, Secondary Storage, Disk I/O management policies, buffered I/O disk caching, Spooling devices. |
| **Course Objective:** |
| * List the most fundamental subsystems of an OS and the functions that each subsystem is responsible of. * List several ways in which operating systems differ from each other * Use techniques to protect OS from malicious activities * To know how programs, interact with each other in multi-tasking environment. * Know how the operating systems transport the application requests to the hardware. * Understand how operating systems managing resources such as processors, memory and I/O. |
| **Teaching Methodology:** |
| Lectures, Assignments, labs, Projects, Presentations, etc. Major component of the course should be covered using conventional lectures. Practical contact hours are compulsory. |
| **Courses Assessment:** |
| Exams, Assignments, Quizzes, Project, Presentations. Course will be assessed using a combination of written examinations and project(s). Practical evaluation, using rubrics, is encouraged and suggested to make up around 20% of the course. |
| **Reference Materials:** |
| * Silberschatz, Galvin, Gagne. Operating System Concepts. John Wiley &sons, inc * Operating systems design and implementation, Andrew s. Tanenbaum, Prentice-Hall. |

|  |  |  |
| --- | --- | --- |
| **Course Learning Outcomes (CLOs):** | | |
| At the end of the course the students will be able to: | **Domain** | **BT Level\*** |
| 1. **Understand** the characteristics of different structures of the Operating Systems and identify the core functions of the Operating Systems. | C | 2 |
| 1. **Analyze** and **evaluate** the algorithms of the core functions of the Operating Systems and explain the major performance issues with regard to the core functions. | C | 4,5 |
| 1. **Demonstrate** the knowledge in applying system software and tools available in modern operating systems | C | 3 |
| \* BT= Bloom’s Taxonomy, C=Cognitive domain, P=Psychomotor domain, A= Affective doma | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Week/Lecture #** | | **Theory** | **Practical** |
| Week 1 | Lect-I | Introduction (give definition of operating system and the function).  Computer system organization (Computer H.W.) includes:   * + 1. Device     2. Device controller.     3. Interrupt.     4. Device and CPU interaction     5. Bootstrap program |  |
| Lect-II&III | I/O structure (some related topics are in chapter 13)   * + 1. Polling, interrupt, and DMA     2. How to resolve interrupt through interrupt vector |  |
| Practical-I |  | Introduction to Linux. Exploring the Linux file system, including the basic concepts of files and directories and their organization in a hierarchical tree structure. Installation of environment on students computers. |
| Practical-II |  | A brief introduction to a few of the basic commands of Linux. |
| Practical-III |  | A simple “Hello World” program is compiled and executed. |
| Week 2 | Lect-I | * Computer System Architecture * Single Processor System * Multiple Processors System   + Parallel system   + Tightly coupled.   + advantages of using multiple processor system |  |
| Lect-II&III | * Single Processor System * Multiple Processors System   + Parallel system   + Tightly coupled.   + advantages of using multiple processor system   + Operating System Structure   + Operating System Operations     - Dual Modes operation     - Timer, Process management     - Storage management |  |
| Practical-I |  | Learn the basic CLI and GUI tools to monitor system resources |
| Practical-II |  | Identify system calls used by programs written in C language |
| Practical-III |  | Learn to write simple batch script using C Language. A simple “Hello World” program is compiled and executed |
| Week 3 | Lect-I | Chapter 3: Processes   * + Process concept, PCB, Process state   + Process scheduling (long, medium and short term schedulers) |  |
| Lect-II&III | * + Process operations   + Intercrosses communication |  |
| Practical-I |  | Write c language code for functions fork() system call in linux environment. Execute and analyze the programs. |
| Practical-II |  | Write c language code for functions getpid(),getppid() in linux environment. Execute and analyze the programs. |
| Practical-III |  | Write c language code for functions wait(),sleep() along with getpid(),getppid() and fork() in linux environment. Execute and analyze the programs. |
| Week 4 | Lect-I | Techniques of Interprocess communication   * Message passing * Shared memory * Client server   (N.B. Excludes section 3.5 Examples of IPC Systems). |  |
| Lect-II&III | Chapter 4: Thread Management  Overview  Multithreading models  Threading issues |  |
| Practical-I |  | Write a C program that receives a message from message queue and display them. |
| Practical-II |  | Write a C program that illustrates two processes communicating using Shared memory. |
| Practical-III |  | Introduction and overview of multithreaded programming. |
| Week 5 | Lect-I | Pthreads  Solaris 2 Threads  Windows 2000 threads  Linux Threads  Java Threads |  |
| Lect-II&III | chapter 5 : CPU scheduling   1. introduction and objectives 2. preemptive and non-preemptive scheduling   criteria |  |
| Practical-I |  | Discuss three types of threads, i.e. POSIX thread, JAVA thread and Windows thread. |
| Practical-II |  | Write a C program that illustrate the implementation of POSIX thread (pthread) in Linux environment. |
| Practical- III |  | Write a Java program that illustrate the implementation of JAVA thread in Linux environment. |
| Week 6 | Lect-I | * algorithms   + - FCFS   SJF + Prediction of next burst of SJF |  |
| Lect-II&III | * + - Priority Scheduling     - RR |  |
| Practical-I |  | Discuss various types of scheduling algorithms. |
| Practical- II |  | Write a program that illustrates communication between two process using named Round Robin scheduling algorithm. |
| Practical- III |  | Write a program that illustrates communication between two process using named FIFO scheduling algorithm. |
| Week 7 | Lect-I | * + - Multilevel Queues     - Multilevel feedback.   (Exclude thread scheduling)  Processor Scheduling (section 5.5) |  |
| Lect-II&III | Chapter 6: Process Synchronization   1. Introduction and background 2. Critical section (C.S.) problem 3. Condition for the solutions of C.S. |  |
| Practical- I |  | Write program to discuss File handling and management in Linux using System calls. |
| Practical- II |  | Write a program that illustrate communication between two process using unnamed pipes |
| Practical- III |  | Write a program that illustrates communication between two process using named pipes |
| Week 8 | Lect-I | Algorithms   * + - Peterson     - Hardware solutions     - Semaphores     - Monitors |  |
| Lect-II&III | Solutions of classical problems using above algorithms |  |
| Practical- I |  | Write C language program to implement semaphores using functions semget(),semctl(),semop() functions. |
| Practical- II |  | Write a C program to simulate the concept of Dining philosopher problem |
| Practical- III |  | Write a C program to simulate the concept of Bounded Buffer problem. |
| **Mid Term Exam** | | | |
| Week 9 | Lect-I | Chapter 7: Deadlock Management  Deadlocks, Resource allocation and Deadlock management |  |
| Lect-II&III | Resource Management and Deadlocks, including how deadlocks happen, |  |
| Practical- I |  | Simulate the following CPU scheduling algorithms. a)FCFS b) SJF c) Round Robin d) Priority. |
| Practical- II |  | Simulate Bankers Algorithm for Dead Lock Avoidance, |
| Practical- III |  | Write a shell script that receives any number of file names as arguments checks if every argument supplied is a file or directory and reports accordingly. Whenever the argument is a file it reports no of lines present in it. |
| Week 10 | Lect-I | modeling of deadlocks using Holt Resource Allocation Graphs, |  |
| Lect-II&III | Detection and Recovery, Deadlock Prevention Schemes, Deadlock |  |
| Practical-I |  | Discuss Dead locks in C language |
| Practical-II |  | Write program to simulate Deadlock avoidance by Dijkstra Bankers algorithm. |
| Practical-III |  | Write program to simulate Dead lock in OS by making use of two threads and two resources to create a circular wait solution. |
| Week 11 | Lect-I | Avoidance Schemes,  Banker's Algorithm,  Resource Trajectories |  |
| Lect-II&III | Chapter 8: Main Memory Management   1. Background 2. Basic Hardware for managing Memory 3. Address binding |  |
| Practical-I |  | Discuss memory allocation graph and resources. |
| Practical-II |  | Write a program to simulate malloc() in c language |
| Practical-III |  | Write a program to demonstrate use of free() function in c language using linux OS. |
| Week 12 | Lect-I | Swapping   * Contiguous allocation (based on fixed and variable partitions)   + Relocation and protection problems   + Fragmentation |  |
| Lect-II&III | Non-contiguous allocation   * Paging + hardware support) * Segmentation * Paging with Segmentation. |  |
| Practical-I |  | Write a program to demonstrate Clloc()-Contiguous allocation function. |
| Practical-II |  | Write a program to demonstrate Realloca()-Reallocation function. |
| Practical-II |  | Demonstration of Dynamic memory allocation using malloc() function. |
| Week 13 | Lect-I | Chapter 9 : Virtual Memory   1. Background 2. Demand paging 3. Basic concept 4. Performance in demand paging |  |
| Lect-II&III | Page replacement algorithms.   1. Allocation of frames. 2. Allocation algorithms 3. Thrashing and its causes   Detecting thrashing |  |
| Practical-I |  | Demonstrate shared memory concept using shmat() function. |
| Practical-II |  | Discuss shmdt,shmctl,ftok functions. |
| Practical-III |  | Demonstrate shared memory concept using shmget() function. |
| Week 14 | Lect-I | Chapter 11 : File System   1. File system structure 2. Implementation 3. Partition and mounting |  |
| Lect-II&III | Allocation methods   * Contiguous * Linked, Indexed |  |
| Week 15 | Lect-I | * Free space Management   + Bit vector, Linked list, Grouping   + Counting   Efficiency reliability and file sharing (Tanenbaum’ Book) | Implementation of files |
| Lect-II&III | Secondary Storage, Disk I/O management policies, buffered I/O disk caching, Spooling devices |  |
| Week 16 | Lect-I | Chapter 9: Virtual Memory   1. Background 2. Demand paging 3. Basic concept 4. Performance in demand paging   Project Demos | Project Demos |
| **Final Term Exam** | | | |