Course:CS603- Operating System

PROGRAMME: **COMPUTERSCIENCE&ENGINEERING**	DEGREE:**B. TECH**
COURSE: **Operating Systems**	SEMESTER: **6** CREDITS: **3**
COURSECODE: **CS603**	COURSE TYPE: **Theory**
COURSE AREA/DOMAIN: **Systems Software**	CONTACTHOURS: **3 (weekly)**
CORRESPONDINGLABCOURSE CODE (IFANY):**CS693**	LABCOURSE NAME:Operating System Lab

Course pre-requisites

CODE	COURSE NAME	DESCRIPTION	SEM
CS201	Basic Computation & Principles of Computer P	Programming basics	II
CS303	Computer Organisation	The components of a computer system	III

Course Objectives

1. To develop an understanding of the functions underlying operating system
2. To learn the techniques for interfacing with the operating systems

Course Outcomes

1. Students would be able to explain the functions of an Operating System
2. Students would be able to understand the interaction between application software, Operating Systems and the hardware.
3. Students would be able to pick and choose the best OS design technique for a given problem
4. Student s would be able to design at least a module od OS.

ProgrammeOutcomes addressed in this course

a. An ability to apply knowledge of mathematics, science, and engineering
b. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice (k)

Syllabus

UNIT	DETAILS	HOURS
I	**Introduction:** Introduction to OS. Operating system functions, evaluation of O.S., Different types of O.S.: batch, multi-programmed, time-sharing, real-time, distributed, parallel.	4
II	**System Structure:** Computer system operation, I/O structure, storage structure, storage hierarchy, different types of protections, operating system structure (simple, layered, virtual machine), O/S services, system calls.	3
III	***Processes :*** Concept of processes, process scheduling, operations on processes, co-operating processes, inter-process communication.	3
IV	***Threads :*** Overview, benefits of threads, user and kernel threads.	2
V	***CPU scheduling :*** Scheduling criteria, preemptive & non-preemptive scheduling, scheduling algorithms (FCFS, SJF, RR, priority), algorithm evaluation, multi-processor scheduling.	3
VI	***Process Synchronization :*** Background, critical section problem, critical region, synchronization hardware, classical problems of synchronization, semaphores. deadlock prevention, deadlock avoidance, deadlock detection, recovery from deadlock.	5
VII	***Deadlocks :*** System model, deadlock characterization, methods for handling deadlocks, deadlock prevention, deadlock avoidance, deadlock detection, recovery from deadlock.	4
VIII	***Memory Management :*** Background, logical vs. physical address space, swapping, contiguous memory allocation, paging, segmentation, segmentation with paging.	5
IX	***Virtual Memory :*** background, demand paging, performance, page replacement, page replacement algorithms (FCFS, LRU), allocation of frames, thrashing.	3
X	***File Systems :*** file concept, access methods, directory structure, file system structure, allocation methods (contiguous, linked, indexed), free-space management (bit vector, linked list, grouping), directory implementation (linear list, hash table), efficiency & performance.	4
XI	***I/O Management :*** I/O hardware, polling, interrupts, DMA, application I/O interface (block and character devices, network devices, clocks and timers, blocking and nonblocking I/O), kernel I/O subsystem (scheduling, buffering, caching, spooling and device reservation, error handling), performance.	4
XII	***Disk Management :*** Disk structure, disk scheduling (FCFS, SSTF, SCAN,C-SCAN) , disk reliability, disk formatting, boot block, bad blocks.	3

XIII	**Protection & Security :** Goals of protection, domain of protection, security problem, authentication, one time password, program threats, system threats, threat monitoring, encryption.	4

Gaps in the syllabus - to meet industry/profession requirements

S.NO.	DESCRIPTION	PROPOSED ACTIONS	PO MAPPING
1	**Batch processing History** Explanation of how batch processing using punch cards used to happen in the early days. Need of code walkthrough to minimize compilation time and remove more errors	Extra Class	a.
2	**How to Design a scheduler in different Operating system** Understand how, we can design a scheduler	Extra Class	b
3	**The role of hardware vis-à-vis software** Understand which tasks are executed by the hardware and which by the software and the cost / time implication.	Extra Class	b
4.	**How to Design a IPC module** Understand how, we can design a IPC Module	Extra Class	b
5.	**How to Design a Boot Strap Loader** Understand how, we can design a Boot Strap Module	Extra Class	b

Topics beyond syllabus/advanced topics

S.NO.	DESCRIPTION	HOURS
1	**Parallel and distributed processing** Understanding the needs of a distributed system, specifically addressing distributed deadlock and consistent global snapshot. .	1

Web Source References

S.NO.	URL
1	http://web.stanford.edu/~ouster/cgi-bin/cs140-spring14/index.php
2	http://www.ics.uci.edu/~ics143/lectures.html

Delivery/Instructional Methodologies

S.NO.	DESCRIPTION
1	Chalk and Talk
2	Study Material
3	Power Point Presentation

Assessment Methodologies

S.NO.	DESCRIPTION	TYPE
1	Student Assignment	Direct
2	Tests	Direct
3	University Examination	Direct
4	Student Feedback	Indirect

Course Plan

S. NO.	Day	Module	Topic
1.	Day 1	I	Introduction to OS.
2.	Day 2		Operating system functions
3.	Day 3		Evaluation of O.S.
4.	Day 4		4.Different types of O.S. Batch, Multiprogrammed
5.	Day 5		Different types of O.S.: Time-sharing, real-time, distributed, parallel.
6.	Day 6	II	Computer system operation, I/O structure, storage structure, storage hierarchy.
7.	Day 7		Different types of protections, operating system structure (simple, layered, virtual machine)
8.	Day 8		O/S services, system calls
9.	Day 9	III	Concept of processes, process scheduling
10.	Day 10		Operations on processes, Co-operating processes
11.	Day 11		Inter-process communication.
12.	Day 12	IV	Overview, Benefits of threads,
13.	Day 13		User and kernel threads.
14.	Day 14	V	Scheduling criteria, preemptive & non-preemptive scheduling,
15.	Day 15		Scheduling algorithms (FCFS, SJF, RR, priority),
16.	Day 16		Algorithm evaluation, multi-processor scheduling
17.	Day 17	VI	Background, critical section problem
18.	Day 18		Semaphores and its applications
19.	Day 19		Synchronization hardware
20.	Day 20		Classical problems of synchronization
21.	Day 21		Critical region, Monitor
22.	Day 22	VII	System model, Deadlock characterization
23.	Day 23		Methods for handling deadlocks
24.	Day 24		Deadlock prevention , Deadlock avoidance
25.	Day 25		Deadlock detection, recovery from deadlock
26.	Day 26	VIII	Background, logical vs. physical address space
27.	Day 27		Swapping, contiguous memory allocation
28.	Day 28		Paging
29.	Day 29		Segmentation
30.	Day 30		Segmentation with paging
31.	Day 31	IX	Background, demand paging, performance,
32.	Day 32		Page replacement, page replacement algorithms (FCFS, LRU),
33.	Day 33		Allocation of frames, thrashing.

S. NO.	*Day*	*Module*	*Topic*
34	Day34	**X**	File concept, Access methods, Directory structure
35	Day 35		File system structure
36	Day 36		Allocation methods (contiguous, linked, indexed), free-space management (bit vector, linked list, grouping)
37	Day 37		Directory implementation (linear list, hash table), efficiency & performance
38	Day 38	**XI**	I/O hardware, polling, interrupts, DMA
39	Day 39		Application I/O interface (block and character devices,
40	Day 40		Network devices, clocks and timers, blocking and nonblocking I/O),
41	Day 41		Kernel I/O subsystem (scheduling, buffering, caching, spooling and device reservation, error handling), performance.
42	Day 42	***XII***	Disk structure, Disk scheduling (FCFS, SSTF, SCAN,C-SCAN)
43	Day 43		Disk reliability
44	Day 44		Disk formatting, Boot block, Bad blocks
45	Day 45	***XIII***	Goals of protection, domain of protection, security problem
46	Day 46		Authentication, one time password
47	Day 47		Program threats, system threats
48	Day 48		Threat monitoring, encryption

Assignment Set I

S. NO.	Question	Assesses CO
1	How contiguous allocation problem is solved using paging	CO.1
2	Hardware need in paging	CO.2
3	Why page size is always powers of 2?	CO.1
4	Describe a mechanism by which one segment could belong to the address space of two different processes.	CO.2
5	In the IBM/370, memory protection is provided through the use of *keys*. A key is a 4-bit quantity. Each 2K block of memory has a key (the storage key) associated with it. The CPU also has a key (the protection key) associated with it. A store operation is allowed only if both keys are equal, or if either is zero.Which of the following memory-management schemes could be used successfully with this hardware? a. Bare machine b. Single-user system c. Multiprogramming with a fixed number of processes d. Multiprogramming with a variable number of processes e. Paging f. Segmentation	CO.3
6	Consider a logical address space of eight pages of 1024 words each, mapped onto a physical memory of 32 frames. a. How many bits are there in the logical address? b. How many bits are there in the physical address?	CO.2
7	What is the effect of allowing two entries in a page table to point to the same page frame in memory? Explain how this effect could be used to decrease the amount of time needed to copy a large amount of memory from one place to another. What effect would updating some byte on the one page have on the other page?	CO.2
8	Describe a mechanism by which one segment could belong to the address space of two different processes.	CO.2
9	Sharing segments among processes without requiring the same segment number is possible in a dynamically linked segmentation system. a. Define a system that allows static linking and sharing of segments without requiring that the segment numbers be the same. b. Describe a paging scheme that allows pages to be shared without requiring that the page numbers be the same.	CO.3

10	Name two differences between logical and physical addresses.	CO.3

Assignment Set II

S. NO.	Question	Assesses CO
1	Under what circumstances do page faults occur? Describe the actions taken by the operating system when a page fault occurs.	CO.1
2	Assume that you have a page-reference string for a process with *m* frames (initially all empty). The page-reference string has length *p*; *n* distinct page numbers occur in it. Answer these questions for any page replacement algorithms: a. What is a lower bound on the number of page faults? b. What is an upper bound on the number of page faults?	CO.1
3	Which of the following programming techniques and structures are "good" for a demand-paged environment ? Which are "not good"? Explain your answers. a. Stack b. Hashed symbol table c. Sequential search d. Binary search e. Pure code f. Vector operations g. Indirection	CO.2
4	Consider the following page-replacement algorithms. Rank these algorithms on a five-point scale from "bad" to "perfect" according to their page-fault rate. Separate those algorithms that suffer from Belady's anomaly from those that do not. a. LRU replacement b. FIFO replacement c. Optimal replacement d. Second-chance replacement	CO.2
5	When virtual memory is implemented in a computing system, there are certain costs associated with the technique and certain benefits. List the costs and the benefits. Is it possible for the costs to exceed the benefits? If it is, what measures can be taken to ensure that this does not happen?	CO.3
6	An operating system supports a paged virtual memory, using a central processor with a cycle time of 1 microsecond. It costs an additional 1 microsecond to access a page other than the current one. Pages have 1000 words, and the paging device is a drum that rotates at 3000 revolutions per minute and transfers 1 million words per second. The following	CO.3

statistical measurements were obtained from the system:
• 1 percent of all instructions executed accessed a page other than the current page.
• Of the instructions that accessed another page, 80 percent accessed a page already in memory.
• When a new page was required, the replaced page was modified 50 percent of the time.
Calculate the effective instruction time on this system, assuming that the system is running one process only and that the processor is idle during drum transfers.

7	Consider the two-dimensional array A:
	
	int A[][] = new int[100][100]; where A[0][0] is at location 200, in a paged memory system with pages of size 200. A small process is in page 0 (locations 0 to 199) for manipulatingthe matrix; thus, every instruction fetch will be from page 0.
	
	For three page frames, how many page faults are generated by the following array-initialization loops, using LRU replacement, and assuming page frame 1 has the process in it, and the other two are initially empty?
	a. for (int j = 0; j < 100; j++)
	for (int i = 0; i < 100; i++)
	A[i][j] = 0;
	
	b. for (int i = 0; i < 100; i++)
	for (int j = 0;
		CO.2

8	Consider the following page reference string: 1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6. How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames? Remember all frames are initially empty, so your first unique pages will all cost one fault each. • LRU replacement • FIFO replacement • Optimal replacement	CO.1
9	Suppose that you want to use a paging algorithm that requires a referencebit (such as second-chance replacement or working-set model), but the hardware does not provide one. Sketch how you could simulate areference bit even if one were not provided by the hardware, or explain why it is not possible to do so. If it is possible, calculate what the cost would be.	CO.1
10	You have devised a new page-replacement algorithm that you think may be optimal. In some contorted test cases, Belady's anomaly occurs. Is the new algorithm optimal? Explain your answer.	CO.2

Assignment Set III

S. NO.	Question	Assesses CO
1.	Design a CPU Scheduler which implements best scheduling algorithm.	CO.4,CO.3
2.	Design an IPC module for an Operating System.	CO. 4
3.	Design a Boot Strap Loader for an Operating System	CO.4