	RV College of Engineering® Department of Computer Science and Engineering CIE - I		
Course & Code	**Operating Systems** **(CS235AI)**		**Semester: III**
Date: 23/10/2024	**Duration:** 120 minutes	**Test Max. Marks:** 50 Marks **Quiz Max Marks:** 10 Marks	**Faculty In Charge:** **AN/JS/PH/NSK/VG/RJ/SK**
USN :	**Name :**		

NOTE: ***Answer all the questions***

Sl.no.	Questions	Marks	* BT	*CO
	QUIZ- 1			
1	To access the services of operating system the interface is provided by _____	1	1	1
2	Which of the type of OS reads and reacts in terms of actual time?	1	1	1
3	The output of the following C program is? int main(){ fork(); fork(); printf("code"); }	1	3	2
4	In a timeshare operating system, when the time slot assigned to a process is completed, the process switches from the current state to _____ state.	1	1	1
5	A thread shares the same _____ space with other threads in the same process.	1	1	2
6	When a process is in the "waiting" state, it is unable to continue execution until some _____	1	1	1
7	The _____ model allows threads to run in parallel, utilizing multiple cores in a multicore processor.	1	1	2
8	_____ swaps in and swaps out processes to improve the program-mix.	1	1	1
9	Illustrate the distinction between concurrency and parallelism with the help of a diagram.	2	2	2

TEST-1				
1.	What is Process? Illustrate and explain Process Control block and Process-State transition diagram using appropriate diagram.	10	2	1
2.	Describe in detail the lifecycle of a child process from creation till termination.	10	2	1
3.	Indicate, with specific justification, whether or not the following claims are true. i) Timesharing systems are identical to multiprogramming systems when the time slice exceeds CPU burst of every program. ii) Kernel mode is essential for isolated and protected execution of processes. iii) Swapping of processes in and out of the memory increases OS Overhead and decreases CPU efficiency. iv) If you were to write a program to do matrix multiplication for a uniprocessor environment. Out of the following options: write more efficient algorithm, use more threads, use more processes. Which one would be viable in terms of performance? Justify your choice. (Assume that you can only choose one).	10	4	2
4.a	Comment on the following aspects about Partition-based resource allocation scheme and Pool-based Resource allocation scheme. i) User Convenience ii) Efficiency of Use	05	2	2
4.b	What are two differences between user-level threads and kernel-level threads? Under what circumstances is one type better than the other?	05	2	1
5.a	Write a C program to illustrates UNIX fork() system call.	06	4	3
5.b	A task has 70% of its code that can be parallelized, and the remaining 30% cannot be parallelized. If you run this task on 5 processors, what is the theoretical maximum speedup according to Amdahl's Law?	04	3	4

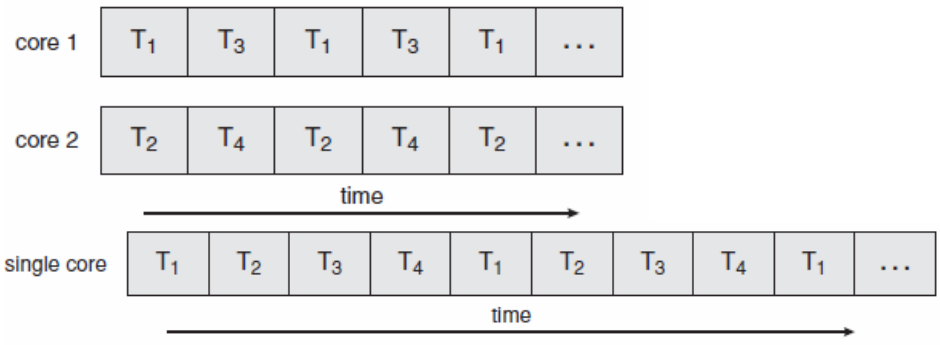
COURSE OUTCOMES:

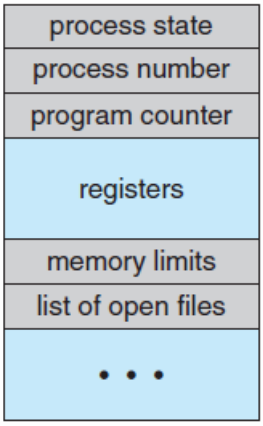
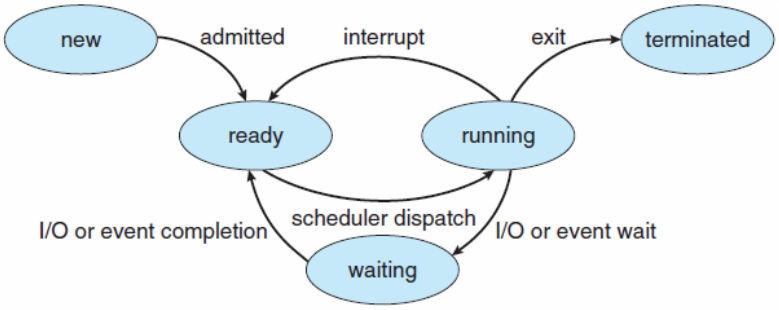
CO1:	Demonstrate the fundamental concepts of operating system like process management, file management, memory management and issues of synchronization.
CO2:	Analyze and interpret operating system concepts to acquire a detailed understanding of the course.
CO3:	Apply the operating systems concepts to address related new problems in computer science domain.
CO4:	Design or develop solutions using modern tools to solve applicable problems in operating systems domain.
CO5:	Extend the theoretical knowledge acquired through the course to demonstrate skills like investigation, effective communication, working in team/Individual, following ethical practices by implementing operating system concepts/applications and engage in lifelong learning.

	L1	L2	L3	L4	L5	L6	CO1	CO2	CO3	CO4	CO5
Marks	08	32	04	16	-	-	30	20	06	04	-

****

	RV College of Engineering® Department of Computer Science and Engineering CIE - I Scheme and Solution		
Course & Code	**Operating Systems** **(CS235AI)**		**Semester: III**
Date: 22/10/2024	**Duration:** 110 minutes	**Test Max. Marks**: 50 Marks **Quiz Max Marks:** 10 Marks	**Faculty In Charge:** **AN/JS/PH/NSK/VG/RJ/SK**
USN :	**Name :**		

Sl.no.	Questions	Marks	* BT	*CO
	Quiz-1 Solution			
1.	System calls	1	1	1
2.	Real-Time OS	1	1	1
3.	code code code	1	3	2
4.	Ready state	1	1	1
5.	Code section, data section and files	1	1	2
6.	Event occurs/ delay system call timer expires/ I/O becomes available.	1	1	1
7.	Many-to-many Multi-threading model.	1	1	2
8.	Medium-term scheduler.	1	1	1
9.		02	2	2
	Test-1 Solution			

1.	Process Definition -02 M  **3.3** Process control block (PCB).-Explanation + Diagram - 04 M State Transition Diagram + Explanation – 04 M As a process executes, it changes **state**. The state of a process is defined in part by the current activity of that process. A process may be in one of the following states: • **New**. The process is being created. • **Running**. Instructions are being executed. • **Waiting**. The process is waiting for some event to occur (such as an I/O completion or reception of a signal). • **Ready**. The process is waiting to be assigned to a processor. • **Terminated**. The process has finished execution.  **Figure 3.2** Diagram of process state.	10	2	1
2	When a process creates a new process, two possibilities for execution exist: 1. The parent continues to execute concurrently with its children. 2. The parent waits until some or all of its children have terminated. here are also two address-space possibilities for the new process: 1. The child process is a duplicate of the parent process (it has the same program and data as the parent). 2. The child process has a new program loaded into it.	10	2	1

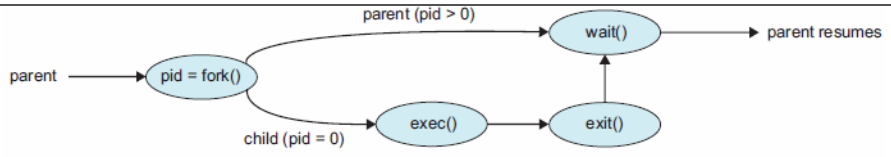


Figure 3.10 Process creation using the `fork()` system call.

A parent process may wait for the termination of a child process by using the `wait()` system call. The `wait()` system call is passed a parameter that allows the parent to obtain the exit status of the child. This system call also returns the process identifier of the terminated child so that the parent can tell which of its children has terminated:

```
pid_t pid;
int status;

pid = wait(&status);
```

Some systems do not allow a child to exist if its parent has terminated. In such systems, if a process terminates (either normally or abnormally), then all its children must also be terminated. This phenomenon, referred to as cascading termination, is normally initiated by the operating system.

When a process terminates, its resources are deallocated by the operating system. However, its entry in the process table must remain there until the parent calls `wait()`, because the process table contains the process's exit status. A process that has terminated, but whose parent has not yet called `wait()`, is known as a zombie process. All processes transition to this state when they terminate, but generally they exist as zombies only briefly. Once the parent calls `wait()`, the process identifier of the zombie process and its entry in the process table are released.

Now consider what would happen if a parent did not invoke `wait()` and instead terminated, thereby leaving its child processes as orphans. Linux and

3.	i) Timesharing systems are identical to multiprogramming systems when the time slice exceeds CPU burst of every program. **False.** While both timesharing and multiprogramming aim to maximize CPU utilization by allowing multiple processes to run concurrently, they are not identical. Timesharing systems are designed for interactive use, allowing multiple users to interact with the system simultaneously, while multiprogramming focuses on maximizing CPU usage by keeping several processes in memory at once. Even if the time slice exceeds the CPU burst of every program, the fundamental goal of interactivity in timesharing still distinguishes it from multiprogramming. ii) Kernel mode is essential for isolated and protected execution of processes. **True.** Kernel mode is necessary for managing hardware resources and ensuring that processes are isolated from each other. When a process runs in kernel mode, it has access to all hardware and can execute privileged instructions, which allows the operating system to enforce protection and isolation among processes. Without kernel mode, user processes could interfere with each other and the system itself, leading to security and stability issues. iii) **True.** Swapping involves moving processes between memory and disk, which incurs significant overhead due to the time taken to read from and write to disk. This can lead to increased operating system overhead and reduce CPU efficiency, as the CPU may spend time waiting for processes to be swapped in rather than	10 2.5 M *4	4	2

	executing them. Frequent swapping can also lead to thrashing, where the system spends more time swapping than executing processes. iv) If you were to write a program to do matrix multiplication for a uniprocessor environment. Out of the following options: write more efficient algorithm, use more threads, use more processes. Which one would be viable in terms of performance? Justify your choice. **Write more efficient algorithm.** In a uniprocessor environment, performance gains from parallelism (using more threads or processes) are limited since only one thread/process can execute at a time. Thus, while using multiple threads or processes could introduce some concurrency, the bottleneck would still be the single CPU. On the other hand, improving the algorithm (e.g., optimizing the matrix multiplication technique) directly reduces the number of operations needed, leading to a better overall execution time regardless of the concurrency limitations.			
4.a	Let's break down the comparison of Partition-based and Pool-based resource allocation schemes regarding user convenience and efficiency of use: i) User Convenience **Partition-based Resource Allocation:** • **Pros:** Users typically have dedicated portions of resources (e.g., fixed-size partitions in memory), making it easier to predict resource availability. This setup allows users to have consistent performance for their tasks since they know exactly what resources are allocated to them. • **Cons:** However, if a user's needs exceed their allocated partition, they may experience resource starvation or underutilization of available resources, leading to frustration. Additionally, fixed partitions can lead to fragmentation, where some portions remain unused. **Pool-based Resource Allocation:** • **Pros:** This scheme allows for more flexible resource usage, where resources can be dynamically allocated based on demand. Users can draw from a pool of resources as needed, which can be more convenient in environments with fluctuating workloads. • **Cons:** The unpredictability of resource availability can be a drawback. Users may find it difficult to gauge how many resources will be available at any given time, potentially leading to uncertainty and performance variability. ii) Efficiency of Use **Partition-based Resource Allocation:** • **Pros:** Can lead to predictable performance since each process is allocated a fixed amount of resources. This setup can prevent resource wastage if managed correctly. • **Cons:** However, it often results in poor overall efficiency due to fragmentation. For instance, if partitions are not sized properly for the workload, some resources may sit idle. Additionally, the fixed	05	2	2

	nature can lead to situations where not all allocated resources are fully utilized, creating inefficiencies. **Pool-based Resource Allocation:** • **Pros:** Generally allows for better resource utilization since resources can be allocated and released as needed. This dynamic allocation can adapt to varying workloads, leading to more efficient use of available resources. • **Cons:** However, the overhead of managing the resource pool can reduce efficiency. Allocating and deallocating resources on-the-fly can introduce latency, and if demand exceeds supply, it can lead to contention and reduced performance. To summarize: • **User Convenience:** Partition-based schemes offer more predictability, while pool-based schemes provide flexibility but can introduce uncertainty. • **Efficiency of Use:** Pool-based schemes generally offer better resource utilization but may incur management overhead, whereas partition-based schemes can suffer from fragmentation and underutilization.			
4.b	a. User-level threads are unknown by the kernel, whereas the kernel is aware of kernel threads. b. On systems using either M:1 or M:N mapping, user threads are scheduled by the thread library and the kernel schedules kernel threads. c. Kernel threads need not be associated with a process whereas every user thread belongs to a process. Kernel threads are generally more expensive to maintain than user threads as they must be represented with a kernel data structure.	05	2	1
5.a		06	4	3

	`` #include <sys/types.h> #include <stdio.h> #include <unistd.h> int main() { pid_t pid; /* fork a child process */ pid = fork(); if (pid < 0) { /* error occurred */ fprintf(stderr, "Fork Failed"); return 1; } else if (pid == 0) { /* child process */ execlp("/bin/ls","ls",NULL); } else { /* parent process */ /* parent will wait for the child to complete */ wait(NULL); printf("Child Complete"); } return 0; } ```			
5.b	P=0.7 (the portion of the task that can be parallelized), 1−P=0.31 - P = 0.31−P=0.3 (the portion that cannot be parallelized), n=5 (number of processors). $$S(n) = \frac{1}{(1-P) + \frac{P}{n}}$$ $$S(5) = \frac{1}{(1-0.7) + \frac{0.7}{5}} = \frac{1}{0.3 + \frac{0.7}{5}} = \frac{1}{0.3 + 0.14} = \frac{1}{0.44} \approx 2.27$$ The theoretical maximum speedup when using 5 processors is approximately 2.27 times faster than using just 1 processor.	04	3	4

COURSE OUTCOMES:

CO1 :	Demonstrate the fundamental concepts of operating system like process management, file management, memory management and issues of synchronization.
CO2 :	Analyze and interpret operating system concepts to acquire a detailed understanding of the course.
CO3 :	Apply the operating systems concepts to address related new problems in computer science domain.
CO4 :	Design or develop solutions using modern tools to solve applicable problems in operating systems domain.
CO5 :	Extend the theoretical knowledge acquired through the course to demonstrate skills like investigation, effective communication, working in team/Individual, following ethical practices by implementing operating system concepts/applications and engage in lifelong learning.

	L1	L2	L3	L4	L5	L6	CO1	CO2	CO3	CO4	CO5

Marks	08	32	04	16	-	-	30	20	06	04	-

	RV College of Engineering® Department of Computer Science and Engineering CIE - II		
Course & Code	**Operating Systems(CS235AI)**		**Semester: III**
Date: December 2024	**Duration:** 120 minutes	**Test Max. Marks:** 50 Marks **Quiz Max Marks:** 10 Marks	**Faculty In Charge:** AN/JS/PH/NSK/VG/RJ/SK

NOTE: ***Answer all the questions***

Sl.no.	QUIZ- II	Marks	* BT	*CO
1.	Which scheduling algorithm is most suitable for a time-sharing system?	1	3	2
2.	In a preemptive scheduling algorithm, when can a currently running process be interrupted?.	1	2	2
3.	What is a race condition? Give an example.	2	2	3
4.	List two types of semaphores with an example for each?	2	3	3
5.	Why is mutual exclusion important in solving the critical section problem?	2	2	2
6.	Define CPU utilization and its goal.	2	2	2
	TEST-II			
1a	Consider the methods used by processes P1 and P2 for accessing their critical sections whenever needed, as given below. **Method used by P_1**: while (S_1 == S_2); Critical Section S_1 = S_2 **Method used by P_2**: while (S_1 != S_2); Critical Section S_2 = !S_1 The initial values of shared Boolean variables S1 and S2 are as follows: Case 1: S1 = 0 and S2 = 0 Case 2: S1 = 0 and S2 = 1 Is Mutual exclusion and progress guaranteed in both cases? Justify your answer with relevant explanations.	7	4	4
1b	Given three tasks with the following characteristics: • Task A: Period = 4, Execution Time = 1, Deadline = 4 • Task B: Period = 6, Execution Time = 2, Deadline = 6 • Task C: Period = 8, Execution Time = 2, Deadline = 8 Compute the Processor utilization and show the scheduling using Earliest-Deadline First Scheduling algorithm for first 12 units.	3	4	4
2a	Consider the set of 5 processes whose arrival time and burst time are given below If the CPU scheduling policy is SJF non-preemptive, Draw Gantt chart and calculate the average waiting time and average turn around time.	6	4	3

	Process Id / **Arrival time** / **Burst time** P1 / 3 / 1 P2 / 1 / 4 P3 / 4 / 2 P4 / 0 / 6 P5 / 2 / 3			
2b	Compare Rate monotonic scheduling and Earliest deadline first scheduling.	4	3	2
3a	Discuss the Scheduling Criteria for scheduling algorithms.	5	2	3
3b	Which algorithm encounters Starvation problem. What is the solution for the same . explain with an example.	5	2	4
4a	Given the following set of process: P1: Arrival Time 1, Burst Time 1 P2: Arrival Time 1, Burst Time 3 P3: Arrival Time 0, Burst Time 40 Applying FCFS algorithm draw the Gant chart and computing Average Waiting Time and Average Turn around time. If the arrival time of P3 is changed to 1 and P1,P2 to 0 what would be the variation in the Average Waiting time and Average Turn around time . Does convoy effect appears in the problem or not?	06	4	3
4b	Define Critical section problem with an example. Explain any two conditions to be satisfied for the solution of critical section problem	04	2	3
5a	Explain the Peterson's solution with algorithm for two process synchronization problem.	04	2	5
5b	Explain with an algorithm how test and set instruction solve all the requirements of critical section problem's solution.	06	2	4

COURSE OUTCOMES:

CO1:	Demonstrate the fundamental concepts of operating system like process management, file management, memory management and issues of synchronization.
CO2:	Analyze and interpret operating system concepts to acquire a detailed understanding of the course.
CO3:	Apply the operating systems concepts to address related new problems in computer science domain.
CO4:	Design or develop solutions using modern tools to solve applicable problems in operating systems domain.
CO5:	Extend the theoretical knowledge acquired through the course to demonstrate skills like investigation, effective communication, working in team/Individual, following ethical practices by implementing operating system concepts/applications and engage in lifelong learning.

	L1	**L2**	**L3**	**L4**	**L5**	**L6**	**CO1**	**CO2**	**CO3**	**CO4**	**CO5**
Marks	--	31	7	22	--	--	--	10	25	21	4

****

	RV College of Engineering® Department of Computer Science and Engineering CIE - III		
Course & Code	Operating Systems (CS235AI)		Semester: III
Date: 08/01/2025	Duration: 120 minutes	Test Max. Marks: 50 Marks Quiz Max Marks: 10 Marks	Faculty In Charge: AN/JS/PH/NSK/VG/RJ/SK
USN :	Name :		

NOTE: ***Answer all the questions***

Sl.no.	Questions	Marks	*BT	*CO
	QUIZ- 1			
1	Write the significance of TLB (Translation Look Aside Buffer)	2	1	1
2	In enhanced second chance algorithm, the ordered pair of a page which is the worst choice for page replacement?	1	1	1
3	Define Thrashing. Identify the causes for process thrashing	2	1	2
4	If the total number of available frames is 50, and there are 2 processes one of 10 pages and the other of 5 pages then how much of memory would be proportionally allocated to each of these processes?	2	2	3
5	Compute the Effective Access Time if the hit ratio is 99%, takes 20ns for TLB search and 100ns for memory access.	1	2	2
6	Define Working Set Window.	2	1	1
	TEST-1			
1.a	Consider the following page reference string: 7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1. How many page faults would occur for the following page replacement algorithms assuming 3 frames? All frames are initially empty. i. LRU ii. FCFS iii. Optimal Which algorithm is the most efficient?	07	2	4
1.b	Analyze Linked list allocation using a file allocation table in main memory.	03	2	2

2.a	Discuss the working of Clock replacement Algorithm	05	2	1
2.b	Write a C program to copy a file	05	3	4
3.	Analyse the sequence of steps taken to service a page fault with necessary diagram	10	3	1
4.a	Analyze the advantages and drawbacks of contiguous memory allocation. Given the memory partitions of 100K, 500K, 200K, 300K, AND 600K(in order) how would each of the first fit, best fit and worst fit algorithms place processes of 212K, 417K, 112K and 426K (in order)? Which algorithm makes the most efficient use of memory?	08	3	3
4.b	Differentiate between static and dynamic linking	02	2	1
5.a	Discuss the following page table structures: i. Hashed Page Tables ii. Inverted Page Tables	06	3	1
5.b	Analyze equal and proportional allocation of frames among processes with suitable examples	04	2	2

COURSE OUTCOMES:

CO1:	Demonstrate the fundamental concepts of operating system like process management, file management, memory management and issues of synchronization.
CO2:	Analyze and interpret operating system concepts to acquire a detailed understanding of the course.
CO3:	Apply the operating systems concepts to address related new problems in computer science domain.
CO4:	Design or develop solutions using modern tools to solve applicable problems in operating systems domain.
CO5:	Extend the theoretical knowledge acquired through the course to demonstrate skills like investigation, effective communication, working in team/Individual, following ethical practices by implementing operating system concepts/applications and engage in lifelong learning.

	L1	L2	L3	L4	L5	L6	CO1	CO2	CO3	CO4	CO5
Marks	07	24	29	-	-	-	28	10	10	12	-

****

1.8	At a particular time of computation the value of a counting semaphore is 7. Then $20P$ operations and xV operations were completed on this semaphore. If the new value of semaphore is $5, x$ will be	02	3	2
1.9	Consider a memory management system that uses a page size of $2\,KB 2\,KB$. Assume that both the physical and virtual addresses start from 0. Assume that the pages 0,1,20,1,2, and 33 are stored in the page frames 1,3,21,3,2, and 00, respectively. The physical address (in decimal format) corresponding to the virtual address 25002500(in decimal format) is _____.	01	4	3
1.10	Consider a demand paging system with four page frame(initially empty) and LRU page replacement policy. For the following page reference string 7,2,7,3,2,5,3,4,6,7,7,1,5,6,1 The page fault rate is _____.	01	3	2
1.11	The _____ operation within a file need not involve any actual I/O.	01	1	1
1.12	The MFD is indexed by user name or account number, and each entry points to the _____ for that user.	01	2	1
1.13	What are the four main object types defined by the Linux VFS.	02	1	3

PART-B

2	a	Demonstrate how system calls manage file handling operations in modern operating systems. Provide examples	08	4	1
	b	A uniprocessor computer system has three processes, which alternate $20ms$ CPU bursts with $80ms$ I/O bursts. All the processes were created at nearly the same time. The I/O of all the processes can proceed in parallel. What will be the CPU utilization (over a long period of time) using $FCFS$ and Round Robin (time quantum 10ms) for this system?	08	3	3
3	a	Analyze the benefits of multithreading in modern operating systems. Discuss how multithreading improves responsiveness, resource sharing, and performance in a multicore environment with the help of examples.	08	3	2
	b	With the help of diagrams, explain the different multithreading models. Also explain the types of parallelism.	08	2	4
		OR			
4	a	Develop a program to perform matrix multiplication using pthreads, where each thread computes one row of the resulting matrix.	08	4	2
	b	For the following set of processes, compute the average waiting time and turnaround time using First-Come, First-Served ($FCFS$) and Shortest Job First (SJF) scheduling algorithms and analyze the results.	08	3	3

$Process$	$Arrival\ Time(ms)$	$Burst\ Time(ms)$
$P1$	0	10
$P2$	2	5
$P3$	4	8
$P4$	6	2

5	a	Explain Peterson's Solution and the structure of Pi in this context. What are the disadvantages?	08	2	4
	b	Write a program to demonstrate the use of a mutex lock to solve a critical section problem. Explain the flow of execution and how mutual exclusion is maintained.	08	3	4

USN | 1 | R | V | 2 | 3 | C | E | 0 | 0 | 3 |

RV COLLEGE OF ENGINEERING®

(An Autonomous Institution Affiliated to VTU)

III Semester B. E. Regular / Supplementary Examinations Feb/Mar-2025

OPERATING SYSTEMS

Common to CS / IS / CD / CY

Time: 03 Hours *Maximum Marks: 100*

Instructions to candidates:

1. Answer all questions from Part A. Part A questions should be answered in first three pages of the answer book only.
2. Answer FIVE full questions from Part B. In Part B question number 2 is compulsory. Answer any one full question from 3 and 4, 5 and 6, 7 and 8, 9 and 10.

PART-A

			M	BT	CO
1	1.1	Consider the following code spinet using the fork() and wait() system calls. Assume that the code compiles and runs correctly, and that the system calls run successfully without any errors. $int\ x = 3;$ $while(x > 0)$ { $fork();$ $printf("hello");$ $wait(NULL);$ $x--;$ } Calculate the total number of times the printf statement is executed.	02	4	1
	1.2	Assume that a process executes the following code: $for(i = 1; i <= n;\ i++)$ $fork();$ How many new processes will be created?	01	3	2
	1.3	A _____ is a collection of physically separate, possibly heterogeneous computer systems that are networked to provide users with access to the various resources that the system maintains.	01	1	3
	1.4	Suggest any four challenges in programming for multicore systems.	02	2	2
	1.5	Can one have concurrent execution of threads/processes without having parallelism? Support with reason.	02	3	1
	1.6	The hardware implementation which provides mutual exclusion is _____.	01	1	4
	1.7	Each of a set of n processes executes the following code using two semaphores a and b initialized to 1 and 0, respectively. Assume that count is a shared variable initialized to 0 and not used in Code Section P. What does the code achieve? CODE SECTION P wait(a); count=count+1; if (count==n) signal(b); signal (a);wait(b); signal(b); CODE SECTION Q	02	3	5

		OR			
6	a	Develop a solution for the Readers-Writers problem using semaphores. Explain how the program ensures fairness and synchronization.	08	2	1
	b	Identify the key differences between mutex locks and semaphores in terms of functionality, usage, and overhead. Provide examples to illustrate their application.	08	1	1
7	a	Explain the process of address binding. With the help of a diagram, bring out Multistep processing of a user program.	08	2	2
	b	Consider a computer system with a 32-bit logical address and $4-KB$ page size. The system supports up $512\,MB$ of physical memory. How many entries are there in each of the following? i) A conventional , single-level page table ii) An inverted page table	04	3	5
	C	Give one method that avoids external fragmentation and the need for compaction. Use a diagram to briefly explain the hardware used and the technique followed.	04	2	4
		OR			
8	a	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 three, four, five, six frames: i) *LRU* replacement ii) *FIFO* replacement	08	3	4
	b	Suggest two strategies to prevent thrashing/limit the effects of thrashing.	04	2	2
	c	Consider a logical address space of 64 pages of 1024 words each, mapped onto a physical memory of 32 frames. i) How many bits are there in the logical address? ii) How many bits are there in the physical address?	04	4	3
9	a	Explain the layered file system in detail.	08	1	1
	b	Write a program that demonstrates the use of any three basic file system calls.	08	3	2
		OR			
10	a	Explain the *VFS* architecture in Linux.	08	1	1
	b	Elaborate the concept of file locking in detail.	08	2	2