Gyanmanjari Innovative University

Syllabus
Gyanmanjari Institute of Technology
Semester-4 (B. Tech)

Subject: Operating System: Architecture, Security, and Process Management- BET1CE14311

Type of course: Professional Core

Prerequisite: Linear and non-linear data structures and knowledge of any structured programming language.

Rationale:

This course provides a structured and in-depth understanding of Operating Systems through both theoretical concepts and practical implementation. It covers OS architecture, system components, UNIX fundamentals, shell scripting, process and thread management, CPU scheduling, memory management, synchronization, deadlocks, and inter-process communication. Students also work with system utilities, filters, communication commands, disk scheduling, security principles, and file management techniques. Practical exercises and simulations using tools reinforce learning and enable students to apply core operating system mechanisms effectively.

Teaching and Examination Scheme:

Teaching Scheme			Credits	Examination Marks		Total Marks
CI	T	P	C	SEE	CCE	
4	0	2	5	100	50	150

Legends: CI-Classroom Instructions; T-Tutorial; P -Practical; C –Credit; ESE -End Semester Examination; MSE-Mid Semester Examination; V–Viva; CA-Continuous Assessment; ALA-Active Learning Activities.

GYANMANJARI INNOVATIVE UNIVERSITY BHAVNAGAR GMIU

Course Content:

Sr. No.	Course Content	Hrs.	% Weightage
1	**Introduction to Operating System** ***Theory Topics:*** Introduction to Operating System, the need for Operating System, Components of Computer System, Computer System Architecture, Functions of Operating System: User View and System View, Types of Operating System: Batch Processing, Multiprogramming, Multitasking, Time Sharing, Parallel, Distributed, Real-Time, Embedded, Network, Uniprocessor and Multiprocessor Operating System. Introduction to UNIX Operating System, Features and Architecture of UNIX, Kernel and Shell Concepts, System Calls, UNIX File System and Directory Structure, Introduction to Online UNIX Simulators, Basic UNIX Commands, Introduction to Shell Programming, Types of shell and comparison of Shells (Sh, Csh, Ksh, Tcsh, Bash, Zsh, Pdksh), Creating Shell Scripts using various commands of Linux except Filters, Interactive shell script using read and echo, I/O redirection, Shell scripts based on file attribute testing, Test command, Decision Statements (if then fi, if then else fi, if then elif else fi, Case esac). ***Practical:*** 1. Install UNIX OS or use an online UNIX simulator and explore the interface like WSL (Windows Subsystem for Linux), https://www.tutorialspoint.com/compilers/online-linux-terminal.htm, https://bellard.org/jslinux/, TerminalTemple.com and OneCompiler.com 2. Write a command to display the current shell. 3. Write a script that prints the default shell and checks if Bash, Zsh, Ksh, and Sh are installed. 4. Write a script that displays the current date, current directory, and logged-in users using basic Unix commands. 5. Write a script that stores the output of ls, pwd, and whoami commands into variables and prints them. 6. Write a shell script that creates a new directory using mkdir, changes into it using cd, and displays the current working directory using pwd. 7. Write a shell script that uses the cat command to create a file and display its contents. 8. Write a shell script that uses the ls command to list all files and directories in the current path. 9. Write a shell script that uses echo to print a user-defined message and cat to display the contents of an existing file. 10. Write an interactive script that reads the user's name and age and prints a greeting message.	18	20 %

GYANMANJARI INNOVATIVE UNIVERSITY • BHAVNAGAR • GMIU

11. Write a script that reads two numbers from the user and prints their sum.
12. Write a script that redirects the output of date, who, and pwd commands to system_info.log.
13. Write a script that backs up a file and redirects success or failure status to backup.log.
14. Write a script that checks if a given filename exists and whether it is a regular file or directory.
15. Write a script that checks whether a given file is empty or not using the -s test operator.
16. Write a script that checks if a file has read, write, and execute permissions.
17. Write a script that uses the test command to compare two numbers and prints the larger one.
18. Write a script using if-else to check whether a number is even or odd.
19. Write a script using if-elif-else to determine if a number is positive, negative, or zero.
20. Write a menu-driven shell script using case to list files, show the current directory, display the date and calendar, and exit.
21. Write a shell script to display current user, date, time, and working directory.
22. Write a mini shell program that executes commands with arguments (e.g., ls -l, cat filename).
23. Solve the unsolved questions on decision statements given in the reference book.

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**SmartShell: A Unified File Handling Utility:** Students will create a shell script demonstrating core UNIX concepts, file tests, conditions, I/O redirection, and basic command execution.	20	
2	**Active Learning Assignment: OS Explorer Lab: Discover, Analyze & Automate in UNIX:** Students will explore OS concepts, execute UNIX commands, and create a file-checking shell script in an online UNIX simulator, compiling all observations and outputs into a PDF for submission on the GMIU Portal.		10
	Total	20	10

GYANMANJARI INNOVATIVE UNIVERSITY BHAVNAGAR GMIU

	Examination Style: **SmartShell: A Unified File Handling Utility (20 Marks)** Students will receive a shell script question from faculty to write a comprehensive shell script that demonstrates the fundamental concepts of the UNIX operating system and shell programming. The script should accept user input, perform file attribute testing using the test command, apply conditional statements such as if-then-else and case, utilize I/O redirection, and execute appropriate UNIX commands on files or directories. Your script must clearly reflect the use of shell types, system calls, command execution, and interactive features like read and echo. **OS Explorer Lab: Discover, Analyze & Automate in UNIX (10 Marks)** Students shall perform an interactive activity using any online UNIX simulator (TutorialPoint Linux Terminal, JDoodle, OnlineGDB Terminal, or JSLinux). The task includes writing a short explanation on the need for an Operating System, drawing a simple OS architecture diagram, and exploring the UNIX environment by executing basic commands such as pwd, ls -l, mkdir, cd, touch, rm, cat, echo, and whoami, while noting their relation to system calls like open, read, write, fork, and exec. Students must also create and execute a shell script named filecheck.sh to check file existence and permissions using conditional statements. All observations, screenshots, script code, and outputs must be compiled into a clear PDF report and submitted on the GMIU Portal.		
2	**Process Management and CPU Scheduling** ***Theory Topics:*** Process Concept, Process States and State Transitions, Relationship between Processes, Process Life cycle, Process Control Block (PCB), Implementation of Processes, Context Switching, Process Switching, Schedulers: Long-term scheduler, Short-term scheduler, Medium-term scheduler, CPU Scheduling: Preemptive & Non-preemptive, Scheduling Algorithms: First come first serve (FCFS), SJF, Round Robin, Priority, Real Time, System Calls for Process Control – ps, fork, join, exec, wait, Using OnlineGDB for writing, executing, and debugging process-related programs (fork, exec, wait, thread creation, and CPU scheduling simulations), Advanced Shell Programming, Operators, Arithmetic Operators, Relational Operators, Logical Operators, Arithmetic in Shell script using expr, Looping statements, for loop, while loop, until loop, Break, continue command, Threads – Concept, Advantages, and Types of Thread, Multi-Tasking vs. Multi-Threading, Thread Control Block. ***Practical:***	18	20 %

GYANMANJARI INNOVATIVE UNIVERSITY GMIU BHAVNAGAR

	24. Write a program using fork() to print parent and child process IDs. 25. Write a program to demonstrate process creation using nested fork(). 26. Write a program to display which process executes first (parent or child). 27. Write a program where the parent process uses exec() to run the ls command. 28. Write a program that uses wait() to ensure the parent waits for the child to finish. 29. Write a program to demonstrate a zombie process and an orphan process. 30. Write a shell script to display the current processes using ps command and filter by a specific user. 31. Write a shell script to check if a file exists and whether it is readable, writable, and executable. 32. Write a shell script to perform addition, subtraction, multiplication, and division of two numbers. 33. Write a shell script to calculate the area of a circle. 34. Write a shell script to check whether a number is even or odd. 35. Write a shell script to calculate the square and cube of a number. 36. Write a shell script to check whether a number is greater than, less than, or equal to another number. 37. Write a shell script to check whether a user is eligible to vote. 38. Write a shell script to compare two strings and display whether they are equal. 39. Write a shell script to check if a number lies between 1 and 100. 40. Write a shell script to check if a student has passed using logical conditions. 41. Write a shell script to check whether a character is a vowel or consonant. 42. Write a shell script to calculate salary with DA, HRA, and PF. 43. Write a program to implement both the FCFS CPU scheduling algorithm and SJF CPU scheduling algorithm, analyze its CPU utilization and turnaround time. 44. Write a shell script to find the sum of digits of a number. 45. Write a shell script to convert Celsius temperature to Fahrenheit. 46. Write a program to implement Round Robin scheduling with time quantum. 47. Write a shell script to print numbers from 1 to 10 using a for loop. 48. Write a shell script to print the multiplication table of a number. 49. Write a shell script to print all .txt files in the current directory. 50. Write a shell script to print factorial of a number using a for loop. 51. Write a program to implement Priority Scheduling and calculate its waiting time and turnaround time. 52. Solve the unsolved questions on FCFS, SJF, Round-robin, and Priority CPU scheduling algorithms given in the reference book and analyze their CPU utilization and turnaround time.		

GYANMANJARI INNOVATIVE UNIVERSITY
GMIU
BHAVNAGAR

53. Write a script to list all running processes along with their PID (Process ID) and PPID (Parent Process ID) and terminate a child process using kill().
54. Write a shell script to check if a given process is running.
55. Write a shell script to print numbers from 1 to N using a while loop.
56. Write a shell script to reverse the digits of a number using a while loop.
57. Write a shell script to print numbers from 1 to 10 using an until loop.
58. Write a shell script that keeps taking input until the user enters 0.
59. Write a shell script to create and run threads, each printing a message with thread ID.
60. Write a program to create two threads: one prints Fibonacci series and one prints multiplication table.
61. Write a program using threads to calculate factorial of a number.
62. Write a program to demonstrate a race condition between two threads.

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Shell Scripting Proficiency Check:** Students will write an advanced shell script using operators, expr, loops with break/continue, user input, and provide sample output.	20	
2	**Active Learning Assignment: Real-Time CPU Scheduling Visualization:** Students will use simulators to test CPU scheduling algorithms, compare their performance, and upload an individual report with screenshots and findings to the GMIU web portal.		10
	Total	20	10

Examination Style:

Shell Scripting Proficiency Check (20 Marks)
Students will write a shell script that demonstrates advanced shell programming concepts by using arithmetic, relational, and logical operators, performing arithmetic operations with expr, and implementing looping constructs such as for, while, and until. Your script must include the proper use of break and continue statements, process user input, and display meaningful output. Ensure each part of the script clearly shows the use of operators, loop execution, and control flow. Submit the script along with sample outputs for evaluation.

	Real-Time CPU Scheduling Visualization (10 Marks) Students will use simulators (CSS, PSS, EdSim51, Queuing Theory, and GNS3) to implement and observe CPU scheduling algorithms (FCFS, SJF, Priority, RR). They will input process attributes (arrival time, burst time, priority) and analyze each algorithm's performance, comparing waiting time, turnaround time, and CPU utilization. The final report, including observations, screenshots, charts, and a summary of findings, will be uploaded individually to the GMIU web portal.		
3	**Memory Management** ***Theory Topics:*** Memory Management Concept, Requirements and Goals, Static and Dynamic Memory Allocation, Logical and Physical addresses, Memory Partitioning – Fixed and Variable, Swapping, Contiguous Memory Allocation – First Fit, Best Fit, Worst Fit, Compaction, Non-continuous Memory Allocation: Paging, and Fragmentation, Segmentation, Virtual Memory and Demand Paging, Page Replacement Algorithm (FIFO, LRU, Optimal), Thrashing. Filters: Simple Filters (Cut, Paste, Join, Fold, Sort, Tr, Unique, Head, Tail), Join two filters using Pipe, Advanced Filters (Grep, Regular Basic Expression, Sed, Awk), Formatting output-printf, the comparison operator, Variables, Built-in variables. ***Practical:*** 63. Write a program to demonstrate static memory allocation. 64. Write a program to demonstrate dynamic memory allocation using malloc(), calloc(), realloc(), and free(). 65. Write a shell command to extract specific columns from a text file using filter utilities. 66. Write a shell command to fold long lines into shorter fixed widths. 67. Write a shell command to sort a file in ascending and descending order. 68. Write a shell command to translate lowercase characters to uppercase. 69. Write a program to simulate First Fit, Best Fit, Worst Fit memory allocation method. 70. Write a shell script to find and print only unique words from a given text file. 71. Write a shell script to display only unique lines from a file, the first 10 lines of file and last 10 lines of a file. 72. Write a shell command to combine two filters using a pipe. 73. Write a command using grep to search for a specific pattern in a file. 74. Write commands to use basic regular expressions to match specific strings. 75. Write a command using sed to replace all occurrences of a word in a file.	18	20%

GYANMANJARI INNOVATIVE UNIVERSITY GMIU BHAVNAGAR

76. Write a command using awk to print selected fields from a file.
77. Write a script to format output using printf.
78. Write a script that compares two numbers using comparison operators.
79. Write a program to implement FIFO, LRU, Optimal page replacement algorithm.
80. Solve the unsolved questions on FIFO, LRU, and Optimal page replacement algorithms given in the reference book.
81. Write a script that uses variables to store and display user input.
82. Write a script that prints built-in shell variables such as HOME, USER, and PATH.
83. Write a shell command to merge two files side by side.
84. Write a shell command to join two files based on a common field.

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Filter Forge Shell Challenge:** Write a shell script to process a text file using Unix filters (cut, sort, grep, awk, etc.), apply at least two filters with a pipe, format output with printf, use variables and comparison operators, and display well-formatted results showing each filter's effect.	20	
2	**Active Learning Assignment: Page Fault Adventure: Solve FIFO, LRU and Optimal:** Students will simulate FIFO, LRU, and Optimal page replacement algorithms, compare page faults using the given reference string, and submit their tables and results as a PDF on the GMIU portal.		10
	Total	20	10

Examination Style:

Filter Forge Shell Challenge (20 Marks)
Students will write a shell script that demonstrates the use of essential Unix filters (cut, paste, join, fold, sort, tr, uniq, head, tail) and advanced filters (grep, basic regular expressions, sed, awk). Your script must process a text file, apply at least two filters using a pipe, perform data formatting using printf, use comparison operators, variables, and built-in shell variables. Display meaningful, well-formatted output showing each filter's effect.

GYANMANJARI INNOVATIVE UNIVERSITY GMIU BHAVNAGAR

Page Fault Adventure: FIFO, LRU and Optimal (10 Marks)
Students are required to simulate page replacement algorithms using Python, C, Java, or any suitable online tool. The task involves applying FIFO, LRU, and Optimal page replacement algorithms to a given reference string (e.g., 7 0 1 2 0 3 0 4 2 3 0 3) and observing the loading of pages into frames. Students must construct a table illustrating the state of page frames at each step and compute the total number of page faults for each algorithm. A comparative analysis should be provided to determine which algorithm yields the best performance. All results, tables, and conclusions must be compiled individually into a PDF for submission on GMIU portal.

4 **Concurrency and Deadlock** 18 20%

Theory Topics:

Introduction to Process, Concurrent Processes, Process Communication, Principles of Concurrency and Mutual Exclusion, Synchronization Mechanisms – Semaphores, Monitors, and Message Passing, Deadlock concept, modeling of deadlock, conditions for deadlock (Mutual Exclusion, Hold and Wait, no preemption, Circular wait, dealing with deadlock, Deadlock Prevention, Deadlock Avoidance – Banker's Algorithm, Detection, Recovery, and Starvation, Classical IPC Problems – Producer–Consumer, Reader–Writer, Dining Philosophers. Communication commands: IP related commands (ifconfig, hostname, PING, Domain Information Groper (dig), route), Process Commands (Top, Ps, -r, -x, -e, -A, -a, -f, kill, mount, bg, fg, jobs, -l, -n, -p, -r, -s)

Practical:

85. Write a shell script to create concurrent processes and display their process IDs using ps.
86. Write a shell script to demonstrate inter-process communication by sending data through a pipe.
87. Write a shell script to implement mutual exclusion using a lock file mechanism.
88. Write a shell script to demonstrate process synchronization similar to a semaphore using a temporary file.
89. Write a shell script to detect a possible deadlock situation by checking if a process is waiting indefinitely.
90. Consider a system with 3 processes (P1, P2, P3) and 3 types of resources (A, B, C). The maximum resources required and currently allocated resources are as follows:

Process	Max (A,B,C)	Allocated (A,B,C)
P1	7, 5, 3	0, 1, 0
P2	3, 2, 2	2, 0, 0
P3	9, 0, 2	3, 0, 2

- The available resources are: A=3, B=3, C=2.
- Determine whether the system is in a safe state using the Banker's Algorithm.

GYANMANJARI INNOVATIVE UNIVERSITY
GMIU
BHAVNAGAR

- If a new request from P2 for (1,0,2) arrives, check if it can be safely allocated.

91. Solve the unsolved questions on the Banker's Algorithm provided in the reference book.
92. Write a shell script to simulate the Producer–Consumer problem using a file as a shared buffer.
93. Write a shell script to display the system's IP address, hostname, and test network connectivity using ping.
94. Write a shell script to display domain information and routing details using dig and route commands.
95. Use process commands (top, ps with options -r, -x, -e, -A, -a, -f, -l, -n, -p, -s) to list processes in various formats.
96. Write a shell script to list processes using ps options (-e, -f, -x, -a) and allow the user to kill a selected process ID.
97. Write a shell script to start a background job, bring it to the foreground, and display job status using bg, fg, and jobs commands

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Network & Process Explorer Task:** Students will use network commands to check connectivity and system commands to monitor and manage processes. They must record outputs and briefly explain each command's purpose in viva.	20	
2	**Concurrency & Synchronization Mini Task:** Students will implement one IPC problem using semaphores or message passing, show correct synchronization with outputs, and submit an individual PDF report on the GMIU Portal.		10
	Total	20	10

Examination Style:

Network & Process Explorer Task (20 Marks)

Students will perform a hands-on exercise using essential communication and process management commands. Using IP-related commands such as ifconfig, hostname, ping, dig, and route, students must identify network configurations, test connectivity, and retrieve DNS information. They must also explore system activity using process commands including top, ps (with options -r, -x, -e, -A, -a, -f), and manage processes using kill, bg, fg, jobs (with -l, -n, -p, -r, -s), and mount. Students should record the output of each command, explain its purpose as viva, and summarize how these commands help in monitoring and managing a Unix/Linux system.

	Concurrency & Synchronization Mini Task (10 Marks) Students will write a short program or simulation demonstrating process synchronization using either semaphores or message passing. The program should model one classical IPC problem—Producer–Consumer, Reader–Writer, or Dining Philosophers—and show how mutual exclusion is maintained to avoid race conditions. Students must briefly explain how their solution addresses concurrency, critical sections, and deadlock conditions, and provide output or screenshots as evidence of correct synchronization. Submit an individual report in PDF format on the GMIU Portal.		
5	**File Management, I/O Management, Disk Management, Security and Protection** ***Theory Topics:*** File System Concept, Files and File System, File Structure, File Naming and File Type, File Access, I/O Management: Organization of I/O Functions, Buffering, Disk Management, Disk Scheduling Algorithm (FCFS, SSTF, SCAN, C- SCAN), Solid State Drive, SSD Structure, SSD Features, Security Environment and Threat Models, Design Principles of Security, Computer Security Techniques (Authentication, Access Control, Intrusion Detection), User Authentication Techniques, Protection Mechanisms – Protection Domain, Access Control List (ACL), Malware Defense. File Management and Compression using Shell scripting: Command line argument, Introduction to array: Array Declaration, Printing Element Length, Printing Array Length, searching in the Array, Searching and Replacing in the Array, Deleting an Element from the Array, Printing All Elements, dealing with files (File, Find, locate, Whereis, which), Compressing and decompressing (Gzip, Gunzip, Zip, Unzip and Tar). ***Practical:*** 98. Write a shell script to display details of all files in a directory, including file type and size. 99. Write a shell script to read a filename from the user and display its file structure information (type, permissions, size). 100. Write a shell script to allow the user to enter a file name and check its access permissions (read, write, execute). 101. Write a shell script to simulate simple I/O buffering by reading a file line by line with delays. 102. Write a shell script to display all disks, partitions, and their usage information. 103. Write a shell script to simulate FCFS disk scheduling by taking a list of cylinder requests and printing the seek order. 104. Write a shell script to simulate SSTF disk scheduling for a given set of disk requests. 105. Solve the unsolved questions on Disk Scheduling Algorithms (FCFS, SSTF, SCAN, C-SCAN) given in the reference book.	18	20 %

106. Write a shell script to display user authentication information such as currently logged users and failed login attempts.
107. Write a shell script to display access control information of files using ls and getfacl commands.
108. Write a shell script to accept command-line arguments and display the number of arguments and their values.
109. Write a shell script using arrays to insert, delete, search, and replace an element in the array.
110. Write a shell script to search for files using file, find, locate, whereis, and which commands and display their locations.
111. Write a shell script to compress and decompress files using gzip, gunzip, zip, unzip, and tar based on user choice.

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Array Operations & File Utility Lab:** Write a shell script that performs key array operations and demonstrates file-handling and compression commands, with sample outputs.	20	
2	**Linux Security Case Study:** Students will choose one security scenario, perform required commands and fixes, document steps with screenshots, and submit an individual .docx case study report with key findings and a conclusion on the GMIU Portal.		10
	Total	20	10

Examination Style:

Array Operations & File Utility Lab (20 Marks)
Students will write a shell script that accepts command-line arguments and performs array operations such as declaring an array, displaying element length, printing the total array length, searching and replacing an element, deleting an element, and printing all elements. The script must also demonstrate file-handling commands (file, find, locate, whereis, which) and perform file compression and decompression using gzip, gunzip, zip, unzip, and tar. Include sample outputs for each operation.

Linux Security Case Study (10 Marks)
Students will undertake a practical Linux security exercise by selecting one scenario: system hardening, unauthorized access mitigation, or kernel and process security analysis. They are required to execute appropriate security and diagnostic commands, demonstrate corrective actions such as permission adjustments or process validations, and

GYANMANJARI INNOVATIVE UNIVERSITY GMIU BHAVNAGAR

	document their work through screenshots with brief explanations. Additionally, students must prepare a complete case study report that reflects conceptual understanding, including the background, problem identification, applied commands, results, and a well-structured conclusion, and submit the final report individually in .docx format on the GMIU Portal.		

Suggested Specification table with Marks (Theory): 100

Distribution of Theory Marks (Revised Bloom's Taxonomy)						
Level	Remembrance (R)	Understanding (U)	Application (A)	Analyze (N)	Evaluate (E)	Create (C)
Weightage %	15%	15%	20%	15%	15%	20%

Course Outcome:

After learning the course, the students should be able to:	
CO1	Understand operating systems, UNIX architecture, and shell programming, and develop scripts using decision statements, I/O redirection, and basic commands.
CO2	Analyze process and thread concepts, CPU scheduling, process control system calls, and develop advanced shell scripts with operators, loops, and multi-threading.
CO3	Apply memory management techniques and implement Unix filters and shell scripting concepts, including pipes, formatting, variables, and comparison operations.
CO4	Evaluate concurrency, synchronization, and deadlock handling in processes, and utilize IPC and system commands for process and network management.
CO5	Demonstrate file system management, disk scheduling, computer security principles, and file handling with compression using advanced shell scripting and arrays.

Instructional Method:

The course delivery method will depend upon the requirement of content and the needs of students. The teacher, in addition to conventional teaching methods by black board, may also use any tools such as demonstration, role play, Quiz, brainstorming, MOOCs etc.

The internal evaluation will be done on the basis of continuous evaluation of students in the laboratory and class-room.

Practical examination will be conducted at the end of semester for evaluation of performance of students in the laboratory.

Students will use supplementary resources such as online videos, NPTEL videos, e-courses, Virtual Laboratory

GYANMANJARI INNOVATIVE UNIVERSITY
GMIU
BHAVNAGAR

Reference Books:

[1] Operating System Concepts by Abraham Silberschatz, Peter B. Galvin, Greg Gagne, Wiley

[2] Unix Concepts and Applications by Sumitabha Das, Tata McGraw-Hill.

[3] Principles of Operating Systems by Naresh Chauhan, OXFORD University Press.

[4] Modern Operating Systems by Andrew S Tanenbaum, 3rd edition, PHI

[5] Operating Systems-Internals and Design Principles by William Stallings, 6th Edition, Pearson education.

Suggested Assessment Guidelines:

Module-1: Introduction to Operating System

SmartShell: A Unified File Handling Utility (20 Marks)		
Criteria	**Description**	**Marks**
Problem Understanding	Demonstrates clear understanding of shell scripting requirements: user input, file testing, conditions, I/O redirection, command execution.	5
Design	Use of shell types, system calls, test command, if–else/case statements.	5
Logic	Logical flow, correctness of script structure	5
Implementation	Fully functional shell script with user interaction (read/echo), correct execution of UNIX commands, proper redirection, accurate file/directory operations.	5
	Total	20

Module-2: Process Management and CPU Scheduling

Shell Scripting Proficiency Check (20 Marks)		
Criteria	**Description**	**Marks**
Understanding	Understanding of arithmetic, relational, logical operators and use of expr.	5
Logic & Control Flow	Correct application of loops (for/while/until) with proper use of break and continue.	5
Implementation	Working shell script that processes user input, performs operations correctly, and displays meaningful output	5
Problem Interpretation	Clear interpretation of the question requirements and accurate reflection in the script.	5
	Total	20

Module-3: Memory Management

Filter Forge Shell Challenge (20 Marks)		
Criteria	**Description**	**Marks**
Concept Understanding	Understanding of essential and advanced Unix filters (cut, sort, tr, uniq, grep, sed, awk, etc.) and their purpose in text processing.	5
Filter Application & Pipelining	Correct use of at least two filters with pipes, appropriate sequence of commands, and accurate filter effects on the text file.	5
Use of Shell Features	Proper use of variables, built-in shell variables, comparison operators, and formatting using printf.	5
Implementation & Output Quality	Script executes without errors, processes the input file correctly, and displays meaningful, well-formatted output.	5
	Total	20

Module-4: Concurrency & Deadlock

Network & Process Explorer Task (20 Marks)

Criteria	Description	Marks
Network Command Understanding	Correct use and interpretation of IP-related commands (ifconfig, hostname, ping, dig, route) to identify configuration, test connectivity, and retrieve DNS details.	5
Process Command Understanding	Effective use of process monitoring and management commands (top, ps options, kill, bg, fg, jobs, mount) with accurate interpretation of system activity.	5
Execution & Output Accuracy	Outputs recorded correctly, commands executed properly, and results reflect real system/network behavior.	5
Concept Explanation (Viva Performance)	Ability to clearly explain command purpose, usage, and how the results help in system monitoring and management.	5
	Total	20

Module-5: File Management, I/O Management, Disk Management, Security and Protection

Array Operations & File Utility Lab (20 Marks)

Criteria	Description	Marks
Array Concept Understanding	Correct usage of arrays: declaration, element length, total length, search/replace, delete, and printing all elements.	5
File Utility Command Application	Proper use of file-handling commands (file, find, locate, whereis, which) with correct interpretation of results.	5
Compression & Archiving Operations	Accurate execution of gzip, gunzip, zip, unzip, tar for compression and extraction tasks.	5
Implementation & Output Accuracy	Script runs without errors and includes correct sample outputs demonstrating each operation clearly.	5
	Total	20

GYANMANJARI INNOVATIVE UNIVERSITY GMIU BHAVNAGAR