Kurukshetra University, Kurukshetra

(Established by the State Legislature Act-XII of 1956)
("A++" Grade, NAAC Accredited)

Syllabus
for

Post Graduate Programme

Post Graduate Diploma in Computer Applications

as per NEP-2020
Curriculum and Credit Framework for Postgraduate Programme

With Multiple Entry-Exit, Internship and CBCS-LOCF
With effect from the session 2024-25 (in phased manner)

DEPARTMENT OF COMPUTER SCIENCE AND APPLICATIONS
FACULTY OF SCIENCES

KURUKSHETRA UNIVERSITY, KURUKSHETRA -136119

With effect from the Session: 2024-25			
Part A - Introduction			
Name of the Programme	PGDCA		
Semester	1st		
Name of the Course	Client-side Web Technology		
Course Code	M24-CAP-101		
Course Type	CC-1		
Level of the course (As per Annexure-I	400-499		
Pre-requisite for the course (if any)	-		
Course Objectives	This course aims to provide a comprehensive understanding of front-end development using the MERN stack, covering HTML, CSS, and JavaScript basics. Students will learn about React for building dynamic user interfaces, including components, state management, and event handling. The course also explores advanced topics such as React Router, Redux for state management, and advanced hooks for managing side effects and context.		
Course Learning Outcomes (CLO) After completing this course, the learner will be able to:	CLO-1. Gain an understanding of the web development process and the components of the MERN stack, with a focus on HTML structure, CSS styling, and responsive design. CLO-2 Develop foundational JavaScript skills, including control structures, functions, objects, arrays, and DOM manipulation for dynamic web interactions. CLO-3 Learn the basics of React, including JSX, components, state management, lifecycle methods, and handling events and forms within React applications. CLO-4 Master advanced React topics like React Router for navigation, state management with Redux, and using advanced hooks for managing complex state and side effects.		
Credits	Theory	Practical	Total
	4	0	4
Teaching Hours per week	4	0	4
Internal Assessment Marks	30	0	30
End Term Exam Marks	70	0	70
Max. Marks	100	0	100
Examination Time	3 hours		

Part B- Contents of the Course

Instructions for Paper- Setter: The examiner will set 9 questions asking two questions from each unit and one compulsory question by taking course learning outcomes (CLOs) into consideration. The compulsory question (Question No. 1) will consist at least 4 parts covering entire syllabus. The examinee will be required to attempt 5 questions, selecting one question from each unit and the compulsory question. All questions will carry equal marks.

Unit	Topics	Contact Hours
I	Basics of Front End Development: Overview of web development (Front End vs. Back End), Understanding the MERN stack and its components, Tools and environments (text editors, browsers, version control with Git); HTML (HyperText Markup Language): Structure of an HTML document, HTML elements and attributes, Forms and input types, Semantic HTML (header, footer, article, section, nav); CSS (Cascading Style Sheets): Basics of CSS (syntax, selectors, properties), CSS Box Model, Positioning and layout (float, flexbox, grid), Responsive design (media queries, mobile-first design).	15
II	Basics of JavaScript: Introduction to JavaScript, Variables, data types, and operators, Control structures (if, else, switch, loops); Functions and Scope: Defining and invoking functions, Function expressions and arrow functions, Scope and closures; Objects and Arrays: Creating and manipulating objects, Array methods and iteration; Regular Expressions: Introduction to RegExp, Regular expression usage, Modifiers, RegExp	15

	patterns, RegExp methods, String methods for RegExp; DOM Manipulation and Events: Selecting and manipulating DOM elements, Event handling and delegation, Creating and appending elements dynamically	
III	Introduction to React: Overview and advantages of React, Setting up a React development environment (using Create React App); JSX (JavaScript XML): Understanding JSX syntax, Embedding expressions in JS, JSX best practices; Components and Props: Functional and class components, Props and component communication, Prop types and default props.; State and Lifecycle: Understanding state in React, State management in class components, Lifecycle methods (componentDidMount, componentDidUpdate, componentWillUnmount); Event Handling and Forms: Handling events in React, Controlled vs. uncontrolled components, Form handling and validation	15
IV	React Router: Introduction to React Router, Setting up and configuring routes, Navigating between routes and passing parameters; State Management with Redux: Introduction to Redux, Setting up Redux with React, Actions, reducers, and store, Connecting Redux to React components; Advanced Hooks: Using built-in hooks (useEffect, useContext, useReducer), Creating custom hooks, Managing side effects with useEffect	15
	Total Contact Hours	60

Suggested Evaluation Methods			
Internal Assessment: 30		**End Term Examination: 70**	
➢ **Theory**	**30**	➢ **Theory**	**70**
• Class Participation:	5	Written Examination	
• Seminar/presentation/assignment/quiz/class test etc.:	10		
• Mid-Term Exam:	15		

Part C-Learning Resources

Reference Books:

1) Flanagan, D. (2020). *JavaScript: The Definitive Guide*. O'Reilly Media.
2) Kogent Learning. (2009). *Web Technologies: HTML, JavaScript, PHP, Java, JSP, XML, AJAX – Black Book*. Wiley India Pvt. Ltd.
3) Duckett, J. (2014). *JavaScript and jQuery: Interactive Front-End Web Development*. Wiley.
4) Robson, E., & Freeman, E. (2014). *Head First JavaScript Programming: A Brain-Friendly Guide*. O'Reilly Media.
5) Banks, A., & Chinnathambi, K. (2017). *Learning React: Functional Web Development with React and Redux*. O'Reilly Media.

With effect from the Session: 2024-25			
Part A - Introduction			
Name of the Programme	PGDCA		
Semester	1st		
Name of the Course	Operating System and Linux		
Course Code	M24-CAP-102		
Course Type	CC-2		
Level of the course (As per Annexure-I	400-499		
Pre-requisite for the course (if any)	-		
Course Objectives	This course provides a foundational understanding of operating systems, covering their definition, types, and functions. Students will explore system structures, process management, CPU scheduling, memory management, paging and segmentation, virtual memory, and file systems. Additionally, the course offers an introduction to Linux, including its history, architecture, file system, basic commands, shell scripting, process and user management, networking, system administration, and basic security concepts.		
Course Learning Outcomes (CLO) After completing this course, the learner will be able to:	CLO-1. Understand the fundamental concepts, functions, and structures of operating systems, and apply various CPU scheduling algorithms. CLO-2 Grasp memory hierarchy, allocation techniques, paging, segmentation, virtual memory concepts, and file system management. CLO-3 Learn the history, features, and architecture of Linux, perform basic file operations, and write simple shell scripts. CLO-4 Manage processes, users, and groups in Linux, utilize network commands, perform system administration tasks, and understand basic security measures.		
Credits	Theory	Practical	Total
	4	0	4
Teaching Hours per week	4	0	4
Internal Assessment Marks	30	0	30
End Term Exam Marks	70	0	70
Max. Marks	100	0	100
Examination Time	3 hours		

Part B- Contents of the Course

Instructions for Paper- Setter: The examiner will set 9 questions asking two questions from each unit and one compulsory question by taking course learning outcomes (CLOs) into consideration. The compulsory question (Question No. 1) will consist at least 4 parts covering entire syllabus. The examinee will be required to attempt 5 questions, selecting one question from each unit and the compulsory question. All questions will carry equal marks.

Unit	Topics	Contact Hours
I	Introduction to Operating Systems: Definition, types, and functions of an operating system; System Structures: Operating system services, system calls, system programs, and system structure; Process Management: Process concept, process scheduling, operations on processes, inter-process communication; CPU Scheduling: Scheduling criteria, scheduling algorithms (FCFS, SJF, Priority, Round Robin, Multilevel Queue Scheduling).	15
II	Memory Management: Memory Hierarchy, Types of memory, memory allocation techniques; Paging and Segmentation: Basic concepts, paging, segmentation, segmentation with paging; Virtual Memory: Demand paging, page replacement algorithms, allocation of frames, thrashing; File Systems: File concepts, access methods, directory and disk structure, file system mounting, file sharing, protection.	15
III	Introduction to Linux: History, features, architecture of Linux; Linux File System: File and directory structure, file permissions, standard file types; Basic Commands: File and	15

	directory operations (ls, cp, mv, rm, mkdir), text processing (cat, grep, sort), system status (ps, top, df, du); Shell Scripting: Introduction to shell, shell variables, control structures (if, case, while, for), writing simple shell scripts.	
IV	Process Management in Linux: Managing processes (ps, top, kill, nice), job scheduling (cron, at); User and Group Management: Creating and managing users and groups, file permissions, changing ownership (chown, chgrp); Networking in Linux: Basic network commands (ifconfig, ping, netstat, ssh), configuring network interfaces; System Administration: Package management (installing and removing software using rpm, dpkg, apt-get), backup and restore, logging; Security: Basic security concepts, user authentication.	15
	Total Contact Hours	60

Suggested Evaluation Methods			
Internal Assessment: 30		**End Term Examination: 70**	
➢ **Theory**	**30**	➢ **Theory**	**70**
1) Class Participation:	5	Written Examination	
2) Seminar/presentation/assignment/quiz/class test etc.:	10		
3) Mid-Term Exam:	15		

Part C-Learning Resources

Reference Books:

1) Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). Operating System Concepts (10th ed.). Wiley.
2) Tanenbaum, A. S., & Bos, H. (2014). Modern Operating Systems (4th ed.). Pearson.
3) Stallings, W. (2018). Operating Systems: Internals and Design Principles (9th ed.). Pearson.
4) Love, R. (2013). Linux System Programming (2nd ed.). O'Reilly Media.
5) Nemeth, E., Snyder, G., Hein, T. R., & Whaley, B. (2017). UNIX and Linux System Administration Handbook (5th ed.). Pearson.
6) Sobell, M. G. (2017). A Practical Guide to Linux Commands, Editors, and Shell Programming (4th ed.). Pearson.
7) Das, S. (2012). Your UNIX/Linux: The Ultimate Guide (3rd ed.). McGraw-Hill Education.
8) Kerrisk, M. (2010). The Linux Programming Interface: A Linux and UNIX System Programming Handbook. No Starch Press.

With effect from Session: 2024-25			
Part A - Introduction			
Name of the Programme	PGDCA		
Semester	1st		
Name of the Course	Data Structures		
Course Code	M24-CAP-103		
Course Type	CC-3		
Level of the course (As per Annexure-I	400-499		
Pre-requisite for the course (if any)	-		
Course Objectives	This course introduces fundamental concepts of algorithms and data structures, including algorithmic notation, programming principles, and program analysis. Students will explore arrays, searching and sorting techniques, stacks, queues, and linked lists, along with their applications. The course also covers tree structures such as binary trees, AVL trees, B-trees, and tries, as well as graph terminology, representation, and traversal methods. Additionally, students will learn about set operations, file queries, sequential organization, index techniques, and external sorting.		
Course Learning Outcomes (CLO) After completing this course, the learner will be able to:	CLO-1. Master algorithmic notation, programming principles, and implement arrays, searching and sorting techniques. CLO-2 Apply stack and queue operations, understand linked lists, and their applications including dynamic storage management. CLO-3 Comprehend binary trees, binary search trees, AVL trees, B-trees, B+ tree indexing, Trie tree indexing, and their applications. CLO-4 Utilize graph representations, traversals, applications, sets operations, and file organization techniques.		
Credits	Theory	Practical	Total
	4	0	4
Teaching Hours per week	4	0	4
Internal Assessment Marks	30	0	30
End Term Exam Marks	70	0	70
Max. Marks	100	0	100
Examination Time	3 hours		

Part B- Contents of the Course

Instructions for Paper- Setter: The examiner will set 9 questions asking two questions from each unit and one compulsory question by taking course learning outcomes (CLOs) into consideration. The compulsory question (Question No. 1) will consist at least 4 parts covering entire syllabus. The examinee will be required to attempt 5 questions, selecting one question from each unit and the compulsory question. All questions will carry equal marks.

Unit	Topics	Contact Hours
I	Introduction: Algorithmic notation – Programming principles – Creating programs- Analyzing programs. Arrays: One dimensional array, multidimensional array, pointer arrays. Searching: Linear search, Binary Search, Fibonacci search. Sorting techniques: Internal sorting - Insertion Sort, Selection Sort, Shell Sort, Bubble Sort, Quick Sort, Heap Sort, Merge Sort and Radix Sort.	15
II	Stacks: Definition – operations - applications of stack. Queues: Definition - operations - Priority queues – Dequeues – Applications of queue. Linked List: Singly Linked List, Doubly Linked List, Circular Linked List, linked stacks, Linked queues, Applications of Linked List – Dynamic storage management – Generalized list.	15
III	Trees: Binary tree, Terminology, Representation, Traversals, Applications – Binary search tree – AVL tree. B Trees: B Tree indexing, operations on a B Tree, Lower and upper bounds of a B Tree - B + Tree Indexing – Trie Tree Indexing.	15
IV	Graph: Terminology, Representation, Traversals – Applications - spanning trees, shortest path and Transitive closure, Topological sort. Sets: Representation - Operations on sets – Applications. Files: queries - Sequential organization – Index techniques. External sorting.	15

		Total Contact Hours	60
Suggested Evaluation Methods			
Internal Assessment: 30		**End Term Examination: 70**	
➢ **Theory**	**30**	➢ **Theory**	**70**
• Class Participation:	5	Written Examination	
• Seminar/presentation/assignment/quiz/class test etc.:	10		
• Mid-Term Exam:	15		

Part C-Learning Resources

Reference Books:

1) Horowitz, E., & Sahni, S. (2004). *Fundamentals of Data Structures*. Galgotia Book Source Pvt. Ltd.
2) Samanta, D. (2012). *Classic Data Structures* (2nd ed.). Prentice-Hall of India Pvt. Ltd., India.
3) Kruse, R., Tondo, C. L., & Leung, B. (2007). *Data Structures and Program Design in C* (2nd ed.). Prentice-Hall of India Pvt. Ltd.
4) Weiss, M. A. (2006). *Data Structures and Algorithm Analysis in C* (2nd ed.). Pearson Education.

With effect from Session: 2024-25			
Part A - Introduction			
Name of the Programme	PGDCA		
Semester	1st		
Name of the Course	Programming in JAVA		
Course Code	M24-CAP-104		
Course Type	CC-4		
Level of the course (As per Annexure-I	400-499		
Pre-requisite for the course (if any)	-		
Course Objectives	This course provides a comprehensive introduction to Java, covering its history, features, and applications. Students will learn Java programming basics, including syntax, variables, control flow, methods, and arrays. The course also delves into object-oriented programming concepts such as classes, objects, encapsulation, inheritance, polymorphism, and interfaces. Additionally, students will explore advanced topics like exception handling, file handling, multithreading, event handling, generics, JDBC for database connectivity, and GUI programming with Swing.		
Course Learning Outcomes (CLO) After completing this course, the learner will be able to:	CLO-1. Understand Java's background, features, and apply fundamental programming concepts including variables, operators, control flow, methods, and arrays. CLO-2 Master object-oriented programming principles including classes, objects, inheritance, polymorphism, interfaces, and packaging in Java. CLO-3 Gain proficiency in handling exceptions, working with files, implementing multithreading, and utilizing Java Collections for efficient data management. CLO-4 Explore and utilize advanced Java features such as generics, lambda expressions, JDBC for database connectivity, and GUI programming with JavaFX or Swing.		
Credits	Theory	Practical	Total
	4	0	4
Teaching Hours per week	4	0	4
Internal Assessment Marks	30	0	30
End Term Exam Marks	70	0	70
Max. Marks	100	0	100
Examination Time	3 hours		

Part B- Contents of the Course

Instructions for Paper- Setter: The examiner will set 9 questions asking two questions from each unit and one compulsory question by taking course learning outcomes (CLOs) into consideration. The compulsory question (Question No. 1) will consist at least 4 parts covering entire syllabus. The examinee will be required to attempt 5 questions, selecting one question from each unit and the compulsory question. All questions will carry equal marks.

Unit	Topics	Contact Hours
I	Introduction to Java: History, features, and applications; Basics of Java programming: Syntax, variables, data types, operators, expressions, and statements; Control flow: Decision-making statements (if, else-if, switch), looping statements (for, while, do-while), and branching; Methods: Declaring methods, passing parameters, method overloading, and recursion; Arrays: Declaring, initializing, and manipulating arrays. Array operations and algorithms.	15
II	Classes and Objects: Declaring classes, creating objects, constructors, and instance variables; Encapsulation: Access modifiers (public, private, protected, default), getters, and setters; Inheritance: Extending classes, method overriding, super keyword, and method overloading; Polymorphism: Method overriding, dynamic method dispatch, and abstract classes; Interfaces: Defining interfaces, implementing interfaces, and using interface	15

	references; Packages: Creating and using packages, importing classes and packages.	
III	Exception Handling: Understanding exceptions, try-catch block, throw and throws keywords, and finally block; File Handling: Reading from and writing to files using FileInputStream, FileOutputStream, FileReader, and FileWriter; Multithreading: Creating threads, thread lifecycle, synchronization, thread communication. Applet programming, Applet life Cycle, Applet Graphics programming.	15
IV	Event Handling: AWT Classes, ActionListener, MouseListener, MouseMotionListener, Layout managers, Generics: Introduction to generics, generic classes and generic methods, Java Database Connectivity (JDBC): Connecting to databases, executing SQL queries, handling transactions, and managing resources; GUI Programming: Introduction to Swing for creating graphical user interfaces (GUIs).	15
	Total Contact Hours	60

Suggested Evaluation Methods

Internal Assessment: 30		**End Term Examination: 70**	
➢ **Theory**	**30**	➢ **Theory**	**70**
• Class Participation:	5	Written Examination	
• Seminar/presentation/assignment/quiz/class test etc.:	10		
• Mid-Term Exam:	15		

Part C-Learning Resources

Reference Books:

1) Balaguruswamy, E. (2009). *Programming with JAVA: A Primer*. Tata McGraw Hill.
2) Naughton, P., & Schildt, H. (2002). *The Complete Reference Java 2*. Tata McGraw Hill.
3) Neimeyer, P., & Peck, J. (1996). *Exploring Java*. O'Reilly.
4) Hahn, H. (1996). *Teach Yourself the Internet*. Prentice-Hall of India (P.H.I.).
5) Boone, B., & Stanek, W. (2001). *Java 2 Exam Guide*. Tata McGraw Hill.

With effect from Session: 2024-25			
Part A - Introduction			
Name of the Programme	PGDCA		
Semester	Ist		
Name of the Course	Practical-1		
Course Code	M24-CAP-105		
Course Type	PC-1		
Level of the course	400-499		
Pre-requisite for the course (if any)			
Course objectives	This is a laboratory course and the objective of this course is to acquaint the students with the understanding and implementing of client-side web technologies. Also, the concepts of operating systems and shell programming will be implemented by the students.		
Course Learning Outcomes (CLO) After completing this course, the learner will be able to:	CLO 1: Solve practical problems related to theory courses undertaken in the CC-1 and CC-2 from application point of view. CLO 2: Know how to use the client-side web technologies. CLO 3: implement the various functions of operating systems. CLO 4: Designing and implementing the shell programs in Linux.		
Credits	Theory	Practical	Total
	0	4	4
Teaching Hours per week	0	8	8
Internal Assessment Marks	0	30	30
End Term Exam Marks	0	70	70
Max. Marks	0	100	100
Examination Time	0	4 hours	

Part B- Contents of the Course	
Practicals	**Contact Hours**
Practical course will consist of two components Part-A and Part-B. The examiner will set 5 questions at the time of practical examination asking 2 questions from the Part-A and 3 questions from the Part-B by taking course learning outcomes (CLO) into consideration. The examinee will be required to solve one problem from the Part-A and to write and execute 2 questions from the Part-B.	120
Part-A **HTML/CSS Basics:** • Creating a webpage structure with HTML. • Styling the webpage using CSS (inline, internal, and external styles). **Responsive Design:** • Making the webpage responsive using media queries. • Using frameworks like Bootstrap for responsive design. **JavaScript Basics:** • Adding interactivity with JavaScript (DOM manipulation, event handling). • Working with variables, loops, and conditions. **Frameworks and Libraries:** • Using front-end frameworks React. • Utilizing libraries such as jQuery for DOM manipulation. **Introduction to React:** • Create a simple React component that displays "Hello, World!" on the screen. • Use JSX syntax and explain its advantages over plain JavaScript. **State and Props:** • Build a component that takes props and renders them. • Implement state in a component and update it based on user interaction (e.g., button click). **Basic Todo App:** Develop a Todo application where users can add, delete, and mark tasks as completed. Use state to manage the list of tasks. **Using React Router:**	60

• Set up React Router in a project and create multiple pages (e.g., Home, About, Contact). • Implement navigation between these pages using Link and NavLink. **Redux Integration:** • Integrate Redux for state management in a React application. • Implement actions, reducers, and connect components to Redux store. **Responsive Design with React Router:** • Build a responsive multi-page application using React Router. • Ensure layout adjustments for different screen sizes using CSS media queries or frameworks like Bootstrap.	
Part-B 1) Implement a simple program demonstrating the creation and synchronization of threads or processes. 2) Design and simulate a memory management system (e.g., paging, segmentation). 3) Implement algorithms like First Fit, Best Fit, and Worst Fit for memory allocation. 4) Implement a basic file system with operations like file creation, deletion, reading, and writing. 5) Compare different file allocation methods (e.g., contiguous allocation, linked allocation, indexed allocation). 6) Solve synchronization problems such as the producer-consumer problem or dining philosophers problem using semaphores or mutexes. 7) Implement a solution for deadlock prevention, avoidance, or detection. 8) Profile and analyze the performance of different scheduling algorithms (e.g., FCFS, SJF, Round Robin) using simulations. 9) Evaluate the impact of caching and paging strategies on system performance. 10) Write a shell script named hello.sh that prints "Hello, World!" to the terminal when executed. 11) Demonstrate running the script and explain how to make it executable using chmod. 12) Write a script greet_user.sh that prompts the user for their name and then prints a personalized greeting. 13) Use variables to store user input and demonstrate the use of read command. 14) Create a script check_number.sh that accepts a number as an argument. 15) Check if the number is positive, negative, or zero, and print an appropriate message using conditional statements (if-else). 16) Develop a script countdown.sh that takes a number as input and prints a countdown from that number to 1. 17) Use a loop (e.g., while or for) to implement the countdown. 18) Write a script file_info.sh that accepts a filename as an argument. 19) Check if the file exists and whether it is a regular file or directory. Display appropriate messages based on the checks. 20) Create a script word_count.sh that reads a text file (provided as an argument) and counts the number of words in the file. 21) Utilize command-line tools like wc and cat for reading and counting words.	60 (Lab hours include instructions for writing programs and demonstration by a teacher and for running the programs on computer by students.)

Suggested Evaluation Methods

Internal Assessment: 30		**End Term Examination: 70**	
➢ **Practicum**	**30**	➢ **Practicum**	**70**
• Class Participation:	5	Lab record, Viva-Voce, write-up and execution of the programs	
• Seminar/Demonstration/Viva-voce/Lab records etc.:	10		
• Mid-Term Examination:	15		

Part C-Learning Resources

Recommended Books/e-resources/LMS:

1) Flanagan, D. (2020). *JavaScript: The Definitive Guide*. O'Reilly Media.
2) Kogent Learning. (2009). *Web Technologies: HTML, JavaScript, PHP, Java, JSP, XML, AJAX – Black Book*. Wiley India Pvt. Ltd.
3) Duckett, J. (2014). *JavaScript and jQuery: Interactive Front-End Web Development*. Wiley.
4) Robson, E., & Freeman, E. (2014). *Head First JavaScript Programming: A Brain-Friendly Guide*. O'Reilly Media.

5) Banks, A., & Chinnathambi, K. (2017). *Learning React: Functional Web Development with React and Redux*. O'Reilly Media.
6) Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). Operating System Concepts (10th ed.). Wiley.
7) Tanenbaum, A. S., & Bos, H. (2014). Modern Operating Systems (4th ed.). Pearson.
8) Stallings, W. (2018). Operating Systems: Internals and Design Principles (9th ed.). Pearson.
9) Love, R. (2013). Linux System Programming (2nd ed.). O'Reilly Media.
10) Nemeth, E., Snyder, G., Hein, T. R., & Whaley, B. (2017). UNIX and Linux System Administration Handbook (5th ed.). Pearson.
11) Sobell, M. G. (2017). A Practical Guide to Linux Commands, Editors, and Shell Programming (4th ed.). Pearson.
12) Das, S. (2012). Your UNIX/Linux: The Ultimate Guide (3rd ed.). McGraw-Hill Education.
13) Kerrisk, M. (2010). The Linux Programming Interface: A Linux and UNIX System Programming Handbook. No Starch Press

With effect from Session: 2024-25			
Part A - Introduction			
Name of the Programme	PGDCA		
Semester	I[st]		
Name of the Course	Practical-2		
Course Code	M24-CAP-106		
Course Type	PC-2		
Level of the course	400-499		
Pre-requisite for the course (if any)			
Course objectives	This is a laboratory course and the objective of this course is to acquaint the students with the understanding and implementation of various data structures. Also, the students will implement the concepts of programming with Java.		
Course Learning Outcomes (CLO) After completing this course, the learner will be able to:	CLO 1: Solve practical problems related to theory courses undertaken in the CC-3 and CC-4 from an application point of view. CLO 2: Know how to use and implement the various data structures. CLO 3: Implement the various features of Java Programming by writing suitable programs. CLO 4: Designing and implementing applications in Java.		
Credits	Theory	Practical	Total
	0	4	4
Teaching Hours per week	0	8	8
Internal Assessment Marks	0	30	30
End Term Exam Marks	0	70	70
Max. Marks	0	100	100
Examination Time	0	4 hours	

Part B- Contents of the Course	
Practicals	**Contact Hours**
Practical course will consist of two components Part-A and Part-B. The examiner will set 5 questions at the time of practical examination asking 2 questions from the Part-A and 3 questions from the Part-B by taking course learning outcomes (CLO) into consideration. The examinee will be required to solve one problem from the Part-A and to write and execute 2 questions from the Part-B.	120
Part-A **Task 1: Linked List Implementation** • Implement a singly linked list in a programming language of your choice (e.g., C/C++, Java, Python). • Include functions/methods for insertion (at the beginning, end, and specific position), deletion, and traversal. **Task 2: Stack Operations** • Implement a stack using an array or linked list. • Include functions/methods for push, pop, peek, and checking if the stack is empty or full. **Task 3: Queue Implementation** • Implement a queue using an array or linked list. • Include functions/methods for enqueue, dequeue, peek, and checking if the queue is empty or full. **Task 4: Binary Search Tree (BST) Operations** • Implement a binary search tree (BST) in your chosen programming language. • Include functions/methods for insertion, deletion, searching for a key, finding minimum and maximum values, and traversing the tree (inorder, preorder, postorder). **Task 6: Sorting Algorithms** • Implement at least two sorting and searching algorithms (e.g., selection sort, insertion sort, merge sort, quick sort). • Compare their time complexity and performance using different input sizes.	60

Task 7: Graph Representation and Algorithms • Implement an adjacency list representation of a graph. • Include functions/methods for BFS (Breadth-First Search) and DFS (Depth-First Search) traversal of the graph.	
Part-B 1) Write a Java program that converts temperatures between Celsius and Fahrenheit based on user input using methods for conversion and input validation. 2) Implement a Java program to perform matrix addition, multiplication, and transpose operations using arrays and methods. 3) Develop a Java program that converts a decimal number to its binary, octal, and hexadecimal equivalents using loops and methods. 4) Create a Java program to simulate a simple bank account management system with features like deposit, withdrawal, and balance inquiry using classes, objects, and encapsulation. 5) Write a Java program that reads a text file, counts the occurrences of each word, and displays the top N most frequent words using HashMap for storage and sorting. 6) Implement a Java program to generate the first N prime numbers using a combination of loops, methods, and optimizations like the Sieve of Eratosthenes algorithm. 7) Develop a Java program that takes a month and year as input and prints the calendar for that month using control flow statements and loops for date calculation. 8) Write a Java program that generates different number patterns like pyramid patterns using nested loops and methods for pattern printing. 9) Create a Java program to manage an employee payroll system with features for adding employees, calculating salaries based on hours worked or monthly salary, and generating pay slips using classes, inheritance, and polymorphism. 10) Implement Java programs to compare the performance of different sorting algorithms (like quicksort, mergesort, and heapsort) on large arrays of integers, measuring and analyzing time complexity. 11) Develop a Java program that recursively searches a directory for files matching a given pattern and displays the file paths using recursion and file handling classes. 12) Write a Java program to perform arithmetic operations (addition, subtraction, multiplication, division) on large numbers using BigInteger class and exception handling for division by zero. 13) Implement a Java program to solve the Tower of Hanoi problem for N disks using recursion, demonstrating the steps and movements required. 14) Write a Java program to find the largest and smallest elements in an array. 15) Implement a Java program to sort an array of integers using bubble sort. 16) Create a Java program to find the frequency of each element in an array. 17) Develop a Java program to reverse an array without using an additional array. 18) Write a Java program to merge two sorted arrays into a single sorted array. 19) Define a Java class representing a Student with private instance variables and public getter and setter methods. 20) Create a Java program to demonstrate constructor overloading in a class. 21) Implement a Java program to calculate the area and perimeter of a rectangle using a class and object. 22) Develop a Java program to implement inheritance by creating a base class Animal and derived classes like Dog and Cat. 23) Write a Java program to demonstrate method overriding by implementing a base class Shape and derived classes like Circle and Rectangle.	60 (Lab hours include instructions for writing programs and demonstration by a teacher and for running the programs on computer by students.)

Suggested Evaluation Methods			
Internal Assessment: 30		**End Term Examination: 70**	
➢ **Practicum**	**30**	➢ **Practicum**	**70**
• Class Participation:	5	Lab record, Viva-Voce, write-up and execution of the programs	
• Seminar/Demonstration/Viva-voce/Lab records etc.:	10		
• Mid-Term Examination:	15		

Part C-Learning Resources

Recommended Books/e-resources/LMS:

1) Horowitz, E., & Sahni, S. (2004). *Fundamentals of Data Structures*. Galgotia Book Source Pvt. Ltd.
2) Samanta, D. (2012). *Classic Data Structures* (2nd ed.). Prentice-Hall of India Pvt. Ltd., India.
3) Kruse, R., Tondo, C. L., & Leung, B. (2007). *Data Structures and Program Design in C* (2nd ed.). Prentice-Hall of India Pvt. Ltd.
4) Weiss, M. A. (2006). *Data Structures and Algorithm Analysis in C* (2nd ed.). Pearson Education.
5) Balaguruswamy, E. (2009). *Programming with JAVA: A Primer*. Tata McGraw Hill.
6) Naughton, P., & Schildt, H. (2002). *The Complete Reference Java 2*. Tata McGraw Hill.
7) Neimeyer, P., & Peck, J. (1996). *Exploring Java*. O'Reilly.
8) Hahn, H. (1996). *Teach Yourself the Internet*. Prentice-Hall of India (P.H.I.).
9) Boone, B., & Stanek, W. (2001). *Java 2 Exam Guide*. Tata McGraw Hill.

With effect from Session: 2024-25			
Part A - Introduction			
Name of the Programme	PGDCA		
Semester	1st		
Name of the Course	Computer Fundamentals and Problem Solving Through C		
Course Code	M24-CAP-108		
Course Type	BC-1		
Level of the course (As per Annexure-I	400-499		
Pre-requisite for the course (if any)	-		
Course Objectives	The objective of this course is to provide a foundational understanding of computer systems, including hardware and software components, and to introduce essential concepts of digital systems, number systems, and Boolean logic. The course also aims to develop proficiency in programming using the C language, focusing on control structures, functions, data structures, and pointers. By the end of the course, students will be able to apply fundamental programming techniques to solve computational problems and have a strong grasp of the underlying principles of digital logic and computing.		
Course Learning Outcomes (CLO) After completing this course, the learner will be able to:	CLO-1. Students will be able to explain the basic organization of a computer and understand the purpose and methods of program planning using algorithms, flowcharts, and pseudocodes. CLO-2. Students will develop the ability to represent and manipulate information using various number systems, binary arithmetic, and Boolean logic. CLO-3. Students will acquire proficiency in programming with the C language, including the use of data types, operators, control structures, and input/output operations. CLO-4. Students will demonstrate the ability to create modular programs in C using functions, effectively manage data structures such as arrays, strings, and files, and work with pointers to manipulate memory and data efficiently.		
Credits	Theory	Practical	Total
	0	0	0
Teaching Hours per week	4	0	4
Internal Assessment Marks	30	0	30
End Term Exam Marks	70	0	70
Max. Marks	100	0	100
Examination Time	3 hours		

Part B- Contents of the Course

Instructions for Paper- Setter: The examiner will set 9 questions asking two questions from each unit and one compulsory question by taking course learning outcomes (CLOs) into consideration. The compulsory question (Question No. 1) will consist at least 4 parts covering entire syllabus. The examinee will be required to attempt 5 questions, selecting one question from each unit and the compulsory question. All questions will carry equal marks.

Unit	Topics	Contact Hours
I	Computer Fundamentals: Basics of computers, basic computer organization, storage hierarchy, storage devices, input-output devices. Computer Software. Introduction to operating systems. Planning the computer program: Purpose of program planning, algorithm, flowcharts, decision tables, pseudocodes.	15
II	Digital Fundamentals: Information representation - number systems, number system conversion; Computer codes - BCD code, EBCDIC code, ASCII, Unicode; Binary arithmetic; Binary logic - Boolean algebra, Boolean functions, truth table, simplification of	15

	Boolean functions (upto 4 variables only), K-map, digital logic gates.	
III	Elements of C language: C character set, identifiers & keywords, data types: declaration & definition. Operators: Arithmetic relational, logical, bitwise, unary, assignment and conditional operators & their hierarchy & associativity, Data input/output. Control statements: Sequencing, Selection: if and switch statement; iteration: for, while, and do-while loop; break, continue, goto statement.	15
IV	Functions in C language: Definition, prototype, passing parameters, recursion, Data structure: arrays, structures, union, string, data files. Pointers: Declaration, operations on pointers, array of pointers, pointers to arrays.	15
	Total Contact Hours	60

Suggested Evaluation Methods			
Internal Assessment: 30		**End Term Examination: 70**	
➢ **Theory**	**30**	➢ **Theory**	**70**
• Class Participation:	5	Written Examination	
• Seminar/presentation/assignment/quiz/class test etc.:	10		
• Mid-Term Exam:	15		

Part C-Learning Resources

Reference Books:

1) Balagurusamy, E. *Programming in ANSI C*. 8th ed., McGraw Hill, 2019. ISBN: 9789353165129.
2) Morris Mano, M. *Digital Logic and Computer Design*. 1st ed., Pearson, 2016. ISBN: 9789332551763.
3) Forouzan, Behrouz A. *Fundamentals of Computer Science: Computer Essentials*. 3rd ed., Cengage Learning, 2008. ISBN: 9788131512456.
4) Kernighan, Brian W., and Dennis M. Ritchie. *The C Programming Language*. 2nd ed., Pearson Education, 1988. ISBN: 9780131103627.

With effect from Session: 2024-25			
Part A - Introduction			
Name of the Programme	PGDCA		
Semester	Ist		
Name of the Course	Practical-3		
Course Code	M24-CAP-109		
Course Type	BC-2		
Level of the course	400-499		
Pre-requisite for the course (if any)			
Course objectives	This course focuses on hands-on experience with computer fundamentals. They will engage in program planning by creating and testing algorithms, flowcharts, and pseudocodes. Practical sessions will deepen their understanding of digital fundamentals through exercises on number systems, Boolean logic, and binary arithmetic. The course will provide extensive practice in C programming, allowing students to implement various data types, control structures, functions, and pointers in real-world coding tasks.		
Course Learning Outcomes (CLO) After completing this course, the learner will be able to:	CLO 1: Design and implement efficient algorithms using flowcharts, pseudocodes, and decision tables to solve complex problems. CLO 2: Write C programs that demonstrate a strong understanding of control structures, data types, and operators to create optimized solutions. CLO 3: Develop modular C programs using functions, effectively managing code complexity and promoting reusability. CLO 4: Utilize pointers and data structures in C to enhance program efficiency and handle dynamic memory management in real-world applications.		
Credits	Theory	Practical	Total
	0	0	0
Teaching Hours per week	0	4	4
Internal Assessment Marks	0	15	15
End Term Exam Marks	0	35	35
Max. Marks	0	50	50
Examination Time	0	4 hours	

Part B- Contents of the Course

Practicals	**Contact Hours**
The examiner will set 3 questions at the time of practical examination by taking course learning outcomes (CLO) into consideration. The examinee will be required to write and execute 2 questions.	60
1) Implement a program using the conditional (ternary) operator to find the largest of three numbers. 2) Create a C program that acts as a simple calculator, performing addition, subtraction, multiplication, or division based on user input using the switch statement. 3) Write a program that uses `if-else` statements to determine whether a given year is a leap year or not. 4) Develop a C program using a `for` loop to print the multiplication table of a given number up to 10. 5) Write a C program to calculate the factorial of a number using both `while` and `do-while` loops. 6) Implement a program that uses `break` and `continue` statements within a loop	60 (Lab hours include instructions for writing programs and demonstration by a teacher and for running the programs on computer by students.)

to skip printing even numbers and stop the loop if the number exceeds 50.

7) Write a C program with a function that takes an integer as input and returns the square of the number. Call this function from `main()`.
8) Develop a program that includes a function to calculate the area of a circle given the radius. Use `float` as the return type.
9) Create a C program that calculates the nth Fibonacci number using recursion.
10) Write a program that uses a function to find the maximum value in an integer array. The array should be passed to the function as a parameter.
11) Implement a program that uses functions to reverse a string and check if the string is a palindrome.
12) Write a C program that defines a structure to store student details (name, roll number, marks in three subjects) and calculates the total and average marks. Use a union to demonstrate memory sharing between different types.
13) Pointers in C Language
14) Pointer Basics: Write a program that demonstrates the use of pointers by printing the address and value of a variable using both the variable itself and a pointer to the variable.
15) Create a C program to store an array of strings (names of students) using an array of pointers. Display the names in reverse order.
16) Implement a program that uses a pointer to a function to pass a function as a parameter to another function, e.g., passing a function that calculates the square of a number to another function that prints it.

Suggested Evaluation Methods

Internal Assessment: 15		**End Term Examination: 35**	
➢ **Practicum**	**15**	➢ **Practicum**	**35**
• Class Participation:	4	Lab record, Viva-Voce, write-up and execution of the programs	
• Seminar/Demonstration/Viva-voce/Lab records etc.:	4		
• Mid-Term Examination:	7		

Part C-Learning Resources

Reference Books:

1) Balagurusamy, E. *Programming in ANSI C*. 8th ed., McGraw Hill, 2019. ISBN: 9789353165129.
2) Morris Mano, M. *Digital Logic and Computer Design*. 1st ed., Pearson, 2016. ISBN: 9789332551763.
3) Forouzan, Behrouz A. *Fundamentals of Computer Science: Computer Essentials*. 3rd ed., Cengage Learning, 2008. ISBN: 9788131512456.
4) Kernighan, Brian W., and Dennis M. Ritchie. *The C Programming Language*. 2nd ed., Pearson Education, 1988. ISBN: 9780131103627.