

PROGRAMMING FOR PROBLEM SOLVING THROUGH 'C'

I Year B. Tech. I Semester

(Common to All Branches)

(CHEM, CIVIL, CSE, CSM, CSD, CSC, CSF, ECE, ECE (VLSI), EEE, EEE (EV), IT,
MECH, MECH (R & AI))

Course Code:

L T P C

3 0 2 4

Course Objectives

After completing this course, students should be able to:

1. Develop computational thinking and problem-solving skills by analyzing engineering problems and representing their solutions using algorithms, flowcharts, and pseudocode.
2. Understand and apply fundamental C programming concepts such as data types, operators, input/output statements, decision-making, and looping constructs to solve numerical and logical problems.
3. Develop modular and reusable C programs using functions, parameter passing, recursion, and scope-related concepts.
4. Apply arrays, strings, pointers, and dynamic memory allocation for effective data manipulation and storage.
5. Utilize structures, unions, and file-handling techniques to efficiently represent, organize, store, retrieve, and process structured data in real-world applications.

CO	Course Outcome	Bloom's Level
CO1	Explain computational thinking approaches and programming fundamentals for representing engineering problems using algorithms, flowcharts, pseudocode, and basic C constructs.	L2 – Understand
CO2	Solve numerical and logical problems using operators, input/output statements, decision-making, and looping constructs in C.	L3 – Apply
CO3	Make Use of functions, parameter passing, recursion, and scope-related concepts to organize C programs into reusable components.	L3 – Apply
CO4	Demonstrate the use of arrays, strings, pointers, and dynamic memory allocation for storing, accessing, and manipulating data.	L3 – Apply
CO5	Apply structures, unions, and file-handling mechanisms for representing, storing, retrieving, and processing structured data.	L3 – Apply

UNIT-I: Problem Solving and Introduction to C: Language development, Structured Programming, Problem Solving Methodology, Algorithms, List of C standards,

Introduction to C, C Character Set, Identifiers and Keywords, Variables and Constants, Data Types, Expressions, Statements, Compilation and Execution of C Programs.

UNIT-II: Operators, Input/Output and Decision Making: Arithmetic Operators, Unary Operators, Relational Operators, Logical Operators, Assignment Operators, Conditional Operator, Standard Library Functions.

Data Input and Output: getchar(), putchar(), scanf(), printf(), if, if-else, Nested if-else, switch, while, do-while, for, break, continue, goto, comma operator and Debugging Techniques.

UNIT-III: Arrays, Strings and Pointers: Defining and Processing an array, One-dimensional Arrays, Two-dimensional Arrays, Character Arrays, Defining Strings, Initialization of strings, reading and writing strings, String Library Functions, Pointer Basics, Pointer Arithmetic, Pointers and Arrays, Arrays of Pointers, Dynamic Memory Allocation.

UNIT-IV: Functions and Program Organization: Defining a function, accessing a function, function Prototypes, function Calls, Parameter passing, Recursion, Storage Classes, Scope and Lifetime of Variables, Static Variables, Global Variables.

Structures and Unions: Defining a structure, Processing a Structure, Arrays of Structures, Nested Structures, typedef, Structures and Functions, Structures and Pointers, Self-Referential Structures, and Unions.

UNIT-V: File Handling: Why Files, Opening and Closing a Data File, Reading and Writing a Data File, processing a data File.

Low Level Programming: Register variables, Bitwise Operations, Bit Fields.

Additional Features of C: Enumerations, Command Line Arguments, More about Library functions, Macros, The C Preprocessor. Brief overview of the difference's standards of C Language.

Textbooks:

1. Behrouz A. Forouzan and Richard F. Gilberg, *Computer Science: A Structured Programming Approach Using C*, 4th Edition, Cengage Learning, 2020.
2. Byron S. Gottfried, *Schaum's Outline of Programming with C*, 3rd Edition, McGraw-Hill Education (India) Private Limited, 2018.

Reference Books

1. K. N. King, *C Programming: A Modern Approach*, 2nd Edition, W. W. Norton & Company, 2008.
2. Reema Thareja, *Programming in C*, 3rd Edition, Oxford University Press, 2023.
3. Ashok N. Kamthane and Amit A. Kamthane, *Programming in C*, 3rd Edition, Pearson Education, 2025.

Note: Faculty should demonstrate 2–3 representative examples from the lab programs during the theory session and encourage students to practice the remaining programs independently as part of the laboratory exercise.

Online Resources:

1. <https://ocw.mit.edu/courses/6-087-practical-programming-in-c-january-iap-2010/>
2. <https://pll.harvard.edu/course/cs50-introduction-computer-science>
3. <https://nptel.ac.in/courses/106105171>
4. <https://www.hackerrank.com/domains/c>
5. <https://www.codechef.com/learn/course/c>

PROGRAMMING FOR PROBLEM SOLVING THROUGH 'C' LAB

Course Outcomes: At the end of the course, the student shall be able to:

CO1: Develop C programs using variables, data types, operators, expressions, and input/output statements to solve simple computational problems.

CO2: Develop C programs using decision-making and iterative control statements.

CO3: Develop modular C programs using user-defined functions, recursion, arrays, and strings.

CO4: Develop C programs using pointers, dynamic memory allocation, structures, and unions.

CO5: Develop file-based applications using text files for data storage, retrieval, and processing.

Note:

1. Students must write algorithm and draw flow chart up to Functions concept
2. Any three programs from each experiment must be completed.
3. Minimum 10 experiments must be completed.

Experiment 1: Write C programs to:

1. Display "Hello World"
2. Read and print student details
3. Perform arithmetic operations
4. Convert temperature ($^{\circ}\text{C} \leftrightarrow ^{\circ}\text{F}$)
5. Calculate simple and compound interest

Experiment 2: Write C programs to:

1. Evaluate arithmetic expressions.
2. Demonstrate relational and logical operators.
3. Demonstrate increment/decrement operators.
4. Demonstrate assignment operators.
5. Find the maximum of three numbers using the conditional operator.

Experiment 3: Write programs using if, if-else, nested if, and switch to:

1. Check whether a number is even or odd
2. Find the largest among three numbers
3. Determine whether a year is a leap year
4. Calculate grades based on marks
5. Develop a menu-driven calculator using switch

Experiment 4: Write programs to:

1. Generate multiplication tables
2. Compute factorial of a number
3. Generate Fibonacci series
4. Check prime numbers
5. Check Armstrong, palindrome, perfect, and strong numbers

Experiment 5: Write programs using user-defined functions to:

1. Perform arithmetic operations
2. Find GCD and LCM
3. Compute factorial
4. Find power of a number
5. Demonstrate parameter passing using call by value concept.

Experiment 6: Write recursive programs to:

1. Factorial
2. Sum of first N natural numbers

3. Fibonacci series
4. GCD of two numbers
5. Tower of Hanoi

Experiment 7: Write programs to:

1. Find sum and average of array elements
2. Find largest and smallest element in an array
3. Reverse an array
4. Merge two arrays
5. Remove duplicate elements in an array

Experiment 8: Write programs to:

1. Find string length without strlen()
2. Copy and concatenate strings
3. Compare two strings
4. Reverse a string
5. Check palindrome and count vowels, digits, and special characters

Experiment 9: Write programs to:

1. Matrix addition
2. Matrix subtraction
3. Matrix multiplication
4. Matrix transpose
5. Check whether a matrix is symmetric

Experiment 10: Write programs to:

1. Demonstrate pointer arithmetic.
2. Access one-dimensional arrays using pointers.
3. Swap two numbers using call by reference.
4. Find the largest element in an array using pointers.
5. Demonstrate arrays of pointers.

Experiment 11: Write programs using:

1. malloc()
2. calloc()
3. realloc()
4. free()
5. Dynamically allocate memory for a matrix

Experiment 12: Write programs to:

1. Demonstrate structures, unions and typedef.
2. Pass structures to functions and manipulate structures using pointers.
3. Create and write to a text file
4. Read and display file contents
5. Append records to a file

Experiment 13: Write programs to:

1. Demonstrate bitwise operators (AND, OR, XOR, NOT, shift)
2. Demonstrate bit fields in a structure
3. Demonstrate enumerations (e.g., days of week, traffic light states)
4. Demonstrate command line arguments (sum of numbers passed as arguments)
5. Demonstrate macros and preprocessor directives (#define, #ifdef, conditional compilation)

Each mini project shall be implemented as a menu-driven application. The project should provide facilities for data entry, modification, deletion and report generation wherever applicable.

Min project is a group project, should be done by the group in their own time. One Project for a batch of 5 to 6 students as decided by the faculty.

Project 1: Student Record Management System. The project shall provide the following functionalities:

- Add student
- Delete student
- Search student
- Update student
- Store records in file

Project 2: Library Management System. The project shall provide the following functionalities:

- Add books
- Issue books
- Return books
- Search books

Project 3: Bank Account Management System. The project shall provide the following functionalities:

- Open account
- Deposit
- Withdraw
- Balance enquiry
- File storage

Project 4: Employee Payroll System: The project shall provide the following functionalities:

- Employee details
- Salary calculation
- DA
- HRA
- Income tax
- Payslip generation

Project 5: Hospital Management System: The project shall provide the following functionalities:

- Patient registration
- Doctor allocation
- Billing
- Medical records

Project 6: Inventory Management System: The project shall provide the following functionalities:

- Product entry
- Stock update
- Sales
- Purchase
- Report generation

Project 7: Examination Result Processing System: The project shall provide the following functionalities:

- Marks entry
- Grade calculation
- CGPA
- Rank list
- Result report

Project 8: Electricity Billing System: The project shall provide the following functionalities:

- Consumer details
- Bill calculation
- Tariff slabs
- File storage

Project 9: Address Book Management System: The project shall provide the following functionalities:

- Add contact
- Edit
- Delete
- Search
- Save to file

Project 10: Hotel Room Booking System

The project shall provide the following functionalities:

- Room booking
- Booking cancellation
- Room availability enquiry
- Customer details
- Bill generation
- File storage