# GANPAT UNIVERSITY

## FACULTY OF DIPLOMA ENGINEERING

| | | | |
|---|---|---|---|
| Programme | Diploma in Computer Engineering/ Information Technology/Electronics and Communication | | |
| Semester | I | Version | 1.0.0.0 |
| Effective from Academic Year | 2025-26 | Effective for the batch Admitted in | JULY 2025 |
| Course Code | 1ES1108 | Course Name | Programming in C |

## I.TEACHING-LEARNING AND ASSESSMENT SCHEME

| Course Type | Course Code | Learning Scheme | | | | | | Assessment Scheme | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Actual Contact Hrs./Week | | | SLH | NLH | Credits | Theory | | | | Practical | | | | Based on SL | | Total Marks |
| | | CL | TL | LL | | | | FA-TH | SA-TH | TOTAL | | FA-PR | SA-PR | TOTAL | | SLA | | |
| | | | | | | | | MAX | MAX | MAX | MIN | MAX | MAX | MAX | MIN | MAX | MIN | |
| DSC | 1ES1108 | 4 | - | 4 | 2 | 10 | 5 | 40 | 60 | 100 | 40 | 60 | 40 | 100 | 50 | 20 | 8 | 220 |

| | | | |
|---|---|---|---|
| **Abbreviation:** | **CL-** Classroom Learning | **TL**- Tutorial Learning | **LL**-Laboratory Learning |
| | **SLH**-Self Learning Hours | **NLH**-Notional Learning Hours | **SLA**-Self Learning Assessment |
| | **FA**-Formative Assessment (Term work + Mid Sem Exam Attendance) | | **SA**-Summative Assessment |

## II. PRE-REQUISITES

Familiarity with mathematical logic and problem-solving.

## III. INDUSTRY /EMPLOYER EXPECTED OUTCOMES

Capability to work in cross-platform environments, as C is widely used in Linux/Unix-based development. Strong problem-solving skills applicable in software development, automation, and systems-level programming.

## IV. COURSE LEARNING OUTCOMES

At the end of the course, students will be able to achieve the following course learning outcomes:

**CO1.** Understand the fundamentals of the C programming language, including structure, data types, operators, and I/O functions.
**CO2** Apply control structures, decision-making, and looping constructs to develop logic-based C programs.
**CO3.** Develop modular programs using functions, recursion, storage classes, and preprocessors.
**CO4.** Utilize arrays, strings, and pointers effectively for data manipulation and memory management.
**CO5.** Apply advanced data organization techniques using structures and unions to model real-world entities and develop efficient programs for data management and processing applications.

## V. THEORY LEARNING OUTCOMES AND ALIGNED COURSE CONTENT:

| Name of Unit | Theory Learning outcomes (TLO's) aligned to CO's | Learning Content mapped with Theory Learning outcomes (TLO's)&CO's | Marks | Hours |
|---|---|---|---|---|
| **Unit-1 Basics of C Programming** | **TLO 1.1** Introduction to Programming Languages<br>**TLO 1.2** History and Features of C<br>**TLO 1.3** Structure of a C Program | **1.1** Understand different types of programming languages and their applications.<br>**1.2** Describe the origin and evolution of the C programming language.<br>**1.3** Identify the structure and components of a basic C program. | **10** | **10** |

| | | | | |
|---|---|---|---|---|
| | **TLO 1.4** Algorithm, Flowchart, Compilation and Execution Process<br>**TLO 1.5** Keywords and Identifiers<br>**TLO 1.6** Variables and Data Types<br>**TLO 1.7** Constants and Literals<br>**TLO 1.8** Operators in C (Arithmetic, Relational, Logical etc.)<br>**TLO 1.9** Operator Precedence and Associativity<br>**TLO 1.10** Type Conversion and Type Casting<br>**TLO 1.11** Input/Output Functions (printf, scanf)<br>**TLO 1.12** Writing and Running First C Program | **1.4** Explain the compilation and execution flow of a C program.<br>**1.5** Recognize and use C keywords and valid identifiers correctly.<br>**1.6** Declare and initialize variables and constants appropriately.<br>**1.7** Choose appropriate data types and modifiers for given problems.<br>**1.8** Use arithmetic, relational, logical, and assignment operators in expressions.<br>**1.9** Apply operator precedence and associativity rules in complex expressions.<br>**1.10** Perform type conversions and apply explicit typecasting.<br>**1.11** Write well-documented code using appropriate comments and style guidelines.<br>**1.12** Compile and run simple C programs using an IDE or terminal. | | |
| **Unit-2 Control Flow Statements** | **TLO 2.1** Introduction to Control Flow<br>**TLO 2.2** Decision Making: if, if-else<br>**TLO 2.3** Nested if-else and else-if Ladder<br>**TLO 2.4** switch statement<br>**TLO 2.5** Introduction to Loops<br>**TLO 2.6** while Loop<br>**TLO 2.7** do-while Loop<br>**TLO 2.8** for Loop<br>**TLO 2.9** Nested Loops<br>**TLO 2.10** break and continue Statements<br>**TLO 2.11** goto Statement and Labels | **2.1** Differentiate between various control flow structures in C.<br>**2.2** Use if and if-else constructs for decision-making in code.<br>**2.3** Implement nested if-else and else-if ladder for complex conditions.<br>**2.4** Apply switch statements to replace multiple conditional expressions.<br>**2.5** Recognize the need for loops in repetitive operations.<br>**2.6** Implement while loops for entry-controlled repetition.<br>**2.7** Use do-while loops for exit-controlled repetition.<br>**2.8** Apply for loops for counter-based iteration.<br>**2.9** Use nested loops to handle multi-level iteration problems.<br>**2.10** Control loop execution using break and continue statements.<br>**2.11** Apply the goto statement in special cases while understanding its risks. | **12** | **10** |
| **Unit-3 Functions and Modular Programming** | **TLO 3.1** Introduction to Functions<br>**TLO 3.2** Syntax and Declaration of Functions | **3.1** Explain the concept and benefits of using functions in C.<br>**3.2** Declare and define functions with appropriate syntax. | **12** | **14** |

| | | | | |
|---|---|---|---|---|
| | **TLO 3.3** Function Call<br>**TLO 3.4** Return Values and void Functions<br>**TLO 3.5** Function Parameters: Call by Value and Call by reference<br>**TLO 3.6** Scope Rules: Local and Global Variables<br>**TLO 3.7** Storage Classes (auto, static, extern, register)<br>**TLO 3.8** Function Prototypes<br>**TLO 3.9** Recursion: Concept and Examples<br>**TLO 3.10** Nesting of Functions (Conceptual)<br>**TLO 3.11** Header Files and Modular Programming<br>**TLO 3.12** Preprocessor Directives (#define, #include) | **3.3** Call user-defined and standard library functions in programs.<br>**3.4** Return values from functions and handle different return types.<br>**3.5** Pass arguments to functions using call by value and call by reference.<br>**3.6** Differentiate between local and global variables in function scopes.<br>**3.7** Apply storage class specifiers to control variable life and visibility.<br>**3.8** Use function prototypes to declare functions ahead of their usage.<br>**3.9** Implement recursive functions to solve repetitive problems elegantly.<br>**3.10** Understand the concept of nested function calls (conceptual level).<br>**3.11** Break programs into modules using header files and custom libraries.<br>**3.12** Use preprocessor directives to manage code structure and conditional compilation. | | |
| **Unit-4 Arrays, Strings, and Pointers** | **TLO 4.1** Introduction to Arrays<br>**TLO 4.2** One-Dimensional Arrays<br>**TLO 4.3** Initialization of Arrays<br>**TLO 4.4** Traversing and Manipulating Arrays<br>**TLO 4.5** Two-Dimensional Arrays<br>**TLO 4.6** Multidimensional Arrays (Basics)<br>**TLO 4.7** Introduction to Strings<br>**TLO 4.8** String Handling Functions (strlen, strcpy, strcmp, etc.)<br>**TLO 4.9** Character Arrays vs Strings<br>**TLO 4.10** Introduction to Pointers<br>**TLO 4.11** Pointer Arithmetic<br>**TLO 4.12** Pointers and Arrays Relationship | **4.1** Define and explain the purpose of arrays in programming.<br>**4.2** Declare, initialize, and use one-dimensional arrays effectively.<br>**4.3** Traverse and manipulate array elements using loops.<br>**4.4** Work with initialization lists and boundary conditions in arrays.<br>**4.5** Declare and use two-dimensional arrays for matrix-style data.<br>**4.6** Describe the concept of multidimensional arrays (basic level).<br>**4.7** Define strings and understand how they are stored in C.<br>**4.8** Apply standard string handling functions like strcpy, strcmp, etc.<br>**4.9** Compare character arrays and strings and choose based on context. | **14** | **12** |

| | | | | |
|---|---|---|---|---|
| | | **4.10** Understand pointers and their declaration, initialization, and usage.<br>**4.11** Perform pointer arithmetic for accessing array and memory elements.<br>**4.12** Demonstrate the relationship between pointers and arrays in memory access. | | |
| **Unit-5 Structures, Unions** | **TLO 5.1** Introduction to Structures<br>**TLO 5.2** Defining and Using Structure Variables<br>**TLO 5.3** Nested Structures and Arrays within Structures<br>**TLO 5.4** Functions with Structures<br>**TLO 5.5** Introduction to Unions<br>**TLO 5.6** Differences between Structures and Unions<br>**TLO 5.7** typedef and enum Usage | **5.1** Define and declare structure types for grouping related variables.<br>**5.2** Access and manipulate structure members using dot operator.<br>**5.3** Create nested structures and structures containing arrays.<br>**5.4** Pass structures to functions and return structures from functions**.**<br>**5.5** Declare and use unions for memory-efficient data representation.<br>**5.6** Compare structures and unions and choose based on application needs.<br>**5.7** Use typedef and enum to create readable and maintainable code. | **12** | **14** |

| **VI. LABORATORY LEARNING OUTCOME AND ALIGNED PRACTICAL** | | | |
|---|---|---|---|
| **SR. NO** | **PRACTICAL/LABORATORY LEARNING OUTCOME(LLO)** | **PRACTICAL TITLES** | **RELEVANT COs** |
| 1 | **LLO 1.1** Understand Basic Syntax, Operators, and Control Flow. | To develop fundamental C programs using variables, operators, input/output, and decision-making constructs. | CO1 |
| 2 | **LLO 2.1** Apply the concept of Iteration, Functions, and Recursion. | To use loops and modular programming techniques to build maintainable and reusable code. | CO2 |
| 3 | **LLO 3.1** Understand Arrays, Strings, and Their Applications | To work with arrays and strings for data storage and manipulation tasks. | CO2 |
| 4 | **LLO 4.1** Understand the need of Pointers and Memory Management | To implement efficient memory and data management using pointers. | CO3 |
| 5 | **LLO 5.1** Understand the need of user-defined data Structures and importance of secondary storage with File Handling, and Advanced Features | To manage complex data using structures and file operations for persistent storage. | CO3 |

## VII. SUGGESTED MICRO PROJECT/ASSIGNMENTS/ACTIVITIES FOR SELF LEARNING/SKILL DEVELOPMENT (SELF LEARNING)

**Micro Projects (Mini Applications / Use Cases)**

- Develop a Simple Calculator (Menu-driven using switch)
  *Skills:* Operators, control flow, modular programming
- Create a Student Marks Management System
  *Skills:* Arrays, structures, file I/O
- Build a Temperature Unit Converter (Celsius ↔ Fahrenheit ↔ Kelvin)
  *Skills:* Conditional logic, user input/output
- Design a Library Book Record System using Structures and Files
  *Skills:* Structures, file handling, data processing
- Develop a Password Strength Checker
  *Skills:* Strings, loops, condition checking

**Self-Learning / Skill Building Activities**

1. Create a free account on LeetCode / hackerrank select your preferred programming language, and start with Easy problems. Solve at least 30 problems.
2. Implement User-defined Function-based Code
   *GCD/LCM, Recursion problems*
3. Structure Handling Activity
   *Employee database, Inventory list, etc.*
4. Use Online Compilers (like Replit, CodeChef IDE, HackerRank)
   *Goal:* Practice basic to advanced C problems interactively
5. Prepare a Poster/Flowchart Explaining Memory Allocation in C
   *Goal:* Visualization of stack vs heap, pointers, etc.
6. Record a Screen Video Tutorial on "Creating Your First C Program"
   *Goal:* Improve peer teaching and communication skills
7. Research and Present How C is Used in Embedded Systems
   *Goal:* Industry relevance and cross-domain application understanding
8. Create a C Program to Simulate ATM Transaction Menu
   *Goal:* Use of menus, loops, conditionals, functions
9. Explore and Use gcc Compiler with Command Line Arguments
   *Goal:* Learn real-world compilation and testing
10. Document 10 Common Errors in C Programming and Their Fixes
    *Goal:* Debugging skills and code safety
11. Develop a Personal Portfolio Console App (using C only)
    *Goal:* Strings, file storage, menu systems

## VIII. LIST OF INSTRUMENTS / EQUIPMENT / TRAINER BOARD

| | |
|---|---|
| **1** | Editor: Turbo C / Turbo C++ or Code: Blocks or Visual Studio Code |
| **2** | Compiler: GCC (GNU Compiler Collection) |
| **3** | Online Compilers & Simulators<br>Replit:https://replit.com<br>OnlineGDB: https://www.onlinegdb.com |
| **4** | Collaboration / Documentation Tools:<br>GitHub / Git: Version control and collaboration<br>Google Docs / Word: Report and assignment writing<br>Draw.io / Lucidchart: Flowcharts, memory maps, logic design |

| IX. LIST OF REFERENCE BOOKS | | | |
|---|---|---|---|
| **Sr.No** | **Title** | **Author** | **Publication** |
| 1 | Let Us C | Yashavant Kanetkar | BPB Publications |
| 2 | Programming in ANSI C | E. Balagurusamy | McGraw-Hill Education |
| 3 | The C Programming Language | Brian W. Kernighan & Dennis M. Ritchie | Prentice Hall |
| 4 | C Programming: A Modern Approach | K. N. King | W. W. Norton & Company |
| 5 | Computer Fundamentals and Programming in C | Reema Thareja | Oxford University Press |

| X. LINK OF LEARNING WEB RESOURCE | |
|---|---|
| 1 | https://www.geeksforgeeks.org/c-programming-language/ |
| 2 | https://www.tutorialspoint.com/cprogramming/ |
| 3 | YouTube Chennel: CodeWithHarry : Play List: C Language Course (Hindi) |
| 4 | https://www.w3schools.com/c/ |
| 5 | YouTube Chennel: Apna College Play List: Alpha Series (C Basics + Problem Solving) |

| XI. SUGGESTED WEIGHTAGE TO LEARNING EFFORTS & ASSESSMENT PURPOSE | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Unit** | **Unit Title** | **Aligned COs** | **Learning Hours** | **R-Level** | **U-Level** | **A-Level** | **Total Marks** |
| 1 | Basics of C Programming | CO1 | 10 | 3 | 3 | 4 | 10 |
| 2 | Control Flow Statements | CO2 | 10 | 3 | 4 | 5 | 12 |
| 3 | Functions and Modular Programming | CO3 | 14 | 3 | 5 | 4 | 12 |
| 4 | Arrays, Strings, and Pointers | CO4 | 12 | 4 | 6 | 4 | 14 |
| 5 | Structures, Unions, File Handling, and Advanced Topics | CO5 | 14 | 4 | 5 | 3 | 12 |
| | **Grand Total** | | **60** | **17** | **23** | **20** | **60** |

| XIII. COs AND POs AND PSOs MAPPING | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Course Outcome (Cos)** | **Programme Outcomes (POs)** | | | | | | | **Programme Specific Outcomes (PSOs)** | | |
| | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PSO1 | PSO2 | PSO3 |
| CO1 | 3 | 2 | 1 | 2 | 0 | 1 | 0 | 3 | 1 | 1 |
| CO2 | 3 | 3 | 2 | 2 | 0 | 1 | 0 | 3 | 1 | 2 |
| CO3 | 3 | 3 | 3 | 3 | 1 | 2 | 1 | 3 | 1 | 2 |
| CO4 | 3 | 3 | 2 | 3 | 1 | 2 | 1 | 3 | 2 | 2 |
| CO5 | 3 | 3 | 3 | 3 | 2 | 2 | 1 | 3 | 2 | 3 |

**Legends: -** *3-High; 2-Moderate/Medium; 1-Slight/Low; 0-None*