Gyanmanjari Innovative University

Syllabus
Gyanmanjari College of Computer Science
Semester-I(BSCIT)

Subject: C Programming for Effective Problem Solving - BST2XX11301

Type of course: Core Course

Prerequisite: Basic computer knowledge

Rationale:

This course introduces students to problem-solving using C programming with a strong practical approach. It focuses on developing logical thinking, algorithm design, and real-world problem-solving using structured programming concepts. Through continuous hands-on activities, students will learn to write efficient programs, debug errors, and implement solutions applicable in software development and technical domains.

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-Class Room Instructions; T – Tutorial; P - Practical; C – Credit; SEE - Semester End Evaluation; CCE-Continuous and Comprehensive Evaluation;

Course Content:

Sr. No	Course content	Hrs	% Weightage
1	**Fundamentals of Programming & C Basics** • Introduction to Programming • Problem Solving Approach • Algorithms • Flowcharts • Structure of C Program • Constants, Variables & Data Types • Input and Output Operations • Operators in C **Practical:** 1. Prepare algorithm and flowchart for a simple real-life problem. 2. Prepare algorithm and flowchart for ATM withdrawal system. 3. Prepare algorithm and flowchart for student result calculation.	18	20%

4. Install and configure C compiler using GCC / Code Blocks / VS Code.
5. Write a C program Display Hello World using printf().
6. Write a C program to display formatted output using printf(). including newline (\n). tab space (\t). and structured display of personal information.
7. Write a C program to demonstrate the basic structure of a C program.
8. Write a C program to declare and initialize different types of variables.
9. Write a C program to use constants and symbolic constants.
10. Write a C program to accept and display user input using printf() and scanf().
11. Write a C program to perform addition. subtraction. multiplication. division of two numbers.
12. Write a C program to swap two numbers using a third variable and without using a third variable.
13. Write a C program to calculate simple interest.
14. Write a C program to calculate the area of a circle.
15. Write a C program to calculate the area of a rectangle.
16. Write a C program to calculate the area of a triangle.
17. Write a C program to perform temperature conversion from Celsius to Fahrenheit and vice versa.
18. Write a C program to demonstrate arithmetic operators.
19. Write a C program to demonstrate relational operators.
20. Write a C program to demonstrate logical operators.
21. Write a C program to evaluate expressions using operators.

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Program Implementation Task** Students will develop and execute C programs based on problem-solving concepts, input/output operations. and operators to generate correct output.	**20**	
2	**AlgoFlow Utility** Students will design algorithms and flowcharts in draw.io. for real-life problems such as ATM withdrawal systems and student result calculation.		5
3	**Technical Debate** The faculty will announce the debate topic and explain the rules, evaluation criteria, and time limits. Students will be divided into two teams or assigned individual positions: For and Against the given technical topic. Each team will research the topic and prepare technical arguments. supporting facts. examples. and		5

	evidence from reliable sources and participate in debate.				
	Total	**20**	**10**		

Examination Style:
Program Implementation Task (20 Marks):
Each student will be assigned 2–3 unsolved problem statements by the faculty covering: Basic calculations (I/O based), Operator-based logic, Simple real-life problem (e.g., interest, area, conversion) .Students are required to: Analyze the problem statement .Write a complete C program. Compile and execute the program, Display correct output

AlgoFlowUtility(5 Marks):
Students will design algorithms and flowcharts in draw.io for given real-life problems assigned by faculty. The completed work will then be compiled into a single PDF document and submitted on the GMIU web portal.

Technical Debate (5 Marks):
The Technical Debate is an interactive learning activity designed to enhance students' technical knowledge, critical thinking, analytical reasoning, communication, and presentation skills. Students participate in structured debates on technical topics. Participants present evidence-based arguments, defend their viewpoints, and respond to opposing opinions in a professional and respectful manner.

2	**Programming Constructs in C** • Introduction to decision making in C • Decision making statements: if, if-else, and nested if-else, if–else if ladder • switch statement and its usage • Introduction to looping concepts • for loop • while loop • do-while loop • Loop control statements: break, continue and goto. **Array and String** • Array :One dimensional array, multi-dimensional arrays. • String, gets() and puts() **Functions** • Concept of modular programming • Type of Function • Declaration, Calling, and Defining a function. • Passing Array and string as function argument • Recursion • Built-in Function: string, maths, input output function etc **Practical:** 22.Write a C program to check whether a number is even or odd. 23.Write a C program to find the largest among two and three numbers.	18	20%

24. Write a C program to check whether a number is positive, negative, or zero.
25. Write a C program to calculate and display the grade of a student based on marks using an if–else if ladder.
26. Write a C program to implement menu-driven operations using switch statements.
27. Develop a C program to find the area of different shapes using switch-case.
28. Develop a C program to implement a simple banking menu using switch-case.
29. Write a C program to print numbers from 1 to N using a loop.
30. Write a C program to calculate the sum of first N natural numbers.
31. Write a C program to generate the multiplication table of a number.
32. Write a C program to calculate the factorial of a number using a loop.
33. Write a C program to generate the Fibonacci series.
34. Write a C program to reverse a number and check palindromes.
35. Write a C program to check whether a number is an Armstrong number.
36. Write a C program to print patterns (star and number patterns) using nested loops.

(i) * * * * * * * * * * * * * * *	(ii) * * * * * * * * * * * * * * *	(iii) 1 1 2 1 2 3 1 2 3 4 1 2 3 4 5
(iv) A A B A B C A B C D A B C D E	(v) A B C D E F G H I J K L M N	(vi) * * * * * * * * * * * * * * *
(vii) 5 4 3 2 1 5 4 3 2 5 4 3 5 4 5	(viii) 1 2 3 4 5 6 7 8 9 10	(ix) 1 1 1 1 2 1 1 3 3 1 1 4 6 4 1
(x) 1 2 1 2 3 2 1 2 3 4 3 2 1 2 3 4	(xi) 1 0 1 1 0 1 0 1 0 1	(xii) A A B A A B C B A A B C D C B A
(xiii) *	(xiv) *	(xv) * * * * * * * * * * * * *
(xvi) * * * * * * * * * * * * * * * *	(xvii) * * * * * * * * * * * *	(xviii) * * * * * * * * * * * * * * * *

37. Write a C program to demonstrate the use of break, continue and goto statements.
38. Write a C program that uses a single index array to store and process elements such as finding sum, average, maximum and minimum.
39. Write a C program that uses a matrix to perform operations like matrix addition, multiplication.
40. Write a C program used to find the location of a specific element of an array.
41. Write a C program to calculate the sum of each row and each column separately.
42. Write a program to reverse a string and check if it is a palindrome.
43. Develop a C program to input a string from the user and display its length using the strlen() function.
44. Write a C program that uses the strcpy() function to copy one string to another and display the result.
45. Implement a C program that concatenates two strings using the strcat() function and prints the resulting string.
46. Develop a C program to add two numbers using a function.
47. Create a C program to show the difference between call by value and call by reference using a function that swaps two variables.
48. Implement a C program that uses recursion to calculate the factorial of a number.
49. Demonstrate the use of library functions like pow() and sqrt().

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Program Execution & Logic Task** Students will develop and execute C programs based on decision making and looping, array, string and functions concepts to generate correct output.	20	
2	**Code Annotation Activity** students will be given an unsolved program and asked to develop and write meaningful comments for each line of code. Instead of just running or copying the program, students must explain what the code does (function) and why it is written that way (logic and purpose) and upload a PDF document on GMIU Web portal.		5
3	**Interview Lab - C Programming** Students will be assigned an interviewer (faculty member or peer interviewer). The interviewer will ask questions covering C programming concepts like decision making ,looping, array, string and functions. Students will be evaluated based on technical knowledge, coding ability, communication, confidence, analytical thinking, and overall interview		5

		performance		
		Total	**20**	**10**

Examination Style:
Program Execution & Logic Task (20 Marks):
Each student will be assigned 2–3 unsolved problem statements by the faculty covering: Decision making statements, switch statements, looping concepts, loop control statements. Students are required to: Analyze the problem statement .Write a complete C program. Compile and execute the program, Display correct output.
Code Annotation Activity(5 Marks):
faculty will assign unsolved practical program definitions and ask to develop and write meaningful comments for each line of code. Instead of just running or copying the program, students must explain what the code does (function) and why it is written that way (logic and purpose) and upload a PDF document on GMIU Web portal.

Interview Lab C Programming (5 Marks):
Interview Lab – C Programming is a simulated technical interview activity designed to prepare students for campus placements and technical recruitment processes. The activity provides a realistic interview experience where students answer questions related to C programming concepts, problem-solving, coding, debugging, and programming logic. It helps students improve their technical knowledge, communication skills, confidence, and interview etiquette while receiving constructive feedback for improvement.

3	**Pointers and Memory Management** • Introduction and Features of Pointers • Declaration of Pointer • Initialization of Pointer • Void Pointers • Array of Pointers • Pointers to Pointers • Pointer to Functions • Dynamic Memory Allocation • Allocating a Block of Memory: Malloc • Allocating Multiple Blocks of Memory: Calloc • Releasing the used Space: Free • Altering the size of a Block: Realloc **Practical:** 50. Develop a C program to declare a pointer and initialize it to point to a variable. 51. Develop a C program that demonstrates the use of pointers by swapping two integers using a pointer. 52. Develop a C Program to demonstrate the use of Array of pointers. 53. Develop a C Program to demonstrates the use of pointers to pointers. 54. Develop a C program to use pointers to Functions. 55. Develop a C Program to demonstrate use of malloc, calloc.	18	20%

56. Develop a C Program to find the largest and second largest element using Dynamic Memory Allocation.
57. Develop a C program to calculate sum and average using dynamically allocated arrays.
58. Develop a C program to dynamically allocate memory and sort elements in ascending order.
59. Develop a C program to dynamically allocate memory and sort elements in descending order.
60. Develop a C program to demonstrate use of free, realloc.
61. Develop a program to dynamically allocate memory and search an element in an array.
62. Develop a program to dynamically allocate memory and reverse the elements of an array.
63. Develop a program to dynamically allocate memory and merge two arrays.
64. Develop a program to dynamically allocate memory and remove duplicate elements.

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Code Efficiency Enhancement** Students will develop C programs by focusing on code optimization, performance improvement, and efficient memory usage to ensure faster execution and better resource management.	20	
2	**Code Clinic Activity** Students will be provided with two programs, each carrying 5 marks, containing logical and/or syntactical errors. The programs will be based on unsolved, practical, or competitive coding problems and will be assigned by the faculty. Students are required to identify and resolve the errors, execute the corrected code, and verify the output. After successful execution, students must prepare a PDF document that includes the number of errors with description, corrected program code along with the output/results and the final PDF must be uploaded on the GMIU web portal.		5
3	**Learning Checkpoint** Faculty will upload the MCQ quiz on the GMIU Web Portal. Students must log in to the portal and attempt the quiz within the scheduled date and time. The quiz will be evaluated based on the correctness of the responses submitted through the GMIU Web Portal.		5
	Total	20	10

	Examination Style: **Code Efficiency Enhancement(20 Marks) :** Each student will be assigned two unsolved problem statements of dynamic memory allocation, each carrying 10 marks. Students are required to develop C programs for the given problems by focusing on code optimization, performance improvement, and efficient memory usage to ensure faster execution and effective resource management. This helps students enhance practical coding skills by applying optimization techniques for improving program performance and resource management. **Code Clinic Activity (5 Marks):** Students will be provided with two unsolved programs, each carrying 5 marks, containing logical and/or syntactical errors. The programs will be based on unsolved, practical, or competitive coding problems and will be assigned by the faculty. Students are required to identify and resolve the errors, execute the corrected code, and verify the output. After successful execution, students must prepare a PDF document that includes the number of errors with description, corrected program code along with the output/results and the final PDF must be uploaded on the GMIU web portal. **Learning Checkpoint (5 Marks):** The Online MCQ Quiz is a digital assessment activity conducted through the GMIU Web Portal to evaluate students' understanding of course concepts and technical knowledge. Faculty members will create and upload the quiz on the GMIU Web Portal, and students are required to attempt the quiz within the specified time and submission deadline.		
4	**Structure and Union** **Structure:** • Introduction, Features • Declaration and Initialization and accessing • Array of structures • typedef • Enumerated data type **Union:** • Introduction • Declaration and Initialization and accessing **Practical:** 65. Develop a C program that demonstrates the use of structures to store and display information about students. 66. Create a C program to read data into a structure from the user and display it using a function 67. Write a program in C to calculate the total and average marks of a student using structures. 68. Develop a C program using Array of structures. 69. Write a program in C to store and print information of multiple employees using structures.	18	20%

70. Develop a C program to store employee details using structure and calculate total salary.
71. Develop a C program to store item details using structure and calculate bill amount.
72. Develop a C program to store employee data using structure and calculate bonus.
73. Write a C Program to demonstrate usage of enum and typedef.
74. Develop a C program to use enum with typedef.
75. Develop a C program to use enum for menu-driven programs.
76. Develop a C Program to demonstrate the use of union.
77. Develop a program to show that only one member of a union can hold value at a time.
78. Develop a program to compare structure and union memory usage.

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Interactive Assessment** An Interactive Assessment is a live, two-way evaluation where students actively respond, explain, and demonstrate their understanding of C programming concepts through answering questions and participating in discussions.	20	
2	**Tech Skills Workshop** Students are required to participate in technical workshops recognized by organizations, industries, educational institutions, or online learning platforms to enhance their technical knowledge and practical skills and upload certificate on the GMIU web portal.		10
	Total	20	10

Examination Style:

Interactive Assessment (20 Marks):

An Interactive Assessment is a live, two-way evaluation where students actively respond, explain, and demonstrate their understanding of C programming concepts through answering questions and participating in discussions.

This faculty will ask questions from Unit 4. Students are required to answer the questions interactively to demonstrate their understanding of C programming concepts. This helps assess clarity of thought, communication skills, and conceptual depth.

Tech Skills Workshop(10 Marks):

Participation in technical workshops helps students develop industry-relevant skills, stay updated with emerging technologies, strengthen their technical competency, and improve their readiness for internships, placements, and professional careers.

File Handling

- Definition of file
- Streams and File Types
- Opening ,Reading , closing of File (Text Mode)
- Reading From and writing into Files (Text mode)

Practical:

79. Develop a C Program using file operations. Opening ,Reading(using fgetc().fgets().fscanf().fread()) , closing of File (Text Mode)
80. Develop a C program to append multiple lines to an existing file.
81. Write a C program to write data into a text file using fprintf(). fwrite().
82. Write a C Program to count the number of characters.words and lines of file.
83. Develop a C program to store and retrieve structure data from a file.
84. Develop a C program to store student records in a file and display them.
85. Develop a C program to write multiple lines into a file.
86. Develop a C Program using file operations. Reading From and writing into Files
87. Develop a C program to compare the contents of two files.
88. Develop a C program to reverse the contents of a file.

5 | 18 | 20%

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Competitive Coding Evaluation** Two 10 marks questions will be based on a real-time problem-solving scenario aligned with competitive coding practices, not directly from the syllabus. The objective is to test students' logical thinking, problem-solving ability, and coding skills in unfamiliar yet practical contexts. The question will be provided by the faculty.	**20**	
2	**DataCraft in C** Faculty will assign unique definitions. Each Student will develop Mini C applications using structures and file handling techniques. The final project must be compiled and submitted as a report on the GMIU Web Portal.		**10**
	Total	**20**	**10**

Examination Style:

Competitive Coding Evaluation(20 Marks):

Two 10 marks unsolved questions will be given based on a real-time problem-solving scenario aligned with competitive coding practices, not directly from the syllabus. The objective is to test students' logical

	thinking, problem-solving ability, and coding skills in unfamiliar yet practical contexts. The question will be provided by the faculty. **DataCraft in C(10 Marks):** Faculty will assign unique definitions. Each Student will develop C applications using structures and file handling techniques. The final project must be compiled and submitted as a report on the GMIU Web Portal.		

Suggested Specification :

Distribution of Theory Marks (**Revised Bloom's Taxonomy**)						
Level	Remembrance (R)	Understanding (U)	Application (A)	Analyze (N)	Evaluate (E)	Create (C)
Weightage	10%	15%	30%	15%	10%	20%

Note: This specification table shall be treated as a general guideline for students and teachers. The actual distribution of marks in the question paper may vary slightly from above table.

Course Outcome:

After learning the course the students should be able to:	
CO1	Understand the basic concepts of programming and apply a systematic problem-solving approach to develop solutions.
CO2	Analyze and optimize program flow using control statements in C.
CO3	Demonstrate the use of pointers for dynamic memory access and efficient programming.
CO4	Implement structures, unions, and arrays of structures in C programming to efficiently manage and organize multiple records.
CO5	Apply file handling for data management.

Instructional Method:

The course delivery method will depend upon the requirement of content and need of students.

A minimum of 20–30% of the course content shall be suggested to deliver through Flipped Learning, where students study digital learning resources before the classroom session. Classroom time shall be utilized for discussion, problem-solving, practical applications, and higher-order learning activities.

Students shall be encouraged to use AI-powered learning platforms, simulation software, virtual laboratories, digital libraries, research databases, online certification courses, NPTEL, SWAYAM, MOOC's, and other technology-enabled learning resources to strengthen conceptual understanding and practical competence.

Continuous assessment shall be conducted through Active Learning Assignments (ALAs), quizzes, skill demonstrations, presentations, practical exercises, projects, case studies, innovation activities, peer assessment, reflective learning, and industry-oriented tasks instead of relying primarily on conventional written assignments.

Reference Books:

[1] The The C Programming Language — Brian W. Kernighan & Dennis M. Ritchie — 2nd Edition — Pearson Education / Prentice Hall
[2] C How to Program — Paul Deitel & Harvey Deitel — 9th Edition — Pearson Education.
[3] Programming in ANSI C — E. Balagurusamy — 8th Edition — McGraw Hill Education
[4] C Programming: A Modern Approach — K. N. King — 2nd Edition — W. W. Norton & Company.
[5] Let Us C — Yashavant Kanetkar — 18th Edition — BPB Publications

Suggested Assessment Guidelines:

Module 1: Fundamentals of Programming & C Basics

Program Implementation Task (20 Marks):		
Criteria	**Description**	**Marks**
Correct use of C syntax, variables, data types, operators	Proper use of variables, data types, operators, input/output statements and C syntax.	5
Code formatting, indentation and comments	Proper indentation, readability and meaningful comments in the program.	5
Logic	Ability to develop proper step-by-step logic for solving the problem.	5
Program execution and correct output	Successful compilation, execution and generation of correct output.	5
Total		**20**

Module 2: Programming Constructs in C

Program Execution & Logic Task:(20 Marks):		
Criteria	**Description**	**Marks**
Understanding of problem statement	Ability to understand the given problem and identify required input, process and output.	5
Logic	Ability to develop proper step-by-step logic for solving the problem.	5
Implementation using C Concepts	Proper use of decision making and looping, array, string and functions.	5
Program execution and correct output	Successful compilation, execution and generation of correct output.	5
Total		**20**

Module 3: Pointers and Memory Management

Code Efficiency Enhancement(20 Marks)		
Criteria	**Description**	**Marks**
Logic Optimization	Ability to improve program logic for faster and efficient execution.	5
Program Performance & Execution	Program executes correctly with improved efficiency and optimized output generation.	5
Memory Usage & Resource Management	Efficient utilization of memory and avoidance of unnecessary resource consumption.	5
Viva / Explanation of Optimization	Ability to explain optimization techniques and efficiency improvements implemented in the program.	5
Total		**20**

Module 4: Structure and Union

Interactive Assessment (20 Marks):		
Criteria	**Description**	**Marks**
Conceptual Understanding	Ability to understand and explain C programming concepts from Units 1 to 4.	5
Accuracy of Answers	Ability to provide correct and relevant answers to faculty questions.	5
Problem-Solving Approach	Ability to apply programming concepts logically while answering scenario-based questions.	5
Communication Skills	Clarity, confidence and effectiveness in explaining concepts and program logic.	5
Total		**20**

Module 5: File Handling

ompetitive Coding Evaluation(20 Marks):		
Criteria	**Description**	**Marks**
Handling Unfamiliar Problems	Ability to adapt programming knowledge to solve new and practical coding challenges.	5
Logical Thinking & Approach	Ability to develop proper logic and an efficient problem-solving approach.	5
Output Accuracy	The program generates correct output for the given test cases and scenarios.	5
Viva / Explanation	Clarity, confidence and effectiveness in explaining concepts and Ability to explain the logic, approach and working of the implemented solution.	5
Total		**20**