Gyanmanjari Innovative University

Course Syllabus
Gyanmanjari College Of Computer Application
Semester-1 (BCA)

Subject: C Programming For Effective Problem Solving-BCAXX11301

Type of course: Major (Core)

Prerequisite: Programming fundamental, logic and problem-solving skill, mathematical logic.

Rationale:

The core component which drives these tasks is a piece of code for the machine, known as a program. It is essential for the students to learn basic concepts and methodology to develop computer programs. This first and introductory level Computer Programming Course is intended to develop logical thinking skills and programs using a popular structured programming language 'C'. The programming skills thus acquired can be used for developing programs for the scientific, research and business purposes.

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;LWA - Lab Work Assessment; V – Viva voce; CCE-Continuous and Comprehensive Evaluation; ALA- Active Learning Activities.

CourseContent:

Sr. No	Course Content	Hrs.	% Weightage
1	**Theory Topics** - Overview of C Programming Language, History and features of C, Structure of a C program, Writing and executing the first C program. C Tokens, Keywords and Identifiers, Constants, Variables, Data Types and type modifiers, Types of Operators, Input and output functions: printf(), scanf().	T:05 P:10	20%

Practical:

•Write a simple C program that displays "Hello, World!" and execute it.

•Create a C program that demonstrates the use of keywords, identifiers, constants, and variables.

•Write a C program to perform basic arithmetic operations using different data types and type modifiers.

•Develop a C program that uses input and output functions (printf(), scanf()) to take user input and display the result.

•Write a C program to experiment with different data types and type modifiers, showing how each affects variable size and memory usage.

Examination Style:

Sr. No.	Evolution Methods	SEE	CCE
1	Debugging Task.	05	00
2	Code Troubleshooting	00	05
3	Code Optimization.	10	00

•Debugging Task (05 Marks):

1. In this task, students will be a programs carrying 05 marks containing logical or syntactical errors. The question will be unsolved, practical, or competitive coding type, and will be provided by the faculty. This task is designed to assess students' debugging and problem-solving abilities.

•Code Troubleshooting (05 Marks):

1. In this task, students will be a programs carrying 05 marks containing logical or syntactical errors. The question will be unsolved, practical, or competitive coding type, and will be provided by the faculty. This task is designed to assess students' debugging and problem-solving abilities.

•Code Optimization (10 Marks):

1. In this task, students will be provided with two programs, each carrying 05 marks. Students are expected to improve the logic, reduce time and/or space complexity, and minimize the number of lines of code without affecting the functionality. The questions will be practical in nature and designed by the faculty to assess students' ability to write efficient and concise code.

2	**Theory Topics**: Conditional Statements - if, if-else, nested if, switch-case. Loops - for, while, do-while, Use of break and continue. Logical and Relational Operators. Arrays - Single-dimensional arrays, multi-dimensional arrays, Strings -	T:05 P:10	20%

Introduction to strings, String handling functions (strlen, strcpy, strcat, etc).

Practical:

•Write a C program using if, if-else, and nested if statements to check whether a number is positive, negative, or zero.
•Create a C program that uses a switch-case statement to determine the day of the week based on user input (1 to 7).
•Develop a C program that uses a for loop to print the first 10 natural numbers.
•Create a C program that uses break and continue in a loop to print all numbers from 1 to 20, skipping multiples of 5.
•Write a C program to declare and initialize a single-dimensional array, and then print its elements using a loop.
•Write a program to reverse a string and check if it is a palindrome.
•Develop a C program to input a string from the user and display its length using the strlen() function.
•Write a C program that uses the strcpy() function to copy one string to another and display the result.
•Implement a C program that concatenates two strings using the strcat() function and prints the resulting string.

Examination Style:

Sr. No.	Evolution Methods	SEE	CCE
1	Bug Fixing Activity.	20	00
2	Bug Hunt Challenge.	00	10
3	Practical Task.	20	00

•Bug Fixing Activity (20 Marks):

1. In this task, students will be given two programs, each carrying 10 marks, containing logical or syntactical errors. The questions will be of unsolved, practical, or competitive coding type and will be provided by the faculty. The task aims to assess the students' debugging skills, logical reasoning, and ability to correct code to produce the desired output.
2. Each debugging question carries 10 marks to help the examiner assess code quality and correctness.

•Practical Task (20 Marks):

1. The practical task will consist of four questions carrying 05, marks total20 marks. Questions will be based on practical implementation from the unit. Students are required to write, execute, and submit the program with correct logic and output. All questions will be framed and provided by the

<table>
<tr><td></td><td>faculty to assess students' practical understanding and coding skills.
2. The 05 marks question is evaluated based on code accuracy and correctness.</td><td></td><td></td></tr>
<tr><td>3</td><td>Theory Topics: Introduction to Functions - Syntax and declaration, call by value vs call by reference, Standard Library Functions, Math functions (sqrt, pow, etc.), Recursion.

Practical:
•Write a function to find the factorial of a number using recursion.
•Implement a program to calculate the greatest common divisor (GCD) of two numbers.
•Demonstrate the use of library functions like pow() and sqrt().
•Create a C program to show the difference between call by value and call by reference using a function that swaps two variables.
•Implement a C program that uses recursion to calculate the factorial of a number.

Examination Style:
<table>
<tr><th>Sr. No.</th><th>Evolution Methods</th><th>SEE</th><th>CCE</th></tr>
<tr><td>1</td><td>Active Learning Activities (ALA).</td><td>00</td><td>10</td></tr>
<tr><td>2</td><td>Code Optimization.</td><td>15</td><td>00</td></tr>
<tr><td>3</td><td>Difference and Comparison Questions.</td><td>05</td><td>00</td></tr>
</table>
ALA: Recursion Manifesto: Ethical and Efficient Coding(10 Marks):
Students will work collaboratively to create a comprehensive Recursion Policy for a hypothetical software development organization. The policy should cover best practices, acceptable use, efficiency considerations, debugging strategies, and optimization techniques in recursion for C programming. Students must submit the prepared document on the GMIU web portal.

•Code Optimization (15 Marks):
1. This section consists of three questions, each carrying 05 marks, focusing on code optimization. Students are expected to improve the logic, reduce time and/or space complexity, and minimize the number of lines of code without affecting functionality. The questions will be framed by the faculty and aim to assess the student's ability to write clean, efficient, and optimized code.
2. Each question carries 05 marks for evaluating code optimization and efficiency.</td><td>T:05
P:10</td><td>20%</td></tr>
</table>

<table>
<tr><td></td><td>

•Difference and Comparison Questions (05 Marks):

1. The difference and comparison question carries 05 marks and will be asked from the units covered in the syllabus. Students will be required to clearly differentiate or compare two related concepts, terms, or techniques. This task is designed to test conceptual clarity and the ability to highlight precise distinctions.

</td><td></td><td></td></tr>
<tr><td>4</td><td>

Theory Topics: Pointers and Memory Management -Introduction to Pointers: Declaration and initialization, Pointer arithmetic. Dynamic Memory Allocation - malloc(), calloc(), free().

Practical:

•Write a program to swap two numbers using pointers.
•Write a C program to declare and initialize a pointer, then use it to access and modify the value of a variable.
•Develop a C program that dynamically allocates memory using malloc() for an array, then frees the memory using free().
•Write a C program that uses calloc() to allocate memory for an array of integers and initialize the memory to zero.
•Implement a C program that demonstrates the use of dynamic memory allocation and deallocation by creating and manipulating a dynamically allocated 2D array.

Examination Style:

<table>
<tr><th>Sr. No.</th><th>Evolution Methods</th><th>SEE</th><th>CCE</th></tr>
<tr><td>1</td><td>Active Learning Activities (ALA).</td><td>00</td><td>10</td></tr>
<tr><td>2</td><td>Code Optimization.</td><td>10</td><td>00</td></tr>
<tr><td>3</td><td>Viva/Oral Examination.</td><td>05</td><td>00</td></tr>
</table>

ALA: The Memory Map: Understanding Pointers in C(10 Marks):

In this activity, students will explore different aspects of pointers and memory management, learn about various types of memory allocation techniques, and understand how to handle dynamic memory efficiently in C programming. They will practice implementing pointers, memory allocation, and debugging strategies to optimize memory usage. The final report will be submitted on the GMIU Web Portal.

•Code Optimization (10 Marks):

1. In this task, students will be provided with two programs, each carrying 05 marks. Students are expected to improve the logic, reduce time and/or space complexity, and minimize the number of lines of code without affecting the functionality.

</td><td>T:05
P:10</td><td>20%</td></tr>
</table>

	The questions will be practical in nature and designed by the faculty to assess students' ability to write efficient and concise code. 2. Each question carries 05 marks for evaluating code optimization. **•Viva/Oral Examination Style (05 Marks):** 1. The viva/oral test carries 05 marks and will be conducted individually to evaluate the student's understanding of practical concepts. Questions will be asked from the units covered in the syllabus, including logic explanation, code reasoning, and real-time application relevance. This test helps assess clarity of thought, communication skills, and conceptual depth.		
5	**Theory Topics:** Structures and File Handling - Introduction to Structures: Definition, declaration, and usage. File Handling - File operations: fopen(), fclose(), fprintf(), fscanf(), fread(), fwrite(),Reading from and writing to files. **Practical:** •Write a C program to define a structure to store student information (name, age, and grade) and display it using a pointer. •Create a C program to read data into a structure from the user and display it using a function. •Develop a C program that uses fopen() to open a file in write mode, fprintf() to write data to the file, and fclose() to close the file. •Write a C program to read data from a file using fscanf() and display the contents on the screen. •Implement a C program that uses fread() and fwrite() for binary file handling by reading and writing data from a binary file. **Examination Style:** **Sr. No. | Evolution Methods | SEE | CCE** 1 | Active Learning Activities (ALA). | 00 | 10 2 | Debugging Task. | 00 | 05 3 | Real-Time Examples. | 10 | 00 **ALA: StructBytes: File Handling for Real-World Applications(10 Marks):** In this practical activity, students will work with real-world data to develop C applications using structures and file handling. They will focus on data storage, retrieval, and manipulation through structured records and file operations. Students will implement programs that read, write, and process structured data efficiently using C. The final project will be submitted on the GMIU Web Portal.	T:05 P:10	20%

	•Debugging Task (05 Marks): 1. In this task, students will be given a single program containing logical or syntactical errors, carrying 05 marks. The question will be unsolved, practical, or competitive coding type, and will be provided by the faculty. This task is designed to assess students' debugging and problem-solving abilities. 2. Each debugging question carries 05 marks to help the examiner assess code quality and correctness. **•Real-Time Examples (10 Marks):** 1. One 10 marks question 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. 2. This question carries 10 marks for evaluating students' approach to competitive-style coding.		

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	Develop logical thinking and problem-solving skills using C programming.
CO2	Analyze and optimize program flow using control statements in C.
CO3	Utilize standard library functions to simplify programming tasks.
CO4	Demonstrate the use of pointers for dynamic memory access and efficient programming.
CO5	Develop programs to efficiently store, retrieve, and process data using file handling techniques.

Instructional Method:

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

From the content 10% topics are suggested for flipped mode instruction.

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

The internal evaluation will be done on the basis of Active Learning Assignment.

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

Reference Books:

[1] Programming inAnsi-C, by BalagurusamyTata McGraw-Hill Publishing.
[2] Programming in C, by Kochan Stephen GPublisher:Pearson.
[3]Programming in C, by Kamthane AshokPublisher:Pearson.
[4]Programming in C, by Gotfried BoyronPublisher:Pearson.