Course Code:	**Course Name:** Computer Programming using C	
Credits: 2-0-2	**Contact hours per week:** 2 hours Theory & 2 hours lab per week	
Batch: 2024, 1st Sem **Academic Year:** 2024-25	**Semester Duration:** 5th Aug 2024 to 6th Dec	
Instructor-in-charge: Dr Yogesh Gupta Email - yogesh.gupta@bmu.edu.in Phone - 9068039711 Office – 28, Fourth Floor, E2 Building Dr Kiran Sharma Email - kiran.sharma@bmu.edu.in Phone - 8130030929 Office – 85, Fourth Floor, E2 Building Ms. Monika Mokan Email - monika.mokan@bmu.edu.in Phone -9694934358 Office – NB204, Second Floor, E2 Building Dr. Atul Mishra Email - atul.mishra@bmu.edu.in Phone – 9971263488 Office - 86, Fourth Floor, E2 Building	**Course Coordinator –** Dr Atul Mishra Email - atul.mishra@bmu.edu.in Phone – 9971263488 Office - 86, Fourth Floor, E2 Building	BML MUNJAL UNIVERSITY FROM HERE TO THE WORLD

Aim of the course: The objective of the course is to develop student's programming skills by teaching foundational C programming concepts, introducing the principles of programming in C, and advancing their ability to create both simple and complex programs.

Course Overview and Context: Computer Programming using C is a core course in CSE undergraduate programs. This subject aims to introduce students to the fundamentals of C Programming. Upon successful completion of this subject, the students should be able to able to design basic programming solutions including statements, control structures, and methods.

Course Outcomes: At the end of the course, students should be able to

CO1: Understand the basics of programming and different programming approaches.

CO2: Apply fundamental C programming concepts, including data types, control structures, functions, and arrays.

CO3: Analyze programming techniques to design and implement solutions for real-world problems using C.

Course Competencies: (Course Outcomes further elaborated) and Instruction schedule:

Competency	Sessions	CO
Introduction to C Programming, Flowchart, Pseudocode	3	CO1
Basic Syntax and Structure	2	CO1
Data Types and Variables	2	CO1
Operators and Expressions	3	CO1
Control Structures: If-Else, Switch	3	CO2
Loops: For, While, Do-While	3	CO2
Functions: Definition and Declaration, applications	4	CO2, CO3
Arrays: One-dimensional and multi-dimensional	4	CO2
Pointers: Basics and Pointer Arithmetic	3	CO2, CO3
Strings and Character Arrays	3	CO2, CO3
Structures and Unions	4	CO2, CO3

Total Hours for Course: 32

CO-PO Mapping

PO and PSO →	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO10	PO11	PO12	PSO 1	PSO 2	PSO 3	PSO 4
↓ CO Mapping Course Mapping →																
CO1		2	2					2	1	2		2	2			
CO2	3	3	2					2	1	2		2	3			
CO3			3				3	2	1	2		2	3			

1- Low Correlation; 2- Moderate Correlation; 3- Substantial Correlation

Note: For details about PO, PSO, and CO-PO mapping, refer to the OBE document for B. Tech. – CSE

Week Wise Topics

Week 1-2: Introduction to C Programming - history, features, and applications of the language. Basic syntax and structure, writing simple programs, understanding the compilation process, and Integrated Development Environments (IDEs) for C programming.

Week 3-4: Data types and variables - different data types, variable declaration, and initialization. , the importance of scope, lifetime of variables, and constant variables. Operators and expressions - arithmetic, relational, logical, bitwise, and assignment operators, and explain operator precedence and associativity.

Week 5-6: Control structures - if-else statements and switch-case constructs, control structures for decision-making in programs. Loops- while, and do-while loops, and their use in iterative tasks.

Week 7-8: Functions - definition, declaration, and function calls, importance of functions in modular programming, parameter passing (by value and by reference), return types, and recursion. Practical examples and exercises to solidify these concepts.

Week 9: Mid Semester Exam Week

Week 10-11: Arrays, starting with one-dimensional arrays and progressing to multi-dimensional arrays. Explain array declaration, initialization, and access. Discuss common operations on arrays, such as traversal, insertion, deletion, and searching.

Week 12-13: Introduce pointers, starting with the basics of pointer variables, pointer arithmetic, and the relationship between arrays and pointers. Cover pointer operations, such as dereferencing and pointer-to-pointer concepts. Provide examples to illustrate the use of pointers in various scenarios.

Week 14: Strings and character arrays, explaining string handling functions, string manipulation techniques, and common string operations. Structures - Define the use of structures to group different data types, structure variables, accessing members, and the use of dot and arrow operators.

Week 16: Unions - definition and use of unions to store different data types in the same memory location. Compare and contrast structures and unions.

Learning Resources:

Textbook:

1. B. W. Kernighan, D. Ritchie, "The C Programming Language", 2nd edition, Pearson Education India, 2015
2. H. Schildt, "C: The Complete Reference", 4th edition, Mcgraw Hill
3. B. S. Gottfried, Programming with C, 4th edition, Mcgraw Hill

Reference Books:

1. P. Deitel and H. Deitel, "C How to Program", 8th edition, Pearson Education, 2015

E-Learning (and other) Resources: Other relevant study material, MOOCs (and other e-learning material) will be suggested and discussed during the lecture session(s) as and when deemed suitable.

Assessment Pattern:
The final grade will be based on the marks/ grades obtained in the end-semester exam and other assessments defined in the assessment table below. The relative grading method described in the university's academic regulations will be followed to grade the students.

Evaluation Component	Weightage (%)	Evaluation Schedule	Rubrics/Remarks
Lab Assignment	40% (20% each)	Sep 24 and Nov 24 (Last weeks)	Code Quality (30%) • Clean, readable, and well-documented code • Proper use of syntax and coding conventions Functionality (50%) • Code runs without errors • Meets all specified requirements and produces correct outputs Timeliness (20%) • Assignments submitted on or before deadlines

Mid Semester Examination	20%	As per University Calendar	Understanding of Concepts (40%) • Correct explanations of C programming concepts • Proper use of terminology Problem Solving (40%) • Ability to write correct code snippets • Logical problem-solving approach Code Efficiency (20%) • Efficient use of resources and minimal redundancy
End Semester Examination	40%	As per University Calendar	Comprehensive Knowledge (50%) • Demonstrates understanding of entire course material • Ability to apply concepts to solve complex problems Practical Application (30%) • Ability to write and debug complete programs • Application of theoretical knowledge in practical scenarios Critical Thinking (20%) • Analytical skills and innovative solutions • Ability to optimize and improve existing code

Experiential Learning: By incorporating problem-based learning and 50 days of coding, we aim to provide an experiential learning component encompassing more than 50% of the curriculum.

List of Experiments:

1. Implementing Basic Data Types and Variable Operations
2. Creating and Evaluating Expressions with Various Operators
3. Developing Programs Using If-Else and Switch Statements
4. Constructing Loops with For, While, and Do-While Constructs
5. Defining and Utilizing Functions for Specific Applications
6. Manipulating One-Dimensional and Multi-Dimensional Arrays
7. Exploring Pointer Basics and Performing Pointer Arithmetic
8. Handling Strings and Character Arrays with Built-In Functions
9. Utilizing Structures and Unions to Manage Complex Data

Student Responsibilities:

- Regular and attentive during lectures/tutorial classes
- Write class notes and submit assignments on time.
- Check announcements at Maitri / Google Classroom / E-mail regularly.
- Regularly check marks and attendance on the Maitri/Google Classroom and inform if there are any concerns.

Support for Slow Learners:

- Extra classes as per the requirements.
- Additional assignments.
- Group them with advanced learners for project development.

Challenges for Advanced Learners:

- Working on a mini project.

Attendance Policy: As per the University policy.

Quiz / Assignments: Two quizzes (each one before and after the mid-semester exam), two assignments, before and after mid-semester.

Evaluation procedures for tests, assignments, and lab: As per the weightage mentioned above, continuous evaluation in the lab.

Late assignment submission policy: Student loses one mark for each late day except for an exceptional situation like a medical problem.

Make-up examination/work policy: As per the University policy

Behaviour expectations: Attend classes regularly and be punctual. The use of mobile phones during lectures and lab work is strictly prohibited. Ensure academic integrity – no copying, no cheating, focus on learning and gaining knowledge.

Academic dishonesty/cheating/plagiarism: Any student violating expected behaviour will be punished according to the University's rules and regulations.