CURRICULUM FRAMEWORK FOR THREE YEAR UG DEGREE / FOUR YEAR UG DEGREE (HONOURS) / FOUR YEAR UG DEGREE (HONOURS WITH RESEARCH)

MINOR for UG Programme

Department of Computer Science

North Lakhimpur University

(As approved by Board of Studies in Computer Science held on 18th June 2026 and passed by Academic Council held on .....)

Semester-I

Course Code : **MIN –5-CSC-101**
Course Title : Programming in C
Nature of the Course : **MINOR**
Total Credit assigned : **5**
Component of the Course : **Lecture: 3 Practical: 2 Tutorial: 0**
Distribution of Marks : **End Semester: 42 Practical: 40 In Semester: 18**

Course Objectives

The main objectives of this course are:

Obj-1. Introduce the C programming environment, program structure, data types, operators, input/output operations and compilation-execution process.
Obj-2. Develop problem-solving ability using conditional and iterative constructs, functions, recursion, arrays and strings.
Obj-3. Enable students to use structures, unions, pointers, dynamic memory allocation, pre-processor directives and file handling in C.
Obj-4. Develop ability to design, test, debug and document modular C programs for given computational problems.

Course Content

Unit	Content	Marks	L	P	T
I	• Introduction to C • Overview of programming languages • Introduction to Algorithms, Flow Charts, Decision chart • History and features of C • Basic structure of a C program • Compilation and execution process • Distinction between C and other programming languages • Data types and variables • Constants and literals • Operators and expressions • Basic input/output operations • Formatted I/O • Introduction to files	5	7	8	0
II	Expressions, Conditional Statements and Iterative Statements • Conditional statements: if, if-else, nested if-else and switch • Iterative statements: while, do-while and for • Nested loops and nested conditional statements • Use of break and continue statements	9	13	14	0
III	Functions, Arrays and Strings • Introduction to functions • Function prototypes and function calls • Call by value and address-based parameter passing • Return values and void functions	16	13	20	0

	• Recursive functions • One-dimensional arrays • Multi-dimensional arrays • Strings and string manipulation				
IV	Derived Data Types: Structures and Unions • Structures • Unions • Pointers to structures and unions • Arrays of structures and unions	8	7	10	0
V	Pointers • Introduction to pointers • Pointers to basic data types • Pointers to arrays, strings, structures and unions • Pointer arithmetic and pointer-based data access	4	5	8	0
	Total	**42**	**45**	**60**	**0**

Mode of In-Semester Assessment

Mode of In-Semester assessment (18 Marks):

A. Sessional Activity

B. Sessional Tests: (two Sessional Tests)

Sessional activity:

- Quizzes& Assignments
  - Weekly quizzes on C syntax, data types, control structures, and functions.
  - Assignments on arrays, pointers, memory allocation, and file handling.
- Viva& Problem-Solving Sessions
  - Oral viva covering concepts of functions, recursion, and dynamic memory allocation.
  - Problem-solving discussions on complex programming scenarios and debugging techniques.

Suggested List of Laboratory Assignments

1. Write a program to calculate the sum and product of digits of an integer number.
2. Write a program to reverse a number.
3. Write a program to compute the sum of the first n terms of the series S = 1 + 1/2 + 1/3 + 1/4 + ...
4. Write a program to compute the sum of the first n terms of the series S = 1 - 2 + 3 - 4 + 5 - ... .
5. Write a program to check whether a given string is a palindrome.
6. Write a program to test whether a given number is prime and use it to generate all prime numbers less than 100.
7. Write a program to compute the factors of a given number.
8. Write a program to swap two numbers using macros.
9. Write a program to perform insertion, deletion, searching and traversal operations on an array.
10. Write a program to print the number of occurrences of each alphabet in text entered through command-line arguments.
11. Write a program to swap two numbers using pointers.
12. Write a program in which a function receives the addresses of two variables and modifies their contents.
13. Write a program to compute the area and circumference of a circle from the radius entered by the user.
14. Write a program to find the sum of n elements entered by the user using dynamic memory allocation.

15. Write a menu-driven program to perform string operations.
16. Write a program to merge two ordered integer arrays into one ordered array.
17. Write a program to display the Fibonacci series using recursion and iteration.
18. Write a program to calculate factorial using recursion and iteration.
19. Write a program to calculate GCD of two numbers using recursion and iteration.
20. Write a program to implement matrix operations using arrays and functions.
21. Write a program to create records of 10 students using structures and store them in a file.
22. Write a program to retrieve student information from a file and display it.
23. Write a program to copy the contents of one text file to another after removing white spaces.

Expected Course Learning Outcomes:

By the end of the course, students will be able to:

CLO1: Explain the C programming environment, program structure, data types, operators, input/output operations and compilation-execution process.

CLO2: Apply control statements, functions, recursion, arrays and strings to solve basic computational problems.

CLO3: Implement C programs using structures, unions, pointers, dynamic memory allocation, macros and file input/output.

CLO4: Analyze, debug and create modular C programs by selecting suitable data representations, functions and file-processing strategies.

Cognitive Map of Course Learning Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge	CLO1	CLO1				
Conceptual Knowledge		CLO1, CLO2	CLO2	CLO3		
Procedural Knowledge			CLO2, CLO3	CLO3, CLO4		
Metacognitive Knowledge						

Mapping of CLO with Programme Specific Outcomes (PSOs)

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9	PSO10

CLO1	2									
CLO2	3	2								
CLO3	3		2	1						
CLO4		2	2	1		1				
Average Contribution	**2**	**1**	**1**	**0.5**		**0.25**				

Scale: 3 = High contribution, 2 = Moderate contribution, 1 = Low contribution; blank = No direct mapping.

Suggested Readings

1. Kanetkar, Y. (2018). Let Us C. BPB Publications.
2. Kernighan, B. W., & Ritchie, D. M. (1988). The C Programming Language. Prentice Hall.
3. Griffiths, D., & Griffiths, D. (2012). Head First C: A Brain-Friendly Guide. O'Reilly Media.
4. Balagurusamy, E. (2016). Programming in ANSI C. McGraw Hill Education.

Semester-II

Course Code : **MIN –5-CSC-201**
Course Title : **Data Structure**
Nature of the Course : **MINOR**
Total Credit assigned : **5**
Component of the Course : **Lecture: 3 Practical: 2 Tutorial: 0**
Distribution of Marks : **End Semester: 42 Practical: 40 In Semester: 18**

Course Objectives

The main objectives of this course are:

Obj-1. Introduce abstract data types, data-structure classification, representation and algorithmic complexity.
Obj-2. Develop skills to implement and analyse arrays, sparse matrices and linked-list representations.
Obj-3. Enable students to apply stacks, queues, recursion and tree structures to solve computing problems.
Obj-4. Develop the ability to compare, evaluate and design data-structure-based solutions using searching, sorting and hashing techniques.

Course Content

Unit	Content	Marks	L	P	T
I	Introduction to Data Structures • Introduction to data structures and algorithms • Basic terminology and concepts • Linear and non-linear data structures • Abstract Data Types (ADTs) • Complexity analysis	6	6	0	0
II	Arrays and Linked Lists • Array implementation and analysis • Single and multidimensional arrays • Sparse matrices: array and linked representation • Linked list implementation and analysis • Singly, doubly and circular lists: array and linked representation	9	10	14	0
III	Searching, Sorting and Hashing • Linear search and binary search • Comparison of linear and binary search • Sorting: such as Bubble sort, selection sort, insertion sort, quick sort and merge sort • Comparison of sorting techniques • Introduction to hashing and hash tables • Collision resolution techniques • Separate chaining and open addressing	9	13	18	0

IV	Stacks, Queues and Recursion • Stack implementation and analysis • Applications of stack and limitations of array representation • Infix, prefix and postfix expressions and conversions • Queue implementation and analysis • Dequeue and priority queue • Recursion: recursive definitions, implementation, advantages, limitations and internal stack mechanism	9	8	14	0
V	Trees • Introduction to tree as a data structure • Binary trees and binary search trees • Insertion, deletion, recursive and iterative traversals in BST • Balanced trees and operations on AVL trees	9	8	14	0
	Total	**42**	**45**	**60**	**0**

Mode of In-Semester assessment (18 Marks):

A. Sessional Activity

B. Sessional Tests: (two Sessional Tests)

Sessional activity:

1. Implementation of Data Structures

- Implement **arrays, linked lists, and stacks** using a programming language of choice.
- Compare **array-based and linked-list-based** representations for efficiency.

2. Expression Conversion and Evaluation

- Develop a program to convert **infix expressions to postfix** and evaluate postfix expressions using stacks.

3. Tree Implementation

- Implement **Binary Search Trees (BST)** and perform insert, delete, and traversal operations (recursive and iterative).
- Compare **balanced (AVL) trees** with unbalanced trees in terms of efficiency.

4. Sorting and Hashing Techniques

- Implement and compare sorting algorithms (**Quick Sort, Merge Sort, and Insertion Sort**).
- Design and implement **hashing functions** with collision resolution techniques (separate chaining and open addressing).

Suggested List of Laboratory Assignments

1. Implement and analyse the time complexity of insertion, deletion and searching operations on an array.
2. Implement a singly linked list and perform insertion, deletion and searching operations.
3. Implement a stack using an array and analyse push, pop and peek operations.
4. Implement a queue using an array and analyse enqueue, dequeue and peek operations.
5. Implement conversion of infix expressions to postfix using a stack.
6. Implement a binary search tree and perform insertion, deletion and searching.
7. Implement a heap and perform heap operations such as insert, delete and build-heap.
8. Implement bubble sort, selection sort, insertion sort, quick sort and merge sort, and analyse their time complexity.
9. Implement a hash table using separate chaining and analyse its time complexity.
10. Implement dynamic hashing and analyse its performance.

Expected Course Learning Outcomes:

By the end of the course, students will be able to:

CLO1: Explain abstract data types, data-structure classification, representations and complexity analysis.
CLO2: Implement and analyse arrays, sparse matrices, linked lists, stacks, queues and recursive solutions.
CLO3: Apply trees, searching, sorting and hashing techniques to solve computing problems.
CLO4: Compare, evaluate and design suitable data-structure-based solutions for given constraints and justify implementation choices.

Cognitive Map of Course Learning Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge	CLO1	CLO1				
Conceptual Knowledge		CLO1, CLO2	CLO2	CLO3		
Procedural Knowledge			CLO2, CLO3	CLO3, CLO4		
Metacognitive Knowledge						

Mapping of CLO with Programme Specific Outcomes (PSOs)

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9	PSO10
CLO1	3									
CLO2	3	2								
CLO3	2	2	1							
CLO4		3	2	1						
Average Contribution	**2**	**1.75**	**0.75**	**0.25**						

Scale: 3 = High contribution, 2 = Moderate contribution, 1 = Low contribution; blank = No direct mapping.

Suggested Readings

1. Kanetkar, Y. (2019). Let Us C++: Dive into the Nitty-Gritties of C++ Language and Learn Why Programmers Prefer OOPs and C++. BPB Publications.
2. Drozdek, A. (2005). Data Structures and Algorithms in C++. Thomson Course Technology.
3. Langsam, Y., Augenstein, M. J., & Tenenbaum, A. M. (1996). Data Structures Using C and C++. Prentice Hall.
4. Anggoro, W. (2018). C++ Data Structures and Algorithms. Packt Publishing.

5. Malik, D. S. (2006). C++ Programming: From Problem Analysis to Program Design. Course Technology Cengage Learning.

SEMESTER-III

Course Code : MIN-2-CSC-301
Course Title : Programming with Python
Nature of the Course : MINOR
Total Credit assigned : 2
Component of the Course : Lecture: 1 Practical: 1 Tutorial: 0
Distribution of Marks : End Semester: 28 Practical: 20 In Semester: 12

Course Objectives:

The main objectives of this course are:

Obj-1. Introduce Python syntax, data types, operators, strings, input/output and control-flow statements.

Obj-2. Develop skills to use functions, modules, lists, tuples, dictionaries and arrays for computational problem solving.

Obj-3. Enable students to build, test and debug Python programs for file handling, exception handling, CSV/JSON processing and basic data visualization.

Course Content

Unit	Content	Marks	L	P	T
I	Python Basics, Strings and Control Flow • History and application areas of Python • Structure of a Python program • Identifiers and keywords • Operators and precedence • Basic data types and type conversion • Statements and expressions • Input/output statements • Strings: creation, storage, built-in functions, operators, slicing, joining and formatting • Conditional statements and loop-control statements • Nested control flow, continue, break, pass and exit	8	8	11	0
II	Functions and Core Data Structures • Built-in functions • Function definition and function call • Scope and lifetime of variables • Default parameters and command-line arguments • Lambda functions and assert statement • Importing user-defined modules • Mutable and immutable objects • Lists, tuples and dictionaries • Commonly used functions on lists, tuples and dictionaries • Passing lists, tuples and dictionaries as arguments • Use of math and NumPy modules for lists and arrays	10	12	9	0

III	Files, Exception Handling and Data Visualization • Types of files • Creating, reading and writing text and binary files • Pickle module • Reading and writing CSV and JSON files • Exception handling: try-except-else-finally, raise statement and hierarchy of exceptions • User-defined exceptions • Data visualization using Python libraries • 2D and 3D plots, histograms, pie charts, sine and cosine curves	10	10	10	0
	Total	**28**	**30**	**30**	**0**

Mode of In-Semester assessment (12 Marks):

A. Sessional Activity

B. Sessional Tests: (two Sessional Tests)

Sessional activity:

- **String Manipulation Exercise:**Practice string creation, slicing, joining, and formatting using built-in string functions.
- **Control Flow Project:**Create a program that uses conditional and loop control statements, including nested flows, continue, and break.
- **Function Implementation:**Write functions with different parameter types (default, command-line, lambda), and use scope and lifetime concepts.
- **Data Structures Challenge:**Implement operations on lists, tuples, and dictionaries, passing them as arguments to functions.
- **File Handling Task:**Read from and write to text, binary, CSV, and JSON files, using the Pickle module for serialization.
- **Exception Handling Simulation:**Build a program with custom exceptions, using try-except blocks and the raise statement.
- **Data Visualization Project:**Create visualizations (histograms, pie charts, sine/cosine curves) using Python libraries like Matplotlib

Suggested List of Laboratory Assignments

1. Write a program to find whether a number is prime.
2. Write a program to calculate m raised to the power n, where m and n are read from the user.
3. Write a parameterised function that returns True or False depending on whether the parameter passed is even or odd.
4. Write a program to print the summation of the series up to n terms: 1 - 2 + 3 - 4 + 5 - 6 + ...
5. Write a menu-driven program to find the frequency of a character, replace a character, remove the first occurrence of a character and remove all occurrences of a character in a string.
6. Write a program that accepts two strings and returns the indices of all occurrences of the second string in the first string; return -1 if it is absent.
7. Using NumPy, write a menu-driven program to create an array filled with 1s, find maximum/minimum values, compute dot product of two arrays and reshape a 1-D array to 2-D.
8. Write a function that takes a sentence as input and calculates the frequency of each letter using a dictionary.
9. For tuple t1 = (1, 2, 5, 7, 9, 2, 4, 6, 8, 10), print contents in two halves, print all even values as another tuple, concatenate t2 = (11, 13, 15) with t1, and return maximum and minimum values from t1.
10. Write a function that reads file1 and copies only alternate odd-numbered lines to file2.
11. Write a Python program to handle ZeroDivisionError when dividing a number by zero.
12. Write a program that reads a list of integers from the user and raises an exception if duplicate values are present.

13. Write a program using functions to display sine, cosine, polynomial and exponential curves.
14. Take months and profits made by company ABC over a year as input and represent the data using a line plot with suitable axis labels.

Expected Course Learning Outcomes:

By the end of the course, students will be able to:

CLO1: Explain Python syntax, data types, operators, strings, input/output and control-flow constructs.

CLO2: Develop Python programs using functions, modules, command-line arguments, lambda functions and variable scope.

CLO3: Apply lists, tuples, dictionaries and NumPy arrays to process and transform data.

CLO4: Create and evaluate Python programs that use file handling, CSV/JSON processing, exception handling and basic data visualization.

Cognitive Map of Course Learning Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge	CLO1	CLO1				
Conceptual Knowledge		CLO1, CLO2	CLO2	CLO3	CLO4	
Procedural Knowledge			CLO2, CLO3	CLO3, CLO4	CLO4	CLO4
Metacognitive Knowledge						

Mapping of CLO with Programme Specific Outcomes (PSOs)

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9	PSO10
CLO1	2			1						
CLO2	2			1						
CLO3	2	2		1						
CLO4		2	2	2		1				
Average Contribution	**1.5**	**1**	**0.5**	**1.25**		**0.25**				

Scale: 3 = High contribution, 2 = Moderate contribution, 1 = Low contribution; blank = No direct mapping.

Suggested Readings

1. Venkatesh, & Nagaraju, Y. (2021). Introduction to Python Programming. Khanna Publishing House.
2. Jose, J. (2023). Introduction to Computing and Problem Solving with Python. Khanna Publishing House.
3. Taneja, S., & Kumar, N. (2017). Python Programming: A Modular Approach with Graphics, Database, Mobile and Web Applications. Pearson.
4. Downey, A. (2015). Think Python (2nd ed.). O'Reilly Media.
5. Guttag, J. (2016). Introduction to Computation and Programming Using Python (2nd ed.). PHI Learning.

Semester-VII

Course Code : MIN –4-CSC-701
Course Title : OOP with JAVA
Nature of the Course : MINOR
Total Credit assigned : 4
Component of the Course : Lecture: 3 Practical: 1 Tutorial: 0
Distribution of Marks : End Semester: 42 Practical: 20 In Semester: 18

Course Objectives

The main objectives of this course are:

Obj-1. Introduce the Java platform, JVM, JDK/JRE, language features, data types, operators and control structures.

Obj-2. Develop skills to manipulate arrays, strings, command-line arguments and wrapper classes in Java.

Obj-3. Enable students to design classes and objects using constructors, methods, overloading, access control and nested or abstract classes.

Obj-4. Develop ability to build Java applications using inheritance, interfaces, packages, exception handling, multithreading and input/output programming.

Course Content

Unit	Content	Marks	L	P	T
I	Basics of Java • Features of Java • Bytecode and Java Virtual Machine • JDK and JRE • Data types and operators • Control statements: if-else, switch, while, do-while, for, for-each, break and continue	6	6	4	0
II	Arrays and Strings • Single and multidimensional arrays • String class and StringBuffer class • Operations on strings • Command-line arguments • Use of wrapper classes	6	6	6	0
III	Classes, Objects and Methods • Class, object and object reference • Constructor and constructor overloading • Method overloading and recursion • Passing and returning objects from methods • new operator, this and static keywords • finalize() method • Access control and modifiers • Nested class, inner class, anonymous inner class and abstract class	9	10	6	0

IV	Inheritance and Interfaces • Use of inheritance • Inheriting data members and methods • Constructors in inheritance • Multilevel inheritance • Method overriding and super keyword • final keyword and prevention of inheritance • Creation and implementation of interfaces • Interface reference • Comparison between abstract class and interface	6	12	6	0
V	Packages, Exception Handling, Multithreaded Programming and I/O Programming • Use of packages and import statement • Built-in and user-defined packages • Exception and error handling • try, catch, throw, throws and finally blocks • Built-in and custom exceptions • Throwable class • Overview of multithreaded programming • Introduction to streams, byte streams, character streams, readers and writers • File class, FileInputStream, FileOutputStream, InputStreamReader, OutputStreamWriter, FileReader, FileWriter and BufferedReader	15	11	8	0
	Total	**42**	**45**	**30**	**0**

Mode of In-Semester assessment (18 Marks):

A. Sessional Activity

B. Sessional Tests: (two Sessional Tests)

Sessional activity:

1. Basics of Java

- Write and execute **simple Java programs** using JDK and JRE.
- Implement **control statements (if-else, switch, loops)** with real-world examples.
- Demonstrate the use of **operators** through coding exercises.

2. Array and String

- Create and manipulate **single and multidimensional arrays** in Java.
- Perform **string operations** using String and StringBuffer classes.
- Write programs to demonstrate **command-line arguments** and **wrapper classes**.

3. Classes, Objects, and Methods

- Develop **Java classes with constructors and methods**.
- Implement **method overloading and recursion** with examples.
- Demonstrate **object creation, object passing, and returning** in methods.
- Work with **access control modifiers**, static and this keywords.

4. Inheritance and Interfaces

- Implement **single and multilevel inheritance** in Java.
- Demonstrate **method overriding, use of super, and final keyword**.
- Write Java programs using **interfaces and abstract classes**.
- Compare **abstract classes and interfaces** through practical examples.

5. Package, Exception Handling, Multithreading, and I/O Programming

- Develop **user-defined packages** and use built-in Java packages.
- Implement **exception handling mechanisms** using try, catch, throw, throws, and finally.
- Write programs to handle **custom exceptions and built-in exceptions**.
- Demonstrate **multithreading concepts** with thread creation and synchronization.

- Work with **Java I/O classes**, including FileReader, FileWriter, BufferedReader, and Streams.

Suggested List of Laboratory Assignments

1. Write a program to convert rupees to dollars using a given conversion rate.
2. Write a program to calculate the percentage marks of a student when marks of six subjects are given.
3. Write a program to enter two numbers and perform addition, subtraction, multiplication and division.
4. Write a program to find the length of a string and print its second half.
5. Write a program to accept a line and count vowels and consonants.
6. Write a program to count the number of words that start with capital letters.
7. Write a program to check whether a given number or string is a palindrome.
8. Create a class that repeatedly accepts sentences and displays the count of each vowel until the user enters "quit"; display total vowel counts for all sentences.
9. Write an interactive program to print a string in pyramid form.
10. Write an interactive program to print a diamond shape based on user input.
11. Create a Student class and write a student manager program to manipulate student information from files using FileInputStream and FileOutputStream.
12. Refine the student manager program using BufferedReader and BufferedWriter.
13. Refine the student manager program using DataInputStream and DataOutputStream.

Expected Course Learning Outcomes:

By the end of the course, students will be able to:

CLO1: Explain the Java platform, JVM, JDK/JRE, data types, operators and control-flow constructs.
CLO2: Apply arrays, strings, command-line arguments and wrapper classes to solve basic Java programming problems.
CLO3: Design classes and objects using constructors, methods, overloading, recursion, access control, static/this keywords and nested or abstract classes.
CLO4: Develop and evaluate object-oriented Java applications using inheritance, interfaces, packages, exception handling, multithreading and file I/O.

Cognitive Map of Course Learning Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge	CLO1	CLO1				
Conceptual Knowledge		CLO1, CLO2	CLO2	CLO3	CLO4	
Procedural Knowledge			CLO2, CLO3	CLO3, CLO4	CLO4	CLO4
Metacognitive Knowledge						

Mapping of CLO with Programme Specific Outcomes (PSOs)

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9	PSO10
CLO1	2									
CLO2	2	1								

CLO3	2		2	1						
CLO4		2	3	2		1				
Average Contribution	**1.5**	**0.75**	**1.25**	**0.75**		**0.25**				

Scale: 3 = High contribution, 2 = Moderate contribution, 1 = Low contribution; blank = No direct mapping.

Suggested Readings

1. Schildt, H., &Skrien, D. (2012). Java Programming: A Comprehensive Introduction. McGraw-Hill Higher Education.
2. Balagurusamy, E. (2014). Programming with Java: A Primer. McGraw-Hill Education.
3. Horstmann, C. S. (2024). Core Java, Volume I: Fundamentals. Pearson Education.