Assam University, Silchar

Four Year Undergraduate Programme

Implemented under NEP 2020

Effective from the Academic Year 2023-24

Syllabus of Computer Science

Approved in the 94th meeting of the Academic Council on 20th July 2023
vide Resolution No AC:94:07-23:6

Head
Department of Computer Science
Assam University, Silchar
PIN 788011

Programme Specific Outcome

On successful completion of the FYUG Computer Science programme, students will be able to:

1. Understand the concepts of computer architecture, data structure, networking and operating system.
2. Write and execute programs in various programming languages to solve real life problems.
3. Learn the importance of application of computer science in various fields.
4. Exhibit reasoning required for solving artificial intelligence problems.
5. Develop skills of analysis of algorithms related to space and time complexity of Algorithms.
6. Acquaint them with the knowledge of Cyber Security.
7. Develop theoretical as well as practical knowledge of Software development.
8. Seek admission in Post-graduate/Research Programmes in Computer Science, Information Technology and Computer Application etc.
9. Seek employment in various jobs in the Government sector as well as IT and related industries and perform various roles related to software development, testing and maintenance.

Semester wise list of Computer Science Discipline Specific Core (DSC) Papers

SEMESTER	COURSE CODE	TITLE OF COURSES	CREDITS
I	**CSCDSC101**	Digital Computer Fundamentals	3
	CSCDSC102	Discrete Mathematics	3
II	**CSCDSC151**	Data Structure	3
	CSCDSC152	Lab on Data Structure	3
III	**CSCDSC201**	Computer Organization and Architecture	4
	CSCDSC202	Operating Systems	4
IV	**CSCDSC251**	Object Oriented Programming with Java	4
	CSCDSC252	Database Management System	4
	CSCDSC253	Lab on Java & DBMS	4
V	**CSCDSC301**	Computer Graphics	4
	CSCDSC302	System Analysis and Design	4
	CSCDSC303	Lab on Graphics Programming	4
VI	**CSCDSC351**	Computer Network and Internet Technology	4
	CSCDSC352	Theory of Computation	4
	CSCDSC353	Microprocessor and Systems Programming	4
	CSCDSC354	Lab on Internet Technology & Microprocessor and Systems Programming	4
VII	**CSCDSC401**	Design & Analysis of Computer Algorithms	4
	CSCDSC402	Principles of Compiler Design	4
	CSCDSC403	Artificial Intelligence	4
	CSCDSC404	Lab on DACA & Compiler Design	4
VIII	**CSCDSC451**	Software Engineering	4
	CSCDSC452(A)	Image Processing	4
	CSCDSC452(B)	Data Analytics	4
	CSCDSC453	Natural Language Processing	4
	CSCDSC454(A)	IoT	4
	CSCDSC-454(B)	Cloud Computing	4
	OR		
	CSCDSC451	Research Methodology	4
	CSCDSC455	Research Project/Dissertation	12

Semester wise list of Computer Science Discipline Specific Minor (DSM) Papers

SEMESTER	COURSE CODE	TITLE OF COURSES	CREDITS	DSM1/DSM2
I	**CSCDSM101**	Programming with C	3	**DSM1**
II	**CSCDSM151**	Programming with C	3	**DSM2**
III	**CSCDSM201**	Database Management System	4	**DSM1**
IV	**CSCDSM251**	Lab on C & DBMS	3	**DSM1**
	CSCDSM252	Database Management System	3	**DSM2**
V	**CSCDSM301**	Operating Systems	3	**DSM1**
	CSCDSM302	Operating Systems	3	**DSM2**
VI	**CSCDSM351**	Lab on C & DBMS	4	**DSM2**
VII	**CSCDSM401**	Internet Technologies	4	**DSM1**
VIII	**CSCDSM451**	Internet Technologies	4	**DSM2**

Semester wise list of Computer Science Skill Enhancement Course (SEC) Papers

SEMESTER	COURSE CODE	TITLE OF COURSES	CREDITS
I	**CSCSEC101**	Programming with C	3
II	**CSCSEC151**	Python Programming	3
III	**CSCSEC201**	Programming with C++ & Lab on OS and C++	3

Semester wise list of Computer Science Interdisciplinary Course (IDC) Papers

SEMESTER	COURSE CODE	TITLE OF COURSES	CREDITS
I	**CSCIDC101**	Computer Fundamentals & Applications	3
II	**CSCIDC151**	Programming Fundamentals with C	3
III	**CSCIDC201**	Introduction to Web Designing & Cyber Security	3

Syllabi of Computer Science DSC Courses

Semester	**: I**
Course Type	**: DSC**
Course Code	**: CSCDSC101**
Name of the Course	**: Digital Computer Fundamentals**
Learning level	**: Foundation or Introductory Course**
Credits	**: 3**
Contact Hours	**: 45**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives:

1. Familiarize with the fundamental concepts, terminology, and building blocks of digital logic circuits.
2. Introduce learners to Boolean algebra, logic gates, truth tables, and logic expressions, enabling them to understand and manipulate digital signals and logic functions.
3. How to design and analyze combinational logic circuits using logic gates and Boolean algebra, including applications such as arithmetic circuits, multiplexers, and decoders.
4. Introduce learners to sequential logic circuits, including flip-flops, registers, counters, and state machines, enabling them to design and analyze circuits that store and process information over time.
5. Provide an overview of digital memory elements and storage devices, including registers, RAM, ROM, and non-volatile memory, emphasizing their role in data storage and retrieval.

UNIT I

Computer Definition, Characteristics of Computers, Evolution of Computers & its applications, Types of Computers, Basic Organization of a Digital Computer, Computer design, Computer Architecture, Hardware and Software, Central Processing Unit, Input devices, Output devices, Computer Memory & Storage.

UNIT II

Number System, Boolean Algebra and Logic gates, simplification of Boolean functions- Map Method, Two and Three- variable Maps, Product of sums, Simplification, NAND and NOR implementation of logic gates, Don't Care Condition, The Tabulation Method, Determination of Prime-implicants.

UNIT III

Combinational Logic: Introduction, Design Procedures, Adders, Subtractors, Code Conversion, Analysis Procedure, Multilevel NAND Circuits, Exclusive-OR and Equivalence

Functions, Binary Parallel Adder, Decimal Adder, Magnitude Comparator, Decoders, Multiplexers, Read Only Memory (ROM), Programmable Logic Array (PLA).

UNIT IV

Sequential Logic Circuits, Introduction, Flip-Flops, Analysis of Clocked Sequential circuits, State reduction and Assignment, Flip Flop Excitation Table, Design Procedure, Design of Counters, Design with State Equations.

UNIT V

Registers, Counters, Memory Unit: Introduction, Registers, shift Registers, Ripple Counters, Synchronous Counters, Timing Sequences, The Memory Unit, Examples of Random Access Memories.

Course outcomes: *After successful completion of the course, the students will be able to:*

1. Demonstrate a solid understanding of the fundamental principles, terminology, and building blocks of digital logic circuits.
2. Develop skills in designing and analyzing combinational logic circuits using logic gates, Boolean algebra, and truth tables.
3. Acquire knowledge and skills in designing and analyzing sequential logic circuits using flip-flops, registers, counters, and state machines.
4. Demonstrate proficiency in implementing and optimizing digital logic circuits for performance, power consumption, and cost considerations.

Text Books:

1. M. Morris Mano, **Digital Design**, Third Edition, Prentice Hall India, 2009.
2. Donald P. Leach, Albert Paul Malvino&GoutamSaha, **Digital Principles and Applications**, Eighth Edition, Tata McGraw Hill, 2014.

Reference Books:

1. P. V. S. Rao, **Perspectives in Computer Architecture**, Prentice Hall India, 2004.
2. R. P. Jain, **Modern Digital Electronics**, Fourth Edition, McGraw Hill Education, 2009.
3. Thomas C. Bartee, **Digital Computer Fundamental**, Sixth Edition, McGraw Hill Education.

Semester	**: I**
Course Type	**: DSC**
Course Code	**: CSCDSC102**
Name of the Course	**: Discrete Mathematics**
Learning level	**: Foundation or Introductory Course**
Credits	**: 3**
Contact Hours	**: 45**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives:

1. The purpose of the course is to familiarize the concepts of mathematical structures that are fundamentally discrete.
2. To introduce the concepts of mathematical logic.
3. To provide an overview of sets, relations and functions; and their associated operations.
4. Give an understanding of graphs and trees.

UNIT I

Mathematical Logic: Statements and Notations, Connectives, Normal forms, Equivalences, Predicate calculus, Quantifiers, Inference theory of the predicate calculus.

UNIT II

Set Theory and Ordered Sets: Basic concept of set theory, Operations with Sets, Function, Relations, Properties of Relations, Representing Relations, Composition of Relations, Closures of Relations, Ordered Sets, Hasse Diagrams of Partially Ordered Sets

UNIT III

Ordering Relations, Lattices and Boolean Algebra: Partial Ordering Relations; Equivalence Relations, Lattices, Bounded Lattices, Distributive Lattices, Complements, Complemented Lattices, Introduction to Boolean algebra, Boolean Functions, Representation and Minimization of Boolean functions

UNIT IV

Trees: Basic Concepts of Tree, Properties of Trees, Pendant vertices in a Tree, Centre of a Tree, Rooted binary trees, Complete and extended Binary Tree.

UNIT V

Graph theory with applications: Basic concepts of Graph Theory, Incidence and degree, Isomorphism, Homomorphism, Sub graphs and Union of graphs, multigraphs and weighted graphs, Planar Graphs, Walks, Paths and Circuits, Components and Connectedness, Eulerian graph, Hamiltonian graph, Complete Graph, Regular Graph, Bipartite graph, cut-sets and cut-vertices.

Course Outcomes: *After successful completion of the course, the students will be able to*

1. Apply mathematical logic to solve problems.
2. Learn the concept of sets, relations, functions and lattice.
3. Model and solve real world problems using graphs and trees.

Text Books:

1. Seymour Lipschutz and Marc Lars Lipson, **Discrete Mathematics**, Fourth Edition Schaum's Outline Series, McGraw Hill, 2022.
2. Kenneth H. Rosen, **Discrete Mathematics and Its Applications**, Seventh Edition, McGraw Hill, 2012.
3. Swapan K Sarkar, **A Textbook of Discrete Mathematics**, 9th Edition, S Chand & Co Ltd, 2016.

Reference Books:

1. D.J. Hunter, **Essentials of Discrete Mathematics**, Jones and Bartlett Publishers, 3rd Edition, 2008.
2. C.L. Liu, D.P. Mahopatra, **Elements of Discrete mathematics**, 2nd Edition, Tata McGraw Hill, 1985.
3. Deo N., **Graph Theory with Applications to Engineering and Computer Science**, PHI; 6th edition 2010.

Semester	**: II**
Course Type	**: DSC**
Course Code	**: CSCDSC151**
Name of the Course	**: Data Structure**
Learning level	**: Foundation or Introductory Course**
Credits	**: 3**
Contact Hours	**: 45**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives:

1. *Introduce the basic concepts and principles of data structures, including their definition, properties, and characteristics.*
2. *Familiarize students with the implementation of various data structures using programming languages, including arrays, linked lists, stacks, queues, trees, graphs, and hash tables.*
3. *How to analyze the time and space complexity of different data structures and algorithms, enabling them to make informed decisions regarding their selection and usage.*
4. *Cover various searching and sorting algorithms, including linear search, binary search, bubble sort, insertion sort, selection sort, merge sort, quicksort, and their analysis.*

5. *Cover the concepts of hashing, hash functions, collision resolution techniques, and the implementation and applications of hash tables.*

UNIT I

Arrays: Single and Multi-dimensional Arrays, Sparse Matrices (Array and Linked Representation).

Stacks: Implementing single / multiple stack/s in an Array; Prefix, Infix and Postfix expressions, Utility and conversion of these expressions from one to another; Applications of stack; Limitations of Array representation of stack.

Recursion: Developing Recursive Definition of Simple Problems and their implementation; Advantages and Limitations of Recursion.

UNIT II

Linked Lists: Singly, Doubly and Circular Lists (Array and Linked representation); Normal and Circular representation of Stack in Lists; Self Organizing Lists; Skip Lists. **Queues:** Array and Linked representation of Queue, De-queue, Priority Queues.

UNIT III

Trees: Introduction to Tree as a data structure; Binary Trees (Insertion, Deletion, Recursive and Iterative Traversals in Binary Search Trees); Threaded Binary Trees (Insertion, Deletion, Traversals); Height-Balanced Trees (Various operations on AVL Trees), Heap Tree.

UNIT IV

Searching and Sorting: Linear Search, Binary Search, and Comparison of Linear and Binary Search, Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, Quick Sort, and Comparison of Sorting Techniques.

UNIT V

Hashing: Introduction to Hashing, Hash Table, Hash Key, Hash Function, Characteristics of Good Hash Functions, Types of Hash Functions, Collision, Resolving Collision by Open Addressing & closed Addressing: Linear probing, Quadratic probing, Double Hashing.

Course outcomes: *After successful completion of the course, the students will be able to*

1. *Demonstrate a solid understanding of various data structures, including arrays, linked lists, stacks, queues, trees and hash tables.*
2. *Develop proficiency in implementing data structures using programming languages, including creating and manipulating data structures through appropriate algorithms.*
3. *Apply analytical skills to analyze the time and space complexity of algorithms associated with different data structures, allowing for informed decision-making in algorithm selection.*
4. *Enhance critical thinking skills and problem analysis abilities by identifying the appropriate data structures and algorithms to solve given problems efficiently.*

Text Books:

1. Seymour Lipschutz, **Data Structures**, Schaum's Outline Series, TMH, 4th Edition, 2019.
2. Adam Drozdek, **Data Structures and Algorithms in C++**, Cengage Learning, 3rd Edition, 2012.
3. SartajSahni, **Data Structures, Algorithms and Applications in C++**, Universities Press, 2nd Edition, 2011.

Reference Books:

1. D.S Malik, **Data Structure using C++**, Cengage Learning, Second Edition, 2010.
2. Aaron M. Tenenbaum, Moshe J. Augenstein, YedidyahLangsam, **Data Structures Using C and C++**, PHI, 2nd Edition, 2009.
3. Robert L. Kruse, **Data Structures and Program Design in C++**, Pearson, 3rd Edition, 1999.

Semester	**: II**
Course Type	**: DSC**
Course Code	**: CSCDSC152**
Name of the Course	**: Lab on Data Structure**
Learning level	**: Foundation or Introductory Course**
Credits	**: 3**
Contact Hours	**: 90**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives:

1. *Help students apply the theoretical concepts of data structures in a practical setting. It should provide exercises and programming assignments that require students to implement and manipulate different data structures.*
2. *Enhancing students' programming skills by providing practical programming exercises.*
3. *It should encourage students to write code, debug, and test their implementations of data structures and associated algorithms.*

*This paper provides practical knowledge of data structure. List of laboratory programming assignments (not limited to these)***:**

1. Write a program to search an element from a list. Give users the option to perform Linear or Binary search. Use Template functions.
2. WAP using templates to sort a list of elements. Give users the option to perform sorting using Insertion sort, Bubble sort or Selection sort.
3. Implement Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list and concatenate two linked lists (include a function and also overload operator +).
4. Implement Doubly Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.

5. Implement Circular Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.
6. Perform Stack operations using Linked List implementation.
7. Perform Stack operations using Array implementation. Use Templates.
8. Perform Queues operations using Circular Array implementation. Use Templates.
9. Create and perform different operations on Double-ended Queues using Linked List implementation.
10. WAP to scan a polynomial using a linked list and add two polynomials.
11. WAP to calculate factorial and to compute the factors of a given no. (i)using recursion, (ii) using iteration
12. WAP to display fibonacci series (i)using recursion, (ii) using iteration
13. WAP to calculate GCD of 2 number (i) with recursion (ii) without recursion
14. WAP to create a Binary Search Tree and include following operations in tree: (a) Insertion (Recursive and Iterative Implementation) (b) Deletion by copying (c) Deletion by Merging (d) Search a no. in BST (e) Display its preorder, postorder and inorder traversals Recursively (f) Display its preorder, postorder and inorder traversals Iteratively (g) Display its level-by-level traversals (h) Count the non-leaf nodes and leaf nodes (i) Display height of tree (j) Create a mirror image of tree (k) Check whether two BSTs are equal or not
15. WAP to convert the Sparse Matrix into non-zero form and vice-versa.
16. WAP to reverse the order of the elements in the stack using additional stack.
17. WAP to reverse the order of the elements in the stack using additional Queue.
18. WAP to implement Diagonal Matrix using a one-dimensional array.
19. WAP to implement Lower Triangular Matrix using a one-dimensional array.
20. WAP to implement the Upper Triangular Matrix using a one-dimensional array.
21. WAP to implement Symmetric Matrix using a one-dimensional array.
22. WAP to create a Threaded Binary Tree as per inorder traversal, and implement operations like finding the successor / predecessor of an element, insert an element, inorder traversal.
23. WAP to implement various operations on AVL Tree.

Course outcomes: *After successful completion of the course, the students will be able to*

1. *Students should be able to demonstrate a solid understanding of various data structures such as arrays, linked lists, stacks, queues, trees, graphs, and hash tables. They should be able to explain the characteristics, operations, and applications of these data structures.*
2. *Students should be able to implement data structures and associated algorithms using a programming language.*
3. *Students should be able to analyze the time and space complexity of algorithms associated with different data structures.*

Syllabi of Computer Science DSM Courses

Semester : I
Course Type : DSM
Course Code : CSCDSM101
Name of the Course : Programming with C
Learning level : Foundation or Introductory Course
Credits : 3
Contact Hours : 45
Total Marks : 100
End Semester Marks : 70
Internal Marks : 30

Course objectives:

1. *To introduce the concepts of programming and programming language C.*
2. *To explain the concepts of functions and programme structure in C*
3. *To explain how to write and implement C programs*
4. *To explain the concept and working of pointers and files in C*
5. *To introduce the low level programming in C*

UNIT-I

Fundamentals of computer programming with C – Data Types, Expressions, Operations – input, output; Writing simple C programs; Control structures (WHILE, DO-WHILE, FOR, IF-ELSE, SWITCH, BREAK, CONTINUE, GOTO STATEMENTS, nested loops etc.) and writing programs using control structures; solving elementary programming problems from various areas of applications including mathematics and statistics.

UNIT-II

Functions and program structure – Defining and accessing functions in C, passing arguments to a function, specifying argument data types – Illustration with example programs and problem solving through programs; Function prototypes, Functions returning non integers; Storage classes – Automatic, External, Static and Register variables, Scope rules, Header files, Block structure; Recursion in C – writing recursive programs and problem solving, The C Preprocessor

UNIT-III

Definition and array processing, passing arrays to a function, multidimensional arrays, Arrays and Strings; POINTERS – pointers and addresses, pointer declaration, pointers and function arguments – passing pointers to a function, Pointers and one dimensional arrays; Address arithmetic – operations on pointers, character pointers and functions; Pointer arrays/arrays of pointers, pointers to pointers, initialization of pointer arrays, pointers and multidimensional arrays; Command line arguments, Pointers to functions, passing functions to other functions.

UNIT-IV

Structures and Unions – Basics of structures, processing of structures, user defined data types (typedef), Structures and Pointers, Structures and functions – passing structures to a function, Arrays of structures, Pointers to structures, Self-referential structures, Table lookup, UNIONS. writing programs and problem solving with structure and union

UNIT-V

Input and output – Standard input and output, Formatted output – printf, Variable length argument, Formatted input - scanf; Data files – opening and closing data file, creating a data file, processing a data file, file access, unformatted data files, miscellaneous function in C; Low Level programming – Register variables, Bitwise operations, Bit fields, Enumeration, Commands Line arguments/parameters, Library functions, Macros, The C preprocessor.

Course outcomes: *After successful completion of the course, the students will be able to*

1. *Learn the concepts of Programming in C*
2. *Learn thoroughly the building blocks of C programming language*
3. *Write and implement C programs and solve problems through programming*
4. *Learn the Concept of pointers and files in C & Programming with C.*
5. *Design and implement programs using pointers and files in C.*

Text Books:

1. Brian W. Kernighan and Denis M. Ritchie, **The C Programming Language**, Pearson Education India; 2nd edition, 2015.
2. Byron S Gottfried, **Programming with C**, McGraw Hill Education; 4th Edition, 2018.
3. E Balagurusamy, **Programming in ANSI C**, McGraw Hill Education; Eighth edition, 2019
4. V. Rajaraman, **Computer Programming in C**, PHI Learning Private Limited; Second edition, 2019

Reference Books:

1. Yashavant P. Kanetkar, **Let Us C**, BPB; 16th edition, 2017.
2. Kamthane, **Programming in C**, Pearson Education India; 3rd edition, 2015.

Semester	**: II**
Course Type	**: DSM**
Course Code	**: CSCDSM151**
Name of the Course	**: Programming with C**
Learning level	**: Foundation or Introductory Course**
Credits	**: 3**
Contact Hours	**: 45**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course objectives:

1. *To introduce the concepts of programming and programming language C.*
2. *To explain the concepts of functions and programme structure in C*
3. *To explain how to write and implement C programs*
4. *To explain the concept and working of pointers and files in C*
5. *To introduce the low level programming in C*

UNIT-I

Fundamentals of computer programming with C – Data Types, Expressions, Operations – input, output; Writing simple C programs; Control structures (WHILE, DO-WHILE, FOR, IF-ELSE, SWITCH, BREAK, CONTINUE, GOTO STATEMENTS, nested loops etc.) and writing programs using control structures; solving elementary programming problems from various areas of applications including mathematics and statistics.

UNIT-II

Functions and program structure – Defining and accessing functions in C, passing arguments to a function, specifying argument data types – Illustration with example programs and problem solving through programs; Function prototypes, Functions returning non integers; Storage classes – Automatic, External, Static and Register variables, Scope rules, Header files, Block structure; Recursion in C – writing recursive programs and problem solving, The C Preprocessor

UNIT-III

Definition and array processing, passing arrays to a function, multidimensional arrays, Arrays and Strings; POINTERS – pointers and addresses, pointer declaration, pointers and function arguments – passing pointers to a function, Pointers and one dimensional arrays; Address arithmetic – operations on pointers, character pointers and functions; Pointer arrays/arrays of pointers, pointers to pointers, initialization of pointer arrays, pointers and multidimensional arrays; Command line arguments, Pointers to functions, passing functions to other functions.

UNIT-IV

Structures and Unions – Basics of structures, processing of structures, user defined data types (typedef), Structures and Pointers, Structures and functions – passing structures to a function, Arrays of structures, Pointers to structures, Self-referential structures, Table lookup, UNIONS. writing programs and problem solving with structure and union

UNIT-V

Input and output – Standard input and output, Formatted output – printf, Variable length argument, Formatted input - scanf; Data files – opening and closing data file, creating a data file, processing a data file, file access, unformatted data files, miscellaneous function in C; Low Level programming – Register variables, Bitwise operations, Bit fields, Enumeration, Commands Line arguments/parameters, Library functions, Macros, The C preprocessor.

Course outcomes: *After successful completion of the course, the students will be able to*

1. *Learn the concepts of Programming in C*
2. *Learn thoroughly the building blocks of C programming language*

3. Write and implement C programs and solve problems through programming
4. Learn the Concept of pointers and files in C & Programming with C.
5. Design and implement programs using pointers and files in C.

Text Books:

1. Brian W. Kernighan and Denis M. Ritchie, **The C Programming Language**, Pearson Education India; 2nd edition, 2015.
2. Byron S Gottfried, **Programming with C**, McGraw Hill Education; 4th Edition, 2018.
3. E Balagurusamy, **Programming in ANSI C**, McGraw Hill Education; Eighth edition, 2019
4. V. Rajaraman, **Computer Programming in C**, PHI Learning Private Limited; Second edition, 2019

Reference Books:

1. Yashavant P. Kanetkar, **Let Us C**, BPB; 16th edition, 2017.
2. Kamthane, **Programming in C**, Pearson Education India; 3rd edition, 2015.

Syllabi of Computer Science SEC Courses

Semester : **I**
Course Type : **SEC**
Course Code : **CSCSEC101**
Name of the Course : **Programming with C**
Learning level : **Foundation or Introductory Course**
Credits : **3**
Contact Hours : **30**
Total Marks : **100**
End Semester Marks : **80 [50 (Theory) + 30 (Lab)]**
Internal Marks : **20**

Course objectives:

1. *To introduce the concepts of programming and programming language C.*
2. *To explain the concepts of functions and programme structure in C*
3. *To explain how to write and implement C programs*
4. *To explain the concept and working of pointers and files in C*
5. *To introduce the low level programming in C*

UNIT-I

Fundamentals of computer programming with C – Data Types, Expressions, Operations – input, output; Writing simple C programs; Control structures (WHILE, DO-WHILE, FOR, IF-ELSE, SWITCH, BREAK, CONTINUE, GOTO STATEMENTS, nested loops etc.) and writing programs using control structures; solving elementary programming problems from various areas of applications including mathematics and statistics.

UNIT-II

Functions and program structure – Defining and accessing functions in C, passing arguments to a function, specifying argument data types – Illustration with example programs and problem solving through programs; Function prototypes, Functions returning non integers; Storage classes – Automatic, External, Static and Register variables, Scope rules, Header files, Block structure; Recursion in C – writing recursive programs and problem solving, The C Preprocessor

UNIT-III

Definition and array processing, passing arrays to a function, multidimensional arrays, Arrays and Strings; POINTERS – pointers and addresses, pointer declaration, pointers and function arguments – passing pointers to a function, Pointers and one dimensional arrays; Address arithmetic – operations on pointers, character pointers and functions; Pointer arrays/arrays of pointers, pointers to pointers, initialization of pointer arrays, pointers and multidimensional arrays; Command line arguments, Pointers to functions, passing functions to other functions.

UNIT-IV

Structures and Unions – Basics of structures, processing of structures, user defined data types (typedef), Structures and Pointers, Structures and functions – passing structures to a function, Arrays of structures, Pointers to structures, Self-referential structures, Table lookup, UNIONS. writing programs and problem solving with structure and union

UNIT-V

Input and output – Standard input and output, Formatted output – printf, Variable length argument, Formatted input - scanf; Data files – opening and closing data file, creating a data file, processing a data file, file access, unformatted data files, miscellaneous function in C; Low Level programming – Register variables, Bitwise operations, Bit fields, Enumeration, Commands Line arguments/parameters, Library functions, Macros, The C preprocessor.

Course outcome: After successful completion of the course, the students will be able to

1. *Learn the concepts of Programming in C*
2. *Learn thoroughly the building blocks of C programming language*
3. *Write and implement C programs and solve problems through programming*
4. *Learn the Concept of pointers and files in C & Programming with C.*
5. *Design and implement programs using pointers and files in C.*

Text Books:

1. Brian W. Kernighan and Denis M. Ritchie, **The C Programming Language**, Pearson Education India; 2nd edition, 2015.
2. Byron S Gottfried, **Programming with C**, McGraw Hill Education; 4th Edition, 2018.
3. E Balagurusamy, **Programming in ANSI C**, McGraw Hill Education; Eighth edition, 2019
4. V. Rajaraman, **Computer Programming in C**, PHI Learning Private Limited; Second edition, 2019

Reference Books:

1. Yashavant P. Kanetkar, **Let Us C**, BPB; 16th edition, 2017.
2. Kamthane, **Programming in C**, Pearson Education India; 3rd edition, 2015.

LAB: PART B: Programming with C (Practical): 30 Hours. (Practical /Project/Field work): Total marks: 30

Pass marks: 12

This part provides the practical knowledge of Programming with C.

Course Objectives:

1. *To explain design and implementation of C programs*

2. *To explain writing and executing programs with control structures and functions*
3. *To explain writing and executing programs with pointers in C*
4. *To explain writing and executing programs with structures and unions*
5. *To explain writing and executing programs with files in C*

Problem solving of various nature by implementing programs in C Programming languages based on unit wise contents of the theory paper Programming with C**.** Following are some programming tasks for laboratory programming assignments but the assignments are not limited to these only.

List of laboratory programming assignments (*not limited to these*)***:***

1. Write a program to
   a) Produce ASCII equivalent of given number
   b) Find the divisor or factorial of a given number.
   c) Evaluate the following algebraic expressions after reading necessary values from the user (ax+b)/ (ax-b) -2.5 log x-cos 30+|x2-y2|+sqrt (2xy) - (x5+10x4+8 x 3+4x+2)
   d) Find sum of a geometric series
   e) Cipher a string
   f) Check whether a given string follows English capitalization rules
   g) Find sum of the following series 1+ ½ + 1/3 +_____+1/20
   h) Search whether a given substring exists in an input string or not and then delete this string from the input string.
2. Write a recursive program for tower of Hanoi problem
3. The Fibonacci sequence of numbers is 1, 1, 2, 3, 5, 8……. Based on the recurrence relation F(*n*)=F(*n*-1)+F(*n*-2) for n>2 Write a recursive program to print the first *n* Fibonacci number
4. Write a menu driven program for matrices to do the following operation depending on whether the operation requires one or two matrices
   a) Addition of two matrices
   b) Subtraction of two matrices
   c) Finding upper and lower triangular matrices
   d) Trace of a matrix
   e) Transpose of a matrix
   f) Check of matrix symmetry
   g) Product of two matrices.
5. Write a program that takes two operands and one operator from the user perform the operation and then print the answer

7. Write functions to add, subtract, multiply and divide two complex numbers (x+iy) and (a+ib) Also write the main program.

8. Write a menu driven program for searching and sorting with following options:

   a) Searching (1) Linear searching (2) Binary searching

b) Sorting (1) Insertion sort (2) Selection sort

9. Write a program to copy one file to another, use command line arguments.
10. Write a program to mask some bit of a number (using bit operations)
11. An array of records contains information of managers and workers of a company. Print all the data of managers and workers in separate files.

Semester	**: II**
Course Type	**: SEC**
Course Code	**: CSCSEC151**
Name of the Course	**: Python Programming**
Learning level	**: Foundation or Introductory Course**
Credits	**: 3**
Contact Hours	**: 30**
Total Marks	**: 100**
End Semester Marks	**: 80 [50 (Theory) + 30 (Lab)]**
Internal Marks	**: 20**

Course Objectives:

1. *Familiarize students with the basics of Python programming language, including syntax, variables, data types, and control structures syntax, variables, data types, and control structures.*
2. *Introduce students to fundamental programming concepts such as variables, data types, operators, conditional statements, loops, and functions, within the context of Python.*
3. *Explore various data structures in Python, such as lists, tuples, dictionaries, and sets, and develop the skills to manipulate, access, and organize data using these structures.*
4. *Teach students how to read from and write to files using Python, including text files and CSV files, enabling them to handle data stored in external files.*
5. *Develop skills in handling errors and exceptions in Python programs, allowing for graceful error handling and robustness.*

UNIT-I

Introduction: Basic Elements of Python, Python character set, Python tokens (keyword, identifier, literal, operator, punctuator), variables, concept of l-value and r-value, use of comments, Knowledge of data types: Number(integer, floating point, complex), boolean, sequence(string, list, tuple), None, Mapping(dictionary), mutable and immutable data types.Operators: arithmetic operators, relational operators, logical operators, assignment operators, augmented assignment operators, identity operators (is, is not), membership operators (in not in)Expressions, statement, type conversion, and input/output: precedence of operators, expression, and evaluation of an expression, type-conversion (explicit and implicit conversion), accepting data as input from the console and displaying output.

UNIT-II

Flow of Control: introduction, use of indentation, sequential flow, conditional and iterative flow; Conditional statements: if, if-else, if-elif-else, flowcharts, simple programs: e.g.:

absolute value, sort 3 numbers and divisibility of a number. Iterative Statement: for loop,range(), while loop, flowcharts, break and continue statements, nested loops, suggested programs: generating pattern, summation of series, finding the factorial of a positive number, etc.Strings: introduction, string operations (concatenation, repetition, membership and slicing), traversing a string using loops, built-in functions/methods.

UNIT-III

Lists: introduction, indexing, list operations (concatenation, repetition, membership and slicing), traversing a list using loops, built-in functions/methods, nested lists. **Tuples:** introduction, indexing, tuple operations (concatenation, repetition, membership and slicing); built-in functions. Set: creating set, changing/ adding elements to a set, removing elements to a set, python set operations, python set methods. **Dictionary:** introduction, accessing items in a dictionary using keys, mutability of a dictionary (adding a new term, modifying an existing item), traversing a dictionary, built-in functions/methods. Introduction to Python modules: Importing module using 'import <module>' and using from statement, importing math Module**. String Manipulation:** Basic Operations, Slicing

UNIT-IV

Python functions, types of functions, function definition, function call, types of function arguments, pas by value and pass by reference/object reference, recursion, advantages and disadvantages of recursion, scope and lifetime variables. Python Modules, Python Package.

UNIT-V

Python Files: Python File Operation, Python Directory , Python Exception Handling, python built-in exception, Try, Except and Finally, Python user defined exception, Python graph plotting using Matplotlib.

Course outcomes: *After successful completion of the course, the students will be able to*

1. *Demonstrate a solid understanding of Python syntax, including variables, data types, control structures, functions, classes, and modules.*
2. *Develop problem-solving skills and the ability to design, implement, and debug Python programs to solve a variety of computational problems.*
3. *Demonstrate the ability to read from and write to files, process data from external sources, and handle input/output operations using Python.*

Text Books

1. John V. Guttag, Introduction to Computation and Programming Using Python, 2nd Edition, PHI 2 Core, 2016
2. R. Nageswara Rao, Python Programming, dreamtech Press, 2nd Edition, 2018.

Reference Books

3. Ramsey Hamilton, Python: Programming: A Beginner's Guide, Create space Independent Pub, 2nd Edition, 2016
4. YashavantKanetkar, Let Us Python, BPB Publications, 5th Edition, 2022
5. R.S. Salaria, Programming in Python, Khanna Publishing House, 2nd Edition, 2019.
6. P. Sharma, Programming in Python, BPB, 2nd Edition, 2014
7. T. Budd, Exploring Python, TMH, 1st Edition, 2011

LAB: PART B: Python Programming (Practical): 30 Hours. (Practical/Project/Field work): Total marks: 30 (ESE+CCA)

Pass marks: 12 (ESE+CCA)

This part provides the practical knowledge of Programming with Python.

***Course Objectives*:**

1. *Provide students with hands-on experience in writing Python code.*
2. *Allows practicing programming concepts, syntax, and techniques in a real-world coding environment.*
3. *Help students understand and apply fundamental programming concepts such as variables, data types, control structures (loops and conditionals), functions, and input/output operations in Python.*
4. *Introduce students to various Python libraries and modules that extend the functionality of Python. Students learn how to leverage these libraries to solve complex problems and perform tasks such as data manipulation, visualization, and scientific computing.*

This paper provides practical knowledge of python programming. List of laboratory programming assignments (not limited to these)**:**

1. Using a loop, print a table of Celsius/Fahrenheit equivalences. Let c be the Celsius temperatures ranging from 0 to 100, for each value of c, print the corresponding Fahrenheit temperature.
2. Using a while loop, produce a table of sins, cosines and tangents. Make a variable x in range from 0 to 10 in steps of 0.2. For each value of x, print the value of sin(x), cos(x) and tan(x).
3. Write a program that reads an integer value and prints “leap year” or “not a leap year”.
4. Write a program that takes a positive integer n and then produces n lines of output shown as follows.
5. For example enter a size: 5
   *
   **
   ***
   ****
   *****
6. Write a function that takes an integer ‘n’ as input and calculates the value of
   1 + 1/1! + 1/2! + 1/3! + … + 1/n
7. Write a function that takes an integer input and calculates the factorial of that number.
8. Write a function that takes a string input and checks if it’s a palindrome or not.
9. Write a list function to convert a string into a list, as in list (‘abc’) gives [a, b, c].
10. Write a program to generate Fibonacci series.
11. Write a program to check whether the input number is even or odd.

12. Write a program to compare three numbers and print the largest one.
13. Write a program to print factors of a given number.
14. Write a method to calculate GCD of two numbers.
15. Write a program to sort a list using insertion sort and bubble sort and selection sort.
16. Problems related to File handling, exception handling in python, usage of user defined exceptions and assertions.
17. Problems related to string manipulations, basic operations , slicing.
18. Write a program to draw a line, bar, histograms, pie chart etc.

Course outcomes: *After successful completion of the course, the students will be able to*

1. *Students should develop a strong foundation in Python programming language, including syntax, data types, control structures, functions, and file handling.*
2. *Students should be able to analyze problems, design algorithms, and implement efficient solutions using Python.*
3. *Students should be capable of developing small-scale applications using Python.*
4. *Students should develop skills in identifying and resolving errors in Python code. They should be proficient in using debugging tools and techniques to identify and fix issues, ensuring the correctness and reliability of their programs.*

Syllabi of Computer Science IDC Courses

Semester	**: I**
Course Type	**: IDC**
Course Code	**: CSCIDC101**
Name of the Course	**: Computer Fundamentals and Applications**
Learning level	**: Foundation or Introductory Course**
Credits	**: 3**
Contact Hours	**: 45**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives:

1. To introduce the basic components of a computer and software
2. To provide overview of number system, operating system and computer language
3. To provide overview and use of Internet and online services.

UNIT I

Introduction to Computer: Computer Definition, Characteristics of Computers, Evolution of Computers & its applications, Types of Computers, Basic Organization of a Digital Computer, Hardware and Software, Central Processing Unit, Input devices, Output devices, Application Software, Systems Software, Utility Software, Open source and Proprietary Software, Mobile Apps.

UNIT II

Data Representation:Number systems and character representation, Binary, Octal, Decimal and Hexadecimal numbers.

Operating System: Basics of Operating System, Functions of Operating System.

Computer Language: Definition, Types of Languages, Language Processors: Assembler, Interpreter, Compiler, Linker and Loader; Algorithm and flowchart.

UNIT III

Memory: Primary, secondary, auxiliary memory, RAM, ROM, cache memory, harddisks, optical disks.

UNIT IV

IT Tools overview: Word Processing Basics, features of word processor, clipboard, font, page and paragraph formatting, table creation, page setup and spelling and grammar. Spreadsheet concept, Elements of Spreadsheet, Creating of Spreadsheet, cell address, formula bar, formulas and chart. Creation of Slides, Inserting & Editing Text on Slides, Slide transition and Animation.

UNIT V

Overview of Internet and IT Enabled Services: Internet, WWW, URL, E-mail, Using E-mails: Opening, Creating and Sending, forwarding and Replying to an E-mail message, Introduction to Blogs, Basics of E-commerce, Overview of e-Governance Services, e-

Governance Services on Mobile Using "UMANG APP", Digital Locker. Digital Financial Tools, eWallet, PoS, Internet Banking.

Course Outcomes: *After successful completion of the course, the students will be able to*

1. Describe the hardware, software and components of a digital computer
2. Explain number systems, functions of operating systems and language processors.
3. Create and use of email, e-Governance, and financial services

Text Books:

1. Pradeep K. Sinha and Priti Sinha, **Computer Fundamentals**, BPB Publication, 8th Edition, 2018.
2. V. Rajaraman, **Introduction to Information Technology**, PHI Learning; 3rd edition, 2018
3. Anita Goel, **Computer Fundamentals**, Pearson Education India; First Edition, 2010

Reference Books:

1. David Riley and Kenny Hunt, **Computational Thinking for Modern Solver**, Chapman and Hall/CRC; 1st edition, 2014.
2. Glenn Brookshear, **Computer Science: An Overview**, Pearson Education; Twelfth edition, 2017.
3. Puneet Kumar, Sushil Bhardwaj, *et al.*, **Introduction to Information Technology**, Kalyani Publishers; 2018th edition, 2018.

Semester	**: II**
Course Type	**: IDC**
Course Code	**: CSCIDC151**
Name of the Course	**: Programming Fundamentals with C**
Learning level	**: Foundation or Introductory Course**
Credits	**: 3**
Contact Hours	**: 45**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives:

1. Write algorithms, flowcharts and programs.
2. Implement different programming constructs and decomposition of problems into functions.
3. Use and implement data structures like arrays to obtain solutions.

UNIT I

Introduction to Programming: Computer Programs, Natural Language vs Programming Language, Concepts of Machine Level, Assembly Level and High-level Programming Language, Compiler, Interpreter.

Programming terms: Source Code, Target Code, Input, Output, Compiling, Warning, Running, Debugging, Testing.

Errors: Errors in Computer Programs, Different types of Errors in Computer Programs.

UNIT II

Introduction to Computing: Art of Programming through Algorithms and Flowcharts, Qualities of Good Algorithm, Flowchart Symbols, Rules for designing Flowchart.

Overview of C Programming: History and importance of C, Basic structure of C program, executing a C program.

UNIT III

Constants, Variable and Data Types: Introduction, Character Set, C Tokens, Keywords and Identifiers, Constants, Variables, Data Types, Declaration of Variables, Assigning Values to Variables, Defining Symbolic Constants.

Operators and Expressions: Introduction, Arithmetic Operators, Relational Operators, Logical Operators, Assignment Operators, Increment and Decrement Operators, Conditional Operator, Arithmetic Expressions.

UNIT IV

Decision Making and Branching: Introduction, Decision Making with Simple IF Statement, IF-ELSE Statement, Nested of IF-ELSE Statements, ELSE IF Ladder, Switch statement.

Looping: Introduction, while Statement, do while statement, for statement.

UNIT V

Arrays: One-dimensional Arrays, Declaration of One-dimensional Arrays, Initialization of One-dimensional Arrays.

User-defined Functions: Need for functions, Elements of User-defined Functions, Definition of Functions, Return Values and their Types, Function Calls, Function Declaration, Category of Functions, No Arguments and no Return Values, Arguments but no Return values, Arguments with Return Values, No Arguments but Returns a Value, Passing Arrays to Functions, Recursion.

Pointers: Introduction to pointers in C (basic Concepts)

Course Outcomes:*After successful completion of the course, the students will be able to*

1. Demonstrate problem solving skills by developing and implementing algorithms to solve problems.
2. Apply appropriate Control structures to solve problems.
3. Describe the concept of Arrays.
4. Write user defined functions and apply the concept of function to solve problems.

Text Books:

1. Brian W. Kernighan and Denis M. Ritchie, **The C Programming Language**, Pearson Education India; 2nd edition, 2015.
2. Byron S Gottfried, **Programming with C**, McGraw Hill Education; 4th Edition, 2018.
3. E. Balagurusamy, **Programming in ANSI C**, McGraw Hill Education; Eighth edition, 2019
4. V. Rajaraman, **Computer Programming in C**, PHI Learning Private Limited; Second edition, 2019

Reference Books:

1. Yashavant P. Kanetkar, **Let Us C**, BPB; 16th edition, 2017.
2. Kamthane, **Programming in C**, Pearson Education India; 3rd edition, 2015.

Assam University, Silchar

Four Year Undergraduate Programme

Implemented under NEP 2020

Effective from the Academic Year 2023-24

Syllabus of Computer Science
(2nd Year)

Approved in the 96th meeting of the Academic Council on 12th April 2024
vide Resolution No. AC:96:04-24:5

Semester wise list of Computer Science Discipline Specific Core (DSC) Papers

SEMESTER	COURSE CODE	TITLE OF COURSES	CREDITS
I	**CSCDSC101**	Digital Computer Fundamentals	3
	CSCDSC102	Discrete Mathematics	3
II	**CSCDSC151**	Data Structure	3
	CSCDSC152	Lab on Data Structure	3
III	**CSCDSC201**	Computer Organization and Architecture	4
	CSCDSC202	Operating Systems	4
IV	**CSCDSC251**	Object Oriented Programming with Java	4
	CSCDSC252	Database Management System	4
	CSCDSC253	Lab on Java & DBMS	4
V	**CSCDSC301**	Computer Graphics	4
	CSCDSC302	System Analysis and Design	4
	CSCDSC303	Lab on Graphics Programming	4
VI	**CSCDSC351**	Computer Network and Internet Technology	4
	CSCDSC352	Theory of Computation	4
	CSCDSC353	Microprocessor and Systems Programming	4
	CSCDSC354	Lab on Internet Technology & Microprocessor and Systems Programming	4
VII	**CSCDSC401**	Design & Analysis of Computer Algorithms	4
	CSCDSC402	Principles of Compiler Design	4
	CSCDSC403	Artificial Intelligence	4
	CSCDSC404	Lab on DACA & Compiler Design	4
VIII	**CSCDSC451**	Software Engineering	4
	CSCDSC452(A)	Image Processing	4
	CSCDSC452(B)	Data Analytics	4
	CSCDSC453	Natural Language Processing	4
	CSCDSC454(A)	IoT	4
	CSCDSC-454(B)	Cloud Computing	4
	OR		
	CSCDSC451	Research Methodology	4
	CSCDSC455	Research Project/Dissertation	12

Semester wise list of Computer Science Discipline Specific Minor (DSM) Papers

SEMESTER	COURSE CODE	TITLE OF COURSES	CREDITS	DSM1/DSM2
I	**CSCDSM101**	Programming with C	3	**DSM1**
II	**CSCDSM151**	Programming with C	3	**DSM2**
III	**CSCDSM201**	Database Management System	4	**DSM1**
IV	**CSCDSM251**	Lab on C & DBMS	3	**DSM1**
	CSCDSM252	Database Management System	3	**DSM2**
V	**CSCDSM301**	Operating Systems	3	**DSM1**
	CSCDSM302	Operating Systems	3	**DSM2**
VI	**CSCDSM351**	Lab on C & DBMS	4	**DSM2**
VII	**CSCDSM401**	Internet Technologies	4	**DSM1**
VIII	**CSCDSM451**	Internet Technologies	4	**DSM2**

Semester wise list of Computer Science Skill Enhancement Course (SEC) Papers

SEMESTER	COURSE CODE	TITLE OF COURSES	CREDITS
I	**CSCSEC101**	Programming with C	3
II	**CSCSEC151**	Python Programming	3
III	**CSCSEC201**	Programming with C++ & Lab on OS and C++	3

Semester wise list of Computer Science Interdisciplinary Course (IDC) Papers

SEMESTER	COURSE CODE	TITLE OF COURSES	CREDITS
I	**CSCIDC101**	Computer Fundamentals & Applications	3
II	**CSCIDC151**	Programming Fundamentals with C	3
III	**CSCIDC201**	Introduction to Web Designing & Cyber Security	3

Syllabi of Computer Science DSC Courses

Semester	**: III**
Course Type	**: DSC**
Course Code	**: CSCDSC201**
Name of the Course	**: Computer Organization and Architecture**
Learning level	**: Intermediate-level course**
Credits	**: 4**
Contact Hours	**: 60**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives: The course objective is to

1. *Students should gain knowledge about the various components of a computer system.*
2. *Learn about different types of instruction sets (e.g., CISC, RISC) and understand how instructions are executed by the CPU.*
3. *Introducing assembly language programming and its relationship to computer architecture.*
4. *Exploring the memory hierarchy, including registers, cache memory, main memory (RAM), and secondary storage.*
5. *Studying the organization and design principles of processors, including pipelining, parallelism, and micro architecture.*
6. *Understanding how input/output devices are interfaced with the CPU and memory, including concepts such as I/O controllers, interrupts, and DMA.*

Unit-I

Register transfer and Micro operation: Register transfer language, bus & memory transfer, Arithmetic; Logic; Shift Micro operation , Arithmetic Logic unit. Computer organization design- Computer Instruction with details like timing; control and Instruction Cycle, memory Reference; Input and output and Interrupt Instructions, Design of a Basic Computer.

Unit-II

Programming the Basic Computer: Machine and Assembly Language with Programming Details. Microprogrammed Control – Control memory Address sequencing, Microprogram Examples, Design of control unit.

Unit-III

Central processing Unit: General Register organization, Stack organization, Instruction format, addressing modes, Data transfer and Manipulation, Program control, Length and type of instruction, RISC and CISC.

Unit-IV

Computer Arithmetic: Addition; Subtraction; Multiplication; Division Algorithms with hardware implementation, Floating point and Decimal Arithmetic Operations**.**

Unit-V

Input output Organization: Input Output Interface, Asynchronous Data Transfer, Modes of Transfer, Priority Interrupt, DMA, Input Output Processor, Serial Communication. Advances in Memory system- Memory Hierarchy, Different Memory Organization (Main, Auxiliary, Associative, and Cache) and Virtual Memory**.**

***Course Learning Outcomes**: After successful completion of the course, the students will be able to:*

1. *Understand Fundamental Concepts of computer organization and architecture.*
2. *Analyze and Evaluate Architectural Designs.*
3. *Apply Assembly Language Programming and understanding how high-level language constructs map to machine instructions and memory organization.*
4. *Design and optimize computer systems for performance.*
5. *Analyze Real-World Architectures*

Text Books:

1. M. M. Mano, **Computer System Architecture**, Pearson Education Asia, 3rd Edition, 2015.
2. V. Carl Hamacher, Zvonko G. Vranesic, Safwat G. Zaky, **Computer organization**, McGraw Hill, 5th Edition,2010.

Reference Books:

1. PVS Rao, **Perspectives in Computer Architecture**, PHI, 2nd Edition, 2005.

Semester	**: III**
Course Type	**: DSC**
Course Code	**: CSCDSC202**
Name of the Course	**: Operating System**
Learning level	**: Intermediate-level course**
Credits	**: 4**
Contact Hours	**: 60**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives: The course objective is to

1. *Understand the fundamental concepts of operating systems, including processes, threads, scheduling, etc.*
2. *Learn about the structure and components of operating systems, such as the kernel, device drivers, system calls, and user interface layers.*
3. *Explore process management concepts, including process creation, scheduling algorithms, inter-process communication, synchronization mechanisms, and deadlock handling.*
4. *Understand memory management techniques, including virtual memory, paging, segmentation, memory allocation algorithms*
5. *Explore operating system security mechanisms, including authentication, authorization, access control, encryption, and security policies*

Unit I

Introduction: Basic OS functions, resource abstraction, types of operating systems– multiprogramming systems, batch systems , time sharing systems; operating systems for personal Computers & workstations, process control & real time systems.

Operating System Organization: Processor and user modes, kernels, system calls and

system programs.

Unit II

Process Management: System view of the process and resources, process abstraction, process hierarchy, threads, threading issues, thread libraries; Process Scheduling, non-pre-emptive and preemptive scheduling algorithms.

Unit III

Process Coordination: Synchronization, concurrent processes, critical section, semaphores, methods for inter-process communication; deadlocks.

Unit IV

Memory Management: Physical and virtual address space; memory allocation strategies –fixed and variable partitions, paging, segmentation, virtual memory.

Unit V

File and I/O Management: Directory structures, file operations, file allocation methods, device management.
Protection and Security: Policy mechanism, Authentication, Internal access Authorization.

Course Learning Outcomes*: After successful completion of the course, the students will be able to:*

1. *Understanding of Operating System Concepts including processes, threads, memory management, file systems, and I/O operations.*
2. *Skills to analyze and solve problems related to process scheduling, resource allocation, etc.*
3. *Develop the ability to implement basic operating system functionalities such as process management, memory management, and file system operations.*
4. *Proficiency in file system organization, file I/O operations, directory management, and file allocation methods.*

Text Books:

1. A Silberschatz, P.B. Galvin, G. Gagne, **Operating Systems Concepts**, John Wiley Publications. 8th Edition, 2008.
2. A.S. Tanenbaum, **Modern Operating Systems**, Pearson Education, 3rdEdition, 2007.

Reference Books:

1. G. Nutt, Operating Systems: A Modern Perspective, Pearson Education, 2ndEdition 1997.
2. W. Stallings, Operating Systems, Internals & Design Principles, PHI, 5th Edition, 2008.

Semester	**: IV**
Course Type	**: DSC**
Course Code	**: CSCDSC251**
Name of the Course	**: Object Oriented Programming with Java**
Learning level	**: Intermediate-level course**
Credits	**: 4**
Contact Hours	**: 60**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives: The course objective is to

1. *Understand the basics of Java programming language syntax, including variables, data types, operators, and expressions.*
2. *Learn the principles of object-oriented programming, including classes, objects, inheritance, polymorphism, and encapsulation*
3. *Learn how to handle exceptions in Java using try-catch blocks, and understand the concept of checked and unchecked exceptions*
4. *Learn how to perform input and output operations in Java, including file handling, reading from/writing to files, and using streams.*
5. *Gain an introduction to GUI development in Java using Swing or JavaFX, including creating windows, panels, buttons, and event handling*
6. *Learn how to connect Java applications to databases using JDBC (Java Database Connectivity), and perform database operations such as querying and updating data.*

Unit I

Introduction to Java : Java Architecture and Features, Understanding the semantic and syntax differences between C++ and Java, Compiling and Executing a Java Program, Variables, Constants, Keywords Data Types, Operators (Arithmetic, Logical and Bitwise) and Expressions, Comments, Doing Basic Program Output, Decision Making Constructs (conditional statements and loops) and Nesting, Java Methods (Defining, Scope, Passing and Returning Arguments, Type Conversion and Type and Checking, Built-in Java Class Methods)

Unit II

Arrays, Strings and I/O: Creating & Using Arrays (One Dimensional and Multidimensional), Referencing Arrays Dynamically, Java Strings: The Java String class, Creating & Using String Objects, Manipulating Strings, String Immutability & Equality, Passing Strings To & From Methods, String Buffer Classes. Simple I/O using System.out and the Scanner class, Byte and Character streams, Reading/Writing from console and files.
Object-Oriented Programming Overview: Principles of Object-Oriented Programming, Defining & Using Classes, Controlling Access to Class Members, Class Constructors, Method Overloading, Class Variables & Methods, Objects as parameters, final classes, Object class, Garbage Collection.

Unit III

Inheritance, Interfaces, Packages, Enumerations, Autoboxing and Metadata:

Inheritance: (Single Level and Multilevel, Method Overriding, Dynamic Method Dispatch, Abstract Classes), Interfaces and Packages, Extending interfaces and packages, Package and Class Visibility, Using Standard Java Packages (util, lang, io, net),Wrapper Classes, Autoboxing/Unboxing, Enumerations and Metadata.

Unit IV

Exception Handling, Threading, Networking and Database Connectivity: Exception types, try, catch, nested try, finally, uncaught exceptions, throw, built-in exceptions, Creating your own exceptions; Multi-threading: The Thread class and Runnable interface, creating single and multiple threads, Thread, Accessing and manipulating databases using JDBC.

Unit V

Applets and Event Handling: Java Applets: Introduction to Applets, Writing Java Applets, Working with Graphics, Incorporating Images & Sounds. Event Handling Mechanisms, Listener Interfaces, Adapter and Inner Classes. The design and Implementation of GUIs using the AWT controls, Swing components of Java Foundation Classes such as labels, buttons, list, choices, text fields, layout managers, menus, events and listeners; Graphic objects for drawing figures such as lines, rectangles, ovals, using different fonts. Overview of servlets.

Course Learning Outcomes*: After successful completion of the course, the students will be able to:*

1. *Demonstrate a solid understanding of fundamental programming concepts.*
2. *Apply object-oriented programming principles*
3. *Understand and apply the syntax and features of the Java programming language, including packages, access modifiers, interfaces, etc.*
4. *Demonstrate proficiency in handling exceptions in Java programs using try-catch blocks, throwing and catching exceptions*
5. *Design and develop graphical user interfaces (GUIs) using Java Swing or JavaFX, including creating windows, panels, buttons, text fields, and event handling*

Text Books:

1. E. Balaguruswamy, **"Programming with Java"**, 6th Edition, McGraw Hill.2019.
2. Ken Arnold, James Gosling, David Homes, **"The Java Programming Language"**, 4thEdition, 2005.
3. James Gosling, Bill Joy, Guy L Steele Jr, Gilad Bracha, Alex Buckley, **"The Java Language Specification, Java SE 8 Edition (Java Series)**", Published by Addison Wesley, 2014.

Reference Books:

1. Bruce Eckel, **"Thinking in Java"**, 3rd Edition, PHI, 2002.
2. Paul Deitel, Harvey Deitel, **"Java: How to Program"**, 11th Edition, Pearson, 2018.
3. Joshua Bloch, **"Effective Java"**, Addison-Wesley, 2nd Edition, 2008.

Semester	**: IV**
Course Type	**: DSC**
Course Code	**: CSCDSC252**
Name of the Course	**: Database Management System**
Learning level	**: Intermediate-level course**
Credits	**: 4**
Contact Hours	**: 60**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives: The course objective is to

1. *Understand the fundamentals of databases, including definitions, types of databases and their applications in various domains*
2. *Learn data modeling techniques such as Entity-Relationship Diagrams (ERDs) and normalization to design efficient and scalable database schemas.*
3. *Understand the principles of relational databases, including tables, rows, columns, keys (primary, foreign), relationships, and constraints.*
4. *Gain proficiency in SQL for database querying, data manipulation (insertion, deletion, modification), data definition (creating tables, indexes) etc.*
5. *Learn about transaction properties (ACID), concurrency control mechanisms (locking, timestamps), and recovery techniques (undo, redo, logging) to ensure data consistency and reliability.*
6. *Understand database security principles, including authentication, authorization, encryption, and auditing, to protect sensitive data.*

Unit I

Introduction: Characteristics of database approach, data models, database system architecture and data independence.
Entity Relationship(ER) Modeling: Entity types, relationships, constraints.

Unit II

Relation data model: Relational model concepts, relational constraints, relational algebra, SQL
Queries.

Unit III

Database design: Mapping ER/EER model to relational database, functional dependencies, Lossless decomposition, Normal forms (upto BCNF).

Unit IV

Transaction Processing: ACID properties, concurrency control, Locking protocols, Deadlock detection and prevention.

Unit V

File Structure and Indexing: Operations on files, File of Unordered and ordered records, overview of File organizations, Indexing structures for files (Primary index, secondary index, clustering index), Multilevel indexing using B and B+ trees.

***Course Learning Outcomes**: After successful completion of the course, the students will be able to:*

1. *Demonstrate a solid understanding of fundamental database concepts, including data models, schemas, keys, relationships, and database management system architectures*
2. *Demonstrate proficiency in SQL (Structured Query Language) for database querying, data manipulation, data definition, and data control operations on relational databases.*
3. *Apply database management concepts and techniques to analyze, design, and implement solutions for real-world database problems and scenarios*
4. *Evaluate and critique database designs, implementations, and performance optimizations, and propose improvements*

Text Books:

1. R. Elmasri, S.B. Navathe, **Fundamentals of Database Systems,** Pearson Education, 6th Edition, 2010.
2. A. Silberschatz, H.F. Korth, S. Sudarshan, **Database System Concepts,** McGrawHill, 6th Edition,2010.

Reference Books:

1. C. J. Date, **An Introduction to Database Systems**, Pearson India, 8th edition, 2005.
2. R. Ramakrishanan, J. Gehrke, **Database Management Systems,** 3rd Edition, McGraw Hill,2002.

Semester	**: IV**
Course Type	**: DSC**
Course Code	**: CSCDSC253**
Name of the Course	**: Lab on Java & DBMS**
Learning level	**: Intermediate-level course**
Credits	**: 4**
Contact Hours	**: 120**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives:

1. *Java Programming Skills Development.*
2. *Object-Oriented Design and Implementation.*
3. *Data Structure and Algorithm Implementation.*
4. *Graphical User Interface (GUI) Development.*
5. *Database Application Development*
6. *SQL Querying and Data Manipulation*
7. *Database Design and Implementation*
8. *Database Connectivity and Interaction*

This paper provides practical knowledge of Java Programming and SQL queries. List of laboratory programming assignments (not limited to these):

Lab on Java

1. To find the sum of any number of integers entered as command line arguments
2. To find the factorial of a given number
3. To learn use of a single dimensional array by defining the array dynamically.
4. To learn use of length in case of a two dimensional array
5. To convert a decimal to binary number
6. To check if a number is prime or not, by taking the number as input from the keyboard.
7. To find the sum of any number of integers interactively, i.e., entering every number from the keyboard, whereas the total number of integers is given as a command line argument.
8. Write a program that shows working of different functions of String and StringBufferClasss likesetCharAt(), setLength(), append(), insert(), concat()and equals().
9. Write a program to create a distance class with methods where distance is computed in terms of feet and inches, how to create objects of a class and to see the use of this pointer.
10. Modify the distance class by creating a constructor for assigning values (feet and inches) to the distance object. Create another object and assign a second object as reference variable to another object reference variable. Further create a third object which is a clone of the first object.
11. Write a program to show that during function overloading, if no matching argument is found,then java will apply automatic type conversions(from lower to higher data type).
12. Write a program to show the difference between public and private access specifiers. The Program should also show that primitive data types are passed by value and objects are passed by reference and to learn use of final keywords.
13. Write a program to show the use of static functions and to pass variable length arguments in function.
14. Write a program to demonstrate the concept of boxing and unboxing.
15. Create a multi-file program where in one file a string message is taken as input from the userand the function to display the message on the screen is given in another file (make use ofScanner package in this program).
16. Write a program to create a multilevel package and also creates a reusable class to generateFibonacci series, where the function to generate Fibonacci series is given in a different file belonging to the same package.
17. Write a program that illustrates different levels of protection in classes/subclasses belonging to the same package or different packages.
18. Write a program DivideByZero that takes two numbers a and b as input, computes a/b,and invokes Arithmetic Exception to generate a message when the denominator is zero.
19. Write a program to show the use of nested try statements that emphasizes the sequence of checking for catch handler statements.
20. Write a program to create your own exception types to handle situations specific to your application (Hint: Define a subclass of Exception which itself is a subclass of Throwable).
21. Write a program to demonstrate priorities among multiple threads.

22. Write a program to demonstrate multi thread communication by implementing synchronization among threads (Hint: you can implement a simple producer and consumer problem).
23. Write a program to demonstrate different mouse handling events like mouseClicked(),mouseEntered(), mouseExited(), mousePressed, mouseReleased() and mouseDragged().
24. Write a program to demonstrate different keyboard handling events.
25. Write a program using JDBC to perform: a) insert b) delete c) update and d) search operations.

Lab on DBMS

1. Implementation of DDL commands of SQL with suitable examples a) Create table b) Alter table c) Drop Table.
2. Implementation of DML commands of SQL with suitable examples a) Select b) Insert c) Update d) Delete.
3. Implementation of different types of function with suitable examples a) Number function b) Aggregate Function c) Character Function d) Conversion Function e) Date Function.
4. Implementation of different types of operators in SQL a) Arithmetic Operators b) Logical Operators c) Comparison Operator d) Special Operator e) Set Operation.
5. Implementation of different types of Joins a) Inner Join b) Outer Join c) Natural Join etc.
6. Study and Implementation of a) Group By & having clause b) Order by clause c) Indexing.
7. Study & Implementation of a) Sub queries b) Views.
8. Study & Implementation of different types of constraints.
9. Study & Implementation of Database Backup & Recovery commands. Study & Implementation of Rollback, Commit, Savepoint.
10. Creating Database /Table Space a) Managing Users: Create User, Delete User b) Managing roles:-Grant, Revoke.

Course Learning Outcomes: After successful completion of the course, the students will be able to.

1. *Demonstrate proficiency in Java programming by implementing various programming tasks, exercises, and projects using Java language features and libraries.*
2. *Design and develop graphical user interfaces (GUIs) for Java applications using Swing including creating interactive components and event handling.*
3. *Integrate Java applications with relational databases using JDBC (Java Database Connectivity) to perform database operations such as querying, insertion, deletion, and modification.*
4. *Write SQL queries to retrieve, update, delete, and manipulate data stored in relational databases.*
5. *Establish database connections from Java applications, handle database transactions, manage database resources, and implement error handling and exception management strategies.*

Syllabi of Computer Science DSM Courses

Semester	**: III**
Course Type	**: DSM**
Course Code	**: CSCDSM201**
Name of the Course	**: Database Management System**
Learning level	**: Intermediate-level course**
Credits	**: 4**
Contact Hours	**: 60**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives: The course objective is to

1. *Understand the fundamentals of databases, including definitions, types of databases and their applications in various domains*
2. *Learn data modeling techniques such as Entity-Relationship Diagrams (ERDs) and normalization to design efficient and scalable database schemas.*
3. *Understand the principles of relational databases, including tables, rows, columns, keys (primary, foreign), relationships, and constraints.*
4. *Gain proficiency in SQL for database querying, data manipulation (insertion, deletion, modification), data definition (creating tables, indexes) etc.*
5. *Learn about transaction properties (ACID), concurrency control mechanisms (locking, timestamps), and recovery techniques (undo, redo, logging) to ensure data consistency and reliability.*
6. *Understand database security principles, including authentication, authorization, encryption, and auditing, to protect sensitive data.*

Unit I

Introduction: Characteristics of database approach, data models, database system architecture and data independence.
Entity Relationship(ER) Modeling: Entity types, relationships, constraints.

Unit II

Relation data model: Relational model concepts, relational constraints, relational algebra, SQL Queries.

Unit III

Database design: Mapping ER/EER model to relational database, functional dependencies, Lossless decomposition, Normal forms (upto BCNF).

Unit IV

Transaction Processing: ACID properties, concurrency control, Locking protocols, Deadlock detection and prevention.

Unit V

File Structure and Indexing: Operations on files, File of Unordered and ordered records, overview of File organizations, Indexing structures for files (Primary index, secondary index, clustering index), Multilevel indexing using B and B+ trees.

***Course Learning Outcomes**: After successful completion of the course, the students will be able to:*

5. *Demonstrate a solid understanding of fundamental database concepts, including data models, schemas, keys, relationships, and database management system architectures*
6. *Demonstrate proficiency in SQL (Structured Query Language) for database querying, data manipulation, data definition, and data control operations on relational databases.*
7. *Apply database management concepts and techniques to analyze, design, and implement solutions for real-world database problems and scenarios*
8. *Evaluate and critique database designs, implementations, and performance optimizations, and propose improvements*

Text Books:

3. R. Elmasri, S.B. Navathe, **Fundamentals of Database Systems,** Pearson Education, 6th Edition, 2010.
4. A. Silberschatz, H.F. Korth, S. Sudarshan, **Database System Concepts,** McGrawHill, 6thEdition,2010.

Reference Books:

3. C. J. Date, **An Introduction to Database Systems**, Pearson India, 8th edition, 2005.
4. R. Ramakrishanan, J. Gehrke, **Database Management Systems,** 3rd Edition, McGraw Hill,2002.

Semester : IV
Course Type : DSM
Course Code : CSCDSM251
Name of the Course : Lab on C & DBMS
Learning level : Intermediate-level course
Credits : 3
Contact Hours : 90
Total Marks : 100
End Semester Marks : 70
Internal Marks : 30

Course objective: To provide students with practical skills and experiences in C programming and DBMS application

1. *Develop proficiency in the C programming language by implementing various programming tasks.*
2. *Understand memory management concepts in C, including dynamic memory allocation.*
3. *Gain a basic understanding of Database Management System (DBMS) concepts.*
4. *Understand the basics of Structured Query Language (SQL) for database querying and manipulation, including SELECT, INSERT, UPDATE, DELETE statements, and simple SQL queries.*

Lab on C

Problem solving of various natures by implementing programs in C Programming languages

based on unit wise contents of the theory paper Programming with C**.** Following are some programming tasks for laboratory programming assignments but the assignments should not be limited to these only.

*List of laboratory programming assignments (not limited to these)**:***

1. Write a program to

   a) Produce ASCII equivalent of given number

   b) Find divisor or factorial of a given number

   c) Evaluate the following algebraic expressions after reading necessary values from the user (ax+b)/ (ax-b) -2.5 log x-cos 30+|x2-y2|+sqrt (2xy) - (x5+10x4+8x3+4x+2)

   d) Find sum of a geometric series

   e) Cipher a string

   f) Check whether a given string follows English capitalization rules

   g) Find sum of the following series 1+ ½ + 1/3 +_____+1/20

   h) Search whether a given substring exist in an input string or not and then delete this string from input string

2. Write a recursive program for tower of Hanoi problem.
3. The Fibonacci sequence of numbers is 1, 1, 2, 3, 5, 8……. Based on the recurrence relation F(*n*)=F(*n*-1)+F(*n*-2) for n>2 Write a recursive program to print the first *n* Fibonacci number.
4. Write a menu driven program for matrices to do the following operation depending on whether the operation requires one or two matrices
   a) Addition of two matrices
   b) Subtraction of two matrices
   c) Finding upper and lower triangular matrices
   d) Trace of a matrix
   e) Transpose of a matrix
   f) Check of matrix symmetry
   g) Product of two matrices.
5. Write a program that takes two operands and one operator from the user perform the operation and then print the answer
7. Write functions to add, subtract, multiply and divide two complex numbers (x+iy) and (a+ib) Also write the main program.
8. Write a menu driven program for searching an sorting with following options:-

a) Searching (1) Linear searching (2) Binary searching

b) Sorting (1) Insertion sort (2) Selection sort

9. Write a program to copy one file to other, use command line arguments.
10. Write a program to mask some bit of a number (using bit operations)
11. An array of record contains information of managers and workers of a company.

Print all the data of managers and workers in separate files.

Lab on DBMS

1. Implementation of DDL commands of SQL with suitable examples a) Create table b) Alter table c) Drop Table.
2. Implementation of DML commands of SQL with suitable examples a) Select b) Insert c) Update d) Delete.
3. Implementation of different types of function with suitable examples a) Number function b) Aggregate Function c) Character Function d) Conversion Function e) Date Function.
4. Implementation of different types of operators in SQL a) Arithmetic Operators b) Logical Operators c) Comparison Operator d) Special Operator e) Set Operation.
5. Implementation of different types of Joins a) Inner Join b) Outer Join c) Natural Join etc.
6. Study and Implementation of a) Group By & having clause b) Order by clause c) Indexing.
7. Study & Implementation of a) Sub queries b) Views.
8. Study & Implementation of different types of constraints.
9. Study & Implementation of Database Backup & Recovery commands. Study & Implementation of Rollback, Commit, Savepoint.
10. Creating Database /Table Space a) Managing Users: Create User, Delete User b) Managing roles:-Grant, Revoke.

Course Learning Outcomes: After successful completion of the course, the students will be able to.

1. *Demonstrate proficiency in programming using the C language, including understanding of syntax, data types, control structures, functions, and pointers.*
2. *Utilize file handling techniques in C for reading from and writing to files, including text and binary files, and perform input/output operations effectively.*
3. *Understand memory management concepts in C, including dynamic memory allocation*
4. *Understand the basics of Structured Query Language (SQL) for database querying and manipulation, including SELECT, INSERT, UPDATE, DELETE statements, and simple SQL queries.*

Semester	**: IV**
Course Type	**: DSM**
Course Code	**: CSCDSM252**
Name of the Course	**: Database Management System**
Learning level	**: Intermediate-level course**
Credits	**: 3**
Contact Hours	**: 45**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives: The course objective is to

1. *Understand the fundamentals of databases, including definitions, types of databases and their applications in various domains*

2. *Learn data modeling techniques such as Entity-Relationship Diagrams (ERDs) and normalization to design efficient and scalable database schemas.*
3. *Understand the principles of relational databases, including tables, rows, columns, keys (primary, foreign), relationships, and constraints.*
4. *Gain proficiency in SQL for database querying, data manipulation (insertion, deletion, modification), data definition (creating tables, indexes) etc.*
5. *Learn about transaction properties (ACID), concurrency control mechanisms (locking, timestamps), and recovery techniques (undo, redo, logging) to ensure data consistency and reliability.*
6. *Understand database security principles, including authentication, authorization, encryption, and auditing, to protect sensitive data.*

Unit I

Introduction: Characteristics of database approach, data models, database system architecture and data independence.
Entity Relationship(ER) Modeling: Entity types, relationships, constraints.

Unit II

Relation data model: Relational model concepts, relational constraints, relational algebra, SQL Queries.

Unit III

Database design: Mapping ER/EER model to relational database, functional dependencies, Lossless decomposition, Normal forms (upto BCNF).

Unit IV

Transaction Processing: ACID properties, concurrency control, Locking protocols, Deadlock detection and prevention.

Unit V

File Structure and Indexing: Operations on files, File of Unordered and ordered records, overview of File organizations, Indexing structures for files (Primary index, secondary index, clustering index), Multilevel indexing using B and B+ trees.

***Course Learning Outcomes**: After successful completion of the course, the students will be able to:*

1. *Demonstrate a solid understanding of fundamental database concepts, including data models, schemas, keys, relationships, and database management system architectures*
2. *Demonstrate proficiency in SQL (Structured Query Language) for database querying, data manipulation, data definition, and data control operations on relational databases.*
3. *Apply database management concepts and techniques to analyze, design, and implement solutions for real-world database problems and scenarios*
4. *Evaluate and critique database designs, implementations, and performance optimizations, and propose improvements*

Text Books:

1. R. Elmasri, S.B. Navathe, **Fundamentals of Database Systems,** Pearson Education, 6th Edition, 2010.
2. A. Silberschatz, H.F. Korth, S. Sudarshan, **Database System Concepts,** McGrawHill, 6th Edition,2010.

Reference Books:

1. C. J. Date, **An Introduction to Database Systems**, Pearson India, 8th edition, 2005.
2. R. Ramakrishanan, J. Gehrke, **Database Management Systems,** 3rd Edition, McGraw Hill,2002.

Syllabi of Computer Science SEC Courses

Semester	**: III**
Course Type	**: SEC**
Course Code	**: CSCSEC201**
Name of the Course	**: Programming with C++ & Lab on OS and C++**
Learning level	**: Intermediate-level course**
Credits	**: 3**
Contact Hours	**: 30**
Total Marks	**: 100**
End Semester Marks	**: 80 [50 (Theory) + 30 (Lab)]**
Internal Marks	**: 20**

Course Objectives:

1. *Familiarize students with the basics of C++ programming language, including variables, data types, and control structures*
2. *To explain the concepts of functions and program structure in C++*
3. *To explain the concept and working of class, object and files in C++*
4. *To introduce the concepts of object-oriented programming*
5. *Develop skills to write C++ programs and handling errors, exceptions in the programs*

UNIT-I

Introduction to C++: History of C++, Object-Orientation Programming Paradigm, Basic concepts of Object-Orientation Programming: Object, Class, Data Abstraction and Encapsulation, Inheritance, Polymorphism, Dynamic binding, Message Passing. Overview of Object-based Programming Language and Object-oriented programming Languages

UNIT-II

Beginning with C++, Tokens, Expressions and Control structures: Structure of C++ Program, A Simple C++ Program, Creating the Source File, Compiling and Linking, Tokens, Identifiers and Constants, Basic Data Types, User-Defined Data Types, Derived Data types, Symbolic Constants, Type Compatibility, Declaration of Variables, Dynamic Initialization of Variables, Reference Variables, Operators in C++, Memory Management Operators, Type Cast Operator, Expressions and Their Types, Implicit Conversion, Operator Precedence, Control Structures

UNIT-III

Function in C++, Classes and Objects, Constructors, Destructors: Utility of functions, Function Prototyping, Call by Reference, Return by Reference, Inline Functions, Friend Function, Functions parameters.

Defining & Using Classes, Class Variables & Member Functions, Memory Allocations for Objects, Constructors and Destructors, Constructor Overloading, Function overloading in classes, Objects as parameters, specifying the Protected and Private Access

UNIT-IV

Operator Overloading and Type Conversions: Defining Operator Overloading, Rules for Overloading Operators, Overloading Unary and Binary Operators, Overloading Binary Operators Using Friends, Manipulation of Strings using Operators

Inheritance-Extending Classes: Introduction, Defining Derived Classes, Single Inheritance,

Multilevel Inheritance, Multiple Inheritance, Hierarchical Inheritance, Hybrid Inheritance, Virtual Base Classes, Abstract Classes.

UNIT-V

Pointers, Virtual Functions and polymorphism and Exception Handling: Compile-Time and Run-Time Polymorphism, Pointers to Objects, *this* Pointer, Pointers to Derived Classes, Virtual Functions, Basics Exceptional Handling (using catch and throw, multiple catch statements), Catching all exceptions, Restricting exceptions, Rethrowing exceptions

Working with Files: Classes for File Stream Operation (use of fstream, ifstream, ofstream and fstream classes), Opening and closing a file, Reading and writing Text Files, Using put(), get(), read() and write() functions, Random access in files, File Pointers and their Manipulations.

.

Course Outcomes: *After successful completion of the course, the students will be able to*

1. *Learn detailed concepts of C++, including variables, data types, control structures, functions, classes, and objects.*
2. *Understand Object-Oriented Programming Paradigm*
3. *Develop problem-solving skills and the ability to implement and debug C++ programs*
4. *Demonstrate the ability to read from and write to files, use pointers, and handle input/output operations using C++.*

Text Books:

1. E Balaguruswamy, "Object Oriented Programming with C++", 8th Edition, Tata McGraw-Hill Education, 2020.
2. R. S. Salaria, “A Complete Reference to C++ Language” 1st Edition, Khanna Book Publishing Company, 2017
3. Herbtz Schildt, "C++: The Complete Reference", Fourth Edition, McGraw Hill, 2017

Reference Books:

1. Bjarne Stroustrup, "The C++ Programming Language", 4th Edition, Pearson, 2022.
2. Bjarne Stroustrup, "Programming -- Principles and Practice using C++", 2nd Edition, Addison-Wesley, 2014.
3. Paul Deitel, Harvey Deitel, "C++ How to Program", 10th Edition, Prentice Hall, Pearson, 2017.
4. John R. Hubbard, "Programming with C++", Schaum's Outlines, McGraw Hill, 2nd Edition, 2000.

Lab on OS and C++: 30 Hours. (Practical /Project/Field work):
Total marks: 30 Pass marks: 12

This part provides the practical knowledge of Programming with C++.

Course Objectives:

1. *Provide students with hands-on experience in writing C++ Programs.*
2. To *understand and apply fundamental programming concepts such as control structures, functions, and input/output operations in C++.*
3. *To gain knowledge about practical implementations of constructors and destructors.*

4. *To understand the implementation of object-oriented concepts including inheritance and polymorphism.*
5. *Provide students the practical knowledge of file handling*
6. *To gain basic knowledge about UNIX/LINUX programming*

*This paper provides practical knowledge of OS and C++ programming. List of laboratory programming assignments (not limited to these)***:**

Lab on C++

1. Write a C++ Program to Check Whether a Number is Palindrome or Not
2. Write a C++ Program to Multiply Two Matrices.
3. Write a C++ Program to Find Largest Element of an Array
4. C++ program to create a class for student to get and print details of a student.
5. C++ program to demonstrate example of friend function with class.
6. C++ program for Banking Management System using Class.
7. C++ Program to calculate Volume of Cube using constructor and destructor
8. C++ Program to determine the Area of Rectangle using constructors
9. C++ Program to enter student details by Passing parameters to constructors
10. C++ Program to demonstrate Constructor Overloading
11. C++ Program To calculate Volume of Box using Constructor
12. Create the Person class. Create some objects of this class (by taking information from the
    user). Inherit the class Person to create two classes Teacher and Student class. Maintain the respective information in the classes and create, display and delete objects of these two classes.

Lab on OS

13. Adding and managing user accounts.
14. Monitoring disk space quota and memory usage and redirect the output in a file.
15. Compression and extracting a file. Use the command line.
16. Configuring a network interface and assigning a default route.
17. Changing the ownership and access permission of file or directory. Use command line.
18. Copy, move and rename a file.
19. Configuring a ftp server
20. Assigning address of DNS.
21. Use of ssh, telnet, netstat, ping, route commands.
22. Use grep, awk, sed commands.
23. Monitoring and managing system log information.
24. Write shell script to
    i. Find factorial of a given number
    ii. To check a given number is odd or even
    iii. Convert a decimal number to hexadecimal number

Syllabi of Computer Science IDC Courses

Semester	**: III**
Course Type	**: IDC**
Course Code	**: CSCIDC201**
Name of the Course	**: Introduction to Web Designing & Cyber Security**
Learning level	**: Intermediate-level course**
Credits	**: 3**
Contact Hours	**: 45**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives:

1. To Comprehend the basics of the internet and web terminologies.
2. To introduce the client-side scripting language concepts for developing client-side applications.
3. To prepare students with technical knowledge and skill needed to protect and defend computer systems, networks and data from unauthorized access, attacks and damage.

UNIT I

Basics of Internet and Web: The basics of Internet, World Wide Web, IP Address, Web page, Home page, Web site, Static, Dynamic and Active web page, Web Server, Web Browser, Web Hosting, DNS, Domain Registration, URL, **Overview of Protocols**: SMTP, FTP, HTTP etc, HTTP request and response.

UNIT II

Introduction to HTML: HTML Basics, Essential Tags, Tags and Attributes, Open tags & Closed tags, Text Styles and Text Arrangements, Exposure to Various Tags, Color and Background of Web Pages, Lists and their Types, Order and their Types, Hypertext, Hyperlink, Links, Anchors and URLs, Links to External Documents, Creating Table, Frame, Form and Style Sheet.

UNIT III

Java Script: Scripting language, Client-Side scripting language, Java Script, Simple Java Script, variables, functions, conditions, loops, Operators, Web forms and validations.

DHTML: Features of DHTML, Combining HTML, CSS, Java Scripts, events and buttons, controlling browser.

UNIT IV

Introduction to Cyber Security & Cyber Law: Cyber Law- Importance of Cyber Law, Cybercrime, categories of Cybercrime, Cyber criminology, **Cyber security:** Importance of Cyber security, Different domain of Cyber security, hardware vulnerability, software

vulnerability, **Threat**: Definition, Types of Threat; Cyber-attack.

UNIT V

Terms associated with Cyber Crime & Cyber Security: Hacking, Cracking, Phishing, Spoofing, Data masking, Cryptanalysis, Cyber warfare, Scanning, Session hijacking, Malicious software, Strong Password, Weak Password.

Information Gathering Techniques: Tools or techniques of the attacker to gather information.

IT Act 2008: Importance, different sections of IT Act 2008.

Course Outcomes: *After successful completion of the course, the students will be able to*

1. Learn the basics of the internet and web.
2. Design and develop the web applications using HTML and Java Scripts.
3. Understand the importance of Cyber Laws and Cyber Security.
4. Know the techniques to prevent cyber attack and different sections of IT Act 2008.

Text Books:

1. N.P. Gopalan and J. Akilandeswari, **Web Technology: A Developer's Perspective,** PHI Publication, 7th Edition, 2016.
2. Nilakshi Jain and Ramesh Menon, **Cyber security and Cyber Laws**, Wiley Publication, 2nd Edition, 2020.

Reference Books:

1. Ivan Bayross, **Web Enabled Commercial Application Development Using HTML, DHTML, Java Script, Perl CGI**, BPB Publications, 2009.
2. M. Merkow, J. Breithaupt, **Information Security Principles and Practices**, Pearson Education.2005