Darrang College (Autonomous), Tezpur-784001

Syllabus for FYUGP BCA

Approved by :

Board of Studies meeting held on 26-12-2025 &
&
Academic Council vide Resolution no. 2, dated 29-12-2025

SYLLABUS

Bachelor of Computer Application

(FYUGP)

Department of Computer Science

Darrang College (Autonomous)

PROGRAMME STRUCTURE (FYUGP BCA)

Semester	Course Name	Credit	L+T+P	Course Type
1	Introduction to C- Programming	4	**3+0+1**	**Core**
	Mathematics 1	4	**3+1+0**	**Core**
	Computer Fundamentals and Application Software	3	**2+0+1**	**SEC**
	Alternative English	2		
	MDC 1	3		
	VAC	4		
2	Data Structures & Algorithms Using C	4	**3+0+1**	**Core**
	Digital Logic	4	**3+1+0**	**Core**
	Web Designing	3	**2+0+1**	**SEC**
	English Communication			
	MDC 2			
	VAC 2			
3	Computer Organization and Architecture	4	**4+0+0**	**Core**
	Object Oriented Programming through C++	4	**3+0+1**	**Core**
	Automata Theory and Languages	4	**4+0+0**	**Core**
	Mobile Application Development	3	**2+0+1**	**SEC**
	MDC 3			
	VAC 3			
4	Database Management System	4	**3+0+1**	**Core**
	Mathematics & Numerical Methods	4	**3+0+1**	**Core**
	Software Engineering	4	**4+0+0**	**Core**
	System Software	4	**3+0+1**	**Core**
	Internship	4		**Core**
5	Operating System	**4**	**3+1+0**	**Core**
	Artificial Intelligence	4	**3+0+1**	**Core**
	Programming in Python	4	**3+0+1**	**Core**
	Operating System Lab	2	**0+0+2**	**Core**
	Minor Project	**6**	**1+0+5**	**Core**
6	Computer Networks	**4**	**4+0+0**	**Core**
	Information Security & Cyber Law	4	**4+0+0**	**Elective 1**
	Graph Theory			
	Cloud Computing	4	**3+0+1**	**Core**
	MAJOR PROJECT	9	**---**	**Core**

PROGRAM STRUCTURE (4th Year)

7	Research Methodology	4	4+0+0	Core
	Cryptography and Network Security	4	4+0+0	Core
	System Administration	4	3+0+1	Core
	Computer Graphics	4	3+0+1	Elective 2
	Optimization Technique		4+0+0	
	Machine Learning	4	3+0+1	Elective 3
	Embedded System		4+0+0	
8	Dissertation	4	0+0+4	Core
	Data Mining	4	3+0+1	Core
	Cyber Security	4	3+0+1	Core
	UI/UX Development	4	2+0+2	Elective 4
	IoT System		3+0+1	

	Compiler Design	4	4+0+0	Elective 5
	Data Science		4+0+0	

Programme outcomes of FYUGP BCA Programme

The completion of the BCA Programme shall enable a student to:

i) To communicate technical information both orally and in writing

ii) Apply the knowledge gained in core courses to a broad range of advanced topics in computer science, to learn and develop sophisticated technical products independently.

iii) To design, implement, and evaluate computer-based system, process, component, or program to meet desired needs by critical understanding, analysis, and synthesis

iv) Identify applications of Computer Science in other fields in the real world to enhance the career prospects

v) Realize the requirement of lifelong learning through continued education and research.

vi) Use the concepts of best practices and standards to develop user interactive and abstract application

vii) To prepare the students for a teaching career up to class 10 level.

viii) Understand the professional, ethical, legal, security, social issues and responsibilities

Introduction to C-Programming (BCA-CR-01014)

1. Learning Outcomes: At the end of the course, students will be able to:
   (a) Understand the basics of C programming like data types and operators
   (b) Understand and write program in C to implement conditions, loops, functions
   (c) Work on arrays, strings and basic file operations

2. Prerequisites: NIL

3. Semester: 1

4. Course type: Compulsory

5. Theory credit: 3

6. Practical credit: 1

7. Number of required hours:
   (a) Theory: 45 hours (45 classes)
   (b) Practical: 30 hours (15 classes)

Detailed Syllabus (Theory)

Unit 1: Basics of C Programming (10 Lectures)

Introduction to programming languages. Comparative study of different types of Programming languages. Translator and types. Structure of a C program. Introduction to Header files. Main function and a simple program execution. Compiling, linking and executing a program. C tokens – keywords, identifiers, constants, operators. Statements and expressions in C. Basic data types in C - integers, float, double, character, void. Size and range of data types. Variables. Storage Class. Constants – integer constant, real constant, character constant, string constant. Declaration and initialization of variables and constants. Assigning values to variables. Operators in C – binary and unary operators. Arithmetic, assignment, logical, comparison, bitwise and conditional operators. Precedence and Associativity of operators.Input and output statements – getchar(), getch(), getche(), gets(), putchar(), puts(), scanf(), printf(), format specifiers. Typecasting.

Unit 2: Control Structures in C (9 Lectures)

Need and use of control structure. Conditional statements – if, else, switch case, nested if-else. Loops – while loop, for loop, do-while loop. Using loop for counting iterations. Using while loop for indefinite iterations. Nested loops. Use of Break and continue statements.

Unit 3: Arrays and Strings (8 Lectures)

Introduction to Arrays. Types of Arrays. Declaration and initialization of arrays. Processing/Accessing array elements. Multidimensional arrays. Introduction to Strings. Declaration and initialization of strings. String input and output in C.

Unit 4: Pointers and Functions (9 Lectures)

Introduction to Pointers. Pointer declaration and initialization. Pointers and addresses. Pointers and Arrays. Basic concept of dynamic memory allocation, malloc(), calloc(). Introduction to user defined functions. Function declaration and definition. Return types of function. Function arguments. Function calling – call by value vs call by reference. Passing an array as argument to a function. Recursive Function.

Unit 5: Introduction to Structures and Unions (4 Lectures)

Basic concept of Structures and Unions in C. Structure declaration and initialization. Union declaration and initialization. Difference between structures and unions.

Unit 6: File Processing and Preprocessor Directives (5 Lectures)

Basic concept of file handling. Comparative study of Different types of files and their uses. Opening and closing file. Reading contents from file, writing to files. Random access to files. File pointers. Error handling in file operations. Preprocessor directives in C - #define, #ifdef, #include, #ifndef, and #endif directives. Using preprocessor directives to define constants and macros.

List of Practical

1. Write a program to take input of two numbers and print their sum, product, difference.
2. Write a program to find the smallest or greatest of three numbers given as input.
3. Write a program to print the sum and product of digits of an integer.
4. Write a program to check whether an input number is palindrome or not.
5. Write a program to take a number representing a month and print the name of the month using switch case.
6. Write a program that calculates the grade of a student based on their marks in a subject using nested if-else statements. Also print the range of marks for each grade using switch case.
7. Write a program to take a number as input and print all the even numbers up to that number using while and for loop.
8. Write a program to ask the user for an input to stop a loop or continue repeating after printing the iteration count using a do-while loop.
9. Write a program to find the maximum, minimum, sum and average of n numbers without using array.
10. Write a program that takes two integers as input and finds their greatest common divisor (GCD) and LCM

11. Write a program that calculates the sum of the first n terms of the Fibonacci sequence, where n is entered by the user, using a for-loop.
12. Write a program that takes an integer as input and checks if it is a prime number.
13. Write a program to create an array with inputs from the user and print the same.
14. Write a program to read an integer and display the binary/octal equivalent of the number.
15. Write a program to take a matrix from the user and print the transpose of the same.
16. Write a program to find the sum, product of 2 matrices.
17. Write a program to take a string of length more than 10 and find the number of vowels in the string. Also print the position of the vowels in the string.
18. Write a program using pointers to copy a string to another string variable without using library function.
19. Write a program to check whether an input string is palindrome or not.
20. Write a program that swaps two numbers using pointers.
21. Write a program to calculate Factorial of a number (i) using recursion, (ii) using iteration
22. Write a program to find sum of n elements entered by the user. To write this program, allocate memory dynamically using malloc() / calloc() functions or new operator.
23. Write a function to accept two arrays as argument and returns their sum as an array.
24. Write a program to implement struct in C. Create a structure of Student with RNo, Name and other credentials with proper datatype and print the same.
25. Write a program to implement union in C. Create a structure of Person with Pid, Name and other credentials with proper datatype and print the same.
26. Write a C program that opens a file for reading and displays the contents of the file in binary mode and text mode.
27. Write a C program that opens a file for reading and displays the contents of the file line by line on the screen.
28. Write a C program that opens a file in append mode and allows the user to add text to the end of the file.

Mathematics 1 (BCA-CR-01024)

1. Learning Outcomes: After successful completion of this course, students will be able to:
   (a) Learn the concepts of set, relation, and function from Computer Science point of view.
   (b) Know how to view a table/database as an n-ary relation.
   (c) Learn what a matrix is and relate it with arrays used in programming.
   (d) Understand determinants and how determinants are used in solving simultaneous equations.
   (e) Get familiar with statistical and probabilistic measures that are used in computation related software/packages.

2. Prerequisites: NIL

3. Semester: 1

4. Course type: Compulsory

5. Theory credit: 4

6. Number of required hours: Theory: 60 hours (60 classes)

Detailed Syllabus (Theory)

UNIT 1: Sets, Relations and Functions (13 Lectures)

Sets: definition of set, Representation of set, Different types of set, cardinality of sets, finite, countable and infinite sets. Properties of Set Operations on sets, Venn diagram. Relations: Definition and properties of binary relations, closures of relations, equivalence relations. Functions: Definition of function, one-to-one and onto, principles of mathematical induction.

UNIT 2: Matrices (17 Lectures)

Definition and different types of matrices, row and column operations; vectors and matrices, Addition, subtraction and multiplication of matrices, Properties of matrix operations, Existence of additive and multiplicative identity and additive inverse of a matrix. Representing relations using matrices. Transpose of a matrix and its properties. Symmetric and skew symmetric matrices, Elementary transformation of a matrix, Invertible matrices.

UNIT 3: Determinants (17 Lectures)

Determinant of a square matrix, minor, cofactor, Adjoint of a matrix and matrix inversion. Inverse of a matrix using elementary transformation. Rank of a matrix and determination of rank of a matrix. Eigen values and Eigen vectors of a matrix. Cayley-Hamilton theorem – Cramer's rule, Consistency of a system of linear non-homogenous equations and existence of solutions, Solutions of simultaneous linear equations by Gaussian elimination method, Gauss Jordan Method.

UNIT 4: Statistics and Probability (13 Lectures)

Data, Attributes, and variables; Frequency distribution, Cumulative frequency. Graphical representation of Frequency distribution: Histogram, Frequency Polygon, Frequency Curve and Cumulative Frequency curves (Ogive). Bar Diagram, Subdivided Bar Diagram, Pie diagrams. Measures of central tendency-Mean, Median and Mode. Measures of variation – Range, Interquartile range, Standard Deviation and Variance.

Sample space, events, random variables, basic probability. Conditional Probability and Bayes theorem.

Data Structure & Algorithm Using C (BCA-CR-02014)

1. Learning Outcomes: At the end of the course, students will be able to:
   a) Understand and apply the fundamental data structures and algorithms – such as arrays, linked lists, stacks, queues, trees, sorting and searching algorithms using C programming language.
   b) Analyze the time and space complexity of different algorithms and choose the appropriate algorithm for a given problem.
   c) Develop efficient algorithms to solve various computational problems by utilizing data structures and algorithms covered in the course.

2. Prerequisites: Basic Knowledge of C programming

3. Semester: 2

4. Course Type: Compulsory

5. Theory Credit: 3

6. Practical Credit: 1

7. No of required hours:
   a) Theory: 45 hours (45 Classes)
   b) Practical: 30 hours (15 Classes)

Detailed Syllabus (Theory)

UNIT 1: Data Structures Overview (8 Lectures)

Concepts of Data Types, Abstract Data Type, Data Structure, Fundamental and Derived Data Types. Importance of data structures. Array as a data structure (characteristics, advantages, disadvantages). Representation of arrays – single and multidimensional. Address calculation of array element using column and row major ordering. Address translation functions for one- & two-dimensional arrays. Insertion and deletion in arrays. Use of arrays for large number representation.

UNIT 2: Linked Lists (9 Lectures)

Initialization and implementation of structures. Structure and pointers. Self-referential structure. Introduction to linked lists. Singly linked list, doubly linked list, circular linked list. Operations on lists – creation, insertion, deletion, traversal.

UNIT 3: Stacks and Queues (10 Lectures)

Definition of Stack and Queue. Representation of stacks and queues using arrays and linked lists. Stack operations – push, pop. Queue operation – enqueue, dequeue. Circular Queue, Priority Queue, Conversion of infix arithmetic expression containing arithmetic operators and parenthesis to postfix and prefix expression. Evaluation of postfix expression.

UNIT 4: Tree (8 Lectures)

Definition of Trees – General tree and Binary tree. Basic terminologies – parent, child, height, depth, leaf, node, internal nodes, external nodes. Brief concept of Forest, ordered trees, strictly binary tree, complete binary tree. Representation of trees using arrays and linked lists. Binary tree traversal methods – pre-order, in-order, post-order. Recursive and non-recursive algorithms for traversal methods. Binary search trees. Operation on BST – creation, insertion and deletion of anode. Definition and characteristics of threaded binary trees.

UNIT 5: Searching and Sorting (5 Lectures)

Linear and binary search. Indexed search. Hashing. Sorting algorithms – Insertion Sort, Selection Sort, Bubble Sort, Merge Sort, Quick Sort.

UNIT 6: Analysis of Algorithm and Complexity (5 Lectures)

Complexity measures of an algorithm – Time and space complexity. Average case and worst-case analysis. Asymptotic notation as a measure of algorithm complexity, O and θ notations. Best Case, Worst Case and Average Case Analysis of sorting algorithms-Selection sort, Bubble sort, Insertion sort, Heap sort, Quick sort and Searching algorithms – linear search and binary search.

List of Practical

1. Write a program to declare an array and initialize the values according to the user. Now ask the user for a number n and return the nth element from the array.
2. Implement linked list in a program by writing functions for the following:
   a. Create a singly linked list of n nodes
   b. Count the number of nodes in the list
   c. Print the values of all the nodes
   d. Add a node at first, last and kth position in the linked list
   e. Delete a node from first, last and kth position
   f. Search for an element in the list. If found, return the position of the node. If not found, return a negative value.
3. Write a program to implement doubly linked list.
4. Write a function to concatenate two linked lists.
5. Write a program to take a number k and split the linked list after kth position.
6. Write a program to merge two sorted linked lists.
7. Write a program to implement list of lists.
8. Write a program to implement stack using array. Use push and pop operations on the array representation of the stack. Check whether the stack is full or empty.
9. Write a program to implement stack using linked list. Use push and pop operations on the stack by inserting nodes and deleting nodes from the linked list. Also check if the stack is full or empty.
10. Write a program to evaluate a simple postfix expression using stack.

11. Write a program to convert a decimal number into binary number using stack.
12. Write a program to implement queue using array. Add new elements to the queue and remove elements from the queue represented by array. Check whether the queue is full or empty.
13. Write a program to implement queue using linked list. Add new elements to the queue and remove elements from the queue represented by linked list. Also check whether the queue is full or empty.
14. Implement binary search and linear search algorithms on arrays.
15. Implement following sorting algorithms: Bubble sort, Insertion sort, Selection sort.

Digital Logic (BCA-CR-02024)

1. Learning Outcomes: After completing this course, students will have grasp of Fundamental concepts of digital logic that will make their base to understand the concepts of computer architecture and organization.
2. Prerequisites: NIL
3. Semester: 2
4. Course type: Compulsory
5. Theory credit: 4
6. Number of required hours: Theory: 60 Hours (60 classes)

Detailed Syllabus (Theory)

UNIT 1: Introduction to Number System (10 Lectures)

Binary numbers, octal and hexadecimal numbers. Conversion of number. 1's complement and 2'scomplement, representation of signed binary number: 1's complement, 2's complement and signed magnitude, subtraction with complements, arithmetic addition and subtraction of signed binary numbers, binary codes: BCD, Excess-3, error detection code: parity bit, error correction code: Hamming code, gray code, ASCII, EBCDIC.

UNIT 2: Boolean Algebra and Logic Gates (25 Lectures)

Definition of Boolean algebra, two valued Boolean algebra, duality principle, theorems and postulates of boolean algebra, precedence of boolean operators, boolean expression and Venn diagram, Boolean functions and truth tables, complement of a boolean function, min terms and max terms, canonical forms of a Boolean function, sum of min terms and its short notation, product of max terms and its short notation, conversion between canonical forms, standard form of a boolean function. Karnaugh Map (upto 4 variable), don't-care conditions. Logic Gates. NAND and NOR implementation of boolean functions.

UNIT 3: Combinational Circuits (12 Lectures)

Definition of combinational circuit, half adder, full adder, half subtractor, full subtractor, BCD-to-Excess-3 code converter, encoders and decoders, multiplexers, de-multiplexers.

UNIT 4: Sequential Circuits (13 Lectures)

Flipflops, RS flip flop, D flip flop, JK flip flop, T flip flop, master slave flip flop. State table of a sequential circuit, state diagram, characteristic tables of flip flops, Analysis of Clocked Sequential circuits, State Reduction and Assignment, Flip –Flop Excitation tables, Design procedure of clocked sequential circuit.

COMPUTER ORGANIZATION AND ARCHITECTURE (BCA-CR-03014)

1. Learning Outcomes: At the end of the course, students will be able to:

   (a) Understand the basics of organizational and architectural issues of a digital computer and classify and compute the performance of machines, Machine Instructions.

   (b) Demonstrate an understanding of the design of the functional units of a digital computer system.

   (c) Recognize and manipulate representations of numbers stored in digital computers

2. Prerequisites: NIL
3. Semester: 3
4. Course type: Compulsory
5. Theory credit: 4
6. Practical credit: 0
7. Number of required hours:
   (a) Theory: 60 hours (60 classes)
   (b) Practical: 0 hours (0 classes)

Detailed Syllabus (Theory)

UNIT 1: Basic Concept of Computer Architecture (10 Lectures)

Introduction to Computer Organization, Von Neumann Architecture, Functional units of a computer: *Input unit*, *Output unit*, *Memory, ALU*, *Control unit*, Stored Program Concept, Instruction codes, Computer Registers.

UNIT 2: Micro Operations and Assembly Language (10 Lectures)

Register Transfer, Bus Transfer, memory transfer, Micro Operations and types - logic Micro Operation, Shift Micro Operation, Arithmetic micro-operation, Arithmetic Logic Shift Unit, Assembly Language: Assembler, Assembler commands, Assembly and execution of programs.

UNIT 3: Central Processing Unit (15 Lectures)

Brief introduction to CPU, General Register Organization, Stack Organization – Register Stack, Memory Stack, Instruction set architecture - types, formats, Addressing Modes and different types of Addressing Modes, Data Transfer Instructions, Data Manipulation Instructions, Arithmetic Instructions, Logical and Bit Manipulation Instructions, Shift Instructions, Program Control, Reduced Instruction Set Computer- CISC Characteristics, RISC Characteristics, RISC vs CISC.

UNIT 4: Input-Output Organization (15 Lectures)

Peripheral devices, Input-Output Interface - I/O Bus and Interface Modules, I/O versus Memory bus, Memory mapped I/O vs. Isolated I/O, Accessing I/O devices, Data transfer between the CPU and I/O devices*: Programmed I/O, Interrupt initiated I/O and Direct Memory Access (DMA) - DMA Controller, DMA Transfer*, Example of I/O interface, Asynchronous data transfer - Strobe control, Handshaking.

UNIT 5: Memory Organization (10 Lectures)

Memory Hierarchy, Main Memory - *RAM and ROM*, Memory Address Map, Memory Connections to CPU, Auxiliary Memory, Cache Memory - *Associative Mapping, Direct Mapping, Set-Associative Mapping*, Writing into Cache, Cache Initialization, Virtual Memory- Address and Memory Space, Address Mapping using papers, virtual memory, memory management hardware, memory protection. Secondary Storage devices.

OBJECT ORIENTED PROGRAMMING THROUGH C++ (BCA-CR-03024)

Learning Outcomes: At the end of the course, students will be able to:

(a) Understand the fundamental principles of Object-Oriented Programming.

(b) Develop the ability to design, implement, test, and debug C++ programs using OOP

(c) Enhance problem-solving and software development skills by applying object-oriented design techniques

2. Prerequisites: NIL

3. Semester: 3

4. Course type: Compulsory

5. Theory credit: 3

6. Practical credit: 1

7. Number of required hours:

(a) Theory: 45 hours (45 classes)

(b) Practical: 30 hours (30 classes)

Detailed Syllabus (Theory)

UNIT 1: Object Oriented Programming Cconcepts (8 Lectures)

Principles of OOP, OOP Paradigm, Basic Concepts of OOP, comparison of procedural programming and OOP, advantages of OOP, Application of OOP.

UNIT 2: Introduction to C++ (9 Lectures)

What is C++, Tokens, C++ data types, Operators in C++, Class, objects, Access specifiers: public, private, protected, Functions - Function Prototyping, Call by reference, Return by reference, Inline function, Function Overloading, Defining Member Functions, Static data members and functions, Array of objects, Friend functions, Constructor and Destructors - Constructors, Parameterized Constructors, Multiple Constructors in a class, Copy constructor, Destructors.

UNIT 3: Inheritance and Polymorphism (9 Lectures)

Inheritance: Types of Inheritance – Single, Multilevel, Multiple, Hierarchal, Hybrid, Defining Derived Classes, Virtual Base Classes, Constructors in derived classes, Pointers: Addresses and pointers, Pointer to object, Compile time and Runtime Polymorphism, this pointer, Pointers to derived classes, Virtual functions, Pure virtual functions.

UNIT 4: Operator overloading and Templates (9 Lectures)

Defining Operator Overloading, Operator Overloading: Overloading unary and binary operators, Overloading with Friend functions, Type conversion, Necessity of friend function, Templates- Generic functions, generic class, template functions, Function templates, Class templates, Simple applications of templates.

UNIT 5: I/O Operations, Files & Exception Handling (10 Lectures)

Definition of Streams, Stream class hierarchy, Unformatted I/O Operations, Formatted I/O operations, Classes for File Streams, Opening and Closing a File : open() and close() functions, Manipulators of File Pointers : seekg(), seekp(), tellg(), tellp() functions, Sequential Input and output Operations : put (), get(), write(), read() functions, Error handling File Operations : eof(), fail(), bad(), good(), Exception Handling: Introduction, Exception Handling Mechanism, Concept of throw & catch with example Case studies in object-oriented application design.

List of Suggested Practical

1. Write a C++ program to create a class Student with data members: name, roll, and marks. Include a input function to read details and a display function to print details. Create two student objects and display their details.

2.Create a class Rectangle with two data members: length and breadth. Include functions: getData() to input values and area() to return area (length × breadth). In main(), create two Rectangle objects, input values, and display the area of each rectangle.

3.Define a class Number with one data member n. Include a function read() to input the number and a function check() to print whether the number is even or odd. Create an object of the class in main() and test the program.

4. Write a C++ program to create a class Box with data members: length, breadth, and height. The class should include a default constructor (initialize all values to 1), A parameterized constructor (initialize with user-defined values), A function volume() to calculate and return the volume, A destructor that displays a message when an object is destroyed. In main(), create one object using the default constructor, one object using the parameterized constructor. Display the volume of both boxes.

5. Create a class Counter with a static int count, a constructor that increments and displays the count and a destructor that decrements and displays "Object destroyed". In main(), create multiple objects inside a block to show how constructors and destructors are called automatically.

6. Write a C++ program to demonstrate the use of pointers by declaring an integer variable, storing its address in a pointer, and displaying both the value of the variable and its address using the pointer.

7. Write a C++ program to create a function int findMax(int a, int b, int c) that returns the largest of three numbers. In the main() function ask the user to enter three integers, Call the function to find the maximum and Display the result.

8. Write a C++ program to demonstrate function overloading by creating multiple area() functions to calculate the area of a circle, a rectangle, and a triangle based on different parameter lists.

9. Write a C++ program to demonstrate the difference between call by value and call by reference by creating two swap functions—one using value parameters and one using reference parameters—and show how the original values change or remain unchanged after each function call.

10. Write a C++ program to demonstrate single inheritance by creating a base class Person with data members for name and age, and a derived class Student that adds roll number and marks; take input and display all details using objects of the derived class.

11. Write a C++ program to demonstrate multilevel inheritance by creating a base class Vehicle, a derived class Car, and another class ElectricCar derived from Car, and display the details of an electric car object using data from all three classes.

12. Write a C++ program to overload the + operator to add two complex numbers by creating a class Complex with real and imaginary parts, and display the result using an object returned by the overloaded operator.

13. Write a C++ program to demonstrate function overloading by creating multiple area() functions to calculate the area of a circle, a rectangle, and a triangle based on different parameter lists.

14. Write a C++ program to demonstrate both formatted and unformatted I/O operations by using cin.get(), getline(), cout.put(), and cout.write() to read and display different types of input from the user.

15. Write a C++ program to create a text file and write user-entered data into it using ofstream, open(), write(), and put() functions.

16. Write a C++ program to copy the contents of one file into another file using read() and write() functions.

17. Write a C++ program to demonstrate exception handling by creating a function that divides two numbers. If the denominator is zero, throw an exception. In the main() function, use try, catch, and throw to handle the division operation and display an appropriate error message when division by zero occurs.

AUTOMATA THEORY AND LANGUAGES (BCA-CR-03034)

Learning Outcomes: At the end of the course, students will be able to:

(a) Design and analyze finite automata, regular expressions, context-free grammars, and pushdown automata for various language-processing problems.
(b) Understand and apply key theoretical concepts such as decidability, computability, and the hierarchy of formal languages.
(c) Model real-world computation problems using automata and evaluate the computational power and limitations of different computational models.

2. Prerequisites: NIL

3. Semester: 3

4. Course type: Compulsory

5. Theory credit: 4

6. Practical credit: 0

7. Number of required hours:

    (a) Theory: 60 hours (60 classes)

Detailed Syllabus (Theory)

UNIT 1: Introduction to Automata (12 Lectures)

Finite automata concepts, alphabets, strings, languages, deterministic finite automata (DFA), nondeterministic finite automata (NFA), ε-moves, DFA to NFA conversions, NFA to DFA conversions.

UNIT 2: Regular Languages and Regular Grammar (12 Lectures)

Regular expressions, closure properties of regular languages, Connection between regular expressions and regular languages, regular grammar, minimization of DFA, pumping lemma for regular languages, Moore Machine and Mealy Machine

UNIT 3: Context-Free Grammar (CFG) and Languages (12 Lectures)

Definition of CFG, leftmost and rightmost derivations, derivation trees, Parsing and Ambiguity in grammars and languages, Simplification of Context free Grammars- removing useless productions, empty - productions and unit-productions, Pumping Lemma for CFL, Using Pumping Lemma to show that certain languages are not Context free parse trees, ambiguity, simplification of CFG (removal of null, unit, useless productions), Chomsky Normal Form (CNF), Greibach Normal Form (GNF).

UNIT 4: Pushdown Automata (PDA) (12 Lectures)

Definition of PDA, deterministic vs nondeterministic PDA, PDA construction from CFG and vice-versa, language acceptance by empty stack and final state.

UNIT 5: Turing Machines and Computability (12 Lectures)

Turing machine definition, types of TM, TM as acceptor and transducer, Church-Turing thesis, decidability, halting problem, recursively enumerable languages, hierarchy of formal languages.

Questions

- Design a DFA to check whether a binary string contains an even number of 0s.
- Construct an NFA for strings ending with "ab" over the alphabet {a, b}.
- Convert the given NFA to an equivalent DFA.
  (Give any small NFA—for example: start → a → final)
- Write the regular expression for all strings that contain at least one 'a' over {a, b}.
- Design a Moore machine that outputs 1 when the input bit is 1, otherwise outputs 0.
- Construct a Mealy machine that outputs 1 whenever two consecutive 1s appear.
- Convert the regular expression (a + b)*ab into a finite automaton.
- Eliminate left recursion from a simple grammar.
- Convert grammar G: S → aS | bS | ε into a regular expression.
- Check whether the grammar is ambiguous using a small example.

DATABASE MANAGEMENT SYSTEM (BCA-CR-04014)

Learning Outcomes: At the end of the course, students will be able to:

(a) Understand the fundamental concepts of relational database management systems (RDBMS) and their applications.
(b) Demonstrate proficiency in designing entity-relationship (ER) models to represent real-world scenarios.
(c) Utilize SQL to manipulate and query relational databases effectively.
(d) Develop practical skills through hands-on exercises and projects using a popular RDBMS software.

2. Prerequisites: NIL
3. Semester: 4
4. Course type: Compulsory
5. Theory credit: 3
6. Practical credit: 1
7. Number of required hours:
    (a) Theory: 45 hours (45 classes)
    (b) Practical: 30 hours (30 classes)

Detailed Syllabus (Theory)

UNIT 1: Introduction and DBMS concepts (8 Lectures)

Importance of database, difference between database approach and File oriented approach, meaning of database system, Database users, Database management systems (DBMS) Meaning, advantages of using a DBMS, Types of DBMS - Hierarchical, network, relational, object-oriented, object-relational, Database system concepts and architecture - Data Models, schemas, and instances, Three schema architecture, Data Independence. Introduction to non-relational database.

UNIT 2: ER model and RDBMS (9 Lectures)

Overview, ER-Model, Components of ER model - Entities and attributes, Entity types, entity sets, keys and value set, symbols of ER-Diagram Constraints, ER-Diagrams, Relationship: one-to-one, one-to-many, many-to-many, examples of ER-Diagram, Enhanced Entity-Relationship (EER): Super classes, subclasses and inheritance, Specialization and Generalization and their constraints. Introduction to RDBMS Terminologies: characteristics of relation, Domain constraints, Entity, attributes and tuples, relational data integrity, Relational Schemas, Keys (Super key, candidate key, primary key, foreign key), finding keys.

UNIT 3: Normalization, Relational Algebra (9 Lectures)

Definition of Functional dependency, Interference Rules for Functional Dependencies, equivalence of sets of Functional Dependencies, Minimal sets of Functional Dependencies, Introduction to Normalization, Definitions of INF, 2NF, 3NF, BCNF, Relational algebra - Relational algebraic operations - (union, intersection, difference, project, Cartesian product, rename, select, division, join.), Examples of queries in Relational Algebra.

UNIT 4: Introduction to SQL (10 Lectures)

Characteristics of SQL, advantage of SQL, data types, literals, string, numeric, specifying constraints in SQL, Types of SQL commands: DDL, DML, DCL, SQL operations - arithmetic, comparison, logical and set operators, Operator precedence, Examples of basic Queries and sub queries, Aggregate functions - Applications, general rules, examples of Aggregate functions provided by SQL, Table - create, modify, alter, drop. Views and indexes, Insert, update and Delete operations. Joints, unions, Intersections, Minus. Cursors in SQL, Embedded SQL.

UNIT 5: Transaction Processing Concepts (9 Lectures)

Introduction to Transaction Processing, Read & Write Operations, Need for Concurrency Control, Transaction States, Commit Point of a Transaction, Transaction Properties, Schedules and Recoverability - Schedules, Characterizing Schedules, Serializability of Schedules, Testing for Conflict Serializability of a Schedule, Uses of Serializability, Concurrency Control Techniques - The Locking Protocol, Locks. Two Phase Locking (2PL), Deadlock and its Prevention.

Practical

1. Case Study of ER Diagram, Relational Model, Normalization
2. Practical Assignments: Use any appropriate RDBMS to be made available – preferably MySQL)
    a. Exercises using DDL Commands
    b. Exercises using DML Commands
    c. Querying using ANY, ALL, IN, Exists, Not Exists, Union, Intersection Constraints etc.
    d. Querying using Aggregate functions, group by, having and Creation and dropping of views.
3. Create a database and a table named STUDENT with fields (RollNo, Name, Course, Marks), insert at least five records, and display all records.

4.Write SQL queries to update a student's marks, delete a record, and retrieve students scoring more than 80 marks.

5.Create two tables DEPARTMENT and EMPLOYEE, insert records, and perform INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL JOIN.

6.Write queries to demonstrate GROUP BY and HAVING using an EMPLOYEE table (group by department and show average salary).

7.Create a table PRODUCT and perform operations on constraints: PRIMARY KEY, UNIQUE, NOT NULL, DEFAULT, and CHECK.

8.Demonstrate use of ORDER BY, DISTINCT, BETWEEN, IN, LIKE, and aggregation functions (SUM, MAX, MIN, AVG, COUNT).

9. Find SK, CK, no of SK and FK from given FDs.

MATHEMATICS & NUMERICAL METHODS (BCA-CR-04024)

1. Learning Outcomes: At the end of the course, students will be able to understand basic numerical and statistical problems and to have skills to solve these problems.

2. Prerequisites: NIL

3. Semester: 4

4. Course type: Compulsory

5. Theory credit: 3

6. Practical credit: 1

7. Number of required hours:

    (a) Theory: 45 hours (45 classes)

    (b) Practical: 30 hours (30 classes)

Detailed Syllabus (Theory)

UNIT 1: Calculus (10 Lectures)

Intuitive idea of limits and continuity. Limits of polynomials and rational functions. Derivatives, Algebra of derivative of a function, Derivative of polynomials and trigonometric functions. Roll's theorem, Lagrange's Mean Value theorem and Taylor's theorem. Meaning of the sign of derivative, indeterminate forms, maxima and minima (single variable only)

UNIT 2: Permutation and Combination (6 Lectures)

Permutations and combinations, circular permutations, permutations with repetitions, combinations with repetitions, permutations of sets with indistinguishable objects.

UNIT 3: Mathematical Logic (10 Lectures)

Connectives, truth tables, Tautologies and Contradictions, Equivalence and Implications, NAND and NOR, Normal forms- CNF, DNF, Converting expressions to CNF and DNF, Theory of inference, Propositional Calculus, Predicate calculus (only introduction), predicates and quantifiers.

UNIT 4: Solution of non-linear equation (5 Lectures)

Bisection method, Newtons method, Regula Falsi method.

UNIT 5: Solution of simultaneous linear equation (4 Lectures)

Basic elimination method, method of successive approximation.

UNIT 6: Interpolation (5 Lectures)

Newton's interpolation, Lagrange's interpolation, Newton's divided difference method.

UNIT 7: Numerical integration **(5 Lectures)**

Trapezoidal rule, Simpson's 1/3rd and Simpson's 3/8th rule

List of Suggested Practical:

C/C++ programs to implement the methods of Unit 4, 5, 6, 7

SOFTWARE ENGINEERING (BCA-CR-04034)

1. **Learning Outcome**:
   On successful completion of this course, the student should be able to:
   - Determine the primary problems that impact all software development processes.
   - Choose relevant software development processes models, methodologies, and strategies for managing a specific software development process, and justify the choices
   - Implement different software estimation metrics such as cost, effort size, staffing etc.
   - Describe various software design approaches and various coding and testing strategies used in software engineering principles
   - Know about software reliability and how to calculate software maintenance cost.

2. Prerequisites: NIL

3. Semester: 4

4. Course Type: Compulsory

5. Theory Credit: 4

6. Practical Credit: 0

7. No of Hours:

Theory: 60 Hours (60 classes)

8. List of Books:

(a) Rajib Mall: Fundamentals of Software Engineering; PHI Learning Pvt. Ltd.

(b) Roger S. Pressman: Software Engineering: A practitioner's Approach; McGraw Hill.

Detailed Syllabus:

Unit 1: Introduction **6 Lectures**

Definition of Software Engineering, differentiation between Computer Science, Software Engineering and System Engineering, software crisis, software development process

Unit 2: Software Development Life Cycle models **10 lectures**

Definition of software development Life cycle (SDLC) models, Various life cycle modes: Classical Waterfall model, Iterative Waterfall model, Prototyping model, Evolutionary (Incremental) model, Spiral model, Agile Model, Agile V/s traditional SDLC Models, SCRUM model, Advantages and disadvantages of each of these SDLC models.

Unit 3: Software requirement analysis: **10 lectures**

Requirements analysis, requirements elicitation, analysis tools, requirements definition, requirements specification, need for software requirement specification (SRS), characteristics and components of SRS, estimation in Project Planning Process, Project Scheduling.

Unit 4: Software design: **8 lectures**

Concept of fundamental design; Design approaches- top-down & bottom-up, structured, function-based, object-based & object-oriented design; design specification and notations.

Unit 5: Implementation and testing: **10 lectures**

Coding standards and procedures, modularity, software testing fundamentals, approaches to software testing: unit testing, integration testing and system testing and their types, path testing.

Unit 6: Maintenance: **6 lectures**

Maintenance problems, the nature of maintenance, planning for maintenance. Version Control

Unit 7: Testing Strategies & Tactics: **10 lectures**

Software Testing Fundamentals, Strategic Approach to Software Testing, Test Strategies for Conventional Software, Validation Testing, System testing, Black-Box Testing, White-Box Testing and their type, Basis Path Testing.

SYSTEM SOFTWARE (BCA-CR-04044)

1. **Learning Outcome:** After completing this course, students will have understanding of various types of system software.

2. Prerequisites: NIL

3. Semester: 4

4. Course Type: Compulsory

5. Theory Credit: 3

6. Practical Credit: 1

7. No of Hours:

Theory: 45 hours (45 classes)

Practical: 30 hours (15 classes)

8. List of Books:

a) System Software: An Introduction to Systems Programming, Leland L. Beck, D. Manjula, Pearson

b) Systems Programming, Dhananjay Dhamdhere, McGraw Hill Education

Detailed Syllabus (Theory)

Unit I: Introduction to System Software **10 hours**

Types of software, Application software and system software, examples of system software, system programming, system software and machine architecture, the simplified instructional computer (SIC): memory, registers, data formats, instruction formats, addressing modes, instruction set, input and output, programming examples in SIC. Case study on Android.

Unit II: Assemblers **12 hours**

Assembler definition, basic assembler functions, assembler algorithm and data structure, handling instruction formats and addressing modes, program relocation, handling literals, symbol defining statements, expressions, assembler design options: one pass assemblers and multi pass assemblers, introduction to NASM assembler

Unit III: Loaders and Linkers **7 hours**

Loading, relocation and linking, loader, absolute loader, bootstrap loader, relocating loader, program linking, linking loader, linkage editor, static and dynamic linking.

Unit IV: Macro processor **6 hours**

Definition of macro processor, macro definition and expansion, macro processor algorithm and data structures, conditional macro expansion, general purpose macro processors, macro processing within language translators

Unit V: Compilers **10 hours**

Compiler definition, grammars, lexical analysis, syntactic analysis, operator precedence parsing, recursive descent parsing, code generation, intermediate form, code optimization: machine dependent and machine independent, interpreter

Practical

A) Case Study of Linux Operating System and System Administration using Linux
B) Implementation of lexical analyser and Symbol Table

OPERATING SYSTEM (BCA-CR-05014)

1. Learning Outcomes: At the end of the course, students will be able to:

   a) Understand the basics of operating systems like kernel, shell, types and views of operating systems.
   b) Understand various process management concepts including scheduling, synchronization, and deadlocks.
   c) Understand memory management concepts.
   d) Understand the file systems, access methods, directory structures, and file system implementation techniques, which are essential for organizing and managing data on storage devices.

2. Prerequisites: NIL

3. Semester: 5

4. Course type: Compulsory

5. Theory credit: 4

6. Practical credit: 0

7. Number of required hours:

   (a) Theory: 60 hours (60 classes)

Detailed Syllabus (Theory)

UNIT 1: Introduction (10 Lectures)

Definition of Operating Systems, Functions of Operating Systems, Services of Operating Systems, Types of Operating Systems: Batch OS, Multiprogrammed OS, Time sharing, Personal computer systems, Parallel systems, Real time and Distributed Operating Systems. Case study on Android.

UNIT 2: Process Management (15 Lectures)

Definition of a Process, Process states, Process Control Block, Process Synchronization: The critical section problem, two process solution and multiple process solution, Synchronization hardware, Semaphores—implementation, Scheduling Criteria, Schedulers: Short term, medium term and long-term schedulers. Scheduling Algorithms: FCFS, Round Robin, SJF and Priority Algorithms (preemptive and non-preemptive), Deadlocks: Deadlock Characterization, Methods for Handling Deadlocks, Deadlock Prevention, Deadlock Avoidance, Deadlock Detection, Recovery from Deadlock. Threads: Overview and benefits.

UNIT 3: Memory Management (15 Lectures)

Logical and physical address space, Swapping, Contiguous allocation – memory protection, memory allocation, fragmentation, Paging: Basic method, hardware support, protection. Demand Paging: Basic concepts, basic scheme of page replacement, page replacement algorithms: FIFO and LRU

UNIT 4: File System (10 Lectures)

File concept, Access methods - Sequential Access and Direct Access, Directory Structure - Single-Level Directory, Two-Level Directory, Tree Structured Directories, Acyclic-Graph Directories, File System Mounting, file Sharing, File Protection - Access Control List (ACL), other protection approaches, File System Structure, File-System Implementation, Directory Implementation.

UNIT 5: Basic shell Commands in Linux (10 Lectures)

Elementary Linux Utilities: cal, date, who, uname, passwd, echo, tput, bc.

Elementary Linux commands; cd, mkdir, pwd, rmdir, chmod, chown, ls, cat, cp, rm, more, wc, split, cmp.

Simple Filters: pr, head, tail, cut, paste, sort, uniq, tr.

ARTIFICIAL INTELLIGENCE (BCA-CR-05024)

1. Learning Outcome:

After completing this course, students will know the fundamentals of artificial intelligence (AI), identify problems where artificial intelligence techniques are applicable and able to apply basic principles of AI in solutions that require problem solving, inference, perception, knowledge representation, and learning.

2. Prerequisites: NIL

3. Semester: 5

4. Course Type: Elective

5. Theory Credit: 3

5. Practical Credit: 1

6. No of Hours:

a) Theory: 45 hours (45 classes)

b) Practical: 30 hours (15 classes)

Detailed Syllabus (Theory)

UNIT 1: Introduction (4 Hours)

Introduction to Artificial Intelligence, Background and Applications, Turing Test and Rational Agent approaches to AI, Introduction to Intelligent Agents, their structure, behavior and environment.

UNIT 2: Problem Solving and Searching Techniques (14 Hours)

Problem Characteristics, Production Systems, Control Strategies, Breadth First Search, Depth First Search, Hill climbing and its Variations, Heuristics Search Techniques: Best First Search, A* algorithm, Constraint Satisfaction Problem, Means-End Analysis, Introduction to Game Playing, Min-Max and Alpha-Beta pruning algorithms.

UNIT 3: Knowledge Representation (12 Hours)

Introduction to First Order Predicate Logic, Resolution Principle, Unification, Semantic Nets, Conceptual Dependencies, Frames, and Scripts, Production Rules, Conceptual Graphs.

Programming in PYTHON

UNIT 4: Dealing with Uncertainty and Inconsistencies (6 Hours)

Truth Maintenance System, Default Reasoning, Probabilistic Reasoning, Bayesian Probabilistic Inference, Possible World Representations.

UNIT 5: Understanding Natural Languages (5 Hours)

Parsing Techniques, Context-Free and Transformational Grammars, Recursive and Augmented Transition Nets. LLM

UNIT 6: Ethics and AI for Professionals (4 Hours)

Ethics in implementation of AI technologies, Generative AI. Use of AI technologies in different profession.

List of Suggested Practical:

1. Write a program to create a simple rule-based chatbot.
2. Implement a program to check sentiment (positive/negative/neutral) from user text.
3. Write a Python program to classify a message as spam or not spam.
4. Implement a simple expert system using IF–ELSE rules.
5. Write a program to recommend movies based on genre using Python dictionaries.
6. Implement BFS (Breadth-First Search) in Python.
7. Implement DFS (Depth-First Search) in Python.
8. Write a program to solve a simple 8-puzzle problem using BFS.
9. Implement A* Search (basic example).
10. Implement greedy best-first search.

PROGRAMMING IN PYTHON (BCA-CR-05034)

1. Learning Outcomes: At the end of the course, students will know about fundamentals of Python Programming and Problem Solving.
2. Prerequisites: NIL
3. Semester: 5
4. Course type: Compulsory
5. Theory credit: 3
6. Practical credit: 1
7. Number of required hours:
    (a) Theory: 45 hours (45 classes)
    (b) Practical: 30 hours (30 classes)

Detailed Syllabus (Theory)

UNIT 1: Programming and Problem Solving (7 Lectures)

Concept of problem solving, Problem definition, Program design, Debugging, Types of errors in programming, Documentation. Flowchart, decision table, algorithms, Structured programming concepts, Programming methodologies viz. top-down and bottom-up programming.

UNIT 2: Introduction to Python (8 Lectures)

Structure of a Python Program, Elements of Python. Python Interpreter, Using Python as calculator, Python shell, Indentation, Atoms, Identifiers and keywords, Literals, Strings, Operators (Arithmetic operator, Relational operator, Logical or Boolean operator, Assignment Operator, Ternary operator, Bit wise operator, Increment or Decrement operator)

UNIT 3: Creating Python Programs (7 Lectures)

Input and Output Statements, Control statements (Branching, Looping, Conditional Statement, exit function, Difference between break, continue and pass), Defining Functions, default arguments, Errors and Exceptions.

UNIT 4: User Defined Function and Modules (2 Lectures)

User defined function, the return statement, Recursive functions, Multiple assignment.

Use of Modules.

UNIT 5: Strings and Lists **(8 Lectures)**

String as a compound data type, Length, Traversal and the for loop, String slices, String comparison, A find function, Looping and counting, List values, accessing elements, List length, List membership, Lists and for loops, List operations, List deletion. Cloning lists, Nested lists

UNIT 6: Data Structures **(5 Lectures)**

Arrays, list, set, stacks and queues.

UNIT 7: Searching and Sorting **(4 Lectures)**

Linear and Binary Search, Bubble sort, Selection sort and Insertion sort.

List of Suggested Practical

1. Write a function that takes an integer input and calculates the factorial of that number.
2. Write a function that takes a string input and checks if it is a palindrome or not.
3. Write a list function to convert a string into a list, as in list ('abc') gives [a, b, c].
4. Write a program to generate Fibonacci series.
5. Write a program to check a number is Armstrong or not
6. Write a program to check whether the input number is even or odd.
7. Write a program to print all even number between a range (for example between 1 and 100).
8. Write a program to print all prime number between a range (for example between 1 and 100).
9. Write a program to compare three numbers and print the largest one.
10. Write a program to print factors of a given number.
11. Write a method to calculate GCD of two numbers.
12. Write a program to implement linear and binary search on lists.
13. Write a program to sort a list using insertion sort and bubble sort and selection sort.

OPERATING SYSTEM LAB (BCA-CR-05042)

1. Learning Outcomes: At the end of the course, students will be able to:
   a) Understand the basics of operating systems like kernel, shell, types and views of operating systems.
   b) Understand various process management concepts including scheduling, synchronization, and deadlocks.
   c) Understand memory management concepts.
   d) Understand the file systems, access methods, directory structures, and file system implementation techniques, which are essential for organizing and managing data on storage devices.
2. Prerequisites: NIL
3. Semester: 5
4. Course type: Compulsory
5. Practical credit: 2
6. Number of required hours: Practical: 30 classes

List of Suggested Practical

1. Introduction to shell script: Shell variables, data types.
2. File and directory commands: ls, pwd, cd, cp, mv, rm, mkdir, rmdir, File permissions: chmod, chown, System commands: date, cal, who, ps, kill, Redirection and pipes: >, >>, |
3. Advanced commands: echo, cat, sudo, df, tar, apt-get, chmod, hostname, useradd, passwd, groupadd, grep, sed, uniq, wc, od, gzip, gunzip, find, date, cal, clear, top, ps, kill.
4. String and array manipulation, Control structures, Functions, Command -Line Arguments and Options, Input and Output Redirection, Pipes, Tees, Command Substitution.
5. Using system calls in C program in linux: fork(), exec(), exit(), getpid(), mkdir(), rmdir() etc.
6. Introduction to regular expressions. Pattern matching and substitution: use of the following tools: sed. awk, grep
7. Error handling and debugging techniques

MINOR PROJECT (BCA-CR-05056)

COMPUTER NETWORKS (BCA-CR-06014)

1. Learning Outcomes: At the end of the course, students will be able to:

   (a) Explain the fundamental concepts of computer networks, including network architectures, protocols, OSI & TCP/IP models, and data transmission techniques.

   (b) configure, analyze, and troubleshoot basic networking devices and protocols, such as IP addressing, routing algorithms, switching, and error-control mechanisms.

   (c) evaluate different network technologies and communication methods to design secure, efficient, and reliable small-scale network solutions.

2. Prerequisites: NIL

3. Semester: 6

4. Course type: Compulsory

5. Theory credit: 4

6. Practical credit: 0

7. Number of required hours:

   (a) Theory: 60 hours (60 classes)

Detailed Syllabus (Theory)

UNIT 1: Basic Concept of Networking (10 Lectures)

Introduction, Components of network, Networking Advantages, Types of networks, Network devices: Bridges, Repeaters, Switches, Routers, Modem, Gateway, Hub, Wifi-routers, Client/Server Method of Connecting Computers, Physical structure and topology, categories of network, ISO OSI model, Data and signals: Analog and digital signal, transmission impairment, performance.

UNIT 2: Physical layer (10 Lectures)

Different transmission Media, Guided Media, Unguided Media, Digital and analogue transmission, Digital to analogue conversion - ASK, FSK, PSK, Analog to digital conversion - PCM, Multiplexing – Frequency Division Multiplexing, Wavelength Division Multiplexing, and Time Division Multiplexing, Switching - Circuit switch network, datagram network.

UNIT 3: Data Link Layer (15 Lectures)

Introduction to DLL, Workings of DLL - Error detection and correction, Linear Block code - Simple parity check code, hamming codes, Cyclic Redundancy Check, Flow and error control - Stop and wait, Stop-and-Wait ARQ, Go Back-N ARQ and Selective repeat ARQ, PPP - framing, transition phase.

UNIT 4: Network and Transport Layer (15 Lectures)

Network Layer: Introduction to Network layer, IPv4 overview, IPv4 and IPv6 address, IPv4 vs IPv6, Network address translation, Network Management, Network Elements, ICMP, Routing algorithms - Flooding, Distance-vector routing, link state routing. Transport layer: function of Transport layer, process to process delivery, UDP operation, TCP - feature, segment, connection establishment and termination.

UNIT 5: Application layer and LAN's (10 Lectures)

Application layer: DNS, TELNET, E-mail, FTP, WWW (architecture), HTTP, RTP. LAN: Ethernet (Standard, Fast and Gigabit Ethernet), IEEE 802.11 (Architecture, MAC sublayer), Bluetooth (Architecture, layer), Subnetting.

INFORMATION SECURITY AND CYBER LAW (BCA-CR-06024)

1. Learning Outcome: After the completion of the course, the students will be able to develop basic understanding of security, cryptography, system attack and defences against them.

2. Prerequisites: NIL

3. Semester: 6

4. Course Type: Elective

5. Theory Credit: 4

6. Practical Credit: 0

7. No of Hours: Theory: 60 hours (60 classes)

List of Books:

(a) Merkow, M., & Breithaupt, J.(2005) Information Security Principles and Practices. 5th edition. Prentice Hall.

(b) Cryptography And Network Security Principles And Practice, Fourth or Fifth Edition, William Stallings, Pearson Edition.

(c) Cyber Law & Cyber Crimes, Advocate Prashant Mali; Snow White publications, Mumbai

(d) The Information Technology Act, 2000; Bare Act – Professional Book Publishers, New Delhi

Detailed Syllabus:

UNIT 1: Introduction (15 Lectures)

Basic components of security (Confidentiality, Integrity and Availability), Attacks, Computer Crime, Security Services, Security Mechanism, Cyber Crimes, information Technology ACT, Cryptography, Substitution Cipher, Transposition Cipher, Block Cipher, Stream Cipher, Confusion, Diffusion, Symmetric Key, Asymmetric Key, Encryption, DES Algorithm, Hash Function, Digital Signature, Digital Certificate.

UNIT 2: Program Security (10 Lectures)

Program Security, Program Errors, Buffer Overflow, Incomplete mediation, Time-of-check to Time of- use Errors, Malicious codes, Virus, Threats, Control against Programs, Program Security Issues. Protection in OS: Memory and Address protection, Access control, File protection, User Authentication.

UNIT 3: Database Security **(10 Lectures)**

Reliability, Integrity, Sensitive Data, Inference, Multilevel Security, Issues regarding the right to access information: Protecting Data, Multiple security level and categorization of data and users, Loss of integrity, Loss of availability, Loss of confidentiality, Access control, Inference control, flow control, data encryption

UNIT 4: Security in Networks (Cyber Attack) **(15 Lectures)**

Threats in Networks, Security Controls- Architecture, Encryption, Content Integrity, Strong Authentication, Firewalls: Design and Types of Firewalls, Intrusion Detection System, Secure Email, Denial-of-service attacks, Man in the middle Attack, Phishing, Spoofing and Spam Attacks, Drive-by attack, SQL Injection, Birthday attack, Social Engineering attack, Password Attack. Cross site scripting Attack, Malware Attack, Administering Security, Security Planning, Risk Analysis, Organizational Security Policy, Web Servers and Browsers, HTTP, Cookies, Caching, Secure Socket Layer (SSL), Secure Electronic Transaction (SET), E-mail Risks, Spam, E-mail Protocols, Simple Mail Transfer Protocol (SMTP), Post office Protocol (POP), Internet Access Message protocol (ICMP), Secured Mail: Pretty Good Privacy (PGP), S/MIME (Secure/Multipurpose Internet Mail Extensions)

UNIT 5: Cyber Laws **(10 Lectures)**

Cyber-crime, Types of crimes, Information technology Act 2000: Salient Feature of IT Act 2000, various authorities under IT Act and their powers, Penalties & Offences, amendments, Sections under the Information Technology Act such as:

- [Section 43] Penalty and compensation for damage to computer etc.
- [Section 65] Penalty for temping with the computers source documents
- [Section 66] Punishment for hacking with computer system, data alteration etc
- [Section 66A] Punishment for sending offensive messages through any communication services
- [Section 66B] Receiving stolen computer's resources or communication devices dishonestly
- [Section 66C] Punishment for identity theft
- [Section 66D] Punishment for cheating by impersonation by using computer resource
- [Section 66E] Punishment for violation of privacy
- [Section 66F] Punishment for cyber terrorism
- [Section 67] Punishment for publishing or transmitting obscene material in electronic form
- [Section 67A] Punishment for publishing or transmitting of material containing sexually explicit act, etc. in electronic form
- [Section 67B] Punishment for publishing or transmitting of material depicting children in sexually explicit act, etc. in electronic form
- [Section 72] Breach of confidentiality and privacy

GRAPH THEORY (BCA-CR-06034)

1. Learning Outcome: After completing this course, students will have understanding of graph theoretic concepts, problems and associated algorithmic solutions.

2. Prerequisites: NIL

3. Semester: 6

4. Course Type: Elective

5. Theory Credit: 4

6. Practical Credit: 0

7. No of Hours: Theory: 60 hours (60 classes)

List of Books:

(a) Introduction to Graph Theory, Douglas B. West, Pearson

(b) Introduction to Graph Theory, Robin J. Wilson, Pearson Education Limited

(c) Graph Theory with Applications to Engineering and Computer Science, Narasingh Deo, PHI

Detailed Syllabus:

Unit I: Introduction **5 hours**

Graph, directed and undirected graph, weighted and unweighted graph, simple and multigraph, degree, in degree and out degree, Handshaking theorem, complete graph, bipartite graph, cut set, cut vertices, graph representations: incidence matrix, adjacency matrix and adjacency list, BFS traversal and DFS traversals on a graph using stack and queue data structures, isomorphism, homomorphism

Unit II: Connectivity, paths and cycle **15 hours**

Walk, path and cycle, connected graphs, disconnected graphs, components, Hamiltonian path, Hamiltonian cycle, Hamiltonian graphs, Dirac's theorem, Eulerian path, Eulerian cycle, Euler graphs, Fleuri's algorithm, 2-connected graphs, connectivity and digraph, k-connected and k-edge connected graphs, application of Menger's theorem, Shortest path problem, variations of shortest path problem: single source shortest path problem, single pair shortest path problem and all pairs shortest path problem, Dijkstra's algorithm, Bellman Ford algorithm, Floyd Warshall's algorithm, Johnson's algorithm.

Unit III: Tree **12 hours**

Tree, forest, properties of tree, spanning tree, spanning forest, counting trees, Cayley's theorem, matrix-tree theorem, minimum spanning tree, Kruskal's algorithm, Prim's algorithm, disjoint spanning trees, graph decomposition, graceful labeling, graceful graph, binary tree, binary search tree, AVL tree, multiway search tree, B tree, B+ tree

Unit IV: Matching and coloring **13 hours**

Matching, bipartite matching, maximum bipartite matching, Ford Fulkerson's algorithm for finding maximal bipartite matching, perfect bipartite matching, non-bipartite matching, maximal nonbipartite matching, largest maximal matching, perfect non-bipartite matching, Hall's Marriage theorem, vertex cover, vertex cover and matching, independent sets, dominating sets, atable matching, Hungarian algorithm, introduction to Edmonds Blossom shrinking algorithm, vertex coloring, k-colorable graph, chromatic number, Brook's theorem, clique number, map coloring problem

Unit V: Digraph **7 hours**

Digraph, simple digraph, connected and strongly connected digraph, orientable graph, Eulerian digraph, Hamiltonian digraph, tournament, Markov chains, Flow networks, residual graph, augmenting path, Ford Fulkerson's algorithm

Unit VI: Classical problems **8 hours**

Travelling Salesman Problem, variants of Travelling Salesman Problem, Chinese Postman Problem, variants of Chinese Postman Problem, the minimum connector problem, Huffman coding and Huffman tree, Konisgsberg bridge problem, three utilities problem

CLOUD COMPUTING (BCA-CR-06044)

1. Learning Outcomes:

At the end of the course, students will be able to: Understand basic concepts of cloud computing and its need in modern computing. Understand virtualization and how it supports cloud technology. Use basic cloud services like virtual machines, storage, and applications. Understand cloud security basics including data privacy and authentication.

2. Prerequisites: NIL

3. Semester: 1

4. Course type: Compulsory

5. Theory credit: 3

6. Practical credit: 1

7. Number of required hours:

(a) Theory: 45 hours (45 classes)

(b) Practical: 30 hours (15 classes)

Unit 1: Introduction to Cloud Computing: (4 hours)

Cloud-definition, benefits, usage scenarios, History of Cloud Computing, Cloud Architecture, Types of Clouds, Cloud Service models, issues in Clouds, Cloud cost benefits

Unit 2: Types of cloud computing services (8 hours)

Types of Cloud Services and Providers Types of Cloud services, Software as a Service, Platform as a Service, Infrastructure as a Service, Database as a Service, Monitoring as a Service, Communication as services. Service Providers Google, Amazon, Microsoft Azure, IBM, Sales force.

Unit 3: Virtualization (16 hours)

Virtualization for Cloud and Cloud Security Virtualization for Cloud Need for Virtualization, Pros and cons of Virtualization, Types of Virtualizations, System VM, Process VM, Virtual Machine monitor, Virtual machine properties, HLL VM, Hypervisors, Xen, KVM, VMWare, Virtual Box, Hyper-V

Unit 4: Cloud Storage & Management: (9 hours)

Cloud storage fundamentals, Storage as a Service (STaaS), Data centers & distributed file systems, Cloud resource allocation & scheduling, Cloud performance monitoring tools, Cloud load balancing and auto-scaling

Unit 5: Future Trends and cloud security (8 hours)

Future Trends in Mobile Communication Cloud Security: Infrastructure Security- Network level security, Host level security, Application-level security, Data security, Authentication in cloud computing, Cloud security challenges.

Practical Questions:

- Create a free cloud account (AWS / Azure / GCP)
- Create a VM instance on EC2 / Azure VM
- Deploy a simple HTML website on cloud
- Use cloud storage (S3, Azure Blob, Google Cloud Storage)

MAJOR PROJECT (BCA-CR-06059)