Four Year Undergraduate Programme
Subject: BCA

Programme Name	Eligibility Criteria of the Programme, (if any)	Semester	Course Name	Course Code	Credits	Credit Distribution of the Course			Pre-requisite of the Course (if any)	Internal Marks	External Marks	Practical
						L	T	P				
FYUGP in BCA	10+2 Qualified	1	Introduction to C- Programming	CIT0100104	4	3	0	1	No	30	45	25
			Mathematics I	CIT0100204	4	4	0	0	No	40	60	0
		2	Data Structures & Algorithms Using C	CIT0200104	4	3	0	1	No	30	45	25
			Digital Logic Fundamentals	CIT0200204	4	3	0	1	No	30	45	25
		3	Computer Organization and Architecture	CIT0300104	4	3	0	1	No	30	45	25
			Mathematics II	CIT0300204	4	4	0	0	No	40	60	0
			Object Oriented Programming through C++	CIT0300304	4	3	0	1	No	30	45	25
		4	Database Management System	CIT0400104	4	3	0	1	No	30	45	25
			Operating System	CIT0400204	4	3	0	1	No	30	45	25
			Automata Theory and Languages	CIT0400304	4	4	0	0	No	40	60	0
			Python Programming	CIT0400404	4	3	0	1	No	30	45	25
			System Software	CIT0400504	4	4	0	0	No	40	60	0
		5	Software Engineering	CIT0500104	4	4	0	0	No	40	60	0
			Java Programming	CIT0500204	4	3	0	1	No	30	45	25
			Computer Networks	CIT0500304	4	4	0	0	No	40	60	0
			Information Security and Cyber Laws OR Computer Oriented Numerical and Statistical Methods	CIT0500404 CIT0500504	4	4	0	0	No	40	60	0
			Internship	CIT0500604	4	0	0	0	No	0	100	0
		6	Computer Graphics OR Optimization Techniques	CIT0600104 CIT0600204	4	4	0	0	No	40	60	0
			Artificial Intelligence	CIT0600304	4	3	0	1	No	30	45	25
			Data Mining and Warehousing	CIT0600404	4	4	0	0	No	40	60	0
			Graph Theory	CIT0600504	4	4	0	0	No	40	60	0
			Project	CIT0600604	4	4	0	0	No	40	60	0

Template for Common Courses in FYUGP in BCA (for 1st and 2nd Semesters)

Programme Name (AEC/VAC/ MDC/SEC)	Eligibility Criteria of the Programme , (if any)	Semester	Course Name	Course Code	Credits	Credit distribution of the course			Pre-requisite of the Course (if any)	Internal Marks	External Marks
						L	T	P			
SEC-1	No	1	Computer Fundamentals & Application Software		3	2	0	1	No	20	30
SEC-2	No	2	Introduction to Web Technologies		3	1	0	2	No	20	30

Template for BCA (Fourth Year) (FYUGP in BCA with Honours): (Students need to take 5 papers in 7th and 8th semester each)

Programme Name	Eligibility Criteria of the Programme if any)	Semester	Course Name	Course Code	Credits	Credit Distribution of the Course			Pre-requisite of the Course (if any)	Internal Marks	External Marks	Practical
						L	T	P				
FYUGP in BCA (Honors)	BCA upto 3rd Year	7 (***Any four*** *Electives (E)*)	Research Methodology (C)	CIT0700104	4	4	0	0	No	40	60	0
			Fundamentals of Machine Learning (E)	CIT0700204	4	3	0	1	No	30	45	25
			Advance Operating System (E)	CIT0700304	4	3	0	1	No	30	45	25
			Advanced Computer Organization and Architecture (E)	CIT0700404	4	4	0	0	No	40	60	0
			Cryptography and Network Security (E)	CIT0700504	4	4	0	0	No	40	60	0
			Advanced DBMS (E)	CIT0700604	4	3	0	1	No	30	45	25
			Compiler Design (E)	CIT0700704	4	4	0	0	No	40	60	0
		8 (***Any four*** *Electives*	Advanced Data Structure (E)	CIT0800104	4	3	0	1	No	30	45	25
			Embedded Systems (E)	CIT0800204	4	4	0	0	No	40	60	0
			Mobile Application Development (E)	CIT0800304	4	3	0	1	No	30	45	25
			System Administration and Networking (E)	CIT0800404	4	3	0	1	No	30	45	25
			Mobile Computing (E)	CIT0800504	4	4	0	0	No	40	60	0

		(E))	Project and Presentation (C)	CIT0800604	4	0	0	4	No	40	60	0

Template for BCA (Fourth Year) (FYUGP in BCA Honours with Research)

Programme Name	Eligibility Criteria of the Programme, if any	Semester	Course Name	Course Code	Credits	Credit Distribution of the Course			Pre-requisite of the Course (if any)	Internal Marks	External Marks	Practical
						L	T	P				
FYUGP in BCA (Honors with Research)	BCA upto 3rd Year	7 (***Any four*** *Electives (E)*)	Research Methodology (C)	CIT0700104	4	4	0	0	No	40	60	0
			Fundamentals of Machine Learning (E)	CIT0700204	4	3	0	1	No	30	45	25
			Advance Operating System (E)	CIT0700304	4	3	0	1	No	30	45	25
			Advanced Computer Organization and Architecture (E)	CIT0700404	4	4	0	0	No	40	60	0
			Cryptography and Network Security (E)	CIT0700504	4	4	0	0	No	40	60	0
			Advanced DBMS (E)	CIT0700604	4	3	0	1	No	30	45	25
			Compiler Design (E)	CIT0700704	4	4	0	0	No	40	60	0
		8	Dissertation (c)	CIT- Dissertation	16	16	0	0	No			
			Project and Presentation	CIT0800104	4	0	0	4	No	40	60	0

CIT0100104: INTRODUCTION TO C-PROGRAMMING

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. Course level: 100-199

6. Theory credit: 3

7. Practical credit: 1

8. Number of required hours:

a) Theory: 45 hrs (45 classes)

b) Practical: 30 hrs (15 classes)

c) Non Contact: NIL

9. Reference books:

(a) B.S. Gottfried, *Schaum's Outline of Theory and Problems of Programming with C*, Mcgraw-Hill, 2007.
(b) B. Kernighan, D. Ritchie, *The C Programming Language*, Second Edition, Prentice Hall, 1988.
(c) E. Balaguruswami, *Programming in ANSI C*, 2nd Ed., Tata McGraw Hill, 2004.
(d) P. Greg, D. Miller, *C Programming: Absolute Beginner's Guide*, 3rd ed. Que, 2016.

10. Detailed Syllabus:

A. Theory

UNIT 1: Getting started with C programming (10 Lectures)

Introduction to programming languages- High-level vs low level languages, compiled vs interpreted languages. Structure of a C program. Introduction to Header files. Main function and a simple program execution. Compiling and executing a program. C tokens – keywords, identifiers, constants, operators. Statements and expressions in C. Basic data types in C - integers, floats, doubles, characters. Void. Size and range of values of data types. Variables. 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. Order of precedence of operators. Associativity of operators. Input and output statements – getchar(), getc(), getch(), putchar(), putc(), puts(), scanf(), printf(), format specifiers. Typecasting.

UNIT 2: Control Structures in (9 Lectures)

Control Structures in C. Basic programming constructs- Sequence, selection and iteration. Conditional statements – if, else, switch case. Nested conditions. Loops – for loop, while loop, do- while loop. Using loop for counting iterations. Using while loop for indefinite iterations. Nested loops. Break and continue statements.

UNIT 3: Arrays and Strings (8 Lectures)

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

UNIT 4: Functions and Pointers (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 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. Basic concept of recursion.

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 in C (5 Lectures)

Basic concept of file handling. Opening and closing file using fopen() and fclose(). Binary vs text files. Reading and writing files – fgets(), fscanf(), fprintf(). Random access to files.

B. List of Practical

(This is a suggestive list only. Questions need not be restricted to this list. The practical are advised to be performed in Linux environment)

1. Write a program in C to print “Hello World”
2. Write a program to take input of two numbers and print their sum, product, difference.
3. Write a program to find the smallest or greatest of three numbers given as input.
4. Write a program to print the sum and product of digits of an integer.
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) using nested while loops and if statements.
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 that calculates the sum of the first *n* terms of an arithmetic series, where *n*, the first term and common difference of the series are entered by the user.
14. Write a program to compute the sum of the first *n* terms of the following series S =1-2+3-4+5……
15. Write a program to create an array with inputs from the user and print the same.
16. Write a program to perform following actions on an array entered by the user:
    a) Print the even-valued elements
    b) Print the odd-valued elements
    c) Print the array in reverse order
17. Write a program to take a matrix from the user and print the transpose of the same.
18. Write a program to ask for the name of the user and print the same.
19. 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.
20. Write a program using pointers to copy a string to another string variable without using library function.
21. Write a program that swaps two numbers using pointers.
22. Write a program to calculate Factorial of a number (i) using recursion, (ii) using iteration
23. Write a program which takes the radius of a circle as input from the user, passes it to another function that computes the area and the circumference of the circle and displays the value of area and circumference from the main() function.
24. 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.
25. Write a function to accept two arrays as argument and returns their sum as an array.
26. 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.
27. 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.
28. Write a C program that opens a file for reading and displays the contents of the file in binary mode and text mode.
29. Write a C program that opens a file for reading and displays the contents of the file line by line on the screen.
30. Write a C program that opens a file in append mode and allows the user to add text to the end of the file.

CIT0100204: MATHEMATICS I

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. Course level: 100-199

6. Theory credit: 4

7. Practical credit: 0

8. Number of required hours:

a) Theory: 60 hrs (60 classes)

b) Practical: NIL

c) Non Contact: NIL

9. List of Reference Books:

(a) J. P. Tremblay and R. Manohar, *Discrete Mathematics Structures withApplications to Computer Science,* Mc-Graw Hill.
(b) N. Ch.SN Iyengar, K.A. Venkatesh, V. M. Chandrasekaran, P. S. Arunachalam, *Discrete Mathematics,* Vikash Publishing House Pvt Ltd.
(c) C.L.Liu, *Elements of Discrete Mathematics,* Mc-GrawHillInternationalEd.

10. Course Details:

A. Theory

UNIT 1: Sets, Relations and Functions (16 Lectures)

Sets: Definition of set, cardinality of sets, finite, countable and infinite sets. Operations on sets, Venn diagram. Principle of inclusion and exclusion and their applications on simple problems. Multisets.

Relations: Definition and properties of binary relations, closures of relations, equivalence relations,equivalenceclassesandpartitions,n-aryrelationsandrepresentationofn-ary relations as tables. Partial ordering relations and lattices,

Functions: Definition of function**,** one-to-one and onto, principles of mathematical induction. Concave and convex functions.

UNIT 2: Matrices (15 Lectures)

Definition and different types (such as identity matrix, diagonal matrix etc) 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 inverseofamatrix.Representingrelationsusingmatrices.Transposeofamatrixandits

properties. Symmetric and skew symmetric matrices, Elementary transformation of a matrix, Invertible matrices.

UNIT 3: Determinants (16 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 (Stressing on symmetric matrices), Cayley-Hamilton theorem – Cramer's rule, Consistency of a system of linear non-homogenous equations and existence of solutions (statement only),Simple problems, Solutions of simultaneous linear equations by Gaussian elimination method.

UNIT 4: Fundamentals of Statistics and Discrete Probability (13 Lectures)

Types of Data, Attributes and variables, Construction of Frequency, Cumulative frequency, Graphical representation of Frequency distribution: Histogram, Frequency Polygon, Frequency Curve and Cumulative Frequency curves (Ogivecurves), Diagrammatic representations: Simple bar, Subdivided bar, 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.

CIT0200104: DATA STRUCTURE & ALGORITHM USING C

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: NIL

3. Semester: 3

4. Course Type: Elective

5. Course Level: 100-199

6. Theory Credit: 3

7. Practical Credit: 1

8. No of required hours:

a) Theory: 45 hrs

b) Practical: 30 hrs

c) Non Contact: 5 hrs

9. List of Reference Books:

a) Weiss, Mark Allen, *Data Structures and Algorithm Analysis in C*, 3rd ed., Pearson, 2012

b) Sedgewick, Robert, *Algorithms in C, Parts 1-5 (Bundle): Fundamentals*, *Data Structures, Sorting, Searching, and Graph Algorithms*, 3rd ed., Addison-Wesley Professional, 2002.

c) Goodrich, Michael T., and Roberto Tamassia, *Data Structures and Algorithms in C*, 2nd ed., Wiley, 2011.

d) Gilberg, Richard F., and Behrouz A. Forouzan, *Data Structures: A Pseudocode Approach with C*, Narosa Publishing House, 2009.

10. Detailed Syllabus:

A. Theory

UNIT 1: Data Structures Overview and Arrays (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, merging and splitting.

UNIT 3: Stacks and Queues (9 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: Binary Trees **(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 a node. Definition and characteristics of threaded binary trees. Min heap and Max heap.

UNIT 5: Searching and Sorting **(6 Lectures)**

Linear and binary search. Indexed search. Hashing. Hash Functions – division method, mid square method, folding. Conflict resolution – linear and quadratic probe. Sorting algorithms – Insertion sort, Selection sort, Bubble sort, Merge sort, Quick sort, Counting sort, Heap sort. In-place sorting and stable sorting.

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. Analysis of sorting algorithms and Searching algorithms in terms of time and space complexity in best, average and worst case.

Time and Space complexity of algorithms, average case and worst case analysis, asymptotic notation as a measure of algorithm complexity, Θ and O notation. Analysis of sorting algorithms- Selection sort, Bubble sort, Insertion sort, Heap sort, Quick sort and analysis of searching algorithms – linear search and binary search.

B. List of Practical

(This is a suggestive list only. Questions need not be restricted to this list. The practical are advised to be performed in Linux environment using C programming language.)

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 n^{th} element from the array.
2. Write a program to implement array initialized with the numbers divisible by three up to 30. Write a function which accepts the array and return the positions of the even numbers in the array.
3. 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 k^{th} position in the linked list
    e. Delete a node from first, last and k^{th} position
    f. Search for an element in the list. If found, return the position of the node. If not found, return a negative value.
4. Write a program to implement doubly linked list.

5. Write a function to concatenate two linked lists.
6. Write a program to take a number k and split the linked list after k^{th} position.
7. Write a program to merge two sorted linked lists.
8. Write a program to implement list of lists.
9. 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.
10. 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.
11. Write a program to evaluate a simple postfix expression using stack.
12. Write a program to convert a decimal number into binary number using stack.
13. 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.
14. 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.
15. Implement binary search and linear search algorithms on arrays.
16. Implement binary search tree using array by writing a program to:
    a. Create a binary search tree using array
    b. Print the prefix notation of the BST
    c. Print the infix notation of the BST
    d. Print the postfix notation of the BST
    e. Search for an element in the BST
17. Implement binary search tree using linked list by writing a program to:
    a. Create a binary search tree using linked list
    b. Print the prefix notation of the BST
    c. Print the infix notation of the BST
    d. Print the postfix notation of the BST
    e. Search for an element in the BST
18. Implement following sorting algorithms:
    Bubble sort, Insertion sort , Selection sort, Counting sort

CIT0200204: DIGITAL LOGIC FUNDAMENTALS

1. **Learning Outcomes:** After completing this course, students will have grasp of-

   a) 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. **Course level:** 100-199

6. **Theory credit:** 4

7. **Practical credit:** 0

8. **Number of required hours:**

   a) Theory: 60 hrs (60 classes)

   b) Practical: NIL

   c) Non Contact: NIL

9. **List of Reference Books:**

   (a) M. Morris Mano, *Digital Logic and Computer Design*, Pearson India
   (b) V. Rajaraman, T. Radhakrishnan, *Digital Logic and Computer Organization*, PHI Learning

10. **Contents of Syllabus:**

A. Theory

UNIT 1: Introduction to Binary Number System (10 Lectures)

Binary numbers, number base conversions, octal and hexadecimal numbers,1's complement and 2's complement, 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, binary logic, logic gates: AND, OR, inverter, buffer, NAND, NOR, XOR and equivalence.

UNIT 2: Boolean Algebra, Logic Gates and Integrated Circuits (15 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, digital logic gates, integrated circuits and levels of integration, digital logic families

UNIT 3: Simplification of Boolean Functions (10 Lectures)

Map minimization method, two variable map, three variable maps, four variable map, five variable map,

NAND and NOR implementation of boolean functions, don't-care conditions, tabulation method

UNIT 4: Combinational Circuits (12 Lectures)

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

UNIT 5: Sequential Circuits (13 Lectures)

Flipflops, RSflip flop, D flip flop, JK flip flop, T flip flop, master slave flip flops and edge triggered flip flops, state table of a sequential circuit, state diagram, characteristic tables of flip flops, Mealy and Moore machine, flip flop excitation tables, design procedure of clocked sequential circuit, 3-bit binary counter, shift register, ripple counter, RAM.

CIT0300104: COMPUTER ORGANIZATION AND ARCHITECTURE

1. **Learning Outcome:**
   a) Student will able to learn about the structure, function and characteristics of computer systems.
   b) Student will understand the design of the various functional units and components of computers.
   c) Student will identify the elements of modern instructions sets and their impact on processor design.
   d) Student will able to learn about the function of each element of a memory hierarchy.
   e) Student will able to learn about identify and compare different methods for computer I/O.
   f) Student will able to understand the basic architecture of 8085 microprocessor.
2. **Prerequisite:** NIL
3. **Semester:** 2
4. **Course Type:** Compulsory
5. **Course Level:** 200-299
6. **Theory Credit:** 4
7. **Practical Credit:** 0
8. **Number of required hours:**
   a) Theory: 60 hrs
   b) Practical: 0 hrs
   c) Non Contact: 5 hrs
9. **List of reference books:**
   a) M. Morris Mano, *Computer System Architecture*, PHI publication.
   b) Hamachar, Vranesic and Zaky, *Computer Architecture*.
   c) William Stallings, *Computer Organization and Architecture*; Pearson.
   d) Ramesh Gaonkar, *Microprocessor Architecture, Programming, and Applications with the 8085*, 5th Edition.

10. Detailed Syllabus

A. Theory

UNIT 1: Introduction and Digital Logic (10 Lectures)
Definitions of Computer Organization and Architecture, History of computer architecture, Basic functional blocks of a computer, Logic Gates, Boolean Algebra, Map Simplifications, Combinational Circuits, Flip-Flops, Sequential Circuits, Decoders, Multiplexers, Registers and Counters

UNIT 2: Data Representation (8 Lectures)
Number Systems and Conversion, Complements, Fixed Point Representation, Floating Point Representation, Error Detection Codes, Computer Arithmetic - Addition, Subtraction, Other Binary codes: BCD, Excess-3, Gray code**.**

UNIT 3: Register Transfer and Micro-operations (6 Lectures)
Introduction to Register Transfer Language, Register transfer, Bus and Memory transfers, Arithmetic micro-operation- Binary adder, Binary adder-subtractor, Binary incrementer, Arithmetic circuit, Logic micro-operation, Shift micro-operation, Arithmetic logic shift unit.

UNIT 4: Basic Computer Architecture and Design (10 Lectures)

Instruction Codes: Stored Program Organization, Data path in a CPU, Computer Instructions, Instruction Cycle, Memory-Reference Instructions, I/O Interrupt: Types of Interrupt, Interrupt Cycle, Control Unit: Operations of a control unit, Hardwired control unit, Micro-programmed control unit

UNIT 5: Central Processing Unit (10 Lectures)

Computer registers, Types of register- general purpose registers, special purpose registers, index registers, General register organization, Stack organization, Computer instructions: Operands, Instruction format- Three-address instructions, Two-address instructions, One-address instructions, Zero- address instructions, Addressing modes, Data Transfer and Manipulation: Data transfer instructions, Data manipulation instructions, Arithmetic instructions, Logical and Bit manipulation instructions, Shift instructions, Program Control: Status bit conditions, Conditional branch instructions, Subroutine call and return, CISC and RISC architectures.

UNIT 6: Memory Organization (6 Lectures)

Semiconductor memories, Memory cells - SRAM and DRAM, Concept of hierarchical memory organization, Cache memory unit - Concept of cache memory, Mapping methods, Organization of a cache memory unit, Cache replacement policies, Write policy, Concept of virtual memory.

UNIT 7: I/O Organization (6 Lectures)

Access of I/O devices, I/O ports, I/O Interface, Modes of Transfer - Program controlled I/O, Interrupt driven I/O, DMA controlled I/O, Priority Interrupts, Handling interrupts.

UNIT 8: Basics of Microprocessor (4 Lectures)

Introduction to microprocessors, Case Study: 8085 Microprocessor – operations, instructions and its addressing modes.

CIT0300204: MATHEMATICS II

1. Learning Outcomes: After successful completion of this course, students will be able to:

a) Learn the basic concepts of limit, continuity and derivatives.
b) Understand graphs and its different representations in Computers. How to model real life problems using graphs. Learn a few basic graph traversal algorithms.
c) Understand the basic idea of counting and use it in counting under various constraints.
d) Understand Mathematical Logic from algorithmic point of view.

2. Prerequisites: NIL

3. Semester: 2

4. Course type: Compulsory

5. Course level: 200-299

6. Theory credit: 4

7. Practical credit: 0

8. Number of required hours:

a) Theory: 60 hrs (60 classes)

b) Practical: NIL

c) Non Contact: NIL

9. List of Reference Books:

a) J. P. Tremblay, R. Manohar, *Discrete Mathematics structures with applications to Computer Science,* Mc-Graw Hill.
b) N. Ch.SN Iyengar, K.A. Venkatesh, V. M. Chandrasekaran, P. S. Arunachalam, *Discrete Mathematics,* Vikash Publishing House Pvt Ltd.
c) C.L.Liu, *Elements of Discrete Mathematics,* Mc-Graw Hill International Ed.

10. Course Details:

A. Theory

UNIT 1: Calculus (15 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: Graph Theory (15 Lectures)

Basic Definition of graph, Directed, Undirected and Weighted Graphs. Representation of graphs in Computers – Adjacency Matrix and Adjacency Lists. Degree of vertices – in degree and out degree. Paths, Cycles and Acyclic graphs. Simple operations on graphs and amount of computations required for each operation. Connected graph, Tree and Forest. Bipartite graph, Algorithms on graph traversals- Breadth first search, Depth first search.

UNIT 3: Combinatorics **(15 Lectures)**

Basic of counting principles, principle of inclusion-exclusion, application of inclusion and exclusion, Pigeonhole principle, generalized Pigeonhole principle and its application, permutations and combinations, circular permutations, permutations with repetitions, combinations with repetitions, permutations of sets with indistinguishable objects.

UNIT 4: Mathematical Logic **(15 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.

CIT0300304: OBJECT ORIENTED PROGRAMMING USING C++

1. **Learning Outcomes:** After successful completion of this course, students will be able to:
   a) Differentiate between Structured programming and Object-Oriented Programming.
   b) Learn the concept of objects and develop the ability of imagining real life concepts as objects and derive their properties and functions to operate these objects.
   c) Develop programs using different object-oriented programming features such as data abstraction, polymorphism, inheritance, exception handling etc.
2. **Prerequisites:** NIL
3. **Semester:** 3
4. **Course Type:** Compulsory
5. **Course Level:** 200-299
6. **Theory Credit:** 3
7. **Practical Credit:** 1
8. **No of required hours:**
   a) Theory: 45 hrs
   b) Practical: 30 hrs
   c) Non Contact: 5 hrs
9. **List of Reference Books:**
   a) M. T. Somashekara, D. S. Guru et-al, *Object-Oriented Programming with C++,* 2nd Edition, PHI, 2012.
   b) Bjarne Stroustrup, *The C++ Programming Language,* Special Edition*,* Pearson Education, 2004.
   c) Deitel & Deitel, *C++ How to program,* Pearson Education Asia, 6th Edition, 2008.
   d) Schildt Herbert, *The Complete Reference C++,* Tata McGraw Hill, 4th Edition, 2003.
10. **Detailed Syllabus:**

A. Theory

UNIT 1: Introduction to Object-Oriented Programming (3 Lectures)

Basic Concepts of Object-Oriented Programming and design, Benefits and applications of OOP.

UNIT 2: Introduction to C++ (6 Lectures)

Structure of a Simple C++ program, Output operator, Input operator, Cascading of I/O operators, Tokens- keyword, identifiers, constants, strings and operators. Basic data types, User defined data types, Dynamic initialization of variables, Reference variables, Operators in C++, Scope resolution operator & applications, Member dereferencing operators, Memory Management operators, new and delete, Control Structures-simple if, if else, nested if, switch, while do, break and continue statements, Introduction to Functions-Function Prototyping, Call-by-reference, Return by reference, Inline functions, Default arguments, Const arguments.

UNIT 3: Classes and Objects (11 Lectures)

Introduction - Defining a class; class versus structures, creating objects, accessing class members, defining member functions- outside the class definition and inside the class definition, outside functions as inline. Nesting of member functions, private member functions, memory allocation for objects. Array-declaring an array, accessing elements of an array, array of objects. Friendly functions.

Basic Concepts of constructors and destructors with examples. Default constructor, Parameterized constructor, multiple constructors in a class. Constructor with default arguments, Copy constructor. Dynamic initialization of objects. Dynamic constructors and destructors.

UNIT 4: Function and Operator Overloading (10 Lectures)

Concept of Overloading. Function Overloading: Functions with different sets of parameters, default and constant parameters, Rules for overloading operators, defining operator overloading. Overloading unary operators -prefix and postfix operators. Overloading Binary operators and relational operators, Overloading using friend functions.

UNIT 5: Inheritance (12 Lectures)

Concept of Inheritance -defining derived classes. Types of inheritances, Making a private member inheritable, multilevel inheritance, multiple inheritance, Hierarchical inheritance, Hybrid inheritance, Virtual base classes, Abstract classes, Constructors in derived classes, nesting of classes, polymorphism-Compile time and Runtime polymorphism, Pointers to objects, "this" pointer, Pointer to derived classes, Virtual functions, Rules for virtual functions, Pure virtual functions.

UNIT 6: Exception handling (3 Lectures)

Examples of exceptions and handling exceptions using try, catch and throw statements.

B. List of Practical

Following Practical / Lab works to be performed preferably in Linux Environment

1. Define a class named "triangle" to represent a triangle using the lengths of the three sides. Write a constructor to initialize objects of this class, given the lengths of the sides. Also write member functions to check
   (a) if a triangle is isosceles
   (b) if a triangle is equilateral
   Write a main function to test your functions.
2. Define a structure "employee" with the following specifications.
   empno : integer
   ename : 20 characters
   basic, hra, da : float
   calculate() : a function to compute net pay as basic+hra+da with float return type.
   getdata() : a function to read values for *empno*, ename, basic, hra, da.
   dispdata() : a function to display all the data on the screen
   Write a main program to test the program.

3. Define a class "circle" to represent circles. Add a data member radius to store the radius of a circle. Write member functions *area*() and *perimeter*() to compute the area and perimeter of a circle.

4. Define a class "complex" with two data members "real" and "imag" to represent real and imaginary parts of a complex number. Write member functions
   rpart() : to return the real part of a complex number

ipart() : to return the imaginary part of a complex number
add() : to add two complex numbers.
mul() : to multiply two complex numbers.
Write constructors with zero, one and two arguments to initialize objects.

5. Define a class "point" with two data members "*xordinate*" and "*yordinate*" to represent all points in the two-dimensional plane by storing their x co-ordinate and y co-ordinate values. Write member functions

dist(): to return the distance of the point from the origin.
slope(): to return the slope of the line obtained by joining this point with the origin.

Write constructors with zero, one and two arguments to initialize objects. Also write a friend function to compute the distance between two points.

6. Define a class "string" with the following data members char *p; int size; and write member functions to do the following (without using library function) and using dynamic memory allocation.

- Length of the string
- Compare two strings
- Copy one string to another
- Reverse the string

Write suitable constructors and destructors. Also write a copy constructor for the class.

7. For the class "complex" defined in 4 above, overload the <<, >>, + and * operators in the usual sense. Also overload the unary – operator.

8. Define a class "time" to store time as hour, minute and second, all being integer values. Write member functions to display time in standard formats. Also overload the ++ and – operators to increase and decrease a given time by one second where the minute and hour values will have to be updated whenever necessary.

9. Define a class to store matrices. Write suitable friend functions to add and multiply two matrices.

10. Write a class-based program implementing static members.

11. Define a class student with the following specification:
rollno : integer sname : 20 characters
Derive two classes *artst* and *scst*. The class *artst* will represent students belonging to arts stream and the class *scst* will represent students belonging to science stream. The *artst* class will have additional data members *ph*, *hs*, *en* and as to store marks obtained by a student in three subjects Philosophy, History, English and Assamese. The class scst will have additional data member *sph*, *ch*, *ma* and *en* to store marks obtained in *Physics*, *Chemistry*, *Mathematics* and *English*.
Write the following member functions in the classes *artst* and *scst*; *ctotal*() : a function to calculate the total marks obtained by a student; *takedata*() : a function to accept values of the data members and show *data*() : a function to display the marks sheet of a student .

12. Define an abstract base class printer. Derive three classes' laser-printer, line-printer and inkjet-printer. The derived classes will have data members to store the features of that particular printer. Write pure virtual function *display*() in the base class and redefine it in the derived classes.

13. Define a abstract base class figure and add to it pure virtual functions

display() : to display a figure
get() : to input parameters of the figure
area() : to compute the area of a figure
perimeter() : to compute the perimeter of a figure.

Derive three classes circle, rectangle and triangle from it. A circle is to be represented by its radius, rectangle by its length and breadth and triangle by the lengths of its sides. Write again function and write necessary statements to achieve run time polymorphism.

14. Write an interactive program to compute square root of a number. The input value must be tested for validity. If it is negative, the user defined function *my_sqrt*() should raise an exception.

CIT0400104: DATABASE MANAGEMENT SYSTEM

1. **Learning Outcome:** On successful completion of this course, the student should be able to:
   a) Learn database concepts and its architectural components.
   b) Describe different data models used for designing a database.
   c) To create a database using relational models and entity relationships concepts
   d) Normalize a database into various normal forms
   e) Design SQL queries to handle a relational database.
2. **Prerequisite:** NIL
3. **Semester:** 4
4. **Course Type:** Compulsory
5. **Course Level:** 200-299
6. **Theory Credit:** 3
7. **Practical Credit:** 1
8. **Number of required hours:**
   a) Theory: 45 hrs
   b) Practical: 30 hrs
   c) Non Contact: 5 hrs

9. List of Reference Books:

a) Dr. Satinder Bal Gupta and Aditya Mittal, *Introduction to Database Management System,* University Science Press.
b) A. Silberschatz, H.F. Korth, S. Sudarshan, *Database System Concepts*, McGraw Hill.
c) R. Elmasri, S.B. Navathe, *Fundamentals of Database Systems*, Pearson Education.
d) Dr. Rajive Chopra, *Database Management System (DBMS): A Practical Approach*, S. Chand Publication.

10. Detailed Syllabus:

A. Theory

UNIT 1: Introduction to Database Management Systems (5 Lectures)

Basic Definition and Concepts: *Data, Information, Meta Data, Data Dictionary, Database, Fields*, *Records* and *Files*. Definition of Database Management System (DBMS), Primary Functions of DBMS, Traditional File approach, Traditional file approach versus database management system approach, Disadvantages of Traditional File System, Need of a DBMS, Components of a DBMS, Advantages of DBMS, Disadvantages of Database Systems, Various uses of database System Applications, Database Users: *End users or naive users*, *Onlineusers*, *Application Programmers*, *Database Administrator* (*DBA*), Responsibilities of DBA.

UNIT 2: Database Management System Architecture (6 Lectures)

Definition of *Schemas*, *sub-schema* and *Instances*. Data Independence: *Physical Data Independence* and *Logical data Independence*. Three-tier architecture of DBMS, Advantages of three-level Architecture, basic concept of data model, Characteristics of Data Models, Types of Data models: *Record Based Data Models*, *Object Based Data Model* and *Physical Data Models*. Relational Data Model, Types of database Systems:

Single-user database systems, *Multiuser* database systems, *Centralized* database systems, *Distributed* database systems and *Client/Server* database systems.

UNIT 3: E-R Modeling (8 Lectures)

Basic Concepts: *Entity*, *Attributes*, *Entity Sets*, *Domain*. Types of attributes: *Simple and Composite Attributes*, *Single Valued and Multi-valued Attributes*, *Derived Attributes and Stored Attributes*. Types of Entity Sets*: Strong Entity Sets* and *Weak Entity Sets.* Concept of Relationship and Relationship sets, Types of Relationship: One-to-One, One-to-Many, Many-to-One and Many-to Many, Various Symbols used in ER Diagram, Mapping constraints: *Mapping Cardinalities (Cardinality Ratios)* and *Participation Constraints*. Definition of Key, Types of Keys: *Super Key*, *Candidate Key, Primary Key, Alternate Key* and *Foreign Key*. Symbols used in E-R diagrams, Conversion of an ER and Diagram in to Relational Tables

UNIT 4: Relational Model and Relational Algebra (7 Lectures)

Definition of Relation, Data Structure of Relational Database: *Relation*, *Tuples*, *Attributes Domain, Degree* and *Cardinality.* Integrity Constraints, Domain Constraints, Key Constraints, Advantages and Disadvantages of Relational Model, Relational, Definition of Relational algebra, Operations in Relational Algebra: *Selection*, *Projection, Division, Rename, Union, Intersection, Set Difference, Natural-join operation, Outer join, Inner Join, Cartesian Product* and *Assignment operation*. Aggregate Functions and Operations: *Average, Maximum, Minimum, Sum* and *Count*.

UNIT 5: Functional Dependency and Normalization (8 Lectures)

Definition of Functional Dependency, Armstrong's Axioms in Functional Dependency, Types of Functional Dependency: *Partial Dependency, Full Functional Dependency, Transitive and Non-transitive Functional Dependency*, Armstrong's Axiom, Closure of a set of Functional Dependency, Closure of an Attribute, Definition of Canonical Cover, Algorithm to find the canonical cover of a FD set, Anomalies in relational database*: Insertion*, *Deletion* and *Update* anomalies, Concepts of Normalization, Benefits of Normalization, Types of Normal Forms: First Normal Form (1NF), Second Normal Form (2NF), Third Normal Form (3NF) and Boyce–Codd Normal Form (BCNF)

UNIT 6: Transaction and Concurrency Control (4 Lectures)

Definition of Transaction, ACID Properties of transaction, Transaction States, Definition of Concurrency Control, Need of Concurrency Control, The Lost Update Problem, The Uncommitted Dependency Problem, The Inconsistent Analysis Problem, Serializability: *View Serializability* and *Conflict Serializability*

UNIT 7: SQL Queries (7 Lectures)

Database Languages (Data Definition Languages, Data Manipulation Languages), Characteristics of SQL, Basic data types in SQL, Data-definition language (DDL) commands: *Create Database, Create Table*, *Drop Table*, *Alter Table*. SQL Constraints: *Primary Key, Foreign Key, Not Null, Unique, Check, Default,* .Data Manipulation Language (DML) commands: *Insert Into*, Delete, *Select*, *Update*. SQL clauses: *Where, Order By, Having, Group By* and *Like*. SQL join operations: *Inner Join, Left Outer Join, Right Outer Join* and *Full Join*. SQL aggregate functions: *sum(), count(), max(), min()* and *avg()*

B. List of Practical

Practical / Lab work to be performed:

1. Implementation of SQL DDL statements in MySQL DBMS: CREATE DATABASE, CREATE TABLE, ALTER TABLE, RENAME, DROP DATABASE/TABLE
2. Use of SQL DML statements in MySQL DBMS: INSERT, SELECT, UPDATE, DELETE SQL commands
3. Implementing following constraints in MySQL DBMS: PRIMARY KEY, FOREIGN KEY, NOT NULL, UNIQUE and DEFAULT
4. Handling following SQL clauses in MySQL DBMS: WHERE, GROUP BY, ORDER BY, HAVING, IN, BETWEEN, LIKE
5. Working with following aggregate functions in MySQL DBMS: COUNT, AVG, MAX, MIN and SUM
6. Working with transaction processing command in MySQL DBMS: START TRANSACTION, COMMIT and ROLLBACK Statements, SET auto commit

CIT0400204: OPERATING SYSTEM

1. Learning Outcomes: After successful completion of this course, students will be able to:

a) After completing this course, students will have understanding of the internal structure and usage of various components related to an operating system.

2. **Prerequisites**: NIL

3. Semester: 4

4. Course Type: Compulsory

5. Course Level: 200-299

6. Theory Credit: 3

7. Practical Credit: 1

8. No of required hours:

a) Theory: 45 hrs

b) Practical: 30 hrs

c) Non Contact: 5 hrs

9. List of Reference Books:

a) Abraham Silberschatz, Peter B. Galvin, *Operating System Concepts*, Greg Gagne,Wiley

b) Andrew S. Tanenbaum, *Modern Operating Systems*, Prentice-Hall Of India Pvt. Limited

10. Contents of Syllabus:

A. Theory

UNIT 1: Introduction (7 Lectures)

Application vs system software, operating system as system software, operating structure structure, types of operating systems: batch operating system, multiprogramming operating system, multitasking operating system, distributed operating system, real time operating system, multi user operating system, major functions of operating system: Process Management, Process Synchronization, Memory Management, CPU Scheduling, File Management, I/O Management, Security, virtualization, cloud computing, open source operating system, history of operating system, the shell, system call, system boot

UNIT 2: Process and threads (10 Lectures)

Process, process states: new, running, waiting, ready and terminated, Process Control Block (PCB),information stored in PCB, scheduling queue: job queue, ready queue and device queue, schedulers: long term schedulers, medium term scheduler and long term scheduler, swapping, degree of multiprogramming, I/O-bound and CPU-bound processes, context switching, inter-process communication: shared memory systems and message passing systems, socket, remote procedure call, threads, user threads, kernel threads, multi threading models: Many-to-One Model, One-to-One Model, Many-to-Many Model, CPU scheduling, Scheduling Criteria, scheduling algorithms: First-Come, First-Served Scheduling, Shortest-Job-First Scheduling, Priority Scheduling, Round-Robin Scheduling, Multilevel Queue Scheduling, Multilevel Feedback Queue Scheduling

UNIT 3: Process synchronization (8 Lectures)

Race condition, critical section problem, Peterson's algorithm, Bakery algorithm, synchronization hardware: locking, synchronization software tools: mutex lock, semaphore (counting and binary), semaphore

implementation, classic synchronization problems: bounded buffer problem, the readers–writers Problem, the dining-philosophers problem, monitor, synchronization in windows, synchronization in Linux

UNIT 4: Deadlock (10 Lectures)

Deadlock, operations of a process performs while using a resource: Request. Use and Release, physical and logical resources, Necessary conditions: mutual exclusion, hold & wait, no preemption and circular wait, resource allocation graph, deadlock prevention: definition, preventing mutual exclusion, preventing hold & wait, preventing no preemption and preventing circular wait, deadlock avoidance: definition, safe state, safe sequence, resource allocation graph based algorithm and Banker's algorithm, deadlock detection: definition, wait-for graph, algorithm to detect deadlock for single instance resources, algorithm to detect deadlock for multiple instance resources and recovery from deadlock: process termination and resource preemption

UNIT 5: Memory Management (10 Lectures)

Memory hierarchy, base register, limit register, address binding, logical and physical address spaces, memory management unit, relocation register, swapping, contiguous memory allocation: definition, memory protection, fixed partition scheme, variable partition scheme, first-fit, best-fit &worst-fit allocation strategies, non-contiguous memory allocation: simple paging and simple segmentation, internal and external fragmentation, TLB, virtual memory, demand paging, page fault, locality of reference principle, performance of demand paging, page replacement algorithms: FIFO, Optimal and LRU, allocation of frames: equal allocation and proportional allocation, global and local page replacement algorithms, thrashing

B. List of Practical

1. Basic linux commands: pwd, ls, cd, mkdir, rmdir, rm, touch, man, cp, mv, locate, head, tail 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
2. Shell scripting in linux: shell, types of shell, shell script, echo command, shell variables,
3. special variables ($$, $0, $n, $#, $?, $!), array, assignment operator (=), equality operator (==), not equality operator (!=), arithmetic operators (+,-, *, /, %), comparison operators (-eq, -neq, -gt, -lt, - ge, -le), logical operators (!,-o, -a), if...else statement, case...esac statement, while loop, for loop, break statement, continue statement, shell functions 7 classes
4. Using system calls in C program in linux: fork(), exec(), exit(), getpid(), mkdir(), rmdir() etc.

COM0400304: AUTOMATA THEORY AND LANGUAGES

1. Learning Outcome: After completing this course, students

- Understand the Mathematical model of a finite state machine. Know deterministic and non-deterministic versions of Finite automata.
- Grasp the mathematical concepts of languages and grammar.
- Know Pushdown Automata and the associated grammar/language.
- Know the properties of Regular languages and Context free languages.

2. Prerequisites: NIL

3. Semester: 4

4. Course Type: Compulsory

5. Course Level: 200-299

6. Theory Credit: 4

7. Practical Credit: 0

8. No of Hours:

a) Theory: 60 hrs
b) Practical: 0 hrs
c) Non Contact: 5 hrs

9. List of Books:

a) Peter Linz, *An introduction to Formal Languages and Automata*, Narosa.
b) Hopcroft, Motwani and Ullman, *Introduction to Automata Theory, Languages and Computation*, Pearson.
c) K. L. P. Mishra, N. Chandrasekaran, *Theory of Computer Science (Automata, Languages and Computation*), P. H.I.

10. Contents of Syllabus:

A. Theory

UNIT 1: Finite Automata (10 Lectures)

DFA, NFA, NFA with empty-moves, Equivalence of DFA and NFA, Reduction of the number of states in finite automata.

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

Concept of languages and grammar, Regular expressions, Connection between regular expressions and regular languages, Regular grammars, Right and Left-Linear Grammars, Equivalence between Regular languages and Regular grammars.

UNIT 3: Properties of Regular Languages (13 Lectures)

Closure under simple set operations- union, intersection, concatenation, complementation and star closure, Decision algorithms for emptiness, finiteness and infiniteness, equality, Proof of non- regularity using Pigeonhole principle and using pumping lemma for regular languages.

UNIT 4: Context Free languages (15 Lectures)

Context-free grammars, 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. Normal forms- Chomsky and Greibach normal forms, Pumping Lemma for CFL, Using Pumping Lemma to show that certain languages are not Context free.

UNIT 5: Pushdown Automata **(10 Lectures)**

Definition and language accepted (acceptance by empty stack and final state and their equivalence), Pushdown Automata and Context free languages. Deterministic PDA and Deterministic Context free Languages.

CIT0400404: PYTHON PROGRAMMING

1. Learning Outcome: After completing this course, students

a) Know about fundamentals of Python Programming and Problem Solving.

2. Prerequisites: NIL
3. Semester: 4
4. Course Type: Compulsory
5. Course Level: 200-299
6. Theory Credit: 3
7. Practical Credit: 1
8. No of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

9. List of Books:

a) R. Nageswara Rao, *Core Python Programming*, Dreamtech Press.
b) Martin C. Brown, *Python: The Complete Reference*, McGraw Hill Education.
c) http://docs.python.org/3/tutorial/index.html

10. Contents of Syllabus:

A. Theory

UNIT 1: Introduction to Python Programming (7 Lectures)

Introduction, Installation of Python Interpreter, Python Shell, Code Indentation, Identifiers and Keywords, Literals, Strings, Operators (Arithmetic, Relational, Logical, Assignment, Ternary, Bitwise, Increment and Decrement Operators), Input and output statements, Output Formatting.

UNIT 2: Control Statements and Functions (7 Lectures)

Branching, Looping, Conditional Statement, Exit Functions, Break, Continue, Pass, Defining Functions, Default Arguments. Scope of Functions, Function Documentation, Lambda Functions & Map.

UNIT 3: Python Data Structures (6 Lectures)

List (List, Nested List, List as Matrix), Tuple, Set, Dictionary.

UNIT 4: Exception Handing (4 Lectures)

Errors, Exception Handling with try, Multiple Exception Handling, Writing own Exception.

UNIT 5: File Handling (6 Lectures)

Understanding read function, read(), readline() and readlines(), Understanding write functions, write() and writelines(), Programming using file operations, Reading config files, Writing log files in python.

UNIT 6: OOP in Python (3 Lectures)

Creating Classes in Python, Instance Methods, Inheritance, Polymorphism, Exception Classes and Custom Exceptions.

UNIT 7: Introduction to Libraries in Python **(5 Lectures)**

NumPy, Matplotlib, OpenCV, Tkinter.

UNIT 8: Python SQL Database Access **(7 Lectures)**

Introduction to database driven program, Database Connection, Database Operations: INSERT, READ, UPDATE, DELETE, COMMIT AND ROLLBACK.

(B) List of Practical

1. Introduction to Python console, operators, input and output statements.
2. Python control statements and functions
3. Data Structures in python
4. Exception Handling
5. File Handling
6. Object Oriented Python programming
7. Introduction to libraries (NumPy, Matplotlib, OpenCV)
8. Python SQL Database Connection and database operations

CIT0400504: SYSTEM SOFTWARE

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. Course Level: 200-299

6. Theory Credit: 3

7. Practical Credit: 1

8. No. of Hours:

Theory: 45 hrs (45 classes)

Practical: 30 hrs (15 classes)

Non Contact: NIL

9. List of Reference Books:

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

b) Dhananjay Dhamdhere, *Systems Programming,* Mc Graw Hill Education

10. Content of Syllabus:

A. Theory

UNIT 1: Introduction to Operating System (10 Lectures)

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.

UNIT 2: Assemblers (12 Lectures)

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 3: Loaders and Linkers (7 Lectures)

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

UNIT 4: Macro processor (6 Lectures)

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 5: Compilers (10 Lectures)

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

B. List of Practical

a. Introduction to NASM assembler.
b. Introduction to segments and registers.
c. A simple assembly program to print hello.
d. Input and output in assembly language.
e. Conditional statements in assembly language.
f. Looping in assembly language.
g. An assembly language program that accepts two numbers from the user and displays sum of the numbers.
h. An assembly language program that changes case of accepted characters.
i. An assembly program that accepts a number and displays whether the number is odd or even.
j. An assembly program that accepts a number n from the user and displays “hello world” n number of times.
k. An assembly program that accepts a number from the user and displays factorial of the number.
l. An assembly program that accepts a number n from the user and displays whether the number is prime.

CIT0500104: SOFTWARE ENGINEERING

1. Learning Outcome: On successful completion of this course, the student should be able to:

a) Determine the primary problems that impact all software development processes.
b) Choose relevant software development processes models, methodologies, and strategies for managing a specific software development process, and justify the choices
c) Implement different software estimation metrics such as cost, effort size, staffing etc.
d) Describe various software design approaches and various coding and testing strategies used in software engineering principles
e) Know about software reliability and how to calculate software maintenance cost

2. Prerequisites: NIL
3. Semester: 5
4. Course Type: Compulsory
5. Course Level: 300-399
6. Theory Credit: 4
7. Practical Credit: 0
8. No of Hours:

a) Theory: 60 hrs
b) Practical: 0 hrs
c) Non Contact: 5 hrs

9. 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.

10. Contents of Syllabus:

A. Theory

UNIT 1: Introduction (4 Lectures)

Definition of Software Engineering, differentiation between Computer Science, Software Engineering and System Engineering, Program V/s software product, Exploratory style and modern style of software development, need of software engineering, characteristics of good software product

UNIT 2: Software Development Life Cycle models (7 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: Requirement Analysis and Specification (7 Lectures)

What is Requirement Analysis and Gathering, Concept and Importance of Feasibility Study in Software design, Types of Feasibility: *Technical*, *Economical* and *Operational* feasibility, Software Requirement Specification (SRS) document, Components of an SRS (Software Requirement Specification): Functional and Non-Functional Component, Properties of a good SRS, Different users of SRS, Techniques to represent Complex Logic in SRS: Decision Tree and Decision Table.

UNIT 4: Software Project Management (15 Lectures)

Basic idea of Software Project Management, Job Responsibilities of a Software Project Manager, Need of SPMP (Software Project Management Plan) document, Contents of SPMP, Need of Software documentation, Internal and External documentation, Software size estimation using Lines of Code (LOC), Merits and Demerits of LOC metric, Function Point Metric, 3D Function Point metrics, Project Estimation Techniques: *Empirical estimation* and *Heuristics estimation* techniques. Empirical estimation techniques: *Delphi Cost Estimation* and *Delphi Cost Estimation.* Heuristic Estimation Techniques: *Basic COCOMO model* and *Intermediate COCOMO model.* Project Scheduling: *Work break down structure, Activity Networks* and *Critical Path Method.* Project Team structure: *Chief Programmer team* and *Democratic team* structure.

UNIT 5: Software Design principles and Methodology (12 Lectures)

Top down and bottom up approach, External Design, Architectural Design and Detailed design, Concept of Cohesion in software design, Classification of Cohesions, Basic concept of Coupling, Classification of Couplings, Introduction to software Analysis and Software Design (SA/SD), Introduction to Data Flow Diagram, Symbols used in DFD, Context Diagram in DFD, Advantages and Disadvantages of DFDs., Balanced DFD, Structured Design: *Transaction Analysis* and *Transform Analysis.* Need of Object Oriented Design and Analysis, UML (Unified Modeling Language), different views of UML, Various UML Diagrams: *Use Case diagram, Class Diagram, Object Diagram*, *Sequence Diagram* and *Collaboration diagram.*

UNIT 6: Coding and Testing (9 Lectures)

Goals of coding, Code Review techniques: Code Walkthrough, Code Inspection, Definition of Test cases, test suits, negative testing and positive testing. Different levels of software testing: *unit testing*, *Integration Testing*, *System Testing* and *acceptance testing*. Differentiation between Verification and Validation, Black box testing approaches: *Equivalent Class Partitioning* and *Boundary Value Analysis*, White Box testing approaches: *Statement Coverage*, *Branch Coverage, Condition* Coverage and Path *Coverage. Approach*, MaCabe's Cyclomatic Complexity, Basic idea of various system testing approaches: *Smoke testing, Stress testing, Volume testing* and *Compatibility testing*

UNIT 7: Software Reliability and Maintenance (6 Lectures)

What is reliability? Reliability metrics of Software Products: ROCOF, MTTF, MTTR, MTBF, POFOD and availability. ISO 9000 Certification, need of ISO Certification, How to get ISO 9000 certification, Definition of Software Maintenance, Types of Software maintenance: *Corrective, Adaptive* and *Perfective* maintenance, Estimation of Software Maintenance Cost.

CIT0500204: JAVA PROGRAMMING

1. **Learning Outcome:** After completing this course, students will be
   a) Familiar with the core concepts of java programming and classes of swing package.

2. Prerequisites: NIL
3. Semester: 5
4. Course Type: Compulsory
5. Course Level: 300-399
6. Theory Credit: 3
7. Practical Credit: 1
8. No of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

9. List of Books:

a) Herbert Schildt, *Java: The Complete Reference*, McGrawHill
b) Paul Deitel, *Java How to Program,* Harvey Deitel, Pearson

10. Contents of Syllabus:

A. Theory

UNIT 1: Introduction (3 Lectures)

High level language, compiled and interpreted languages, history of java programming language, compilation of java code, bytecode, java interpreter, javac and java command, path environmental variable, Java IDE, features of java programming language: simple, object oriented, robust, architecture neutral and interpreted

UNIT 2: Data types, operators and control statements (12 Lectures)

Java as strongly typed language, primitive data types, integer data types: byte, short, int and long, floating point data types: float and double, character data type, boolean data type, literals: integer literals, floating-point literals, boolean literals, character literals and string literals, declaring a variable, dynamic Initialization, the scope and lifetime of variables, type-casting in java, one dimensional array, multi dimensional array, arithmetic operators: the basic arithmetic operators, the modulus operator, arithmetic compound assignment operators, increment operator and decrement operator, bitwise operators, relational operators, short circuit logical operator, the assignment operator, branching statements: if-else and switch-case statements, looping statements: while, do-while, for and for-each statements, jump statements: break and continue

UNIT 3: Object oriented features of java (10 Lectures)

Defining a class, member variable and member methods, access specifies: default, private and public, declaring objects, assigning object reference variables, constructors, parameterized constructors, the this keyword, garbage collection, the finalize( ) method, overloading methods, overloading constructor, static keyword, final keyword, command line arguments in java, inheritance, super class and sub class, protected access specified, super keyword, constructor call in multilevel inheritance, method overriding, dynamic method dispatch, abstract class, interfaces, type wrappers

UNIT 4: String handling and packages (5 Lectures)

String class, String constructors, String length, special string operations: string literals, string concatenation, string concatenation with other data types, string conversion and toString( ), character extraction: charAt( ), getChars( ), string Comparison: equals( ) and equalsIgnoreCase(), regionMatches( ), startsWith( ) and endsWith( ), equals( ) Versus ==, compareTo( ), searching strings, data conversion using valueOf( ), StringBuffer, StringBuffer constructors, length( ) and capacity( ), ensureCapacity( ), setLength( ), charAt( ) and setCharAt( ), getChars( ), package, defining a package, CLASSPATH, importing packages

UNIT 5: Exception handling and I/O (5 Lectures)

Exception-handling, exception types, uncaught exceptions, try and catch block, multiple catch blocks, nested try statements, throw, throws, finally, java's built-in exceptions, creating own exception classes, java I/O classes, reading console input, writing console output, reading and writing files

UNIT 6: Swing package and database connectivity (10 Lectures)

Swing package, simple GUI-Based Input/Output with JoptionPane, Jframe, JLabel, JTextField, Jbutton, handling event in a Jframe object, layout managers: BorderLayout, FlowLayout, GridLayout, CardLayout, GridBagLayout, JtoggleButton, JCheckBox, JRadioButton, Jlist, JcomboBox, JDBC, JDBC driver, connectivity steps, connectivity with MySQL, DriverManager class, Connection class, Statement class, Result Set class, Prepared Statement class

B. List of Practical

a) Java programs to demonstrate the use of data types and operators
b) Java input through Scanner class and JOptionPane class
c) Java programs to demonstrate the use of control statements.
d) Java programs to demonstrate the use of classes, objects, visibility modes, constructors and destructor.
e) Java programs to demonstrate the use of inheritance and polymorphism.
f) Java programs to demonstrate the use of polymorphism.
g) Java programs to handle strings, Java programs implementing exception handling.
h) Demonstrating the use and creation of packages in java.
i) Java program with JFrame, JTextfield and JButton with event handling
j) Using JLabel, JTextArea and JPasswordField in java with event handling
k) Working with layout managers in JFrame
l) Using JCheckBox, JRadioButton and JComboBox in a JFrame
m) Connecting JFrame components to a DBMS

CIT0500304: COMPUTER NETWORKS

1. Learning Outcome: After completing this course, students

a) Student will able to learn about the general principles of data communication.
b) Student will able to learn about how computer networks are organized with the concept of layered approach.
c) Student will able to learn about how signals are used to transfer data between nodes.
d) Student will able to learn about how packets in the Internet are delivered.
e) Student will able to learn about how routing protocols work.
f) Student will able to learn about functions of transport layer
g) Student will able to learn about functions of application layer

2. Prerequisites: NIL

3. Semester: 5

4. Course Type: Compulsory

5. Course Level: 300-399

6. Theory Credit: 3

7. Practical Credit: 1

8. No of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

9. List of Books:

a) B. A. Forouzan: *Data Communications and Networking*, Fourth edition, THM, 2007.
b) A. S. Tanenbaum: *Computer Networks***,** Fourth edition**,**PHI , 2002.

10. Contents of Syllabus:

A. Theory

UNIT 1: Introduction to Computer Networks (5 Lectures)

Data communication system and its components, Definition of network, Types of network, Network topologies, Network protocol, Layered network architecture, Overview of OSI reference model, Overview of TCP/IP protocol suite.

UNIT 2: Physical Layer Communication (10 Lectures)

Analog and digital signal, Definition of bandwidth, Maximum data rate of a channel, Line encoding schemes, Transmission modes, Modulation techniques, Multiplexing techniques- FDM and TDM, Transmission media- Guided and Unguided, Switching techniques- Circuit switching, Packet switching, Connectionless datagram switching, Connection-oriented virtual circuit switching.

UNIT 3: Data Link Layer Functions and Protocol (10 Lectures)

Definition of Framing, Framing methods, Error detection techniques, Error correction techniques, Flow control mechanisms- Simplex protocol, Stop and Wait ARQ, Go-Back-N ARQ, Point to Point protocol.

UNIT 4: Multiple Access Protocol and Networks (5 Lectures)

Basics of ALOHA protocols, Basics of CSMA/CD protocols, Ethernet LANS, Connecting LAN and back-bone networks- Repeaters, Hubs, Switches, Bridges, Router and Gateways

UNIT 5: Networks Layer Functions and Protocols (8 Lectures)

Connection oriented vs. Connectionless services, Definition of Routing, Routing algorithms, IP protocol, IP addresses, ARP, RARP

UNIT 6: Transport Layer Functions and Protocols (4 Lectures)

Transport services, TCP vs. UDP protocol, TCP connection establishment- Three way handshakes, TCP connection release

UNIT 7: Overview of Application Layer Protocols (3 Lectures)

Overview of DNS, Overview of WWW, URL, Email architecture, HTTP protocol

B. List of Practical

1. Implement the data link layer farming methods such as Bit Stuffing.
2. Study of different types of Network cables.
3. Study of network IP.
4. Connect the computers in Local Area Network.
5. Study of basic network command and Network configuration commands.
6. Socket programming in C language.
7. Configure a Network topology using packet tracer software.
8. Simulate Cyclic Redundancy Check (CRC) error detection algorithm for noisy channel.
9. Simulate and implement Stop and Wait protocol for noisy channel.
10. Simulate and implement Go-Back-N sliding window protocol.
11. Simulate and implement Selective Repeat sliding window protocol.
12. Simulate and implement Dijkstra Algorithm for shortest path routing.
13. Simulate and implement Distance vector routing algorithm

CIT0500404: INFORMATION SECURITY AND CYBER LAWS

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

2. Prerequisites: NIL
3. Semester: 5
4. Course Type: Elective
5. Course Level: 300-399
6. Theory Credit: 4
7. Practical Credit: 0
8. No. of Hours:
9. List of Books:

(a) Merkow, M., & Breithaupt, J., *Information Security Principles and Practices,* 5th edition. Prentice Hall.
(b) William Stallings, *Cryptography and Network Security Principles and Practice,* Fourth or Fifth Edition, Pearson Edition.
(c) Advocate Prashant Mali, *Cyber Law & Cyber Crimes,* Snow White Publications, Mumbai
(d) *The Information Technology Act,* Bare Act–Professional Book Publishers, New Delhi

10. Contents of 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 Oveflow, 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:

- [Section43] Penalty and compensation for damage to computer etc.
- [Section65] Penalty for temping with the computers source documents.
- [Section66] Punishment for hacking with computer system, data alteration etc.
- [Section66A] Punishment for sending offensive messages through any communication services
- [Section66B] Receiving stolen computer's resources or communication devices dishonestly
- [Section66C] Punishment for identity theft.
- [Section66D] Punishment for cheating by impersonation by using computer resource.
- [Section66E] Punishment for violation of privacy.
- [Section66F] Punishment for cyber terrorism.
- [Section67] Punishment for publishing or transmitting obscene material in electronic form.
- [Section67A] Punishment for publishing or transmitting of material containing sexually explicit act, etc. in electronic form.

- [Section67B] Punishment for publishing or transmitting of material depicting children in sexually explicit act, etc. in electronic form.
- [Section72] Breach of confidentiality and privacy.

CIT0500504: COMPUTER ORIENTED NUMERICAL AND STATISTICAL METHODS

1. **Learning Objective:**
   The objective of this course is to provide students the understanding of basic numerical and statistical problems and to provide skills to solve these problems.
2. **Prerequisites:** NIL
3. **Semester:** 5
4. **Course Type:** Elective
5. **Course Level:** 300-399
6. **Theory Credit:** 3
7. **Practical Credit:** 1
8. **No. of Hours:**
   a) Theory: 45 hrs (45 classes)
   b) Practical: 30 hrs (15 classes)
   c) Non Contact: NIL
9. **List of Reference Books:**
   (a) Rajaraman,V, *Computer Oriented Numerical Methods,* 3rd edition, Prentice Hall
   (b) Balaguruswami, E., *Computer Oriented Statistical and Numerical Methods,* Macmillan Publishers India Limited
10. **Contents of Syllabus:**

A. Theory

UNIT 1: Introduction to Computer Arithmetic (7 Lectures)
Representation of numbers-Fixed Point and Floating point representations, Normalized Floating Representation, Floating Point Arithmetic, Properties of Floating Point, Numbers and their accuracy, Approximations and errors, Errors: truncation error, rounded off error, absolute error, relative error, percentage error and error propagation.

UNIT 2: Algebraic and Transcendental Equations (8 Lectures)
Introduction to linear and nonlinear equations, measures of accuracy, Properties of polynomial equations, Initial approximation to a root, Solution of algebraic/transcendental equations: Bisection Method, Iteration method, Method of false position, Newton-Raphson method, Rate of convergence of Iterative methods, Solution of simultaneous linear equations by using Gauss
elimination method.

UNIT 3: Interpolation (6 Lectures)
Polynomial Interpolation, Finite Differences, Newton's Forward Difference Interpolation, Newton's Backward Difference Interpolation, Newton's Divided Difference Interpolation

UNIT 4: Solution of Differential Equation (6 Lectures)
Taylor series method, Euler's method, Runge-Kutta method of 1st, 2nd & 4th order.

UNIT 5: Descriptive Statistic (6 Lectures)
Types of Data, Attributes and Variables, Construction of Frequency, Cumulative frequency, Graphical Representation of Frequency distribution: Histogram, Frequency Polygon, Frequency Curve and Cumulative Frequency Curves (Ogivecurves), Diagrammatic Representations: Simple bar, Subdivided bar, Pie Diagrams.

UNIT 6: Measure of Central Tendency (4 Lectures)
Measure of central tendency-Mean, Median and Mode, Measure of variation-Range, Inter quartile range, Standard Deviation and Variance.

UNIT 7: Probability Theory (8 Lectures)

Sample Space, events, random variables, Discrete probability, Conditional Probability and Bayes theorem, Linear Regression and Correlation, Probability Distribution Functions-Binomial, Random and Poisson.

B. List of Practical

Practical/Lab work to be performed using C/C++/Java programming Language

1. Apply the Bi-section method for approximation of root for a given polynomial equation.
2. Apply the False Position method for approximation of root for a given polynomial equation.
3. Implement Newton Raphson method for approximation of root for a given polynomial equation.
4. Implement Gausse limination method to solve simultaneous linear equations.
5. Develop programs to implement Newton's Forward Difference Interpolation.
6. Develop programs to implement Newton's Backward Difference Interpolation.
7. Develop programs to implement Newton's Divided Difference Interpolation.
8. Develop program to apply Taylor's series for e raise to the power x.
9. Implement Euler's method for solving a differential equation.
10. Implement Runge-Kutta method of 1^{st} , 2^{nd} & 4^{th} order for solving a differential equation.
11. Write programs to find Mean, Median and Mode for a given set of data.

CIT0500604: INTERNSHIP

1. **Learning Outcome:**
    a. Utilize programming, software development, or data analysis skills in a real-world environment.
    b. Identify and troubleshoot technical issues in projects or software systems.
    c. Assess the effectiveness of algorithms, frameworks, or tools used in the industry.
    d. Develop a functional project, application, or report demonstrating internship learnings.
2. **Prerequisite:** NIL
3. **Semester:** 5
4. **Course Type:** Compulsory
5. **Course Level:** 300-399
6. **Theory Credit:** 0
7. **Practical Credit:** 4
8. **Number of required hours:**
    a) Theory: 0 hrs
    b) Practical: 60 hrs
    c) Non Contact: 0hrs
9. **Course Content:**

9.1 Introduction

Students will have the opportunity to participate in summer internships with local industries, businesses, artists, and craftspeople. Additionally, they can engage in research internships with faculty members and researchers at their institution or other universities and research centers. These internships will allow them to apply their learning in real-world settings, gain hands-on experience, and enhance their employability.

9.2 Categories of internship

There will be two categories of internship:

1. **Internship for enhancing employability**:
   Minimizing the gap between theoretical knowledge gained from learning and practical IT skills, enabling graduates to acquire the necessary attributes to join the workforce. This effort may involve collaboration with IT companies, startups, and higher education institutions to develop web applications, mobile applications, cloud service systems, IoT and hardware systems, as well as real-world AI-based tools and blockchain systems.
2. **Internship for developing research aptitude:**
   Providing exposure to actual research environment and develop skills in research tools and techniques including theoretical computer science,

9.3 Duration and timing of internship: 1 Month

CIT0600104: COMPUTER GRAPHICS

1. Learning Outcome: After completing this course, students will know about-

a) Basic elements of Computer Graphics
b) Fundamental of Computer graphics algorithms along with basic mathematical foundations of computer graphics.

2. Prerequisites: NIL
3. Semester: 6
4. Course Type: Elective
5. Course Level: 300-399
6. Theory Credit: 3
7. Practical Credit: 1
8. No of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

9. List of Books:

a) D. Hearn, M. Baker, *Computer Graphics,* Prentice Hall of India 2008.
b) J.D.Foley, A.Van Dan, Feiner, Hughes, *Computer Graphics Principles & Practice*, 2nd edition Publication Addison Wesley 1990.
c) D.F.Rogers, *Procedural Elements for Computer Graphics*, McGraw Hill 1997.
d) D.F.Rogers, Adams, *Mathematical Elements for Computer Graphics*, McGraw Hill, 2nd edition 1989.

10. Contents of Syllabus:

A. Theory

UNIT 1: Introduction (2 Lectures)

Basic elements of Computer Graphics, Applications of Computer Graphics

UNIT 2: Graphics Hardware (5 Lectures)

Input Devices: Keyboard, Mouse, Trackball & Space ball, Joystick, Data Glove, Digitizers, Image Scanners, Touch panels, Light Pens systems. Output display devices: Refresh CRT, Raster-Scan display and Random-scan display technique, Color display techniques-Beam penetration method and Shadow-mask method, Direct view storage tubes, Emissive & Non-emissive flat-panel, Displays-Plasma panels, LED and LCD monitor, Three-dimensional viewing devices and Virtual-Reality systems Display processor: Raster-scan systems, Random-scan systems

UNIT 3: Fundamental Techniques in Graphics (20 Lectures)

Line-drawing algorithms: DDA algorithm and Bresenham's Line drawing Algorithm, Midpoint Algorithm for Circle and Ellipse Generation, Curve generation. Attributes for output primitives: Area-filling Algorithms - Scan-line Polygon-fill, 2-D Geometric Transformations: Basic transformations-translation, Rotation and Scaling Matrix representations and Homogeneous Co-ordinate representations, Composite transformations among translation, Rotation and Scaling, 2-D viewing: Definition, Viewing transformation pipeline, Window-to-viewport Co-ordinate transformation.

2-D Clipping: Concept and Algorithm: Point clipping, Line clipping - Cohen-Sutherland algorithm, Area clipping, Text clipping, Polygon clipping.3-D concepts: Display methods-Parallel projection, perspective projection3-D geometric transformations: Transformation, Translation, Rotation and Scaling around axes, 3-D Viewing Projections – Parallel and Perspective.

UNIT 4: Geometric Modelling (8 Lectures)

Representing curves and surface, Bezier curves and surfaces – Definition of Bezier curve and its properties, Algorithms for Bezier curves and surfaces, Hermite curve

UNIT 5: Visible Surface determination (5 Lectures)

Definition, approaches for visible surface detection, object-space methods- Back-Face Detection, Image space methods: Depth Buffer Methods, A Buffer Method, Scan Line Method, Depth-Sorting Method

UNIT 6: Surface rendering (5 Lectures)

Definition and importance, light sources, Basic illumination models-Ambient light, Diffuse reflection, Specula reflector and Phong model

B. List of Practical

1. Write a program to implement DDA algorithm for line drawing.
2. Write a program to implement Bresenham's line drawing algorithm.
3. Write a program to implement mid-point circle drawing algorithm.
4. Write a program to clip a line using Cohen-Sutherland line clipping algorithm.
5. Write a program to clip a polygon using Sutherland Hodgeman algorithm.
6. Write a program to apply 2D translation on a 2D object (use homogenous coordinates).
7. Write a program to apply 2D rotation on a 2D object (use homogenous coordinates).
8. Write a program to apply 2D scaling on a 2D object (use homogenous coordinates).
9. Write a program to apply 2D reflection of a 2D object (use homogenous coordinates).
10. Write a program to apply 2D shear operation on a 2D object (use homogenous coordinates).
11. Write a program to apply 3D translation on a 3D object (use homogenous coordinates).
12. Write a program to apply 3D rotation on a 3D object (use homogenous coordinates).
13. Write a program to apply 3D scaling on a 3D object (use homogenous coordinates).
14. Write a program to apply 3D reflection of a 3D object (use homogenous coordinates).
15. Write a program to apply 3D shear operation on a 3D object (use homogenous coordinates).
16. Write a program to draw Hermite/Bezier curve.

CIT0600204: OPTIMIZATION TECHNIQUES

1. **Learning Outcomes**
   a) Understand the need for optimization.
   b) Learn and apply different optimization techniques.
   c) Learn among different algorithms used for optimizations.
2. **Prerequisites:** NIL
3. **Semester:** 6
4. **Course Type:** Elective
5. **Course Level:** 300-399
6. **Theory Credit:** 4
7. **Practical Credit:** 0
8. **No of Hours:** 60
9. **List of books:**

   a) Gillette, B.G., *Introduction to operations research - A Computer oriented algorithmic approach*, McGraw Hill.
   b) N.S. Kambo, *Mathematical Programming Techniques*, EWP.
   c) K. V. Mital, *Optimization Methods*, Wiley Eastern, 3 rd Edition.
   d) G. Hadley, *Linear Programming*, Narosa Publications
   e) C. H. Papadimitriou, K. Steiglitz, *Combinatorial Optimization - Algorithms and Complexity*, Prentice Hall.

10. **Content of syllabus:**

A. Theory

UNIT 1: Unconstrained Optimization (15 Lectures)

Necessary and sufficient conditions for optima, convex sets, convex functions, optima of convex functions, steepest descent, Newton and quasi Newton methods, conjugate direction methods.

UNIT 2: Constrained Optimization (20 Lectures)

Linear Programming - Mathematical model, Basis, feasible solutions and basic feasible solutions, Graphical solution method, unboundedness, Simplex method, Revised simplex method, Applications, Duality, Dual simplex method, Primal Dual Algorithms. Complexity of the algorithms studied. Ellipsoid Method, Karmakar's algorithm.

UNIT 3: Special Models of Linear Programming Problems (15 Lectures)

Transportation and assignment problems, Maxflow and shortest path problems, Ford and Fulkerson algorithm, Djikstra's algorithm. Integer programming: Introduction, Travelling Salesman Problem (TSP), Branch and Bound techniques.

UNIT 4: Constrained Convex Optimization (10 Lectures)

Problem definition, Kuhn-Tucker Conditions and projected gradient methods.

CIT0600304: ARTIFICIAL INTELLIGENCE

1. Learning Outcome: After completing this course, students will know-

a) The fundamentals of artificial intelligence (AI),
b) Identify problems where artificial intelligence techniques are applicable.
c) Able to apply basic principles of AI in solutions that require problem solving, inference, perception, knowledge representation, and learning.

2. Prerequisites: NIL

3. Semester: 6

4. Course Type: Compulsory

5. Course Level: 300-399

6. Theory Credit: 3

7. Practical Credit: 1

8. No of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

9. List of Books:

a) Rich & Knight, *Artificial Intelligence* – Tata McGraw Hill, 2nd edition, 1991.
b) Russell &Norvig, *Artificial Intelligence*-A Modern Approach, LPE, PearsonPrentice Hall, 2nd edition, 2005.
c) W.F. Clocksin and Mellish, *Programming in PROLOG,* Narosa Publishing House,3rd edition, 2001.
d) DAN.W. Patterson, *Introduction to A.I and Expert Systems* – PHI, 2007.
e) Ivan Bratko, *Prolog Programming for Artificial Intelligence*, Addison-Wesley, Pearson Education, 3rd edition, 2000.

10. Contents of Syllabus:

A. Theory

UNIT 1: Introduction (4 Lectures)

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

UNIT 2: Problem Solving and Searching Techniques (16 Lectures)

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 (14 Lectures)

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

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

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

UNIT 5: Understanding Natural Languages (5 Lectures)

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

B. List of Practical

1. Write a prolog program to calculate the sum of two numbers.
2. Write a prolog program to find the maximum of two numbers.
3. Write a prolog program to calculate the factorial of a given number.
4. Write a prolog program to calculate the nth Fibonacci number.
5. Write a prolog program, insert_nth (item, n, into_list, result) that asserts that result is the list into_list with item inserted as the nth element into every list at all levels.
6. Write a Prolog program to remove the nth item from a list.
7. Write a Prolog program, remove nth (Before, After) that asserts the After list is the Before list with the removal of every nth item from every list at all levels.
8. Write a Prolog program to implement append for two lists.
9. Write a Prolog program to implement palindrome (List).
10. Write a Prolog program to implement max(X,Y,Max) so that Max is thegreater of two numbers X and Y.
11. Write a Prolog program to implement maxlist(List,Max) so that Max is thegreatest number in the list of numbers List.
12. Write a Prolog program to implement sumlist(List,Sum) so that Sum is thesum of a given list of numbers List.
13. Write a Prolog program to implement two predicates evenlength(List) and oddlength (List) so that they are true if their argument is a list of even or oddlength respectively.
14. Write a Prolog program to implement reverse (List, Reversed List) thatreverses lists.
15. Write a Prolog program to implement maxlist (List, Max) so that Max is the greatest number in the list of numbers List using cut predicate.
16. Write a Prolog program to implement GCD of two numbers.
17. Write a prolog program that implements Semantic Networks/Frame Structures.

CIT0600404: DATA MINING AND WAREHOUSING

1. Learning Outcome:

a) Understanding the process of Knowledge Discovery in Databases
b) Understand the functionality of the various data warehousing component.
c) Characterize the kinds of patterns that can be discovered by association rule mining.
d) Analysis of different types of data by clustering and classification.

2. Prerequisites: NIL

3. Semester: 6

4. Course Type: Compulsory

5. Course Level: 300-399

6. Theory Credit: 3

7. Practical Credit: 1

8. No. of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

9. List of Books:

a) A.K. Puzari, *Data Mining Techniques*, University Press.
b) J. Han, J. Pie and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann.
c) P. Tan, M. Steinbach and V. Kumar, *Introduction to Data Mining*, Pearson Education (LPE).
d) G. K. Gupta, *Introduction to Data Mining with Case Studies*, PHI.

10. Contents of Syllabus:

A. Theory

UNIT 1: Overview **(4 Lectures)**

What is Data Mining?, Knowledge Discovery in Databases (KDD) vs. Data Mining, Types of Data, Basic Data Mining Tasks, Predictive and Descriptive data mining techniques, Supervised and Unsupervised learning techniques, Basics of Pre-processing methods- Data Cleaning, Data Integration and Transformation, Data Reduction, Data Visualization.

UNIT 2: Data Warehousing **(6 Lectures)**

What is Data Warehouse? Multidimensional Data Model, Data Cube, Basic Components of Multidimensional Data Model, OLAP Operations- Slicing, Dicing, Drilling, Drill-Up, Drill-Down, Drill-Within, Drill-Across, Pivot(Rotate), Schema of Warehouse, Data Warehouse Architecture, Metadata.

UNIT 3: Association Rule Mining **(12 Lectures)**

What is Market Basket Data?, k-Itemset, Support of an Itemset, Frequent Itemsets, Infrequent Itemsets, Maximal Frequent Itemsets, Closed Frequent Itemsets, Association Rules, Confidence of a Rule, Problem of Mining Association Rules, Algorithm for Mining Frequent Itemsets- Apriori Algorithm, Pincer-Search Algorithm, DIC (Dynamic Itemset Counting) Algorithm, Steps of Mining Association Rules.

UNIT 4: Clustering **(12 Lectures)**

What is Clustering, Partitional vs Hierarchical Clustering, Types of Data in Clustering, Distance Measures used in Clustering- Euclidean Distance, Manhattan Distance, Similarity Measures used in Clustering- Cosine Similarity, Jacquard Coefficient, Partitional Clustering Methods- K-Means, K-Mediods, PAM, CLARA, CLARANS, Density Based Clustering Methods- DBSCAN, Introduction to Hierarchical Clustering.

UNIT 5: Classification (8 Lectures)

What is Classification? Issues Regarding Classification, K-Nearest Neighbor Classifiers, Bayesian classification, Introduction to Decision Tree.

UNIT 6: Recent Trends and Techniques used in Data mining (3 Lectures)

Basic Concepts of- Web Mining, Spatial Data Mining, Temporal Data Mining, Big Data Mining, Concept of Neural Network, Genetic Algorithm.

B. List of Practical

1. Implement ***any one*** from the following-
    a. Write a computer program to implement A priori algorithm to mine all frequent itemsets from a transactional dataset. Use hashing to store the item sets in the level wise generation of candidate sets.
    b. Write a computer program to implement the Pincer Search algorithm.
    c. Write a computer program to implement the DIC (Dynamic Item set) algorithm.
2. Implement ***any four*** from the following-
    a. Write computer program to implement the K-Means algorithm using different distance measures stated in the syllabus.
    b. Write computer program to implement the PAM algorithm using different similarity measures stated in the syllabus.
    c. Write a computer program to implement the CLARA algorithm.
    d. Write a computer program to implement the CLARANS algorithm.
    e. Write a computer program to implement the DBSCAN algorithm.
    f. Write a computer program to implement the K-NN algorithm.

CIT0600504: GRAPH THEORY

1. Learning Outcome:

a) 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: Compulsory

5. Course Level: 300-399

6. Theory Credit: 4

7. Practical Credit: 0

8. No of Hours:

a) Theory: 60 hrs

b) Practical: 0 hrs

c) Non Contact: 5 hrs

9. List of Books:

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

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

c) Narasingh Deo, *Graph Theory with Applications to Engineering and Computer Science*, PH

10. Contents of Syllabus:

A. Theory

UNIT 1: Introduction (5 Lectures)

Graph, directed and undirected graph, weighted and un weighted 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 2: Connectivity, paths and cycle (15 Lectures)

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 3: Tree (12 Lectures)

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 4: Matching and coloring (13 Lectures)

Matching, bipartite matching, maximum bipartite matching, Ford Fulkerson's algorithm for finding maximal bipartite matching, perfect bipartite matching, non-bipartite matching, maximal non-bipartite 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 5: Digraph (7 Lectures)

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 6: Classical problems **(8 Lectures)**

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

CIT0600604: PROJECT

1. Learning Outcome:

a) Students will recall and describe the problem statement, objectives, and methodologies employed in their project.

b) Students will demonstrate an understanding of the technologies explored during the project, explaining their relevance, functionality, and potential applications.

c) Students will apply the acquired knowledge and skills to develop a solution or prototype addressing the identified problem, utilizing the chosen technologies effectively.

d) Students will critically analyze the project outcomes, assessing the strengths and weaknesses of their approach, and identifying areas for improvement or further exploration.

e) Students will synthesize their findings into a coherent dissertation, presenting their research methodology, results, and conclusions while evaluating the implications and significance of their work within the broader context of the field.

2. Prerequisite: Basic Subject knowledge

3. Semester: 6

4. Course Type: Compulsory

5. Course Level: 300-399

6. Theory Credit: 0

7. Practical Credit: 4

8. Number of required hours:

a) Theory: 0 hrs
b) Practical: 60 hrs
c) Non Contact: 0hrs

9. Course Content:

At the onset of their sixth semester, each student will receive an assignment for a project. Students, either individually or in pairs, will delve into a unique problem under the mentorship of a faculty member from the department. The chosen problem should allow students to delve deeply into one or two specific technologies, fostering a strong understanding and proficiency in those areas upon project completion.

To promote innovation and avoid redundancy, previously tackled problems should be avoided unless they hold exceptional research significance and expansive scope. While application-based problems spurred by specific demands may be considered, simplistic information management systems comprising only a few database tables or data entry forms should be discouraged.

Interdisciplinary collaboration where applicable, enabling students to draw insights from diverse fields and perspectives to enrich their projects will be encouraged. Students also have the option to conduct their projects in collaboration with other institutes or organizations, subject to approval from the relevant institute organization. However, at least one project supervisor must be affiliated with the institute or organization.

Regular progress updates must be reported by meetings with the project supervisor throughout the project duration.

Students should look for opportunities to publish their project findings in academic journals, conferences, or other relevant platforms to disseminate their research outcomes and contribute to the academic community.

Projects must culminate in the submission of a dissertation. Evaluation and presentation of projects will adhere to the regulations outlined in the PG course semester system of G.U., with choice-based credit and grading system.

10. Course Assessment Details

Internal assessment: seminars, presentations, viva, project implementation

SEC 1: COMPUTER FUNDAMENTALS

1. Learning Outcomes: After completing this course, students will know about fundamentals of Computer System and Software.

2. Prerequisites: NIL

3. Semester: 1

4. Course type: Compulsory

5. Course level: 100-199

6. Theory credit: 3

7. Practical credit: 1

8. Number of required hours:

a) Theory: 45 hrs (45 classes)

b) Practical: 30 hrs (15 classes)

c) Non Contact: NIL

9. List of Reference Books:

(a) E Balagurusamy, *Fundamentals of Computers,* Mc Graw Hill Education.

(b) V.Rajaraman, Neeharika Adabala, *Fundamentals of Computers,* PHI Learning.

(c) Anita Goel, *Computer Fundamentals,* Pearson Education.

10. Contents of Syllabus:

A. Theory

UNIT 1: Introduction to Computers and Number Systems (7 Lectures)

Number system, decimal, binary, octal and hexadecimal number system, conversion among number systems, definition of computer, basic components of computer, bus, evolution of computers, Generations of computers, classification of computers, data representation in a computer, ASCII, Unicode

UNIT 2: Memory and Storage Devices (8 Lectures)

Memory, memory hierarchy, registers, general purpose and special purpose registers, primary and secondary memory, volatile and non volatile memory, semiconductor memory, SRAM and DRAM, Read Only Memory, magnetic storage devices, optical storage devices, solid state devices, flash memory, storage evaluation criteria

UNIT 3: Input Devices (7 Lectures)

Input device, keyboard, keyboard layouts, pointing devices, mechanical and optical mouse, scanner, hand-held and flat-bed scanners, OMR, OCR, MICR, digital camera, touch pad, track ball, joystick, digitizer, digital microphone.

UNIT 4: Output Devices (7 Lectures)

Monitor, LCD, LED, plasma monitor, printers, impact printers, non-impact printers, dot

matrix printers, inkjet printers, laser printers, thermal printers, plotters, voice output systems, projector,

UNIT 5: Programming Languages and Software (11 Lectures)

CPU, control unit, computer instruction, instruction set, instruction execution life cycle, program,
programminglanguages,machinelevellanguage,assemblylanguage,lowlevellanguage,high level language, language translators, assembler, compiler, interpreter, algorithm, definition of pseudocode, flowchart, flowchart of algorithm to find maximum of n numbers, software, flowchart of algorithm to find minimum of n numbers, flowchart of algorithm to find average of n numbers, software, flow chart of algorithm to display first n terms of Fibonacci series, flow chart of algorithm to check whether a given number is prime, software, software, application software, examples of application software, system software, examples of system software, what is operating system, what is device driver, open source software, proprietory vs open source software, examples of proprietory and open source software.

UNIT 6: Computer Network and Internet (5 Lectures)

Computer network, network topologies, LAN and WAN, internet, ISP, services over internet, www, webserver, webbrowser, HTML, HTMLtags: <html>, <head>, <title>, <body>, <h1>, <h2>,<h3>, <h4>, <h5>, <h6>,
, <p>, <a>, <ul>, <ol>, <li>, , <table>, <th>, <tr>, <td>, introduction to CSS, domain name, URL, DNS, E-mail, telnet, FTP.

B.List of Practical

1. Using a word processing software such as Libre Office Writer.
2. Using a spread sheet software such as Libre Office Calc.
3. Using a presentation software such as Libre Office Impress.
4. Using an image editing software such as GIMP.
5. Using an audio editing software such as Audacity.
6. Using a video editing software such as Openshot.
7. Designing HTML webpages.

SEC 2: WEB TECHNOLOGIES

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

a) Understand the basic concept of web applications and web services.
b) Design basic well-structured web page using HTML and CSS
c) Develop the ability to implement interactive elements and dynamic content using basic JavaScript
d) Develop a foundational understanding of server-side scripting using PHP

2. Prerequisites: NIL

3. Semester: 2

4. Course Type: Compulsory

5. Course Level: 100-199

6. Theory Credit: 3

7. Practical Credit: 1

8. No of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

9. List of Books:

a) Jackson J.C. (2007). *Web Technologies: A Computer Science Perspective*. Pearson.
b) Duckett, J. (2011). *HTML and CSS: Design and Build Websites*. John Wiley & Sons.
c) Robbins, J. N. (2018). *A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics*. O'Reilly Media.
d) Robbins, J. N. (2018). *Learning Web Design: A Beginner's Guide.* O'Reilly Media.
e) Haverbeke, M. (2018). *Eloquent JavaScript*. No Starch Press.
f) Welling, L., & Thomson, L. (2016). *PHP and MySQL Web Development* (5th ed.). Addison-Wesley Professional.

10. Contents of Syllabus:

A. Theory

UNIT 1: Introduction to Web Technologies (8 Lectures)

Concepts of the Internet and the World Wide Web (WWW), Overview of web browsers and their functionalities. Client-Server Architecture in Web Applications. Communication Protocols – HTTP, HTTPS, FTP. Working of DNS. Brief concepts of port, URL, cache and cookies. Web Content Accessibility Guidelines. Privacy concerns and data protection regulations, GDPR. Introduction to Web Hosting and control panels.

UNIT 2: Front End Development using HTML (10 Lectures)

Website and Webpage. Basic concept of Markup Language. Introduction to HTML. Basic HTML structure. Text formatting Tags – headings, paragraph, line break, horizontal rule. Link and Navigation – anchor tags. Lists - ordered, unordered, definition list. Image and multimedia tags. Tables in HTML. Forms and Input types – text, email, password, radio, select, checkbox, textarea, date, url, submit, button. Semantic HTML. Sectioning elements – header, nav, main, section, article, aside, footer.

UNIT 3: Front End Design using CSS (9 Lectures)

Introduction to CSS. CSS syntax and rule structure. Inline, Internal and External CSS. CSS selectors – element, class, ID, attribute. Combinators – descendant, child, adjacent sibling, general sibling. Understanding the CSS Box Model – content, padding, border, margin. CSS colours and backgrounds

– background-color, background-image, background-repeat. CSS typography – font properties, text properties.

UNIT 4: Client-Side Scripting with JavaScript (10 Lectures)

JavaScript as a high-level interpreted language. JavaScript code execution in web browsers – JavaScript execution context. JavaScript syntax and datatypes. JavaScript variables – var, let, const. Assignment and scope of JavaScript variables. Operators in JavaScript – arithmetic, comparison, logical, assignment. Conditional Statements. Looping Structures. Function declaration and Invocation in JavaScript. Introduction to the Document Object Model. Accessing HTML elements in DOM – by id, by tag name, by class name, query selectors. Manipulating DOM elements – create, add, append, remove. InnerText vs InnerHTML. Manipulating CSS styles using DOM. Event handling and delegation with the DOM using JavaScript. Client-side form validation using JavaScript. Handling form validation and processing data.

UNIT 5: Server-Side Programming with PHP (8 Lectures)

Introduction to PHP and role in Web development. PHP syntax and variables. Basic PHP functions – Built-in PHP functions, string manipulation functions, mathematical functions, date and time functions. PHP forms and form handling. Form submission methods – GET and POST. Handling form data with PHP. Uploading files with PHP. Introduction to the tech-stack. Role of Apache, PHP, MySQL etc. Introduction to Databases and SQL. Connecting to databases with PHP. Executing SQL queries with PHP. Retrieving, inserting, updating and deleting data from databases using PHP.

B. List of Practical

(This is a suggestive list only. Questions need not be restricted to this list.)

1. Create a basic HTML webpage structure with a heading, paragraph, and an image.
2. Build a navigation menu using an unordered list (<ul>) with clickable links.
3. Implement a form with input fields for name, email, and a submit button.
4. Create a table with multiple rows and columns to display tabular data.
5. Design an image gallery using HTML and CSS with proper padding and border.
6. Embed a YouTube video on a webpage using the <iframe> tag.
7. Implement an ordered list (<ol>) to display a step-by-step tutorial or instructions.
8. Create a dropdown select menu (<select>) with multiple options.
9. Use HTML5 semantic tags (such as <header>, <nav>, <section>, <article>, <footer>) to structure and organize content on a webpage.
10. Build a registration form with fields for name, email, password, date of birth, address and other such fields with a submit button. Include appropriate input types, labels and placeholders.
11. Style a heading element with a custom font, colour and background.
12. Apply different background colors to alternate rows in a table.
13. Implement a hover effect on a button that changes its background colour or adds a solid border.
14. Style a form input field with custom border, padding, and background color.
15. Implement a CSS tooltip that displays additional information when hovering over an element.
16. Build a simple JavaScript calculator that can perform basic arithmetic operations.
17. Create a button that, when clicked, appends a new paragraph element with a specific text content to an existing div element.
18. Implement a function that changes the innerText of a paragraph element to display a random number between 1 and 10 every time a button is clicked.

19. Build a form with input fields for name and email. When the form is submitted, use innerHTML to display a confirmation message with the entered name and email on the webpage.
20. Build a form with input fields for email, password and confirm password. When the form is submitted, use an alert to display a success message if the password and confirm password values matches, otherwise show an error alert. Use JavaScript for the validation.
21. Create a list of items. Add a click event listener to each item so that when clicked, the background color of the clicked item changes.
22. Write a PHP script to display the current date and time on a webpage.
23. Write a PHP script to connect to a MySQL database and fetch data from a table.
24. Create a registration form with fields for username, email, and password. Implement server-side validation to check for duplicate usernames or invalid email formats. Store the user registration data in a MySQL database. Provide feedback to the user upon successful registration or display appropriate error messages.
25. Design a webpage that displays a list of notices retrieved from a MySQL database. Implement functionality to add new notices to the database using a form. Allow users to view and delete individual notices. Apply appropriate styling to the notices and ensure proper validation and sanitization of user input.

CIT070104: RESEARCH METHODOLOGY

1. **Learning Outcome:**
   a) To introduce the basic concepts in research methodology in Computer Science.
   b) To familiarize the issues inherent in selecting a research problem
   c) To discuss the techniques and tools to be employed in completing a research project.
   d) Enable the students to prepare report writing and framing Research proposals.
2. **Prerequisites:** NIL
3. **Semester:** 7
4. **Course Type:** Compulsory
5. **Course Level:** 400-499
6. **Theory Credit:** 3
7. **Practical Credit:** 1
8. **No of Hours:**
   a) Theory: 60 hrs
   b) Practical: 0 hrs
   c) Non Contact: 5 hrs
9. **List of Books**
   a) C. R. Kothari, *Research Methodology-Methods and Techniques*, New-age International Publishers
   b) K. Prathapan, *Research Methodology for Scientific Research*, Wiley Publication
10. **Contents of the Syllabus**

A. Theory

UNIT 1: Introduction to Computer Science Research (4 Lectures)

What is Research? Types of Research, Why Research, Significance & Status of Research in Computer Science.

UNIT 2: Steps in Research (6 Lectures)

Major Journals & Publications in Computer Science, Major Research Areas of Computer Science, Identification, selection & formulation of research problems. Developing a research proposal, planning your research. How engineering and technological research differs from other scientific research.

UNIT 3: Research Data (6 Lectures)

Data Collection: Methods of Data Collection, Theory of Sampling, Sampling techniques. Size of a sample.

UNIT 4: Data Analysis and interpretation (12 Lectures)

Statistical analysis of data: Measures of central tendency, dispersion; Associations/Relations; Regression and Co-relation analysis; Hypothesis testing and tests of significance. Data processing software and statistical inference, Interpretation of results.Use of R-Programming in Statistical Analysis and Data Visualization.

UNIT 5: Simulation and tools **(6 Lectures)**

Concept of Simulation. Time and randomness in simulation, Application of simulations. How a simulation model works, tools for simulations.

UNIT 6: Literature Survey **(6 Lectures)**

Finding out about your research area, Literature search strategy. Writing critical reviews. Identifying venues for publishing your research.

UNIT 7: Plagiarism **(6 Lectures)**

Concept and Importance of understanding what is plagiarism and what is not plagiarism. Methods and ways to detect and avoid plagiarism. Available tools and software for plagiarism check.

UNIT 8: Writing Papers and the Review Process **(10 Lectures)**

Preparing and presenting your paper. The conference review process. Making use of the referees' reports. The journal review process, Group exercise in reviewing research papers. Tools for paper formatting and Reference Management. Use of LaTeX.

UNIT 9: Ethical Issues and Professional Conducts **(4 Lectures)**

Ethics in general, Professional Ethics, Ethical Issues that Arise from Computer Technology, General Moral Imperatives, More Specific Professional Responsibilities, Organizational Leadership Imperatives.

B. List of Practical

Practical sessions will be held for the following

1. R-Programming in Statistical Analysis and Data Visualization.
2. Explore available tools and software for plagiarism check.
3. Use of LaTeX in writing documents with equations, figures etc. and report generation.

CIT070204: FUNDAMENTALS OF MACHINE LEARNING

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

(a) Understand what is machine learning and the major machine learning approaches
(b) Differentiate between supervised and unsupervised learning
(c) Understand linear and non- linear classification.
(d) Understand the role and importance of feature extraction.
(e) Apply machine learning algorithms to real world problems.

2. Prerequisites: Fundamentals of Python programming

3. Semester: 7

4. Course type: Elective

5. Course level: 400-499

6. Theory credit: 3

7. Practical credit: 1

8. Number of required hours:

a) Theory: 45 hrs (45 classes)

b) Practical: 30 hrs (15 classes)

c) Non- Contact: 10 hrs

9. Reference books:

1. J. Shavlik and T. Dietterich (Ed), *Readings in Machine Learning*, Morgan Kaufmann, 1990.
2. Chris Albon, *Machine Learning with Python Cookbook*, Pub: Shroff/O'Reilly, 2018.
3. Shai Shalev-Shwartz and Shai Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. E-book: https:// kdnuggets.com
4. NPTEL

10. Contents of the Syllabus

A. Theory

UNIT 1: Introduction to Machine Learning (12 Lectures)

Definition and scope of machine learning, History and Goals of ML. Types of machine learning: supervised, unsupervised, reinforcement learning, Applications of machine learning, Data Preprocessing, Dealing with heterogeneous data. Introduction to Python libraries for machine learning (NumPy, pandas, scikit-learn).

UNIT 2: Linear Regression (05 Lectures)

Simple linear regression, Multiple linear regression, Gradient descent, Regularization techniques: L1 and L2 regularization

UNIT 3: Supervised Learning and Classification (07 Lectures)

Logistic regression, Decision trees – Entropy, Splitting attribute and Splitting criteria. Support Vector Machines (SVM), k-nearest Neighbors (k-NN) algorithm.

UNIT 4: Model Evaluation and Selection **(04 Lectures)**

Training and testing datasets, Cross-validation, Evaluation metrics: accuracy, precision, recall, F1-score, ROC curve, AUC.Concept of Ensemble and Ensemble techniques- Bagging, Boosting, Stacking and Voting.

UNIT 5: Unsupervised Learning **(08 Lectures)**

Partitional Clustering - K-means and k-medoids algorithm, Hierarchical clustering –Agglomerative and Divisive, Density based clustering – DBScan algorithm.

UNIT 6: Dimensionality Reduction **(06 Lectures)**

Feature selection, Feature extraction, Principal Component Analysis (PCA).

UNIT 7: Applications in Different Fields **(03 Lectures)**

Working on real-world datasets to show applications of machine learning in Natural Language Processing (NLP), Image processing, Recommender Systems, Healthcare etc.

CIT0700304: ADVANCED OPERATING SYSTEM

1. **Learning Outcomes**: After successful completion of this course, students will be able to:
    a) Understand advanced operating system concepts and architectures.
    b) Explore virtualization technologies and their applications.
    c) Study real-time operating systems and their characteristics.
2. **Prerequisites:** Operating Systems
3. **Semester:** 7
4. **Course Type:** Elective
5. **Course Level:** 400-499
6. **Theory Credit:** 4
7. **Practical Credit:** 0
8. **No of required hours:**
    a) Theory: 60 hours
    b) Practical: 0 hours
    c) Non Contact: 5 hours
9. **List of Reference Books:**
    a) Andrew S. Tanenbaum, *Modern Operating Systems*, 4th edition, Pearson Education
    b) Abraham Silberschatz, Peter B. Galvin, Greg Gagne, *Operating System Concepts*, Wiley
    c) Tanenbaum, Steen, *Distributed Systems Principles and Paradigms*, 2nd edition, Pearson Education
    d) Jane W. S. Liu, *Real-Time Systems*, 1st edition, Pearson Education India

10. **Contents of Syllabus**:

A. Theory

UNIT 1: Review of Operating System Concepts (2 Lectures)

Process and threads: Process Control Block (PCB), process scheduling, Process synchronization: critical section problem, semaphore, process deadlock.

UNIT 2: Multiprocessor Systems (12 Lectures)

Introduction to multiprocessor systems, multiprocessor hardware: UMA (Uniform Memory Access), NUMA (Non Uniform Memory Access), Multicore Chips,Moore's law, Manycore chips. Multiprocessor Operating System Types: private OS, Master-Slave Multiprocessors, Symmetric Multiprocessors, Multiprocessor Synchronization: TSL (Test and Set Lock), Peterson's protocol, Multiprocessor scheduling: Space Sharing, Gang Scheduling.

UNIT 3: Multicomputer Systems (13 Lectures)

Introduction to multicomputer systems, multicomputer Hardware organization: interconnection Technology, Network Interfaces, Low-Level Communication Software, User-Level Communication Software: Send and Receive, Blocking versus Nonblocking Calls, Remote Procedure Call : client stub, server stub, marshalling, Load Balancing, processor allocation algorithms: Graph-Theoretic Deterministic Algorithm, Sender-Initiated Distributed Heuristic Algorithm, Receiver-Initiated Distributed Heuristic Algorithm.

UNIT 4: Distributed System (10 Lecture)

Definition of a distributed system. Characteristics of distributed systems, distributed systems vs centralized systems. Examples of Distributed Systems in Real-world Applications, System models: Fundamental and Architectural model, System architectures- client-server architectures, Synchronization: Needs of clock synchronization, external and internal clock synchronization, Logical and vector clocks, Lamport's logical clock, Vector clocks, Causal Order of messages. Global state, Chandy Lamport snapshot algorithm

UNIT 5: Virtualization and Containerization (10 Lectures)

Definition of virtualization, need of virtualization, examples of virtual machines, advantage and disadvantage of virtualization, hypervisor architecture : type 1 and type 2 hypervisor, guest OS, host OS, Introduction to Containerization Technologies (e.g., Docker, Kubernetes), Containers vs. Virtual Machines, benefits of containerization, use cases of containerization, container orchestration

UNIT 6: Real-time Operating Systems (10 Lectures)

Definition and Characteristics of Real-Time Systems(RTOS), Differences Between General-Purpose Operating Systems and RTOS, Importance of Timing Constraints in RTOS, Types of RTOS, Hard vs Soft RTOS, Classification of Real-Time Tasks: Periodic, Aperiodic, Sporadic tasks, Real-Time Scheduling Algorithms: Earliest Deadline First, Least Laxity First.

UNIT 7: Case study on real-time operating systems (3 Lectures)

FreeRTOS, FreeRTOS features, Applications of FreeRTOS, FreeRTOS architecture, FreeRTOS supported architectures, FreeRTOS libraries.

CIT0700404: ADVANCED COMPUTER ORGANIZATION AND ARCHITECTURE

1. Learning Outcomes:

a) Understand the importance of multiprocessor and multicomputer.
b) Learn about advance architecture designs.
c) Understand the concepts of interconnected structures.
d) Learn about data flow computer architectures.

2. Prerequisites: Fundamental knowledge of computer Organization and Architecture

3. Semester: 7

4. Course Type: Elective

5. Course Level: 400-499

6. Theory Credit: 4

7. Practical Credit: 0

8. No of Hours:

a) Theory: 60 hrs
b) Practical: 0 hrs
c) Non-Contact: 5 hrs

9. List of Books:

a) Govindarajalu, B. *Computer Architecture and Organization*, TMH publication.
b) Richard Y. Kain, *Advanced Computer Architecture: A systems Design Approach*, PHI Publication
c) Stallings William, *Computer Organization and Architecture Designing for Performance*, Pearson Education
d) Kai Hwang, *Advanced Computer architecture Parallelism, scalability, Programmability*, McGraw-Hill, N.Y, 2003
e) Kai Hwang and Faye Briggs, *Computer Architecture and Parallel Processing*, McGraw-Hill International Edition, 2000

10. Contents of Syllabus:

A. Theory

UNIT 1: Computer Arithmetic (10 Lectures)

Serial adder, Parallel adder, Ripple carry adder, Carry look-ahead adder, Multiplication of signed and unsigned numbers, Booth's algorithm, Division of integer, Floating point arithmetic.

UNIT 2: Advanced Architectures (15 Lectures)

Classification: SISD, SIMD, MISD, MIMD, Bus data path: one, two and three bus, Scalar, vector, superscalar and pipelined processor,VLIW architecture, EPIC architecture, Pipelining: Instruction pipeline, pipeline bubbles, Hazards: resource conflicts, data dependency, branch difficulty, Vector computing: arithmetic pipeline, vector and scalar register, chaining, scatter gather operations, vector-register processor, Memory vector processor, Array processor.

UNIT 3: Assembly Language (10 Lectures)

Machine Language, Register transfer language, Assembly Language, Assembler, Program Loops, Subroutines, Developing counters and Time delay routines, Input-Output Programming. Interfacing concepts.

UNIT 4: 8085 Microprocessor Architecture and Memory Interface (10 Lectures)

Pin Description, Operating Modes, Instruction Set of 8085: Instructionset, Data formats, Addressing modes, Opcode & Operands, Data and Storage, Word size, 8085 Instructions: Counter and Time delays, Stack, Subroutines, Call &Return statements, Interrupts in 8085: generation of RST codes, interrupt priority, SIM & RIM instructions.

UNIT 5: Multiprocessor (8 Lectures)

Characteristics of Multiprocessors, Interconnection Structures, Interprocessor Arbitration, Interprocessor Communication and Synchronization, Cache Coherence, Multicore Processors

UNIT 6: Dataflow Computers and VLSI Computations (7 Lectures)

Data flow architecture: static and dynamic, VLSI Computing Structures, Array Architectures, Mapping algorithms, Reconfigurable processor array, VLSI matrix arithmetic models and processors, Matrix Algorithms and Pipelines

CIT0700504: CRYPTOGRAPHY AND NETWORK SECURITY

1. **Learning Outcome:**
   a) Demonstrate a comprehensive understanding of fundamental concepts, principles, and technologies related to network security, including encryption techniques, authentication protocols, and security architectures.
   b) Apply their knowledge and skills to analyze, design, and implement effective security measures to protect networked systems and data from unauthorized access, malicious attacks, and other security threats.
2. **Prerequisite:** NIL
3. **Semester:** 7
4. **Course Type:** Elective
5. **Course Level:** 400-499
6. **Theory Credit:** 4
7. **Practical Credit:** 0
8. **Number of required hours:**
   a) Theory: 60 hrs
   b) Practical: 0 hrs
   c) Non Contact: 5 hrs
9. **List of reference books:**
   a) W. Stallings, *Cryptography & Network Security Principles & Practices* (7th Edition), Pearson Education, 2017.68
   b) Wade Trappe, Lawrence C Washington, *Introduction to Cryptography with coding theory*, Pearson.
   c) D. R. Stinson, *Cryptography: Theory & Practice (3rdEdition)*, CRC Press, 2006.
10. **Detailed Syllabus:**

A. Theory

UNIT 1: Introduction (8 Lectures)

OSI Security Architecture -Security Attacks-Security Services-A Model for Network Security-Classical Encryption techniques – Cipher Principles – Symmetric Cipher Model; Substitution Techniques; Transposition Techniques; Steganography

UNIT 2: Modern Block Ciphers (7 Lectures)

Data Encryption Standard – Block Cipher Design Principles and Modes of Operation - Evaluation criteria for AES – AES Cipher – Triple DES – Placement of Encryption Function – Traffic Confidentiality

UNIT 3: Public Key Cryptography and Hash Functions (15 Lectures)

Public Key Cryptosystems, Applications, Requirements, Cryptanalysis, RSA Algorithm, Key Management - Diffie-Hellman key Exchange – Cryptographic Hash Functions: Applications of Cryptographic Hash Functions, Simple Hash Functions, Authentication requirements – Authentication functions – Message Authentication Codes-Message Authentication Requirements, Authentication Functions, Requirements of Message authentication codes, Security of MACs.

UNIT 4: Network Security (15 Lectures)

Digital Signatures; Key Management and Distribution; Authentication Applications: Kerberos – X.509 Authentication Service – Electronic Mail Security – PGP – S/MIME - IP Security – Web Security; Intrusion detection – password management – Viruses and related Threats – Virus

Counter measures – Firewall Design Principles – Trusted Systems.

CIT0700604: ADVANCED DATABASE MANAGEMENT SYSTEMS

1. **Learning Outcome:**
   a) Students will be able to recall key concepts and terminologies related to SQL, object databases, XML, distributed databases, and NoSQL systems.
   b) Students will analyze the effectiveness and efficiency of distributed database architectures, query processing, and transaction management, as well as the advantages and limitations of different NoSQL systems.
   c) Students will evaluate advanced database models and systems, such as active, temporal, spatial, and multimedia databases, and assess data mining techniques for extracting meaningful insights from large datasets.
2. **Prerequisite:** Database Management Systems
3. **Semester:** 7
4. **Course Type:** Elective
5. **Course Level:** 400-499
6. **Theory Credit**: 4
7. **Practical Credit**: 0
8. **Number of required hours**:
   (a) Theory: 60 hrs
   (b) Practical: 0 hrs
   (c) Non Contact: 5 hrs
9. **List of reference books:**
   (a) Elmasri Ramez, Navathe Shamkant, *Fundamentals of Database System*, 7edition, Pearson
   (b) R. Ramakrishnan, J. Gehrke, *Database Management Systems*, McGraw Hill, 2004

10. Detailed Syllabus:

A. Theory

UNIT 1: SQL Revisited **(9 Lectures)**

SQL Data Definition and Data Types; Specifying Constraints in SQL; Basic Retrieval Queries in SQL; INSERT, DELETE, and UPDATE Statements in SQL; Additional Features of SQL; More Complex SQL Retrieval Queries; Specifying Constraints as Assertions and Actions as Triggers; Views (Virtual Tables) in SQL; Schema Change Statements in SQL

UNIT 2: Object Database Concepts and XML **(12 Lectures)**

Overview of Object Database Concepts- Introduction, Object Identity, and Objects versus Literals, Object Identity, and Objects versus Literals, Encapsulation, Type Hierarchies and Inheritance ; Object Database Extensions to SQL; The ODMG Object Model and the Object Definition Language ODL; Object Database Conceptual Design; The Object Query Language OQL; Overview of the C++ Language Binding in the ODMG Standard; Structured, Semi-structured, and Unstructured Data; XML Hierarchical (Tree) Data Model; XML Documents, DTD, and XML Schema; Storing and Extracting XML Documents from Databases; XML Languages; Extracting XML Documents from Relational Databases; XML/SQL: SQL Functions for Creating XML Data.

UNIT 3: Distributed Databases **(6 Lectures)**

Distributed Database Concepts; Data Fragmentation, Replication, and Allocation Techniques for Distributed Database Design; Overview of Concurrency Control and Recovery in Distributed Databases; Overview of Transaction Management in Distributed Databases; Query Processing and Optimization in Distributed Databases; Types of Distributed Database Systems; Distributed Database Architectures; Distributed Catalog Management;

UNIT 4: NOSQL Databases **(6 Lectures)**

Introduction to NOSQL Systems; The CAP Theorem; Document-Based NOSQL Systems and MongoDB; NOSQL Key-Value Stores; Column-Based or Wide Column NOSQL Systems; NOSQL Graph Databases and Neo4j

UNIT 5: Advanced Database Models and Data Mining (12 Lectures)

Active Database Concepts and Triggers; Temporal Database Concepts; Spatial Database Concepts; Multimedia Database Concepts; Introduction to Deductive Databases; Active Database Concepts and Triggers; Temporal Database Concepts; Spatial Database Concepts; Multimedia Database Concepts; Introduction to Deductive Databases; Trends in Information Retrieval; Overview of Data Mining Technology; Association Rules -Market-Basket Model, Support, and Confidence, Apriori Algorithm, ; Classification; Clustering; Applications of Data Mining;

COM0700704: COMPILER DESIGN

1. Learning Outcome:

a) Use compiler construction tools and describes the Functionality of each stage of compilation process
b) Construct Grammars for Natural Languages and find the Syntactical Errors/Semantic errors during the compilations using parsing techniques
c) Analyze different representations of intermediate code.
d) Construct new compiler for new languages.
e) Participate in GATE, PGECET and other competitive examinations

2. Prerequisites: NIL

3. Semester: 7

4. Course Type: Elective

5. Course Level: 400-499

6. Theory Credit: 4

7. Practical Credit: 0

8. No of Hours:

a) Theory: 60 hrs
b) Practical: 0 hrs
c) Non Contact: 5 hrs

9. List of Books:

a) Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman, *Compilers:Principles, Techniques and Tools*, 2nd edition, Pearson Education, New Delhi, India.
b) Alfred V. Aho, Jeffrey D. Ullman, *Principles of compiler design*, Indian student edition, Pearson Education, New Delhi, India.
c) Kenneth C. Louden ,*Compiler Construction– Principles and Practice*, 1st edition, PWS Publishing.
d) K. L. P Mishra, N. Chandrashekaran, *Theory of computer science- Automata Languages and computation*, 2nd edition, Prentice Hall of India, New Delhi, India.
e) Andrew W. Appel (2004), *Modern Compiler Implementation C*, Cambridge University Press, UK.
f) John R. Levine, Tony Mason, Doug Brown, *Lex &Yacc*, O'reilly

11. Contents of Syllabus:

A. Theory

UNIT 1: Introduction to Compiler (12 Lectures)

Definition of compiler, Phases of a compiler, Lexical analysis, Role of lexical analyzer, Tokens, Patterns, Lexemes, Input buffering, Specification of tokens-strings and languages, operations on languages, regular expressions, regular definitions, Recognition of tokens, Lexical analyzer generator-Lex, Finite automata, From Regular expressions to automata.

UNIT 2: Syntax Analysis (16 Lectures)

Parsing, Role of parser, Context free grammar, Parse tree and derivations, Ambiguity, Eliminating ambiguity from dangling-else grammar, Elimination of left recursion, Left factoring,
Top down Parsing- Recursive descent parser, Predictive parser- LL (1) Grammar, construction of predictive parsing table. Bottom Up Parsing- Reductions, Handle pruning, Shift-Reduce parsing, Conflicts during shift-reduce parsing, LR Parser-Items, Kernel items, Non-kernel items, closure of Item Sets, The function GOTO, LR (0) automaton, Construction of SLR parsing table, Basics of LALR parser, Automatic parser generator-YACC.

UNIT 3: Syntax Directed Translation (12 Lectures)

Syntax directed definition- inherited and synthesized attributes, evaluating an SDD at the nodes of a parse tree, Evaluation orders of SDD's- dependency graphs, ordering the evaluation of attributes, S-attributed and L-attributed definitions, Applications of syntax-directed translation- construction of syntax trees, the structure of a Type, Syntax directed translation schemes- postfix translation schemes, SDT's with actions inside productions, eliminating left recursion from SDT's, Variants of syntax trees- directed acyclic graphs (DAG) for expressions, The value-number method for constructing DAG's, Three address code- Quadruples, Triples and Indirect triples, Static single-assignment form, Types and Declarations, Translation of expressions, Type Checking, Basics of Control flow, Basics of Back patching.

UNIT 4: Run Time Environments (10 Lectures)

Storage organization, Stack allocation of space, Access to non-local data on Stack, Basics of Heap management, Basics of garbage collection

UNIT 5: Code Generation and Optimization (10 Lectures)

Machine dependent code generation, Issues in design of code generator, The target language, Addresses in the target code, Basic blocks and flow graphs, Optimization of basic blocks- the DAG representation of Basic blocks, Finding local common sub-expression, dead code elimination, A simple code generator, Basics of Peephole optimization, The Principal Sources of Optimization, Introduction to Data-Flow Analysis.

CIT0800104: ADVANCED DATA STRUCTURE

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: Fundamentals of C/C++programming

3. Semester: 7

4. Course type: Elective

5. Course level: 400-499

6. Theory credit: 3

7. Practical credit: 1

8. Number of required hours:

a) Theory: 45 hrs (45 classes)

b) Practical: 30 hrs (15 classes)

c) Non Contact: 10

9. Reference books:

a) Cormen T.H., Leiserson C.E., RivestR.L.; *Introduction to Algorithms*; Tata-McGraw Hill Publishers

b) Aho A., HopcroftJ.E.,UllmanJ.D.;*Data Structures and Algorithms*;Addison-Wesley

c) Horowitz, Sahani; *Fundamentals of Data StructuresinC/C++*;ComputerSciencePress

d) AhoA.,HopcroftJ.E.,UllmanJ.D.;*Design and Analysis of Computer Algorithms*; Addison-Wesley.

e) NPTEL

10. Detailed Syllabus:

A. Theory

UNIT 1: Review of Basic Concepts in Data Structure (7 Lectures)

A quick review of array versus linked list structure, binary tree, binary search tree, traversal, insertion and deletion in binary search trees.

UNIT 2: Dictionary ADT (9 Lectures)

Search trees, balancing of search trees – AVL trees, Red-Black trees, multi way search trees, 2-3 trees, splay trees, Insertion and Deletion in each of the above data structures, Hashing.

UNIT 3: Sorting and Selection Techniques (9 Lectures)

Quick sort, Heap sort, Shell sort, sorting in linear time – Counting sort, Radix sort. Medians and order Statistics. Selection and Adversary arguments. Lower bound on sorting.

UNIT 4: Priority Queue ADT (8 Lectures)

Heaps-extended priority queue, min(max) heaps, binomial heap, fibonacci heap and its amortized analysis.

UNIT 5: Partition ADT (5 Lectures)

Union-find algorithms through weighted merge and path compression.

UNIT 6: Data Structure for external storage operations (7 Lectures)
B-tree, insertion and deletion in B-trees, external sorting, B+ tree.

B. List of Practical

(This is a suggestive list only. Questions need not be restricted to this list. The practical are advised to be performed in Linux environment using C programming language.)

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 n^{th} element from the array.
2. Write a program to implement array initialized with the numbers divisible by three up to 30. Write a function which accepts the array and return the positions of the even numbers in the array.
3. 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 k^{th} position in the linked list
    e. Delete a node from first, last and k^{th} position
    f. Search for an element in the list. If found, return the position of the node. If not found, return a negative value.
4. Write a program to implement doubly linked list.
5. Write a function to concatenate two linked lists.
6. Write a program to take a number k and split the linked list after k^{th} position.
7. Write a program to merge two sorted linked lists.
8. Write a program to implement list of lists.
9. 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.
10. 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.
11. Write a program to evaluate a simple postfix expression using stack.
12. Write a program to convert a decimal number into binary number using stack.
13. 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.
14. 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.
15. Implement binary search and linear search algorithms on arrays.
16. Implement binary search tree using array by writing a program to:
    a. Create a binary search tree using array
    b. Print the prefix notation of the BST
    c. Print the infix notation of the BST
    d. Print the postfix notation of the BST
    e. Search for an element in the BST
17. Implement binary search tree using linked list by writing a program to:
    a. Create a binary search tree using linked list
    b. Print the prefix notation of the BST
    c. Print the infix notation of the BST
    d. Print the postfix notation of the BST
    e. Search for an element in the BST
    f. Display inorder , preorder and postorder list
18. Implement following sorting algorithms:
    a. Heap sort

    b. Quick sort
    c. Radix sort
    d. Counting sort
19. Create AVL tree, also delete node from AVL tree
20. Create Red-BlackAVL tree, also delete node from Red-Black tree

CIT0800204: EMBEDDED SYSTEM

1. Learning Outcome: After completion of this course, students will be able to

a) Understand the basic concepts of embedded system
b) Understand how an entire embedded system is designed
c) Understand how an embedded system is assessed

2. Prerequisite: NIL

3. Semester: 7

4. Course Type: Elective

5. Course Level: 400-499

6. Theory Credit: 4

7. Practical Credit: 0

8. Number of required hours:

a) Theory: 60 hrs
b) Practical: 0 hrs
c) Non Contact: 5 hrs

9. List of reference books:

a) Peter Marwedel, *Embedded System Design*, Springer
b) Elecia White, *Making Embedded Systems*, O′Reilly

10. Detailed Syllabus:

A. Theory

UNIT 1: Introduction (3 Lectures)

Definition of Embedded system, Need of embedded systems, challenges in embedded system design, common characteristics of embedded systems

UNIT 2: Specifications and Modeling (15 Lectures)

Requirements, Models of computation, StateCharts: Modeling of hierarchy, Timers, Edge labels and StateCharts semantics, Evaluation and extensions, General language characteristics: Synchronous and asynchronous languages, Process concepts, Synchronization and communication, Specifying timing, Using non-standard I/O devices, SDL, Petri nets: Introduction, Condition/event nets, Place/transition nets, Predicate/transition nets, Evaluation, Message Sequence Charts, UML, Process networks, Task graphs, Asynchronous message passing, Synchronous message passing, Java, VHDL, SystemC, Verilog and SystemVerilog, SpecC, Levels of hardware modeling

UNIT 3: Embedded System Hardware (12 Lectures)

Components of embedded system hardware, Input: Sensors, Sample-and-hold circuits, A/D-converters, Communication: Requirements, Electrical robustness, Guaranteeing real-time behavior, Examples, Processing Units: Overview, Application-Specific Circuits (ASICs), Processors, Reconfigurable Logic, Memories, Output, D/A-converters, Actuators

UNIT 4: Scheduling and Operating System (10 Lectures)

Prediction of execution times, Scheduling in real-time systems: Classification of scheduling algorithms, Aperiodic scheduling, Periodic scheduling, Resource access protocols, Embedded operating systems: General requirements, Real-time operating systems, Middleware: Real-time data bases, Access to remote objects

UNIT 5: Hardware/ Software Codesign (15 Lectures)

Task level concurrency management, High-level optimizations: Floating-point to fixed-point conversion, Simple loop transformations, Loop tiling/blocking, Loop splitting, Array folding, Hardware/software partitioning: Introduction, COOL, Compilers for embedded systems: Introduction, Energy-aware compilation, Compilation for digital signal processors, Compilation for multimedia processors, Compilation for VLIW processors, Compilation for network processors, Compiler generation, retargetable compilers and design space exploration, Voltage Scaling and Power Management: Dynamic Voltage Scaling, Dynamic power management (DPM), Actual design flows and tools, SpecC methodology, IMEC tool flow

UNIT 6: Validation **(5 Lectures)**

Introduction, Simulation, Rapid Prototyping and Emulation, Test: Scope, Design for testability, Self-test programs, Fault simulation, Fault injection, Risk- and dependability analysis, Formal Verification

CIT0800304: MOBILE APPLICATION DEVELOPMENT

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

a) Understand the basics of mobile application development
b) Adopt the developmental environment to develop robust applications.
c) Understand, write and debug applications.

2. Prerequisite: NIL

3. Semester: 8

4. Course Type: Elective

5. Course Level: 400-499

6. Theory Credit: 3

7. Practical Credit: 1

8. Number of required hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

9. List of reference books:

a) B. Phillips, C. Stewart, K. Marsicano, B. Gardner, *Android Programming: The Big Nerd Ranch Guide*, 5thEd., O'Reilly Media, 2022.
b) R. Meier, I. Lake, *Professional Android*, 4th Ed., Wiley, 2018
c) *Android Documentation* - https://developer.android.com/

10. Detailed Syllabus:

A. Theory

UNIT 1: Fundamentals of Mobile Applications (5 Lectures)

Introduction to mobile app development. Overview of different mobile platforms – Android, iOS. Platforms of mobile application development – Native vs Cross-platform. Understanding mobile app design and user experience principles. UI/UX design tools, concept of wire-framing and user research.

UNIT 2: Stepping into Mobile App Development (18 Lectures)

Basics of Java or Kotlin. Getting started with Android Studio IDE. Setting up development environments – editor, emulators and debugger. Exploring SDKs. Deploying and Debugging the basic "Hello World" app. Running on emulator, deployment as SaaS or PaaS.

Understanding Android *Activity* and its lifecycle, various events associated with Activity Lifecycle – *onCreate()*, *onStart()*, *onPause()*, *onResume()*, *onStop()*, *onRestart()*, *onDestroy()* etc. Understanding various essential folders and files associated with an Android App stored inside*manifests, java* and *res* directories. Basic understanding about *Gradle*.

Understanding the Mobile Application Development Lifecycle. Exploring app components – Activity, View, Fragments. Using *RecyclerView*, *ListView*, *ImageView* and *WebView*. Understanding and working with fragments. Using of intents – broadcasting and receiving intents. Passing data and objects through intents. Adapting display orientation. Managing notifications, action bar. Basics of action handling. Basics of Model View Controller (MVC) architecture. Building a simple app using MVC pattern.

UNIT 3: Working with Database and API (10 Lectures)

Introduction to database for mobile application. Using SQLite for local database. Interacting with Remote Databases using JSON. Using input methods with Input Method Editor (IME). Introduction to

Application Programming Interface (API). Using common APIs like Google Maps, YouTube, Open Weather in an application. Features of RESTful API. Integrating RESTful API for designing a CRUD application.
Setting up real-time databases like Firebase. Working with an application with Firebase as backend.

UNIT 4: Advanced Topics and Deployment (12 Lectures)
Introduction to mobile development frameworks – Flutter, React Native. Introduction to cross-platform mobile application development. Getting started with flutter. Creating and deploying a basic Flutter app in both Android and iOS using Android Studio and Xcode at Andoid and iOS respectively. Overview of social SDK. Setting permissions – install time, runtime permissions. Exploring and updating application manifest. Publishing into App Stores.

B. List of Practical

1. Develop a simple Android app using Java/Kotlin that displays "Hello World" on the screen.
2. Set up Android Studio IDE and create an emulator for testing your apps.
3. Create a basic Android app that utilizes different Activity lifecycle methods and displays log messages for each lifecycle event.
4. Implement a Recycler View in an Android app to display a list of items retrieved from a hardcoded array.
5. Create a multi-pane UI using Fragments for both smart phones and tablets, with navigation between fragments.
6. Develop an app that uses intents to navigate between activities and passes data (e.g., name, age) from one activity to another.
7. Build a CRUD (Create, Read, Update, Delete) app in Android that uses SQLite for local database storage.
8. Integrate Firebase into an Android app to store and retrieve data in real-time, such as a simple chat application.
9. Use JSON parsing to fetch data from a public API (e.g., weather data) and display it in your Android app.
10. Implement Google Maps API in an Android app to display a map with a marker at a specific location.
11. Create a basic Flutter app that displays text and images on both Android and iOS platforms.
12. Explore and implement runtime permissions in an Android app for accessing device features like camera or location.
13. Update the Android app manifest file to include necessary permissions, activities, and services.
14. Add notifications to an Android app using the Notification Compat API for displaying messages to the user.
15. Implement basic user authentication using Firebase Authentication in an Android app.
16. Develop a simple app using React Native that displays a list of items with basic CRUD functionality.
17. Create a responsive UI for different screen sizes and orientations in both Android and iOS using Flutter.
18. Integrate social media SDK (e.g., Facebook SDK) into your Android app for sharing content.
19. Implement error handling and logging mechanisms in your Android app to capture and report app crashes or exceptions.
20. Prepare your Android app for deployment by generating a signed APK, conducting testing on different devices, and optimizing app performance.

CIT0800404: SYSTEM ADMINISTRATION AND NETWORKING

1. **Learning Outcomes**: After successful completion of this course, students will be able to:
   a) Understand the roles and responsibilities of system administrators.
   b) Explain and manage operating systems, services, and user accounts.
   c) Utilize network tools to analyze, diagnose network traffic.
2. **Prerequisites**: Computer networks
3. **Semester**: 8
4. **Course Type**: Elective
5. **Course Level**: 400-499
6. **Theory Credit**: 3
7. **Practical Credit**: 1
8. **No of required hours**:
   a) Theory: 45 hrs
   b) Practical: 30 hrs
   c) Non Contact: 5 hrs
9. **List of Reference Books**:
   a) UNIX and Linux System Administration Handbook, Trent R. Hein, Evi Nemeth, Garth Snyder, Ben Whaley, Dan Mackin, 5th edition, Addison-Wesley
   b) Computer Networking: A Top-Down Approach by James F. Kurose and Keith W. Ross, 6th edition,Pearson Education
   c) Network Warrior, Gary A. Donahue, 2nd edition, O′Reilly

10. **Contents of Syllabus**:

A. Theory

UNIT 1: Introduction to System Administration (5 Lectures)
Role and responsibilities of a system administrator, Overview of system administration tools and technologies: vim, nano, wireshark, Clonezilla, PuTTy, FileZilla etc, Introduction to various operating systems (Windows, GNU/Linux, macOS), Introduction to virtualization software: VirtualBox, VMware, Windows Subsystem Linux(WSL) etc.

UNIT 2: Introduction to GNU/Linux (2 Lectures)
History and brief overview of GNU/Linux: Free Software Movement, Free software vs Open source software, General Public License (GPL), Linux distribution (distro)

UNIT 3: Linux Kernel and File System (10 Lectures)
Major components of the Linux operating systems, Linux kernel : linux kernel architectures, linux kernel features, Linux booting up process: POST(Power-On Self-Test), BIOS(Basic input/output system), MBR(Master boot record)/UEFI(Unified Extensible Firmware Interface), Linux File systems, linux directory structure, Types of files in linux file system.

UNIT 4: Operating System Administration using Linux (10 Lectures)
Managing users and groups: Create, modify, delete user and group accounts, File ownership of user and group, File access permissions of group and users, Process management : starting, monitoring and terminating processes, File system management: mounting and unmounting file systems, Disk management: monitoring disk usage, Backup and restore in linux.

UNIT 5: Networking Fundamentals (10 Lectures)
Basics of IPv4 and IPv6 addressing, IP address classes : Classful address and Classless Inter-Domain Routing (CIDR) notation, subnet, subnet mask, Network interfaces: ethernet, loopback, wireless etc.

IANA(Internet Assigned Numbers Authority) assigned well known ports. Brief overview of The Network Information System, Structure and function of the Domain Name
Service (DNS), DHCP (Dynamic Host Configuration Protocol),

UNIT 6: Network Management using Linux (5 Lectures)

Network interface management using ifconfig, ip tools: set, update IP address and subnet mask, bringing interfaces up and down. Usage of diagnostics tools: ping(Packet Internet or Inter-Network Groper), traceroute, Netstat and tcpdump command.

UNIT 7: Network Security (3 Lectures)

Overview of network security threats and vulnerabilities, Firewalls and intrusion detection/prevention systems (IDS/IPS), VPN (Virtual Private Network) for secure remote access, Encryption techniques (SSL/TLS, IPsec), Secure Shell (SSH)

B. List of practical

1. Design a wired LAN network using Cisco Packet Tracer - Networking Simulation Tool.
2. Design a wireless LAN network using Cisco Packet Tracer - Networking Simulation Tool.
3. Design two LAN networks with different IP ranges and connect both networks using a router using Cisco Packet Tracer - Networking Simulation Tool.
4. Design a DHCP server using Cisco Packet Tracer - Networking Simulation Tool.
5. Design an email server using Cisco Packet Tracer - Networking Simulation Tool.
6. Usage of commands ls, cat, pwd, cd, mkdir, man etc.
7. Understand the /etc/passwd file, /etc/shadow, /etc/group file.
8. Manage users and groups (create, delete, modify) using useradd, usermod, userdel, groupadd, groupmod, groupdel commands.
9. Change file ownership using the chown command.
10. Change file access permissions using the chmod command.
11. Usage of process management tools like ps, top, kill etc.
12. Usage of disk utility tools like du, df, mount etc.
13. Usage of backup and restore tools like tar, cpio.
14. Usage of net-tools like ifconfig, ip for assigning, deleting, modifying IP, subnet mask for network interfaces .
15. Analysis of network traffic with diagnostics tools like ping, net-stat, traceroute, tcpdump.

CIT0800504: MOBILE COMPUTING

1. Learning Outcomes:

a) Learn the basic concepts and principles in mobile computing including major techniques involved, and networks & systems issues for the design and implementation of mobile computing systems and applications.
b) Understand both theoretical and practical issues of mobile computing.
c) Understand the key components and technologies involved and to gain hands-on experiences in building mobile applications.

2. Prerequisites: Computer Networks

3. Semester: 8

4. Course Type: Elective

5. Course Level: 400-499

6. Theory Credit: 4

7. Practical Credit: 0

8. No of Hours:

a) Theory: 60 hrs
b) Practical: 0 hrs
c) Non-Contact: 5 hrs

9. List of Books:

a) Jochen Schilller, *Mobile Communication*, Addision wisely, Pearson Education
b) William Stallings, *Wireless Communications & Networks*, Second Edition, Pearson Education
c) Christopher Cox, *An Introduction to LTE: LTE, LTE-Advanced, SAE and 4G Mobile Communications*, Wiley publications
d) Raj Kamal, *Mobile Computing*, 2/e, Oxford University Press-New

10. Contents of Syllabus:

A. Theory

UNIT 1: Introduction to Mobile Computing (4 Lectures)

Introduction to Mobile Computing, Applications of Mobile Computing, Telecommunication Generations,Cellular systems,Electromagnetic Spectrum, Antenna, Signal Propagation, SignalCharacteristics, Multiplexing, Spread Spectrum: DSSS & FHSS, Co-channel interference, MAC Protocols.

UNIT 2: GSM Mobile Services (8 Lectures)

GSM Mobile services, System Architecture, Radio interface, Protocols,Localization and Calling, Handover, security (A3, A5 & A8), GPRS system and protocol architecture, GPRS, UTRAN, UMTS core network; Improvements on Core Network, Security.

UNIT 3: Mobile Networking (8 Lectures)

Medium Access Protocol, Internet Protocol and Transport layer, Mobile IP: IP Packet Delivery, Agent Advertisement and Discovery, Registration, Tunneling and Encapsulation, Reverse Tunneling, Mobile TCP: Traditional TCP, Classical TCP Improvements like Indirect, TCP, Snooping TCP & Mobile TCP, Fast Retransmit/ Fast Recovery,Transmission/Timeout Freezing, Selective Retransmission

UNIT 4: Wireless Local Area Networks (8 Lectures)

Wireless Local Area Networks: Introduction, Infrastructure and ad-hoc network, IEEE 802.11: System architecture, Protocol architecture, Physical layer, Medium access control layer, MAC management, 802.11a, 802.11bstandard, Wi-Fi security: WEP,WPA, Wireless LAN Threats, Securing Wireless Networks, Bluetooth: Introduction, User Scenario, Architecture, protocol stack

UNIT 5: Mobility Management (8 Lectures)

Mobility Management: Introduction, IP Mobility, Optimization, IPv6, Macro Mobility: MIPv6, FMIPv6, Micro Mobility: CellularIP, HAWAII, HMIPv6,

UNIT 6: Long-Term Evolution (LTE) of 3GPP (9 Lectures)

Long-Term Evolution (LTE) of 3GPP: LTE System Overview, Evolution from UMTS to LTE, LTE/SAE Requirements, SAE Architecture, EPS: Evolved Packet System, E-UTRAN, Voice over LTE (VoLTE),Introduction to LTE-Advanced, Self-Organizing Network (SON-LTE), SON for Heterogeneous Networks, (HetNet), Comparison between Different Generations (2G, 3G, 4G and 5G),Introduction to 5G.

CIT0800104: PROJECT AND PRESENTATION

1. **Learning Outcome:**
   a) Students will recall and describe the problem statement, objectives, and methodologies employed in their project.
   b) Students will demonstrate an understanding of the technologies explored during the project, explaining their relevance, functionality, and potential applications.
   c) Students will apply the acquired knowledge and skills to develop a solution or prototype addressing the identified problem, utilizing the chosen technologies effectively.
   d) Students will critically analyze the project outcomes, assessing the strengths and weaknesses of their approach, and identifying areas for improvement or further exploration.
   e) Students will synthesize their findings into a coherent dissertation, presenting their research methodology, results, and conclusions while evaluating the implications and significance of their work within the broader context of the field.
2. **Prerequisite:** Basic Subject knowledge
3. **Semester:** 8
4. **Course Type:** Compulsory
5. **Course Level:** 400-499
6. **Theory Credit:** 0
7. **Practical Credit:** 4
8. **Number of required hours:**
   a) Theory: 0 hrs
   b) Practical: 60 hrs
   c) Non Contact: 0hrs
9. **Course Content:**
   At the onset of their eighth semester, each student will receive an assignment for a project. Students, either individually or in pairs, will delve into a unique problem under the mentorship of a faculty member from the department. The chosen problem should allow students to delve deeply into one or two specific technologies, fostering a strong understanding and proficiency in those areas upon project completion.

   To promote innovation and avoid redundancy, previously tackled problems should be avoided unless they hold exceptional research significance and expansive scope. While application-based problems spurred by specific demands may be considered, simplistic information management systems comprising only a few database tables or data entry forms should be discouraged.

   Interdisciplinary collaboration where applicable, enabling students to draw insights from diverse fields and perspectives to enrich their projects will be encouraged. Students also have the option to conduct their projects in collaboration with other institutes or organizations, subject to approval from the relevant institute organization. However, at least one project supervisor must be affiliated with the institute or organization.

Regular progress updates must be reported by meetings with the project supervisor throughout the project duration.

Students should look for opportunities to publish their project findings in academic journals, conferences, or other relevant platforms to disseminate their research outcomes and contribute to the academic community.

Projects must culminate in the submission of a dissertation. Evaluation and presentation of projects will adhere to the regulations outlined in the PG course semester system of G.U., with choice-based credit and grading system.

10. Course Assessment Details:

Internal assessment: seminars, presentations, viva, project implementation