MK UNIVERSITY

PATAN, GUJARAT

ESTABLISHED BY THE GUJARAT GOVT. **RECOGNIZED BY UGC UNDER SECTION 2(f) OF UGC ACT,1956**

MK University, Patan
Faculty of Engineering Technology,
Department of Computer Science Engineering

DIPLOMA (COMPUTER SCIENCE ENGINEERING) SEM-I									
SR NO.	COURSE TYPE	COURSE CODE	COURSE NAME	LECTURE (HRS.)/WEEK	PRACTICAL (HRS.)/WEEK	CREDITS	EXAMINATION		TOTAL MARKS
							INTERNAL	EXTERNAL	
1	MAJOR	DCSE101	ENGINEERING MATHEMATICS-I	4	0	4	40	60	100
2	MAJOR	DCSE102	ENGINEERING PHYSICS	4	2	6	90	60	150
3	MAJOR	DCSE103	ENGINEERING CHEMISTRY	4	2	6	90	60	150
4	MAJOR	DCSE104	PROGRAMMING IN C	4	2	6	90	60	150
5	MINOR	DCSE105	WORKSHOP PRACTICE	0	2	2	50	00	50
TOTAL				16	8	24	360	240	600

DIPLOMA (COMPUTER SCIENCE ENGINEERING) SEM-II									
SR NO.	COURSE TYPE	COURSE CODE	COURSE NAME	LECTURE (HRS.)/WEEK	PRACTICAL (HRS.)/WEEK	CREDITS	EXAMINATION		TOTAL MARKS
							INTERNAL	EXTERNAL	
1	MAJOR	DCSE201	ENGINEERING MATHEMATICS-II	4	0	4	40	60	100
2	MAJOR	DCSE202	DATA STRUCTURES AND ALGORITHMS	4	2	6	90	60	150
3	MAJOR	DCSE203	DIGITAL ELECTRONICS	4	2	6	90	60	150
4	MINOR	DCSE204	WEB TECHNOLOGIES	4	2	6	90	60	150
5	SEC	DCSE205	COMMUNICATION SKILL	2	0	2	00	50	50
TOTAL				18	6	24	310	290	600

MK UNIVERSITY

PATAN, GUJARAT

ESTABLISHED BY THE GUJARAT GOVT. **RECOGNIZED BY UGC UNDER SECTION 2(f) OF UGC ACT,1956**

DIPLOMA (COMPUTER SCIENCE ENGINEERING) SEM-III									
SR NO.	COURSE TYPE	COURSE CODE	COURSE NAME	LECTURE (HRS.)/ WEEK	PRACTICAL (HRS.)/ WEEK	CREDITS	EXAMINATION		TOTAL MARKS
							INTERNAL	EXTERNAL	
1	MAJOR	DCSE301	OBJECT ORIENTED PROGRAMMING	4	2	6	90	60	150
2	MAJOR	DCSE302	COMPUTER NETWORKS	4	2	6	90	60	150
3	MAJOR	DCSE303	DATABASE MANAGEMENT SYSTEMS	4	2	6	90	60	150
4	MINOR	DCSE304	INDUSTRIAL VISIT REPORT	0	2	2	50	00	50
5	IKS	DCSE305	IKS-ANNCIENT INDIAN ENGINEERING PRACTICE	0	2	2	50	00	50
TOTAL				12	10	22	370	180	550

DIPLOMA (COMPUTER SCIENCE ENGINEERING) SEM-IV									
SR NO.	COURSE TYPE	COURSE CODE	COURSE NAME	LECTURE (HRS.)/ WEEK	PRACTICAL (HRS.)/ WEEK	CREDITS	EXAMINATION		TOTAL MARKS
							INTERNAL	EXTERNAL	
1	MAJOR	DCSE401	OPERATING SYSTEMS	4	2	6	90	60	150
2	MAJOR	DCSE402	SOFTWARE ENGINEERING	4	2	6	90	60	150
3	MAJOR	DCSE403	JAVA PROGRAMMING	4	0	4	40	60	100
4	MINOR	DCSE404	CLOUD COMPUTING BASICS	4	0	4	40	60	100
5	VAC	DCSE405	ENVIRONMENTAL SCIENCE	2	0	2	00	50	50
TOTAL				18	4	22	260	290	550

MK UNIVERSITY

PATAN, GUJARAT

ESTABLISHED BY THE GUJARAT GOVT. **RECOGNIZED BY UGC UNDER SECTION 2(f) OF UGC ACT,1956**

DIPLOMA (COMPUTER SCIENCE ENGINEERING) SEM-V									
SR NO.	COURSE TYPE	COURSE CODE	COURSE NAME	LECTURE (HRS.)/ WEEK	PRACTICAL (HRS.)/WEEK	CREDITS	EXAMINATION		TOTAL MARKS
							INTERNAL	EXTERNAL	
1	MAJOR	DCSE501	PYTHON PROGRAMMING	4	2	6	90	60	150
2	MAJOR	DCSE502	CYBER SECURITY FUNDAMENTALS	4	0	4	40	60	100
3	MAJOR	DCSE503	WEB DEVELOPMENT WITH PHP/MY SQL	4	2	6	90	60	150
4	MINOR	DCSE504	MOBILE APP DEVELOPMENT	4	0	4	40	60	100
5	SEC	DCSE505	MIN PROJECT	0	2	2	50	00	50
TOTAL				16	6	22	310	240	550

DIPLOMA (COMPUTER SCIENCE ENGINEERING) SEM-VI									
SR NO.	COURSE TYPE	COURSE CODE	COURSE NAME	LECTURE (HRS.)/ WEEK	PRACTICAL (HRS.)/WEEK	CREDITS	EXAMINATION		TOTAL MARKS
							INTERNAL	EXTERNAL	
1	MAJOR	DCSE601	INTERNET OF THINGS (IOT)	4	2	6	90	60	150
2	MAJOR	DCSE602	SOFTWARE TESTING & DEVOPS	4	2	6	90	60	150
3	MAJOR	DCSE603	ARTIFICIAL INTELLIGENCE BASICS	4	2	6	90	60	150
4	MINOR	DCSE604	DIPLOMA PROJECT	0	6	6	150	00	150
TOTAL				12	12	24	420	180	600

MK UNIVERSITY

PATAN, GUJARAT

ESTABLISHED BY THE GUJARAT GOVT. RECOGNIZED BY UGC UNDER SECTION 2(f) OF UGC ACT,1956

SEMESTER-I

COURSE CODE: DCSE101

COURSE NAME: ENGINEERING MATHEMATICS-I

Course Objectives:

- To develop foundational mathematical skills essential for mechanical engineering applications
- To apply algebraic, trigonometric, and calculus concepts to solve engineering problems
- To interpret and analyze data using statistical methods
- To build problem-solving abilities through applied mathematics
- To prepare students for advanced engineering mathematics in subsequent semesters

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

CO1	**Solve** algebraic equations and apply them to engineering problems
CO2	**Apply** trigonometric functions to analyze mechanical systems
C03	**Perform** basic differentiation and integration relevant to engineering applications
C04	**Analyze** data using measures of central tendency and dispersion

Unit	Content	Credit	Weightage
I	**Algebra and Trigonometry** **Topics:** • **Algebra:** Quadratic equations, simultaneous linear equations (2 and 3 variables) • Arithmetic and geometric progressions • **Trigonometry:** Trigonometric ratios, identities, compound angles • Heights and distances (engineering applications) • Complex numbers: basics and operations • **Applications:** Simple harmonic motion, projectile motion, force resolution	1	25%

II	**Differential Calculus**
	Topics:
	• Functions, limits, and continuity
	• Derivatives: Standard formulas
	• Rules of differentiation: Product, quotient, chain rule
	• Applications of derivatives:
	  ○ Rate of change (velocity, acceleration)
	  ○ Maxima and minima (optimization problems)
	  ○ Tangents and normals
	• Partial differentiation (introduction)
	• **Applications:** Optimization in design, motion analysis, slope of curves
		1	25%

III	**Integral Calculus** **Topics:** • Indefinite integrals: Standard formulas	1	25%

	• Methods of integration: Substitution, integration by parts
	• Definite integrals and properties
	• Applications of integration:
	  ○ Area under curves
	  ○ Volume of solids of revolution
	  ○ Center of gravity/centroid (basic concepts)
	• **Applications:** Area calculation in engineering drawings, volume of tanks/containers		

IV	**Statistics and Probability** **Topics:** • **Statistics:** Data classification, frequency distribution • Measures of central tendency: Mean, median, mode • Measures of dispersion: Range, standard deviation, variance • Graphical representation: Histogram, frequency polygon, ogive • **Probability:** Basic concepts, addition and multiplication theorems • **Applications:** Quality control, measurement analysis, manufacturing data interpretation	1	25%

Textbooks:

- Primary: *Engineering Mathematics* — NP Bali & Dr. Manish Goyal
- Primary: *A Textbook of Engineering Mathematics* — B.S. Grewal

Reference books:

- *Advanced Engineering Mathematics* — H.K. Das
- *Engineering Mathematics* — D. G. Gupta
- *Basic Technical Mathematics with Calculus* — Allyn J. Washington
- *Mathematics for Mechanical Engineering* — B.V. Ramana

Online Platforms:

1. NPTEL Videos: "Basic Course in Mathematics" for engineering
2. Coursera: "Pre-Calculus" by University of California, Irvine

COURSE CODE: DCSE102

COURSE NAME: ENGINEERING PHYSICS

Course Objectives:

- To understand fundamental physics principles relevant to mechanical engineering
- To apply physics concepts to solve practical engineering problems
- To develop skills in measurement, experimentation, and data analysis
- To correlate theoretical physics with mechanical systems and applications
- To build foundation for advanced engineering courses

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

CO1	**Apply** mechanics principles to analyze forces, motion, and energy in mechanical systems
CO2	**Explain** thermal physics concepts relevant to heat engines and refrigeration
C03	**Demonstrate** understanding of optics and acoustics in

	engineering contexts
C04	**Perform** measurements using physical instruments and analyze experimental data

Unit	Content	Credit	Weightage
I	**Mechanics & Properties of Matter** **Topics:** • Scalars and vectors, force resolution, moment of force • Laws of motion, friction, work, energy, power • Circular motion, centripetal force • Elasticity: Stress, strain, Hooke's law, Young's modulus • Surface tension and viscosity (basic concepts) • **Applications:** Machine design, material strength, fluid mechanics basics	1	25%
II	**Thermal Physics & Thermodynamics** **Topics:** • Heat and temperature, thermal expansion • Calorimetry, specific heat capacity • Laws of thermodynamics (zeroth, first, second) • Heat transfer: conduction, convection, radiation • Kinetic theory of gases (basic) • **Applications:** Heat engines, refrigeration, insulation materials	1	25%
III	**Waves, Optics & Acoustics** **Topics:** • Simple harmonic motion, wave motion • Sound: characteristics, intensity, Doppler effect • Ultrasonics and applications • Reflection, refraction, lenses, optical instruments • Fiber optics (basic principles) • **Applications:** Machine vibration, NDT, optical measurements, noise control	1	25%
IV	**Modern Physics & Material Science** **Topics:** • Quantum physics basics: photons, matter waves • Lasers: principles, types, applications • Semiconductors: basics • Superconductivity (elementary concepts) • Nanotechnology introduction • **Applications:** Laser machining, sensors, advanced materials	1	25%

Textbooks:

- Primary: *Engineering Physics* — D. R. Khanna & H. N. Srivastava
- Primary: *Engineering Physics* — R. K. Gaur & S. L. Gupta

Reference books:

- *Fundamentals of Physics* — Halliday, Resnick & Walker
- *Concepts of Physics* — H. C. Verma

- *Engineering Physics* — M. N. Avadhanulu & P. G. Kshirsagar
- *Practical Physics* — C. L. Arora

Online Platforms:

- SWAYAM/NPTEL: "Engineering Physics" courses by IITs/NITs

PRACTICAL LIST:

Section A: Mechanics

1. Vernier Calipers & Screw Gauge: Measurement of dimensions of given objects
2. Simple Pendulum: Determination of 'g' and study of laws of pendulum
3. Young's Modulus: By Searle's method or cantilever
4. Coefficient of Friction: Using inclined plane
5. Force Table: Verification of law of parallelogram of forces

Section B: Thermal Physics

6. Specific Heat Capacity: Of solid/liquid using calorimeter
7. Thermal Conductivity: Of good conductor (Searle's apparatus)
8. Mechanical Equivalent of Heat: Using Joule's calorimeter
9. Coefficient of Linear Expansion: Using optical lever

Section C: Waves & Optics

10. Sonometer: Verification of laws of vibrating strings
11. Melde's Experiment: Transverse and longitudinal modes
12. Compound Pendulum: Determination of 'g' and radius of gyration
13. Optical Bench: Focal length of convex lens
14. Prism: Refractive index using spectrometer

Section D: Modern Physics

15. LASER: Determination of wavelength using diffraction grating
16. Photoelectric Effect: Verification of Einstein's equation
17. PN Junction Diode: Characteristics
18. Thermistor: Temperature-resistance characteristics

COURSE CODE: DCSE103

COURSE NAME: ENGINEERING CHEMISTRY

Course Objectives:

- To provide fundamental knowledge of chemistry relevant to mechanical engineering applications
- To understand material properties, corrosion, fuels, and lubricants from chemical perspective
- To develop skills in chemical analysis, quality control, and material testing
- To correlate chemical principles with mechanical systems and manufacturing processes
- To build foundation for materials science, metallurgy, and environmental engineering

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

CO1	**Explain** water treatment processes for industrial applications
CO2	**Analyze** properties of fuels and lubricants used in mechanical systems
C03	**Identify** corrosion mechanisms and prevention methods
C04	**Apply** principles of electrochemistry to batteries and corrosion control

Unit	Content	Credit	Weightage

I	**Water Chemistry & Treatment** **Topics:** • Water impurities: hardness, alkalinity, pH • Water softening methods: lime-soda, ion exchange • Boiler feed water treatment: scale and sludge formation, prevention • Cooling water treatment • Drinking water standards • **Applications:** Boiler operations, cooling systems, industrial water supply	1	25%
II	**Fuels & Combustion** **Topics:** • Classification of fuels: solid, liquid, gaseous • Calorific value determination: bomb calorimeter • Solid fuels: coal analysis (proximate & ultimate) • Liquid fuels: petroleum refining, petrol, diesel, octane/cetane number • Gaseous fuels: LPG, CNG, biogas • Combustion calculations • **Applications:** IC engines, furnaces, power generation	1	25%
III	**Lubricants & Corrosion** **Topics:** • Lubrication: mechanisms, types of lubricants • Properties: viscosity index, flash point, pour point • Additives in lubricants • Corrosion: types, mechanisms (electrochemical) • Factors affecting corrosion • Corrosion prevention methods • **Applications:** Machine maintenance, automotive, industrial equipment	1	25%
IV	**Engineering Materials & Polymers** **Topics:** • Cement: composition, setting and hardening • Refractories: properties, classification • Polymers: addition, condensation, engineering plastics • Composite materials: introduction • Batteries: primary, secondary, fuel cells • **Applications:** Construction materials, polymers in engineering, energy storage	1	25%

Textbooks:

- Primary: *Engineering Chemistry* — Jain & Jain
- Primary: *Engineering Chemistry* — Dr. O. P. Verma

Reference books:

- *A Textbook of Engineering Chemistry* — S. S. Dara & S. S. Umare
- *Engineering Chemistry* — R. P. Mani & K. N. Mishra
- *Chemistry for Engineering Students* — B. S. Jai Prakash & R. Venugopal
- *Applied Chemistry* — H. D. Gesser

Online Platforms:

- SWAYAM/NPTEL: "Engineering Chemistry" courses by IITs
- Khan Academy: Chemistry fundamentals
- MERLOT: Chemistry learning materials

PRACTICAL LIST:

Section A: Water Analysis

1. Determination of Hardness: By EDTA titration method
2. Alkalinity Determination: Using acid-base titration
3. pH Measurement: Using pH meter/universal indicator
4. Chloride Content: By argentometric method

Section B: Fuels & Lubricants

5. Viscosity Measurement: Using Ostwald viscometer/Redwood viscometer
6. Flash Point Determination: Using Abel/Pensky Martens apparatus
7. Calorific Value: Bomb calorimeter demonstration
8. Saponification Value: Of oil sample

Section C: Corrosion & Electrochemistry

9. Corrosion Rate Measurement: Weight loss method
10. Galvanic Series Determination
11. Electroplating: Copper plating on iron
12. EMF Measurement: Of simple galvanic cell

Section D: Materials & Polymers

13. Cement Setting Time: Initial and final setting time
14. Polymer Identification Tests
15. Preparation of Polymer: Phenol-formaldehyde/Bakelite
16. Refractory Properties: Porosity, thermal shock resistance

COURSE CODE: DCSE104

COURSE NAME: PROGRAMMING IN C

Course Objectives:

- To introduce fundamental programming concepts and problem-solving techniques.
- To develop logical thinking and algorithmic reasoning using structured programming.
- To enable students to write, debug, and execute programs in the C language.
- To prepare a strong foundation for advanced programming, data structures, and software development.
- To familiarize students with good programming practices and documentation.

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

CO1	**Explain** basic programming concepts, algorithms, and flowchart design.
CO2	**Develop** programs using control structures, loops, and functions.
C03	**Implement** arrays, strings, and structures to handle data.
C04	**Apply** pointers and file handling concepts in real-world applications.

Unit	Content	Credit	Weightage
I	**Introduction to Programming & C Basics**	1	25%

	• Problem-solving approaches: Algorithms, Flowcharts, Pseudocode • Introduction to C: History, features, structure of a C program • C tokens: Keywords, identifiers, constants, operators • Data types: int, float, char, double • Input/output functions: printf(), scanf(), getchar(), putchar() • Operators: Arithmetic, relational, logical, assignment, bitwise • **Applications**: Simple calculator, temperature conversion		
II	**Control Structures & Loops** • Decision making: if, if-else, nested if, switch-case • Loops: while, do-while, for • Loop control statements: break, continue, goto • Nested loops and pattern printing • **Applications**: Menu-driven programs, number series, star patterns	1	25%
III	**Arrays, Strings & Functions** • Arrays: 1D and 2D arrays, initialization, traversal • Strings: Declaration, standard library functions (strlen, strcpy, strcmp) • Functions: Definition, declaration, call by value, call by reference • Recursion: Concept, examples (factorial, Fibonacci) • Storage classes: auto, static, register, extern • **Applications**: Matrix operations, sorting, searching, string manipulation	1	25%
IV	**Pointers, Structures & File Handling** • Pointers: Concept, arithmetic, pointer to array, pointer to function • Structures: Definition, accessing members, array of structures • File handling: Opening, closing, reading, writing files • Dynamic memory allocation: malloc(), calloc(), free() • **Applications**: Student record system, file-based data storage, memory management	1	25%

Textbooks:

- *Let Us C* – Yashwant Kanetkar
- *Programming in ANSI C* – E. Balagurusamy

Reference books:

- *The C Programming Language* – Brian W. Kernighan & Dennis M. Ritchie
- *C: The Complete Reference* – Herbert Schildt
- *Problem Solving and Program Design in C* – Jeri R. Hanly & Elliot B. Koffman
- *Programming with C* – Byron Gottfried

Online Platforms:

NPTEL

Programming in C by Prof. Satyadev Nandakumar
Problem Solving through Programming in C by IIT Kharagpur

Coursera

C for Everyone: Programming Fundamentals by University of California, Santa Cruz

PRACTICAL LIST:

Module I: Basics & Control Structures

1. Write a program to find the sum and average of three numbers.
2. Write a program to swap two numbers (with and without third variable).
3. Write a program to check whether a number is even or odd.
4. Write a program to find the largest among three numbers using nested if-else.
5. Write a program to display the multiplication table of a given number.
6. Write a program to print the Fibonacci series up to n terms.
7. Write a program to check whether a number is prime or not.

Module II: Arrays & Strings

8. Write a program to find the largest and smallest element in an array.
9. Write a program to add two matrices.
10. Write a program to multiply two matrices.
11. Write a program to sort an array using bubble sort.
12. Write a program to search an element in an array using linear search.
13. Write a program to reverse a string without using library functions.
14. Write a program to check whether a string is palindrome or not.

Module III: Functions & Recursion

15. Write a function to calculate factorial of a number using recursion.
16. Write a function to check whether a number is Armstrong or not.
17. Write a program to find GCD of two numbers using recursion.
18. Write a program to swap two numbers using call by reference.
19. Write a program to demonstrate the use of a static variable.

Module IV: Pointers, Structures & Files

20. Write a program to demonstrate pointer arithmetic.
21. Write a program to store student information (roll, name, marks) using structures and display it.
22. Write a program to read from a file and count vowels, consonants, and digits.
23. Write a program to copy contents from one file to another.
24. Write a program to dynamically allocate memory for an array and find its sum.

MK UNIVERSITY

PATAN, GUJARAT

ESTABLISHED BY THE GUJARAT GOVT. RECOGNIZED BY UGC UNDER SECTION 2(f) OF UGC ACT,1956

SEMESTER-II

COURSE CODE: DCSE201

COURSE NAME: ENGINEERING MATHEMATICS-II

Course Objectives:

- To build advanced mathematical skills for engineering problem-solving
- To apply differential equations to model mechanical systems
- To understand vector algebra and its applications in mechanics
- To develop skills in numerical methods for engineering computations
- To prepare mathematical foundation for subjects like Mechanics, Thermodynamics, and Machine Design

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

CO1	**Solve** ordinary differential equations relevant to engineering systems
CO2	**Apply** vector algebra to analyze forces and motions in 3D space
C03	**Perform** numerical computations using interpolation, differentiation, and integration methods
C04	**Analyze** data using probability distributions and statistical methods

Unit	Content	Credit	Weightage
I	**Differential Equations** **Topics:** • First order differential equations: variable separable, homogeneous, exact • Linear differential equations of first order • Applications: Newton's law of cooling, growth and decay, simple circuits • Second order linear differential equations with constant coefficients • Complementary function and particular integral methods • **Applications:** Spring-mass systems, electrical circuits, vibration analysis	1	25%
II	**Vector Algebra & 3D Geometry** **Topics:** • Vectors: dot product, cross product, scalar triple product • Vector differentiation • Gradient, divergence, curl (basic concepts) • Lines and planes in 3D space • Direction cosines and ratios • **Applications:** Force analysis, moment of force, work done by force	1	25%
III	**Numerical Methods** **Topics:**	1	25%

	• Solution of algebraic equations: Bisection method, Newton-Raphson method • Interpolation: Newton's forward and backward difference formulas • Numerical differentiation and integration • Trapezoidal rule and Simpson's rules • **Applications:** Root finding in design equations, area calculations, data analysis		
IV	**Probability & Laplace Transforms** **Topics:** • **Probability:** Basic concepts, addition and multiplication theorems • Random variables, probability distributions (Binomial, Poisson, Normal) • Mean, variance, standard deviation • **Laplace Transforms:** Definition, basic transforms • Properties: linearity, shifting, differentiation • Application to differential equations • **Applications:** Quality control, reliability analysis, system dynamics	1	25%

Textbooks:

- Primary: *Higher Engineering Mathematics* — B.S. Grewal
- Primary: *Engineering Mathematics* — NP Bali & Dr. Manish Goyal

Reference books:

- *Advanced Engineering Mathematics* — H.K. Das
- *Numerical Methods* — S.S. Sastry
- *Probability and Statistics for Engineers* — Dr. J. Ravichandran
- *Mathematical Methods* — B.V. Ramana

Online Platforms:

- NPTEL Videos: "Differential Equations for Engineers"
- Khan Academy: Complete probability and statistics
- MIT OCW: "Single Variable Calculus" continuation
  Coursera: "Introduction to Numerical Methods"

COURSE CODE: DCSE202

COURSE NAME: DATA STRUCTURES AND ALGORITHMS

Course Objectives:

- To introduce fundamental data structures and their implementation.
- To develop problem-solving skills using appropriate data structures.
- To analyze algorithm efficiency using time and space complexity.
- To prepare students for advanced topics, machine learning, and software development.
- To enable students to implement and apply data structures in real-world scenarios.

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

CO1	**Explain** the concepts of arrays, linked lists, stacks, and queues.
CO2	**Implement** trees and graphs for hierarchical and network data representation.
C03	**Apply** sorting and searching algorithms to solve computational

	problems.
C04	**Analyze** algorithm efficiency and select appropriate data structures for given problems.

Unit	Content	Credit	Weightage
I	**Introduction to Data Structures & Linear Lists** Introduction: Data types, ADTs, time and space complexity - Arrays: 1D, 2D, operations, applications - Linked Lists: Singly, doubly, circular linked lists - Stacks: Array and linked list implementation, applications - Queues: Simple, circular, priority queues - **Applications:** Expression evaluation, job scheduling, memory management	1	25%
II	**Trees & Hierarchical Data Structures** Trees: Terminology, binary trees, traversal (inorder, preorder, postorder) - Binary Search Trees (BST): Insertion, deletion, searching - AVL Trees: Rotations, balancing - Heaps: Min-heap, max-heap, heap operations - **Applications:** File systems, database indexing, priority queues	1	25%
III	**Graphs & Hashing** Graphs: Terminology, representation (adjacency matrix, list) - Graph traversals: BFS, DFS - Hashing: Hash functions, collision resolution (chaining, open addressing) - **Applications:** Social networks, GPS navigation, spell checking	1	25%
IV	**Sorting, Searching & Algorithm Design** Sorting algorithms: Bubble, selection, insertion, merge, quick, heap sort - Searching algorithms: Linear, binary, interpolation search - Algorithm design techniques: Greedy, divide and conquer, dynamic programming (intro) - **Applications:** Data analysis, AI search algorithms, database query optimization	1	25%

Textbooks:

- *Data Structures and Algorithms in C* — Reema Thareja
- *Data Structures Using C* — Aaron M. Tenenbaum

Reference books:

- *Introduction to Algorithms* — Thomas H. Cormen et al.
- *Data Structures and Algorithm Analysis in C* — Mark Allen Weiss
- *The Algorithm Design Manual* — Steven S. Skiena
- *Data Structures Through C* — Yashavant Kanetkar

Online Platforms:

- NPTEL:
  1. *Data Structures and Algorithms* by Prof. Naveen Garg (IIT Delhi)

2. *Programming, Data Structures and Algorithms* by IIT Madras

- Coursera:
  1. *Data Structures and Algorithms* by University of California, San Diego
  2. *Algorithms Specialization* by Stanford University

PRACTICAL LIST:

- Section A: Linear Data Structures
- Implement stack using array and linked list with push, pop, display.
- Implement queue using array and linked list with enqueue, dequeue, display.
- Implement circular queue with basic operations.
- Implement singly linked list with insertion, deletion, traversal.
- Implement doubly linked list with insertion, deletion, display.
- Implement polynomial addition using linked list.
- Section B: Trees & Heaps
- Implement binary tree traversal (inorder, preorder, postorder) recursively.
- Implement binary search tree with insertion, deletion, searching.
- Implement heap (max-heap) with insertion and deletion.
- Implement AVL tree with rotations (LL, RR, LR, RL).
- Implement expression tree from postfix expression and evaluate it.
- Section C: Graphs & Hashing
- Implement graph using adjacency matrix and perform BFS and DFS.
- Implement graph using adjacency list and perform BFS and DFS.
- Implement hashing with separate chaining for collision resolution.
- Implement hashing with linear probing for collision resolution.
- Find shortest path in unweighted graph using BFS.
- Section D: Sorting & Searching Algorithms
- Implement bubble sort, selection sort, and insertion sort.
- Implement merge sort and quick sort.
- Implement heap sort.
- Implement binary search recursively and iteratively.
- Implement linear search and count occurrences in an array.
- Implement a menu-driven program to compare sorting algorithm performance.

MK UNIVERSITY

PATAN, GUJARAT

ESTABLISHED BY THE GUJARAT GOVT. RECOGNIZED BY UGC UNDER SECTION 2(f) OF UGC ACT,1956

COURSE CODE: DCSE203

COURSE NAME: DIGITAL ELECTRONICS

Course Objectives:

- To introduce fundamental concepts of digital systems, binary arithmetic, and logic gates.
- To analyze and simplify Boolean functions using algebraic and K-map techniques.
- To design and implement combinational and sequential logic circuits.
- To develop skills in working with digital ICs, flip-flops, counters, and registers.

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

CO1	Convert number systems, apply Boolean algebra, and implement logic gates.
CO2	Simplify logic expressions and design combinational circuits (adders, multiplexers, encoders).
C03	Analyze and design sequential circuits using flip-flops, counters, and registers.
C04	Interface digital circuits with basic real-world applications.

Unit	Content	Credit	Weightage
I	**Number Systems & Logic Gates** • Number systems: binary, octal, hexadecimal, conversions. • Binary arithmetic: addition, subtraction (1's and 2's complement), multiplication. • Logic gates: AND, OR, NOT, NAND, NOR, XOR, XNOR – truth tables, symbols, ICs. • Boolean algebra: laws, De Morgan's theorem. • Logic families: TTL and CMOS characteristics, comparison.	1	25%
II	**Combinational Logic Design** • Canonical forms: SOP and POS. • Karnaugh map (up to 4 variables) – simplification of Boolean functions. • Arithmetic circuits: half adder, full adder, half subtractor, full subtractor. • Multiplexers and demultiplexers (4:1, 8:1). • Encoders and decoders (binary to decimal, BCD to 7-segment). • Parity generators and checkers.	1	25%
III	**Sequential Logic Fundamentals** • Latches: SR latch. • Flip-flops: SR, JK, D, T – truth tables, excitation tables, race-around condition in JK FF. • Master-slave JK flip-flop. • Registers: SISO, SIPO, PISO, PIPO, shift registers (bidirectional). • Counters: asynchronous (ripple) counters, synchronous counters, up/down counters.	1	25%

IV	**Advanced Sequential Circuits & Applications** • Mod-N counters, design using flip-flops. • Ring counter, Johnson counter. • Introduction to finite state machines (Mealy and Moore models). • Memory devices: ROM, RAM (basic concepts). • Introduction to PLDs: PROM, PAL, PLA (basic block diagrams only). • Simple digital applications: digital clock, traffic light control (concept only).	1	25%

Textbooks:

- *Digital Logic and Computer Design* – M. Morris Mano
- *Modern Digital Electronics* – R.P. Jain
- *Digital Electronics* – S. Salivahanan & S. Arivazhagan

Reference books:

- *Fundamentals of Digital Circuits* – A. Anand Kumar
- *Digital Design* – M. Morris Mano & Michael D. Ciletti
- *Digital Systems: Principles and Applications* – Ronald J. Tocci

Online Platforms:

- NPTEL: *Digital Circuits* by Prof. S. Srinivasan

PRACTICAL LIST:

- Verification of Truth Tables of Basic and Universal Logic Gates.
- Implementation of Boolean Functions Using Logic Gates.
- Design and Verification of Half/Full Adder and Subtractor Circuits.
- Design and Testing of 4-bit Parity Generator/Checker.
- Implementation and Testing of Multiplexer (4:1) and Demultiplexer (1:4).
- Design and Testing of Encoder (8:3) and Decoder (3:8).
- Study and Testing of Flip-Flops (SR, JK, D, T).
- Design and Testing of 4-bit Shift Register (SISO, SIPO).
- Design and Testing of Asynchronous (Ripple) Up/Down Counter.
- Design and Testing of Synchronous Mod-10 Counter.
- Interfacing BCD to 7-Segment Decoder and Display.
- Simulation of Digital Circuits Using Software (Logisim/Proteus).
- Mini-Project: Simple Digital Clock/Traffic Light Controller.

COURSE CODE: DCSE204

COURSE NAME: WEB TECHNOLOGIES

Course Objectives:

- To introduce the fundamentals of web technologies, internet protocols, and web architecture.
- To develop skills in designing responsive web pages using HTML, CSS, and JavaScript.
- To enable students to create dynamic web applications using front-end and back-end technologies.
- To understand web APIs, AJAX, and integration with databases.
- To prepare students for building web-based AI applications and interactive dashboards.

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

CO1	**Explain** web architecture, internet protocols, and client-server models.
CO2	**Design** responsive and interactive web pages using HTML, CSS, and JavaScript.
C03	**Develop** dynamic web applications using front-end frameworks and back-end scripting.
C04	**Integrate** databases and web APIs to create data-driven web applications.

Unit	Content	Credit	Weightage
I	**Web Fundamentals & HTML5** Internet basics: WWW, HTTP/HTTPS, DNS, web browsers - Web architecture: Client-server model, static vs dynamic websites - HTML5: Structure, tags, forms, multimedia, semantic elements - HTML5 APIs: Geolocation, local storage, canvas basics - **Applications:** Portfolio websites, forms, simple web pages	1	25%
II	**CSS3 & Responsive Web Design** CSS3: Selectors, box model, positioning, flexbox, grid - Responsive design: Media queries, mobile-first approach - CSS frameworks: Bootstrap basics - Animations and transitions - **Applications:** Responsive layouts, navigation bars, styled forms	1	25%
III	**JavaScript & Front-End Development** JavaScript basics: Syntax, DOM manipulation, events - ES6 features: let/const, arrow functions, promises - AJAX and Fetch API for async requests - Front-end frameworks: Introduction to React.js/Vue.js - **Applications:** Interactive UI, form validation, dynamic content loading	1	25%
IV	**Back-End Basics & Web APIs** Back-end introduction: Node.js, Express.js basics - RESTful APIs: CRUD operations, JSON handling - Database integration: MongoDB/MySQL with Node.js - Authentication: JWT, session-based auth basics - **Applications:** Simple web apps, API integration, user authentication	1	25%

Textbooks:

- *Web Technologies: HTML, JavaScript, PHP, Java, JSP, XML and AJAX* — Uttam K. Roy
- *HTML & CSS: Design and Build Websites* — Jon Duckett

Reference books:

- *Eloquent JavaScript* — Marijn Haverbeke
- *Learning PHP, MySQL & JavaScript* — Robin Nixon
- *Node.js Design Patterns* — Mario Casciaro

- *React: Up & Running* — Stoyan Stefanov

Online Platforms:

- NPTEL:
  1. *Web Technologies* by Prof. D. Janakiram (IIT Madras)
  2. *Introduction to Modern Application Development* by IIT Madras
- Coursera:
  1. *HTML, CSS, and JavaScript for Web Developers* by Johns Hopkins University
  2. *Web Design for Everybody* by University of Michigan

PRACTICAL LIST:

Section A: HTML5 & CSS3

1. Create a personal portfolio website using HTML5 and CSS3.
2. Design a responsive registration form with validation using HTML5 form elements.
3. Build a webpage layout using CSS Grid and Flexbox.
4. Implement a navigation bar with dropdown menus using CSS.
5. Create a photo gallery with hover effects and transitions.

Section B: JavaScript & DOM Manipulation

6. Create a calculator using JavaScript.
7. Build a to-do list application with add, edit, delete, and mark-as-done features.
8. Develop a digital clock with date and time display.
9. Implement form validation (email, password, phone) using JavaScript.
10. Create a slideshow/carousel using JavaScript and CSS.

Section C: Responsive Design & Bootstrap

11. Convert a given webpage into a mobile-responsive layout using media queries.
12. Design a landing page using Bootstrap 5 (navbar, cards, modal, forms).
13. Create a responsive dashboard layout with Bootstrap grid system.
14. Build a blog template with Bootstrap components.

Section D: Back-End Integration & APIs

15. Create a simple REST API using Node.js and Express that returns JSON data.
16. Build a weather app using a public API (e.g., OpenWeatherMap) with Fetch/AJAX.
17. Develop a note-taking app with local storage (CRUD operations).

MK UNIVERSITY

PATAN, GUJARAT

ESTABLISHED BY THE GUJARAT GOVT. RECOGNIZED BY UGC UNDER SECTION 2(f) OF UGC ACT,1956

SEMESTER-III

COURSE CODE: DCSE301

COURSE NAME: OBJECT ORIENTED PROGRAMMING C++

Course Objectives:

- To introduce the principles and concepts of Object-Oriented Programming (OOP).
- To develop skills in designing, implementing, and testing C++ programs using OOP features.
- To enable students to understand and apply encapsulation, inheritance, polymorphism, and abstraction.
- To prepare a foundation for advanced programming, software development, and system design.
- To foster good programming practices and problem-solving using OOP.

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

CO1	**Explain** the fundamentals of OOP and the structure of C++ programs.
CO2	**Implement** classes, objects, constructors, and destructors in C++.
C03	**Apply** inheritance, polymorphism, and operator overloading in program design.
C04	**Develop** C++ programs using file handling, templates, and exception handling.

Unit	Content	Credit	Weightage
I	**Introduction to OOP & C++ Basics** • Introduction to OOP: Concepts, advantages over procedural programming • C++ basics: Structure of a C++ program, iostream, namespace • Data types, operators, and control structures in C++ • Functions: Function overloading, default arguments, inline functions • Introduction to classes and objects (basic) • **Applications**: Simple calculator, menu-driven programs	1	25%
II	**Classes, Objects & Constructors** • Classes and objects: Declaration, definition, access specifiers • Member functions: Inside and outside class definition • Constructors: Default, parameterized, copy constructors • Destructors • this pointer • Friend functions and friend classes • **Applications**: Student record system, bank account management	1	25%
III	**Inheritance & Polymorphism**	1	25%

	• Inheritance: Types (single, multiple, multilevel, hierarchical, hybrid) • Access specifiers in inheritance: public, protected, private • Polymorphism: Compile-time and run-time polymorphism • Function overriding and virtual functions • Abstract classes and pure virtual functions • Operator overloading: Unary and binary operators • **Applications**: Payroll system, shape hierarchy, library management		
IV	**Advanced C++ Features** • File handling in C++: if stream, of stream, f stream • Templates: Function templates and class templates • Exception handling: try, catch, throw • Standard Template Library (STL) basics: vector, list, map • Dynamic memory allocation: new and delete operators • **Applications**: File-based data storage, generic programming, error handling	1	25%

Textbooks:

- *Object-Oriented Programming with C++* – E. Balagurusamy
- *Let Us C++* – Yashwant Kanetkar

Reference books:

- *The C++ Programming Language* – Bjarne Stroustrup
- *C++: The Complete Reference* – Herbert Schildt
- *Object-Oriented Programming in C++* – Robert Lafore
- *Programming: Principles and Practice Using C++* – Bjarne Stroustrup

Online Platforms:

- NPTEL
  - *Programming in C++* by Prof. P. P. Chakraborty (IIT Kharagpur)
  - *Object-Oriented Programming in C++* by IIT Bombay
- Coursera
  - *Object-Oriented Programming in C++* by University of London
  - *C++ For C Programmers* by University of California, Santa Cruz

PRACTICAL LIST:

Module I: C++ Basics & Functions

1. Write a C++ program to find the sum and average of three numbers.
2. Write a program to swap two numbers using call by reference.
3. Write a program to check whether a number is prime or not.
4. Write a program to display the Fibonacci series using recursion.
5. Write a program to demonstrate function overloading (add numbers and concatenate strings).
6. Write a program to find the factorial of a number using inline function.

Module II: Classes & Objects

7. Write a program to create a class Student with data members roll_no, name, marks and member functions to input and display data.
8. Write a program to implement a class Circle with member functions to calculate area and circumference.
9. Write a program to demonstrate the use of constructors (default, parameterized, copy).

10. Write a program to show the use of this pointer.
11. Write a program to demonstrate friend function accessing private members of two different classes.

Module III: Inheritance & Polymorphism

12. Write a program to implement single inheritance: class Employee (base) and Manager (derived).
13. Write a program to demonstrate multiple inheritance.
14. Write a program to implement function overriding and use of virtual functions.
15. Write a program to create an abstract class Shape with pure virtual function area().
16. Write a program to overload + operator to add two complex numbers.

Module IV: Advanced C++ Features

17. Write a program to read from a file and count the number of words.
18. Write a program to write student records into a file and display them.
19. Write a program to demonstrate function template for swapping two values of any data type.
20. Write a program to implement exception handling for division by zero.

COURSE CODE: DCSE302

COURSE NAME: COMPUTER NETWORKS

Course Objectives:

- To introduce fundamental concepts, architectures, and models of computer networks.
- To develop understanding of network protocols, data communication, and network devices.
- To enable students to design, configure, and troubleshoot basic network setups.
- To prepare students for understanding network-related aspects in AI, IoT, and cloud systems.
- To foster awareness of network security, performance, and ethical considerations.

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

CO1	**Explain** network models, topologies, and transmission media.
CO2	**Describe** the functions and protocols of data link, network, and transport layers.
C03	**Configure** basic network services and understand routing principles.
C04	**Analyze** network security threats and apply basic security measures.

Unit	Content	Credit	Weightage
I	**Introduction to Computer Networks & Physical Layer** Basics: Components, network types (LAN, MAN, WAN), topologies - Network models: OSI model, TCP/IP model, layers and functions - Transmission media: Guided (twisted pair, coaxial, fiber) and unguided (radio, microwave) - Switching techniques: Circuit, packet, message switching - **Applications:** Home networks, office networks, internet connectivity	1	25%
II	**Data Link Layer & Media Access Control** Functions of data link layer: Framing, error detection (CRC), flow control	1	25%

	- MAC protocols: CSMA/CD, CSMA/CA, Ethernet (IEEE 802.3) - Network devices: Hubs, switches, bridges - VLANs (Virtual LANs) – basic concepts - **Applications:** Ethernet LANs, Wi-Fi networks, network segmentation		
III	**Network Layer & Routing** IPv4 addressing: Classes, subnetting, CIDR - IPv6 basics - Routing algorithms: Static vs dynamic routing, distance vector, link state - Routers and routing tables - ARP, ICMP, DHCP basics - **Applications:** IP addressing, routing in networks, internet connectivity	1	25%
IV	**Transport Layer, Application Layer & Network Security** Transport layer: TCP and UDP, ports, sockets, congestion control - Application layer protocols: HTTP, HTTPS, DNS, SMTP, FTP - Network security: Threats (malware, DoS, phishing), firewalls, encryption basics - Introduction to IoT networks and cloud networking - **Applications:** Web browsing, email, secure communication, IoT devices	1	25%

Textbooks:

- *Data Communications and Networking* — Behrouz A. Forouzan
- *Computer Networks* — Andrew S. Tanenbaum

Reference books:

- *TCP/IP Illustrated* — W. Richard Stevens
- *Computer Networking: A Top-Down Approach* — James F. Kurose & Keith W. Ross
- *Network Security Essentials* — William Stallings
- *CCNA Routing and Switching Complete Study Guide* — Todd Lammle

Online Platforms:

- NPTEL:
  1. *Computer Networks* by Prof. Sujoy Ghosh (IIT Kharagpur)
  2. *Introduction to Computer Networks* by IIT Bombay
- Coursera:
  1. *The Bits and Bytes of Computer Networking* by Google
  2. *Computer Networks* by University of Washington

PRACTICAL LIST:

Section A: Network Basics & Configuration

1. Identify network components and create a simple LAN topology diagram.
2. Configure IP addresses on multiple devices and verify connectivity using ping.
3. Use basic network commands: ipconfig, ifconfig, ping, tracert/traceroute.
4. Create a crossover cable and test connectivity between two PCs.

Section B: Data Link Layer & Switching

5. Configure a VLAN on a switch and assign ports to VLANs.
6. Analyze Ethernet frames using Wireshark.
7. Simulate CSMA/CD in a network simulator (e.g., Cisco Packet Tracer).
8. Configure port security on a switch.

Section C: Network Layer & Routing

9. Perform subnetting exercises: Calculate subnets, network addresses, broadcast addresses.
10. Configure static routing between two or more routers.
11. Configure DHCP server on a router and assign IPs dynamically.
12. Analyze IP packets using Wireshark.

Section D: Transport/Application Layer & Security

13. Configure a simple web server and access it via HTTP/HTTPS.
14. Use nslookup and dig commands for DNS query analysis.
15. Set up a basic firewall rule (using Windows Firewall or iptables basics).
16. Capture and analyze TCP/UDP packets using Wireshark.
17. Mini-project: Design and simulate a small office network with multiple VLANs, routing, and basic security.

COURSE CODE: DCSE303

COURSE NAME: DATABASE MANAGEMENT SYSTEMS

Course Objectives:

- To introduce fundamental concepts of databases, data models, and DBMS architecture.
- To develop skills in designing databases using ER modeling and normalization.
- To enable students to write SQL queries for data definition, manipulation, and control.
- To understand transaction management, concurrency control, and database security.
- To prepare students for real-world database applications in AI and data science.

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

CO1	**Explain** DBMS architecture, data models, and ER diagrams.
CO2	**Design** normalized relational databases using ER modeling and normalization.
C03	**Write** SQL queries for data definition, manipulation, and retrieval.
C04	**Analyze** transaction management, concurrency control, and database security mechanisms.

Unit	Content	Credit	Weightage
I	**Introduction to DBMS & Data Modeling** Database concepts: Data, database, DBMS, advantages - DBMS architecture: 3-tier architecture, data independence - Data models: Hierarchical, network, relational, ER model - ER diagrams: Entities, attributes, relationships, keys - **Applications:** Banking, inventory, student management systems	1	25%
II	**Relational Database Design & SQL** Relational model: Relations, tuples, attributes, keys - Relational algebra: Operations, selection, projection, join - SQL: DDL, DML, DCL commands	1	25%

	- SQL queries: SELECT, WHERE, GROUP BY, HAVING, ORDER BY - Joins: Inner, outer, self, cross joins - **Applications:** Querying real databases, report generation		
III	**Normalization & Advanced SQL** Normalization: 1NF, 2NF, 3NF, BCNF, functional dependencies - Advanced SQL: Subqueries, views, indexes, sequences - PL/SQL basics: Cursors, procedures, functions, triggers - **Applications:** Data integrity, performance tuning, automation	1	25%
IV	**Transaction Management & Security** Transaction concepts: ACID properties, states - Concurrency control: Locks, two-phase locking, deadlock - Database recovery: Log-based recovery, checkpoints - Database security: Authentication, authorization, encryption - **Applications:** Banking transactions, multi-user systems, secure databases	1	25%

Textbooks:

- *Database System Concepts* — Abraham Silberschatz, Henry F. Korth, S. Sudarshan
- *Fundamentals of Database Systems* — Ramez Elmasri, Shamkant B. Navathe

Reference books:

- *SQL: The Complete Reference* — James R. Groff, Paul N. Weinberg
- *Database Management Systems* — Raghu Ramakrishnan, Johannes Gehrke
- *An Introduction to Database Systems* — C.J. Date
- *Oracle PL/SQL Programming* — Steven Feuerstein

Online Platforms:

- NPTEL:
  1. *Database Management System* by Prof. P.K. Biswas (IIT Kharagpur)
  2. *Introduction to Database Systems* by IIT Madras
- Coursera:
  1. *Databases and SQL for Data Science* by IBM
  2. *SQL for Data Science* by University of California, Davis

PRACTICAL LIST:

Section A: Database Design & ER Modeling

1. Draw ER diagrams for:
   - Library Management System
   - Hospital Management System
   - Online Shopping System
2. Convert ER diagrams into relational schemas.
3. Create tables using DDL commands with primary keys, foreign keys, and constraints.

Section B: SQL Queries – Basic to Intermediate

4. Create a database for Student Management System with tables: Student, Course, Enrollment.
5. Insert sample data into all tables.
6. Write SQL queries to:
   - Retrieve all students enrolled in a specific course.
   - Find the average marks of students.

- List students with marks above average.
- Count number of students per course.

7. Implement different types of joins (inner, left, right, full) on the database.

Section C: Advanced SQL & PL/SQL

8. Create views for:
   - Students with marks > 75%
   - Course-wise student count
9. Write SQL queries using:
   - Subqueries (nested, correlated)
   - GROUP BY, HAVING, ORDER BY
10. Create indexes on frequently queried columns.
11. Write a PL/SQL procedure to calculate total marks of a student.
12. Create a trigger to log changes in the Student table.

Section D: Normalization & Transaction Control

13. Take an unnormalized table and normalize it to 3NF.
14. Demonstrate transaction control commands:

- COMMIT, ROLLBACK, SAVEPOINT

15. Simulate concurrent transactions and demonstrate locking.
16. Implement user roles and grant/revoke permissions.
17. Perform backup and recovery operations on a sample database.

MK UNIVERSITY

PATAN, GUJARAT

ESTABLISHED BY THE GUJARAT GOVT. RECOGNIZED BY UGC UNDER SECTION 2(f) OF UGC ACT,1956

SEMESTER-IV

COURSE CODE: DCSE401

COURSE NAME: OPERATING SYSTEMS

Course Objectives:

- To introduce fundamental concepts, functions, and structures of operating systems.
- To develop understanding of process management, CPU scheduling, and synchronization.
- To enable students to comprehend memory management, file systems, and storage organization.
- To prepare students for understanding system-level programming and modern OS architectures.
- To foster awareness of security, protection, and distributed systems basics.

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

CO1	**Explain** OS structures, functions, and types of operating systems.
CO2	**Analyze** process management, scheduling algorithms, and synchronization mechanisms.
C03	**Describe** memory management techniques, paging, segmentation, and virtual memory.
C04	**Understand** file systems, storage management, and basic OS security concepts.

Unit	Content	Credit	Weightage
I	**Introduction to OS & System Structures** • Introduction to OS: Definition, objectives, functions • Types of OS: Batch, multiprogramming, time-sharing, real-time, distributed • OS structure: Monolithic, layered, microkernel, modular • System calls and types • Operating system services • **Applications**: Command-line interface, shell scripting basics	1	25%
II	**Process Management & CPU Scheduling** • Process concept: Process states, PCB, operations • Threads: Multithreading models, benefits • CPU scheduling: Criteria, algorithms (FCFS, SJF, Priority, Round Robin) • Process synchronization: Critical section, semaphores, mutex • Deadlocks: Conditions, prevention, avoidance, detection, recovery • **Applications**: Task scheduling in OS, multi-threaded applications	1	25%
III	**Memory Management** • Memory hierarchy: Registers, cache, RAM, secondary storage • Contiguous memory allocation: Fixed and variable partition	1	25%

	• Fragmentation: Internal and external • Paging: Basic concept, page table, TLB • Segmentation • Virtual memory: Demand paging, page replacement algorithms (FIFO, LRU, Optimal) • **Applications**: Memory allocation in programs, virtual memory management		
IV	**File Systems & OS Security** • File concepts: Attributes, operations, types • Directory structure: Single-level, two-level, tree-structured • File allocation methods: Contiguous, linked, indexed • Disk scheduling algorithms: FCFS, SSTF, SCAN, C-SCAN • Protection and security: Access control, authentication, threats, malware • **Applications**: File management in OS, basic disk management, user authentication	1	25%

Textbooks:

- *Operating System Concepts* – Abraham Silberschatz, Peter B. Galvin, Greg Gagne
- *Operating Systems: A Concept-Based Approach* – D. M. Dhamdhere

Reference books:

- *Modern Operating Systems* – Andrew S. Tanenbaum
- *Operating Systems: Principles and Design* – P. C. P. Bhatt
- *Operating Systems: Internals and Design Principles* – William Stallings
- *Operating Systems* – Achyut S. Godbole & Atul Kahate

Online Platforms:

- NPTEL
  - *Operating Systems* by Prof. P. K. Biswas (IIT Kharagpur)
  - *Introduction to Operating Systems* by IIT Madras
- Coursera
  - *Operating Systems and You: Becoming a Power User* by Google
  - *Introduction to Operating Systems* by University of Colorado

PRACTICAL LIST:

Module I: Introduction & System Calls

1. Write a shell script to display system information (OS version, kernel, memory).
2. Write a shell script to list files and directories in a given path.
3. Write a C program to create a child process using fork() system call.
4. Write a program to demonstrate the use of exec() family of system calls.
5. Write a shell script to automate backup of a directory.

Module II: Process Scheduling & Synchronization

6. Write a C program to simulate FCFS CPU scheduling algorithm.
7. Write a C program to simulate Round Robin CPU scheduling.
8. Write a program to implement SJF (Shortest Job First) scheduling.
9. Write a program to demonstrate producer-consumer problem using semaphores.
10. Write a program to simulate deadlock detection algorithm.

Module III: Memory Management

11. Write a C program to simulate First Fit memory allocation.
12. Write a program to simulate Best Fit memory allocation.

13. Write a program to simulate FIFO page replacement algorithm.
14. Write a program to simulate LRU page replacement algorithm.
15. Write a program to calculate internal and external fragmentation.

Module IV: File Systems & Disk Scheduling

16. Write a C program to simulate single-level directory structure.
17. Write a program to simulate two-level directory structure.
18. Write a program to simulate FCFS disk scheduling algorithm.
19. Write a program to simulate SCAN disk scheduling algorithm.

COURSE CODE: DCSE402

COURSE NAME: SOFTWARE ENGINEERING

Course Objectives:

- To introduce fundamental concepts, principles, and processes of software engineering.
- To develop skills in software requirements analysis, design, and modeling techniques.
- To enable students to understand software development methodologies and testing strategies.
- To prepare students for roles in software project planning, management, and quality assurance.
- To foster awareness of software maintenance, reusability, and ethical practices in software development.

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

CO1	**Explain** software engineering concepts, life cycle models, and requirement engineering.
CO2	**Apply** software design principles, modeling techniques, and architectural styles.
C03	**Implement** software testing strategies, verification, and validation methods.
C04	**Analyze** software project management, maintenance, and quality assurance practices.

Unit	Content	Credit	Weightage
I	**Introduction to SE & Process Models** • Introduction to Software Engineering: Definition, importance, challenges • Software Development Life Cycle (SDLC) phases • Process models: Waterfall, Iterative, Incremental, Spiral, V-Model • Agile methodologies: Scrum, Kanban, Extreme Programming (XP) • Software process metrics and improvement • **Applications**: Case studies of software projects	1	25%
II	**Requirement Engineering & Design** • Requirements engineering: Elicitation, analysis, specification, validation • Types of requirements: Functional and non-functional • Use case modeling and diagrams • Software design: Architectural design, modular design, cohesion, coupling	1	25%

	• Design principles: Abstraction, decomposition, modularity, information hiding • UML diagrams: Class, sequence, activity, state diagrams • **Applications**: SRS preparation, design document creation		
III	**Software Testing & Quality Assurance** • Software testing fundamentals: Objectives, principles, levels • Testing strategies: Unit testing, integration testing, system testing • Testing techniques: Black-box, white-box, regression, acceptance • Software quality assurance: Standards, quality metrics, reviews, inspections • Introduction to automated testing tools • **Applications**: Test case design, bug tracking, quality metrics calculation	1	25%
IV	**Project Management & Maintenance** • Software project management: Planning, scheduling, risk management • Cost estimation techniques: COCOMO, Function Point Analysis • Software configuration management: Version control, change management • Software maintenance: Types, maintenance models, re-engineering • Software ethics and professional practices • **Applications**: Project planning using Gantt charts, version control with Git	1	25%

Textbooks:

- *Software Engineering: A Practitioner's Approach* – Roger S. Pressman
- *Software Engineering* – Ian Sommerville

Reference books:

- *Fundamentals of Software Engineering* – Rajib Mall
- *Software Engineering: Concepts and Practices* – Ugrasen Suman
- *Software Engineering Principles and Practice* – Hans van Vliet
- *Agile Software Development: Principles, Patterns, and Practices* – Robert C. Martin

Online Platforms:

- NPTEL
  - *Software Engineering* by Prof. Rajib Mall (IIT Kharagpur)
  - *Software Project Management* by IIT Bombay
- Coursera
  - *Software Development Lifecycle* by University of Minnesota
  - *Introduction to Software Engineering* by IBM

PRACTICAL LIST:

Module I: Process Models & Agile

1. Prepare a report comparing Waterfall and Agile methodologies with examples.
2. Create a case study analysis of a software project using the Spiral model.

3. Develop a basic Scrum board (digital/paper) for a small project.
4. Draw a V-Model diagram and explain each phase with examples.

Module II: Requirement Analysis & Design

5. Write a Software Requirements Specification (SRS) for a Library Management System.
6. Create use case diagrams for an Online Shopping System.
7. Design class diagrams for a Hospital Management System.
8. Create sequence diagrams for ATM transaction processing.
9. Prepare a modular design structure for a Student Information System.

Module III: Testing & Quality Assurance

10. Write test cases for a login module (black-box testing).
11. Perform white-box testing on a small program (e.g., find maximum of three numbers).
12. Design test cases for integration testing of a payroll system.
13. Prepare a bug report using a bug tracking template.
14. Calculate software metrics (e.g., cyclomatic complexity) for a given code.

Module IV: Project Management & Tools

15. Create a project schedule using Gantt chart (manual/tool-based).
16. Estimate the cost of a software project using basic COCOMO.
17. Set up a Git repository and perform basic version control operations (clone, commit, push, branch).
18. Perform a change management simulation for a software requirement change.
19. Prepare a software maintenance plan for an existing system.
20. Write a report on ethical issues in software development (privacy, security, piracy).

COURSE CODE: DCSE403

COURSE NAME: JAVA PROGRAMMING

Course Objectives:

- To introduce fundamental concepts and principles of Java programming.
- To develop skills in object-oriented programming using Java syntax and constructs.
- To enable students to implement exception handling, multithreading, and file I/O in Java.
- To prepare students for developing GUI-based applications using AWT/Swing.
- To foster understanding of Java collections, networking, and database connectivity (JDBC).

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

CO1	**Explain** Java fundamentals, object-oriented principles, and language syntax.
CO2	**Develop** Java applications using classes, inheritance, polymorphism, and interfaces.
C03	**Implement** exception handling, multithreading, and file handling in Java.
C04	**Design** GUI applications and integrate databases using JDBC.

Unit	Content	Credit	Weightage
I	**Introduction to Java & OOP Basics** • Introduction to Java: Features, JDK, JRE, JVM • Java program structure, compilation, and execution • Data types, operators, control structures	1	25%

	• Classes and objects: Declaration, constructors, methods • Method overloading, static keyword • Arrays and strings in Java • **Applications**: Simple calculator, student grade system		
II	**Advanced OOP & Exception Handling** • Inheritance: Types, super keyword, method overriding • Polymorphism: Dynamic binding, abstract classes, interfaces • Packages: Creation, import, access protection • Exception handling: try, catch, throw, throws, finally • User-defined exceptions • **Applications**: Banking system, employee payroll system	1	25%
III	**Multithreading & File I/O** • Multithreading: Thread life cycle, creating threads (extends Thread, implements Runnable) • Thread synchronization, inter-thread communication • File handling: FileInputStream, FileOutputStream, BufferedReader, BufferedWriter • Serialization and deserialization • **Applications**: File copy utility, multithreaded number printer	1	25%
IV	**GUI, JDBC & Networking Basics** • GUI programming: AWT components, event handling, layout managers • Swing basics: JFrame, JButton, JTextField, JLabel • Introduction to JDBC: Architecture, drivers, CRUD operations • Basic networking: Socket and ServerSocket (client-server model) • **Applications**: Student registration form, database connectivity app, simple chat client	1	25%

Textbooks:

- *Java: The Complete Reference* – Herbert Schildt
- *Core Java: Volume I – Fundamentals* – Cay S. Horstmann

Reference books:

- *Head First Java* – Kathy Sierra & Bert Bates
- *Thinking in Java* – Bruce Eckel
- *Java Programming: From Problem Analysis to Program Design* – D. S. Malik
- *Programming with Java* – E. Balagurusamy

Online Platforms:

- NPTEL
  - *Programming in Java* by Prof. Debasis Samanta (IIT Kharagpur)
  - *Object-Oriented Programming in Java* by IIT Madras
- Coursera
  - *Java Programming and Software Engineering Fundamentals* by Duke University
  - *Object-Oriented Programming in Java* by University of California San Diego

MK UNIVERSITY

PATAN, GUJARAT

ESTABLISHED BY THE GUJARAT GOVT. RECOGNIZED BY UGC UNDER SECTION 2(f) OF UGC ACT,1956

COURSE CODE: DCSE404

COURSE NAME: CLOUD COMPUTING BASICS

Course Objectives:

- To introduce the fundamental concepts, models, and architectures of cloud computing.
- To develop skills in deploying, managing, and scaling applications using major cloud platforms.
- To enable students to leverage cloud services for AI/ML model training, storage, and deployment.
- To prepare students for cloud-native development and understanding of security, compliance, and cost management.
- To foster awareness of emerging trends such as serverless computing, containers, and hybrid cloud.

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

CO1	**Explain** cloud computing models, service models, and deployment models.
CO2	**Deploy** and manage virtual machines, storage, and networking in a cloud environment.
C03	**Utilize** cloud-based AI/ML services for model training and deployment.
C04	**Apply** cloud security best practices and manage cloud costs effectively.

Unit	Content	Credit	Weightage
I	**Introduction to Cloud Computing** What is Cloud Computing? Evolution, characteristics, benefits, and challenges - Cloud service models: IaaS, PaaS, SaaS, FaaS - Cloud deployment models: Public, private, hybrid, community cloud - Key cloud providers: AWS, Azure, GCP overview - **Applications:** Web hosting, data storage, scalable computing	1	25%
II	**Cloud Infrastructure & Services** Virtualization: Hypervisors, virtual machines, containers (Docker basics) - Compute services: EC2 (AWS), VM (Azure), Compute Engine (GCP) - Storage services: S3, Blob Storage, Cloud Storage - Networking: VPC, load balancers, CDN basics - **Applications:** Deploying web apps, media streaming, backup solutions	1	25%
III	**Cloud for AI & DevOps** AI/ML cloud services: AWS SageMaker, Azure ML, GCP AI Platform - Serverless computing: AWS Lambda, Azure Functions - Containers and orchestration: Docker, Kubernetes basics - DevOps in the cloud: CI/CD pipelines (GitHub Actions, Jenkins basics) - **Applications:** Training ML models, auto-scaling apps,	1	25%

	microservices		
IV	**Cloud Security, Management & Trends** Cloud security: Identity and Access Management (IAM), encryption, compliance - Cost management: Pricing models, budgeting, monitoring tools - Migration strategies: Lift-and-shift, re-platforming, cloud-native - Emerging trends: Edge computing, hybrid cloud, sustainable cloud - **Applications:** Secure cloud deployments, cost-optimized architectures, green IT	1	25%

Textbooks:

- *Cloud Computing: Concepts, Technology & Architecture* — Thomas Erl et al.
- *Cloud Computing for Dummies* — Judith Hurwitz et al.

Reference books:

- *Architecting the Cloud* — Michael J. Kavis
- *Cloud Native Transformation* — Pini Reznik et al.
- *AWS Certified Solutions Architect Study Guide* — Ben Piper & David Clinton
- *Cloud Computing Bible* — Barrie Sosinsky

Online Platforms:

- NPTEL:
  1. *Cloud Computing* by Prof. Soumya Kanti Ghosh (IIT Kharagpur)
  2. *Introduction to Cloud Computing* by IIT Madras
- Coursera:
  1. *Cloud Computing Specialization* by University of Illinois
  2. *AWS Fundamentals* by Amazon Web Services

MK UNIVERSITY

PATAN, GUJARAT

ESTABLISHED BY THE GUJARAT GOVT. RECOGNIZED BY UGC UNDER SECTION 2(f) OF UGC ACT,1956

SEMESTER-V

COURSE CODE: DCSE501

COURSE NAME: PYTHON PROGRAMMING

Course Objectives:

- To introduce fundamental concepts of Python programming and its applications.
- To develop skills in writing Python programs using control structures, functions, and data structures.
- To enable students to implement file handling, exception handling, and modular programming in Python.
- To prepare students for advanced topics like OOP, database connectivity, and data processing.
- To foster problem-solving abilities using Python for real-world applications.

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

CO1	**Explain** Python basics, data types, control structures, and functions.
CO2	**Implement** programs using lists, tuples, dictionaries, sets, and strings.
C03	**Apply** file handling, exception handling, and modular programming in Python.
C04	**Develop** Python applications using OOP, modules, and database connectivity.

Unit	Content	Credit	Weightage
I	**Python Basics & Control Structures** • Introduction to Python: Features, installation, IDLE, Jupyter Notebook • Python syntax, variables, data types, operators • Input/output functions: input(), print() • Control structures: if, if-else, nested if, loops (while, for) • Loop control: break, continue, pass • **Applications**: Simple calculator, number guessing game, pattern printing	1	25%
II	**Data Structures & Functions** • Python data structures: Lists, tuples, dictionaries, sets • String manipulation: Methods, slicing, formatting • Functions: Definition, parameters, return values, scope • Lambda functions, map, filter, reduce • List comprehensions • **Applications**: Student mark processing, word frequency counter, to-do list app	1	25%
III	**File Handling & Exception Handling** • File handling: Opening, reading, writing, appending (open(), read(), write()) • File modes, with statement • Exception handling: try, except, else, finally	1	25%

	• User-defined exceptions • **Applications**: Log file analyzer, data backup script, error logging system		
IV	**OOP, Modules & Database Connectivity** • Object-Oriented Programming: Classes, objects, constructors, methods • Inheritance, polymorphism, encapsulation • Modules and packages: Creating, importing • Introduction to database connectivity using sqlite3 or mysql-connector • CRUD operations in Python • **Applications**: Employee management system, library system, simple CRM	1	25%

Textbooks:

- *Python Programming: A Modern Approach* – Vamsi Kurama
- *Learning Python* – Mark Lutz

Reference books:

- *Python Crash Course* – Eric Matthes
- *Core Python Programming* – R. Nageswara Rao
- *Python Cookbook* – David Beazley & Brian K. Jones
- *Head First Python* – Paul Barry

Online Platforms:

- NPTEL
  - *Programming, Data Structures and Algorithms in Python* by IIT Madras
  - *Python for Data Science* by IIT Roorkee
- Coursera
  - *Python for Everybody* by University of Michigan
  - *Crash Course on Python* by Google

PRACTICAL LIST:

Module I: Python Basics & Control Structures

1. Write a program to calculate the area of a circle, rectangle, and triangle.
2. Write a program to check if a number is even or odd.
3. Write a program to print the multiplication table of a given number.
4. Write a program to find the factorial of a number using a loop.
5. Write a program to check if a string is a palindrome.
6. Write a program to generate Fibonacci series up to n terms.

Module II: Data Structures & Functions

7. Write a program to find the largest and smallest elements in a list.
8. Write a program to count the frequency of each character in a string using a dictionary.
9. Write a program to merge two dictionaries.
10. Write a function to check if a number is prime.
11. Write a program using list comprehension to create a list of squares from 1 to 10.
12. Write a program to sort a list of tuples based on the second element.

Module III: File Handling & Exception Handling

13. Write a program to read a text file and count the number of lines, words, and characters.
14. Write a program to copy the contents of one file to another.
15. Write a program to handle ZeroDivisionError and ValueError exceptions.

16. Write a program to log errors into a file when exceptions occur.
17. Write a program to read CSV file data and display it in tabular format.
    Module IV: OOP, Modules & Database Connectivity
18. Create a class Student with attributes name, roll, marks and methods to display details.
19. Implement inheritance: class Vehicle (base) and Car (derived).
20. Create a module calculator.py with functions for basic operations and import it in another script.
21. Write a program to connect to a SQLite database and create a table Employees.
22. Write a program to insert, update, delete, and retrieve records from the Employees table.

COURSE CODE: DCSE502

COURSE NAME: CYBER SECURITY FUNDAMENTALS

Course Objectives:

- To introduce the foundational concepts, principles, and importance of cybersecurity.
- To develop an understanding of common cyber threats, vulnerabilities, and attack vectors.
- To enable students to implement basic security measures, cryptography, and access controls.
- To prepare students for roles in network security, ethical hacking, and security policy implementation.
- To foster awareness of legal, ethical, and regulatory aspects of cybersecurity.

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

CO1	**Explain** core cybersecurity concepts, threats, and attack methodologies.
CO2	**Apply** basic cryptographic techniques and access control mechanisms.
C03	**Analyze** network security vulnerabilities and implement protective measures.
C04	**Evaluate** security policies, ethical hacking principles, and incident response strategies.

Unit	Content	Credit	Weightage
I	**Introduction to Cybersecurity & Threats** • Introduction to cybersecurity: Definition, importance, CIA triad (Confidentiality, Integrity, Availability) • Cyber threats: Malware, phishing, ransomware, DDoS, insider threats • Attack vectors: Social engineering, password attacks, SQL injection, XSS • Vulnerability assessment basics • Cybersecurity roles and certifications • **Applications**: Case studies of recent cyber attacks	1	25%
II	**Cryptography & Access Control** • Cryptography basics: Symmetric vs asymmetric encryption • Common algorithms: AES, DES, RSA • Hash functions: MD5, SHA-1, SHA-256 • Digital signatures and certificates • Access control models: DAC, MAC, RBAC	1	25%

	• Authentication methods: Passwords, biometrics, MFA • **Applications**: Secure messaging, password hashing, access control in OS		
III	**Network Security & Defense** • Network security fundamentals: Firewalls, IDS, IPS • VPNs and secure communication protocols (SSL/TLS) • Wireless security: WEP, WPA, WPA2, WPA3 • Basic packet analysis using Wireshark • Security policies: BYOD, remote work security • **Applications**: Configuring firewall rules, setting up VPN, analyzing network traffic	1	25%
IV	**Ethical Hacking & Incident Response** • Introduction to ethical hacking: Phases, tools, legal aspects • Footprinting, scanning, enumeration • Penetration testing basics • Incident response: Steps, roles, recovery • Cybersecurity laws and regulations: IT Act, GDPR, Cybercrime laws • **Applications**: Simulated penetration testing, incident response planning	1	25%

Textbooks:

- *Cybersecurity Fundamentals* – Rick Howard
- *Computer Security: Principles and Practice* – William Stallings & Lawrie Brown

Reference books:

- *The Web Application Hacker's Handbook* – Dafydd Stuttard & Marcus Pinto
- *Cryptography and Network Security* – Atul Kahate
- *Hacking: The Art of Exploitation* – Jon Erickson
- *Cybersecurity for Dummies* – Joseph Steinberg

Online Platforms:

- NPTEL
  - *Introduction to Cybersecurity* by IIT Madras
  - *Cryptography and Network Security* by IIT Kharagpur
- Coursera
  - *Introduction to Cybersecurity Tools & Cyber Attacks* by IBM
  - *Cybersecurity for Everyone* by University of Maryland

MK UNIVERSITY

PATAN, GUJARAT

ESTABLISHED BY THE GUJARAT GOVT. **RECOGNIZED BY UGC UNDER SECTION 2(f) OF UGC ACT,1956**

COURSE CODE: DCSE503

COURSE NAME: WEB DEVELOPMENT WITH PHP/ MY SQL

Course Objectives:

- To introduce fundamental concepts of server-side web development using PHP and MySQL.
- To develop skills in creating dynamic, database-driven web applications.
- To enable students to implement user authentication, session management, and form handling.
- To prepare students for building full-stack web applications with front-end and back-end integration.
- To foster understanding of web security, optimization, and deployment practices.

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

CO1	**Explain** PHP syntax, control structures, and MySQL database fundamentals.
CO2	**Develop** dynamic web pages using PHP forms, sessions, and cookies.
C03	**Integrate** PHP with MySQL to perform CRUD operations.
C04	**Design** secure and responsive web applications with authentication and validation.

Unit	Content	Credit	Weightage
I	**PHP Basics & Control Structures** • Introduction to PHP: Features, server setup (XAMPP/WAMP) • PHP syntax, variables, data types, operators • Control structures: if, if-else, switch, loops (for, while, do-while) • Functions: Built-in and user-defined • Arrays: Indexed, associative, multidimensional • **Applications**: Simple calculator, number operations, string manipulation	1	25%
II	**PHP Forms, Sessions & Cookies** • Form handling: GET vs POST, form validation • Superglobals: $_GET, $_POST, $_SERVER, $_SESSION, $_COOKIE • Session management: Starting, storing, destroying sessions • Cookies: Creation, modification, deletion • File handling: Uploading and processing files • **Applications**: User login system, feedback form, visitor counter	1	25%
III	**MySQL Database & PHP Integration** • Introduction to MySQL: Database creation, tables, data types • SQL commands: CREATE, INSERT, SELECT, UPDATE, DELETE • PHP MySQLi functions: mysqli_connect(), mysqli_query(), etc. • Performing CRUD operations using PHP and MySQL • Preventing SQL injection with prepared statements • **Applications**: Student record system, product catalog, blog system	1	25%
IV	**Advanced PHP & Web Application Security** • Object-Oriented PHP: Classes, objects, inheritance, interfaces • PHP includes: include(), require() • Error handling and debugging	1	25%

	• Web security: Input validation, XSS prevention, CSRF protection • Introduction to AJAX with PHP • **Applications**: Secure user registration, dynamic content loading, admin panel		

Textbooks:

- *PHP and MySQL Web Development* – Luke Welling & Laura Thomson
- *Beginning PHP and MySQL* – Jason Gilmore

Reference books:

- *PHP: The Complete Reference* – Steven Holzner
- *Learning PHP, MySQL & JavaScript* – Robin Nixon
- *Pro PHP and MySQL* – APress
- *Web Development with PHP and MySQL* – Matt Doyle

Online Platforms:

- NPTEL
  - *Web Development using PHP and MySQL* by IIT Kharagpur
  - *Server-Side Web Development* by IIT Madras
- Coursera
  - *Building Web Applications in PHP* by University of Michigan
  - *PHP with MySQL* by University of California, Davis

PRACTICAL LIST:

Module I: PHP Basics & Control Structures

1. Write a PHP program to display "Hello, World!".
2. Create a PHP program to calculate the sum, difference, product, and quotient of two numbers.
3. Write a PHP script to check whether a number is prime.
4. Develop a program to print Fibonacci series up to n terms.
5. Write a program to find the largest number in an array.
6. Create a function to reverse a string.

Module II: PHP Forms, Sessions & Cookies

7. Design a registration form with validation (name, email, password).
8. Create a login system using sessions.
9. Build a page visit counter using cookies.
10. Develop a file upload form and display the uploaded file details.
11. Create a simple CAPTCHA verification in a form.
12. Build a user profile page that displays session-stored data.

Module III: MySQL Database & PHP Integration

13. Create a database school and table students with fields (id, name, marks).
14. Write a PHP program to insert, update, and delete records in the students table.
15. Develop a search form to find students by name or ID.
16. Create a pagination system to display 5 records per page.
17. Build a product catalog with categories using MySQL joins.
18. Design a simple blog system with posts and comments.

Module IV: Advanced PHP & Web Application Security

19. Create a class User with methods for registration and login.
20. Implement password hashing using password_hash() and password_verify().
21. Develop an admin panel with session-based access control.
22. Build an AJAX-based live search feature.
23. Create a secure contact form with CSRF token validation.

24. Implement error logging in a file for debugging.

COURSE CODE: DCSE504

COURSE NAME: MOBILE APP DEVELOPMENT

Course Objectives:

- To introduce fundamental concepts of mobile app development and platform ecosystems.
- To develop skills in designing, building, and testing mobile applications for Android and cross-platform environments.
- To enable students to implement UI/UX principles, data storage, and API integration in mobile apps.
- To prepare students for deploying, publishing, and maintaining mobile applications.
- To foster awareness of mobile security, performance optimization, and emerging trends.

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

CO1	**Explain** mobile app development platforms, architectures, and lifecycle.
CO2	**Design** user interfaces and implement navigation in mobile apps.
C03	**Develop** functional mobile apps with data storage, APIs, and device features.
C04	**Deploy** and test mobile applications on real/virtual devices and app stores.

Unit	Content	Credit	Weightage
I	**Introduction to Mobile App Development** • Overview of mobile platforms: Android, iOS, cross-platform • Mobile app architectures: Native, Hybrid, Web apps • Setting up development environment: Android Studio, Flutter/React Native basics • App lifecycle and components (Activity, Fragment in Android) • Introduction to Kotlin/Java for Android or Dart for Flutter • **Applications**: Hello World app, basic calculator app	1	25%
II	**UI/UX Design & Core Components** • UI design principles: Material Design (Android), Human Interface Guidelines (iOS) • Layouts: Linear, Relative, Constraint (Android) or Column/Row (Flutter) • UI components: Buttons, TextViews, EditText, ListView, RecyclerView • Navigation: Intents, Navigation Component, Bottom Navigation • Responsive design for multiple screen sizes • **Applications**: Login screen, profile page, news feed UI	1	25%
III	**Data Handling & APIs** •Data persistence: Shared Preferences, SQLite, Room	1	25%

	Database (Android) • File storage: Internal and external storage • Networking: HTTP requests, REST API integration (Retrofit/Volley) • JSON parsing and handling API responses • Using device sensors: Camera, GPS, accelerometer • **Applications**: Weather app, todo list with local storage, news app with API		
IV	**Advanced Features & Deployment** • Notifications: Local and push notifications (Firebase Cloud Messaging) • Multimedia: Playing audio/video, image handling • App security: Data encryption, secure storage, permissions • Testing: Unit testing, UI testing (Espresso/JUnit) • App deployment: Generating APK, app signing, Google Play Store submission • **Applications**: Music player, location-based reminder, deployed app on Play Store	1	25%

Textbooks:

- *Android Programming: The Big Nerd Ranch Guide* – Bill Phillips & Chris Stewart
- *Flutter Complete Reference* – Alberto Miola

Reference books:

- *Head First Android Development* – Dawn Griffiths & David Griffiths
- *Android App Development for Dummies* – Michael Burton
- *Learning React Native* – Bonnie Eisenman
- *Mobile App Development with Ionic* – Chris Griffith

Online Platforms:

- NPTEL
  - *Mobile Application Development* by IIT Kharagpur
  - *Android App Development* by IIT Madras
- Coursera
  - *Android App Development* by Vanderbilt University
  - *Flutter Development Bootcamp* by London App Brewery

MK UNIVERSITY

PATAN, GUJARAT

ESTABLISHED BY THE GUJARAT GOVT. RECOGNIZED BY UGC UNDER SECTION 2(f) OF UGC ACT,1956

SEMESTER-VI

COURSE CODE: DCSE601

COURSE NAME: INTERNET OF THINGS (IOT)

Course Objectives:

- To introduce fundamental concepts, architectures, and applications of IoT.
- To develop skills in designing, prototyping, and implementing IoT systems using sensors, microcontrollers, and communication modules.
- To enable students to integrate IoT devices with cloud platforms and data analytics tools.
- To prepare students for roles in smart systems, industrial automation, and embedded IoT solutions.
- To foster awareness of IoT security, privacy, and ethical considerations.

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

CO1	**Explain** IoT architecture, components, protocols, and application domains.
CO2	**Design** and implement basic IoT systems using microcontrollers and sensors.
C03	**Integrate** IoT devices with cloud platforms for data storage and visualization.
C04	**Develop** secure and scalable IoT solutions with real-world applications.

Unit	Content	Credit	Weightage
I	**Introduction to IoT & Architecture** • Introduction to IoT: Definition, evolution, characteristics • IoT architecture: 3-layer and 5-layer models • IoT components: Sensors, actuators, microcontrollers, gateways, cloud • IoT communication models: Request-Response, Publish-Subscribe, Push-Pull • IoT application domains: Smart home, healthcare, agriculture, industrial automation • **Applications**: Case studies of IoT in daily life and industry	1	25%
II	**IoT Hardware & Communication Protocols** • Microcontrollers: Arduino, ESP8266/ESP32, Raspberry Pi • Sensors and actuators: Temperature, humidity, motion, IR, relays, motors • Communication protocols: Wi-Fi, Bluetooth, Zigbee, LoRa, MQTT, CoAP • Embedded programming basics using Arduino IDE • IoT prototyping platforms: Node-RED, Blynk • **Applications**: Weather monitoring station, smart lighting system	1	25%
III	**IoT Data Handling & Cloud Integration** • IoT data acquisition and preprocessing	1	25%

	• Cloud platforms for IoT: AWS IoT, Google Cloud IoT, Thing Speak, Adafruit IO • Data storage and visualization on cloud dashboards • REST APIs and MQTT brokers for IoT communication • Edge computing basics • **Applications**: Cloud-based sensor monitoring, real-time data logging		
IV	**IoT Applications & Security** • Smart systems: Home automation, smart agriculture, health monitoring • Industrial IoT (IIoT): Predictive maintenance, asset tracking • IoT security challenges: Device security, network security, data privacy • Security protocols: TLS/SSL, encryption, authentication • Ethical and legal considerations in IoT • **Applications**: Secure smart door lock, IoT-based irrigation system	1	25%

Textbooks:

- *Internet of Things: A Hands-On Approach* – Arshdeep Bahga & Vijay Madisetti
- *Building the Internet of Things* – Maciej Kranz

Reference books:

- *IoT Fundamentals: Networking Technologies, Protocols, and Use Cases* – David Hanes et al.
- *Getting Started with the Internet of Things* – Cuno Pfister
- *Practical Internet of Things Security* – Brian Russell & Drew Van Duren
- *IoT and Edge Computing for Architects* – Perry Lea

Online Platforms:

- NPTEL
  - *Introduction to Internet of Things* by Prof. Sudip Misra (IIT Kharagpur)
  - *IoT and Cloud Computing* by IIT Bombay
- Coursera
  - *An Introduction to Programming the Internet of Things (IoT)* by University of California, Irvine
  - *IoT (Internet of Things) Specialization* by University of Illinois

PRACTICAL LIST:

Module I: Introduction & Architecture

1. Research and present a case study on an IoT application (smart home/healthcare).
2. Draw and explain the 5-layer IoT architecture with real-world components.
3. Identify IoT sensors and actuators used in daily appliances.
4. Prepare a report on IoT communication models with examples.

Module II: IoT Hardware & Communication

5. Interface a temperature sensor (DHT11) with Arduino and display readings on the serial monitor.
6. Control an LED using a smartphone via Bluetooth (HC-05 module).
7. Send sensor data to a mobile app using Blynk platform.
8. Implement MQTT protocol to publish and subscribe sensor data between two devices.
9. Build a motion-activated security light using PIR sensor and relay.

Module III: Cloud Integration & Data Handling

10. Send temperature and humidity data to ThingSpeak cloud and visualize it.
11. Create a cloud dashboard (Adafruit IO) to monitor sensor data remotely.
12. Use Node-RED to create a flow for processing and visualizing IoT data.
13. Store IoT sensor data in a cloud database (Firebase/Google Sheets).
14. Implement a simple alert system (email/SMS) when sensor thresholds are crossed.
    Module IV: Applications & Security
15. Develop a smart home system with multiple sensors and remote control.
16. Build an IoT-based plant watering system with soil moisture sensor.
17. Implement basic encryption for data transmitted from an IoT device.
18. Design an access control system using RFID/NFC and IoT.
19. Mini-Project: Create an end-to-end IoT solution (e.g., smart weather station, health monitor, smart parking system).

COURSE CODE: DCSE602

COURSE NAME: SOFTWARE TESTING AND DEVOPS

Course Objectives:

- To introduce fundamental concepts, types, and methodologies of software testing.
- To develop skills in designing, executing, and automating test cases.
- To enable students to understand and implement DevOps practices for continuous integration and delivery.
- To prepare students for roles in quality assurance, test automation, and DevOps engineering.
- To foster awareness of testing tools, CI/CD pipelines, and collaboration between development and operations.

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

CO1	**Explain** software testing fundamentals, levels, and techniques.
CO2	**Design** and execute test cases using manual and automated testing tools.
C03	**Apply** DevOps principles, version control, and CI/CD pipelines in software projects.
C04	**Implement** continuous testing, deployment automation, and monitoring practices.

Unit	Content	Credit	Weightage
I	**Software Testing Fundamentals** • Introduction to software testing: Importance, objectives, principles • Software Development Life Cycle (SDLC) and testing phases • Levels of testing: Unit, integration, system, acceptance • Types of testing: Black-box, white-box, regression, performance, security • Test planning, test cases, and test scenarios • Bug lifecycle and defect tracking • **Applications**: Test case design for a login module, bug reporting	1	25%
II	**Test Automation & Tools**	1	25%

	• Introduction to test automation: Benefits and challenges • Automation tools overview: Selenium, JUnit, TestNG, Postman • Web automation using Selenium WebDriver • API testing using Postman/REST Assured • Performance testing basics with JMeter • Mobile testing basics (Appium) • **Applications**: Automated login test, API endpoint validation		
III	**DevOps Principles & CI/CD** • Introduction to DevOps: Culture, practices, benefits • Version control with Git: Basics, branching, merging, pull requests • Continuous Integration (CI): Concepts, tools (Jenkins, GitHub Actions) • Continuous Delivery vs Continuous Deployment • Infrastructure as Code (IaC) basics: Docker, Ansible (overview) • Containerization: Docker fundamentals • **Applications**: Setting up a Git repository, CI pipeline for a simple app	1	25%
IV	**Continuous Testing & Deployment** • Continuous testing in DevOps pipeline • Automated testing in CI/CD using Jenkins/GitHub Actions • Deployment strategies: Blue-green, canary, rolling updates • Monitoring and logging basics: Introduction to ELK stack, Grafana • Cloud platforms for DevOps: AWS/Azure basics • Security in DevOps (DevSecOps) overview • **Applications**: Deploying a web app with automated testing, monitoring setup	1	25%

Textbooks:

- *Software Testing: Principles and Practices* – Srinivasan Desikan & Gopalaswamy Ramesh
- *The DevOps Handbook* – Gene Kim, Jez Humble, Patrick Debois & John Willis

Reference books:

- *Foundations of Software Testing* – Dorothy Graham, Erik van Veenendaal, Isabel Evans, Rex Black
- *Continuous Delivery* – Jez Humble & David Farley
- *Automate the Boring Stuff with Python* – Al Sweigart (for test automation scripting)
- *Learning DevOps* – Mikael Krief

Online Platforms:

- NPTEL
  - *Software Testing* by IIT Kharagpur
  - *DevOps Practices and Principles* by IIT Madras
- Coursera
  - *Introduction to Software Testing* by University of Minnesota

- *DevOps and Software Engineering Professional Certificate* by IBM

PRACTICAL LIST:

Module I: Software Testing Fundamentals

1. Write test cases for a login functionality (positive and negative scenarios).
2. Perform boundary value analysis for a password field (length 8–16 characters).
3. Create a defect report using a bug tracking template (e.g., JIRA-like format).
4. Design test scenarios for an e-commerce "Add to Cart" feature.
5. Perform compatibility testing of a website on different browsers.

Module II: Test Automation & Tools

6. Automate login test using Selenium WebDriver (Java/Python).
7. Write and execute a JUnit/TestNG test suite for a simple calculator class.
8. Test REST APIs using Postman (GET, POST, PUT, DELETE).
9. Create a performance test script using JMeter for a web page.
10. Automate a mobile app test case using Appium (emulator/simulator).

Module III: DevOps Principles & CI/CD

11. Set up a Git repository and perform basic commands (clone, commit, push, branch).
12. Create a Jenkins pipeline to build a Java/Python application.
13. Set up a CI pipeline using GitHub Actions for automatic code testing.
14. Containerize a simple web app using Docker and run it locally.
15. Write a basic Ansible playbook to install and start a web server.

Module IV: Continuous Testing & Deployment

16. Integrate Selenium tests into a Jenkins pipeline.
17. Deploy a Dockerized application on a cloud platform (AWS EC2/Azure VM).
18. Implement blue-green deployment simulation using Docker Compose.
19. Set up basic monitoring using Prometheus and Grafana (demo).
20. Mini-Project: Build a full CI/CD pipeline for a sample application with automated testing, containerization, and deployment.

MK UNIVERSITY

PATAN, GUJARAT

ESTABLISHED BY THE GUJARAT GOVT. RECOGNIZED BY UGC UNDER SECTION 2(f) OF UGC ACT,1956

COURSE CODE: DCSE603

COURSE NAME: ARTIFICIAL INTELLIGENCE BASICS

Course Objectives:

- To introduce fundamental concepts, history, and scope of Artificial Intelligence.
- To develop an understanding of search algorithms, knowledge representation, and reasoning techniques.
- To enable students to implement basic AI techniques using Python.
- To prepare students for advanced topics in machine learning, NLP, and intelligent systems.
- To foster awareness of ethical considerations and real-world applications of AI.

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

CO1	**Explain** AI concepts, history, types, and applications.
CO2	**Apply** search algorithms and knowledge representation techniques.
C03	**Implement** basic machine learning models and natural language processing tasks.
C04	**Analyze** ethical implications and future trends in AI.

Unit	Content	Credit	Weightage
I	**Introduction to AI & Problem Solving** • Introduction to AI: History, definitions, goals, and applications • Types of AI: Narrow AI vs General AI, Strong AI vs Weak AI • Intelligent agents: PEAS representation, types of agents • Problem-solving agents: State space, goal state, actions • AI programming tools: Introduction to Python libraries (NumPy, pandas) • **Applications**: AI in gaming, chatbots, recommendation systems	1	25%
II	**Search Algorithms & Knowledge Representation** • Uninformed search: BFS, DFS, uniform cost search • Informed search: Greedy best-first, A* algorithm • Constraint satisfaction problems • Knowledge representation: Propositional logic, first-order logic, semantic networks • Inference in AI: Forward and backward chaining • **Applications**: Pathfinding in maps, puzzle solving (8-puzzle), rule-based systems	1	25%
III	**Machine Learning Basics** • Introduction to machine learning: Supervised, unsupervised, reinforcement learning • Regression: Linear regression basics • Classification: Decision trees, k-NN • Clustering: k-Means algorithm	1	25%

	• Model evaluation: Accuracy, precision, recall, confusion matrix • **Applications**: Email spam detection, customer segmentation, house price prediction		
IV	**NLP, Ethics & AI Trends** • Natural Language Processing basics: Tokenization, stemming, lemmatization • Sentiment analysis using simple classifiers • AI ethics: Bias, fairness, transparency, accountability • Future trends: AI in healthcare, autonomous vehicles, AI and jobs • Introduction to neural networks (overview) • **Applications**: Sentiment analysis of reviews, ethical case studies, AI in robotics	1	25%

Textbooks:

- *Artificial Intelligence: A Modern Approach* – Stuart Russell & Peter Norvig
- *Artificial Intelligence Basics* – Tom Taulli

Reference books:

- *Artificial Intelligence with Python* – Prateek Joshi
- *AI for Everyone* – Andrew Ng (Online resource)
- *Introduction to Artificial Intelligence* – E. Rich & K. Knight
- *Machine Learning for Absolute Beginners* – Oliver Theobald

Online Platforms:

- NPTEL
  - *Introduction to Artificial Intelligence* by IIT Madras
  - *Artificial Intelligence: Search Methods for Problem Solving* by IIT Delhi
- Coursera
  - *AI For Everyone* by Andrew Ng
  - *Introduction to Artificial Intelligence with Python* by Harvard (edX)

PRACTICAL LIST:

Module I: Introduction to AI & Python Basics

1. Install Python and necessary libraries (NumPy, pandas, matplotlib).
2. Write a Python program to implement a simple rule-based chatbot (if-else based).
3. Create a Tic-Tac-Toe game using a basic rule-based AI.
4. Write a program to calculate the factorial of a number using recursion.

Module II: Search Algorithms

5. Implement Breadth-First Search (BFS) to find the shortest path in a graph.
6. Implement Depth-First Search (DFS) for tree traversal.
7. Write a program to solve the 8-puzzle problem using A* algorithm (heuristic: Manhattan distance).
8. Implement a simple expert system for medical diagnosis (rule-based).

Module III: Machine Learning Basics

9. Implement linear regression using Python to predict house prices.
10. Build a decision tree classifier for the Iris dataset using scikit-learn.
11. Implement k-Means clustering for customer segmentation.
12. Create a k-NN classifier for email spam detection (using sample dataset).
13. Evaluate a classification model using confusion matrix and accuracy score.

Module IV: NLP & Ethics

14. Perform tokenization and stemming on a given text using NLTK.
15. Build a simple sentiment analyzer for movie reviews.
16. Write a case study report on ethical issues in AI (e.g., bias in hiring algorithms).