NATIONAL INSTITUTE OF ELECTRONICS AND INFORMATION TECHNOLOGY

DEEMED TO BE UNIVERSITY UNDER DISTINCT CATEGORY

(AN AUTONOMOUS INSTITUTION UNDER MINISTRY OF ELECTRONICS & IT, GOVT. OF INDIA)

Curriculum and Syllabi for

B.Tech - Computer Science and Engineering (Internet of Things and Cyber Security including Blockchain Technology)

()

रा.इ.सू.प्रौ.सं

NIELIT

Deemed to be University

Main Campus at Ropar and Constituent Units at Aizawl, Agartala, Aurangabad, Calicut, Gorakhpur, Imphal, Itanagar, Kekri, Kohima, Patna and Srinagar

Vision

To be a centre of excellence in Computer Science & Engineering with a specialized focus on IoT, Cybersecurity, and Blockchain Technology, empowering students with advanced technological skills, ethical values, and innovative capabilities to address global challenges in a secure and connected digital world.

Mission

1. To impart quality education in Computer Science & Engineering with a strong foundation in IoT, Cybersecurity, and Blockchain Technology, fostering analytical thinking, problem-solving skills, and technical proficiency. 2. To promote research and innovation by encouraging students and faculty to work on real-world projects in emerging areas of secure and connected technologies. 3. To develop industry-ready professionals through practical training, internships, and collaborations with academic and industrial partners. 4. To instill ethical values and a sense of responsibility in students for designing secure, sustainable, and inclusive digital solutions for societal and global needs. 5. To encourage lifelong learning and entrepreneurship by nurturing creativity, leadership, and adaptability to thrive in a dynamic technological landscape.

Program Education Objectives

PEO1: Core Competency: Graduates will have a strong foundation in computer science and engineering with specialized knowledge in IoT, Cybersecurity, and Blockchain to solve real-world problems effectively. PEO2: Professional Growth: Graduates will engage in lifelong learning through advanced studies, certifications, or professional development to adapt to evolving technologies and industry demands. PEO3: Innovation and Research: Graduates will contribute to technological advancement through innovation, research, and development in secure and intelligent systems. PEO4: Ethics and Responsibility: Graduates will practice ethical and responsible computing, ensuring data privacy, security, and societal well-being in their professional endeavors. PEO5: Leadership and Teamwork: Graduates will demonstrate leadership, communication, and collaborative skills, enabling them to work effectively in multidisciplinary teams and global environments.

Program Outcomes

PO1: Core Knowledge Application: Apply fundamental knowledge of computer science, IoT, cybersecurity, and blockchain to solve real-world engineering problems. PO2: Secure System Design: Design and develop secure, efficient, and scalable computing systems with a focus on IoT and blockchain technologies. PO3: Problem Solving and Innovation: Analyze complex problems and develop innovative solutions using modern tools, programming platforms, and emerging technologies. PO4: Professional Ethics and Responsibility: Demonstrate ethical behavior, cybersecurity awareness, and commitment to societal and environmental well-being. PO5: Effective Communication and Teamwork: Communicate effectively and work collaboratively in multidisciplinary teams to deliver quality engineering solutions.

Program Specific Outcomes

PO1: Core Knowledge Application: Apply fundamental knowledge of computer science, IoT, cybersecurity, and blockchain to solve real-world engineering problems. PO2: Secure System Design: Design and develop secure, efficient, and scalable computing systems with a focus on IoT and blockchain technologies. PO3: Problem Solving and Innovation: Analyze complex problems and develop innovative solutions using modern tools, programming platforms, and emerging technologies. PO4: Professional Ethics and Responsibility: Demonstrate ethical behavior, cybersecurity awareness, and commitment to societal and environmental well-being. PO5: Effective Communication and Teamwork: Communicate effectively and work collaboratively in multidisciplinary teams to deliver quality engineering solutions.

CATEGORY WISE CREDIT DISTRIBUTION

Category	Category Code	Credits
Audit Courses (without grade or credit)	AU	0
Value Added Courses	VA	6
Professional Elective Courses	PE	20
Open Elective Courses	OE	12
Ability Enhancement Courses	AE	8
Skill / Employment Enhancement Courses (Project/Internship/Seminar/Dissertation/Research)	EE	31
Program Core Courses	PC	88
Total Credits (Common track)		**165**

EVALUATION AND ASSESSMENT

1. Performance of a student in a semester shall be evaluated through continuous Class assessment, Tutorial/Lab assessment, Mid-Semester Examination (MSE) and End-Semester Examination (ESE). Both the MSE and ESE shall be the University examination and will be conducted as notified by the Controller of Examinations (CoE) of the University.

2. The continuous assessment shall be based on assignments, tutorials, paper presentation / quizzes/ viva-voce / flipped classes, lab work / projects / fieldwork and attendance, etc.

3. The MSE/ESE shall be comprising of written papers.

4. The overall assessment of the students will be done as per the following scheme:

S. No.	Assessment Type	Marks Weightage
1.	Mid Semester Examination	20
2.	End Semester Examination	40
3.	Continuous Assessment - Practical /Lab/ Tutorial	25
4.	Continuous Assessment - Theory	15
Total		**100**

For more details, please refer the applicable ordinance for this programme and Assessment SOPs.

CURRICULUM

Semester 1						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
PC-BTSIOT-101	DASH250038	Engineering Mathematics I	3	1	0	4
PC-BTSIOT-102	DCSE250039	Programming Fundamentals	2	0	4	4
PC-BTSIOT-103	DOEE250033	Basic Electrical & Electronics Engineering	2	1	2	4
EE-BTSIOT-104	DASH250042	Engineering Physics	3	0	2	4
AE-BTSIOT-105	DASH250016	Communication Skills	1	0	2	2
OE-BTSIOT-106	Elective	Open Elective				4

Semester 2						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
PC-BTSIOT-201	DASH250039	Engineering Mathematics II	3	1	0	4
PE-BTSIOT-202	Elective	Program Elective				4
PC-BTSIOT-203	DOEE250059	Digital Electronics	3	0	2	4
EE-BTSIOT-204	DCSA250016	Computer Aided Graphics Design	1	1	4	4
VA-BTSIOT-205	DASH250030	Economics for Engineers	1	1	0	2
OE-BTSIOT-206	Elective	Open Elective				4

Semester 3						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
PC-BTSIOT-301	DCSE250113	Programming in Python	2	0	4	4
PC-BTSIOT-302	DCSE250055	Data Structures	3	0	2	4
PC-BTSIOT-303	DOEE250044	Computer Organization and Architecture	3	0	2	4
PC-BTSIOT-304	DOEE250022	Assembly Language and Microprocessors	3	0	2	4
AE-BTSIOT-305	DASH250104	Technical Presentation and Report Writing	0	0	4	2
PE-BTSIOT-306	Elective	Program Elective				4
AU-BTSIOT-307	Elective	Audit Elective				0

Semester 4						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
PC-BTSIOT-401	DCSE250060	Database Management Systems	2	0	4	4
PC-BTSIOT-402	DCSA250056	Numerical Methods	3	1	0	4
PC-BTSIOT-403	DCSE250102	Operating Systems	3	0	2	4
VA-BTSIOT-404	DASH250052	Environmental Science	2	0	0	2
OE-BTSIOT-405	Elective	Open Elective				4
EE-BTSIOT-406	EE	Research and Survey Report				4

Semester 5						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
PC-BTSIOT-501	DCSE250010	Design and Analysis of Algorithms	3	0	2	4
PC-BTSIOT-502	DOAI250030	Introduction to Artificial Intelligence	3	1	0	4
PC-BTSIOT-503	DCSE250035	Computer Networks	3	0	2	4
AE-BTSIOT-504	DASH250026	Design Thinking	1	0	2	2
PE-BTSIOT-505	Elective	Program Elective				4
EE-BTSIOT-506	EE	Internship				4

Semester 6						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
PC-BTSIOT-601	DCSE250107	Principles of Compiler Design	3	0	2	4
PC-BTSIOT-602	DCSE250002	Blockchain and Cryptocurrency	3	1	0	4
PC-BTSIOT-603	DCSE250043	Cryptography and Network Security	3	0	2	4
AE-BTSIOT-604	DASH250056	Financial Accounting & Management	2	0	0	2
VA-BTSIOT-605	DASH250091	Professional Values and Ethics	2	0	0	2
PE-BTSIOT-606	Elective	Program Elective				4

Semester 7						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
PC-BTSIOT-701	DCSE250004	Advanced Data Structures and Algorithms	3	0	2	4
PC-BTSIOT-702	DCSA250071	Software Engineering	3	1	0	4
PC-BTSIOT-703	DCSE250073	Ethical Hacking	2	1	2	4
PC-BTSIOT-704	DOEE250131	Industrial Robotics	2	0	4	4
PE-BTSIOT-705	Elective	Program Elective				4
EE-BTSIOT-706	EE	Mini Project				5

Semester 8						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
EE-BTSIOT-801	EE	Project and Thesis				10

Elective Courses

Elective Courses for PE Category						
Course Code	**Course Title**	**L**	**T**	**P**	**Cr**	**For Sem./Code**
DCSE250007	Cyber Law and IPR	3	0	2	4	
DCSE250013	Intrusion Detection System (IDS)	3	0	2	4	
DCSE250015	Cyber Attacks	3	0	2	4	
DCSE250020	Cloud Computing	3	0	2	4	
DCSE250021	Vulnerability Analysis and Penetration Testing	3	0	2	4	
DOAI250019	Data Mining and Warehousing	3	0	2	4	
DOAI250041	Natural Language Processing	3	0	2	4	
DOEE250053	Data Acquisition Techniques	3	0	2	4	
DOEE250094	Embedded Systems	3	0	2	4	
DOEE250121	Internet of Things	3	0	2	4	
DOEE250161	Principles of Industrial IoT	3	0	2	4	

Elective Courses for OE Category
Any Scheduled Course as per the Course Schedule of the respective Constituent Unit, and meeting the credit requirements can be chosen, subject to availability of seats. Students having different stream background from the admitted program stream shall have to mandatorily utilize Open Electives to bridge the knowledge gap, by choosing from the list of bridge courses decided and approved by Dean (Academics) of the respective NDU Constituent Unit.

Elective Courses for AU (Audit) Category					
Course Code	**Course Title**	**L**	**T**	**P**	**Cr**
DASH240027	Disaster Management	2	0	0	0
DASH240047	English for Research Paper Writing	2	0	0	0
DASH240085	Pedagogy Studies	2	0	0	0
DASH240086	Personality Development through Life Enlightenment Skills	2	0	0	0
DASH240097	Sanskrit for Technical Knowledge	2	0	0	0
DASH240103	Stress Management by Yoga	2	0	0	0
DASH240104	Technical Presentation and Report Writing	0	0	2	0
DASH240106	Value Education	2	0	0	0
DASH250023	Constitution of India	2	1	0	0
DASH250027	Disaster Management	3	0	0	0
DASH250034	Energy and Environmental Engineering	3	0	0	0
DASH250047	English for Research Paper Writing	2	0	0	0
DASH250054	Essence of Indian Knowledge and Tradition	2	0	0	0
DASH250085	Pedagogy Studies	2	0	0	0
DASH250086	Personality Development	2	0	0	0
DASH250097	Sanskrit for Technical Knowledge	3	0	0	0
DASH250103	Stress Management by Yoga	3	0	0	0

DASH250106	Value Education	2	0	0	0
DASH250117	Innovative Thinking and Entrepreneurship Skills	1	0	0	0
DASH250118	Intellectual Property Rights	2	0	0	0
DCSA240080	Training on Computer Networking	0	0	2	0
DCSA240304	Lab Based on Statistical Package	0	0	2	0
Any other Scheduled Course as per the Course Schedule of the respective Constituent Unit can also be chosen, subject to availability of seats. However, there shall be no assessment and grading for the course chosen as Audit Course.					

The choice of all type of Electives available shall be as per the Course Schedule of the respective Constituent Unit.

Guidelines for Massive Open Online Courses (MOOC) / approved Online Courses

1. Relevant Swayam, MOOC, NPTEL, NIELIT NSQF and any other online courses approved by UGC/AICTE shall also be allowed as Program Electives(PE) & Open Electives(OE).

2. To choose an approved online course as per above as Program Elective, the student shall identify an approved Online Course and submit a request to their department or designated authority in the NDU.

3. The Dean (Academics) or Chairman, Board of Studies, shall approve the online credit course for credit transfer on the recommendation of the Head of the Department, keeping in view academic requirements of the programme and alignment of Course Outcomes with the Programme Objectives.

4. On receipt of approval, the student may enroll and complete the course.

5. A student can choose any approved online course as Open Elective, subject to minimum credit requirements.

6. The student has to submit certificate issued by the institution offering the MOOCs course, along with the number of credits and grades, to get credits transferred into his/her marks certificate issued by NDU. The transfer of credits shall be as per rules and procedure stipulated by AICTE / UGC.

Detailed Syllabus

Course Code	Course Name	L	T	P	Cr
DASH250038	Engineering Mathematics I	3	1	0	4

Course Outcomes:

CO1 : Apply matrix operations to solve systems of linear equations, find eigenvalues/eigenvectors, and use the Cayley-Hamilton theorem.
CO2 : Analyze functions of a single variable for limit, continuity and differentiability, Mean value theorems, definite integrals in engineering applications such as areas and volumes.
CO3 : Solve first and second-order ordinary differential equations, including linear, exact, and simultaneous forms, and interpret their geometric and analytical solutions.
CO4 : Differentiate multivariable functions, use Euler's theorem, Jacobians, and Taylor series, and solve problems involving maxima and minima of functions of two variables.
CO5 : Evaluate double and triple integrals in different coordinate systems and apply them to compute areas and volumes, including changing the order of integration when required.

Module 1:

Matrices (9 hrs)
Rank of a matrix, Consistency of a system of linear equations, Characteristic equation of a matrix, Eigen values and Eigen vectors, Cayley-Hamilton theorem (excluding proof), Verification, Application (Finding Inverse of a matrix), Diagonalization of a matrix by orthogonal transformation.

Module 2:

Differential Calculus and Applications (9 hrs)
Limit, Continuity and differentiability of a function of single variable, Rolle's theorem, Lagrange's theorem, and Cauchy's mean value theorem (Statements and simple applications), Applications of definite integrals to evaluate surface areas and volumes of revolutions of curves in Cartesian coordinates.

Module 3:

Differential Equations (7 hrs)
Ordinary Differential Equations (ODE): First order differential equations, Exact differential equations and integrating factor, Linear differential equations of first and second order, Simultaneous linear differential equations by elimination method.

Module 4:

Multivariable Calculus (Partial Differentiation and Applications) (11 hrs)
Partial differentiation, Partial derivatives of first, second and third order, Partial differentiation of implicit functions, Euler's theorem, Total derivative, Jacobian and its Properties, Applications: Maxima and minima of functions of two variables.

Module 5:

Multivariate Calculus (Multiple Integrals) (9 hrs)
Double integral (Cartesian, polar form), Application of double integral: Area by double integration, Change of Order of Integration. Triple Integral in Cartesian coordinate only, Application of triple integral: Volume by triple integration.

Labs / Practicals:

N/A

References:

1. Grewal, B. S. Higher Engineering Mathematics (43rd ed.). Khanna Publishers.
2. Bali, N. P., & Goyal, M. A Text Book of Engineering Mathematics. Laxmi Publications (P) Ltd.
3. Kreyszig, E. Advanced Engineering Mathematics (9th ed.). John Wiley & Sons.
4. Ramana, B. V. Higher Engineering Mathematics. Tata McGraw-Hill.
5. Jeffery, A. Advanced Engineering Mathematics. Academic Press.

Course Code	Course Name	L	T	P	Cr
DCSE250039	Programming Fundamentals	2	0	4	4

Course Outcomes:

CO1: Remember the syntax and basic operators in C language and its descendants.

CO2: Understand how to represent algorithms using flowcharts and pseudo-codes.

CO3: Apply well-known fundamental programming techniques for problem solving.

CO4: Analyze programming bugs through testing, stack traces, and core dumps.

CO5: Evaluate the outcomes of programs manually with dry runs and trace tables.

CO6: Create bug-free and efficient programming solutions for real-world problems.

Module 1:

Fundamentals of programming, program vs algorithm, implementing algorithms.
Language-independent algorithm representation using flowcharts and pseudo-codes.
Functionality vs implementation, usage of right abstractions, code documentation.
Runtime error vs logical bug, black box vs white box testing, debugging techniques.

Module 2:

Components of a program: constants, identifiers, operators, keywords, comments.
Editor vs compiler, compilation vs interpretation, basic I/O using printf and scanf.
Variable declaration vs definition, initialization, assignment, type promotion rules.
Arithmetic, relational, logical, bitwise operators, arity, precedence and associativity.

Module 3:

Sequential control flow, expression, statement, block, identifier scope, object lifetime.
Conditional operator, selection statements: if, else, switch; nested conditions.
Iteration blocks: for, while, do-while; loop invariants, nested loops, Duff's device.
Labels, jump statements: break, continue, goto; structured programming.

Module 4:

Function declaration vs definition, parameter vs argument, call by value, void type.
Local vs external variables, static variables, recursion, tail call, inline definition.
Arrays, random access, string, array vs pointer, array of array, variable-length array.
Enums, structures, bit-fields, padding, unions, compound literals, incomplete types.

Module 5:

Stack and heap layout, dynamic memory allocation, dangling pointer, memory leak.
Headers, preprocessing directives, multiple source files, internal vs external linkage.
Command-line arguments, C standard library, file I/O, buffering, signal handling.
Portability issues, undefined behavior, buffer overflow and other vulnerabilities.

Labs / Practicals:

All programming solutions should conform to ISO C99 standard (or later revisions).

01. Test the precedence and associativity of arithmetic operators in C: *, /, %, +, -.

02. Test the outcomes of relational and equality operators in C.

03. Test the short-circuit evaluation of && and || using function calls as operands.

04. Test the bitwise operators in C with unsigned and signed types: left shift, right shift, &, ^, |.

05. Convert from metric prefix (KB/MB/GB/TB) to binary prefix (KiB/MiB/GiB/TiB).

06. Use if-else-if ladder to print the grade as per the score obtained in a subject.

07. Find the mean, median, mode, variance, and standard deviation for a list of scores.

08. Print values using additive/subtractive rules of roman numerals: I, V, X, L, C, D, M.

09. Use nested loops to generate asymmetric and symmetric patterns of arbitrary size.

10. Check whether a given string is a palindrome (or not) without reversing the string.

11. Write a function to check whether an array of integers is already sorted (or not).

12. Write a function for primality testing, and use it to find first n pairs of twin primes.

13. Write a function to generate the Collatz sequence (up to 1) for a positive integer.

14. Generate the terms of Fibonacci sequence using iterative and recursive approaches.

15. Print optimal sequence of moves for a given instance of “Towers of Hanoi” puzzle.

16. Find all combinations to make a given payment from a limited set of coin types.

17. Find vowel count in a string using a lookup table for case-insensitive vowel check.

18. Write a function to check whether a string is a valid email ID (or not).

19. Use case-insensitive and dot-insensitive matching to check if two email IDs are same.

20. Display a countdown timer in HH:MM:SS format using carriage return ("\r").

21. Get current time and find clockwise angles of hour hand, minute hand, second hand.

22. Find the day of week for any date in Gregorian calendar using doomsday algorithm.

23. Populate an array with student details and write the attributes in a new CSV file.

24. Sort an array of student details as per roll number using qsort from stdlib.h .

25. Use bsearch from stdlib.h to fetch student details from an array sorted by roll.

References:

1. "Computer Science I" by Chris Bourke
https://cse.unl.edu/~cbourke/ComputerScienceOne.pdf

2. "How to Think Like a Computer Scientist (C Version)" by Thomas Scheffler and Allen Benjamin Downey
https://github.com/tscheffl/ThinkC/raw/master/PDF/Think-C.pdf

3. "Foundations of Computer Science (C Edition)" by Alfred Vaino Aho and Jeffrey David Ullman
http://infolab.stanford.edu/~ullman/focs.html

4. "The C Programming Language (Second Edition)" by Brian Wilson Kernighan and Dennis MacAlistair Ritchie
https://www.oreilly.com/library/view/c-programming-language/9780133086249

5. "C Programming: A Modern Approach (Second Edition)" by Kim N. King
http://knking.com/books/c2

6. "Modern C (Third Edition)" by Jens Gustedt
https://inria.hal.science/hal-02383654v2/file/modernC.pdf

7. "The C Book (Second Edition)" by Mike Banahan, Declan Brady, and Mark Doran
https://publications.gbdirect.co.uk/c_book

8. "Beej's Guide to C Programming" by Brian Hall
https://beej.us/guide/bgc/pdf/bgc_a4_c_2.pdf

Course Code	Course Name	L	T	P	Cr
DOEE250033	Basic Electrical & Electronics Engineering	2	1	2	4

Course Outcomes:

CO1: Analyze basic DC and AC electrical circuits using Ohm's Law, Kirchhoff's Laws, and network theorems.
CO2: Understand and explain power generation methods, transmission, and distribution, and describe the components of an electrical power system.
CO3: Identify and describe the construction and working principles of transformers, DC machines, and induction motors.
CO4: Analyze semiconductor devices like diodes, BJTs, and their applications in rectifiers, clippers, and amplifiers.
CO5: Explain the structure and operation of MOSFETs and CMOS logic circuits and their use in embedded and VLSI systems.

Module 1:

Introduction to Electrical Systems and DC Circuits
Basic concepts of electric current, voltage, power, and energy. Ohm's Law, Kirchhoff's Laws (KCL and KVL). Series and parallel circuits. Mesh and nodal analysis. Star-delta transformations. Energy sources: ideal and practical voltage and current sources.
Introduction to Power Generation and Distribution: Elementary idea of generation methods (thermal, hydro, solar, nuclear), transmission and distribution of electrical power; structure of power system; role of substations and transformers.

Module 2:

AC Circuits and Measurement
Sinusoidal voltage and current: amplitude, frequency, phase, time period. RMS and average values, form factor, peak factor. Phasor representation of sinusoidal quantities. AC analysis of pure R, L, and C components. RLC series and parallel circuits. Power in AC circuits: real, reactive, and apparent power. Power factor and its significance. Basics of three-phase systems.
Basic measuring instruments: voltmeter, ammeter, wattmeter, and energy meter.

Module 3:

Electrical Machines and Domestic Wiring
Magnetic circuits and electromagnetic induction. Transformers: principle, construction, EMF equation, efficiency, and applications. DC Machines: construction, types (motor/generator), working principle and applications. Three-phase induction motors: construction and working.
Basic concepts of domestic wiring: types of wiring, star case wiring, fuses, circuit breakers, earthing.

Module 4:

Semiconductor Devices and Applications
Semiconductors: intrinsic and extrinsic types. PN junction diode: V-I characteristics, rectifiers (half-wave and full-wave), clippers and clampers. Zener diode: breakdown and voltage regulation. BJT: construction, input-output characteristics in CE mode, application as a switch and amplifier. Basic introduction to optoelectronic devices (LED, photodiode).

Module 5:

MOSFET and CMOS Technology
MOSFET: structure, operation of nMOS and pMOS, output and transfer characteristics.

MOSFET as a switch and amplifier. CMOS logic: complementary MOS operation, CMOS inverter, static and dynamic behavior. Advantages of CMOS technology: low power, high noise margin, scalability. Applications of MOSFETs and CMOS in logic circuits, embedded systems, and VLSI design.

Labs / Practicals:

Electrical Engineering Experiments

1. Verification of Ohm's Law and Kirchhoff's Laws (KVL & KCL)
o Study linear resistive networks and validate current-voltage relationships.
2. Measurement of Power and Power Factor in a Single-Phase AC Circuit
o Calculate real, reactive, and apparent power using wattmeters and determine power factor.
3. Series and Parallel RLC Circuit Analysis
o Observe resonance and calculate impedance, current, and phase angle.
4. Load Test on Single-Phase Transformer
o Measure efficiency and voltage regulation under different loads.
5. Speed Control and Operation of a DC Motor
o Study working, direction control, and variation of speed with field/armature voltage.
6. Star Case Wiring and Testing of Light/Fan Circuit

• Perform practical wiring of a basic domestic setup using star configuration; verify connections and functionality.

Electronics Engineering Experiments

7. V-I Characteristics of PN Junction Diode
o Plot forward and reverse bias curves; identify cut-in and breakdown voltages.
8. Half-Wave and Full-Wave Rectifier with and without Filter
o Measure output DC voltage, ripple factor, and waveform using CRO.
9. Zener Diode as Voltage Regulator
o Test voltage regulation with varying load and input voltage.
10. Input and Output Characteristics of BJT in CE Configuration
o Measure current gain (β) and plot characteristics for amplification.
11. MOSFET Characteristics and CMOS Inverter Operation

References:

1. V.K. Mehta, Rohit Mehta – Principles of Electrical Engineering and Electronics, S. Chand Publishing
2. D.P. Kothari, I.J. Nagrath – Basic Electrical and Electronics Engineering, McGraw-Hill Education
3. M.S. Sukhija, T.K. Nagsarkar – Basic Electrical and Electronics Engineering, Oxford University Press

Course Code	Course Name	L	T	P	Cr
DASH250042	Engineering Physics	3	0	2	4

Course Outcomes:

CO-1: Be able to understand the phenomenon of interference in thin films and diffraction by slits
CO-2: Be able to understand the principle behind the working of LASERS and Optical Fibers
CO-3: Be able to understand concept of quantum mechanics and one dimensional motion of particle
CO-4: Be able to understand various properties of semiconductor, superconductor and nanomaterial
CO-5:Be able to understand electric and magnetic field behavior and drive Maxwell equations

Module 1:

Wave Optics (8 hours)
Concept of interference, Interference in thin films (reflected light)-Newton's Rings and Michelson's Interferometer, Application as wavelength measurement. Concept of Diffraction, Single slit Fraunhofer Diffraction, Diffraction Grating and Spectrum, Determination of Wavelength, Application of Grating as wavelength splitter.

Module 2:

Lasers & Optical Fibers (8 hours)
Laser: Einstein's Theory of laser action, Einstein's coefficients, population inversion and lasing action, Properties of Laser beam, Construction and working of He-Ne and semiconductor lasers, Applications of Lasers in Science and engineering. Optical Fiber: Structure, Types, Features, Light guiding mechanism, Acceptance angle and Numerical Aperture.

Module 3:

Quantum Mechanics (9 hours)
Concepts and Experiments that led to the discovery of Quantum Nature. Heisenberg uncertainty principle, Wave function and basic postulate of wave mechanics, Schrodinger time independent and time dependent wave equations, Physical interpretation of wave function and properties. The free particle problem - Particle in 1-dimensional and 3-dimensional boxes, Concept of Quantum mechanical tunneling.

Module 4:

Physics of Advanced Materials (8 hours)
Types of semiconductors, Conductivity in semiconductors,
Energy Band Gap, Hall Effect: Theory and applications, Superconductors: Properties, Meissner effect, Type I & II superconductors, Applications of superconductors, Nanomaterials: Significance of nanoscale, Properties of nanomaterials, Basics of Synthesis of nanomaterials: top-down and bottom-up approach, Applications of nanomaterials, X-ray Diffraction.

Module 5:

Introduction to Electromagnetism (7 hours)
Gradient, divergence and curl and their physical significance, Divergence and Curl of electrostatic and static Magnetic Fields, Faraday's law, equation of continuity, Displacement current, Maxwell's equations, Electromagnetic wave propagation in free space Flow of energy and Poynting vector.

Labs / Practicals:

1. To study the formation of Newton's rings and determine the wavelength of light (Sodium lamp/LASER).
2. To determine the wavelength of light (Sodium lamp/LASER) with the help of Michelson interferometer.

3. To determine the wavelength of prominent lines of light (mercury) by using plane transmission diffraction grating.
4. To determine specific rotation of sugar using half shade/ biquartz polarimeter.
5. To determine the dispersive power of material of a prism with the help of spectrometer.
6. To determine the height of given object with the help of sextant.
7 To determination of band gap of semiconductor using a P-N junction diode.
8. To study the Hall Effect and determination of hall coefficient and charge carrier concentration.
9. To measure the numerical aperture of an optical fiber.
10. To determine the coherence length and coherence time of laser using He -Ne laser.
11. To study the charge and discharge of a condenser and hence determine the time constant.
12. To determination of resonating frequency and bandwidth by LCR circuit.
13. To study the B-H/I-H curve and hysteresis losses in a given magnetic material.

References:

1. Halliday, Resnic and Walker, "Fundamentals of Physics", Publisher: John Wiley, Ninth Edition.
2 Beiser A, "Concepts of Modern Physics", Publisher: McGraw Hill International, Fifth Edition.
3 Ajoy Ghatak, "Optics", Publisher: Tata McGraw Hill, Fifth Edition.
4 S. O. Pillai, "Solid State Physics", Publisher: New Age Publishers.
5 A. Ghatak, K. Thyagarajan, "Introduction To Fiber Optics", Publisher: Cambridge University Press.
6 W.T Silfvast, "Laser Fundamentals", Publisher: Cambridge University Press.
7 R. Shankar, "Fundamentals of Physics", Publisher: Yale University Press, New Haven and London.
8 R. Shankar, Fundamentals of Physics II", Publisher: Yale University Press, New Haven and London.
9 David J. Griffiths, "Introduction to Electrodynamics", Publisher: Cambridge University Press.
10 K. K. Chattopadhyay and A. N. Banerjee, "Introduction to Nanoscience and Nanotechnology", Publisher: PHI Learning Pvt. Limited.
11 T. Pradeep, "NANO: The Essentials, understanding Nano science and Nanotechnology", Publisher: Tata McGraw-Hill Publishing Company Limited.

Course Code	Course Name	L	T	P	Cr
DASH250016	Communication Skills	1	0	2	2

Course Outcomes:

CO1 : Understand the need and importance of English language.
CO2 : Develop proficiency in the language.
CO3 : Able to use technology to enrich communication skills.
CO4 : Gain self-confidence with improved command over English.
CO5 : Apply English communication skills effectively in academic, professional, and social environments.

Module 1:

Communication Fundamentals
Communication ;Meaning, Process and importance of Communication, Types and Channels of communications , Scope and Significance of Listening ,Speaking, Reading, Writing Skills.

Module 2:

Writing Skills
Basics of Grammar: Parts of Speech, Uses of Tenses, Active Passive, Narration.

Module 3:

Vocabulary Building (6 hrs)
Word Formation, Synonyms , Antonyms, Words often Confused, One Word Substitutes, Idioms and Phrasal verbs, Abbreviations of Scientific and Technical Words.

Module 4:

Speaking Skills
Introduction to Phonetic Sounds & Articulation, Word Accent, Rhythm and Intonation, Interpersonal Communication, Oral Presentation, Body Language and Voice Modulation (Para linguistics and Non Verbal), Negotiation and Persuasion, Group Discussion, Interview Techniques (Telephonic and Video conferencing).

Module 5:

Technical Writing
Job application, CV writing, Business Letters, Notices, Report Writing, Minutes, E-mail Etiquette, Blog Writing.

Labs / Practicals:

1. Introducing Oneself, Exercise on Parts of Speech & Exercise on Tense.
2. Exercise on Agreement, Narration, Active Passive Voice & Dialogue Conversation.
3. Exercise on Writing Skills and Listening Comprehension.
4. Practice of Phonemes, Word Accent, Intonation, JAM Session.
5. Individual Presentation, Extempore and Picture Interpretation.
6. Vocabulary Building Exercises (One Word Substitute, Synonyms, Antonyms, Words Often Confused etc.) & Group Discussion.

References:

1. "The Essence of Effective Communication”, Ludlow R. and Panton F., Pubs: Prentice Hall.
2. “Effective Communication Skills”, Kulbhushan Kumar, Khanna Publishing House.
3. “A University Grammar of English”, Quirk R. and Sidney G., 3rd Edition, Pubs: Pearson Education.
4. “High School English Grammar”, Wren and Martin, Pubs: S. Chand & Company Ltd.
5. “Essentials of Business Communication”, Guffrey M.E., 8th Edition, Pubs: South-Western College Publishing.
6. “Technical Communication: Principles and Practice”, Raman M. and Sharma S., Pubs: Oxford University Press.
7. “Effective Business Communication”, Rodrigues M.V., Pubs: Concept Publishing Company, Delhi.

8. “English Vocabulary in Use”, McCarthy M. and Felicity O’ Dell.

Course Code	Course Name	L	T	P	Cr
DASH250039	Engineering Mathematics II	3	1	0	4

Course Outcomes:

CO1 : Analyze the behavior of sequences and series using convergence tests and apply them in evaluating infinite processes.
CO2 : Construct Fourier series representations of periodic functions and apply them in engineering and data modeling contexts.
CO3 : Compute Laplace transforms of various functions including step and impulse functions, and use them for solving engineering problems.
CO4 : Perform inverse Laplace transforms and use them effectively to solve initial value problems in engineering models.
CO5 : Use Fourier transforms and their properties to switch between time and frequency domains, aiding AI applications such as image/signal processing and neural network optimization.

Module 1:

Sequences and Series (9 hrs)
Limit of a sequence, monotone and Cauchy sequences, properties of convergent sequences, examples. Infinite series, positive series, tests for convergence and divergence, integral test, alternating series, Leibnitz test.

Module 2:

Fourier Series (9 hrs)
Fourier Series of Periodic functions (Euler's Formulas and Dirichlet's conditions), Even and odd function expansions (Half range sine cosine series), Fourier series over arbitrary intervals (Change of interval techniques).

Module 3:

Laplace Transformation (7 hrs)
Laplace Transforms: Laplace Transform of standard functions, First shifting theorem, Second shifting theorem, Unit step function, Dirac delta function, Laplace transforms of functions when they are multiplied and divided by 't', Laplace transforms of derivatives (First and second derivative) of functions, Laplace Transformation of integral of functions, Evaluation of integrals by Laplace transforms.

Module 4:

Inverse Laplace Transformation and Applications (11 hrs)
Inverse Laplace transform by partial fractions and shifting theorem, convolution theorem (without proof). Applications: solving Initial value problems of first order by Laplace Transform method.

Module 5:

Fourier Transformation (9 hrs)
Introduction to Fourier Transform, Properties- Linearity, Time and frequency shifting property (1st and 2nd shifting theorem), Differentiation and integration in time domain and their transforms, Fourier Integral theorem (statement only), Fourier transform and its inverse, Convolution theorem (Statement and applications).

Labs / Practicals:

N/A

References:

1. Bali, N. P., & Goyal, M. A Text Book of Engineering Mathematics. Laxmi Publications (P) Ltd.
2. Kreyszig, E. Advanced Engineering Mathematics (9th ed.). John Wiley & Sons, New Delhi.
3. Ramana, B. V. Higher Engineering Mathematics. Tata McGraw-Hill, New Delhi.
4. Grewal, B. S. Higher Engineering Mathematics. Khanna Publishers, New Delhi.
5. H.K. Dass, Advanced Engineering Mathematics. S. Chand & Company Pvt. Ltd., New Delhi.
6. Jain, R. K., & Iyengar, S. R. K. Advanced Engineering Mathematics. Narosa Publishing House, New Delhi.
7. Dass, H. K., & Verma, R. Engineering Mathematics. S. Chand & Company Pvt. Ltd., New Delhi.
8. Pundir, R. S., & Pundir, S. Engineering Mathematics. Laxmi Publications (P) Ltd.
9. Ghosh, S. C. Engineering Mathematics. PHI Learning Pvt. Ltd.
10. Gupta, C. S., & Malik, A. Mathematical Analysis and Applications. Wiley Eastern / Narosa Publishing House

Course Code	Course Name	L	T	P	Cr
DOEE250059	Digital Electronics	3	0	2	4

Course Outcomes:

CO1: Perform conversions between number systems and design logic functions using basic and universal gates.
CO2: Simplify Boolean expressions using Karnaugh Maps and Boolean algebra for circuit minimization.
CO3: Design and analyze combinational logic circuits like adders, subtractors, multiplexers, and decoders.
CO4: Understand and implement sequential circuits such as flip-flops, counters, and shift registers.
CO5: Understand the architecture and applications of memory units and programmable logic devices.

Module 1:

Number System and Logic Gates: Representation and interconversion among binary, octal, decimal, and hexadecimal number systems, Binary arithmetic operations: addition, subtraction, multiplication, and division, Binary Coded Decimal (BCD), Excess-3, and Gray codes, Signed and unsigned binary number representation,1's and 2's complement techniques for binary subtraction, Code conversions: Binary to BCD, Excess-3, and Gray codes, and vice versa, Logic gates: AND, OR, NOT, NAND, NOR, XOR, XNOR, Universal gates and implementation of basic logic functions using universal gates.

Module 2:

Boolean Algebra and Logic Simplification: Boolean laws and theorems, DeMorgan’s theorems, SOP and POS forms, Canonical forms, Karnaugh Map (2, 3, 4 variables) – minterms and maxterms, Quine-McCluskey method (basic idea), Logic expression simplification and gate minimization.

Module 3:

Combinational Logic Circuits: Adders: Half adder, full adder, ripple carry adder, carry look-ahead adder, Subtractors: Half and full subtractors, Comparators, Multiplexers (MUX) and De-multiplexers (DEMUX) Encoders and Decoders, BCD to 7-segment decoder, Design of combinational logic using MUX, DEMUX.

Module 4:

Sequential Logic Circuits: Latches and Flip-flops: SR, D, T, JK, Master-Slave, Characteristic tables and excitation tables, Edge and level triggering, Timing diagrams, Counters: Asynchronous (ripple) and Synchronous counters, Up/down counters, modulo-N counters, Shift registers: SISO, SIPO, PISO, PIPO, Ring counter and Johnson counter, Mealy and Moore Finite State Machines (FSMs).

Module 5:

Memory & Logic Families: Classification of memories: RAM, ROM, PROM, EPROM, EEPROM, Memory decoding and address space, Overview of programmable logic devices: PLA, PAL, CPLD, FPGA (introductory), Logic families: TTL, CMOS. Characteristics: propagation delay, power dissipation, noise margin, fan-out, fan-in.

Labs / Practicals:

1. Verification of Logic Gates: Implement and verify the truth tables of AND, OR, NOT, NAND, NOR, XOR, and XNOR gates.
2. Universal Gates Implementation: Realize basic logic functions using only NAND or NOR gates.
3. Boolean Expression Simplification: Simplify given Boolean expressions and implement the minimized circuits using logic gates.
4. Design of Half Adder and Full Adder: Construct and test circuits for half and full adders using logic gates.
5. Design of Half Subtractor and Full Subtractor: Implement and verify half and full subtractor circuits.
6. Multiplexer and Demultiplexer: Design and test 4:1 MUX and 1:4 DEMUX circuits.
7. Flip-Flop Implementations: Implement and observe SR, D, T, and JK flip-flops using basic gates or ICs.
8. Counters: Design and test asynchronous and synchronous up/down counters (e.g., mod-8, mod-10).
9. Shift Registers: Construct and analyze SIPO, SISO, PIPO, and PISO shift registers.

10. BCD to 7-Segment Display: Design a decoder to display BCD inputs on a 7-segment LED display.

References:

1. M. Morris Mano, Michael D. Ciletti – Digital Design, Pearson Education
2. R.P. Jain – Modern Digital Electronics, McGraw-Hill Education
3. Thomas L. Floyd – Digital Fundamentals, Pearson Education
4. John F. Wakerly – Digital Design: Principles and Practices, Pearson
5. Tokheim – Digital Electronics: Principles and Applications, McGraw-Hill Education

Course Code	Course Name	L	T	P	Cr
DCSA250016	Computer Aided Graphics Design	1	1	4	4

Course Outcomes:

CO1: Understand the fundamental principles of computer graphics and graphic design.
CO2: Apply transformations, projections, and rendering techniques for 2D and 3D modeling.
CO3: Develop proficiency in using CAD and vector graphic design tools for technical drawings and creative graphics.
CO4: Demonstrate the ability to perform digital editing, modeling, and visualization using software like AutoCAD, Adobe Illustrator, or Blender.
CO5: Create complete design projects from concept to digital representation, including rendering and documentation.

Module 1:

Basics of Computer Graphics: Introduction to Computer Graphics, Applications of Graphics in Design, Pixel and Resolution Concepts, Raster vs Vector Graphics, Coordinate Systems, Line and Circle Drawing Algorithms (DDA, Bresenham).

Module 2:

2D Transformations and Viewing: Basic 2D Transformations: Translation, Rotation, Scaling, Reflection, Shearing. Homogeneous Coordinates, Matrix Representation of Transformations, Composite Transformations. Windowing and Clipping Techniques: Cohen–Sutherland, Liang–Barsky.

Module 3:

3D Graphics and Modeling: 3D Coordinate Systems, 3D Transformations, Projection Techniques (Perspective and Parallel), Hidden Surface Removal, Basics of 3D Modeling and Visualization, Introduction to 3D Modeling Tools (e.g., Blender/AutoCAD).

Module 4:

Graphic Design Tools and Techniques: Introduction to Design Software: Adobe Illustrator/CorelDRAW/Inkscape. Vector Graphics Tools: Pen, Shape, Path Editing. Typography, Colors, Layers, and Effects. Designing Logos, Brochures, and Posters.

Module 5:

CAD and Real-World Applications: Introduction to AutoCAD or Equivalent CAD Tool, Drawing 2D and 3D Objects, Layers and Blocks, Dimensioning and Annotation, Rendering and Exporting Drawings. Application of CAD in Engineering Drawings, Architecture, and Product Design.

Labs / Practicals:

Module 1 & 2 – 2D Graphics and Transformations
1. Implement DDA Line Drawing Algorithm using C/C++ or Python.
2. Perform 2D Transformations (Translation, Rotation, Scaling, Reflection) on a geometric shape.
3. Implement Cohen–Sutherland Line Clipping Algorithm for windowing and clipping.

Module 3 – 3D Graphics and Modeling
4. Create and Transform 3D Objects (e.g., Cube, Sphere) using Blender.
5. Apply Perspective and Parallel Projections to a 3D model and visualize the differences.

Module 4 – Graphic Design Tools
6. Design a Logo using vector graphic tools in Adobe Illustrator or Inkscape.
7. Create a Poster or Brochure using typography, color themes, and shapes.

Module 5 – CAD Applications
8. Draw a 2D Engineering Component using AutoCAD with layers and dimensions.
9. Model a Simple 3D Object in AutoCAD and apply rendering.

10. Export and Annotate Drawings from AutoCAD for documentation or print-ready formats.

References:

1. Donald Hearn and M. Pauline Baker – Computer Graphics with OpenGL, Pearson Education.
2. D. F. Rogers and J. A. Adams – Mathematical Elements for Computer Graphics, McGraw-Hill.
3. Sham Tickoo – Learning AutoCAD: A Project-Based Tutorial, CADCIM Technologies.
4. Adobe Creative Team – Adobe Illustrator Classroom in a Book, Adobe Press.
5. Blender Foundation – Blender Basics: Classroom Tutorial Book, Blender Foundation.

Course Code	Course Name	L	T	P	Cr
DASH250030	Economics for Engineers	1	1	0	2

Course Outcomes:

CO1 : Understand basic economic concepts, including demand, supply, cost, and market structures.
CO2 : Analyze the economic feasibility of engineering projects using cost-benefit and break-even analysis.
CO3 : Evaluate different investment alternatives using engineering economic tools like NPV, IRR, and Payback Period.
CO4 : Apply knowledge of macroeconomic indicators and fiscal/monetary policies in the context of engineering.
CO5 : Develop the ability to assess the economic impact of engineering decisions in real-world scenarios.

Module 1:

Basic Economic Concepts
Introduction to Economics: Micro vs Macro, Utility, Demand and Supply Analysis, Elasticity of Demand and Supply, Equilibrium, Law of Diminishing Returns, Opportunity Cost.

Module 2:

Cost and Production Analysis
Types of Costs: Fixed, Variable, Marginal, Average, Total, Cost-Output Relationship, Break-even Analysis, Law of Variable Proportions, Returns to Scale, Cost-Volume-Profit Analysis.

Module 3:

Market Structures and Pricing
Types of Market: Perfect Competition, Monopoly, Oligopoly, Monopolistic Competition, Pricing Strategies, Price Discrimination, Government Policies and Pricing.

Module 4:

Engineering Economics
Time Value of Money, Present Worth and Future Worth, Net Present Value (NPV), Internal Rate of Return (IRR), Payback Period, Comparison of Investment Alternatives, Depreciation and Inflation Considerations.

Module 5:

Macroeconomic Concepts
National Income, GDP, GNP, Inflation, Deflation, Unemployment, Fiscal and Monetary Policy, Role of RBI and Government in Economic Planning, Economic Indicators Relevant to Engineers.

Labs / Practicals:

N/A

References:

1. R. Panneerselvam, Engineering Economics, PHI Learning.
2. D.M. Mithani, Managerial Economics, Himalaya Publishing House.
3. M.L. Jhingan, Microeconomic Theory, Vrinda Publications, Latest Edition.
4. Chan S. Park, Contemporary Engineering Economics, Pearson Education.
5. Paul A. Samuelson and William D. Nordhaus, Economics, McGraw Hill.

Course Code	Course Name	L	T	P	Cr
DCSE250113	Programming in Python	2	0	4	4

Course Outcomes:

CO1: Remember syntax rules, basic operators, and built-in features of Python 3.

CO2: Understand guiding principles and coding conventions from Zen of Python.

CO3: Apply object-oriented programming paradigms in design and practice.

CO4: Analyze the runtime efficiency and resource utilization of Python programs.

CO5: Evaluate the outcomes of programs manually with dry runs and trace tables.

CO6: Create modular and maintainable programs following Pythonic practices.

Module 1:

Constants, types, identifiers, keywords, comments, print, input, help functions.
Operator precedence and associativity, arithmetic operators (**, *, /, //, %, +, -).
Binary number system, bin function, bitwise operators (shift, &, ^, |), bit masking.
Relational and equality operators (== vs "is"), not, short-circuit evaluation (and, or).

Module 2:

Selection: if, elif, else, mutual exclusion, indent (space vs tab), nesting, pass.
Iteration: while loop, range of integers, in operator, for loop, break vs continue.
Function: parameters, arguments, return value, global, recursion, lambda, closure.
Modules: from, import, as, __name__, top-level module, avoiding circular imports.

Module 3:

ASCII and UTF-8, ord and chr functions, escape sequence, strings, docstring, regex.
Mutability, id function, object vs reference, is vs ==, interning, datetime module.
Ordered: list, tuple, negative indexing, slicing, sort with key, pack and unpack.
Unordered: set, dictionary (keys, values, items), subscript vs get method, None.

Module 4:

Class definition, self parameter in methods, constructor, attributes, instantiation.
Dot notation, type function, __str__ vs __repr__, methods for built-in operators.
Static attributes, annotations, private name mangling, getter and setter methods.
Inheritance, object, super, C3 linearization, ambiguous MRO, abstract base class.

Module 5:

Syntax errors vs runtime exceptions, traceback, try, except, else, finally, raise.
Matching except clause, except with multiple exceptions, finally with return.
Built-in exceptions, user-defined exception classes, exceptions with arguments, as.
Command-line arguments, file opening modes (read/write/append), with statement.

Labs / Practicals:

01. Test the precedence and associativity of arithmetic operators.

02. Test the bitwise operators and verify the results with bin function.

03. Test the relational operators (with and without chaining).

04. Test the short-circuit evaluation of and, or operators using function calls as operands.

05. Convert from metric prefix (KB/MB/GB/TB) to binary prefix (KiB/MiB/GiB/TiB).

06. Use if-elif-else ladder to print the grade as per the score obtained in a subject.

07. Find the mean, median, mode, variance, and standard deviation for a list of scores.

08. Print values using additive/subtractive rules of roman numerals: I, V, X, L, C, D, M.

09. Use string repetition with multiplication to generate patterns without nested loops.

10. Use nested loops to generate asymmetric and symmetric patterns of arbitrary size.

11. Check whether a given string is a palindrome (or not) without reversing the string.

12. Test positional arguments, default arguments, and keyword arguments for functions.

13. Write a function to check whether an indexed collection of numbers is already sorted.

14. Write a function for primality testing, and use it to find first n pairs of twin primes.

15. Write a function to generate the Collatz sequence (up to 1) for a positive integer.

16. Generate the terms of Fibonacci sequence using iterative and recursive approaches.

17. Print optimal sequence of moves for a given instance of “Towers of Hanoi” puzzle.

18. Find all combinations to make a given payment from a limited set of coin types.

19. Check if a letter is a vowel using two approaches: if-elif-else and in operator.

20. Find vowel count in a string using a lookup table for case-insensitive vowel check.

21. Write a regular expression to check whether a string is a valid email ID (or not).

22. Use case-insensitive and dot-insensitive matching to check if two email IDs are same.

23. Display a countdown timer in HH:MM:SS format using carriage return ("\r").

24. Get current time and find clockwise angles of hour hand, minute hand, second hand.

25. Find the day of week for any date in Gregorian calendar using doomsday algorithm.

26. Define Matrix class and implement these operators: unary +, unary -, +, -, *, ==.

27. Define Person class in a module and extend it with Student class in another module.

28. Incrementally generate roll numbers in Student constructor using a static counter.

29. Populate a list with Student instances and write their attributes in a new CSV file.

References:

1. "Python Tutorial" by Guido van Rossum and the Python development team
https://bugs.python.org/file47781/Tutorial_EDIT.pdf

2. "Think Python: How to Think Like a Computer Scientist (Second Edition)" by Allen Benjamin Downey
https://greenteapress.com/thinkpython2/thinkpython2.pdf

3. "Python for Everybody: Exploring Data in Python 3" by Charles Russell Severance
https://do1.dr-chuck.com/pythonlearn/EN_us/pythonlearn.pdf

4. "Beej's Guide to Python Programming" by Brian Hall
https://beej.us/guide/bgpython/pdf/bgpython_a4_c_2.pdf

5. "Learning Python: Powerful Object-Oriented Programming (Sixth Edition)" by Mark Lutz
https://learning-python.com

6. "Fluent Python: Clear, Concise, and Effective Programming (Second Edition)" by Luciano Ramalho
https://www.fluentpython.com

Course Code	Course Name	L	T	P	Cr
DCSE250055	Data Structures	3	0	2	4

Course Outcomes:

CO1: Remember the different types of linear and non-linear data structures.

CO2: Understand how structuring of data can reduce the average access time.

CO3: Apply concrete data structures to implement abstract data structures.

CO4: Analyze the fundamental operations supported by various data structures.

CO5: Evaluate the number of steps required to do an operation on a data structure.

CO6: Create efficient implementations that are suitable for real-world applications.

Module 1:

Multi-dimensional arrays, addressing in row-major and column-major orderings.
Searching: linear search, binary search, uniform binary search, interpolation search.
Sorting by comparison: selection, bubble, insertion, Shell sort, quicksort, mergesort.
Non-comparison sorting: counting sort, pigeonhole sort, radix sort (LSD and MSD).

Module 2:

Linked lists types: linear and circular, unidirectional and bidirectional.
Operations on linked lists: create, insert, delete, traverse, reverse, find.
Open address hashing, collision, linear probing, quadratic probing, double hashing.
Hashing with separate chaining, load factor, applications of hash tables.

Module 3:

Abstract data structures, concrete implementation using arrays and linked lists.
Stacks: LIFO access, operations (push, pop, top), call stack, expression evaluation.
Queues: FIFO access, operations (enqueue, dequeue, peek), linear queue, ring buffer.
Double-ended queue (input-restricted, output-restricted), priority queue, DEPQ.

Module 4:

Graphs (undirected and directed), adjacency matrix, adjacency list, weighted graphs.
Graph traversals: breadth-first search, depth-first search, iterative deepening search.
Graph connectivity, directed acyclic graphs, topological sorting, spanning trees.
Arborescence, branching factor, binary trees (skew, strict, complete, perfect).

Module 5:

Tree traversal: pre-order (NLR), in-order (LNR), post-order (LRN), reverse variants.
Binary heap (min-heap and max-heap), Fibonacci heap, binomial heap, heapsort.
Binary search tree (BST), sorting with BST, threaded BST (single and double).
Disjoint-set data structure, operations: makeset, find, union (by rank or by size).

Labs / Practicals:

Each data structure and supporting algorithms should be implemented in C or Python.

01. Use arrays to store univariate polynomials for doing addition and multiplication.

02. Write a function to find min/max element and use it to implement selection sort.

03. Implement an adaptive variant of bubble sort that runs in linear time for best case.

04. Implement insertion sort with a type-agnostic design (like qsort from stdlib.h).

05. Perform non-comparative sorting with counting sort and radix sort (LSD and MSD).

06. Create unidirectional and bidirectional linked lists with linear and circular variants.

07. Create open address hash tables with linear probe, quadratic probe, double hashing.

08. Create hash table with an array of pointers for separate chaining using linked lists.

09. Implement stack, queue (linear and circular), deque using arrays and linked lists.

10. Convert among prefix, infix, postfix expressions, and evaluate the postfix notation.

11. Perform breadth-first traversal, depth-first traversal, topological sort on a digraph.

12. Implement priority queue using binary max heap, supporting insertion and deletion.

13. Heapify an array of strings in linear time and do heap sort in lexicographic order.

14. Perform insert and delete on a binary search tree threaded for in-order traversal.

15. Implement disjoint-set data structure supporting makeset, find, union operations.

References:

1. "Think Data Structures" by Allen Benjamin Downey
https://greenteapress.com/thinkdast/thinkdast.pdf

2. "Open Data Structures: An Introduction" by Pat Morin
https://www.aupress.ca/app/uploads/120226_99Z_Morin_2013-Open_Data_Structures.pdf

3. "An Open Guide to Data Structures and Algorithms" by Paul W. Bible and Lucas Moser
https://pressbooks.palni.org/anopenguidetodatastructuresandalgorithms

4. "Computer Science II" by Chris Bourke
https://cse.unl.edu/~cbourke/ComputerScienceTwo.pdf

5. "Data Structures and Algorithms" by Alfred Vaino Aho, John Edward Hopcroft, Jeffrey David Ullman
https://dl.acm.org/doi/10.5555/577958

6. "Data Structures and Algorithm Analysis in C (Second Edition)" by Mark Allen Weiss
https://dl.acm.org/doi/10.5555/248735

Course Code	Course Name	L	T	P	Cr
DOEE250044	Computer Organization and Architecture	3	0	2	4

Course Outcomes:

CO1: Explain processor fundamentals, instruction representation, and hardware operations including arithmetic and logical functions.
CO2: Analyze instruction set architectures, addressing modes, and computer arithmetic with parallelism and SIMD extensions.
CO3: Design basic datapath and pipelined architectures, and evaluate instruction-level parallelism and hazards in modern processors.
CO4: Demonstrate memory mapping techniques, cache design, memory hierarchy, and virtual memory mechanisms.
CO5: Evaluate memory management strategies including RAID levels, disk storage, multithreading, and multiprocessor systems.

Module 1:

Introduction of Processor:Introduction, Technologies for building Processors and Memory, Performance, The Power Wall, Operations of the Computer Hardware, Operands Signed and Unsigned numbers, Representing Instructions, Logical Operations, Instructions for Making Decisions

Module 2:

Instructions Set:MIPS Addressing for 32-Bit Immediate and Addresses, Parallelism and Instructions: Synchronization, Translating and Starting a Program, Addition and Subtraction, Multiplication, Division, Floating Point, Parallelism and Computer Arithmetic: Sub word Parallelism, Streaming SIMD Extensions and Advanced Vector Extensions in x86.

Module 3:

Architecture Building Block: Logic Design Conventions, building a Datapath, A Simple Implementation Scheme, overview of Pipelining, Pipelined Datapath, Data Hazards: Forwarding versus Stalling, Control Hazards, Exceptions, Parallelism via Instructions, The ARM Cortex – A8 and Intel Core i7 Pipelines, Instruction –Level Parallelism and Matrix Multiply Hardware Design language.

Module 4:

Memory Mapping: Memory Technologies, Basics of Caches, Measuring and Improving Cache Performance, dependable memory hierarchy, Virtual Machines, Virtual Memory, Using FSM to Control a Simple Cache, Parallelism and Memory Hierarchy: Redundant Arrays of Inexpensive Disks, Advanced Material: Implementing Cache Controllers.

Module 5:

Memory Management: Disk Storage and Dependability, RAID levels, performance of storage systems, Introduction to multi- threading clusters, message passing multiprocessors.

Labs / Practicals:

Module 1 & 2: Processor Basics and Instruction Set
1. Simulation of Basic Arithmetic Operations in MIPS Assembly
o Objective: Write MIPS assembly code to perform addition, subtraction, multiplication, and division of two integers.
o Tool: MIPS Simulator (e.g., QtSpim, MARS)
2. Binary and Hexadecimal Conversion & Representation
o Objective: Implement a program (in C or Assembly) to convert between binary, hexadecimal, and decimal formats; demonstrate signed and unsigned number representations.

3. Instruction Encoding and Decoding
o Objective: Manually encode and decode sample MIPS instructions and verify using a simulator.

Module 3: Datapath and Pipelining
4. Design a Basic Arithmetic Logic Unit (ALU) Using VHDL/Verilog
o Objective: Implement an ALU capable of performing AND, OR, ADD, SUB, and SLT operations.
o Tool: Xilinx Vivado / ModelSim / Logisim
5. Simulate Single-Cycle and Multi-Cycle Datapaths
o Objective: Model a simplified processor datapath (single-cycle and pipelined) and compare performance.
o Tool: Digital Simulator / Logisim / Ripes (RISC-V simulator)
6. Pipeline Hazard Detection and Resolution
o Objective: Simulate and analyze data and control hazards in a pipelined processor; demonstrate stalling and forwarding.

Module 4 & 5: Memory Mapping and Management
7. Cache Memory Simulation
o Objective: Simulate direct-mapped, fully-associative, and set-associative caches; compute hit/miss ratio.
o Tool: Custom Python/C program or existing cache simulator tools.
8. Implementation of Virtual Memory using Paging
o Objective: Simulate address translation using page tables; demonstrate page faults and TLBs.
o Tool: Python or Java-based simulation
9. Disk Scheduling Algorithm Simulation
o Objective: Implement and compare performance of FCFS, SSTF, SCAN, and LOOK disk scheduling algorithms.
10. RAID Level Simulation and Performance Comparison
• Objective: Simulate different RAID levels (RAID 0, 1, 5) and evaluate redundancy, speed, and fault tolerance.

References:

1. David A. Patterson and John L. Hennessey, “Computer organization and design, The Hardware/Software interface”, Morgan Kauffman / Elsevier, Fifth edition, 2014
2. V. Carl Hamacher, Zvonko G. Varanesic, and Safat G. Zaky, “Computer Organization “, 6 th edition, McGraw-Hill Inc, 2012.
3. William Stallings, “Computer Organization and Architecture”, 8th Edition, Pearson Education, 2010

Course Code	Course Name	L	T	P	Cr
DOEE250022	Assembly Language and Microprocessors	3	0	2	4

Course Outcomes:

CO1. Describe the architecture, pin configuration, and functional units of 8085/8086 microprocessors.
CO2. Explain various addressing modes and instruction sets of 8085/8086 microprocessors.
CO3. Write, assemble, and execute simple assembly language programs for arithmetic, logic, and data transfer operations.
CO4. Design small embedded system applications using microprocessor/microcontroller programming.

Module 1:

Introduction to Assembly Language:
Basics of low-level programming, Machine language vs assembly language, Role of assembler, Structure of an assembly language program, Instruction formats and addressing modes

Module 2:

Introduction to Microprocessor: Definition of Microprocessor, Bit, Byte, Nibble, Instruction, Mnemonics, Assembly Level Programming, Assembler, Hexadecimal to Decimal conversion and vice-versa, Evolution of microprocessors, Block diagram of microprocessor based digital computer: functions of each Block, Application of microprocessors.

Module 3:

Architecture and Organization of 8085: Central Processing Unit, Accumulator, General Purpose Register, Status Register, ALU, Program Counter (PC), Stack Pointer (SP), Control Unit, The Clock, Reset, Interrupt, Timing and Control Unit, Pin Configuration of 8085: functions of each pin.

Module 4:

8085 Instruction set: Machine language instruction format : Single byte, two byte, three byte instructions, Various addressing modes, Data transfer operation and instruction, Arithmetic operation and instruction, Logical operation and instruction, Branch operation and instruction, Stack operation and instruction, Input/Output and machine control operation and instruction

Module 5:

Introduction to 16 bit microprocessors and other advanced microprocessors, 8086 pin diagram: Functions of each pin, 8086 Architecture, various registers: General Purpose, Segment, Index,
Base, Pointers, flags; Interrupts, 8086 Addressing Modes, Instruction Format.

Labs / Practicals:

1. To write a program to add two 8 bit number using 8085 microprocessor.
2. To write a program to subtract two 8 bit numbers using 8085 microprocessor.
3. To write an assembly language program for multiplication of two 8-bit numbers using 8085 instructions.
4. To write an assembly language program for division of two 8-bit numbers using 8085 instructions.
5. To write an assembly language program for addition of two 16-bit number without carry using 8086 instruction.
6. To write an assembly language program for 1’s complement of an 8-bit numbers using 8085 instructions.
7. To write an assembly language program for 2’s complement of an 8-bit numbers using 8085 instructions.
8. To write an assembly language program for multiplication of two 16-bit number using 8086 instructions.

References:

1. Microprocessor and Microcomputer: Gaonkar (PRI)
2. Microprocessor and microcontroller: Senthilkumar (Oxford)

3. 8085 Microprocessor, programming and interface: Srinath (PHI)
4. Microprocessors: Theory and application: MahammadRafiquzzaman(PHI)

Course Code	Course Name	L	T	P	Cr
DASH250104	Technical Presentation and Report Writing	0	0	4	2

Course Outcomes:

CO1 : Understand the principles and formats of technical communication including different types of reports and presentations.
CO2 : Prepare clear, concise, and well-structured technical documents for academic and professional purposes.
CO3 : Design and deliver effective oral presentations using appropriate tools and visual aids.
CO4 : Demonstrate proficiency in grammar, technical vocabulary, and formal style required in technical writing.
CO5 : Collaborate in teams to produce technical content and receive peer feedback constructively.

Module 1:

Introduction to Technical Communication (6 hrs)
Principles and characteristics of technical communication, Purpose and types of technical documents, Features of a good technical report, Audience analysis.

Module 2:

Writing Skills for Technical Documents (8 hrs)
Structure of reports: title page, abstract, table of contents, introduction, body, conclusion, and references. Writing instructions, lab reports, project reports, and proposals. Use of charts, graphs, and tables.

Module 3:

Grammar and Language in Technical Writing (6 hrs)
Use of formal tone and objective language, sentence construction, common errors, punctuation, and vocabulary building for technical writing. Editing and proofreading techniques.

Module 4:

Preparing Technical Presentations (8 hrs)
Planning a presentation, structuring content, use of slides, visual aids, and multimedia tools (e.g., PowerPoint, Prezi). Non-verbal communication and handling audience questions.

Module 5:

Practice and Evaluation (8 hrs)
Student individual and group presentations on technical topics, peer reviews, instructor feedback. Writing mini technical reports based on lab/project work or case studies.

Labs / Practicals:

1. Analyzing Samples of Technical Documents
Objective: Identify the structure and features of different technical documents (e.g., lab report, project report, instruction manual).
2. Writing a Lab Report Based on Practical Work
Objective: Write a complete lab report including abstract, objectives, procedure, observations, and conclusion.
3. Creating a Project Proposal Document
Objective: Prepare a short technical proposal on a selected topic including problem statement, objectives, methodology, and timeline.
4. Designing Visual Elements (Graphs, Charts, Tables)
Objective: Use data to create visual aids using Excel/PowerPoint and integrate them into a technical report.
5. Grammar and Proofreading Workshop
Objective: Identify and correct errors in sample texts focusing on sentence construction, punctuation, and vocabulary.

6. Preparing and Delivering an Individual Presentation
Objective: Create and present a 5-minute talk on a technical topic using PowerPoint or Prezi.
7. Group Presentation on a Technical Case Study
Objective: Collaboratively research and present findings on a real-world technical issue or innovation.
8. Peer Review and Feedback Session
Objective: Evaluate and provide constructive feedback on peers' written reports and oral presentations.
9. Editing and Improving a Poorly Written Report
Objective: Revise and rewrite a given flawed report to meet academic/professional standards.
10. Writing a Mini Technical Report on a Chosen Topic
Objective: Submit a short report (3–5 pages) on a lab experiment, research article, or emerging technology.

References:

1. Meenakshi Raman and Sangeeta Sharma – Technical Communication: Principles and Practice, Oxford University Press.
2. M. Ashraf Rizvi – Effective Technical Communication, Tata McGraw-Hill.
3. Sharma, R.C. and Krishna Mohan – Business Correspondence and Report Writing, Tata McGraw-Hill.
4. Lesikar, Raymond V. et al. – Basic Business Communication, Tata McGraw-Hill.
5. David F. Beer and David McMurrey – A Guide to Writing as an Engineer, Wiley India.

Course Code	Course Name	L	T	P	Cr
DCSE250060	Database Management Systems	2	0	4	4

Course Outcomes:

CO1. Understand the basic database concepts, data models, schemas and instances.
CO2. Learn database design using Entity Relationship model and learn the use of constraints and relational algebra operations.
CO3. Gain proficiency in SQL and construct queries using SQL.
CO4. Understand the importance of normalization in database design.
CO5. Understand transaction processing, concurrency control and recovery

Module 1:

Introduction to Databases, File systems vs. Database systems, Characteristics of DBMS, Applications of DBMS, Advantages of using a DBMS, Database users and administrators, Three-level architecture of DBMS (ANSI/SPARC architecture), Data independence, Database schema, Instance, Generalization, Specialization, Structure of a DBMS, Overview of DBMS software (e.g., MySQL, SQL Server, Oracle, PostgreSQL)

Module 2:

Introduction to Data models: Hierarchical, Network, Relational, Object-oriented, Data Modeling using Entity Relationship (ER) model: entities, entity types, attributes, relationships, relationship types, E/R diagram notation, examples.
Relational data model: Concepts, schema, instances, keys, constraints, Relational algebra: Select, project, union, set difference, Cartesian product, joins, ER to Relational mapping, Relational algebra and relational calculus, Tuple Relational Calculus, Domain Relational Calculus

Module 3:

Structured Query Language (SQL): Basics of SQL, DDL, DML,DCL, structure – creation, alteration, defining constraints – Primary key, foreign key, unique, not null, check, IN operator, Functions - aggregate functions, Built-in functions – numeric, date, string functions, set operations, sub-queries, correlated sub-queries, Use of group by, having, order by, join and its types, Exist, Any, All , view and its types. Transaction control commands – Commit, Rollback, Save point, Cursors, Indexes, stored procedures, Triggers

Module 4:

Normalization – Introduction, Non loss decomposition and functional dependencies, First, Second, and third normal forms – dependency preservation, Boyce/Codd normal form.
Higher Normal Forms - Introduction, Multi-valued dependencies and Fourth normal form, Join dependencies and Fifth normal form.

Module 5:

Transaction management and Concurrency control: Transaction processing and Error recovery - concepts of transaction processing, ACID properties, and serializability concurrency control, Lock based concurrency control (2PL, Deadlocks), Time stamping methods, optimistic methods, and database recovery management. Error recovery and logging, undo, redo, undo-redo logging and recovery methods, Introduction to database security concepts.

Labs / Practicals:

1. Create a database, define tables using DDL, apply constraints (Primary, Foreign, Not Null, Unique).
2. Insert, update, delete, and retrieve records using SQL DML commands.
3. Write SQL queries using WHERE, ORDER BY, GROUP BY, HAVING, LIKE, etc.
4. Write SQL queries to demonstrate the use of INNER JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN, and SELF JOIN using appropriate sample tables.
5. Write nested subqueries with IN, EXISTS, and ANY/ALL.

6. Draw ER diagram for a sample application (e.g., Library or Hospital), map it to relational schema.
7. Normalize sample unstructured data up to 3NF or BCNF. Implement before-and-after schemas in SQL.
8. Create and execute stored procedures/functions for common operations.

References:

1. Silberschatz, Korth, Sudarshan – Database System Concepts, McGraw Hill Year: 2020 (7th edition).
2. Elmasri and Navathe – Fundamentals of Database Systems, Pearson Edition 2016
3. Raghu Ramakrishnan, Johannes Gehrke – Database Management Systems, McGraw Hill
4. Database Systems Concepts Author H.f.Korth and Silberschatz Publisher McGraw Hill
5. Database System Author Hector Garcia-Molina, Jeff Ullman, and Jennifer Widom Publisher Pearson Edition 2nd Edition
6. C.J. Date – An Introduction to Database Systems, Pearson

Course Code	Course Name	L	T	P	Cr
DCSA250056	Numerical Methods	3	1	0	4

Course Outcomes:

CO1: Solve algebraic and transcendental equations using appropriate numerical methods.
CO2: Apply interpolation techniques to estimate values for given discrete data.
CO3: Use numerical techniques to solve systems of linear equations efficiently.
CO4: Compute derivatives and definite integrals using numerical differentiation and integration methods.
CO5: Solve ordinary differential equations using numerical approaches and analyze the accuracy of solutions.

Module 1:

Solution of Equations and Interpolation:
Bisection method, Regula-Falsi method, Newton-Raphson method, Forward & Backward differences, Newton's & Gauss interpolation, Lagrange interpolation, Divided difference method

Module 2:

Numerical Differentiation and Integration:
Numerical differentiation using finite differences
Numerical integration using: Trapezoidal rule, Simpson's 1/3 rule, Simpson's 3/8 rule

Module 3:

Solution of Ordinary Differential Equations (ODEs):
Taylor series method, Euler's method, Modified Euler's method, Runge-Kutta methods (2nd and 4th order)

Module 4:

Predictor-Corrector Methods:
Milne's predictor-corrector method, Adams-Bashforth and Adams-Moulton methods

Module 5:

Numerical Solution of Partial Differential Equations (PDEs):
Finite difference method for Laplace and Poisson equations, Heat equation (Explicit and Crank-Nicholson methods), Wave equation using explicit method

Labs / Practicals:

N/A

References:

1. P. Kandasamy, K. Thilagavathy & K. Gunavathi, Numerical Methods (S. Chand, 2nd Ed, 2012)
2. S. S. Sastry, Introductory Methods of Numerical Analysis (PHI, 4th Ed, 2005)
3. E. Kreyszig, Advanced Engineering Mathematics (Wiley, 9th Ed, 2006)
4. B. S. Grewal, Higher Engineering Mathematics (Khanna, 35th Ed, 2010

Course Code	Course Name	L	T	P	Cr
DCSE250102	Operating Systems	3	0	2	4

Course Outcomes:

CO1: Understand the basics of operating systems like kernel, shell, types and services of operating systems.
CO2:Understand the concept of program, process and thread and analyse various CPU scheduling Algorithms.
CO3: Describe and analyse the memory management and its allocation policies.
CO4: Understand the issues related to file system interface and implementation.
CO5: Understand disk management and explain disk scheduling algorithms for better utilization of external memory.
6: Configure OS in an efficient and secure manner.

Module 1:

Introduction: Overview of Operating System, basic concepts, UNIX/LINUX Architecture, Kernel, services and systems calls, system programs.

Module 2:

Process Management and Memory management: Process Management: Process concepts, operations on processes, IPC, Process Scheduling, Multithreaded programming. Memory management: Memory allocation, Swapping, Paging, Segmentation, Virtual Memory, various faults.

Module 3:

File management: Concept of a file, access methods, directory structure, file system mounting, file sharing and protection, file system structure and implementation, directory implementation, free space management, efficiency and performance. Different types of file systems.

Module 4:

I/O System: Mass storage structure - overview, disk structure, disk attachment, disk scheduling algorithms, swap space management, RAID types.

Module 5:

OS Security: Authentication, Access Control, Access Rights, System Logs

Labs / Practicals:

1. Revision practice of various commands like man, cp, mv, ln, rm, unlink, mkdir, rmdir, etc and many more that were learnt in IT Workshop course and later.
2. Implement two way process communication using pipes.
3. Implement message queue form of IPC
4. Implement shared memory and semaphore form of IPC
5. Simulate the CPU scheduling algorithms - Round Robin, SJF, FCFS, priority
6. Simulate all FIFO Page Replacement Algorithm using C program
7. Simulate all LRU Page Replacement Algorithms using C program
8. Simulate Paging Technique of Memory Management
9. Practice various commands/utilitiessuch as catnl, uniq, tee, pg, comm, cmp, diff, tr, tar, cpio, mount, umount, find, umask, ulimit, sort, grep, egrep,fgrep cut, paste, join, du, df , ps, who, etc and many more.

References:

1. "Operating System Concepts (Tenth Edition)" by Abraham Silberschatz, Peter Baer Galvin, and Greg Gagne
https://os-book.com/OS10/index.html

2. "Operating Systems: Internals and Design Principles (Ninth Edition)" by William Stallings
http://williamstallings.com/OperatingSystems

3. "Modern Operating Systems (Fifth Edition)" by Andrew Stuart Tanenbaum and Herbert Bos
https://dl.acm.org/doi/10.5555/2655363

4. "Operating System Design: The Xinu Approach (Second Edition)" by Douglas Earl Comer
https://www.oreilly.com/library/view/operating-system-design/9781498712439

5. "The Design of the UNIX Operating System" by Maurice J. Bach
https://dl.acm.org/doi/10.5555/8570

6. "Advanced Programming in the UNIX Environment (Third Edition)" by William Richard Stevens and Stephen A. Rago
http://www.apuebook.com/apue3e.html

7. "Beej's Guide to Interprocess Communication" by Brian Hall
https://beej.us/guide/bgipc/pdf/bgipc_a4_c_2.pdf

Course Code	Course Name	L	T	P	Cr
DASH250052	Environmental Science	2	0	0	2

Course Outcomes:

CO1: Understand the importance and dimension of a healthy environment, become environmentally conscious, skilled and responsible in all their actions with a concern for sustainable development.
CO2: Comprehend the significance and issues related to ecosystems, natural resources and bio-diversity and become aware of the need and ways to protect/ preserve them.
CO3: Grasp the issues related to environmental pollution, solid waste management and climate change, and become conscious and proactive in the discharge of their responsibilities towards the environment.
CO4: Become aware and appreciate the values and concerns of environmental movements and policies and the role of communities, and act responsibly on environment-related issues.
CO5: Understand environmental policies, ethics, and community roles in sustainable development.

Module 1:

Introduction to Environmental Studies and Ecosystems
Environmental Studies: Importance and scope, Multidisciplinary nature, Concept of sustainability and sustainable development, Ecosystems : Concept, Structure and Function, Pond and forest ecosystems, Food chains and food webs, Concept of ecological succession, Bio-geographical zones of India, Levels of biological diversity: Genetic, Species, and Ecosystem.

Module 2:

Biodiversity and Conservation
Biodiversity Hotspots with special reference to India, Threats to biodiversity, Conservation of biodiversity: In-situ and Ex-situ, Endangered and endemic species : Concept Afforestation : Social forestry, Agroforestry, Green belt.

Module 3:

Pollution and Waste Management
Air, water, and noise pollution :Causes, effects, and control measures, Climate change, global warming, ozone depletion, acid rain : Impact on humans and agriculture, Solid waste management: biodegradable and non-biodegradable waste.
- Segregation of domestic waste at source
- Impact of plastic on human and animal health

Module 4:

Natural Resources and Disaster Management
Land resources and land-use changes, Land degradation, soil erosion, and desertification, Water: Use and over-exploitation, Water conservation : Rainwater harvesting, Watershed management : Meaning and importance, Energy resources: Renewable and non-renewable, Alternate energy sources, Disaster management – Types and self-protection during disasters Environmental Policy, 2006 : Provisions and importance; Environmental Impact Assessment – Concept; Swachh Bharat Mission Objectives; International agreements , Montreal and Kyoto protocols.

Module 5:

Environmental Policies and Human Role
Population growth: Impact on environment, health, and welfare, Environmental ethics : Religion and cultural role, Environmental movements: Chipko, Narmada Bachao, Silent Valley, Bishnois, Community initiatives : Salu Marada Thimmakka, Sacred Groves (Devarakadu), Environment Protection Act, Biodiversity Act (2002), National Environmental Policy (2006), Environmental Impact Assessment, Swachh Bharat Mission, International agreements : Montreal and Kyoto Protocols.

Labs / Practicals:

NA

References:

1. Asthana, D. K. .Text Book of Environmental Studies. S. Chand Publishing.
2. Basu, M., Xavier, S. . Fundamentals of Environmental Studies, Cambridge University Press, India
3. Basu, R. N. Environment. University of Calcutta, Kolkata
4. Bharucha, E. Textbook of Environmental Studies for Undergraduate Courses. Universities Press, India.
5. De, A.K. Environmental Chemistry, New Age International, .
6. Mahapatra, R., Jeevan, S.S., Das, S. (Eds) . Environment Reader for Universities, Centre for Science and Environment.
7. Masters, G. M., &Ela, W. P. .Introduction to environmental engineering and science. Englewood Cliffs, NJ: Prentice Hall.
8. Odum, E. P., Odum, H. T., & Andrews, J. Fundamentals of ecology. Philadelphia: Saunders.
9. Sharma, P. D., & Sharma, P. D. .Ecology and environment. Rastogi Publications.

Course Code	Course Name	L	T	P	Cr
DCSE250010	Design and Analysis of Algorithms	3	0	2	4

Course Outcomes:

CO1: Understand the fundamental concepts of algorithm design and analysis.
CO2: Implement and analyze algorithms using different design paradigms.
CO3: Have the mathematical foundation in analysis of algorithms
CO4: Understand different algorithmic design strategies like Divide and conquer, Dynamic programming, greedy Algorithms, backtracking.
CO5: Analyze the efficiency of algorithms using time and space complexity theory

Module 1:

Notion of Algorithm, Role of algorithms in computing, Growth of functions- Asymptotic notations, Mathematical Analysis of Non-Recursive and Recursive Algorithms, Brute Force Approaches - Introduction, Selection Sort and Bubble Sort, Sequential Search. Brute Force String Matching.

Module 2:

Divide and Conquer: General Method, Defective Chess Board, Binary Search, Merge Sort, Quick Sort, Heap sort and its performance.

Module 3:

The Greedy Method: The General Method, Amortized Complexity, Knapsack Problem, Job Sequencing with Deadlines, Minimum-Cost Spanning Trees: Prim's Algorithm, Kruskal's Algorithm; Single Source Shortest Paths;
Dynamic Programming:
The General Method, Matrix Chain Multiplication, Longest Common Subsequence, Warshall's Algorithm, Floyd's Algorithm for the All-Pairs Shortest Paths Problem, Single-Source Shortest Paths: General Weights, 0/1 Knapsack, The Traveling Salesperson problem.

Module 4:

Decrease-And-Conquer Approaches, Space-Time Tradeoffs
Decrease and Conquer Approaches: Introduction, Insertion Sort, Depth First Search and Breadth First Search, Topological Sorting, Space-Time Tradeoffs: Introduction, Sorting by Counting, Input Enhancement in String Matching.

Module 5:

Limitations of Algorithmic Power and Coping with them: Lower Bound Arguments, Decision Trees, P, NP, and NP-Complete Problems, Challenges of
Numerical Algorithms;
Coping with Limitations of Algorithmic Power:
Backtracking: n-Queens problem, Hamiltonian Circuit Problem, Subset – Sum Problem. Branch-and-Bound: Assignment Problem, Knapsack Problem, Traveling Salesperson Problem. Approximation Algorithms for NP-Hard Problems, Traveling Salesperson Problem, Knapsack Problem.

Labs / Practicals:

Each algorithm should be implemented in C or Python, and operate on runtime inputs.

01. Implement a linear time greedy algorithm for the fractional knapsack problem.

02. Implement a greedy algorithm for optimal solution of job scheduling with deadlines.

03. Implement Kruskal's algorithm for minimum spanning tree of a connected graph.

04. Implement Prim’s algorithm for minimum spanning tree of a connected graph.

05. Implement Dijkstra’s algorithm for single-source shortest paths (no negative edge).

06. Implement Bellman-Ford algorithm for single-source shortest paths on a digraph.

07. Implement the Euclidean algorithm to find the GCD of two non-negative integers.

08. Implement the binary exponentation algorithm for efficient modular exponenation.

09. Implement binary search, uniform binary search, and interpolation search.

10. Implement quicksort with Hoare partitioning and Lomuto partioning schemes.

11. Implement an adaptive variant of mergesort that runs in linear time for best case.

12. Implement Floyd-Warshall algorithm for all-pairs shortest-path on a weighted graph.

13. Use dynamic programming to find optimal grouping in matrix chain multiplication.

14. Implement a solver for n-queens puzzle, along with a polynomial-time verifier.

15. Implement a polynomial time solver for 2-SAT problem expressed as Krom formula.

References:

1. "Computer Science III" by Chris Bourke
https://cse.unl.edu/~cbourke/ComputerScienceThree.pdf

2. "Algorithms in C (Third Edition)" by Robert Sedgewick
https://www.oreilly.com/library/view/algorithms-in-c/9780768685312

3. "Algorithms Illuminated (Omnibus Edition)" by Timothy Avelin Roughgarden
https://www.algorithmsilluminated.org

4. "The Algorithm Design Manual (Third Edition)" by Steven Sol Skiena
https://www.algorist.com/Algorist_ed2

5. "The Design and Analysis of Computer Algorithms" by Alfred Vaino Aho, John Edward Hopcroft, Jeffrey David Ullman
https://dl.acm.org/doi/10.5555/578775

6. "Computers and Intractability; A Guide to the Theory of NP-Completeness" by Michael Randolph Garey and David Stifler Johnson
https://dl.acm.org/doi/10.5555/578533

Course Code	Course Name	L	T	P	Cr
DOAI250030	Introduction to Artificial Intelligence	3	1	0	4

Course Outcomes:

CO1: Understand AI fundamentals, state-space search, and heuristic problem-solving methods.
CO2: Represent knowledge using logic, rules, and predicate calculus.
CO3: Apply reasoning methods under uncertainty including probabilistic and fuzzy logic approaches.
CO4: Explain key concepts in natural language processing and basics of machine learning.
CO5: Demonstrate the design of expert systems and AI-based planning algorithms.

Module 1:

Introduction and Problems, State Space Search & Heuristic Search Techniques:
What is AI? The AI Problems, The Underlying Assumption, What Is An AI
Techniques, The Level of The Model, Criteria For Success, Some General References, One Final Word, Defining The Problems as a State Space Search, Production Systems, Production Characteristics, Production System Characteristics, And Issues In The Design Of Search Programs, Additional Problems, Generate And-Test, Hill Climbing, Best-First Search, Problem Reduction, Constraint Satisfaction, Means Ends Analysis.

Module 2:

Knowledge representation:
Knowledge Representation Issues -Representations And Mappings, Approaches To Knowledge Representation; Using Predicate Logic - Representation Simple Facts In Logic, Representing Instance And Isa Relationships, Computable Functions And Predicates, Resolution; Representing Knowledge Using Rules -Procedural Versus Declarative Knowledge, Logic Programming, Forward Versus Backward Reasoning.

Module 3:

Reasoning:
Symbolic Reasoning Under Uncertainty -Introduction To Non-monotonic Reasoning, Logics For Non-monotonic Reasoning; Statistical Reasoning -Probability And Bays' Theorem, Certainty Factors And Rule-Base Systems, Bayesian Networks, Dempster-Shafer Theory, Fuzzy Logic; Weak Slot-and-Filler and Strong Slot-and-Filler Structures - Semantic Nets, Frames, Conceptual Dependency, Scripts, CYC

Module 4:

Natural Language Processing and Machine learning and Neural Networks:
Introduction, Syntactic Processing, Semantic Analysis, Semantic Analysis, Discourse and Pragmatic Processing, Spell Checking; Definition and examples of machine learning, supervised learning, unsupervised learning, reinforcement learning, introduction to neural networks.

Module 5:

Expert Systems and AI planning systems:
Representation using domain knowledge, Expert System shell, knowledge acquisition; Definition and examples of Planning Systems, planning as search, operator-based planning, propositional planning.

Labs / Practicals:

NA

References:

1. “Artificial Intelligence” -By Elaine Rich And Kevin Knight (2nd Edition) Tata Mcgraw-Hill
2. “Artificial Intelligence: A Modern Approach”, Stuart Russel, Peter Norvig, PHI
3. Introduction to AI & Expert Sys., Pitterson, D.N

Course Code	Course Name	L	T	P	Cr
DCSE250035	Computer Networks	3	0	2	4

Course Outcomes:

CO1: Understand Network Basics – Learn network models, topologies, and transmission media.
CO2: Analyse Data Link Layer – Evaluate error detection, MAC protocols, and wireless communication.
CO3: Implement Network Layer Functions – Apply IP addressing, subnetting, and routing algorithms.
CO4: Apply Transport & Application Layer Concepts – Compare TCP/UDP and analyse key protocols.
CO5: Demonstrate Security & Emerging Technologies – Identify threats, implement cryptography, explore SDN, IoT, and cloud networking.

Module 1:

Introduction to Computer Networks Overview of Computer Networks, Network Types: LAN, WAN, MAN, PAN, Network Topologies: Bus, Star, Ring, Mesh, Hybrid, OSI Model & TCP/IP Model – Layers and Functions, Transmission Media: Wired & Wireless Technologies, Switching Techniques: Circuit, Packet, and Message Switching

Module 2:

Data Link Layer & MAC Protocols
Error Detection and Correction: Parity, Checksum, CRC, Hamming Code, Flow Control Mechanisms: Stop-and-Wait, Sliding Window Protocol, MAC Protocols: ALOHA, CSMA/CD, CSMA/CA, Ethernet Standards (IEEE 802.3), VLANs, Spanning Tree Protocol, Wireless LAN (Wi-Fi), Bluetooth, and Mobile Networks

Module 3:

Network Layer & Routing Algorithms
IPv4 and IPv6 Addressing, Subnetting, and CIDR, Routing Algorithms: Distance Vector, Link-State, and Hybrid Routing, Routing Protocols: RIP, OSPF, EIGRP, and BGP, Congestion Control Mechanisms and Quality of Service (QoS), Network Address Translation (NAT) and DHCP

Module 4:

Transport Layer & Application Layer Protocols
TCP and UDP: Connection Establishment, Flow Control, and Error Control, Congestion Control: TCP Reno, TCP Tahoe, and Fair Queuing, Application Layer Protocols: HTTP, HTTPS, FTP, SMTP, POP3, IMAP, DNS and URL Resolution, Peer-to-Peer Networks, Security in Transport & Application Layers: TLS, SSL, Firewalls

Module 5:

Network Security and Emerging Trends
Cryptography Fundamentals: Symmetric and Asymmetric Encryption, Authentication and Digital Signatures, Network Security Attacks: DoS, DDoS, Phishing, Man-in-the-Middle, Wireless and IoT Security Challenges, Software-Defined Networking (SDN) and Cloud Networking, Introduction to 5G, Edge Computing, and Quantum Networking

Labs / Practicals:

Preparing

References:

1. "Computer Networking: A Top Down Approach (Eighth Edition)" by James F. Kurose and Keith W. Ross

https://gaia.cs.umass.edu/kurose_ross/about.php

2. "An Introduction to Computer Networks (Second Edition)" by Peter Lars Dordal
https://intronetworks.cs.luc.edu/current2/ComputerNetworks.pdf

3. "Computer Networking: Principles, Protocols and Practice (Third Edition)" by Olivier Bonaventure
https://github.com/cnp3/ebook/releases/download/draft-3rd/CNP3-2021.pdf

4. "Computer Networks: A Systems Approach (Sixth Edition)" by Larry L. Peterson and Bruce S. Davie
https://github.com/SystemsApproach/book/releases/download/v6.1/book.pdf

5. "Computer Networks (Fifth Edition)" by Andrew Stuart Tanenbaum and David J. Wetherall
https://www.oreilly.com/library/view/computer-networks-fifth/9780133485936

6. "Internetworking with TCP/IP: Principles, Protocols, and Architecture (Sixth Edition)" by Douglas Earl Comer
https://www.oreilly.com/library/view/internetworking-with-tcpip/9780137464197

7. "UNIX Network Programming: Volume 1 (Third Edition)" by William Richard Stevens, Bill Fenner, and Andrew M. Rudoff
https://www.oreilly.com/library/view/the-sockets-networking/0131411551

8. "Computer and Network Organization: An Introduction" by Maarten van Steen and Henk Sips
https://www.distributed-systems.net/index.php/books/computer-and-network-organization

9. "Beej's Guide to Network Programming" by Brian Hall
https://beej.us/guide/bgnet/pdf/bgnet_a4_c_2.pdf

Course Code	Course Name	L	T	P	Cr
DASH250026	Design Thinking	1	0	2	2

Course Outcomes:

CO1 : Understand the core principles and process of design thinking including empathy, ideation, and prototyping.
CO2 : Apply user-centric research methods to identify and frame real-world problems.
CO3 : Generate and evaluate innovative ideas through brainstorming and collaborative design activities.
CO4 : Build and test low-fidelity prototypes to explore solution possibilities.
CO5 : Present and communicate design solutions effectively using appropriate tools and storytelling techniques.

Module 1:

Introduction to Design Thinking
Definition and Importance of Design Thinking, Comparison with Traditional Problem Solving, Mindsets for Innovation, Stanford d.school Design Process, Applications in Engineering, Business, and Social Sectors.

Module 2:

Empathize – Understanding Users
Importance of Empathy, Observation and Interview Techniques, User Persona Creation, Empathy Mapping, Journey Maps, Capturing Insights from Field Research.

Module 3:

Define and Ideate
Problem Definition and Framing: Point of View (POV) Statements, Ideation Techniques (Brainstorming, SCAMPER, Mind Mapping), Idea Evaluation and Selection Criteria.

Module 4:

Prototype and Test
Prototyping Tools and Techniques, Rapid Prototyping, Paper Prototypes, Role-Playing, Usability Testing, Feedback Integration and Iteration.

Module 5:

Communication and Implementation
Storyboarding, Pitching Ideas, Design Thinking for Entrepreneurship, Real-life Case Studies, Ethical and Sustainable Design Considerations.

Labs / Practicals:

1. Conduct user interviews and observations to develop empathy maps.
2. Create user personas and problem statements.
3. Brainstorm solutions for a chosen problem and organize ideas using mind maps.
4. Develop low-fidelity prototypes for the selected ideas.
5. Test prototypes with users and gather feedback for refinement.
6. Present project using storyboards and pitch decks.
7. Group project: Apply the design thinking process to a real-world challenge and document the journey.
8. Use tools like Canva, Miro, or Figma for visualizing and presenting ideas.
9. Role-play-based usability tests.
10. Peer review and reflection journal on the design process.

References:

1. Tim Brown, Change by Design, Harper Business.
2. Rolf Faste, Design Thinking Methodology, Stanford University Materials.
3. Jeanne Liedtka & Tim Ogilvie, Designing for Growth: A Design Thinking Toolkit for Managers, Columbia

Business School Publishing.
4. Nigel Cross, Design Thinking: Understanding How Designers Think and Work, Berg Publishers.
5. Plattner, Meinel, Leifer, Design Thinking: Understand – Improve – Apply, Springer.

Course Code	Course Name	L	T	P	Cr
DCSE250107	Principles of Compiler Design	3	0	2	4

Course Outcomes:

CO1: Understand the various phases of a compiler and their roles in program translation.
CO2: Design and implement lexical analyzers and parsers using formal grammar and tools like LEX and YACC.
CO3: Apply syntax-directed translation techniques and manage symbol tables for semantic analysis.
CO4: Generate intermediate code and understand run-time environments and memory allocation strategies.
CO5: Analyze and apply code optimization techniques to improve the performance of the generated code.
CO6: Develop basic compilers or interpreters using compiler construction principles and tools

Module 1:

Introduction & Lexical Analysis: Overview of compilers: phases and architecture, Role and functions of lexical analyzer, Token specification, recognition, pattern matching, Finite automata: DFA, NFA, Tools: Lex/Flex and writing a hand crafted lexer

Module 2:

Syntax Analysis: Context Free Grammars (CFG), parse trees, ambiguity, Top down parsing: FIRST/FOLLOW, LL(1), recursive descent, Bottom up parsing: operator-precedence, LR(0), SLR(1), LR(1), LALR(1), Parser generators: YACC/Bison, error recovery strategies

Module 3:

Semantic Analysis & Syntax-Directed Translation: Syntax-Directed Definitions (SDDs): attributes, evaluation order, Types of SDDs: S attributed, L attributed, Semantic actions during parsing, Symbol tables, scope handling, type checking, semantic error handling

Module 4:

Intermediate Code Generation: Intermediate representations: three-address code, quadruples, triples, abstract syntax trees, SSA, Code translation: expressions, control flow, boolean logic, arrays, Backpatching, flow graphs, control flow translation

Module 5:

Runtime Environments & Code Generation: Activation records, stack and heap allocation, Accessing non-local data (nested procedures), Machine code generation: instruction selection, run time storage management, basic blocks
Register allocation strategies and graph coloring.
Code Optimization: Machine-independent optimizations: local and global (SSA, DAGs), Dataflow analysis, loop optimization, peephole techniques, Machine-dependent optimization fundamentals

Labs / Practicals:

1. Experiment No.1: Lexical Analyzer using LEX Tool: Design a lexical analyzer that identifies tokens such as identifiers, keywords, numbers, and operators.
2. Experiment No.2: Syntax Analyzer using YACC Tool: Implement a parser for arithmetic expressions using grammar rules in YACC and lexical rules in LEX.
3. Experiment No.3: Tokenization of Source Code (Manual or with LEX): Manually tokenize or use LEX to tokenize a given piece of C/C++ source code.
4. Experiment No.4: Implementation of a Recursive Descent Parser: Write a program for parsing a simple grammar using recursive descent parsing technique.
5. Experiment No.5: Construction of FIRST and FOLLOW Sets: Develop a program to compute the FIRST and FOLLOW sets for a given grammar.

6. Experiment No.6: Implementation of LL(1) Parsing Table: Construct the LL(1) parsing table from FIRST and FOLLOW sets and parse input strings accordingly.
7. Experiment No.7: Intermediate Code Generation (Three-Address Code): Generate three-address code (TAC) for assignment statements and arithmetic expressions.
8. Experiment No.8: Backpatching for Boolean Expressions and Control Structures: Simulate backpatching for control flow constructs like if, while, etc., during intermediate code generation.
9. Experiment No.9: Symbol Table Implementation: Implement a symbol table with insertion, lookup, and deletion operations for identifiers.
10. Experiment No.10: Simple Code Optimization Techniques: Apply basic optimization techniques such as constant folding, dead code elimination, or strength reduction on intermediate code.

References:

1. Aho, Sethi, Ullman: Compilers: Principles, Techniques & Tools (Pearson) studocu.com+4id.scribd.com+4en.wikipedia.org+4
2. Louden: Compiler Construction (Cengage) es.scribd.com+3scribd.com+3studocu.com+3
3. Appel: Modern Compiler Implementation
4. Tremblay & Sorrenson: Theory & Practice of Compiler Writing

Course Code	Course Name	L	T	P	Cr
DCSE250002	Blockchain and Cryptocurrency	3	1	0	4

Course Outcomes:

CO1: Explain the foundational principles of blockchain technology, including its structure, working mechanism, and consensus algorithms such as Proof-of-Work and Proof-of-Stake.
CO2: Analyze and interpret cryptocurrency systems like Bitcoin and Ethereum, including transaction validation, mining, wallets, and double-spending prevention.
CO3: Design and develop smart contracts and deploy decentralized applications (dApps) using platforms such as Ethereum and Hyperledger Fabric.
CO4: Identify and evaluate key security threats in blockchain systems and apply cryptographic techniques to enhance system integrity and confidentiality.
CO5: Apply blockchain concepts to real-world use cases in fields such as IoT, supply chain, and finance, and demonstrate awareness of emerging trends like NFTs and DeFi.

Module 1:

Introduction to Blockchain: Public ledgers & their structure, transaction flow Blocks, hashing, hash pointers, Merkle Trees, Consensus mechanisms: proof-of-work, longest-chain rule, Blockchain 2.0 and permissioned ledgers.

Module 2:

Cryptocurrency & Transaction Processing: Cryptocurrency basics (e.g., Bitcoin transactions), Transaction validation logic, UTXO models, Double-spend protection, scripting languages

Module 3:

Smart Contracts & Blockchain Platforms: Smart contract mechanics & use-cases, Hyperledger Fabric, Ethereum ecosystem, Tools and environments for contract development

Module 4:

Blockchain Security: Security issues: 51% attacks, Sybil, network attacks
Consensus vulnerabilities & mitigation strategies, Privacy techniques: mixing, zero-knowledge proofs

Module 5:

Blockchain Applications & Trends: Blockchain-as-a-Service (BaaS) offerings, Other applications: IoT integration, supply chain, healthcare, Emerging trends: DeFi, NFTs, decentralized identity

Labs / Practicals:

N/A

References:

1. Bashir, I. (2023). Mastering Blockchain: Unlocking the power of cryptocurrencies, smart contracts, and decentralized applications (4th ed.). Packt Publishing. ISBN: 9781803241065
2. Antonopoulos, A. M. (2017). Mastering Bitcoin: Unlocking digital cryptocurrencies (2nd ed.). O'Reilly Media. ISBN: 9781491954386
3. Drescher, D. (2017). Blockchain Basics: A non-technical introduction in 25 steps. Apress. ISBN: 9781484226032
4. The Linux Foundation. (n.d.). Hyperledger Fabric Documentation. Retrieved from https://hyperledger-fabric.readthedocs.io/
5. Ethereum Foundation. (n.d.). Ethereum Developer Documentation. Retrieved from https://ethereum.org/en/developers/docs/

Course Code	Course Name	L	T	P	Cr
DCSE250043	Cryptography and Network Security	3	0	2	4

Course Outcomes:

CO1: Explain security fundamentals, including the CIA Triad and various cyber threats.
CO2: Describe security services and mechanisms, such as authentication and encryption.
CO3: Understand the history and evolution of cryptography, including classical techniques.
CO4: Recognize the role of cryptanalysis in breaking cryptographic systems.
CO5: Apply mathematical foundations, including number theory, modular arithmetic, and random number generation, in cryptographic algorithms.

Module 1:

Introduction to Cryptography & Security Concepts
Basics of Security, Security Goals: Confidentiality, Integrity, Availability (CIA), Threats and Attacks: Passive vs. Active Attacks, Security Services & Mechanisms, Cybersecurity Principles, Introduction to Cryptography, History & Evolution of Cryptography, Classical Cryptographic Techniques: Caesar Cipher, Vigenère Cipher, Playfair Cipher, Modern Cryptography & Cryptanalysis
Mathematical Foundations of Cryptography: Number Theory Basics: Prime Numbers, Modular Arithmetic, Euler's Theorem, Greatest Common Divisor (GCD), Fermat's Theorem, Random Number Generation

Module 2:

Symmetric & Asymmetric Cryptography
Symmetric Key Cryptography, Block Ciphers vs. Stream Ciphers, Data Encryption Standard (DES) & Triple DES, Advanced Encryption Standard (AES), Modes of Operation: ECB, CBC, CFB, OFB, CTR, Asymmetric Key Cryptography, Public Key Cryptosystems: RSA Algorithm, Key Exchange Mechanisms: Diffie-Hellman Key Exchange, Elliptic Curve Cryptography (ECC), Comparison of Symmetric & Asymmetric Cryptography

Module 3:

Hash Functions, Digital Signatures & Authentication Cryptographic Hash Functions, Properties of Hash Functions, Common Hash Algorithms: MD5, SHA-1, SHA-256, SHA-3, Message Authentication, Message Authentication Codes (MAC), HMAC (Hash-based Message Authentication Code), Digital Signatures & Certificates, RSA Digital Signature Algorithm, Digital Certificates & Public Key Infrastructure (PKI), SSL/TLS Encryption and HTTPS Authentication Mechanisms, Password-Based Authentication, Multi-Factor Authentication (MFA), Biometric Authentication

Module 4:

Network Security & Secure Protocols
Network Security Fundamentals, Security in OSI & TCP/IP Models, Firewalls and Intrusion Detection Systems (IDS/IPS), Virtual Private Networks (VPNs) & IPsec, Secure Network Protocols, Secure Email: PGP & S/MIME, Secure Web Transactions: HTTPS, SSL/TLS
Wireless Security: WPA, WPA2, WPA3, Network Attacks & Defense Mechanisms, Denial of Service (DoS) & Distributed DoS (DDoS) Attacks, Man-in-the-Middle (MITM) & Eavesdropping, SQL Injection & Cross-Site Scripting (XSS)

Module 5:

Emerging Trends & Cybersecurity Applications
Blockchain & Cryptography, Role of Cryptography in Blockchain, Bitcoin & Ethereum Security, Cloud Security & Zero Trust Security Model, Cloud Encryption Mechanisms, Zero Trust Architecture & Access Control
Cybersecurity & Ethical Hacking, Introduction to Ethical Hacking, Penetration Testing Basics, Legal & Ethical

Issues in Cybersecurity, Future Trends in Cryptography & Security

Labs / Practicals:

Preparing

References:

1. "The Joy of Cryptography" by Michael Rosulek
https://joyofcryptography.com/pdf/book.pdf

2. "Cryptography and Computer Security" by Chris Bourke
https://cse.unl.edu/~cbourke/cryptoNotes.pdf

3. "Applied Cryptography (20th Anniversary Edition)" by Bruce Schneier
https://www.schneier.com/books/applied-cryptography

4. "Cryptography and Network Security (Eighth Edition)" by William Stallings
http://williamstallings.com/Cryptography/Crypto8e-Student

5. "Cracking Codes With Python: An Introduction To Building And Breaking Ciphers" by Al Sweigart
https://inventwithpython.com/cracking

6. "Yet Another Introductory Number Theory Textbook (Cryptology Emphasis Version)" by Jonathan Adam Poritz
https://www.poritz.net/jonathan/share/yaintt.pdf

Course Code	Course Name	L	T	P	Cr
DASH250056	Financial Accounting & Management	2	0	0	2

Course Outcomes:

CO1: Describe fundamental accounting principles and management concepts. (Level 1 – Remember)
CO2: Prepare basic accounting statements such as journal entries, ledger, and trial balance. (Level 3 – Apply)
CO3: Analyze financial statements using ratio analysis and cost analysis techniques. (Level 4 – Analyze)
CO4: Explain budgeting, working capital management, and financial planning in organizations. (Level 2 – Understand)
CO5: Apply financial decision-making concepts in real-world business or startup contexts. (Level 3 – Apply)

Module 1:

Introduction to Accounting and Management (6 Hours)
Need and scope of accounting, principles and concepts of accounting, objectives and functions of management, differences between accounting and management, users of accounting information, relevance of financial literacy for computer professionals and entrepreneurs.

Module 2:

Accounting Process (6 Hours)
Double-entry system of accounting, journal entries, posting to ledger; preparation of trial balance, understanding of cash book and petty cash book, simple problems on journal and ledger for service and product-based businesses.

Module 3:

Financial Statements and Analysis (6 Hours)
Preparation of trading account, profit and loss account, and balance sheet (with adjustments), Basics of financial statement interpretation, Ratio analysis: liquidity, profitability, solvency, turnover ratios, Use of spreadsheet tools (Excel) for analysis.

Module 4:

Cost and Budgetary Control (6 Hours)
Types of costs (fixed, variable, direct, indirect), marginal costing and break-even analysis, preparation of budgets: cash, production, flexible, budgetary control techniques, basics of working capital management.

Module 5:

Financial Management and ICT Integration (6 Hours)
Time value of money, capital budgeting basics, sources of finance, digital tools and ERPs (Tally, Zoho, SAP basics), GST basics and e-invoicing, case studies on fintech, budgeting in startups, digital payment ecosystems in India.

Labs / Practicals:

N/A

References:

1.S.N. Maheshwari & S.K. Maheshwari – An Introduction to Accountancy, Vikas Publishing.
2.T.S. Grewal – Double Entry Book Keeping, Sultan Chand & Sons.
3.M.Y. Khan & P.K. Jain – Financial Management: Text, Problems and Cases, McGraw Hill.

4.R.P. Rustagi – Fundamentals of Financial Management, Taxmann Publications.
5.Dr. Jawahar Lal – Accounting for Management, Himalaya Publishing.
6.NCERT Bookkeeping and Accountancy (Class XI & XII) – Foundation Concepts.

Course Code	Course Name	L	T	P	Cr
DASH250091	Professional Values and Ethics	2	0	0	2

Course Outcomes:

CO1: Describe profession, professionalism, its challenges, effective communication, and the role of regulatory bodies and professional organizations related to engineering.
CO2: Analyze professional values such as integrity, responsibility, respect, fairness, and accountability, and demonstrate their application in professional practice.
CO3: Apply knowledge of ethics and bioethics in ethical decision-making in collaboration with professional teams.
CO4: Evaluate responsibilities related to safety, risk management, and intellectual property in professional practice.
CO5: Examine global ethical issues and professional responsibilities in technological and research contexts.

Module 1:

Values (4 hrs)
Definition and characteristics of values, value clarification, personal and professional values, professional socialization: integration of professional and personal values, importance of professional values in engineering and technology, core values – integrity, honesty, trust, respect, responsibility, fairness, accountability, ethical conduct in workplace relationships and decision-making.

Module 2:

Engineering Ethics (8 hrs)
Senses of engineering ethics, variety of moral issues, types of inquiry, moral dilemmas, moral autonomy, Kohlberg's theory, Gilligan's theory, consensus and controversy, models of professional roles, theories about right action – self-interest, customs and religion, application of ethical theories, valuing time, cooperation, commitment, case studies.

Module 3:

Engineering as Social Experimentation (8 hrs)
Engineering as social experimentation, framing the problem, determining the facts, codes of ethics, clarifying concepts, application issues, common ground, general principles – utilitarian thinking and respect for persons, case studies.

Module 4:

Engineers Responsibility for Safety and Risk (8 hrs)
Safety and risk, assessment of safety and risk, risk-benefit analysis, reducing risk, safety and the engineer – designing for safety, Intellectual Property Rights (IPR).

Module 5:

Global issues (8 hrs)
Globalization, cross-cultural issues, environmental ethics, computer ethics, computers as instruments of unethical behavior, computers as objects of unethical acts, autonomous systems, computer codes of ethics, weapons development, ethics and research, analyzing ethical problems in research, case studies.

Labs / Practicals:

N/A

References:

1. M. Govindarajan, S. Natarajan, V.S. Senthil Kumar – Engineering Ethics Includes Human Values, PHI Learning Pvt. Ltd, 2009.
2. Charles E. Harris, Michael S. Pritchard, Michael J. Rabins – Engineering Ethics, Cengage Learning, India Edition, 2009.
3. Mike W. Martin, Roland Schinzinger – Ethics in Engineering, Tata McGraw Hill, 2003.
4. A.R. Aryasri, Dharanikota Suyodhana – Professional Ethics and Morals, Maruthi Publications.
5. A. Alavudeen, R. Kalil Rahman, M. Jayakumaran – Professional Ethics and Human Values, Laxmi Publications.
6. D.R. Kiran – Professional Ethics and Human Values.
7. P.S.R. Murthy – Indian Culture, Values and Professional Ethics, BS Publications

Course Code	Course Name	L	T	P	Cr
DCSE250004	Advanced Data Structures and Algorithms	3	0	2	4

Course Outcomes:

CO1: Analyze the performance of basic data structures and sorting/searching techniques.
CO2: Implement and apply advanced trees, heaps, and hash tables to solve complex problems.
CO3: Solve problems using graph algorithms and greedy strategies.
CO4: Design algorithms using divide and conquer, backtracking, and dynamic programming.
CO5: Develop efficient and scalable solutions using appropriate algorithmic techniques.

Module 1:

Review of Fundamental Concepts
Pointers and Dynamic Memory Allocation, Arrays, Dynamically Allocated Arrays, Structures, Algorithm Specification, Data Abstraction, Performance Analysis, Performance Measurement, Time and space complexity

Module 2:

Basic Data Structures and Advanced Trees
Stacks, Queues, Linked Lists, Hash Tables, Searching and Sorting Techniques: Linear Search, Bubble Sort, Selection Sort, Insertion Sort, Binary Trees, Binary Tree Traversals, Applications of Binary Trees, Threaded Binary Trees, Binary Search Trees, Advanced Trees: AVL Trees: Rotations, insertion and deletion, Red-Black Trees, Splay Trees, B Trees, B+ Trees, Binary Heap: Min/Max heap operations, Priority Queue applications.

Module 3:

Divide and Conquer, Backtracking, String Matching algorithms
Merge Sort, Quick Sort, Binary Search, Backtracking: N-Queens, Graph Coloring, String matching algorithms: KMP, Rabin-Karp

Module 4:

Graph Algorithms
Graph Representations: Adjacency list, Adjacency Matrix, BFS, DFS, Shortest Path: Dijkstra’s, Greedy algorithms: Activity Selection, Huffman Coding Minimum Spanning Tree: Prim’s and Kruskal’s Algorithms

Module 5:

Dynamic Programming
Basic concepts: Overlapping subproblems, optimal substructure, 0/1 Knapsack, Longest Common Subsequence, Matrix Chain Multiplication

Labs / Practicals:

Preparing

References:

1. "Introduction to Algorithms (Fourth Edition)" by Thomas H. Cormen, Charles Eric Leiserson, Ronald Linn Rivest, Clifford Seth Stein
https://mitpress.mit.edu/9780262046305/introduction-to-algorithms

2. "A Second Course in Algorithms" by Timothy Avelin Roughgarden
https://timroughgarden.org/w16/l

3. "Algorithm Design" by Jon Michael Kleinberg and Eva Tardos
https://dl.acm.org/doi/10.5555/1051910

4. "Computational Intractability: A Guide to Algorithmic Lower Bounds" by Erik D. Demaine, William Ian Gasarch, and Mohammad Taghi Hajiaghayi
https://hardness.mit.edu/drafts/2025-02-02.pdf

5. "The Design and Analysis of Algorithms" by Dexter Campbell Kozen
https://www.cs.cornell.edu/kozen/Papers/daa.pdf

6. "The Art of Computer Programming" by Donald Ervin Knuth
https://www-cs-faculty.stanford.edu/~knuth/taocp.html

Course Code	Course Name	L	T	P	Cr
DCSA250071	Software Engineering	3	1	0	4

Course Outcomes:

CO1: Understand Software Engineering Concepts – Explain core principles, ethical responsibilities, software processes, and critical system properties.
CO2: Analyze Software Requirements – Differentiate functional and non-functional requirements and apply requirements engineering processes.
CO3: Apply System Modeling & Project Management – Develop system models and manage software projects, including planning, scheduling, and risk management.
CO4: Design & Develop Software – Implement architectural and object-oriented design, agile methodologies, and software evolution strategies.
CO5: Perform Software Testing & Cost Estimation – Conduct verification, validation, software testing, team management, and cost estimation techniques

Module 1:

Overview - Introduction, FAQs about software engineering, Professional and ethical responsibility; Socio-technical systems - Emergent system properties, Systems engineering; Critical systems and Software processes - Critical systems, A simple safety-critical system, System dependability, Availability and reliability; Software processes - Software process models, Process iteration, Process activities, The Rational Unified Process, Computer-Aided Software Engineering.

Module 2:

Requirements - Software requirements, Functional and non-functional requirements, User requirements, System requirements, Interface specification, The software requirements document; Requirements engineering processes - Feasibility studies, Requirements elicitation and analysis, Requirements validation, Requirements management.

Module 3:

System models and Project Management - System models, Context models, Behavioural models, Data models, Object models, Structured methods; Project management - Management activities, Project planning, Project scheduling, Risk management

Module 4:

Software Design - Architectural design, Architectural design decisions, System organisation, Modular decomposition styles, Control styles; Object-oriented design - Objects and object classes, An object-oriented design process, Design evolution; Development - Rapid software development, Agile methods, Extreme programming, Rapid application development; Software evolution - Program evolution dynamics, Software maintenance, Evolution processes, Legacy system evolution.

Module 5:

Verification and validation - Verification and validation, Planning verification and validation, Software inspections, Automated static analysis, Verification and formal methods; Software testing - System testing, Component testing, Test case design, Test automation; Management - Managing people, Selecting staff, Motivating people, Managing groups, The People Capability Maturity Model; Software cost estimation, Software productivity, Estimation techniques, Algorithmic cost modelling, Project duration and staffing

Labs / Practicals:

N/A

References:

1.Ian Sommerville: Software Engineering, 8th Edition, Pearson Education, 2007.
2.Roger S. Pressman: Software Engineering - A Practitioner’s Approach, 7th Edition, Tata McGraw Hill, 2007.
3.Pankaj Jalote: An Integrated Approach to Software Engineering, Wiley India, 2009.

Course Code	Course Name	L	T	P	Cr
DCSE250073	Ethical Hacking	2	1	2	4

Course Outcomes:

CO1: Explain the ethical hacking process and distinguish between various attack vectors, hacker types, and phases of penetration testing.
CO2: Apply reconnaissance techniques such as footprinting, scanning, and enumeration to gather information about target systems.
CO3: Demonstrate the ability to exploit system and network vulnerabilities using tools and techniques like password cracking, sniffing, and privilege escalation.
CO4: Identify and exploit web application and wireless network vulnerabilities using ethical hacking tools and analyze the associated risks.
CO5: Prepare and document penetration testing reports while adhering to legal and ethical standards of cybersecurity practice.

Module 1:

Ethical Hacking Process & Reconnaissance: Ethical hacking methodology
Hacker mindset, anonymity, legality, ethics, Information gathering, techniques (active & passive), Physical security weaknesses, Internal & external testing, Penetration test planning and reporting

Module 2:

Social Engineering & Password Attacks: Types of social engineering, Phishing, pretexting, tailgating, Password cracking techniques (brute force, dictionary), Privilege escalation, Countermeasures

Module 3:

Sniffing & Network Scanning: Packet sniffing methods, ARP spoofing, DNS/IP sniffing, Network scanning tools and techniques

Module 4:

System and Wireless Hacking: System hacking: vulnerability discovery, keystroke logging, remote exploits, Wireless hacking: WEP/WPA weakness, wireless traffic capturing, cracking

Module 5:

Web & Application Hacking: Web-based attacks: SQL Injection, XSS, file inclusion, Use of tools like Nikto, Burp Suite, Metasploit, Wireless, Web app vulnerability exploitation and reporting

Labs / Practicals:

1. Experiment No.1: Footprinting and Reconnaissance: Perform passive and active information gathering on a target website or IP using tools like whois, nslookup, the Harvester, and Recon-ng.
2. Experiment No.2: Scanning Networks: Use Nmap or Zenmap to discover open ports, services, and operating systems on a target machine.
3. Experiment No.3: Enumeration: Enumerate usernames, shared resources, and services using tools like NetBIOS, SNMPwalk, or enum4linux.
4. Experiment No.4: System Hacking – Password Cracking: Perform password cracking using tools like John the Ripper, Hydra, or Hashcat.
5. Experiment No.5: Malware Analysis (Static and Dynamic): Analyze a sample malware in a sandbox environment using tools like PEStudio, Process Monitor, or Wireshark.
6. Experiment No.6: Sniffing and Spoofing: Capture and analyze network packets using Wireshark or tcpdump; perform ARP spoofing using Ettercap or Cain and Abel.
7. Experiment No.7: Session Hijacking: Demonstrate session hijacking using tools like Ettercap or browser

cookie stealing techniques in a controlled environment.
8. Experiment No.8: Denial of Service (DoS) Attack: Simulate a DoS attack using tools like Hping3, LOIC, or custom scripts to understand impact and prevention.
9. Experiment No.9: Web Server Footprinting and Vulnerability Scanning: Perform vulnerability scanning on a web server using tools like Nikto, OpenVAS, or Nessus.
10. Experiment No.10: SQL Injection Attack: Demonstrate SQL injection on a test web application using tools like sqlmap or manual techniques.
11. Experiment No11: Cross-Site Scripting (XSS) Attack: Perform stored and reflected XSS attacks on a vulnerable web application like DVWA (Damn Vulnerable Web Application).
12. Experiment No.12: Cross-Site Request Forgery (CSRF) Attack: Simulate CSRF attacks using web applications like DVWA or custom scripts to understand defense mechanisms.
13. Experiment No.13: Privilege Escalation: Perform privilege escalation in a Linux or Windows environment using local exploits or misconfigurations.
14. Experiment No.14: Wireless Hacking: Crack WPA/WPA2 passwords using aircrack-ng suite and perform wireless network analysis.
15. Experiment No.15: Creating a Penetration Testing Report: Conduct a full penetration test on a test system or network and document findings, tools used, vulnerabilities, and mitigation strategies.

References:

1. Harris, S., Harper, A., Eagle, C., & Ness, J. (Year). Gray Hat Hacking: The Ethical Hacker’s Handbook. TMH.
2. Erickson, J. (Year). Hacking: The Art of Exploitation. SPD.
3. Beaver, K. (2013). Hacking for Dummies (3rd ed.). Wiley.
4. Additional: Shon Harris et al., Gray Hat Hacking (TMH); Thomas Mathew, Ethical Hacking (OSB).

Course Code	Course Name	L	T	P	Cr
DOEE250131	Industrial Robotics	2	0	4	4

Course Outcomes:

CO1: Explain the structure, components, and applications of industrial robots.
CO2: Apply kinematic and dynamic principles to robot modeling and motion planning.
CO3: Select and integrate appropriate sensors, actuators, and end-effectors.
CO4: Program and simulate industrial tasks using robotic platforms.
CO5: Discuss recent trends in industrial robotics and suggest practical use-cases.

Module 1:

Introduction to Industrial Robotics
- History and Evolution of Robotics
- Types and Classifications of Robots
  - Based on structure (Cartesian, SCARA, Articulated, Delta, etc.)
  - Based on control (servo, non-servo)
  - Based on applications
- Specifications: Payload, DOF, repeatability, accuracy, work envelope
- Components of a Robot: Manipulator, controller, sensors, actuators, end-effectors
- Applications in manufacturing, healthcare, agriculture, etc.
- Industrial safety standards in robotics (ISO 10218, ANSI/RIA)

Module 2:

Robot Kinematics and Workspace Analysis
•Coordinate Systems and Transformations
•Denavit-Hartenberg (DH) Parameters
•Forward Kinematics
•Inverse Kinematics (analytical and numerical approaches)
•Jacobian matrix: velocity and force transformation
•Workspace and singularity analysis

Module 3:

Dynamics and Control of Robots
•Robot Dynamics:
  o Lagrangian and Newton-Euler formulations
•Joint space and Cartesian space dynamics
•Trajectory Planning:
  o Joint interpolation, Cartesian interpolation
•Control Systems:
  o PID control
  o Force/torque control
  o Hybrid position/force control

Module 4:

Sensors, Actuators, and End-Effectors
• Types of actuators:
  o Electric motors (DC, Servo, Stepper)
  o Hydraulic and pneumatic systems
• End-Effectors:
  o Grippers, tools, welding guns, suction cups

  o End-effector design principles
• Robotic Sensors:
  o Position and velocity sensors
  o Proximity sensors
  o Force and torque sensors
  o Vision sensors (2D and 3D)
• Signal conditioning and data acquisition basics

Module 5:

Programming, Integration & Emerging Trends
• Robot Programming Approaches:
  o Manual (lead-through)
  o Online (Teach pendant)
  o Offline programming (RoboDK, MATLAB)
• Introduction to industrial robot languages (RAPID, KRL, VAL, etc.)
• PLC and SCADA integration basics
• Human-Robot Interaction (HRI)
• Introduction to Collaborative Robots (Cobots)
• Vision-based robotics and AI integration
• Trends: Industry 4.0, IoT with Robotics, Digital Twins

Labs / Practicals:

1. Introduction to Industrial Robots and Safety Practices
Objective: Understand robot types, configurations, and lab safety protocols.

2. Study of Robot Components and Configurations
Objective: Identify joints, links, actuators, and sensors of a robotic arm.

3. Teach Pendant Programming for Basic Robot Movements
Objective: Use teach pendant to control and record robot positions.

4. Pick and Place Operation using a Robotic Arm
Objective: Program the robot to execute pick and place tasks.

5. Tool Center Point (TCP) and Work Coordinate Calibration
Objective: Calibrate robot tool frames and coordinate systems.

6. Simulation of Forward and Inverse Kinematics using MATLAB / RoboDK
Objective: Analyze and visualize robotic motion in a virtual environment.

7. Trajectory Planning and Smooth Motion Execution
Objective: Develop and test motion trajectories using interpolation techniques.

8. Integration of IR/Proximity Sensors with Robot for Feedback Control
Objective: Use sensor data to control or interrupt robot actions.

9.Vision-Based Object Detection using Camera and OpenCV
Objective: Implement vision system for identifying and tracking objects.

10. Mini Project: Automation of an Industrial Task (e.g., sorting, palletizing)
Objective: Apply programming, sensing, and actuation skills to solve a real-world problem.

References:

1. Mikell P. Groover, "Industrial Robotics: Technology, Programming and Applications", McGraw-Hill
2. John J. Craig, "Introduction to Robotics: Mechanics and Control", Pearson

3. R.K. Mittal & I.J. Nagrath, “Robotics and Control”, Tata McGraw Hill
4. Saeed B. Niku, “Introduction to Robotics: Analysis, Control, Applications”, Wiley
5. Deb & Deb, “Robotics Technology and Flexible Automation”, McGraw-Hill

Course Code	Course Name	L	T	P	Cr
DCSE250007	Cyber Law and IPR	3	0	2	4

Course Outcomes:

CO1: Identify statutory, regulatory, constitutional, and organizational laws that affect the information technology professional.
CO2: Categorize case law and common law to current legal dilemmas in the technology field.
CO3: Outline the primary forms of intellectual property rights.
CO4: Compare the different forms of intellectual property protection in terms of their key differences and similarities.
CO5: Analyze the effects of intellectual property rights on society as a whole.

Module 1:

Introduction: Overview of Computer and Web Technology, Need for Cyber Law, Cyber Jurisprudence at International and Indian Level, Jurisdictional Aspects in CyberLaw Issues of jurisdiction in cyberspace, Types of jurisdiction, Minimum Contacts Theory, Sliding Scale Theory, Effects Test and International targeting,Jurisdiction under IT Act, 2000.

Module 2:

Cyber Crimes& Legal Framework: Cyber Crimes against Individuals, Institution and State, Hacking, Digital Forgery, Cyber Stalking/Harassment, Ethics and Etiquettes of Cyber World, Cyber Pornography, Identity Theft & Fraud, Cyber Terrorism, Cyber Defamation, Right to Privacy and Data Protection on Internet, Concept of privacy, Threat to privacy on internet, Self-regulation approach to privacy.

Module 3:

Overview of Intellectual Property: introduction and the need for intellectual property right (IPR), IPR in India – Genesis and Development IPR in abroad, Data Protection, Open Source Software, Macro economic impact of the patent system, Patent and kind of inventions protected by a patent, Patent document How to protect your inventions?, Granting of patent, Rights of a patent

Module 4:

Copyright, Related Rights and Trademarks: What is copyright? Latest editions of Designs, what is covered by copyright? How long does copyright last? Why protect copyright? What are related rights?,Distinction between related rights and copyright?, What is a trademark? Rightsof trademark?, What kind of signs can be used as trademarks?, types oftrademark, function does a trademark perform, How is a trademark protected?,
How is a trademark registered?

Module 5:

Copyright, Related Rights and Trademarks: What is a trademark? Rightsof trademark?, What kind of signs can be used as trademarks?, types oftrademark, function does a trademark perform, How is a trademark protected?, How is a trademark registered?

Labs / Practicals:

1. Experiment No.1: Study of IT Act, 2000 (with Amendments): Analyze key sections of the Information Technology Act, 2000 related to cybercrimes, digital signatures, and data protection.
2. Experiment No.2: Case Study: Cyber Crime Investigation: Analyze a real-world cybercrime case (e.g., phishing, hacking, identity theft) and present findings with legal outcomes.
3. Experiment No.3: Drafting a Cybercrime FIR / Complaint: Prepare a mock complaint or FIR based on a given cybercrime scenario following legal procedures.
4. Experiment No.4: Simulation: Filing a Cyber Crime Complaint Online (State/National Cybercrime Portal):

Navigate the National Cyber Crime Reporting Portal and simulate a complaint submission (without actually filing).
5. Experiment No.5: Analysis of a Legal Judgement Related to Cyber Law: Study and present a summary of a landmark Indian or international cyber law judgment (e.g., Shreya Singhal v. Union of India).
6. Experiment No.6: Understanding Digital Signature and E-Governance: Demonstrate how digital signatures are created and verified using tools (e.g., DSC utility or eSign services).
7. Experiment No.7: Study of Data Protection and Privacy Laws (India vs Global): Compare Indian data protection laws (e.g., Digital Personal Data Protection Act 2023) with GDPR or CCPA.
8. Experiment No.8: Mock Trial or Debate on Cyber Law Violations: Conduct a role-play or debate where students act as prosecution and defense in a cybercrime case.
9. Experiment No.9: Awareness Campaign Creation: Cyber Hygiene and Legal Awareness: Design posters, leaflets, or videos to educate the public on cyber safety and legal remedies.
10. Experiment No.10: Exploration of IPR Types with Examples: Identify examples of Patents, Copyrights, Trademarks, and Trade Secrets in software and hardware industries.
11. Experiment No.11: Filing a Patent – Procedure and Document Preparation: Simulate the preparation of a patent application for a software innovation, including patent claims and abstract.
12. Experiment No.12: Copyright Filing Process (India): Study the online procedure for filing a copyright and create a sample copyright form for a software or design.
13. Experiment No.13: Trademark Search and Application Simulation: Perform a trademark search using the IP India website and simulate a basic trademark application process.
14. Experiment No.14: Plagiarism Detection and Copyright Infringement: Use tools like Turnitin or online plagiarism checkers to detect infringement

References:

1. Anirudh Rastogi. Cyber Law, LexisNexis.
2. Vakul Sharma. Information Technology Law and Practice Cyber Laws and Laws Relating to E-Commerce, Universal Law Publishing.
3. Pankaj Sharma. Information Security and Cyber Laws, Kataria, S. K., & Sons.
4. Navneet Nagpal. Intellectual Property Right, Ebooks2go Inc.
5. Dr. S.K. singh. Intellectual Property Rights, Central Law Agency.

Course Code	Course Name	L	T	P	Cr
DCSE250013	Intrusion Detection System (IDS)	3	0	2	4

Course Outcomes:

CO1: Understand the fundamental principles, objectives, and architectural components of Intrusion Detection Systems (IDS).
CO2: Differentiate between various types of IDS, including host-based, network-based, signature-based, and anomaly-based systems.
CO3: Analyze and evaluate different intrusion detection techniques for their effectiveness in identifying security threats and attacks.
CO4: Configure and deploy open-source IDS tools (e.g., Snort, Suricata, Zeek) to monitor and detect suspicious activities in networks.
CO5: Interpret and correlate IDS logs and alerts to identify potential threats, system behaviors, and recommend appropriate response strategies.

Module 1:

History of Intrusion detection, Audit, Concept and definition , Internal and external threats to data, attacks, Need and types of IDS, Information sources Host based information sources, Network based information sources.

Module 2:

Intrusion Prevention Systems, Network IDs protocol based IDs, Hybrid IDs, Analysis schemes, thinking about intrusion. A model for intrusion analysis, techniques Responses requirement of responses, types of responses mapping responses to policy Vulnerability analysis, credential analysis non credential analysis

Module 3:

Introduction to Snort, Snort Installation Scenarios, Installing Snort, Running Snort on Multiple Network Interfaces, Snort Command Line Options. Step-By-Step Procedure to Compile and Install Snort Location of Snort Files, Snort Modes Snort Alert Modes

Module 4:

Working with Snort Rules, Rule Headers, Rule Options, The Snort Configuration File etc. Plugins, Preprocessors and Output Modules, Using Snort with MySQL

Module 5:

Using ACID and Snort Snarf with Snort, Agent development for intrusion detection, Architecture models of IDs and IPs.

Labs / Practicals:

1. Experiment No.1: Introduction to IDS and Packet Capturing Tools: Install and explore tools like Wireshark or tcpdump to capture and analyze network traffic.
2. Experiment No.: Installation and Configuration of Snort (Signature-based IDS): Install Snort on a Linux environment and run it in packet logging mode to monitor traffic.
3. Experiment No.3: Creating Custom Snort Rules: Write and test basic Snort rules to detect specific packet patterns (e.g., ICMP, port scan, HTTP requests).
4. Experiment No.4: Testing Snort for Port Scanning Attack Detection: Use Nmap to simulate a port scan and detect the intrusion using Snort.
5. Experiment No.5: Log Analysis for Suspicious Activities: Analyze system logs, auth logs, or web server logs to identify potential intrusion attempts.
6. Experiment No.6: Implementing Host-Based IDS using OSSEC: Install and configure OSSEC, analyze real-time log monitoring and rootkit detection features.
7. Experiment No.7: Simulating a DoS Attack and Detection with IDS: Launch a DoS attack using hping3 or

similar tools and observe detection with Snort or Suricata.
8. Experiment No.8: Intrusion Detection Using Suricata (Modern IDS/IPS): Install Suricata, configure YAML rules, and compare it with Snort for protocol-based detection.
9. Experiment No.9: Detecting Malware Command and Control (C&C) Traffic: Use simulated malware behavior and detect C&C traffic patterns with IDS logs.
10. Experiment No.10: Anomaly-based IDS using Machine Learning (Intro Experiment): Load a sample dataset (e.g., KDD Cup '99 or NSL-KDD) and apply basic ML models to detect intrusions using Python/Scikit-learn.
11. Experiment No.11: Integration of IDS with SIEM Tools: Demonstrate how Snort or OSSEC alerts can be integrated with Security Information and Event Management (SIEM) tools like Splunk or Graylog.
12. Experiment No.12: Real-Time Alert Generation and Email Notification: Configure Snort or OSSEC to send alert notifications via email or SMS on intrusion detection.
13. Experiment No.13: Creating a Simple Honeypot and Monitoring Attacks: Deploy a low-interaction honeypot (like Honeyd) and analyze attack attempts with IDS.
14. Experiment No.14: Comparison between Signature-Based and Anomaly-Based IDS: Conduct experiments to compare detection capabilities of both types of IDS using simulated attacks.
15. Experiment No.15: Report Generation and Analysis of Detected Attacks: Generate and present a detailed penetration test report including attack types, IDS rules

References:

1. Rafeeq Rehman : " Intrusion Detection with SNORT, Apache, MySQL, PHP and ACID," 1st Edition, Prentice Hall , 2003.
2. Christopher Kruegel,Fredrik Valeur, Giovanni Vigna: "Intrusion Detection and Correlation Challenges and Solutions", 1st Edition, Springer, 2005.
3. Carl Endorf, Eugene Schultz and Jim Mellander " Intrusion Detection & Prevention", 1st Edition, Tata McGraw-Hill, 2004.
4. Stephen Northcutt, Judy Novak : "Network Intrusion Detection", 3rd Edition, New Riders Publishing, 2002.
5. T. Fahringer, R. Prodan, "A Text book on Grid Application Development and Computing Environment". 6th Edition, KhannaPublihsers, 2012.

Course Code	Course Name	L	T	P	Cr
DCSE250015	Cyber Attacks	3	0	2	4

Course Outcomes:

CO1: Identify and describe various types of cyber attacks and assess their impact on digital systems, data, and networks.
CO2: Analyze common cyber attack techniques and tools such as malware, phishing, denial-of-service (DoS/DDoS), and SQL injection.
CO3: Examine the stages of a cyber attack lifecycle and evaluate methods used during reconnaissance, exploitation, and data exfiltration.
CO4: Investigate real-world cyber attack case studies to extract insights on system vulnerabilities, attacker methods, and response strategies.
CO5: Apply defensive strategies and use basic security tools (e.g., firewalls, IDS, threat intelligence platforms) to detect and mitigate cyber threats.

Module 1:

Introduction to Cyber Attacks: Introduction to Cyber Security, Overview of Cyber Attacks: Types, Vectors, and Targets, Motivations behind Cyber Attacks (Political, Financial, etc.), Recent trends and case studies, Attack Lifecycle and Kill Chain model

Module 2:

Malware and Attack Techniques: Malware types: Virus, Worm, Trojan, Ransomware, Spyware, Rootkits, Delivery methods (email, drive-by download, infected USBs), Botnets and Command & Control (C2) mechanisms, Code injection attacks (SQL injection, Command injection), Exploits and vulnerability scanning

Module 3:

Network and Web-Based Attacks: Denial of Service (DoS) and Distributed DoS attacks, Man-in-the-Middle (MITM) attacks, Packet sniffing and spoofing, ARP poisoning, DNS poisoning, Cross-Site Scripting (XSS) and Cross-Site Request Forgery (CSRF), Clickjacking, Session hijacking, Case studies: DDoS on DNS providers, Bank MITM attacks

Module 4:

System and Application Level Attacks: Privilege escalation techniques, Buffer overflows and heap overflows, Backdoors and rootkits, Exploitation of web and desktop applications, Reverse engineering basics. Social Engineering and Insider Threats: Phishing, Spear phishing, Whaling
Pretexting, Baiting, Tailgating, Insider threats – malicious and negligent insiders, Social engineering in corporate and military settings

Module 5:

Detection, Prevention, and Mitigation: Basics of Intrusion Detection and Prevention Systems (IDS/IPS), Firewalls and Honeypots, Endpoint Detection and Response (EDR), Threat Intelligence and SIEM systems, Legal and ethical aspects of cyber attack response, Cybercrime laws (IT Act, GDPR, etc.)

Labs / Practicals:

1. Experiment No.1: Reconnaissance and Footprinting: Use tools like whois, nslookup, theHarvester, or Recon-ng to gather information about a target domain or IP.
2. Experiment No.2: Scanning and Enumeration: Perform network scanning using Nmap and enumeration using enum4linux to identify open ports and services.
3. Experiment No.3: Brute Force Attack using Hydra: Use Hydra to perform a brute-force login attack on SSH or a web form (in a test setup).

4. Experiment No.4: Denial of Service (DoS) Attack: Simulate a DoS attack using Hping3 or LOIC on a local web server and monitor its impact.
5. Experiment No.5: Man-in-the-Middle (MitM) using ARP Spoofing: Execute a MitM attack using Ettercap to intercept communication and extract login credentials.
6. Experiment No.6: Phishing Attack Simulation: Create a mock phishing page and demonstrate how users can be tricked into submitting credentials.
7. Experiment No.7: SQL Injection Attack on DVWA: Use test inputs to perform SQL injection and extract sensitive data from a vulnerable web application.
8. Experiment No.8: Cross-Site Scripting (XSS) Attack: Demonstrate stored and reflected XSS attacks using a test platform like DVWA or OWASP Juice Shop.
9. Experiment No.9: DNS Spoofing Attack: Perform a DNS spoofing attack to redirect a legitimate request to a fake website (within a closed lab setup).
10. Experiment No.10: Attack Detection using Snort IDS: Simulate a known attack (e.g., port scan) and detect it using custom rules in Snort.

References:

1. William Stallings, Computer Security: Principles and Practice, Pearson
2. Nina Godbole & Sunit Belapure, Cyber Security, Wiley
3. Thomas Ristenpart (Coursera), Cybersecurity and Its Ten Domains
4. Real-world reports (e.g., Verizon Data Breach Report, CERT advisories)

Course Code	Course Name	L	T	P	Cr
DCSE250020	Cloud Computing	3	0	2	4

Course Outcomes:

CO1: Explain the evolution, architecture, and key characteristics of cloud computing, including service models (IaaS, PaaS, SaaS) and deployment types (public, private, hybrid).

CO2: Describe various types and implementation levels of virtualization, including CPU, memory, storage, and network virtualization in cloud environments.

CO3: Analyze layered cloud architecture and address design challenges related to inter-cloud resource management and platform deployment.

CO4: Apply distributed and parallel programming paradigms such as MapReduce and Hadoop for cloud-based application development and understand various cloud platforms like AWS, OpenStack, and Google App Engine.

CO5: Evaluate cloud security challenges and solutions, including data and application security, VM security, and risk management strategies in cloud environments.

Module 1:

Introduction to Cloud Computing

Evolution of Cloud Computing, System Models for Distributed and Cloud Computing, NIST Cloud Computing Reference Architecture, Features of Cloud Computing, Cloud Services – IaaS, PaaS, SaaS, Cloud service Providers – Public , Private and Hybrid Clouds.

Module 2:

Introduction to component virtualization

Basics of Virtualization, Types of Virtualization, Implementation Levels of Virtualization, Virtualization of CPU, Memory, I/O Devices, Desktop Virtualization, Server Virtualization, Storage Virtualization, Network Virtualization.

Module 3:

Architectural Design of Compute and Storage Clouds

Layered Cloud Architecture Development, Design Challenges, Inter Cloud Resource Management, Resource Provisioning and Platform Deployment, Global Exchange of Cloud Resources.

Module 4:

Parallel and Distributed Programming Paradigms

Map Reduce, Twister and Iterative MapReduce, Hadoop Library from Apache, Mapping Applications, Programming Support, Google App Engine, Amazon AWS, Cloud Software Environments - Eucalyptus, Open Nebula, OpenStack.

Module 5:

Security Overview

Cloud Security Challenges, Software-as-a-Service Security, Security Governance, Risk Management, Security Monitoring, Security Architecture Design, Data Security, Application Security, Virtual Machine Security.

Labs / Practicals:

1. Exploring Cloud Deployment and Service Models
2. Simulating Public, Private, and Hybrid Clouds
3. Virtualization of CPU and Memory
4. Network and Storage Virtualization
5. Deployment of a Layered Cloud Architecture
6. Resource Provisioning and Scaling
7. Deploying a Web Application
8. Simulating Security Attacks and Defenses in a Cloud VM
9. Implementing Access Control and Encryption on Cloud Storage

References:

[1] R. Buyya, C. Vecchiola, and T. Selvi, Mastering Cloud Computing: Foundations and Applications Programming. New Delhi, India: McGraw-Hill Education, 2013.
[2] A. T. Velte, T. J. Velte, and R. Elsenpeter, Cloud Computing: A Practical Approach. New Delhi, India: McGraw-Hill Education, 2010.
[3] K. Hwang, G. C. Fox, and J. J. Dongarra, Distributed and Cloud Computing: From Parallel Processing to the Internet of Things. Burlington, MA, USA: Morgan Kaufmann, 2012.
[4] T. Erl, R. Puttini, and Z. Mahmood, Cloud Computing: Concepts, Technology & Architecture. Boston, MA, USA: Pearson Education, 2013.
[5] G. Reese, Cloud Application Architectures: Building Applications and Infrastructure in the Cloud. Sebastopol, CA, USA: O'Reilly Media, 2009.

Course Code	Course Name	L	T	P	Cr
DCSE250021	Vulnerability Analysis and Penetration Testing	3	0	2	4

Course Outcomes:

CO1: Understand the fundamental concepts, phases, and legal/ethical considerations involved in vulnerability assessment and penetration testing.
CO2: Conduct reconnaissance and information gathering using both passive and active techniques to identify potential attack surfaces.
CO3: Utilize industry-standard tools (e.g., Nmap, Nessus, OpenVAS) to perform vulnerability scanning on systems, networks, and applications.
CO4: Analyze and exploit identified vulnerabilities using ethical hacking frameworks and tools such as Metasploit in a controlled environment.
CO5: Document and report penetration testing findings with appropriate risk analysis, technical details, and mitigation strategies.

Module 1:

Introduction to VAPT: Introduction to Vulnerability Assessment and Penetration Testing, Differences between Vulnerability Assessment and Penetration Testing, Role of VAPT in Cybersecurity, Phases of Penetration Testing (Planning, Discovery, Attack, Reporting), Legal and ethical considerations. Information Gathering and Footprinting: Passive and Active Reconnaissance
WHOIS, DNS Interrogation, Google Hacking, Network Scanning and Enumeration (Nmap, Netcat), OS Fingerprinting, Social Engineering Basics

Module 2:

Vulnerability Assessment Tools and Techniques: Vulnerability scanners: Nessus, OpenVAS, Nikto, Common Vulnerability Scoring System (CVSS)
Identifying vulnerabilities in web applications, networks, and systems, Interpreting scan results, Manual testing vs automated testing

Module 3:

Penetration Testing Methodologies: Types of Penetration Testing: Black Box, White Box, Gray Box, Web application penetration testing (OWASP Top 10 vulnerabilities), Network penetration testing, Wireless penetration testing, Exploitation techniques and tools (Metasploit Framework)

Module 4:

Exploitation and Privilege Escalation: Exploiting common vulnerabilities (SQL Injection, XSS, CSRF, RCE), Post-exploitation strategies, Privilege escalation on Windows and Linux, Maintaining access (backdoors, persistence techniques), Anti-forensics techniques. Reporting and Risk Mitigation: Documenting test findings and recommendations, Writing professional VAPT reports, Risk rating and prioritization, Remediation strategies and re-testing, Industry compliance and standards (ISO 27001, PCI-DSS)

Module 5:

Recent Trends and Case Studies: Real-world breach case studies, Bug bounty programs and responsible disclosure, AI/ML in penetration testing, Cloud and IoT security testing overview, Emerging threats and advanced testing approaches

Labs / Practicals:

1. Experiment No.1: Information Gathering and Reconnaissance: Use tools like Nmap, theHarvester, Netdiscover, and WHOIS to gather system/network information.

2. Experiment No.2: Vulnerability Scanning using Nessus/OpenVAS: Perform automated vulnerability scans on a target machine using Nessus or OpenVAS and analyze the results.
3. Experiment No.3: Web Application Scanning using Nikto and Burp Suite: Scan a test web application (e.g., DVWA, Juice Shop) for known vulnerabilities using Nikto and analyze HTTP traffic with Burp Suite.
4. Experiment No.4: Exploitation using Metasploit Framework: Exploit known vulnerabilities (e.g., MS08-067) in Metasploitable using Metasploit, and gain access to the system.
5. Experiment No.5: Password Cracking and Hash Cracking: Use John the Ripper, Hashcat, or Hydra to crack weak passwords or hashes extracted from a compromised system.
6. Experiment No.6: Cross-Site Scripting (XSS) and Cross-Site Request Forgery (CSRF): Perform XSS and CSRF attacks on vulnerable web apps (e.g., DVWA) and understand security flaws in input validation.
7. Experiment No.7: SQL Injection and Bypassing Login: Identify and exploit SQL injection vulnerabilities to bypass login forms or extract data from the backend database.
8. Experiment No.8: File Upload Vulnerability Exploitation: Test a vulnerable web app for unrestricted file upload and demonstrate how a malicious file can lead to shell access.
9. Experiment No.9: Local File Inclusion (LFI) and Remote File Inclusion (RFI): Perform LFI/RFI attacks on a web server and observe how attackers can read sensitive files or execute code.
10. Experiment No.10: Vulnerability Reporting and Remediation: Prepare a comprehensive Vulnerability Assessment Report that includes the tools used, findings, CVE references, and recommended fixes.

References:

1. Georgia Weidman, Penetration Testing: A Hands-On Introduction to Hacking, No Starch Press
2. Stuart McClure et al., Hacking Exposed
3. OWASP Top 10 Project Documentation
4. Official Documentation of Nessus, Metasploit, and OpenVAS

Course Code	Course Name	L	T	P	Cr
DOAI250019	Data Mining and Warehousing	3	0	2	4

Course Outcomes:

CO1: Understand and apply data warehousing concepts and architectures.

CO2: Perform clustering, classification, and prediction for pattern discovery.

CO3: Address issues related to data streams and implement stream mining techniques.

CO4: Analyze and apply web mining methodologies for real-world applications.

CO5: Explore current trends and advancements in distributed warehousing and pattern mining.

Module 1:

Data warehousing components, Building a data warehouse, Data warehouse architecture, DBMS schemas for decision support, Data extraction, cleanup, and transformation tools, Metadata, Reporting, Query tools and applications, Online Analytical Processing (OLAP), OLAP and multidimensional data analysis.

Module 2:

Data mining functionalities, Data preprocessing: cleaning, integration, transformation, reduction, discretization, Concept hierarchy generation, Architecture of typical data mining systems, Classification of data mining systems, Efficient and scalable frequent itemset mining methods, Mining various kinds of association rules, From association mining to correlation analysis, Constraint-based association mining.

Module 3:

Issues in classification and prediction, Classification methods: Decision Trees, Bayesian classification, Rule-based classification, Neural networks: backpropagation, Support Vector Machines (SVM), Associative classification, Lazy learners, Other classification methods, Prediction techniques, Accuracy and error measures, Evaluating classifiers/predictors, Ensemble methods, Model selection.

Module 4:

Types of data in clustering, Categorization of major clustering methods, Partitioning methods, Hierarchical methods, Density-based methods, Grid-based methods, Model-based clustering methods, Clustering high-dimensional data, Constraint-based clustering, Outlier analysis.

Module 5:

Mining objects, Spatial data mining, Multimedia data mining, Text mining, Mining the World Wide Web, Multidimensional analysis of complex data, Descriptive mining of complex data objects.

Labs / Practicals:

1. Design a Data Warehouse Schema

Create Star Schema and Snowflake Schema for business data (sales, e-commerce, etc.).

2. ETL Process Implementation

Perform Extract, Transform, Load (ETL) operations on structured data using SQL-based tools (MySQL / PostgreSQL).

3. Association Rule Mining

Apply the Apriori Algorithm for mining frequent itemsets and generating association rules from transactional datasets.

4. Classification Using Decision Trees

Build, visualize, and evaluate Decision Tree Classifiers on datasets like Iris or Titanic.

5. Clustering Using K-Means

Apply K-Means Clustering on multi-dimensional datasets and visualize the results.

6. Data Preprocessing & Dimensionality Reduction

Handle missing values, normalize data, and apply Principal Component Analysis (PCA).

7. OLAP Operations and Cube Construction

Perform OLAP operations (Roll-up, Drill-down, Slice, Dice) using sample datasets.

8. Web Mining Basics

Analyze web log files for pattern discovery (frequent user paths, sessions).

References:

1. Jiawei Han, Micheline Kamber, Jian Pei, Data Mining Concepts and Techniques, Third Edition, Elsevier, 2011.

2. Alex Berson, Stephen J. Smith, Data Warehousing, Data Mining & OLAP, Tata McGraw-Hill, 10th Reprint, 2007.

3. K.P. Soman, Shyam Diwakar, V. Ajay, Insight into Data Mining: Theory and Practice, PHI, 2006.

4. G.K. Gupta, Introduction to Data Mining with Case Studies, PHI, 2006.

5. Pang-Ning Tan, Michael Steinbach, Vipin Kumar, Introduction to Data Mining, Pearson Education, 2007.

Course Code	Course Name	L	T	P	Cr
DOAI250041	Natural Language Processing	3	0	2	4

Course Outcomes:

CO1: Understand the history, stages, challenges, and applications of NLP in real-world systems.
CO2: Apply morphological analysis, finite state methods, stemming, and n-gram models for language processing.
CO3: Implement POS tagging, CFG parsing, and sentence-level syntactic analysis using statistical and ML-based methods.
CO4: Analyze word sense disambiguation techniques using machine learning and dictionary-based approaches.
CO5: Apply discourse analysis techniques including reference resolution, pronoun interpretation, and coherence modeling.

Module 1:

History of NLP; Generic NLP system; Levels of NLP; Knowledge in language processing problem; Ambiguity in natural language; Stages in NLP; Challenges of NLP; Role of machine learning; Brief history of the field; Applications of NLP: Machine translation, Question answering system, Information retrieval, Text categorization, text summarization & Sentiment analysis

Module 2:

Morphology analysis survey of English morphology, inflectional morphology & derivational morphology; Regular expressions; Finite automata; Finite state transducers (FST); Morphological parsing with FST; Lexicon free FST, Porter stemmer, N-Grams, N-gram language model, N-gram for spelling correction.

Module 3:

Part-of-Speech tagging (POS); Lexical syntax tag set for English (Penn Treebank); Rule based POS tagging; Stochastic POS tagging; Issues: Multiple tags & words, unknown words, class-based n-grams, HM Model ME, SVM, CRF; Context Free Grammar; Constituency; Context free rules & trees; Sentence level construction; Noun Phrase; Coordination; Agreement; Verb phrase & sub categorization

Module 4:

Attachment for fragment of English sentences, noun phrases, verb phrases, prepositional phrases; Relations among lexemes & their senses; Homonymy, Polysemy based disambiguation & limitations, Robust WSD; Machine learning approach and dictionary-based approach.

Module 5:

Discourse reference resolution; Reference phenomenon; Syntactic & semantic constraints on co reference; Preferences in pronoun interpretation; Algorithm for pronoun resolution; Text coherence; Discourse structure

Labs / Practicals:

Module 1: Introduction to NLP & Applications
1. Experiment 1: Demonstration of NLP Applications
o Implement basic NLP applications like sentiment analysis, text summarization, question answering, and information retrieval using pre-trained models or APIs.
2. Experiment 2: Text Preprocessing Techniques
o Perform tokenization, stop-word removal, stemming, and lemmatization on sample English texts.

Module 2: Morphology and Language Models
3. Experiment 3: Morphological Analysis using Stemming and Lemmatization
o Analyze inflectional and derivational forms of English words using stemmers and lemmatizers.
4. Experiment 4: Implementation of N-Gram Language Models

o Build unigram, bigram, and trigram models from a text corpus and compute probabilities of word sequences.
5. Experiment 5: Spelling Correction using N-Gram Probabilities
o Implement a simple spelling corrector using edit distance and N-gram based word probability estimation.

Module 3: POS Tagging and Grammar Parsing
6. Experiment 6: POS Tagging using Rule-Based and Statistical Methods
o Perform part-of-speech tagging using both rule-based and statistical approaches; evaluate accuracy on tagged corpora.
7. Experiment 7: Constituency Parsing using Context-Free Grammar (CFG)
o Define a CFG and parse given sentences to generate parse trees showing sentence structure.

Module 4: Word Sense Disambiguation
8. Experiment 8: Word Sense Disambiguation using Dictionary-Based Methods
o Use the Lesk algorithm to identify correct word senses in different sentence contexts.
9. Experiment 9: Word Sense Disambiguation using Supervised Learning
o Implement a supervised machine learning model to classify the correct sense of an ambiguous word using labeled data.

Module 5: Discourse and Reference Resolution
10. Experiment 10: Co-reference Resolution and Discourse Analysis
• Resolve pronouns and noun phrase references in a passage and analyze coherence and discourse structure.

References:

1. James Allen. Natural Language Understanding. The Benajmins/Cummings Publishing Company Inc. 1994. ISBN 0-8053-0334-0.
2. Tom Mitchell. Machine Learning. McGraw Hill, 1997. ISBN 0070428077.
3. Cover, T. M. and J. A. Thomas: Elements of Information Theory. Wiley. 1991. ISBN 0-471-06259-6.

Course Code	Course Name	L	T	P	Cr
DOEE250053	Data Acquisition Techniques	3	0	2	4

Course outcomes and modules not available

Course Code	Course Name	L	T	P	Cr
DOEE250094	Embedded Systems	3	0	2	4

Course Outcomes:

CO1: Understand the fundamental architecture and components of embedded systems.
CO2: Develop embedded programs using Embedded C and relevant development tools.
CO3: Interface sensors, actuators, and peripheral devices using standard communication protocols.
CO4: Apply real-time operating system concepts for task management and synchronization.
CO5: Design and implement embedded solutions for real-world applications through hands-on projects.

Module 1:

Introduction to Embedded Systems: Definition and characteristics of embedded systems, Classification: Standalone, real-time, networked, mobile
Examples and applications in various domains, Design metrics and challenges, Overview of embedded system development lifecycle. Embedded System Architecture: Microprocessor vs Microcontroller, Embedded system hardware: processor, memory, I/O devices, System-on-Chip (SoC), ASIC, and FPGA basics, Bus communication protocols (I2C, SPI, UART, CAN), Harvard vs Von Neumann architectures

Module 2:

Embedded Programming and Tools: Embedded C programming essentials, Cross-compilation and toolchains, IDEs, simulators, debuggers, and emulators, Real-time programming concepts, GPIO, timer, and interrupt programming

Module 3:

Real-Time Operating Systems (RTOS): Introduction to RTOS and its features, Task scheduling and context switching, Inter-process communication: semaphores, queues, mutexes, Case study: FreeRTOS or RTLinux, Task creation and synchronization. Memory Management and Power Optimization: Memory types and hierarchy in embedded systems, Static and dynamic memory allocation, Memory protection and MMU basics, Power-aware design: techniques and trade-offs
Low-power modes and energy-efficient programming

Module 4:

Interfacing and Communication: Interfacing with sensors, actuators, and displays, ADC/DAC interfacing, Serial and parallel communication, Wireless communication (Bluetooth, ZigBee, Wi-Fi basics), Protocol stack for IoT integration

Module 5:

Embedded System Design and Case Studies: Design methodology: hardware-software co-design, Design trade-offs and cost-performance analysis, Case studies: Smart home device, Embedded medical system, Automotive embedded system

Labs / Practicals:

1. Experiment No.1: Introduction to Microcontroller Programming (Blinking LED): Write and upload a program to blink an LED using Arduino or Raspberry Pi GPIO, understanding basic I/O operations.
2. Experiment No.2: Interfacing Push Button and LED (Digital Input/Output): Interface a push button with a microcontroller to control an LED, demonstrating digital input and output handling.
3. Experiment No.3: Analog Sensor Interfacing (e.g., Temperature or Light Sensor): Read data from analog sensors like LM35 (temperature) or LDR using ADC and display results.
4. Experiment No.4: Interfacing LCD Display (16x2 or OLED): Display sensor readings or messages on a 16x2

LCD or OLED screen using I2C or parallel communication.
5. Experiment No.5: PWM Generation and Motor Speed Control: Use Pulse Width Modulation (PWM) to control the speed of a DC motor or the brightness of an LED.
6. Experiment No.6: Serial Communication using UART (PC to Microcontroller): Implement UART-based communication to send/receive data between a microcontroller and a PC via serial monitor.
7. Experiment No.7: I2C or SPI Protocol Implementation: Interface devices like RTC module, EEPROM, or sensors using I2C or SPI communication protocols.
8. Experiment No.8: Interfacing PIR Sensor for Motion Detection: Detect motion using a PIR sensor and trigger an LED or buzzer—demonstrating event-driven response.
9. Experiment No.9: IoT-Based Data Transmission using Wi-Fi Module (ESP8266/NodeMCU): Send sensor data to a web server/cloud platform (like ThingSpeak) using Wi-Fi module.
10. Experiment No.10: Mini Project: Embedded Application (Home Automation/Security System): Design and implement a small embedded system project (e.g., automatic door lock, smart light, fire alarm) integrating sensors and actuators.

References:

1. Raj Kamal, Embedded Systems: Architecture, Programming and Design, McGraw Hill
2. Frank Vahid & Tony Givargis, Embedded System Design, Wiley
3. Shibu K. V., Introduction to Embedded Systems, McGraw Hill
4. FreeRTOS official documentation
5. ARM Cortex-M processor manuals (as applicable)

Course Code	Course Name	L	T	P	Cr
DOEE250121	Internet of Things	3	0	2	4

Course Outcomes:

CO 1 Explain the concept of IoT
CO 2 Networking basics for IoT application development
CO 3 Analyze various protocols for IoT
CO 4 Design a PoC of an IoT system using various hardware platforms
CO 5 Apply data analytics and use cloud offerings related to IoT
CO 6 Analyze applications of IoT in real time scenario

Module 1:

Overview of IoT

Introduction to the Internet of Things, IoT system architecture and standards, Networking Basics - TCP/IP, Networking Basics - IP addressing basics (IPV4 and IPV6), Optimizing IP for IoT: From 6LoWPAN to 6Lo, Routing over Low Power and Lossy Networks.

Module 2:

IoT hardware Platforms

IoT Design Methodology – Embedded computing logic – Microcontroller-System on Chips, Hardware platforms for prototyping IoT node- Arduino, Raspberry Pi, NodeMCU, ESP32, ARM Cortex Microcontrollers, IoT mote hardware platforms, Swadeshi RISC V based solutions, Interfacing sensors and actuators with hardware platforms, Developing IoT applications using Raspberry Pi with Python Programming.

Module 3:

IoT connectivity & Protocols

IoT Access Technologies: WiFi, Zigbee, Zwave, Bluetooth,UWB, sub1GHz, LoRaWAN, Sigfox and NB-IoT, Topology and Security of IEEE 802.15.4, 802.15.4g, 802.15.4e, 1901.2a, 802.11ah and LoRaWAN, IoT application level protocols: MQTT, CoAP, XMPP, HTTP/Rest Services, WebSockets.

Module 4:

Data analytics, Cloud and IoT Security

No SQL Databases Vs SQL Databases, Apache web servers, JSON, Open and commercial Cloud solutions for IoT, Python Web Application Frameworks for IoT, IoT data visualisation tools, IoT Security - Need for encryption, standard encryption protocol, lightweight cryptography, Trust models for IoT, ARM Cortex Microcontroller Security, Root Security Services (RSS).

Module 5:

IoT Case studies

Smart Lighting, Smart home, Smart Agriculture, Smart farming, IoT for health care & patient monitoring, Smart and Connected Cities, Building end-to-end smart applications with TinyML.

Labs / Practicals:

Hands-on Sessions on IoT Application Development using IoT Hardware Kits (ESP32/STM32/Rpi etc.)

- IPv4 and IPv6 Implementation
- 6LoWPAN Network Implementation
- Interfacing multiple sensors and actuators with Processor Hardware using different communication protocols and develop an integrated monitoring system.
- Implementation of different IoT connectivity technologies including WiFi, Bluetooth, and BLE for various application scenarios.
- Implementation of MQTT protocol for IoT device communication
- Implementation of CoAP for resource-constrained IoT devices
- Integration of IoT devices with cloud platforms for data analytics

References:

Reference Books

1. David Hanes, Gonzalo Salgueiro, Patrick Grossetete, Rob Barton and Jerome Henry, —IoT Fundamentals: Networking Technologies, Protocols and Use Cases for Internet of Things, Cisco Press, 2017
2. Alessandro Bassi, Martin Bauer, Martin Fiedler, Thorsten Kramp, Rob van Kranenburg, Sebastian Lange, Stefan Meissner, “Enabling things to talk – Designing IoT solutions with the IoT Architecture Reference Model”, Springer Open, 2016
3. VijayMadisetti , ArshdeepBahga, Adrian McEwen (Author), Hakim Cassimally “Internet of Things: A Hands-on-Approach” ArshdeepBahga& Vijay Madisetti, 2014.
4. Gian Marco Iodice, TinyML Cookbook: Combine artificial intelligence and ultralow-power embedded devices to make the world smarter
5. Olivier Hersent, David Boswarthick, Omar Elloumi , —The Internet of Things – Key applications and Protocols, Wiley, 2012

Course Code	Course Name	L	T	P	Cr
DOEE250161	Principles of Industrial IoT	3	0	2	4

Course Outcomes:

CO1: Understand the enabling technologies, catalysts, and use cases that drive the development of Industrial IoT systems.
CO2: Describe and design IIoT Reference Architecture and identify the role of network, middleware, and application layers.
CO3: Analyze IIoT communication protocols, including Access Networks and WAN technologies, used in industrial environments.
CO4: Examine the importance of information security and data protection strategies in Industrial IoT systems.
CO5: Develop and experiment with IoT-based solutions for real-time industrial applications aligned with Industry 4.0 concepts.

Module 1:

Industry 4.0: (8 Hours)

Cyber Physical Systems and Next Generation Sensors, Collaborative Platform and Product Lifecycle Management, Augmented Reality and Virtual Reality, Artificial Intelligence, Big Data and Advanced Analysis

Module 2:

Industrial IoT: (8 Hours)

IIoT-Introduction, Industrial IoT: Business Model and Reference Architecture: IIoT-Business Models, Industrial IoT- Layers: IIoT Sensing, IIoT Processing, IIoT Communication, IIoT Networking

Module 3:

IIoT Analytics: (8 Hours)

Big Data Analytics and Software Defined Networks, Machine Learning and Data Science, Julia Programming, Data Management with Hadoop.

Module 4:

IoT Security: Industrial IoT (8 Hours)

Security and Fog Computing - Cloud Computing in IIoT, Fog Computing in IIoT, Security in IIoT

Module 5:

Case Study: (8 Hours)

Industrial IoT- Application Domains: Oil, chemical and pharmaceutical industry, Applications of UAVs in Industries, Real case studies: Milk Processing and Packaging Industries, Manufacturing Industries

Labs / Practicals:

1. Experiment No.1: Interfacing Industrial Sensors with Microcontroller (e.g., Temperature, Pressure, Proximity): Read data from industrial-grade sensors (e.g., LM35, MQ-135, ultrasonic, IR) using Arduino or NodeMCU.
2. Experiment No.2: Data Acquisition and Visualization using IoT Platform (e.g., ThingSpeak, Blynk): Send real-time sensor data to the cloud via Wi-Fi and visualize readings on dashboards.
3. Experiment No.3: Industrial Actuator Control using Microcontroller (Relay, Motor): Control devices like relays, fans, or valves using digital outputs based on sensor input or remote commands.

4. Experiment No.4: MQTT Protocol Implementation for IIoT Communication: Set up an MQTT broker (e.g., Mosquitto) and publish/subscribe to topics for device communication using Python or NodeMCU.
5. Experiment No.5: Edge Device Programming for Data Filtering and Local Decision Making: Implement basic logic on microcontrollers (like NodeMCU) to process sensor data locally before transmission.
6. Experiment No.6: Wireless Sensor Network Simulation using LoRa or Zigbee: Demonstrate long-range communication using LoRa or Zigbee modules between multiple IIoT nodes.
7. Experiment No.7: Industry 4.0 Use Case Simulation (Predictive Maintenance or Smart Monitoring): Simulate a condition-monitoring system using vibration, temperature, or pressure sensors and generate alerts.
8. Experiment No.8: Security Implementation in IIoT (Data Encryption using AES or RSA): Encrypt sensor data using AES before transmission and decrypt at the receiver end using Python or C.
9. Experiment No.9: Data Logging and Analysis using Raspberry Pi and Python: Collect sensor data, store it in CSV or database (SQLite/MySQL), and plot trends using Matplotlib or Pandas.
10. Experiment No.10: Mini Project: Industrial IoT Prototype (Smart Factory, Warehouse Monitoring, etc.): Develop a working prototype integrating sensors, edge devices, and a cloud dashboard for an industrial application.

References:

1. Industry 4.0: The Industrial Internet of Things", by Alasdair Gilchrist (Apress), 2017
2. Industrial Internet of Things: Cyber manufacturing Systems "by Sabina Jeschke, Christian Brecher, Houbing Song, Danda B. Rawat (Springer), 2017
3. Hands-On Industrial Internet of Things: Create a powerful Industrial IoT by Giacomo Veneri, Antonio Capasso, Packt, 2018.