SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COURSE STRUCTURE

Under New Education Policy (NEP) 2020: Academic Year 2026-2027 and Onwards

B.Tech. (Computer Science and Engineering) and Specialization in Artificial Intelligence

After Year 1: UG Certificate	After Year 2: UG Diploma	After Year 3: B.Sc. (Computer Science)	After Year 4: B.Tech. Degree

SEMESTER I

S.No.	Course Code	Course Name	L	T	P	Credits
1	CAI101	Problem Solving Using C	3	0	0	3
2	ECE101	Electronics and Electrical Engineering	3	0	0	3
3	PH102	Engineering Physics	3	1	0	4
4	MA111	Applied Mathematics-I	3	1	0	4
5	ICT101	Career Skills	2	0	0	2
6	PH104	Engineering Physics Lab	0	0	2	1
7	ICT181	ICT Workshop I	0	0	2	1
8	CAI181	Problem Solving Using C lab	0	0	2	1
9	ECE185	Electronics and Electrical Engineering La	0	0	2	1
10	XXXXX	Slot for HSMC/VAC	2	0	0	2
		Total Hours and Credits	**16**	**2**	**8**	**22**

SEMESTER II

S.No.	Course Code	Course Name	L	T	P	Credits
1	CAI102	Python Programming	3	0	0	3
2	MA112	Applied Mathematics-II	3	1	0	4
3	CAI104	Introduction to Artificial Intelligence	2	0	0	2
4	ICT102	Discrete Mathematics	3	0	0	3
5	ES101	Environmental Studies	3	1	0	4
6	EN101	English Proficiency	2	0	0	2
7	CAI182	Python Programming Lab	0	0	2	1
8	ICT182	ICT Workshop II	0	0	2	1
9	ICT184	Career Skills Lab	0	0	2	1
10	XXXXX	Slot for HSMC/VAC	2	0	0	2
		Total Hours and Credits	**18**	**2**	**6**	**23**

SEMESTER III

S.No.	Course Code	Course Name	L	T	P	Credits
1	CAI201	Scientific Computing with Python	3	0	0	3
2	CAI203	Data Structures	3	0	0	3
3	CAI205	Machine Learning	3	0	0	3
4	CAI207	Computer Networks	3	0	0	3
5	CAI209	Operating System	3	0	0	3
6	CAI281	Scientific Computing with Python Lab	0	0	4	2
7	CAI283	Data Structures Lab	0	0	4	2
8	CAI285	Computer Networks Lab	0	0	4	2
9	CAI287	Machine Learning Lab	0	0	4	2
		Total Hours and Credits	**15**	**0**	**16**	**23**

SEMESTER IV

S.No.	Course Code	Course Name	L	T	P	Credits
1	CAI202	Design and Analysis of Algorithms	3	0	0	3
2	CAI204	Database Management System	3	0	0	3
3	CAI206	Deep Learning	3	0	0	3
4	CAI208	Theory of Automata	3	0	0	3
5	CAI210	Digital Systems and Computer Architect	3	0	0	3
6	CAI282	Design and Analysis of Algorithms Lab	0	0	4	2
7	CAI284	Database Management System Lab	0	0	4	2
8	CAI286	Deep Learning Lab	0	0	4	2
9	XXXXX	Slot for HSMC/AEC	3	0	0	3
Total Hours and Credits			**18**	**0**	**12**	**24**

SEMESTER V

S.No.	Course Code	Course Name	L	T	P	Credits
1	CAI301	Software Engineering	3	0	0	3
2	CAI303	Compiler Design	3	0	0	3
3	CAI305	Internet of Things	3	0	0	3
4	CAI307	Advanced Generative Intelligence	3	0	0	3
5	CAI3XX	Program Elective 1	3	0	0	3
6	CAI3XX	Program Elective 2	3	0	0	3
7	CAI3XX	Program Elective 1 Lab	0	0	4	2
8	CAI381	Internet of Things Lab	0	0	4	2
Total Hours and Credits			**18**	**0**	**8**	**22**

SEMESTER VI

S.No.	Course Code	Course Name	L	T	P	Credits
1	CAI302	Competitive Programming	3	0	0	3
2	CAI304	Cloud Infrastructure and Virtualization	3	0	0	3
3	CAI306	Unmanned Aerial Vehicle Engineering	3	0	0	3
4	CAI3XX	Program Elective 3	3	0	0	3
5	CAI3XX	Program Elective 4	3	0	0	3
6	XXXXX	Open Elective 1	3	0	0	3
7	CAI382	Competitive Programming Lab	0	0	4	2
8	CAI384	Cloud Infrastructure and Virtualization	0	0	4	2
9	CAI392	Capstone Project-1*	0	0	4	2
Total Hours and Credits			**18**	**0**	**12**	**24**

* The Capstone Project-1/Capstone Project-2 will be completed in a group, and it will also be evaluated as per University Examination Rules.

SEMESTER VII

S.No.	Course Code	Course Name	L	T	P	Credits
3	CAI401	Research Methodology	3	0	0	3
1	XXXXX	Open Elective 2	3	0	0	3
2	XXXXX	Open Elective 3	3	0	0	3
4	XXXXX	Slot for HSMC/AEC	3	0	0	3
5	CAI493	Capstone Project-2*	0	0	8	4
		Total Hours and Credits	**12**	**0**	**8**	**16**

SEMESTER VIII

S.No.	Course Code	Course Name	L	T	P	Credits
1	CAI492	Internship #	0	0	20	10
		Total Hours and Credits	**0**	**0**	**20**	**10**

Internship will be completed by each candidate individually during the 8th semester in Industry / GBU. The duration of the internship will be a minimum of 4 months . It will be evaluated as per the University Examination rules.

Total Credits **164**

Exit at the end of first year, mandatory skill course module on Technical Communication and Computer Proficiency (Data Entry etc.), Civil / Mechanical Draftsmanship, Electrical maintenance/ internship etc of 4 credit should be completed. Exit after 2nd year of B.Tech. course must undergo skill modules on IT/Hardware Networking / MATLAB/ internship etc of 4 credit or Branch specific skill module. Course structure at 3rd year and 4th year of B.Tech. is already Engineering specific, students who exit after 3-years may be awarded B.Sc(CSE).
HSMC/VAC/AEC/SEC and Open Elective courses shall be offered from a basket floated by the respective/other Departments prior to each semester.

SEMESTER I

36th BoS Date: 30-July-2026 Academic Batch: 2026 and Onwards

Problem Solving Using C			
Course Code:	**CAI101**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**1U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To provide knowledge of primary and derived data types used in C
2. To make them understand basic conditional and break statements used in C
3. To provide a basic understanding of pointers and pointers arithmetic
4. To enable the students to explore how pre-defined functions are used and also created in a program
5. Learn the difference between static and dynamic memory allocation methods and also learn various dynamic memory allocation methods.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Understand the basic building blocks of C language like tokens, identifiers, constants and variables.
2. Acquire knowledge of various conditional and loop statements
3. Judge which data structure to use among arrays, struct and union depending on the application
4. Use pointers and tell the difference between call by value and call by reference.
5. Use dynamic memory allocation to create arrays, structures and union and also perform various operations on them.

UNIT I: INTRODUCTION TO COMPUTER AND PROGRAMMING CONCEPTS

Definition, characteristic, generation of computers, basic components of a computer system, memory, input, output and storage units, high level language and low-level language, Soft- ware: system software, application software, hardware, firmware, Operating System, compiler, interpreter and assembler, linker, loader, debugger, IDE. Introduction to algorithm and flowchart; representation of algorithm using flowchart symbol, pseudo code, basic algorithm de- sign, characteristics of good algorithm, development of algorithm.

UNIT II: INTRODUCTION TO C PROGRAMMING LANGUAGE

Introduction to C programming language, declaring variables, preprocessor statements, arithmetic operators, programming style, keyboard input, relational operators, introduction, feature of C language, concepts, uses, basic program structure, simple data types, variables, constants, operators, comments, control flow statement: if, while, for, do-while, switch.

UNIT III: DATA TYPES AND STRUCTURES

bitwise operators, Predefined and User defined data types, arrays, declaration and operations on arrays, searching and sorting on arrays, types of sorting, 2D arrays, passing 2D arrays to functions, structure, member accessing, structure and union, array of structures, functions, declaration and use of functions, parameter passing, recursion.

UNIT IV: FUNDAMENTALS OF POINTERS

Introduction to pointers, pointer notations in C, Declaration and usages of pointers, operations that can be performed on computers, use of pointers in programming exercises, parameter passing in pointers, call by value, call by references, array and characters using pointers, dynamic memory allocation.

UNIT V: FILE HANDLING IN C AND ENUM

Introduction to file handling, file operations in C, defining and opening in file, reading a file, closing a file, input output operations on file, counting: characters, tabs, spaces, file opening modes, error handling in input/output operations, Enumerated data types, use of Enum, declaration of Enum.

Text Books:

1. **Herbert Schildt**, *C: The Complete Reference*, McGraw Hill Education, Latest Edition.
2. **Brian W. Kernighan and Dennis M. Ritchie**, *The C Programming Language* (2nd Edition), Pearson Education, 1988.
3. **E. Balagurusamy**, *Programming in ANSI C*, Tata McGraw Hill Education, Latest Edition.
4. **Alan R. Feuer**, *The C Puzzle Book: Puzzles for the C Programming Language*, Prentice Hall, 1982.
5. **Peter Van Der Linden**, *Expert C Programming: Deep C Secrets*, Dorling Kindersley (India), Latest Edition.

Electronics and Electrical Engineering			
Course Code:	**ECE101**	**Course Credits:**	**3**
Course Category:	**ESC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**1U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To equip students with foundational knowledge of electrical laws and circuit theorems for analysing and designing resistive and reactive networks under DC and AC conditions.
2. To introduce key concepts of electromechanical energy conversion and guide learners in understanding the operation, characteristics, and industrial applications of electrical machines and systems.
3. To develop conceptual clarity on semiconductor materials, doping mechanisms, and PN junction behaviour to support deeper engagement with device-level electronics.
4. To enable students to apply diode properties in designing circuits for signal processing, rectification, and voltage control, integrating theoretical knowledge with practical implementation.
5. To empower learners with the analytical skills to assess and optimize the performance of BJTs, MOSFETs, and Op-Amps in amplifier and feedback configurations within electronic systems.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Apply core electrical laws and theorems to analyze and design DC and AC circuits.
2. Interpret the principles and performance characteristics of electrical machines and power systems, including transformers, motors, and generators.
3. Demonstrate understanding of semiconductor physics including carrier movement, doping, and PN junction behaviour.
4. Design and evaluate diode-based circuits for signal conditioning, rectification, and voltage regulation.
5. Analyse advanced semiconductor devices such as BJTs, MOSFETs, and Op-Amps in various amplifier configurations and control systems.

UNIT I: Fundamentals of Electrical Engineering (Core Concepts & Circuits)

Basics of Charge, Current, Voltage, Power, Energy& Efficiency, Ohm's Law: Practical Relevance & Limitations, Kirchhoff's Laws (KCL & KVL): Network Solving Techniques, Superposition, Thevenin, Norton, and Maximum Power Transfer Theorems.

UNIT II: AC Circuits, Power Systems & Electrical Machines

Sinusoidal AC Signals – Representation and Properties (RMS, Average, Peak Values, Form Factor); Introduction to Single-phase and Three-phase Systems, Power Calculation: Active, Reactive, Apparent Power, Phase Angle, Power Factor and its Improvement Techniques; Transformers: Construction, EMF Equation, Efficiency, Applications; Introduction to Electrical Machines: DC (series and shunt) and AC Machines (single phase &3 phase induction motor, synchronous generator).

UNIT III: Basic Semiconductor Electronics

Metals, Insulators and Semiconductor materials, Intrinsic and Extrinsic Semiconductors, Doping, Mobility, Drift Current and Diffusion Current, Current conduction in Semiconductors.

P-N Junction diode, Characteristics and its operation, P-N Junction resistances and capacitances (depletion and diffusion), Breakdowns in PN junction diodes.

UNIT IV: Diode Applications

Clipping and clamping circuits, Rectifier circuits, Zener diode, Zener diode as voltage regulators, voltage multipliers, switching behaviour of P-N diode. Tunnel Diodes, Varactor Diode, Light Emitting Diode.

UNIT V: Semiconductor Devices

Bipolar junction transistor (BJT): Introduction and types, construction, and characteristics in CB, CE & CC mode.

Metal Oxide Semiconductor Field Effect Transistor (MOSFET): Introduction and types, construction, input and transfer characteristics.

Operational Amplifiers (Op-Amps): Op-Amp, Practical Op-Amp, Open loop and closed loop configurations, Applications of Op-Amps as inverting and non-inverting amplifier.

Text Books:

1. P.V. Prasad, S.Sivanagaraju, "Electrical Engineering: Concepts and Applications," Cengage,
2. 2018.
3. D. P. Kothari and I. J. Nagrath, "Basic Electrical Engineering", Tata McGraw-Hill, 2010.
4. Robert L. Boylestad / Louis Nashelsky, "Electronic Devices and Circuit Theory", Pearson
5. Education.
6. George Kennedy, "Electronic Communication Systems", McGraw-Hill Publication.
7. L. S. Bobrow, "Fundamentals of Electrical Engineering", Oxford University Press, 2011.
8. V. D. Toro, "Electrical Engineering Fundamentals", Pearson India, 1989.
9. David A. Bell, "Electronic Devices and Circuits", Oxford University Press.
10. Jacob Millman, C.C. Halkias, StayabrataJit, "Electronic Devices and Circuits", McGraw-Hill.

Engineering Physics			
Course Code:	**PH102**	**Course Credits:**	**4**
Course Category:	**BSC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**1U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 01**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 15**	**End Sem. Exam Hours:**	**3**

Module I: Optics Interference: Coherent sources; Conditions for interference; Division of wavefront: Young's double-slit experiment, Fresnel's bi-prism; Division of amplitude: Uniform and wedge-shaped films; Newton's rings method **Diffraction:** Difference between interference and diffraction; Fresnel and Fraunhofer diffractions; Fraunhofer diffraction by single slit and double slit; Diffraction grating and Resolving power. **Polarization:** Polarization of light: Brewster's law and Malus law; Concept of double refraction by uni-axial crystals; Polariods; Nicol prism; Quarter and half wave plate.

Module II: Electromagnetic (EM) Theory: Vector algebra; Coordinate systems, Gauss divergence theorem, Stokes theorem; Maxwell's equations: EM wave equations in differential and integral forms; Transverse nature and speed of EM waves; EM energy density and Poynting vector.

Module III: Relativity: Frame of reference; Special theory of relativity; Lorentz transformation; Length contraction and time dilation; Twin paradox; Doppler's effect; Mass and energy equivalence; Massless particles.

Module IV: Quantum Mechanics: Origin of quantum theory; Photo-electric effect; Compton Effect; Dual nature of light; de-Broglie waves; Davisson Germer experiment; Phase and group velocities; Uncertainty principle; Quantum mechanical wave function; Schrodinger wave equation; Particle in a box (1D); Tunnel effect (Qualitative).

Module V: Solid State Physics: Brief discussion of solids, Crystals, and bonds; Band theory of solids; X-ray diffraction (Bragg's law). **Lasers**: Introduction to Laser (He-Ne laser and Ruby laser) **Nanotechnology:** Properties of nanoparticles; Carbon nanotubes; Applications.

Text Books:

1. H. K. Malik & A. K. Singh, Engineering Physics, Tata McGraw Hill Education Pvt. Ltd.
2. Satya Prakash & V. Saluja, Engineering Physics, Pragati Prakashan.
3. D. J. Griffiths, Introduction to Electrodynamics, PHI Learning Pvt. Ltd.
4. Arthur Beiser, Concepts of Modern Physics, Tata McGraw-Hill Edition.
5. G. Aruldhas, Engineering Physics, PHI Learning Pvt. Ltd.
6. Halliday and Resnick's, Principles of Physics, Wiley.

Applied Mathematics-I			
Course Code:	MA111	Course Credits:	4
Course Category:	BSC	Course (U / P)	U
Course Year (U / P):	1U	Course Semester (U / P):	1U
No. of Lectures + Tutorials (Hrs/Week):	03 + 01	Mid Sem. Exam Hours:	1.5
Total No. of Lectures (L + T):	45 + 15	End Sem. Exam Hours:	3

UNIT I: Elementary transformations, linear dependence and independence of vectors, rank of a matrix (echelon & normal form), inverse of a matrix by elementary operations; solution of non-homogeneous and homogeneous systems of linear equations.

UNIT II: Linear transformations; eigenvalues and eigenvectors, diagonalization of a matrix; Cayley-Hamilton's theorem (without proof) and its applications.

UNIT III: Successive differentiation, Leibnitz theorem; functions of several variables: limits, continuity and differentiability, partial differentiation; Euler's Theorem for homogeneous functions; composite functions, total derivatives, change of variables; Taylor's and Maclaurin's Series; maxima and minima of functions of two variables.

UNIT IV: Double integration in cartesian and polar coordinates; evaluation of double integrals by substitution and changing the order of integration; triple integral; applications of multiple integrals to find area as double integral, volume as triple integral and surface area.

Text Books:

1. Jain R. K. and Iyengar S. R. K., "Advanced Engineering Mathematics", 5th Edition, Narosa Publishing House Pvt. Ltd. 2016.
2. Kreyszig E., "Advanced Engineering Mathematics", 10th Edition, John Wiley & Sons, 2010.
3. Greenberg M., "Advanced Engineering Mathematics", 2nd Edition, Pearson Education, 1998.
4. Thomas, G. B., Weir, M. D., & Hass, J. (2018). Thomas' calculus: Multivariable (14th ed.). Pearson.
5. Apostol, T. M. (1969). Calculus, Volume II: Multi-variable calculus and linear algebra, with applications to differential equations and probability (2nd ed.). John Wiley & Sons.

Career Skills			
Course Code:	**ICT101**	**Course Credits:**	**2**
Course Category:	**BSC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**1U**
No. of Lectures + Tutorials (Hrs/Week):	**02 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**30**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 To assess a student's overall cognitive ability.
2. Evaluating their capacity to learn, reason logically.
3 Solve problems across various domains.
COURSE OUTCOMES
At the end of the course the students should be able to:
1. Improved problem-solving skills.
2. Enhanced critical thinking abilities.
3. Better analytical skills.
4. Better decision-making under pressure.

UNIT I – Number System and Work Efficiency

Number System: Divisibility Rules, Remainder Theorem, Unit Digit, Series, Number of Factors, Simplification, LCM and HCF.

Time and Work: Efficiency, Ratio of Work Done, Work Done in Given Time, Combined Work of Multiple People, Individual Work Rates, and Problems Involving Days and Hours.

UNIT II – Time and Distance

Basic Concepts, Average Speed, Relative Speed, Time and Distance Conversions, Problems Involving Uniform Motion, and Different Scenarios like Trains Crossing Each Other or a Man Walking on a Moving Train.

UNIT III – Percentage, Profit & Loss, Interest

Percentage: Basic Percentage Calculations, Percentage Increase and Decrease, Finding Given Percentage of a Number, Percentage Relationship Between Two Numbers.

Profit and Loss: Cost Price, Selling Price, Marked Price, Profit Percentage, Loss Percentage, and Calculation of Profit or Loss.

Simple Interest: Calculating the Interest Earned on a Principal Amount at a Fixed Rate Over a Specific Period of Time, Understanding the Formula, Converting Time Units, Calculating Simple Interest, Finding the Principal Amount.

Compound Interest: Basic Formula, Different Compounding Frequencies, Comparison with Simple Interest.

UNIT IV – Logical and Analytical Reasoning

Number Series, Alphabet Series, Analogy and Classification, Blood Relations, Direction Sense, Coding-Decoding, Ranking and Order, Syllogisms, Puzzles (Seating Arrangement and Scheduling), Statement and Conclusion, Cause and Effect, and Data Sufficiency.

Text Books:

1. Magical Book on Quicker Maths by M. Tyra
2. Quantitative Aptitude for Competitive Examinations by R.S. Aggarwal
3. A Modern Approach to Logical Reasoning by R.S. Aggarwal
4. Advance Maths by Rakesh Yadav
5. SSC Elementary and Advanced Maths BY Kiran

Engineering Physics Lab			
Course Code:	**PH104**	**Course Credits:**	**1**
Course Category:	**BSC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**1U**
No. of Labs (Hrs/Week):	**1 (2 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

List of Experiments

1. Measurement of basic constants: Length, Weight & Time.
2. To Study of force acting on a current carrying conductor in a magnetic field and verify the Lorentz force $F = ILB \sin \Theta$
3. To study the magnetic field variation of paired coils in a Helmholtz arrangement.
4. To study Interference and diffraction of Light using slits.
5. To calculate the wavelength of sodium light using Fresnel's Bi-prism.
6. To study the interference of light by Fresnel's Bi-prism and find the fringe width. (using laser source)
7. To determine the wavelength of sodium light by Newton's Rings.
8. To determine the Cauchy's constants using Prism and Spectrometer.
9. To find wavelength of Mercury light source by using Plane Transmission Diffraction Grating.
10. To study the Polarization of light and verify Malus's Law.
11. Study of Photoelectric effect and calculation of Planck's Constant.
12. To determine the energy band gap of a given semiconductor material using four Probe method.
13. To find the e/m ratio of electron by Thomson's method.
14. To determine the fill factor of Solar Cell.
15. To determine the specific rotation of sugar using half shade polarimeter.
16. Study of Coupled Pendulum.

ICT Workshop I			
Course Code:	**ICT181**	**Course Credits:**	**1**
Course Category:	**ESC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**1U**
No. of Labs (Hrs/Week):	**1 (2 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Enable students to apply the Design Thinking methodology—empathy, ideation, prototyping, and testing—to real-world challenges, with a focus on customer-centric solutions for digital technology and smart systems.
2. Cultivate students' ability to think critically and analytically by identifying barriers to effective problem solving, recognizing the human factors influencing decisions, and evaluating arguments using logic and evidence-based reasoning.
3. Encourage students to explore and utilize open-source and freeware tools in CAD, circuit design, programming, and embedded systems via DIY projects, fostering hands-on innovation and real-life application of Engineering skills.
COURSE OUTCOMES
1. At the end of the course the students should have understood and should be able to apply their learned knowledge in real life applications.
2. Identify common obstacles to effective problem solving and decision making and recognize the human variable M problem solving and decision making.
3. Apply concepts to enhancing personal development and organizational performance and explain the key elements of problem solving and decision making, and the barriers associated with them

In today's increasing complexity of digital technology and modem business, customers am increasingly choosing products and services based on the quality of the experiences they have w. them. To help meet these challenges, an approach known as "Design Thinking, playing a great role in finding meaningful pathways - its process and tools arc increasingly being adopted in Lean Six Sigma processes and in organizational innovation initiatives. Design thinking is a human-centered, iterative problem-solving process of discovery, ideation, and experimentation that employs various design-based techniques to gain insight and yield innovative solutions for virtually any type of organizational or business challenge. A Design Thinking mindset is essential for development of Internet of Things (AT) platform, smart products, and Smart Cities. Industry practitioners of Design Thinking include Apple, Google, Samsung, Libor, Airbnb, IDEO, Nike, Procter 8: Gamble, Singapore Airlines, DBS Bank to name a few.

In this action-oriented workshop-oriented course, students will work in teams (5.6), guided by a facilitator to experience a customer-centric approach A problem solving through re-imagination of the end-to-end customer experience journey. Students will develop skills such as ethnographers, visual thinkers, strategists, and storytellers through a hybrid of workshop discussions and activities. It covers building empathy through ethnographic research, generating ideas, prototyping, and testing new concepts. The goal of this course is that students acquire Design Thinking skills. This is an experiential teaming course where students learn by doing. Nowadays, Design Thinking and its tools are used by product and industrial design firms to ideate products. It is also used to solve so-called "wicked problems" — problems for which neither question nor the answer is well defined. In addition, students are expected to sharpen their Critical Reasoning and problem-solving skills through an online MOOC on Courses & Critical reasoning is a methodology for using objective and evidence-(rased analysis for evaluation of statements and arguments. It also helps understand the fallacies and barriers that prevent reaching conclusions in a logical and cohesive manner.

Students are encouraged to explore by any Open-Source Software or Freeware software's related to (But not limited to) CAD (Computer-Aided Design), Circuit Design, PCB Design, Programming

language, Robotics, Embedded System Development, Documentation) through (do-it-yourself) DIY projects.

List of Experiments:

1. To Understand the basic of computer architecture and organization
2. To Understand basic of Network Topologies
3. To Understand and discuss Open-source Embedded board Development and Working
4. To Understand and discuss different Sensors
5. To Understand the Spice/Circuit Software

Textbook:

Tim Browm Change by Design; Harper Collins (2009)

Problem Solving Using C Lab			
Course Code:	**CAI181**	**Course Credits:**	**1**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**1U**
No. of Labs (Hrs/Week):	**1 (2 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To introduce students to the basic knowledge of programming fundamentals of C language.
2. To impart writing skill of C programming to the students and solving problems.
3. To impart the concepts like looping, array, functions, pointers, file, structure.
4. Write programs to print output on the screen as well as in the files.
5. Apply all the concepts that have been covered in the theory course.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Recognize and understand the syntax and construction of C programming code
2. Able to design and develop Computer programs, analyzes, and interprets the concept of pointers, declarations, initialization, operations on pointers and their usage.
3. Able to define data types and use them in simple data processing applications also he/she must be able to use the concept of array of structures.
4. Students must be able to define union and enumeration user defined data types.
5. Develop confidence for self-education and ability for life-long learning needed for Computer language.

LIST OF EXPERIMENTS:

1. Write a C program to print your name, age, and address using basic I/O functions.
2. Write a program to perform arithmetic operations (addition, subtraction, multiplication, division, modulus) on two integers entered by the user.
3. Implement a program to check whether a given number is prime or not using if-else.
4. Write a program using a loop to print the factorial of a number.
5. Write a program to create a simple calculator using switch-case.
6. Use nested for loops to display a multiplication table up to 10x10.
7. Write a C program to add two 3x3 matrices and display the result.
8. Define a structure for student (roll, name, marks) and write a program to accept and display data for 5 students.
9. Write a function to calculate the average marks using structures and functions.
10. Use malloc() and free() to dynamically allocate memory for storing n integers and compute their sum.
11. Write a program to read a text file and count the number of characters, spaces, tabs, and lines.

Electronics and Electrical Engineering Lab			
Course Code:	**ECE185**	**Course Credits:**	**1**
Course Category:	**ESC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**1U**
No. of Labs (Hrs/Week):	**1 (2 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Familiarize themselves with electronic lab instrumentation, such as CROs, multimeters, function generators, and power supplies.
2. Apply fundamental electrical principles to verify circuit laws and theorems through hands-on experiments.
3. Analyze and interpret behaviors of AC circuits, including resonance and power measurements in three-phase systems.
4. Explore semiconductor device characteristics, including diodes, transistors, and rectifiers, under various biasing conditions.
5. Implement analog circuit configurations using operational amplifiers for arithmetic operations and transistor configurations for amplification modes.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Demonstrate effective use of laboratory instruments (CRO, multimeter, function generator, power supply) and identify active/passive components for circuit analysis.
2. Verify foundational electrical theorems and laws such as KCL, KVL, and Thevenin's Theorem through experimental setups.
3. Analyze the resonance behavior of RLC circuits and measure relevant parameters, including resonance frequency and quality factor.
4. Examine diode-based circuits, including PN junction and Zener characteristics, and determine rectifier performance metrics (Vrms, Vdc, ripple factor).
5. Explore operational amplifier configurations and transistor characteristics in CB, CE, and CC modes to understand analog signal processing.

List of Experiments:

1. Study of Lab Equipment and Components: CRO, Multi-meter, Function Generator, Power Supply; Active, Passive Components, and Bread Board.
2. Study and verify Kirchhoff's Current Law (KCL) and Krichoff's Voltage Law (KVL).
3. Study and verify Thevenin's Theorem.
4. Study and verify Resonance in series and parallel R-L-C series circuit, and measurement of resonance frequencies.
5. Measurement of power in 3- three-phase circuit by the two-wattmeter method and determination of its power factor for star as well as delta-connected loads.
6. Study the PN junction diode characteristics under Forward & Reverse bias conditions and determine the static and dynamic resistance.
7. Study the Half & Full wave rectifier circuit and determine: V_{dc}, V_{ac}, and ripple factor.
8. Study and plot the characteristics of a Zener diode and its breakdown voltage.
9. Study the applications of Operational Amplifier (LM741) as an Adder and a Subtractor.
10. Study and plot the input and output transistor characteristics in Common Base (CB), Common Emitter (CE), and Common Collector (CC) configurations.

SEMESTER II

Python Programming			
Course Code:	**CAI102**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**2U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

1. Master the fundamentals of writing Python scripts.
2. Learn core Python scripting elements such as variables and flow control structures.
3. Discover how to work with lists and sequence data.
4. Write Python functions to facilitate code reuse.
5. Use Python to read and write files.

COURSE OUTCOMES

At the end of the course the students should be able to:

1. Problem solving and programming capability.
2. Explain basic principles of Python programming language
3. Implement database and GUI applications.
4. Implement object oriented concepts
5. Define and demonstrate the use of built-in data structures "lists" and "dictionary"

UNIT I: PYTHON BASSICS, CONDITIONAL & LOOPS

Installation of Python and python Notebook, Python Objects, Number & Booleans, Strings, Container objects, Mutability of objects, Operators - Arithmetic, Bitwise, comparison and Assignment operators, Operators Precedence and associativity. Conditions (If else, if-elif- else), Loops (While ,for), Break and Continue statements, Range Functions.

UNIT II: STRING OBJECTS AND LIST OBJECTS

String object basics, String methods, Splitting and Joining Strings, String format functions, list object basics, list methods, List as stack and Queues, List comprehensions.

UNIT III: TUPLES, SET, DICTIONARIES & FUNCTIONS

Tuples, Sets, Dictionary Object basics, Dictionary Object methods, Dictionary View Objects. Functions basics, Parameter passing, Iterators, Generator functions, Lambda functions, Map, Reduce, filter functions.

UNIT IV: OOPS CONCEPTS & WORKING WITH FILES

OOPS basic concepts, creating classes and Objects, Inheritance, Multiple Inheritance, working with files, Reading and writing files, Buffer read and write, Other File methods.

UNIT V: MODULES, EXCEPTION HANDLING & DATABASE PROGRAMMING

Using Standard Module, Creating new modules, Exceptions Handling with Try-except, Creating, inserting and retrieving Table, Updating and deleting the data. Data Analysis- Numpy variable, Numpy manipulation, Skippy, Pandas intro. Descriptive analysis, Pandas Input-output, Pandas manipulation, Pandas group by.

Text Books:

1. Head First Python 2e: A Brain-Friendly Guide Paperback – Illustrated, 16 by Paul Barry, Oreilly
2. Python: The Complete Reference Paperback – 20 March 2018 by Martin C. Brown (Author), TMH Publication
3. Let Us Python by YashavantKanetkar , 1 January 2019, BPB publication
4. Python Programming, A modular approach , First Edition, By Pearson Publication by Taneja Sheetal and Kumar Naveen , 26 September 2017

Applied Mathematics II			
Course Code:	**MA112**	**Course Credits:**	**4**
Course Category:	**BSC**	**Course (U/ P)**	**U**
Course Year (U/ P):	**1U**	**Course Semester (U/ P):**	**2U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 01**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L +T):	**45 + 15**	**End Sem. Exam Hours:**	**3**

UNIT I: Concept of probability, additive and multiplicative law of probability, total and conditional probabilities, Baye's theorem. Definition and properties of random variables, discrete and continuous random variables, probability mass and density functions, distribution functions.

UNIT II: Concepts of bi-variate random variables-Joint, marginal and conditional distributions. Transformations of one and two-dimensional random variables. Mathematical expectation-Definition and its properties. Variance, standard deviation, Covariance. Moment generating function- Definition and their properties.

UNIT III: Discrete distributions-Binomial, Poisson and geometric distributions with their properties. Continuous distributions- Uniform, Exponential and Normal distributions with their properties.

UNIT IV: Linear Correlation, Correlation Coefficient, Rank Correlation Coefficient, Regression. Central limit theorem.

Text Books:

1. Montgomery, Douglas C., and George C. Runger. "Applied Statistics and Probability for Engineers", Seventh Edition. John Wiley & Sons, 2018.
2. Sheldon Ross M., "Introduction to Probability and Statistics for Engineers and Scientists", Academic Press, 6thEdtion, 2020.
3. Devore, Jay L. "Probability and Statistics for Engineering and the Sciences", 8th Edition, Cengage, 2010.
4. Scheaffer, Richard, Madhuri Mulekar, and James McClave. "Probability and Statistics for Engineers". Nelson Education, 2010.
5. A. M. Mood, Graybill and Boes, "Introduction to the Theory of Statistics". McGraw Hill, 3rd edition.

Introduction to Artificial Intelligence			
Course Code:	**CAI104**	**Course Credits:**	**2**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**2U**
No. of Lectures + Tutorials (Hrs/Week):	**02 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**30**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Provide a strong foundation of fundamental concepts in Artificial Intelligence
2. Enable the student to apply these techniques in applications which involve perception, reasoning and learning
3. Provide a basic exposition to the goals and methods of Artificial Intelligence
4. Explain the role of agents and how it is related to environment and the way of evaluating it and how agents can act by establishing goals.
5. Learn the different machine learning techniques to design AI machine and enveloping applications for real world problems.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Understand the various searching techniques, constraint satisfaction problem and example problems- game playing techniques.
2. Apply these techniques in applications which involve perception, reasoning and learning
3. Acquire the knowledge of real-world Knowledge representation
4. Analyze and design a real-world problem for implementation and understand the dynamic behavior of a system.
5. To enable the student to apply these techniques in applications which involve perception, reasoning and learning

UNIT I: Introduction

Introduction to Artificial Intelligence, History and Evolution of AI, Goals and Characteristics of AI, Types of AI (Narrow AI, General AI, Super AI), Applications of AI, Intelligent Systems, Turing Test, Rationality and Intelligence, AI Ethics and Responsible AI, Explainable AI, AI Safety and Limitations, Weak AI vs Strong AI, Overview of AI Paradigms: Symbolic AI, Statistical AI.

UNIT II: Intelligent Agents and Problem Solving

Intelligent Agents, Agent Architecture, Rational Agents, PEAS Representation, Environment Types, Types of Agents: Simple Reflex, Model-Based, Goal-Based, Utility-Based, Learning Agents, Agent Communication, Multi-Agent Systems (Introduction), Knowledge-Based Agents, Problem Formulation, State Space Representation.

UNIT III: Search Algorithms and Problem Solving

Problem Solving by Searching, State Space Search, Uninformed Search: Breadth First Search, Depth First Search, Uniform Cost Search; Informed Search: Best First Search, Greedy Best First Search, A* Search, Hill Climbing, Beam Search; Adversarial Search: Minimax Algorithm, Alpha-Beta Pruning; Introduction to Constraint Satisfaction Problems; Classical AI

Problems and Applications: Route Finding, Puzzle Solving, Game Playing, Water-Jug Problem, Tic-Tac-Toe.

UNIT IV: Knowledge Representation and Reasoning

Knowledge Representation, Logic-Based Representation, Semantic Networks, Frames, Production Rules, Ontologies, Knowledge Engineering: Knowledge Acquisition, Representation, Storage, Knowledge Base Design, Knowledge-Based Systems, Reasoning: Forward Chaining, Backward Chaining, Rule-Based Reasoning, Uncertainty in AI, Introduction to Bayesian Reasoning, Fuzzy Logic.

UNIT V: AI Techniques and Applications

Introduction of Machine learning, Introduction of Deep Learning, Introduction to Expert system, Introduction to Natural Language Process, AI in future, AI in social media, AI in Entertainment and Education, AI in Drones, AI in Automated customer support, AI in Personalized shopping Experience, AI in Finance, AI in Smart cars, AI in Travel and Navigation, AI in Smart home Devices, AI in Security and Surveillance, AI in Education, AI in Health Care, AI in E-Commerce.

Text Books:

1. Artificial Intelligence, Russel and Norvig, Pearson Prentice Hall Publication, 2006.
2. Artificial Intelligence, Elanie Reich: Tata McGraw-Hill Publishing House, 2008.
3. Artificial Intelligence, Ela Kumar, IK Publishing.
4. Artificial intelligence, Peterson, Tata McGraw-Hill, 2008.
5. Artificial Intelligence, Winston, PHI publication, 2006.
6. Artificial Intelligence – A Modern Approach (3rd Edition) By Stuart Russell & Peter Norvig 7. Artificial Intelligence: The Basics By Kevin Warwick

Discrete Mathematics			
Course Code:	**ICT102**	**Course Credits:**	**3**
Course Category:	**BSC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**2U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Simplify and evaluate basic logic statements including compound statements, implications, inverses, converses, and contrapositives using truth tables and the properties of logic.
2. Express a logic sentence in terms of predicates, quantifiers, and logical connectives
3. Apply the operations of sets and use Venn diagrams to solve applied problems; solve problems using the principle of inclusion-exclusion.
4. Determine the domain and range of a discrete or non-discrete function, graph functions, identify one-to-one functions, perform the composition of functions, find and/or graph the inverse of a function, and apply the properties of functions to application problems.
5. Apply rules of inference, tests for validity, proof by contradiction, proof by cases, and mathematical induction and write proofs using symbolic logic and Boolean Algebra.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Students will be able to express a logic sentence in terms of predicates, quantifiers, and logical connectives
2. Students will be able to apply the rules of inference, proof by contradiction, and mathematical induction
3. Students will be able to evaluate Boolean functions and simplify expressions using the properties of Boolean algebra
4. Students will be able to learn about predicates, quantifiers, and logical connectives
5. Student will be able to use tree and graph algorithms to solve problems

UNIT I: MATHEMATICAL LOGICS

Statements and notations, connectives, well-formed formulas, truth tables, tautology, equivalence implication, normal forms, predicates: predicative logic, free & bound variables, rules of inference, consistency, proof of contradiction, automatic theorem proving, Boolean Algebra: Introduction, Theorems of Boolean algebra, Algebraic manipulation of Boolean expressions. Simplification of Boolean Functions.

UNIT II: SET THEORY

Set Theory: Introduction, Combination of sets, Multi sets, ordered pairs, Set Identities, Properties of binary relations, equivalence relation, compatibility and partial ordering relations, Hasse diagram. functions: definition, types of function, inverse function, lattice and its properties, algebraic structures, semi groups, monads, groups, sub groups" homomorphism, isomorphism.

UNIT III: ELEMENTARY COMBINATORICS

Basis of counting, combinations & permutations, with repetitions, constrained repetitions, binomial coefficients, multinomial theorem, the principles of inclusion – exclusion, pigeon hole principles and it's application.

UNIT IV: RECURRENCE RELATION

Generating functions, function of sequences calculating coefficient of generating function, recurrence relations, solving recurrence relation by substitution and generating funds, characteristics roots solution of in homogeneous recurrence relation.

UNIT V: GRAPH THEORY

Graph theory and its applications, representation of graph (adjacency matrix and adjacency list), types of graphs, degree sequence of graph,Trees: Definition, Binary tree, spanning trees, planar graphs, bipartite graph, basic concepts isomorphism, multi graphs, euler graph and hamiltonian graph, chromatic numbers.

Text Books:

1. Discrete and Combinational Mathematics- An Applied Introduction-5th Edition Ralph. P.Grimaldi, Pearson Education
2. Discrete Mathematical Structures with applications to computer science Trembly J.P. & Manohar.P, TMH
3. Discrete Mathematics and its Applications, Kenneth H. Rosen, Fifth Edition.TMH.
4. Discrete Mathematical structures Theory and application-Malik & Sen
5. Discrete Mathematics for Computer science, Garry Haggard and others, Thomson.
6. Logic and Discrete Mathematics, Grass Man & Trembley, Person Education

Environmental Studies			
Course Code:	**ES101**	**Course Credits:**	**4**
Course Category:	**BSC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**1U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 01**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 15**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVE

To impart knowledge on environment and environmental issues and challenges of local, national and global significance for achieving environmental security and sustainable living

COURSE OUTCOME

To knowledge and awareness so generated will enhance ability of the learners for conservation of environment and natural resources for a healthy planet Earth, and happy living of the present and future generations.

UNIT I: Introduction to Environmental Studies (2 lectures)

- Multidisciplinary nature of environmental studies; components of the Earth's environment-atmosphere, hydrosphere, lithosphere and biosphere
- Scope and importance; Concept of sustainability and sustainable development

UNIT II: Ecosystems (6 lectures)

- What is an ecosystem? Structure and function of ecosystem; Energy flow in an ecosystem: food chain, food web and ecological succession. Case studies of the following ecosystems:
  - **a)** Forest ecosystem
  - **b)** Grassland ecosystem
  - **c)** Desert ecosystem
  - **d)** Aquatic ecosystems (pond, stream, lake, river, ocean, estuary)

UNIT III: Natural Resources : Renewable and Non--renewable Resources (8 lectures)

- Land resources and land-use changes; Land degradation, soil erosion and desertification
- Deforestation: Causes and impacts due to mining and dam building on environment, forest, biodiversity and tribal population
- Water : Use and over-exploitation of surface and ground water, floods, droughts, conflicts over water (international and inter-state)
- Heating of Earth and circulation of air; air mass formation and precipitation
- Energy resources: Renewable and non-renewable energy resources, use of alternate energy sources, growing energy needs, case studies

UNIT IV: Biodiversity and Conservation (8 lectures)

- Levels of biological diversity : genetic, species and ecosystem diversity; Bio-geographic zones of India; Biodiversity patterns and global biodiversity hot spots
- India as a mega-biodiversity nation; Endangered and endemic species of India
- Threats to biodiversity: Habitat loss, poaching of wildlife, human-wildlife conflicts, biological invasion; Conservation of biodiversity: In-situ and Ex-situ conservation of biodiversity
- Ecosystem and biodiversity services: Ecological, economic, social, ethical, aesthetic and informational value

UNIT V: Environmental Pollution (8 lectures)

- Environmental pollution : Types, causes, effects and control; Air, water, soil and noise pollution
- Nuclear hazards and human health risks
- Solid waste management: Control measures of urban and industrial wastes
- Pollution-related case studies

UNIT VI: Environmental Policies and Practices (7 lectures)

- Climate change, global warming, ozone layer depletion, acid rain and impacts on human communities and agriculture
- Environment Laws: Environment Protection Act; Air (Prevention and Control of Pollution) Act; Water (Prevention and Control of Pollution) Act; Wildlife (Protection) Act; Forest (Conservation) Act; International agreements: Montreal and Kyoto protocols, and Convention on Biological Diversity (CBD)
- Nature reserves, tribal population and rights and human-wildlife conflicts in Indian context

UNIT VII: Human Communities and the Environment (6 lectures)

- Human population growth: Impacts on environment, human health and welfare, Carbon foot print
- Resettlement and rehabilitation of project-affected persons; case studies
- Disaster management : Floods, earthquakes, cyclones and landslides
- Environmental movements : Chipko, Silent valley, Bishnois of Rajasthan
- Environmental ethics: Role of Indian and other religions and cultures in environmental
- conservation
- Environmental communication and public awareness, case studies (e.g., CNG vehicles in Delhi)

UNIT VIII: Field Work (Equal to 5 lectures)

- Visit to an area to document environmental assets: river/ forest/ flora/fauna, etc.
- Visit to a local polluted site--Urban/Rural/Industrial/Agricultural
- Study of common plants, insects, birds and basic principles of identification
- Study of simple ecosystems--pond, river, Delhi Ridge, etc.

Text Books:

1. Carson, R., 2002, Silent Spring, Houghton Mifflin Harcourt, Boston.
2. Gadgil, M., and Guha, R. 1993. This Fissured Land: An Ecological History of India, University California Press, California.
3. Gleeson, B. and Low, N. (Eds.) 1999. Global Ethics and Environment, Routledge, London.
4. Gleick, P. H. 1993. Water in Crisis. Pacific Institute for Studies in Development, Environment and Security. Stockholm Environmental Institute, Oxford University Press, Oxford.
5. Groom, M.J., Meffe, G.K. and Carroll, C.R. 2002 Principles of Conservation Biology, Sinauer Associates, Sunderland.
6. Grumbine, R. E., and Pandit, M.K., 2013. Threats from India's Himalayan dams, Science 339: 36-37.
7. McCully, P.,1996. Rivers No More: The Environmental Effects of Dams, Zed Books, London.
8. McNeill, J. R, 2000. Something New Under the Sun: An Environmental History of the Twentieth Century, Norton, New York.

9. Odum, E.P., Odum, H.T. and Andrews, J., 1971, Fundamentals of Ecology, Saunders, Philadelphia.

10. Pepper, I.L., Gerba, C.P. and Brusseau, M.L. 2011, Environmental and Pollution Science, Academic Press, New York.

11. Rao, M.N. and Datta, A.K., 1987. Waste Water Treatment, Oxford and IBH Publishing Co. Pvt. Ltd, New Delhi.

12. Raven, P.H., Hassenzahl, D.M. and Berg, L.R., 2012. Environment, 8th Edition, John Wiley and Sons, New York.

13. Rosencranz, A., Divan, S., and Noble, M. L. 2001. Environmental Law and Policy in India, Oxford University Press, New Delhi.

14. Sengupta, R., 2003. Ecology and Economics: An Approach to Sustainable Development, Oxford University Press, New Delhi.

English Proficiency			
Course Code:	**EN101**	**Course Credits:**	**2**
Course Category:	**HSMC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**2U**
No. of Lectures + Tutorials (Hrs/Week):	**02 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**30**	**End Sem. Exam Hours:**	**3**

Unit 1 Functional Grammar: Form and Functions; Sentences: Simple, Complex, and Compound, Tense, Mood, an Aspect; Sub-Verb Agreement and Concord; Common Errors Vocabulary Building Inflection and Derivation, Conversions, Idioms and Phrases, Words in Context

Unit 2 Language Skills (LSRW): Listening Skills Activity-based. Speaking Skills: Activity-based, Introduction to IPA, Use of Dictionary, Word stress, Reading Skills: Skimming and Scanning, Reading Comprehension, Writing Skills: Paragraph, Précis and Compositions, Note Making and Note Taking, Logical Ordering of Ideas and Contents, Figures of Speech

Unit 3 Learning through thematic texts

- *My Visions for India* — Dr. Abdul Kalam
- *From In an Antique Land* — Amitav Ghosh
- *The Gift of Magi* — O' Henry
- *Master and Man* — Leo N. Tolstoy
- *If* — Rudyard Kipling
- *The Solitary Reaper* — William Wordsworth

Text Books:

1. *Word for Word*, Pointon & Clark, Oxford University Press
2. Carter. Ronald: McCarthy, Michael (2006). *Cambridge Grammar of English. A Comprehensive Guide*. Cambridge University Press.
3. *An English Pronouncing Dictionary,* London. Dent, rpt in facsimile in Jones (2002). 17th edn, P. Roach. Hartman and J. Setter (eds), Cambridge: CUP, 2006.
4. Redman, Stuart. 2011 English Vocabulary 1 Use: Pre-intermediate and intermediate Cambridge CUP *Cambridge Phrasal Verbs Dictionary* Second edition, Cambridge University Press.

Python Programming Lab			
Course Code:	CAI182	Course Credits:	1
Course Category:	PCC	Course (U / P)	U
Course Year (U / P):	1U	Course Semester (U / P):	2U
No. of Lectures + Tutorials (Hrs/Week):	1 (2 HRS)	Mid Sem. Exam Hours:	-
Total No. of Lectures (L + T):	10	End Sem. Exam Hours:	3

COURSE OBJECTIVES

1. Master the fundamentals of writing Python scripts.
2. Learn core Python scripting elements such as variables and flow control structures.
3. Discover how to work with lists and sequence data.
4. Write Python functions to facilitate code reuse.
5. Use Python to read and write files.

COURSE OUTCOMES

At the end of the course the students should be able to:

1. Problem solving and programming capability.
2. Explain basic principles of Python programming language.
3. Implement database and GUI applications.
4. Implement object oriented concepts.
5. Define and demonstrate the use of built-in data structures "lists" and "dictionary".

LIST OF EXPERIMENTS:

1. To install Python and Jupyter Notebook and perform basic programming using Python variables, data types, objects, mutability concepts, and operators.
2. To implement Python programs using conditional statements, looping constructs, range(), break, and continue statements.
3. To perform various string operations using indexing, slicing, string methods, splitting, joining, and string formatting techniques.
4. To create and manipulate Python lists using list methods, implement stack and queue operations, and develop programs using list comprehensions.
5. To implement programs using tuples, sets, dictionaries, dictionary view objects, and user-defined functions with different parameter passing techniques.
6. To develop Python programs using iterators, generators, lambda expressions, and higher-order functions such as map(), filter(), and reduce().
7. To create Python programs using object-oriented programming concepts including classes, objects, inheritance, and multiple inheritance.
8. To perform file handling operations including creating, reading, writing, appending, buffered file operations, and file pointer manipulation using Python file methods.
9. To create user-defined modules, utilize standard Python modules, implement exception handling using try, except, finally, and perform database operations (Create, Read, Update, Delete) using SQLite.
10. To perform numerical computation and data analysis using NumPy, SciPy, and Pandas, including array manipulation, descriptive statistics, data input/output, data manipulation, and GroupBy operations.

ICT Workshop II			
Course Code:	**ICT182**	**Course Credits:**	**1**
Course Category:	**ESC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**2U**
No. of Labs (Hrs/Week):	**1 (2 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Enable students to bridge the gap between theory and practice by designing, implementing, and troubleshooting real-world projects that reinforce and expand on classroom concepts.
2. Encourage students to engage in creative exploration and iterative learning, fostering deeper analytical skills and the ability to independently address complex technical challenges.
3. Promote teamwork, initiative, and effective time management through collaborative project work, while cultivating the autonomy and confidence needed for research, industry roles, or entrepreneurial ventures.

COURSE OUTCOMES
1. Students will be able to integrate and apply scientific or engineering principles to design, build, and test a functional DIY project that addresses a specific problem or objective.
2. Students will develop the ability to identify challenges during the project development process, analyze root causes, and implement effective solutions using appropriate tools and techniques.
3. Students will demonstrate the ability to work collaboratively in teams, document their project development process, and present their findings and results clearly through written reports and oral presentations.

This course encourages students to undertake DIY **(Do-It-Yourself)** projects offers a hands-on approach to learning that significantly enhances their understanding of theoretical concepts. Unlike traditional lectures, DIY projects compel students to actively engage with materials, tools, and techniques, allowing them to directly apply the principles they've learned in class. This experiential learning fosters deeper comprehension, critical thinking, and creative problem-solving—skills that are essential in both academic and professional settings. Through trial and error, students gain a practical appreciation for the intricacies of design, implementation, and troubleshooting, which are often underemphasized in purely theoretical coursework.

Moreover, DIY lab projects cultivate a sense of ownership and initiative in students, motivating them to explore beyond standard curricular boundaries. When students are given the freedom to design and execute their own ideas, they develop a stronger sense of confidence, autonomy, and innovation. These projects also simulate real-world engineering or scientific challenges, encouraging collaboration, project planning, and time management. By bridging the gap between abstract knowledge and tangible outcomes, DIY-focused lab courses prepare students more effectively for research, industry, or entrepreneurship, making them better equipped for success in a rapidly evolving technological landscape.

List of Experiments:

1. Develop a digital resume or portfolio website hosted on GitHub Pages or designed using Canva.
2. create or refine their LinkedIn profile with a professional photo, academic background, projects, skills, and certifications.
3. Students will create a GitHub account, design a professional README for their profile, and upload at least one basic project.
4. Set up a NodeMCU board, connect to Wi-Fi, and send sensor data to a web interface.

5. Record a small demo or presentation of their DIY/academic project and upload it to YouTube or SlideShare with proper description and tags.

Textbook:

1. "Make It: The Engineering & Design Thinking Guide for Teens" by Emily Pilloton
2. "Designing for Growth: A Design Thinking Toolkit for Managers" by Jeanne Liedtka and Tim Ogilvie

YouTube 36.1.1

Career Skills Lab			
Course Code:	**ICT184**	**Course Credits:**	**1**
Course Category:	**HSMC**	**Course (U / P)**	**U**
Course Year (U / P):	**1U**	**Course Semester (U / P):**	**2U**
No. of Labs (Hrs/Week):	**1 (2 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To improve students' communication, presentation, and interpersonal skills.
2. To develop critical thinking, leadership, and team collaboration through interactive sessions.
3. To expose students to entrepreneurial thinking and startup ecosystems.
4. To enhance self-awareness, confidence, and time management skills.
5. To prepare students for interviews, group discussions, and real-world workplace challenges.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Communicate effectively in group discussions and public speaking settings.
2. Demonstrate improved soft skills including confidence, articulation, and critical thinking.
3. Exhibit professional behavior in mock interviews and workplace simulations.
4. To enhance self-awareness, confidence, and time management skills.
5. To prepare students for interviews, group discussions, and real-world workplace challenges.

List of Experiments:

1. Conduct a group discussion session on a general or technical topic to assess communication clarity, confidence, and listening skills.
2. Organize an extempore round where each student speaks for one minute on a surprise topic to improve spontaneous thinking.
3. Conduct a mock HR interview round simulating campus recruitment conditions with personalized verbal feedback.
4. Arrange a session on body language, posture, and presentation skills using mirror practice and peer review.
5. Provide hands-on guidance to create and refine professional resumes and LinkedIn profiles.
6. Conduct a team-building exercise such as a task-based game to encourage leadership, coordination, and time management.
7. Introduce students to startup culture through a short seminar or documentary on entrepreneurship and innovation.
8. Facilitate an elevator pitch activity where each student presents a business idea in under two minutes.
9. Conduct a second round of mock interviews combining technical and HR questions based on student resumes.
10. Organize a second group discussion round to evaluate growth in communication, confidence, and articulation, followed by a detailed feedback session.
11. Conduct a professional email writing and formal communication workshop using real-life workplace scenarios.
12. Arrange a debate session on a trending topic to develop logical reasoning, articulation, and critical thinking.

SEMESTER III

Scientific Computing with Python			
Course Code:	**CAI201**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**3U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Develop proficiency in Python for scientific and engineering applications.
2. Introduce numerical methods and computational techniques for solving scientific problems.
3. Apply scientific Python libraries for numerical computation, optimization, visualization, and data analysis.
4. Develop efficient, reproducible, and scalable scientific computing solutions.
5. Enable students to solve real-world engineering and data-driven computational problems.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Develop Python programs for scientific and engineering computations.
2. Apply numerical methods using NumPy and SciPy to solve computational problems.
3. Analyze, visualize, and interpret scientific datasets using Python libraries.
4. Develop optimization and machine learning solutions for scientific applications.
5. Design efficient, reproducible, and scalable scientific computing workflows using modern Python tools.

UNIT I: FOUNDATIONS OF SCIENTIFIC COMPUTING AND PYTHON

Introduction to Scientific Computing, Python programming for scientific applications, Python data structures and control flow, Functions, modules, packages, exceptions, Object-Oriented Programming in Python, Jupyter Notebook environment, Scientific Python ecosystem, Introduction to version control using Git, Libraries: Python, Jupyter, IPython.

UNIT II: NUMERICAL COMPUTING WITH NUMPY AND SCIPY

NumPy arrays and vectorized programming, Matrix operations, Broadcasting, Random number generation, Linear algebra using NumPy, Sparse matrices, Numerical differentiation, Numerical integration, Root finding, Interpolation, Solving ordinary differential equations (ODEs), Libraries: NumPy, SciPy.

UNIT III: SCIENTIFIC DATA ANALYSIS AND VISUALIZATION

Scientific data formats, Data manipulation using Pandas, Data cleaning and pre-processing, Statistical analysis, Exploratory Data Analysis, Scientific visualization, 2D and 3D plotting, Heatmaps, Interactive visualization, Visualization best practices, Libraries: Pandas, Matplotlib, Plotly, Seaborn.

UNIT IV: OPTIMIZATION AND SCIENTIFIC MACHINE LEARNING

Mathematical optimization concepts, Linear and nonlinear optimization, Constrained optimization, Curve fitting, Least-squares estimation, Introduction to scientific machine learning, Feature engineering, Libraries: SciPy Optimize, Scikit-learn.

UNIT V: ADVANCED SCIENTIFIC COMPUTING

Graph and network analysis, Parallel computing concepts, Performance optimization, Profiling Python code, Introduction to GPU computing, Unit testing, Package development, Reproducible computational research, Applications of scientific computing involving simulation, optimization, or data analysis, Libraries: NetworkX, multiprocessing, Overview of Numba/CuPy, pytest.

Text Books:

1. Hans Petter Langtangen, *A Primer on Scientific Programming with Python* (5th Edition), Springer International Publishing, 2016.
2. Jake VanderPlas, *Python Data Science Handbook: Essential Tools for Working with Data* (1st Edition), O'Reilly Media, 2016.
3. Wes McKinney, *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and Jupyter* (3rd Edition), O'Reilly Media, 2022.
4. Christian Hill, *Learning Scientific Programming with Python* (2nd Edition), Cambridge University Press, 2020.
5. Eric Matthes, *Python Crash Course: A Hands-On, Project-Based Introduction to Programming* (3rd Edition), No Starch Press, 2023.

Data Structures			
Course Code:	CAI203	**Course Credits:**	3
Course Category:	PCC	**Course (U / P)**	U
Course Year (U / P):	2U	**Course Semester (U / P):**	3U
No. of Lectures + Tutorials (Hrs/Week):	03 + 00	**Mid Sem. Exam Hours:**	1.5
Total No. of Lectures (L + T):	45	**End Sem. Exam Hours:**	3

COURSE OBJECTIVES
1. Understand the fundamental concepts of data structures, abstract data types, recursion, and algorithm analysis using asymptotic notations.
2. Analyze, implement, and compare searching, sorting, and hashing techniques based on their efficiency and suitability for different applications.
3. Design and implement linear data structures such as linked lists, stacks, and queues, and apply them to solve computational problems.
4. Apply tree-based data structures, including binary trees and binary search trees, to perform efficient storage, traversal, searching, and expression evaluation.
5. Represent, traverse, and analyze graph data structures using standard graph traversal algorithms for solving connectivity and ordering problems.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Explain the fundamentals of data structures and algorithm analysis.
2. Implement and analyze searching, sorting, and hashing techniques.
3. Apply linked lists, stacks, and queues to solve computational problems.
4. Construct and manipulate tree data structures for efficient data processing.
5. Represent and traverse graphs using standard graph algorithms.

UNIT I: INTRODUCTION TO DATA STRUCTURES

Introduction to Data Structures, Abstract Data Types, Linear and Non-linear Data Structures, Arrays as ADT, One Dimensional Array, Basic Operations on Arrays, Recursion: Introduction and Process, Examples: Factorial, Fibonacci Sequence, Tower of Hanoi; Asymptotic analysis introduction, Asymptotic notations, Big-Oh Notation, Time Complexity.

UNIT II: SEARCHING AND SORTING

Searching: Linear Search, Binary Search; Sorting: Introduction, Efficiency Considerations, Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, Quick Sort, Counting Sort, Radix Sort, Bucket Sort.

UNIT III: LINKED LIST, STACKS AND QUEUES

Linked List: Introduction, Singly Linked List, Basic Operations on Singly Linked List: Insertion, Deletion, Traversal, Searching, Circular Linked List, Basic Operations on Circular Linked List: Insertion, Deletion, Traversal, Introduction to Doubly Linked List. Stacks: Introduction, Insertion, Deletion, Traversal, Applications: Infix to Postfix Conversion, Post-fix Expression Evaluation, Queues: Introduction, Insertion, Deletion, Traversal, Circular Queue: Insertion, Deletion, Traversal.

UNIT IV: TREES

Tree Data Structures, Height, Complete, Full, Perfect Trees, Binary Trees, Properties of Binary Trees, Representation, Traversal: Pre-Order, In-Order, Post-Order, Binary Search Trees, BST Searching, BST Insertion, BST Deletion, Constructing Tree from given Traversals, Evaluating an expression tree, Heaps, Min-Max Heaps, Heap Tree, Heap Sort, Heap as Priority Queue, Insertion, Deletion.

UNIT V: HASHING AND GRAPHS

Hashing: Hash function, Hash Tables, Hashing Collision Resolution Strategies: Separate Chaining, Linear probing, Quadratic probing, Double hashing; Graphs, Introduction, Different Types of Graphs, Graphs Representations, Incidence Matrix, Adjacency Matrix, Graphs Traversals: Breadth First Search, Depth First Search, Applications: Connected Components, Topological Sort.

Text Books:

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest, Introduction to algorithms. MIT Press; 1990 / McGraw-Hill.
2. E. Horowitz, S. Sahni, Fundamentals of data structures, Computer Science Press, 1982.
3. N. Karumanchi, Data Structures and Algorithms Made Easy, CareerMonk Publications, 2017.
4. S. Lipschutz, Data Structures With C, Tata McGraw-Hill, 2011.
5. Aaron M. Tenenbaum, Yedidyah Langsam, Moshe J. Augenstein, Data Structures Using C, Pearson Education, 2004 / PHI.

Machine Learning			
Course Code:	**CAI205**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**3U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To understand the fundamental concepts, paradigms, and applications of Machine Learning.
2. To apply supervised and unsupervised learning techniques to solve real-world problems.
3. To understand the principles of artificial neural networks and deep learning models.
4. To explore reinforcement learning and recent advancements in machine learning techniques.
5. To analyze and select appropriate machine learning algorithms for data-driven decision-making.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Explain the concepts, types, and applications of machine learning.
2. Implement and evaluate supervised learning algorithms for regression and classification tasks.
3. Develop artificial neural network models using appropriate activation functions and learning algorithms.
4. Apply unsupervised learning techniques for clustering and dimensionality reduction.
5. Analyze reinforcement learning, deep learning models, and emerging machine learning techniques for solving practical problems.

UNIT I: INTRODUCTION TO MACHINE LEARNING

What is Machine Learning, Types of Machine Learning, Supervised Learning, Unsupervised Learning, Reinforcement Learning, Applications of Machine Learning – Stock Price Prediction, Face Recognition, Handwriting Recognition, Image Recognition, Virtual Personal Assistants, Medical Diagnosis, Online Fraud Detection.

UNIT II: SUPERVISED LEARNING (REGRESSION/CLASSIFICATION)

Linear models: Linear Regression, Logistic Regression, Generalized Linear Models; Basic methods: Distance-based methods, Nearest-Neighbours, Decision Trees, Naive Bayes, Support Vector Machines, Nonlinearity and Kernel Methods, Ensemble Methods (Boosting, Bagging, Random Forests).

UNIT III: ARTIFICIAL NEURAL NETWORKS

Biological Neurons and Biological Neural Networks, Artificial Neural Network, Types of Neural Network, Perceptron, History behind Perceptron, Importance of Perceptron, Working of Perceptron, Perceptron Learning, Perceptron Learning Rule, Perceptron Learning of AND & OR gate, XOR gate, Activation functions, Binary Activation function, ReLU, Sigmoid, Hyperbolic, Softmax Activation function, Multilayer Perceptrons, Back propagation Neural Networks, and Feed-Forward Neural Networks, Applications and Future of Neural Networks.

UNIT IV: UNSUPERVISED LEARNING

Clustering: K-means/Kernel K-means, Self-Organizing Maps. Dimensionality Reduction: PCA and kernel PCA, Matrix Factorization and Matrix Completion Generative Models (mixture models and latent factor models)

UNIT V: REINFORCEMENT LEARNING AND DEEP LEARNING

Introduction to Reinforcement Learning, Learning Task, Example of Reinforcement Learning in Practice. Deep Learning: Sparse Modeling and Estimation, Modeling Sequence/Time-Series Data, Deep Learning and Feature Representation Learning. Recent trends in various learning techniques of machine learning and classification methods, Case studies in interdisciplinary domains.

Text/ Reference Books:

1. 1.Tom M. Mitchell, Machine Learning, McGraw-Hill Education (India) Private Limited, 2013.
2. Ethem Alpaydin, Introduction to Machine Learning (Adaptive Computation and Machine Learning), The MIT Press 2004.
3. Stephen Marsland, Machine Learning: An Algorithmic Perspective, CRC Press, 2009.
4. Bishop, C., Pattern Recognition and Machine Learning. Berlin: Springer-Verlag.
5. Hastie, T., Tibshirani, R., Friedman, J., The Elements of Statistical Learning, Springer, 2008.
6. James, G., Witten, D., Hastie, T., Tibshirani, R., An Introduction to Statistical Learning, Springer, 2013.

Computer Networks			
Course Code:	**CA1207**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**3U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Describe how computer networks are organized with the concept of layered approach.
2. Implement a simple LAN with hubs, bridges and switches.
3. Analyze the contents in a given Data Link layer packet, based on the layer concept.
4. Describe what classless addressing scheme is.
5. Describe how routing protocols work.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Analyze the requirements for a given organizational structure and select the most appropriate networking architecture and technologies.
2. Have a basic knowledge of the use of cryptography and network security.
3. Specify and identify deficiencies in existing protocols, and then go onto formulate new and better protocols.
4. Analyze specify and design the topological and routing strategies for an IP based networking infrastructure.
5. Have a working knowledge of datagram and internet socket programming.

UNIT I: INTRODUCTION AND PHYSICAL LAYER

Key concepts of computer network, transmission media, network devices, network topology, topology design issues, types of Network: LAN, MAN, WAN, PAN, ISDN systems and ATM network, OSI-reference model, open system standards, characteristics of network, TCP/IP model, protocols and standards, encoding technique.

UNIT II: SWITCHING AND DATA LINK LAYER

Multiplexing Techniques: TDMA, FDMA, WDMA, CDMA, Data link layer: LLC and MAC sublayers, protocols and design issues, issues IEEE 802 LAN Standards, framing, CRC, error control, flow control, HDLC, ALOHA and performance issues. Frames relay networks and performance parameters.

UNIT III: NETWORK LAYER

Network layer design issues, overview of IPv4 and IPv6, addressing: class full and classless, static and dynamic, subnet and super net, auto configuration through DHCP, routing protocols: RIP, DVR, LSR, OSFP, BGP, congestion control algorithm, subnet concept, virtual LAN, ICMP, multicasting, mobile IP.

UNIT IV: TRANSPORT LAYER

Port addressing schemes, connectionless and connection oriented services: TCP and UDP, wireless TCP, Congestion control, queue management, NAT, PAT, socket format at transport level, socket interface and programming.

UNIT V: APPLICATION LAYER

Client server architecture, domain name services, application services: HTTP, TELNET, RLOGIN, FTP, CBR, NFS, SMTP, POP, IMAP, MIME, voice and video over IP, social issues- privacy, freedom of speech, copy right.

Text Books:

1. S. Tanenbaum, Computer Networks, 4th edition, Prentice Hall, 2008.
2. Forouzan, B.A., Data Communication and Networking, Tata McGraw-Hill.
3. W. Stallings, Data and Computer Communications, 8th edition, Prentice Hall, 2007
4. Douglus E. ComerTCP/IP Principles, Protocols and Architecture, Pearson Education.

36.1.1
RLOGIN, freedom

Operating System			
Course Code:	**CAI209**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**3U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Understanding how Operating System is Important for Computer System.
2. To make aware of different types of Operating System and their services.
3. To learn different process scheduling algorithms and synchronization techniques to achieve better performance of a computer system.
4. To know virtual memory concepts and secondary memory management.
5. Understanding of Security & protection in Operating System.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Understands the different services provided by Operating System at different level
2. They learn real life applications of Operating System in every field.
3. Understands the use of different process scheduling algorithm and synchronization techniques to avoid deadlock.
4. They will learn different memory management techniques like paging, segmentation and demand paging etc.
5. Perform implementation of protection mechanisms in operating system

UNIT I: INTRODUCTION TO OPERATING SYSTEM

Importance of operating systems, basic concepts and terminology about operating system, memory management, processor management, device management, information management functions.

UNIT II: PROCESS MANAGEMENT

Elementary concept of process, job scheduler, process scheduling, operation on process, threads, overview, scheduling criteria, scheduling algorithms, algorithm, and deadlocks: system model, deadlock characterization, deadlocks prevention, deadlocks avoidance, deadlocks detection, recovery from deadlock.

UNIT III: MEMORY MANAGEMENT

Memory management, logical versus physical address space, swapping, contiguous allocation, paging, segmentation, demand paging, page replacement, page replacement algorithms, allocation of frames, thrashing, demand segmentation.

UNIT IV: STORAGE MANAGEMENT

File concept, directory structure, protection, file-system structure, allocation method, free-space management, directory implementation.

UNIT V: I/O SYSTEMS

I/O hardware, Application of I/O interface, Overview of Kernel I/O subsystem, three types of I/O systems, memory based I/O, I/O based I/O, peripheral based I/O.

Text Books:

1. Galvin, Wiley, Operating Systems Concepts, 8th edition, 2009.
2. James L Peterson, Operating Systems Concept, John Wiley & Sons Inc, the 6Rev edition, 2007.

Reference Books:

1. Deitel H. M., An Introduction to Operating Systems, Addison-Wesley, 1990.
2. Stallings William, Operating Systems, PHI, New Delhi, 1997.
3. S. Tanenbaum Modern Operating Systems, Pearson Education, 3rd edition, 2007.
4. Nutt, Operating System, Pearson Education, 2009.
5. S. Tanenbaum, Distributed Operating Systems, Prentice Hall, 2nd edition, 2007.

Scientific Computing with Python Lab			
Course Code:	**CAI281**	**Course Credits:**	**2**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**3U**
No. of Labs (Hrs/Week):	**1 (4 HRS)**	**Mid Sem. Exam Hours:**	**–**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Develop proficiency in using Python and scientific computing libraries for solving computational problems.
2. Apply numerical methods and scientific programming techniques to analyze and solve engineering and scientific problems.
3. Perform data processing, visualization, and optimization using modern Python tools and libraries.
4. Design, implement, and evaluate scientific computing solutions through practical experimentation and real-world applications.
5. Develop analytical, programming, and problem-solving skills for scientific and engineering computations.
COURSE OUTCOMES
At the end of the course the students should be able to:
1. Develop Python programs using scientific computing libraries such as NumPy, SciPy, Pandas, and Matplotlib for computational tasks.
2. Apply numerical methods and matrix computations to solve scientific and engineering problems.
3. Perform data analysis, visualization, and optimization using appropriate Python libraries.
4. Design, implement, and evaluate scientific computing solutions for real-world engineering and data-driven applications.
5. Analyze computational performance and develop efficient, reliable, and reproducible scientific computing programs.

LIST OF EXPERIMENTS:

1. Write a Python program to perform basic scientific computations using variables, operators, functions, and control statements in Jupyter Notebook.
2. Write a Python program to create and manipulate NumPy arrays, perform vectorized operations, and compare their performance with Python lists.
3. Implement matrix operations (addition, multiplication, transpose, inverse, and determinant) using the NumPy library.
4. Write a Python program to solve systems of linear equations and compute eigenvalues and eigenvectors using NumPy library.
5. Implement numerical methods for root finding and numerical integration using the SciPy library.
6. Write a Python program to perform interpolation and solve ordinary differential equations (ODEs) using SciPy library.
7. Write a Python program to load, clean, manipulate, and analyze datasets using the Pandas library.
8. Develop Python programs to create line plots, bar charts, scatter plots, histograms, and heatmaps using Matplotlib and Seaborn library.
9. Implement curve fitting, least-squares estimation, and optimization problems using the SciPy Optimize module.
10. Write a Python program to develop and evaluate regression and classification models using the Scikit-learn library.

11. Implement a Python program to profile computational performance, compare execution time of algorithms, and optimize code using vectorization techniques.

Data Structures Lab			
Course Code:	**CAI283**	**Course Credits:**	**2**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**3U**
No. of Labs (Hrs/Week):	**1 (4 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Understand the implementation of fundamental data structures.
2. Develop programs using linear and non-linear data structures.
3. Apply appropriate data structures to solve computational problems.
4. Analyze the performance of data structures and algorithms.
5. Enhance programming, debugging, and problem-solving skills.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Implement array operations, recursion, and basic algorithm analysis.
2. Develop and evaluate searching, sorting, and hashing algorithms.
3. Implement linked lists, stacks, and queues for practical applications.
4. Construct and manipulate tree and graph data structures using standard algorithms.
5. Design efficient solutions by selecting appropriate data structures for real-world problems.

List of Experiments:

1. Implement one-dimensional and two-dimensional array operations (insertion, deletion, searching, traversal, merging).
2. Implement recursive algorithms for Factorial and Fibonacci series. Analyze their time complexity.
3. Implement Linear Search and Binary Search. Compare their execution time and efficiency for different input sizes.
4. Implement Bubble Sort, Selection Sort, and Insertion Sort. Compare their time complexities experimentally.
5. Implement Merge Sort and Quick Sort. Compare recursive divide-and-conquer sorting algorithms.
6. Implement Heap Sort and Counting Sort algorithms.
7. Implement Hash Tables using Separate Chaining and Open Addressing.
8. Implement Singly and Circular Linked Lists with insertion, deletion, searching, and traversal operations.
9. Implement Stack using Arrays and Linked Lists.
10. Implement Circular Queue and Priority Queue using Arrays and Linked Lists.
11. Implement Binary Tree and Binary Search Tree operations including insertion, deletion, searching, and traversals (Preorder, Inorder, Postorder).
12. Implement Graph representations using Adjacency Matrix and Adjacency List. Perform BFS, DFS.

Computer Networks Lab			
Course Code:	**CAI285**	**Course Credits:**	2
Course Category:	**PCC**	**Course (U / P)**	U
Course Year (U / P):	**2U**	**Course Semester (U / P):**	3U
No. of Lectures + Tutorials (Hrs/Week):	**1 (4 HRS)**	**Mid Sem. Exam Hours:**	–
Total No. of Lectures (L + T):	**10**	**End Sem. Exam Hours:**	3

COURSE OBJECTIVES
1. Practical knowledge of working principles of various communication protocols.
2. Analyze structure and formats of TCP/IP layer protocols.
3. Understanding of networking fundamentals.
4. Understanding of learning the process of Internet of Things applications planning.
5. Understanding of configuration of various end devices, server, routers and switches.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Understand the practical approach to network communication protocols.
2. Understand network layers, structure/format and role of each network layer.
3. Able to design and implement various network application such as data transmission between client and server, file transfer, real-time multimedia transmission.
4. Understand the various Routing Protocols/Algorithms and Internetworking.
5. Learn to configure server.

List of Experiments:

1. Introduction to transmission media (CAT5, OFC, COAXIAL CABLE Wireless).
2. Introduction to network interfaces (Wired and Wireless).
3. Configure and installing an Ethernet (10/100).
4. Performance evaluation of Ethernet (10/100).
5. Topology design (Ring, Bus).
6. Generation of data packet and measurement (CBR, VBR, Poison).
7. Implement the following:
    a. Router Configuration
    b. Server Configuration
    c. Server Configuration
8. Congestion control of network and QoS of network
9. Protocols and the configuration
10. Security (WEP, WPA) and Qualnet.

Machine Learning Lab			
Course Code:	**CAI287**	**Course Credits:**	2
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**3U**
No. of Lectures + Tutorials (Hrs/Week):	**1 (4 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Lectures (L + T):	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Develop proficiency in Python programming and machine learning libraries for data analysis.
2. Implement and evaluate supervised and unsupervised machine learning algorithms.
3. Apply appropriate data pre-processing, visualization, and model evaluation techniques.
4. Perform model optimization through hyperparameter tuning and comparative analysis.
5. Solve real-world machine learning problems using standard tools and datasets.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Use Python libraries to perform data pre-processing, statistical analysis, and visualization.
2. Implement and evaluate regression and classification models using appropriate performance metrics.
3. Apply hyperparameter tuning techniques to improve the performance of machine learning models.
4. Implement clustering techniques for document and data analysis using suitable evaluation measures.
5. Develop complete machine learning solutions for real-world datasets using standard Python-based frameworks.

List of Experiments:

1. Implement a Python program to compute measures of central tendency (Mean, Median, and Mode) and measures of dispersion (Variance and Standard Deviation).
2. Implement basic functions of Python libraries such as statistics, math, NumPy, and SciPy for scientific computing.
3. Implement basic functions of Python libraries for machine learning applications, including Pandas and Matplotlib, for data manipulation and visualization.
4. Implement Linear Regression using five datasets of varying sizes. Compare the model performance by plotting evaluation metrics such as MAE, MSE, RMSE, and R^2 Score.
5. Implement the K-Nearest Neighbors (KNN) classifier using scikit-learn. Evaluate the model on five datasets of varying sizes by comparing Accuracy, Precision, Recall, F1-Score, and ROC-AUC. Using one large dataset, vary the value of K (1, 3, 5, 7, 9, 11, 21, and 31) and compare the performance using the same evaluation metrics.
6. Implement the Decision Tree classifier using scikit-learn and perform hyperparameter tuning. Evaluate the model on five datasets of varying sizes and compare Accuracy, Precision, Recall, F1-Score, and ROC-AUC.
7. Implement the Naïve Bayes classifier using scikit-learn. Evaluate the model on five datasets of varying sizes and compare Accuracy, Precision, Recall, F1-Score, and ROC-AUC.

8. Implement the Support Vector Machine (SVM) classifier using scikit-learn. Evaluate the model on five datasets of varying sizes and compare Accuracy, Precision, Recall, F1-Score, and ROC-AUC.
9. Implement the Random Forest classifier using scikit-learn. Evaluate the model on five datasets of varying sizes and compare Accuracy, Precision, Recall, F1-Score, and ROC-AUC.
10. Implement ensemble learning algorithms (AdaBoost and XGBoost). Evaluate the models on five datasets of varying sizes and compare Accuracy, Precision, Recall, F1-Score, and ROC-AUC.
11. Implement an Artificial Neural Network (ANN) using scikit-learn or TensorFlow/Keras and perform hyperparameter tuning. Evaluate the model on five datasets of varying sizes and compare Accuracy, Precision, Recall, F1-Score, and ROC-AUC.
12. Implement K-Means clustering for document clustering using suitable text datasets. Evaluate the clustering performance using Silhouette Score and visualize the resulting clusters.

SEMESTER IV

Design and Analysis of Algorithms			
Course Code:	**CAI202**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**4U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To analyze the asymptotic time and space performance of algorithms using appropriate complexity measures.
2. To develop rigorous correctness proofs for algorithms using formal techniques such as invariants and induction.
3. To acquire and demonstrate familiarity with fundamental and advanced algorithms and data structures.
4. To apply key algorithmic design paradigms and analytical methods to solve computational problems effectively.
5. To synthesize efficient and scalable algorithmic solutions for common engineering and real-world problem scenarios.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Analyze algorithms and data structures using asymptotic notations, recurrence relations, and amortized analysis to evaluate time and space complexity.
2. Design efficient algorithms using appropriate paradigms such as greedy, divide-and-conquer, dynamic programming, backtracking, and branch and bound.
3. Implement and apply advanced data structures and graph algorithms to solve complex computational problems.
4. Design and evaluate advanced algorithmic solutions using randomized, approximation, and online paradigms with respect to correctness, efficiency, and practical applicability.
5. Classify computational problems into complexity classes (P, NP, NP-Complete, NP-Hard) and justify algorithmic choices for intractable problems.

UNIT I: INTRODUCTION

Introduction to Algorithm, Time Complexity and Space Complexity, Order of Growth, Asymptotic Notations: Big Oh, Theta, Omega, Amortized analysis, Analysing control statement, Loop Invariant, Recurrence Relations: Introduction, Substitution Method, Recursion Tree Method, Master's Theorem. Searching and Sorting Algorithms analysis.

UNIT II: ALGORITHM DESIGN TECHNIQUES

Divide and Conquer: Binary Search, Merge Sort, Strassen's Matrix Multiplication; Greedy Algorithm: General Characteristics, Knapsack Problem, Huffman code; Dynamic Programming: Introduction, Principle of Optimality, 0-1 Knapsack, Longest Common Subsequence, Matrix Chain Multiplication.

UNIT III: STATE SPACE SEARCH ALGORITHMS AND PATTERN MATCHING

Backtracking: State-Space Search Tree, N-Queen's problem, Graph Colouring; Branch and Bound: Travelling Salesman Problem; Pattern Matching: The Naïve string-matching algorithm, Rabin-Karp algorithm, Knuth-Morris-Pratt algorithm.

UNIT IV: ADVANCED GRAPH ALGORITHMS

Minimum Spanning Trees: Prim's algorithm, Kruskal's algorithm; Shortest Path: Bellman-Ford algorithm, Dijkstra's algorithm, Floyd Warshall algorithm; Network flow: Flow networks, Max-Flow, Min-Cut, Ford-Fulkerson method.

UNIT V: ADVANCED ALGORITHMS

P and NP, NP Complete and NP-Hard; Advanced Data Structures and their applications: Height Balanced Trees: AVL Trees, Balance Factor, Rotations, Insertion, Deletion, B-Trees, Insertion, Deletion, Advanced Algorithms and their applications.

Text Books:

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, Introduction to Algorithms, 4th Edition, MIT Press.
2. Anany Levitin, "Introduction to the Design and Analysis of Algorithm", Pearson Education Asia, 2003.
3. Sara Baase and Allen Van Gelder,"Computer Algorithms-Introduction to the Design and Analysis ", Pearson Education Asia, 2003.
4. A. V. Aho, J.E. Hopcroft and J.D. Ullman, "The Design and Analysis of Computer Algorithms", Pearson Education Asia, 2003.

Database Management System			
Course Code:	**CAI204**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**4U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Describe the fundamental elements of relational database management systems
2. Explain the basic concepts of relational data model, entity-relationship model, relational database design, relational algebra and SQL.
3. Design ER-models to represent simple database application scenarios
4. Convert the ER-model to relational tables, populate relational database and formulate SQL queries on data.
5. Improve the database design by normalization.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Students will have an understanding of database concepts and thorough knowledge of database software's.
2. Students will be able to model an application's data requirements using ER diagrams
3. Students will be able to write SQL commands to create tables and query data in a relational DBMS
4. Students will be able to execute various advanced SQL queries related to transactions, concurrency
5. Students will be able to explain the principle of transaction management design.

UNIT I: DATABASE SYSTEM

Data base system vs. file system, view of data, data abstraction, instances and schemas, data models, ER model, relational model, DDL, DML, database access for applications programs, data base users and administrator, transaction management, data base system structure, storage manager, query processor, history of data base systems, data base design and ER diagrams, beyond ER design entities, attributes and entity sets, relationships and relationship sets, additional features of ER model, concept design with the ER model, and conceptual design for large enterprises.

UNIT II: RELATIONAL MODEL

Introduction to the relational model, integrity constraint over relations, enforcing integrity constraints, querying relational data, and logical data base design, destroying /altering tables and views. relational algebra and calculus: relational algebra, selection and projection set operations, renaming, joins, division.

UNIT III: BASIC SQL QUERY

Examples of basic SQL queries, nested queries, correlated nested queries set, comparison operators, aggregative operators, NULL values, comparison using null values, logical connectivity's, AND, OR and NOT, impact on SQL constructs, outer joins, disallowing NULL values, complex integrity constraints in SQL triggers and active data bases.

UNIT IV: SCHEMA REFINEMENT

Problems caused by redundancy, decompositions, problem related to decomposition, reasoning about FDS, FIRST, SECOND, THIRD normal form, BCNF, forth normal form, lossless join decomposition, dependency preserving decomposition, schema refinement in data base design, multi valued dependencies.

UNIT V: OVERVIEW OF TRANSACTION MANAGEMENT

ACID properties, transactions and schedules, concurrent execution of transaction, lock based concurrency control, performance locking, and transaction support in SQL, crash recovery, concurrency control, Serializability and recoverability, lock management, lock conversions, dealing with deadlocks, specialized locking techniques, concurrency without locking, crash recovery: ARIES, log, other recovery related structures, the write, ahead log protocol, check pointing, recovering from a system crash, media recovery, other approaches and interaction with concurrency control.

Text Books:

1. Elmasri Navrate, Data Base Management System, Pearson Education, 2008.
2. Raghurama Krishnan, Johannes Gehrke, Data Base Management Systems, TMH, 3rd edition, 2008.
3. C. J. Date, Introduction to Database Systems, Pearson Education, 2009.
4. Silberschatz, Korth, Database System Concepts, McGraw hill, 5th edition, 2005.
5. Rob, Coronel & Thomson, Database Systems Design: Implementation and Management.

Deep Learning			
Course Code:	**CAI206**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P):**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**4U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Understand the basics of artificial neural networks and deep neural networks.
2. Implement deep neural networks with optimization.
3. Build basic deep learning models and analyze the results.
4. Design and implement convolution neural networks.
5. Design and implement sequential and advanced deep architectures.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Understand and implement artificial neural networks and deep neural networks.
2. Analyze the given dataset for designing a neural network-based solution.
3. Build basic deep learning models and interpret the results.
4. Design and implement convolution neural networks.
5. Design and implement sequential and advanced deep architectures to solve real world problems.

UNIT I: INTRODUCTION

Basics of artificial neural networks (ANN): Artificial neurons, Computational models of neurons, Architecture of neural networks, Feedforward neural networks, Backpropagation learning, Cost Function, Regularization, Autoencoders. Logistic Regression vs neural networks, Activation functions, L-layer NN. Optimization: Types of errors, bias-variance trade-off, overfitting-underfitting, brief review of concepts from optimization, variants of gradient descent, momentum-based methods.

UNIT II: DEEP NEURAL NETWORKS

Deep neural networks (DNNs): Difficulty of training DNNs, Bias-Variance Trade-off, Optimization: Optimization algorithms for neural networks (AdaGrad, RMSProp, Adam), Hyperparameters tuning (logarithmic scale), Regularization methods (dropout, drop connect, batch normalization), Multi-class classification.

UNIT III: CONVOLUTION NEURAL NETWORKS

Convolution neural networks (CNNs): Introduction to CNNs – convolution, pooling, Kernel filter, Principles behind CNNs, Deep CNNs, Different deep CNN architectures – LeNet, AlexNet, VGG, ResNet, Training CNNs, Understanding and visualizing CNNs.

UNIT IV: SEQUENTIAL MODELS

Recurrent neural networks (RNNs): Sequence modelling using RNNs, Back propagation through time, Bidirectional RNNs, Gated RNN Architecture, Long Short-Term Memory (LSTM), Bidirectional LSTMs.

UNIT V: ADVANCED DEEP ARCHITECTURES

Generative models: Restrictive Boltzmann Machines (RBMs), Stacking RBMs, Belief nets, Generative Adversarial Networks (GANs). Applications: Applications in vision, speech and natural language processing.

Text/Reference Books:

1. Goodfellow, I., Bengio, Y., and Courville, A., Deep Learning, MIT Press, 2016.
2. Bishop, C. ,M., Pattern Recognition and Machine Learning, Springer, 2006.
3. Yegnanarayana, B., Artificial Neural Networks PHI Learning Pvt. Ltd, 2009.
4. Golub, G.,H., and Van Loan,C.,F., Matrix Computations, JHU Press, 2013.

Course Code:	**Theory of Automata**		
Course Category:	**CAI208**	**Course Credits:**	**3**
Course Year (U / P):	**PCC**	**Course (U / P)**	**U**
No. of Lectures + Tutorials (Hrs/Week):	**2U**	**Course Semester (U / P):**	**4U**
	03 + 00	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To introduce the fundamental concepts of automata, formal languages, and computability.
2. To develop understanding of finite automata and regular expressions for pattern recognition and lexical analysis.
3. To study context-free grammars and pushdown automata for syntax analysis and language processing.
4. To understand the concepts of Turing machines, decidability, and computability.
5. To provide theoretical foundations required for compiler design, artificial intelligence, and advanced computing systems.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Design deterministic and non-deterministic finite automata for regular languages.
2. Apply regular expressions and finite automata for pattern matching and lexical analysis.
3. Construct context-free grammars and pushdown automata for context-free languages.
4. Analyze formal languages using pumping lemmas and closure properties.
5. Design Turing machines and explain the concepts of decidability and computability.
6. Relate automata theory concepts with applications in compiler design, natural language processing, and artificial intelligence.

UNIT I: INTRODUCTION

Alphabets, Strings and Languages; Automata and Grammars, Deterministic finite Automata (DFA)-Formal Definition, Simplified notation: State transition graph, Transition table, Language of DFA, Nondeterministic finite Automata (NFA), NFA with epsilon transition, Language of NFA, Equivalence of NFA and DFA, Minimization of Finite Automata, Distinguishing one string from other, Myhill-Nerode Theorem.

UNIT II: REGULAR EXPRESSION (RE)

Definition, Operators of regular expression and their precedence, Algebraic laws for Regular expressions, Kleen's Theorem, Regular expression to FA, DFA to Regular expression, Arden Theorem, Non-Regular Languages, Pumping Lemma for regular Languages. Application of Pumping Lemma, Closure properties of Regular Languages, Decision properties of Regular Languages, FA with output: Moore and Mealy machine, Equivalence of Moore and Mealy Machine, Applications and Limitation of FA.

UNIT III: CONTEXT FREE GRAMMAR (CFG) AND CONTEXT FREE LANGUAGES (CFL)

Context free grammar (CFG) and Context Free Languages (CFL): Definition, Examples, Derivation, Derivation trees, Ambiguity in Grammar, Inherent ambiguity, Ambiguous to Unambiguous CFG, Useless symbols, Simplification of CFGs, Normal forms for CFGs: CNF and GNF, Closure proper ties of CFLs, Decision Properties of CFLs: Emptiness, Finiteness and Membership, Pumping lemma for CFLs.

UNIT IV: PUSH DOWN AUTOMATA (PDA)

Description and definition, Instantaneous Description, Language of PDA, Acceptance by Final state, Acceptance by empty stack, Deterministic PDA, Equivalence of PDA and CFG, CFG to PDA and PDA to CFG, two stack PDA.

UNIT V: TURING MACHINES (TM)

Basic model, definition and representation, Instantaneous Description, Language acceptance by TM, Variants of Turing Machine, TM as Computer of Integer functions, Universal TM, Church's Thesis, Recursive and recursively enumerable languages, Halting problem.

Text/Reference Books:

1. Hopcroft, Ullman, "Introduction to Automata Theory, Languages and Computation", Pearson Education.
2. Peter Linz, "An Introduction to Formal Language and Automata", Narosa Publishing house.
3. YN Singh "Mathematical Foundation of Computer Science", New Age International.
4. Malviya, AK "Theory of Computation and Application", BPaperback Publications
5. Papadimitrou, C. and Lewis, CL, "Elements of the Theory of Computation", Pearson Publication.
6. K. Krithivasan and R. Rama; Introduction to Formal Languages, Automata Theory and Computation; Pearson Education.
7. KLP Mishra and N. Chandrasekaran, "Theory of Computer Science: Automata, Languages and Computation", PHI Learning Private Limited, Delhi India.
8. Harry R. Lewis and Christos H. Papadimitriou, Elements of the theory of Computation, Second Edition, Prentice-Hall of India Pvt. Ltd.
9. Micheal Sipser, "Introduction of the Theory and Computation", Thomson Learning.
10. Katuri Viswanath, "Introduction to Mathematical Computer Science, An" Universities Press.

Digital Systems and Computer Architecture			
Course Code:	**CAI210**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**4U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To understand the fundamentals of digital logic, Boolean algebra, number systems, data representation, and arithmetic operations used in digital computers.
2. To analyze and design combinational and sequential logic circuits using Boolean minimization techniques and standard digital building blocks.
3. To understand the organization and operation of computer processors, instruction set architectures, and control unit design.
4. To explain memory hierarchy, cache organization, pipelining, and input/output organization in modern computer systems.
5. To develop a strong foundation in digital system design and computer organization for advanced studies in computer architecture, embedded systems, operating systems, and related domains.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Explain digital logic fundamentals, Boolean algebra, and data representation techniques.
2. Design and analyze combinational and sequential digital circuits.
3. Describe processor organization, instruction execution, and instruction set architectures.
4. Analyze memory hierarchy, control unit organization, and pipelined processing.
5. Explain input/output organization and the fundamentals of parallel and multicore architectures.

UNIT I: Digital Logic Fundamentals and Data Representation

Introduction to Digital Systems, Analog and Digital Systems, Number Systems (Binary, Octal, Decimal, Hexadecimal), Base Conversions, Binary Codes (BCD, Gray Code, ASCII), Signed Number Representation (1's and 2's Complement), Fixed-Point and Floating-Point Representation, Binary Arithmetic, Boolean Algebra, Laws of Boolean Algebra, DeMorgan's Theorems, Logic Gates, Universal Gates, NAND/NOR Realization of Boolean Functions, Boolean Functions, Canonical Forms (SOP and POS), Truth Tables, Register Transfer Language (RTL), Register Transfer and Micro-operations.

UNIT II: Boolean Function Minimization and Combinational Logic Design

Boolean Function Minimization using Karnaugh Maps (3, 4 and 5 Variables), Don't Care Conditions, Quine-McCluskey Minimization Technique, Design of Combinational Circuits, Encoders, Decoders, Multiplexers, Demultiplexers, Multiplexers as Boolean Function Generators, Half and Full Adders, Ripple Carry Adder, Carry Look-Ahead Adder, Binary Subtractors, Magnitude Comparators, Arithmetic Logic Unit (ALU), Basic Arithmetic Algorithms: Binary Multiplication and Division Algorithms.

UNIT III: Sequential Logic and Processor Organization

Latches (SR and D), Flip-Flops (SR, JK, D and T), Characteristic Tables, Registers, Shift Registers, Ripple and Synchronous Counters, Finite State Machines (Mealy and Moore Models), State Diagrams, Functional Units of a Computer, Basic Computer Organization, Bus Structures and Interconnections,

CPU and Register Organization, Instruction Cycle, Instruction Formats, Addressing Modes, Instruction Set Architecture (ISA), Instruction Cycle and Instruction Execution, RISC and CISC Architectures.

UNIT IV: Control Unit and Memory Organization

Control Unit Design, Hardwired and Microprogrammed Control Units, Memory Hierarchy, Semiconductor Memories (SRAM and DRAM), Memory Chip Organization, Main Memory, Cache Memory, Cache Mapping Techniques and Cache Performance, Associative Memory, Virtual Memory, Instruction Pipelining, Pipeline Hazards.

UNIT V: Input/Output Organization and Computer Architecture

Input/Output Organization, Programmed I/O, Interrupt-Driven I/O, Direct Memory Access (DMA), I/O Interfaces, Serial and Parallel Communication, PCI Express (PCIe) and USB Interfaces, Input and Output Devices, Secondary Storage Devices, Flynn's Classification, Introduction to Parallel Processing and Multicore Architectures.

Text/ Reference Books:

1. M. Morris Mano and Michael D. Ciletti, Digital Design, 6th Edition, Pearson, 2018.
2. David A. Patterson and John L. Hennessy, Computer Organization and Design: The Hardware/Software Interface, 6th Edition, Morgan Kaufmann, 2021.
3. William Stallings, Computer Organization and Architecture: Designing for Performance, 11th Edition, Pearson, 2022.
4. Carl Hamacher, Zvonko G. Vranesic, Safwat G. Zaky, and Naraig Manjikian, Computer Organization and Embedded Systems, 6th Edition, McGraw-Hill, 2012.
5. Charles H. Roth Jr. and Larry L. Kinney, Fundamentals of Logic Design, 8th Edition, Cengage Learning, 2021.
6. Andrew S. Tanenbaum and Todd Austin, Structured Computer Organization, 6th Edition, Pearson, 2013.
7. John F. Wakerly, Digital Design: Principles and Practices, 5th Edition, Pearson, 2018.
8. John P. Hayes, Computer Architecture and Organization, 3rd Edition, McGraw-Hill.
9. Randal E. Bryant and David R. O'Hallaron, Computer Systems: A Programmer's Perspective, 3rd Edition, Pearson, 2015.

Design and Analysis of Algorithms Lab			
Course Code:	**CAI282**	**Course Credits:**	**2**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**4U**
No. of Lectures + Tutorials (Hrs/Week):	**1 (4 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Lectures (L + T):	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

1. Implement and experimentally analyze algorithms using appropriate measures of time and space complexity.
2. Develop solutions using divide-and-conquer, greedy, dynamic programming, and backtracking paradigms.
3. Implement tree, heap, and graph structures required for efficient algorithmic problem solving.
4. Evaluate alternative algorithms on varied input characteristics and justify an appropriate choice.
5. Develop correct, readable, and testable programs using C, Java, or Python.

COURSE OUTCOMES

At the end of the course the students should be able to:

1. Implement algorithms and empirically analyze their best-, average-, and worst-case behaviour.
2. Design and implement divide-and-conquer and greedy algorithms for sorting, selection, spanning tree, shortest path, scheduling, coding, and knapsack problems.
3. Formulate and implement dynamic programming and backtracking solutions and reconstruct the corresponding optimal or feasible solution.
4. Implement and apply heaps, search trees, balanced trees, and graph traversal techniques.
5. Implement standard string-matching algorithms and compare competing algorithms using reproducible experiments.

PRACTICALS

Note: Students may use C, Java, or Python. Each experiment must include algorithm/pseudocode, source code, test cases, output, and a brief complexity/performance analysis.

Lab 1: Experimental Analysis of Algorithms

1. Implement iterative and recursive versions of factorial and Fibonacci computation; count basic operations and compare their running times for increasing input sizes.
2. Implement linear search and binary search; test best-, average-, and worst-case inputs and plot input size versus execution time.

Lab 2: Elementary Sorting and Heap-Based Processing

1. Implement bubble sort, insertion sort and selection sort; record comparisons and data movements for sorted, reverse-sorted, and random inputs.

Lab 3: Divide-and-Conquer Fundamentals

1. Implement binary search using both iterative and recursive approaches; compare their execution time and auxiliary space.
2. Implement merge sort; test it on arrays containing duplicate, negative, sorted, reverse-sorted, and random values, and analyze its time and auxiliary space.

Lab 4: Advanced Divide-and-Conquer Algorithms

1. Implement quick sort using first-element, last-element, randomized, and median-of-three pivot selection; compare performance on different input distributions.
2. Implement Strassen's matrix multiplication and compare its correctness and execution time with conventional matrix multiplication for compatible matrix sizes.

Lab 5: Greedy Algorithms for Spanning Trees and Scheduling

1. Implement Prim's algorithm to obtain a minimum spanning tree of a weighted, connected, undirected graph and report its total cost.
2. Implement Kruskal's algorithm using disjoint-set operations; compare its output and running time with Prim's algorithm on sparse and dense graphs.

Lab 6: Greedy Shortest Paths, Coding, and Knapsack

1. Implement Dijkstra's algorithm for a graph with non-negative edge weights; display the shortest distance and path from a selected source to every vertex.
2. Implement the fractional knapsack problem; display the selected fraction of each item and compare the result with a simple value-based selection strategy.

Lab 7: Dynamic Programming - Paths and Knapsack

1. Implement the 0/1 knapsack problem using memoization and tabulation; reconstruct the selected items and compare the two implementations.
2. Implement the Floyd-Warshall algorithm for all-pairs shortest paths; construct the distance matrix and identify whether a negative cycle exists.

Lab 8: Dynamic Programming and Backtracking

1. Implement the traveling salesperson problem using dynamic programming for small instances; report the minimum tour cost and the tour.
2. Implement the N-Queens problem using backtracking; generate all valid solutions for a given board size and report the number of states explored.

Lab 9: String Matching and Comparative Study

1. Implement the Naive and Rabin-Karp string-matching algorithms; report all match positions, spurious hits, and character comparisons.
2. Implement the Knuth-Morris-Pratt algorithm; construct and display the prefix table and verify matches for patterns with repeated prefixes.

Lab 10: Advanced Trees and Graph Traversal

1. Implement a binary search tree with insertion, deletion, search, and inorder, preorder, and postorder traversals; verify the BST property after updates.
2. Implement an AVL tree with all required rotations; compare its height and search cost with an ordinary BST for the same insertion sequence.

Database Management System Lab			
Course Code:	**CAI284**	**Course Credits:**	**2**
Course Category:	**PCC**	**Course (U / P)**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**4U**
No. of Labs (Hrs/Week):	**1 (4 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To explain basic database concepts, applications, data models, schemas and instances.
2. To demonstrate the use of constraints and relational algebra operations.
3. To emphasize the importance of normalization in databases.
4. To facilitate students in Database design
5. To familiarize issues of concurrency control and transaction management.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Students get practical knowledge on designing and creating relational database systems.
2. Understand various advanced queries execution such as relational constraints, joins, set operations, aggregate functions, trigger, views and embedded SQL
3. Design a commercial relational database system (Oracle, MySQL) by writing SQL using the system
4. Use the basics of SQL and construct queries using SQL in database creation and interaction.
5. Analyze and Select storage and recovery techniques of database system.

List of Experiments:

1. Introduction to MySQL, an exercise of data types in MySQL & Data Definition Language Commands.
2. Exercise on Data Manipulation Language and Transaction Control Commands.
3. Exercise of Type of Data Control Commands.
4. Exercise of joins (Single Table - Using Normalization)
5. Exercise of joins (Multiple Table - Using Normalization)
6. Exercise on GROUP BY/ORDER BY Clause and Date Arithmetic.
7. Exercise on different Functions (Aggregate, Math and String).
8. Exercise on different types of sub queries.
9. Exercise on Procedures.
10. Exercise on Views.

Deep Learning Lab			
Course Code:	**CAI286**	**Course Credits:**	**2**
Course Category:	**PCC**	**Course (U / P):**	**U**
Course Year (U / P):	**2U**	**Course Semester (U / P):**	**4U**
No. of Lectures + Tutorials (Hrs/Week):	**1 (4 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Lectures (L + T):	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Implement the various deep learning algorithms in Python.
2. Learn to work with different deep learning frameworks like Keras, Tensor flow, PyTorch,.
3. Understand complexity of Deep Learning algorithms and be capable of performing distributed computations.
4. Be capable of confidently applying common Deep Learning algorithms in practice and implementing their own Deep Learning algorithms.
5. Be capable of performing experiments in Deep Learning using real-world data.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Implement deep learning algorithms and understand neural networks.
2. Understand concepts of TensorFlow, its main functions, operations and the execution pipeline.
3. Learn to implement and apply convolution neural networks for solving real-word problems.
4. Learn to implement and apply recurrent neural networks for solving real-word problems.
5. Implement and improve deep learning models in TensorFlow and interpret the results.

List of Practicals

1. To implement Logistic Regression as a neural network for classification problem.
2. To implement one hidden layered neural network for planar data classification
3. To implement deep neural network with forward and backward propagation.
4. To implement deep neural network for image classification.
5. To implement various advanced optimization methods.
6. To build Sign language classifier using neural network in TensorFlow library.
7. To build Sign language classifier using the TensorFlow CNN Convolutional Neural Networks.
8. To implement Emotion Detection in Images of Faces using Keras.
9. To implement the ResNets for image classification.
10. To implement Character level language model using RNNs.
11. To implement a deep learning model using LSTM to generate music.
12. To implement word vector representations to build an Emojifier.