RV UNIVERSITY

Go, change the world

an initiative of RV EDUCATIONAL INSTITUTIONS

NEW-AGE GLOBAL UNIVERSITY
FOR LIBERAL EDUCATION

School of Computer Science and Engineering

Where Ideas Ignite Minds

B.Sc. Computer Science (Hons.)

(2026 Scheme)

I and II Semester
Academic Year 2026-27

Syllabus Book

Programmes:

1. B.Tech. Honors(CSE)
2. B.Sc. Honors(CSE)
3. B.C.A. Honors(CSE)
4. M.Tech. (CSE)
5. M.Tech. Data Science (in collaboration with upGrad Campus) for working professionals
6. Ph.D.

*Proposed infrastructure under construction

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

RV Vidyaniketan, 8th Mile, Mysuru Road, Bengaluru, 560059, India
Ph : +91 80 68199900 | www.rvu.edu.in

School of Computer Science and Engineering (SoCSE)

Bachelor of Science (Honors)

B.Sc(Hons.)

I and II Semester
(2026 Scheme)

Academic Year 2026-27

RV University (RVU)

RV Educational Institutions (RVEI), governed by Rashtreeya Sikshana Samithi Trust (RSST), is recognised among the few value–based and quality–oriented educational groups in the country. The Trust endeavours to impart quality education to all strata of society. RV University is a State Private University which has been established in Karnataka State with RSST as the sponsoring body through Act No.11 of 2019, passed by the Karnataka Legislature. The Missions of RSST and RV University is "Excellence in Education with Societal Commitment".

RVU Vision

To be a World–class, tech–driven, global university for liberal education, empowering citizens of tomorrow.

RVU Mission

RVU M1:
Strive for excellence in teaching, research, capacity building, and community engagement, benchmarking against global universities to lead across disciplines.

RVU M2:
Utilize digital and emerging technologies to enhance teaching-learning and research, accessible to all, while fostering a multidisciplinary, inclusive environment that meets evolving learner needs.

RVU M3:
Cultivate a diverse, global academic community through strong national and international collaborations that enrich learning, facilitate mobility, and drive institutional growth.

RVU M4:
Integrate theory with practical application to develop self-driven, empathetic problem - solvers equipped to create meaningful societal impact.

School of Computer Science and Engineering (SoCSE)

The School of Computer Science and Engineering (SoCSE) focuses on problem–solving, critical thinking, innovation, creativity, communication, entrepreneurship data science to deal with the VUCA (Volatility, Uncertainty, Complexity, and Ambiguity) world. All programmes of the school offer an opportunity to the students to work closely with all stakeholders – industries, government policymakers, researchers, think tanks and

global organizations. School focuses on imparting 21st century skills through experiential, holistic learning. Curriculum supports interdisciplinary studies with minors from other schools and includes internships and student exchange programs with foreign universities.

SoCSE Vision

To be a pioneering school of Computer Science and Engineering committed to fostering liberal education and empowering the next generation of technologists to make a positive global socio-economic impact.

SoCSE Mission

SoCSE M1:
To be a pioneer in computer science education, fostering multidisciplinary research, innovation and entrepreneurship.

SoCSE M2:
To provide state-of-the-art facilities and dynamic curriculum that enables exemplary pedagogy, integrating advanced technology, theoretical foundations, and hands–on applications to address real-world challenges.

SoCSE M3:
To promote diverse communities of faculty and students through national and international academic, industry collaborations contributing to institution-building.

SoCSE M4:
To cultivate a generation of ethical, self-motivated, and empathetic problem solvers dedicated to achieving sustainable development goals.

About the Program – B.Sc(Hons.)

Computer Science is increasingly evolving from an applied branch to a core discipline. Its diverse applications permeate every aspect of human existence. In our B.Sc(Hons.) program at RV University, we welcome students from any PUC/10+2 background, offering a curriculum that accommodates a broad mathematical foundation and nurtures future technocrats. This program strikes a balance between the aspiration to study computer science and the prerequisites needed to excel in the field. Students will have the opportunity to pursue multiple internships, preparing them for three typical career paths: employment, higher studies, and entrepreneurship. The B.Sc(Hons.) aims to establish a new standard in inclusive computer science education.

Program Outcomes (POs)

A graduate of the program will demonstrate:

PO1 — Application of Knowledge:
Apply knowledge from the core/interdisciplinary areas and develop skills to solve complex real-world problems.

PO2 — Critical Thinking & Creativity:
Analyze and evaluate the existing situations, practices and evidence to develop innovative multi-perspective solutions.

PO3 — Research & Innovation:
Cultivate a keen sense of observation and enquiry using appropriate methodology to develop new ideas, processes & solutions in response to contemporary challenges.

PO4 — Digital Transformation:
Instill an ability for confident, critical and responsible use of digital technologies for learning, work and participation in society.

PO5 — Communication & Interpersonal Skills:
Develop skills to effectively express thoughts and ideas both orally and in writing to people from all sections of the society.

PO6 — Leadership & Collaborative Teamwork:
Exhibit skills to set goals and guide people to align effectively in building a team to achieve the goals.

PO7 — Project Management & Entrepreneurial Mindset:
Foster an enterprising spirit by working on diverse projects, demonstrating time management, resource allocation and risk assessment capabilities to create value.

PO8 — Lifelong Learning:
Engage in continuous self-directed learning to adapt to evolving technologies, societal changes and emerging global challenges.

PO9 — Global Perspective & Cultural Awareness:
Acquire knowledge about the values and beliefs of diverse cultures, effectively engaging with empathy.

PO10 — Ethical Reasoning & Social Responsibility:
Demonstrate moral and value-based principles while being socially conscious and accountable.

PO11 — Civic Engagement & Community Service:
Participate actively in civic life exhibiting a commitment to community service, social justice and the common good.

PO12 — Environmental Sustainability:
Show responsibility to manage and work towards achieving the Sustainable Development Goals (SDGs).

Program Educational Objectives (PEOs) – B.Sc(Hons.)

PEO1: Graduates will be able to apply their knowledge and critical thinking to solve real- world problems, demonstrate leadership and ethics, and contribute to societal progress through lifelong learning and interdisciplinary collaboration.

PEO2: Graduates will drive technological innovation and entrepreneurship, leveraging multidisciplinary research to develop sustainable solutions that address societal challenges.

PEO3: Graduates will engage in global academic and industry collaborations, utilizing advanced computing technologies to lead transformative initiatives for organizational development.

Program Specific Outcomes (PSOs) – B.Sc(Hons.)

A graduate of the B.Sc(Hons.) will also demonstrate;

PSO1: Professional Readiness for Evolving Tech Ecosystems:
Demonstrate readiness for professional roles, higher studies, research, or entrepreneurship by continuously adapting to emerging technologies.

PSO2: Tech-Driven Community Engagement:
Engage with societal challenges to develop impactful, technology- driven solutions that serve and empower communities.

PSO3: Building Practical Solutions with Computing Principles:
Apply computing concepts to build reliable systems and solve challenges in emerging areas considering practical and ethical aspects.

Credit Distribution (4-Year UG Honours — 160 Credits)

No.	Category	Credits
1	Core – Major (Core)	83
2	Minor – Minor Stream	32
3	MD – Multidisciplinary	8
4	AEC – Ability Enhancement Courses	8
5	SEC – Skill Enhancement Courses	9
6	VAC – Value Added Courses	8
7	SI – Summer Internship	2
8	Project / Dissertation	12
	Total	**162**

B.Sc(Hons.) Course Matrix

Sem	Core	Minors	MD	AEC	SEC	VAC	SI	PD	Credits
SEM 1	16				2	2			**20**
SEM 2	16			2	2	2			**22**
SEM 3	8	6	2			2	2		**20**
SEM 4	8	6	2	3	3				**22**
SEM 5	13	6	2		2				**23**
SEM 6	9	6	2	3		2			**22**
SEM 7	9	4						4	**17**
SEM 8	4	4						8	**16**
Total	**83**	**32**	**8**	**8**	**9**	**8**	**2**	**12**	**162**

B.Sc(Hons.)2026-27 Course Scheme — Semester 1

Type	Code	Subject	Cr.	L–T–P	Internal		Sem. End		Pass
					Max	Min	Max	Min	
Core	CS1831	Elements of Linear Algebra	2	1-1-0	70	28	30	12	40
Core	CS1081	Programming with C	4	3-0-2	70	28	30	12	40
Core	CS1385	Web and UX Design	3	1-1-2	70	28	30	12	40
Core	CS1150	Digital Logic and Computer Organisation	4	2-1-2	70	28	30	12	40
Core	CS1104	Linux Shell Programming	3	2-0-2	70	28	30	12	40
SEC	CS1929	Structured Innovation with Design Thinking	2	1-0-2	70	28	30	12	40
VAC	CS1908	Universal Human Values	2	2-0-0	70	28	30	12	40
Total			**20**						

B.Sc(Hons.)2026-27 Course Scheme — Semester 2

Type	Code	Subject	Cr.	L–T–P	Internal		Sem. End		Pass
					Max	Min	Max	Min	
Core	CS1812	Discrete Maths and Set Theory	3	2-1-0	70	28	30	12	40
Core	CS1280	Python Programming for Data Analytics	3	1-0-4	70	28	30	12	40
Core	CS1083	Data Structures using C	4	2-0-4	70	28	30	12	40
Core	CS1103	Operating Systems	3	2-0-2	70	28	30	12	40
Core	CS2201	Database Management Systems	3	1-1-2	70	28	30	12	40
AEC	CS1927	English Communication	2	0-1-2	70	28	30	12	40
SEC	CS1940	Entrepreneurial Mindset	2	1-0-2	70	28	30	12	40
VAC	CS1105	Introduction to Version Control	2	0-1-2	70	28	30	12	40
Total			**22**						

L – Lecture, T – Tutorial, P – Practical / Project

Semester - **I**			
Course: **ELEMENTS OF LINEAR ALGEBRA**			
Program	**B.Sc. (Hons) CSE**	Category	**Major (Core)**
Course Code	**CS1831**	CIE Marks	**70**
Credits (L:T:P)	**2 (1:1:0)**	SEE Marks	**30**
Hours	**15L + 15T + 0P = 30**	SEE Mode	**Theory**

Course Objectives: students will be able to	
1	To introduce the basic concepts of vectors and vector algebra for solving geometric and physical problems.
2	To provide a foundational understanding of matrices and determinants and their applications in linear systems.
3	To enable students to solve systems of linear equations using matrix-based methods such as Gauss Elimination and Inverse method.
4	To familiarize students with linear transformations, eigenvalues, and eigenvectors for analyzing linear mappings and systems.

Module - 1	**3 hours**
Basics of vector, vector algebra, dot product, cross product,coplanarity, Linear combination of vectors, Linear dependence, Linear independence, basis	

Module - 2	**3 hours**
Matrix, matrix algebra, minor, cofactor, matrix transpose, adjoint matrix, special matrices, determinant, algebra of determinant.	

Module - 3	**3 hours**
Row Echelon form, Rank of a matrix, Solving system of linear equations using various methods: Inverse method, Gauss Elimination, Row canonical form,Gauss Jordan Method	

Module - 4	**3 hours**
Linear transformation, Properties of linear transformation, Rank, nullity, change of coordinates	

Module - 5	**3 hours**
Eigenvalues and eigenvectors, Characteristic equation ,Diagonalizability, Geometric interpretation and application.	

Course Outcomes: After completing the course, the students will be able to	
1	Understand the vector algebra concepts to solve mathematical and engineering problems effectively.
2	Apply matrix algebra and determinants to solve complex mathematical problems.
3	Evaluate systems of linear equations using various methods.
4	Analyze problems involving calculation related to eigenvalue and eigenvector for a given matrix.

Text Books	
1	G. Shanker Rao, Linear Algebra, 1st edition, Dreamtech Press, (2020). ISBN: 978-9389583304.
2	B.S. Grewal, Higher Engineering Mathematics, 45th edition, Khanna Publications,(2023).ISBN: 978-8900120905.
3	Gilbert Strang, Introduction to Linear Algebra, 6th edition, Wellesley Cambridge Pr, (2023). ISBN:978-1733146678.

Reference Books and Resources	
1	Lipschitz and Seymour, 3000 solved problems in linear algebra, 1st edition, McGraw-Hill Education, (1989). ISBN:978-0070380233.
2	D Lay, J McDonald and S Lay, Linear Algebra and Its Applications, 6th Edition - Pearson Education Limited, (2021). ISBN: 978-1292351216.
3	Erwin Kreyszig, Advanced Engineering Mathematics, John Wiley and Sons 11th Edition (2026). ISBN: 978-9373329949.
4	NPTEL: Linear Algebra through Geometry-IISc, Bangalore https://nptel.ac.in/courses/106108482Basic Calculus - 1, IIT: - 1,
5	NPTEL: Applied Linear Algebra-IIT, Bombay https://nptel.ac.in/courses/108101371
6	MIT-OCW: Linear Algebra, MIT https://ocw.mit.edu/courses/18-06-linear-algebra-spring-2010/

Tutorials	15 Hours
1	Vectors (Desmos Activity : Visualization)
2	Dot and Cross Product
3	Linear Combination, dependence, independence
4	Matrix operations
5	Determinants, minors and cofactors
6	Adjoint, inverse of a matrix (Programming activity)
7	Row operations

Tutorials (continued)	
8	Gauss elimination (Programming activity)
9	Gauss Jordan (Programming activity)
10	Linear transformations
11	Null space, column space
12	Rank and nullity
13	Eigenvalues and eigenvectors (Programming activity)
14	Characteristic polynomial
15	Diagonalization (Programming activity)

Semester - **I**			
Course: **PROGRAMMING WITH C**			
Program	**B.Sc. (Hons) CSE**	Category	**Major (Core)**
Course Code	**CS1081**	CIE Marks	**70**
Credits (L:T:P)	**4 (3:0:2)**	SEE Marks	**30**
Hours	**45L + 0T + 30P = 75**	SEE Mode	**Theory**

Course Objectives: students will be able to	
1	Introduce the fundamental principles of programming and develop essential problem-solving skills.
2	Build proficiency in core C programming constructs, including control structures, functions, arrays, pointers, and strings.
3	Enable students to write, debug, and optimize efficient C programs through practical implementation.
4	Lay a strong foundation for exploring advanced computing topics and further programming languages.

Module - 1	**9 hours**
Introduction to Programming, History and features of the C language, Structure of a C program. Basic syntax: keywords, identifiers, constants, variables, and data types, Operators and expressions, Input and output functions: printf(), scanf().	

Module - 2	**9 hours**
Decision-making statements: if, if-else, nested if-else, switch, Looping statements: while, do-while, for, break, continue and goto statements.	

Module - 3	**9 hours**
Functions: declaration, definition, and calling, Function types: with/without arguments, with/without return value, Storage classes, Recursion, Arrays: one-dimensional and two-dimensional arrays.	

Module - 4	**9 hours**
Introduction to pointers, Pointer arithmetic, Pointers and arrays. Strings: declaration, initialization, and string handling functions.	

Module - 5	**9 hours**
Structures: Definition, declaration, and initialization, Arrays of structures, Pointers to structures, Unions. File handling: File operations, reading from and writing to files.	

Course Outcomes: After completing the course, the students will be able to	
1	Understand the fundamental concepts and basic syntax rules of C programming.
2	Analyze decision-making and looping constructs, and apply string manipulation techniques to solve programming problems.
3	Apply functions, arrays, and pointers, including dynamic memory allocation, to develop modular and efficient code.
4	Create user-defined data types such as structures and unions to organize and manage complex data effectively.

Text Books	
1	Balagurusamy, E. Programming in ANSI C (8th ed.). McGraw Hill Education. (2019)ISBN: 978-9351343202
2	Kanetkar, Y. Let us C (19th ed.). BPB Publications. (2022) ISBN: 978-8183331630
3	Gonzalez-Morris, G., & Horton, I. Beginning C: From Beginner to Pro (7th ed.). Apress.(2024). ISBN: 979-8868801488

Reference Books and Resources	
1	Kernighan, B. W., & Ritchie, D. M. The C programming language (2nd ed.). Prentice Hall. (2015) ISBN: 978-9332549449
2	Schildt, H. C: The complete reference (4th ed.). McGraw Hill Education. (2017). ISBN: 978-0070411838
3	Rajaram, M. Computer Programming with C. Pearson India. (2024). ISBN: 978-9332568822

Lab Programs / Practical	30 Hours
Part A: Lab Programs	
1	Write a C program to print your name entered by the user.
2	Write a C program to perform arithmetic operations.
3	Write a C program to swap 2 numbers a. without using a temporary variable b. using a temporary variable
4	Write a C program to find the largest of three numbers.
5	Write a C program to print the first 10 Fibonacci numbers.
6	Write a C program to print the following Half Pyramid of Numbers pattern. 1 1 2 1 2 3 1 2 3 4 1 2 3 4 5
7	Write a C program to calculate the factorial of a number.
8	Write a C program to check whether a number is prime.
9	Write a recursive function to find the GCD of two numbers.

Lab Programs / Practical (continued)	
10	Write a function to perform matrix addition and multiplication.
11	Write a program to swap two numbers using pointers.
12	Write a program to reverse a string.
13	Write a C program to Find the Frequency of Characters in a String.
14	Write a program to store and display information of a student using structures.
15	Write a program to read and write a file.
16	Write a program to count the number of lines, words, and characters in a file.
Part B: Mini Project	
1	Develop a basic application in C Project Options Choose any one 1. Simple Banking System Deposit, withdraw, check balance Use functions + file handling 2. Library Management System (Basic) Add books, issue/return books Use arrays/structures 3. Quiz Application Multiple choice questions Score calculation Use loops, arrays, functions Mini Project Requirements: The mini project must be a menu-driven application that demonstrates the use of functions, arrays or structures, and file handling, while ensuring proper input/output formatting and basic error handling

Semester - **I**			
Course: **WEB AND UX DESIGN**			
Program	**B.Sc. (Hons) CSE**	Category	**Major (Core)**
Course Code	**CS1385**	CIE Marks	**70**
Credits (L:T:P)	**3 (1:1:2)**	SEE Marks	**30**
Hours	**15L + 15T + 30P = 60**	SEE Mode	**Jury**

Course Objectives: students will be able to	
1	Introduce core technologies, tools, and structural principles of modern web development.
2	Develop the ability to design user-centered, accessible, and responsive websites.
3	Design user-centered solutions using UX processes including research, prototyping, heuristic evaluation, and usability testing.
4	Provide hands-on experience in building, version-controlling, and testing web interfaces.

Module - 1	**3 hours**
Web and HTML Fundamentals : Introduction to the Web and Web Development, Basic Structure of an HTML Document, Basic HTML Tag and Attributes, Semantic HTML5 Tags and Accessibility Considerations, Creating Links, Lists, Tables and Forms	

Module - 2	**3 hours**
CSS Syntax and Selectors, Styling Text and Colors, CSS Box Model, Modern Layout Techniques: Flexbox and Grid, Responsive Design and Media Queries (Mobile-First Approach).	

Module - 3	**3 hours**
JavaScript Syntax and Programming Concepts, Variables, Operators and Control Structures, DOM Manipulation, Event Handling, Introduction to Browser DevTools and Debugging.	

Module - 4	**3 hours**
UX Design Fundamentals : Introduction to UX Design and Its Importance, User Research and Personas, Wireframing and Prototyping (using tools such as Figma), Usability Testing, Heuristic Evaluation (Nielsen's Principles), Basic Accessibility Standards-Web Accessibility and WCAG Basics.	

Module - 5	**3 hours**

Introduction to Server-Side Scripting and PHP, Basic PHP Syntax and Variables, Form Handling and Dynamic Web Page Development, Client-Server Interaction and Data Processing, Integration of HTML, CSS, JavaScript, UX Principles and PHP, Version Control using Git, Documentation and Responsible Use of AI Tools.

Course Outcomes: After completing the course, the students will be able to	
1	Apply semantic HTML5 and CSS layout techniques to create responsive web pages.
2	Manipulate the DOM using JavaScript and debug using browser tools.
3	Apply UX principles including research, prototyping, and heuristic evaluation to design accessible interfaces.
4	Develop PHP-based server-side applications and manage projects using version control tools

Text Books	
1	Duckett, J. (2023). HTML and CSS: Design and Build Websites. John Wiley & Sons. ISBN: 978-1-118-00818-8
2	Duckett, J. (2022). JavaScript and jQuery: Interactive Front-End Web Development. John Wiley & Sons. ISBN: 978-1-118-53164-8
3	Unger, R., & Chandler, C. (2021). A Project Guide to UX Design: For User Experience Designers in the Field or in the Making (3rd ed.). New Riders. ISBN: 978-0-13-431898-6
4	Nixon, R. (2023). Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5 (6th ed.). O'Reilly Media. ISBN: 978-1-098-12384-9

Reference Books and Resources	
1	Robbins, J. N. (2023). Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics (6th ed.). O'Reilly Media. ISBN: 978-1-492-04999-2
2	Haverbeke, M. (2024). Eloquent JavaScript: A Modern Introduction to Programming (4th ed.). No Starch Press. ISBN: 978-1-71850-239-7
3	Welling, L., & Thomson, L. (2020). PHP and MySQL Web Development (5th ed.). Addison-Wesley. ISBN: 978-0-672-33784-0
4	Gothelf, J., & Seiden, J. (2021). Lean UX: Designing Great Products with Agile Teams (3rd ed.). O'Reilly Media. ISBN: 978-1-492-04654-0

Tutorials	**15 Hours**
1	Introduction to Web Development: Learn how the web works, the roles of browsers and servers, and create your first HTML webpage.
2	HTML Structure and Semantic Tags: Understand HTML elements, attributes, and semantic HTML5 tags to structure webpages effectively.

Tutorials (continued)	
3	Organizing Content: Use tables, lists, and forms to organize and collect data on webpages.
4	Introduction to CSS: Explore CSS syntax, selectors, and ways to style webpages using inline, internal, and external methods.
5	Styling Text and Colors: Apply fonts, text styles, colors, and background designs to enhance webpage aesthetics.
6	CSS Layout Principles: Master the box model, positioning, and modern layouts using Flexbox and Grid.
7	Responsive Design: Use media queries and mobile-first principles to make webpages responsive and user-friendly.
8	JavaScript Syntax Basics: Learn variables, operators, and control structures to write simple JavaScript code.
9	DOM Manipulation: Modify HTML content dynamically using the Document Object Model (DOM).
10	Event Handling and Validation: Add interactivity with event listeners like clicks and keypresses, and implement basic form validation.
11	Introduction to UX Design: Understand user-centered design and the importance of creating user-friendly interfaces.
12	User Research: Conduct surveys and interviews to gather insights for designing better webpages.
13	Wireframing and Prototyping: Plan layouts and navigation using tools such as Figma to create wireframes and prototypes.
14	Accessibility in Web Design: Follow WCAG standards to make webpages inclusive and accessible for all users.
15	Combine HTML, CSS, JavaScript, PHP, UX principles, and version control workflow to design and develop a dynamic web application.

Lab Programs / Practical	**30 Hours**
Part A: Lab Programs	
1	Write a program to develop a basic webpage using semantic HTML5 elements with a heading, paragraph, and image, ensuring consistent styling using CSS.
2	Write a program to design a feedback form using various HTML form elements and validation attributes.
3	Write a program to build a table to display a schedule.
4	Write a program to create a two-column layout using modern CSS techniques (Flexbox or Grid).
5	Write a program to make a web page responsive using media queries.

Lab Programs / Practical (continued)	
6	Write a program to style a form and table for better usability and visual appearance.
7	Write a JavaScript program to perform basic arithmetic operations.
8	Write a JavaScript program to manipulate the content and style of HTML elements using the DOM.
9	Write a program to create an interactive form that validates user input using JavaScript.
10	Build a simple to-do list application with dynamic DOM manipulation.
11	Write a program to create a simple interactive web page integrating HTML, CSS, and JavaScript.
12	Write a program to design a sitemap and navigation structure using HTML and CSS.
Part B: Integrated Project	
1	Design and develop a complete functional webpage integrating HTML, CSS, JavaScript, UX principles, and submit the project using basic version control (Git).

Semester - **I**			
Course: **DIGITAL LOGIC AND COMPUTER ORGANISATION**			
Program	**B.Sc. (Hons) CSE**	Category	**Major (Core)**
Course Code	**CS1150**	CIE Marks	**70**
Credits (L:T:P)	**4 (2:1:2)**	SEE Marks	**30**
Hours	**30L + 15T + 30P = 75**	SEE Mode	**Theory**

Course Objectives: students will be able to	
1	To understand digital logic fundamentals and the functional organization of a computer system.
2	To implement combinational and sequential logic circuits.
3	To examine instruction set architecture and processor operations.
4	To construct control units and integrate processor, memory, and I/O subsystems.

Module - 1	**6 hours**
Block Diagram of Digital Computer. Number Systems: Decimal, Binary, Hexadecimal and Octal. Conversions, Basic Logic Gates, Boolean Algebra, Simplification of Boolean Functions using K-MAP, Sum of Product, Product of Sum simplification, Don't Care Conditions.	

Module - 2	**5 hours**
Combinational Logic Circuits: Adder: Half-Adder, Full-Adder, 4-bit Parallel Adder, Carry Look Ahead Adder. Subtractor, Multiplexer, Demultiplexer, Encoder, Decoder.	

Module - 3	**6 hours**
Sequential Circuits: Latches, flip-flops: SR, D, JK and T Flip Flop. Master-slave flip flop. Designing sequential circuits: state table, state diagram, state equations, Flip flop excitation table (JK and D flip flop).	

Module - 4	**5 hours**
Overview of computer components and functions, CPU, memory, I/O and control unit. Instruction Set Architecture (ISA), Types of instructions, instruction formats, Addressing modes: Implied, Register, Direct, Indirect, Register-indirect, Indexed and Base register mode. Data Path and Control Path, Arithmetic Logic Unit (ALU), Instruction Cycle.	

Module - 5	**8 hours**

Fundamental Concepts: Register Transfer, Performing an arithmetic and logic operation, Load and Store operation in memory. Control Unit Design: Hardwired Control and Microprogrammed Control. Memory Organization: SRAM and DRAM. Interrupt driven I/O and DMA

Course Outcomes: After completing the course, the students will be able to	
1	Apply number system conversion techniques and represent signed numbers using complement methods.
2	Analyze and simplify Boolean expressions using algebraic methods and Karnaugh maps.
3	Design combinational logic circuits using standard components such as adders, multiplexers, decoders, and encoders.
4	Design sequential circuits using flip-flops and shift registers for specified applications.

Text Books	
1	M. M. Mano and M. D. Ciletti, Digital Design, Sixth edition. Harlow, UK: Pearson Education, 2018, ISBN: 978-1292231181.
2	D. A. Patterson and J. L. Hennessy, Computer Organization and Design: The Hardware/Software Interface, Fifth edition. Waltham, MA, USA: Morgan Kaufmann, 2013, ISBN: 978-0128201091.
3	V. C. Hamacher, Z. G. Vranesic, and S. G. Zaky, Computer Organization. New York, NY, USA: McGraw-Hill, 2002, ISBN: 978-0070256811.
4	W. Stallings, Computer Organization and Architecture: Designing for Performance, Eighth edition. Noida, India: Pearson Education, 2016, ISBN: 978-8131732458.

Reference Books and Resources	
1	A. S. Tanenbaum and T. Austin, Structured Computer Organization, Sixth edition. Boston, MA, USA: Pearson, 2013, ISBN: 978-0132916523.
2	L. Null, Essentials of Computer Organization and Architecture, Sixth edition. Burlington, MA, USA: Jones & Bartlett Learning, 2023, ISBN: 978-1284259438.

Tutorials	**15 Hours**
1	Convert numbers between decimal, binary, octal, and hexadecimal systems for a given dataset.
2	Perform binary addition and subtraction using 2's complement representation.
3	Construct truth tables and implement basic logic gates using Boolean expressions.

Tutorials (continued)	
4	Design logic circuits using only NAND and NOR gates for given functions.
5	Simplify Boolean expressions using algebraic laws and verify the results.
6	Validate De Morgan's theorems using truth tables and circuit realization.
7	Minimize Boolean functions using 2- and 3-variable K-maps and implement the circuits.
8	Minimize Boolean functions using 4-variable K-maps and realize the simplified circuits.
9	Optimize Boolean expressions using K-maps with don't-care conditions and implement the result.
10	Apply register transfer operations and compute results for given micro-operations.
11	Execute arithmetic and logic micro-operations on given register values.
12	Demonstrate with examples of memory read and write operations for given address and data inputs.
13	Show performance comparison of SRAM and DRAM units.
14	Demonstrate the use of CPU Sim by executing sample instructions and observing register/memory operations. Link: https://www.cs.colby.edu/djskrien/CPUSim.
15	Execute and analyze a given instruction sequence using CPU Sim and interpret the results

Lab Programs / Practical	**30 Hours**
1	Design and implement basic logic gates using Boolean expressions and verify their operation.
2	Design and implement logic gates using only universal gates (NAND/NOR).
3	Design and verify a half-adder and full-adder circuit using basic building blocks.
4	Design and implement a multi-bit adder–subtractor circuit using standard components.
5	Design and realize a carry lookahead adder using suitable logic techniques.
6	Design and construct a multiplexer and demultiplexer circuits using basic gates.
7	Design and construct encoder and decoder circuits using basic gates.
8	Design and implement a parity generator and parity checker circuit.
9	Design and verify an SR flip-flop and analyze its behavior.
10	Design and implement a D flip-flop and evaluate its timing characteristics.

Lab Programs / Practical (continued)	
11	Design and implement a JK flip-flop and verify its functionality
12	Design and implement a T flip-flop using an appropriate flip-flop configuration.
13	Design and realize sequential circuits using the excitation table of a JK flip-flop.
14	Design and realize sequential circuits using the excitation table of a D flip-flop.
15	Design and implement a simple sequential application (e.g., counter or shift register) using flip-flops.

Semester - **I**			
Course: **LINUX & SHELL PROGRAMMING**			
Program	**B.Sc. (Hons) CSE**	Category	**Major (Core)**
Course Code	**CS1104**	CIE Marks	**70**
Credits (L:T:P)	**3 (2:0:2)**	SEE Marks	**30**
Hours	**30L + 0T + 30P = 60**	SEE Mode	**Practical**

Course Objectives: students will be able to	
1	To interpret the Linux operating system architecture and various distributions to effectively navigate the file system and manage files and directories.
2	To apply shell scripting concepts such as variables, input/output operations, control structures, loops, and functions to automate routine tasks and improve system productivity.
3	To analyze text files using Linux text processing tools such as grep, sed, and awk, along with regular expressions for efficient data extraction and manipulation.
4	To evaluate the process management techniques and command-line customization features to optimize system performance and enhance user environment usability.

Module - 1	**6 hours**
Introduction to Linux and Basic Commands: Overview of Linux and distributions, Linux directory structure (/bin, /etc, /home, /var) and architecture. Basic Commands: File management commands (ls, cp, mv, rm, mkdir, rmdir), File search and disk usage (find, locate, df, du), Viewing file content (cat, more, less, head, tail), File permissions and ownership (chmod, chown, chgrp)	

Module - 2	**6 hours**
Shell Basics and Scripting Fundamentals: Overview of Shell: Introduction to shell types (Bash, Zsh, etc.), Creating and executing simple shell scripts, Working with Variables: Defining and using variables in shell scripts, Variables Environment vs. local variables, Special variables ($?, $#, $*, $@), Command-line arguments. Input/Output Operations using echo for output, reading user input with read.	

Module - 3	**6 hours**

Control Structures in Shell Scripts: Control Structures: Conditional statements (if, else, elif, case), Looping constructs (for, while), break and continue, Functions in Shell Scripting: Defining and calling functions, passing arguments to functions, Menu driven scripts using select.

Module - 4	**6 hours**
Text Processing Basics: Text Processing Tools: Introduction to filters (grep, sed), Basic text manipulation with awk, Pipes and redirection (—, ¿, ¿¿), Text processing commands (sort, uniq, cut, tr), Regular Expressions: Basics of regular expressions in Linux, Using regex with tools like grep.	

Module - 5	**6 hours**
Process and Customizing the Environment : Process Basics: Process commands (ps, top), Internal and external commands, Running jobs in the background like (jobs, bg, fg), nice, at, batch, cron, time commands, Background execution (&), Customizing the Environment: .bashrc, System variables, .profile, sty, PWD, Aliases, Command history.	

Course Outcomes: After completing the course, the students will be able to	
1	Understand the fundamental concepts and architecture of the Linux operating system.
2	Apply Linux commands to manage files, processes, and system operations effectively.
3	Develop and debug shell scripts to automate tasks within a Linux environment.
4	Analyze and process data using Linux text-processing tools such as grep, sed, and awk to solve real-world problems.

Text Books	
1	W. Soyinka, Linux Administration: A Beginner's Guide, Sixth edition. New Delhi, India: McGraw Hill Education, 2017, ISBN: 978-1259061189.
2	J. Arthur, Linux: A Complete Guide to Linux Command Line for Beginners, and How to Get Started with the Linux Operating System!, First edition. USA: Ingram Publishing, 2019, ISBN: 978-1925989717.
3	S. Das, UNIX: Concepts and Applications, Fourth edition. New Delhi, India: McGraw Hill Education, 2017, ISBN: 978-0070635463.

Reference Books and Resources	
1	R. K. Michael, Mastering Unix Shell Scripting: Bash, Bourne, and Korn Shell Scripting for Programmers, System Administrators, and UNIX, Second Kindle edition. Indianapolis, IN, USA: Wiley, 2008, ISBN: 978-0470183014.

Reference Books and Resources (continued)	
2	S. K. Joseph, Mastering Linux: A Comprehensive Guide to the Operating System, First edition. Chennai, India: Notion Press, 2023, ISBN: 979-8891330146.

Lab Programs / Practical	30 Hours
Part A: Lab Programs	
1	Perform file and directory operations such as creating, copying, moving, and deleting files and folders using Linux command-line commands.
2	Investigate and modify file permissions and ownership for different users and groups using appropriate Linux commands.
3	Search for files based on specific conditions and analyze disk usage of files and directories using Linux commands.
4	Write a shell script to display system information such as username, current date, system uptime, and present working directory.
5	Develop a shell script to perform basic arithmetic operations (addition, subtraction, multiplication, and division) based on user input.
6	Create a shell script to perform various string operations such as concatenation, length calculation, and substring extraction.
7	Write a shell script to generate the multiplication table for a specified number using looping constructs.
8	Develop a shell script to determine whether a given number is positive, negative, or zero using conditional statements.
9	Create a menu-driven shell script to perform basic file operations such as listing files, displaying the current directory, and viewing file contents.
10	Use Linux commands to count the number of words, lines, and characters in a specified file.
Part B: Lab Programs	
1	Write commands or a shell script to remove lines from a file that contain a specific word or pattern using text processing tools.
2	Perform text processing and data extraction from a file using tools such as awk, grep, or other Linux filters.
3	Use Linux commands to display currently running processes and filter specific processes based on given criteria.
4	Schedule and automate tasks using cron jobs to execute commands at specified time intervals.
5	Customize the Linux command-line environment by creating aliases and managing command history.

Semester - **I**			
Course: **STRUCTURED INNOVATION WITH DESIGN THINKING**			
Program	**B.Sc. (Hons) CSE**	Category	**SEC**
Course Code	**CS1929**	CIE Marks	**70**
Credits (L:T:P)	**2 (1:0:2)**	SEE Marks	**30**
Hours	**15L + 0T + 30P = 45**	SEE Mode	**Jury**

Course Objectives: students will be able to	
1	To apply design thinking methodologies and AI-assisted tools for analyzing, evaluating, and developing user-centered digital and physical products and prototypes.
2	To understand and implement sustainable practices and AI-supported workflows in prototype development and product innovation.
3	To analyze and enhance existing products by improving aesthetics, functionality, usability, and user experience.
4	To innovate within existing frameworks by developing creative prototypes and effectively communicating design processes, outcomes, and solutions.

Module - 1	**3 hours**
Introduction to Design Thinking: Overview of design thinking principles and stages. Use of AI tools for brainstorming, idea generation, and mapping design thinking workflows. Empathy in Design: Techniques for understanding and empathizing with users, including user interviews and persona creation. Use of AI tools for interview transcription, sentiment analysis, empathy mapping, and AI-assisted persona generation. Reverse Engineering of Websites and Apps: Analyzing existing digital products to understand user experience, interface design, workflow logic, and the thought process behind product development and decision-making. Storytelling as a Tool: Using storytelling to convey user needs and problem statements while balancing AI and creative thinking. Case Studies: Analyzing successful design thinking projects that started with empathy.	

Module - 2	**3 hours**
This module introduces recycling and upcycling concepts, their environmental impact, and the role of sustainability in modern product design. Students will explore the triple constraints of design time, scope, and budget and understand how they influence prototype development. The module also integrates AI-assisted prototyping for idea generation, concept visualization, and virtual product development. Through hands-on activities, students will design and create simple prototypes using recycled materials while applying sustainable and AI-supported design practices.	

Module - 3	3 hours
Understanding the influence of aesthetics in consumer behavior and product design. Adaptation of AI tools for visual trend analysis and aesthetic exploration. Dismantling and evaluating existing products to understand their design and functionality. Enhancement Techniques: Methods for improving product design, focusing on aesthetics and functionality. Consumer Feedback: Gathering and utilizing consumer feedback to guide redesign efforts. Reconstruction Projects: Students reconstruct an existing product with improved aesthetics and functionality.	

Module - 4	3 hours
Innovation vs. Invention: Exploring the difference and focusing on innovation within existing frameworks. Analyzing Market Trends: How to use market research to identify opportunities for innovation. Creative Adaptations: Techniques for creative thinking and adaptation of existing products. Prototyping with Purpose: Building prototypes that address specific user needs or market gaps. Case Studies and Discussions: Use of AI tools for market trend analysis, idea generation, rapid prototyping, product visualization, and AI-assisted innovation research using generative AI-based design tools.	

Module - 5	3 hours
This module focuses on storytelling techniques, presentation skills, and effective communication in product development. Students will learn to create compelling narratives, develop engaging presentations, and use AI-based visual and digital tools to enhance communication. The module also covers stakeholder pitching, project documentation, and incorporating feedback for continuous improvement.	

Course Outcomes: After completing the course, the students will be able to	
1	Apply design thinking, AI-assisted analysis, and reverse engineering techniques to evaluate and develop user-centered digital, physical products and prototypes.
2	Process for sustainable & AI prototype development
3	Enhance existing products focusing on market aesthetics
4	Innovate within frameworks to create new prototypes and communicate

Text Books	
1	Jake Knapp, John Zeratsky, and Braden Kowitz, "Sprint: How to Solve Big Problems and Test New Ideas in Just Five Days" Simon & Schuster (2016), ISBN: 978-1501121746, 1st ed.
2	Idris Mootee "Design Thinking for Strategic Innovation: What They Can't Teach You at Business or Design School" Wiley (2013), ISBN: 978-1118620120, 1st ed.

Reference Books and Resources	
1	Jeanne Liedtka and Tim Ogilvie "Designing for Growth: A Design Thinking Toolkit for Managers" Columbia University Press (2011) ISBN: 978-0231158381, 1st ed.
2	Pavan Soni, Design Your Thinking, Penguin Random House India Private Limited (2020), 978-0670094097, 1st ed.

Lab Programs / Practical	30 Hours
1	AI-Assisted Empathy Mapping and Rapid Prototype Visualization
2	Define & Ideate leveraging AI tools
3	Upcycling to create value for increasing existing problems
4	AI-Assisted Product Research and Prototype development
5	Product variation & Testing
6	Product development: A product with purpose

Semester - **I**			
Course: **UNIVERSAL HUMAN VALUES**			
Program	**B.Sc. (Hons) CSE**	Category	**VAC**
Course Code	**CS1908**	CIE Marks	**70**
Credits (L:T:P)	**2 (2:0:0)**	SEE Marks	**30**
Hours	**30L + 0T + 0P = 30**	SEE Mode	**Theory**

Course Objectives: students will be able to	
1	Understand the fundamental human aspirations of happiness and prosperity
2	Cultivate inner harmony and self-awareness
3	Develop harmonious relationships within the family and society
4	Understand the connection between humans and nature, and follow value-based principles in personal and professional life.

Module - 1	**6 hours**
Content and Process for Value Education: Self-Exploration; 'Natural Acceptance' and Experiential Validation; Basic Human Aspirations; Right understanding, Relationship and Physical Facility; fulfilment of aspirations; Understanding Happiness and Prosperity, Harmony at various levels.	

Module - 2	**6 hours**
Harmony in Myself: Understanding human being; Understanding the harmony of Self with the Body: Harmony in Self; Health; Program for Self and Health.	

Module - 3	**6 hours**
Understanding values in human-human relationship; meaning of Justice (universal values in relationships); mutual happiness; Harmony in the society; Visualizing a universal harmonious order in society- Undivided Society, Universal Order- from family to world family.	

Module - 4	**6 hours**
Understanding the harmony in Nature; Interconnectedness and mutual fulfilment among the four orders of nature- recyclability and self-regulation in nature; Understanding Existence as Co-existence of mutually interacting units in all- pervasive space; Holistic perception of harmony at all levels of existence.	

Module - 5	**6 hours**
Professional ethics: people friendly and eco-friendly production systems, appropriate technologies and management patterns. Case studies of typical holistic technologies, management models and production systems	

Course Outcomes: After completing the course, the students will be able to	
1	Understand the importance of Human values
2	Identify the role of value education in overall personality development
3	Interpret the important values that need to be cultivated
4	Develop character and competence

Text Books	
1	R. R. Gaur, R. Asthana, and G. P. Bagaria, A Foundation Course in Human Values and Professional Ethics, 2nd rev. ed. New Delhi, India: Excel Books, 2019. ISBN: 978-9387034471.

Reference Books and Resources	
1	R. R. Gaur, R. Asthana, and G. P. Bagaria, Teacher's Manual for A Foundation Course in Human Values and Professional Ethics, 2nd rev. ed. New Delhi, India: Excel Books, 2019. ISBN: 978-9387034532.
2	A. N. Tripathi, Human Values, 1st ed. New Delhi, India: New Age International Publishers, 2004. ISBN: 978-8122415822.

Semester - **II**			
Course: **DISCRETE MATHS AND SET THEORY**			
Program	**B.Sc. (Hons) CSE**	Category	**Major (Core)**
Course Code	**CS1812**	CIE Marks	**70**
Credits (L:T:P)	**3 (2:1:0)**	SEE Marks	**30**
Hours	**30L + 15T + 0P = 45**	SEE Mode	**Theory**

Course Objectives: students will be able to	
1	Introduce the fundamental concepts of discrete mathematics essential for computer science and engineering.
2	Focus on set theory principles to build a strong mathematical foundation for problem-solving in computing disciplines.
3	Describe and analyse fundamental concepts of logic and set theory, including propositions, predicates, set operations, and relations.
4	Explain the concepts of functions and combinatorics, emphasising their role in the development of algorithms, data structures, and computational models.

Module - 1	**4 hours**
Sets and operations on sets, Set-builder notations, types of sets, union, intersection, complement, difference, symmetric difference, their properties and laws, Venn diagrams, Cartesian Product, cardinality; multisets.	

Module - 2	**8 hours**
Propositions and truth tables, tautologies and contradictions, logical equivalence, algebra of propositions, logical implications, predicate logic.	

Module - 3	**6 hours**
Relations, functions–one-to-one, onto functions, Bijective function. The Pigeon-hole principle, composition of functions; invertibility and Inverse of function.	

Module - 4	**6 hours**
The Rules of Sum and Product, Permutations, Permutations with repetitions, Combinations.	

Module - 5	**6 hours**
Combinations – The Binomial Theorem, Combinations with Repetition, The Catalan numbers.	

Course Outcomes: After completing the course, the students will be able to	
1	Apply the concepts of set theory to solve problems in relevant domains.
2	Apply truth tables, logical equivalences, and predicate logic to solve reasoning-based problems.
3	Apply relations, functions, and the pigeonhole principle to solve computational problems.
4	Apply principles of combinatorics to solve basic counting problems.

Text Books	
1	Rosen K, Discrete Mathematics and Its Applications, McGrawHill, 8th edition, 2021, ISBN: 978-9390727353
2	N Ch S N Iyengar, V M Chandrasekaran, Discrete Mathematics, Vikas Publishers, First Edition, 2003, ISBN :978-8125913627

Reference Books and Resources	
1	Michael Huth Camp; Mark Ryan, "Logic in Computer Science: Modelling and Reasoning about Systems", Cambridge University Press, Second 2004, ISBN- (13):978-0521543101
2	Basavaraj S Anami and Venakanna S Madalli "Discrete Mathematics – A Concept based approach", Universities Press, 2016, ISBN-(13): 978-8173719998
3	Lipschitz, Lipson, Discrete Mathematics, McGrawHill, Second 1992, ISBN : 978- 0070380318
4	G. Shanker Rao, Discrete Mathematical Structures, New Age International, 2007 ; ISBN, 8122426697, 9788122426694
5	Discrete Mathematics - https://nptel.ac.in/courses/106108227
6	Discrete Mathematics - https://nptel.ac.in/courses/106106183
7	Introduction to the Theory of Sets - https://nptel.ac.in/courses/111107058

Tutorials	**15 Hours**
1	Venn diagrams.
2	Cartesian Product
3	Propositions and truth tables.
4	Tautologies and contradictions
5	Algebra of propositions
6	Predicate logic.
7	Bijective functions.
8	The Pigeon-hole principle - AI Tools and Visualization

Tutorials (continued)	
9	Inverse of function.
10	The Rules of Sum and Product.
11	Permutations with repetitions - AI Tools and Visualization
12	Combinations.
13	Combinations – The Binomial Theorem. - AI Tools and Visualization
14	Combinations with Repetition - AI Tools and Visualization
15	The Catalan Numbers.

Semester - **II**			
Course: **PYTHON PROGRAMMING FOR DATA ANALYTICS**			
Program	**B.Sc. (Hons) CSE**	Category	**Major (Core)**
Course Code	**CS1280**	CIE Marks	**70**
Credits (L:T:P)	**3 (1:0:4)**	SEE Marks	**30**
Hours	**15L + 0T + 60P = 75**	SEE Mode	**Practical**

Course Objectives: students will be able to	
1	Apply Python programming concepts including syntax, control structures, functions, and modular programming to solve basic computational problems.
2	Demonstrate the use of Python data structures, object-oriented programming concepts, and file handling techniques (CSV and JSON) for data processing tasks.
3	Utilize Python libraries such as NumPy and Pandas to perform data manipulation, cleaning, and basic statistical analysis.
4	Perform exploratory data analysis and create effective visualizations using appropriate Python tools to interpret data.

Module - 1	**3 hours**
Introduction to Python: History, features, applications. Basic Python syntax: variables, data types, and operators, Basic I/O operations, Control Structures: conditional statements (if, elif, else); loops (for, while, nested loops); and control flow (break, continue, pass). Functions and Modules: Function definition, parameters, return value, recursive functions, and modular programming.	

Module - 2	**3 hours**
Data Structures: Lists, tuples, sets, and dictionaries, List and dictionary comprehensions, Arrays and 2D arrays: creation, manipulation, iterations, nested loops Classes and objects, Instance variables and methods, Lambda functions and nonymous functions Basic use of map(), filter(), reduce() Working with JSON files File Handling: Reading and writing text files, Working with CSV files. Exception handling: Try and except, user defined exceptions.	

Module - 3	**3 hours**

NumPy: Creating and working with NumPy arrays (1D, 2D, n-dimensional), Indexing, slicing, and reshaping arrays, Pandas: Series and DataFrame creation, Basic operations on DataFrames: indexing, slicing, filtering, handling missing data: dropping and filling missing values. Data import from Excel files using Pandas. Descriptive Statistics and Inference: central tendencies: mean, median, and mode. Introduction to data aggregation functions. Variability and dispersion: range, variance, standard deviation. Analysing data distributions (e.g., skew, kurtosis)

Module - 4	**3 hours**
Data Cleaning and Preprocessing: Handling missing and duplicated data, Outlier Detection: Q1, Q3 using IQR. Correlation analysis and interpretation. Encoding categorical variables (label encoding, one-hot encoding), Scaling and normalization (min-max scaling, z-score normalization). Advanced Operations with Pandas: concatenating, merging, and joining DataFrames, groupBy operations, and aggregation.	

Module - 5	**3 hours**
Line plots, bar charts, histograms, scatter plots, customizing plots (titles, labels, and legends), subplots, figure layout, and visualizing IQR with box plots. Seaborn: Statistical plots: box plots, violin plots, pair plots, correlation heatmaps, customizing Seaborn plots (themes, color palettes). Introduction to Plotly. Case Studies: Exploratory data analysis (EDA) on real-world datasets, Examples from domains such as finance, healthcare, and sports.	

Course Outcomes: After completing the course, the students will be able to	
1	Apply Python programming fundamentals, functions, and data structures to develop programs for solving basic computational and data processing problems.
2	Implement file handling, exception handling, and object-oriented programming concepts to manage and organize data efficiently.
3	Perform data preprocessing, statistical analysis, and data manipulation using NumPy and Pandas libraries.
4	Analyze datasets and communicate insights through effective visualizations and basic exploratory data analysis techniques using Python tools.

Text Books	
1	McKinney, W., Python for Data Analysis, 3rd Ed., O'Reilly Media (2022), ISBN: 978-1098104030
2	Zelle, J. M., Python Programming: An Introduction to Computer Science, 3rd Ed., Franklin, Beedle & Associates (2017), ISBN: 978-1590282755

Reference Books and Resources	
1	Sweigart, A., Automate the Boring Stuff with Python: Practical Programming for Total Beginners, 2nd Ed.,No Starch Press (2019), ISBN: 978-1593279929
2	VanderPlas, J., Python Data Science Handbook: Essential Tools for Working with Data, O'Reilly Media (2016), ISBN: 978-1491912058
3	Muller, A. C., and Guido, S., Introduction to Machine Learning with Python: A Guide for Data Scientists, O'Reilly Media (2016), ISBN: 978-1449369415
4	https://www.geeksforgeeks.org/data-analysis-with-python/
5	https://www.freecodecamp.org/learn/data-analysis-with-python
6	https://onlinecourses.nptel.ac.in/noc21_cs45/preview

Lab Programs / Practical	60 Hours
Part A: Lab Programs	
1	Setup and Environment: A. Installation and configuration of Anaconda (install file in USB pendrive), Python 3.12 and Jupyter Notebook 7.1.2 in the local environment Getting comfortable with Google Colab environment
2	Basic Programming and Syntax: A. Simple calculator B. Temperature converter (Celsius to Fahrenheit and vice versa) C. Program to find the largest of three numbers
3	Functions and Recursion: A. Factorial of a number using recursion B. Word count program
4	Data Structures: A. Program to perform various operations on lists (insert, delete, find, etc.) B. Program to merge two dictionaries
5	File Handling: Program to read a CSV file and perform basic analysis Getting awareness about local and Google drive folder and file access, renaming, copying, and moving files
6	NumPy Basics: A. Creating and manipulating NumPy arrays B. Program to perform matrix operations using NumPy
7	Pandas Basics: A. Creating a DataFrame from scratch and from a CSV/JSON file B. Program to clean data (handling missing values, duplicates)
8	Descriptive Statistics: Use mean(), median(), mode(), min(), max(), var(), std() from Pandas to measure central tendencies and dispersion of data
9	Data Distribution: Use describe(), quantile(), skew(), kurt(), value_counts() from Pandas for shape and distribution data
10	Categorical Encoding: Perform categorical encoding—OneHotEncoder, LabelEncoder from Python

Lab Programs / Practical (continued)	
11	Data Scaling: Scale the data using Python: StandardScaler(), Normalizer(), and MinMaxScaler() from sklearn.preprocessing
12	Outlier Detection: Visualize outliers using box plot from Matplotlib and Seaborn B. Identify outliers in datasets using Pandas DataFrame and Interquartile Range (IQR)
13	Advanced Pandas Operations: A. Advanced Pandas operations, GroupBy, merging, joining, concatenating DataFrames B. Time series analysis of stock prices (Optional case study)
14	Advanced Visualization with Seaborn: A. Using Seaborn to visualize data distributions B. Creating a pair plot for a DataFrame
15	Correlation: Visualize correlation matrix using heatmap from Seaborn to enhance readability
16	Identifying Trends and Relationships: A. Identify the trend of any variable using line chart from Matplotlib and Seaborn B. Visualize the relationship between two correlated columns using scatter plot
17	Category and Numeric Relationships: Visualize the relationship between categorical and a numerical column using bar plot
Part B: Mini Project	
1	Apply the learned concepts on real-world datasets to perform end-to-end data analysis, derive meaningful insights, and present their findings.

Semester - **II**			
Course: **DATA STRUCTURES USING C**			
Program	**B.Sc. (Hons) CSE**	Category	**Major (Core)**
Course Code	**CS1083**	CIE Marks	**70**
Credits (L:T:P)	**4 (2:0:4)**	SEE Marks	**30**
Hours	**30L + 0T + 60P = 90**	SEE Mode	**Practical**

Course Objectives: students will be able to	
1	Understand fundamental programming concepts essential for implementing data structures.
2	Apply user-defined data types to implement linear data structures like linked lists, stacks, and queues.
3	Implement binary trees and perform various traversal methods including inorder, preorder, and postorder.
4	Analyze and apply tree-based data structures for efficient data storage and retrieval.

Module - 1	**6 hours**
Functions and Recursion, Introduction to arrays, definition, Operations, Introduction to Data Structures: Definition, need and classification of data structures. Pointer concepts, pointer arithmetic, pointers with arrays and pointers to structures. Structures: Definition and declaration of structures, arrays of structures, nested structures and self-referential structures. Concept of Abstract Data Types (ADT) and separation of interface and implementation.	

Module - 2	**6 hours**
Linked Lists: Definition and representation of linked lists, insertion, deletion and searching operations on singly linked lists. Doubly Linked Lists: Structure and implementation using pointers, insertion, deletion and searching operations. Circular Linked Lists: Structure and operations of circular linked lists including insertion, deletion and traversal.	

Module - 3	**6 hours**
Stack ADT: Definition and fundamental operations such as push, pop, peek, and isEmpty, an array implementation of a stack. Stack applications. Queue ADT: Definition and fundamental operations such as enqueue, dequeue, front, and isEmpty. Queue Applications: Scheduling algorithms, array implementation of queues, and circular queue concept.	

Module - 4	6 hours
Trees: Definitions, terminology, properties of trees, and tree representation. Binary Trees: Binary tree definition, properties, and representation of binary trees and implementation using linked structures. Binary Tree Traversals: Inorder, preorder, and postorder traversal techniques. Binary Search Trees: Concept and operations such as insertion and searching. AVL Trees: Concept of balanced binary search trees, balance factor and basic AVL rotations for maintaining height balance in trees.Applications of trees.	

Module - 5	6 hours
Graph: Definition, properties, and types of graphs. Graph Representations: Adjacency matrix and adjacency list representations of graphs. Graph Traversal Techniques: Breadth First Search (BFS) and Depth First Search (DFS). Applications of graphs, advantages and disadvantages of graphs.	

Course Outcomes: After completing the course, the students will be able to	
1	Understand the basic programming concepts needed to implement data structures.
2	Apply user-defined data types to implement linked lists, stacks, and queues.
3	Implement binary trees and perform different traversal methods like inorder, preorder, and postorder.
4	Analyze graph data structures, perform basic operations, and evaluate traversal techniques like BFS and DFS.

Text Books	
1	Goodrich, M. T., Tamassia, R., & Goldwasser, M. H. (2019). Data Structures and Algorithms in C++ (2nd ed.). Wiley. ISBN-13: 978-1118771334
2	Thareja, R. (2018). Data Structures Using C (2nd ed.). Oxford University Press. ISBN-13: 978-0199488490
3	CareerMonk Publications. ISBN: 978-8193245279
4	Aho, A. V., Hopcroft, J. E., & Ullman, J. D. (1982). Data structures and algorithms (1st ed.). Addison-Wesley. ISBN: 978-0201000238

Reference Books and Resources	
1	Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). Introduction to Algorithms (4th ed.). MIT Press. ISBN: 978-0262046305
2	Sedgewick, R., & Wayne, K. (2011, with ongoing support). Algorithms (4th ed.). Addison-Wesley Professional. ISBN: 978-0321573513
3	Siddiqui, A. T., & Siddiqui, S. A. (2023). Data Structure Using C (1st ed.). CRC Press. ISBN: 978-1003453291

Reference Books and Resources (continued)	
4	https://www.geeksforgeeks.org/data-structures/
5	https://www.scaler.com/topics/c-data-structures/

Lab Programs / Practical	60 Hours
1	Write a program to find the largest and smallest elements in an array.
2	Write a program to find the factorial of a number using recursion. Write a program to find the nth Fibonacci number using recursion.
3	Write a program to implement basic operations on structures. Create a structure Student containing attributes such as id, name, and marks. Perform operations such as inserting student records and displaying the records.
4	Write a program to create a class/structure Node to represent a node in a singly linked list. Each node should have two attributes: data and next. Implement the following operations: 1. insert_at_beginning(data): Insert a new node with the given data at the beginning of the list. 2. .insert_at_end(data): Insert a new node with the given data at the end of the list. 3. delete_node(data): Delete the first node in the list that contains the given data 4. traverse (): Traverse the list and print the data of each node
5	Write a program to implement a stack using an array or a linked list and perform the following operations: 1. push(data): Insert an element into the stack 2. pop(): Remove the top element from the stack 3. peek(): Display the top element 4. Convert an infix expression to a postfix expression using a stack
6	Write a program to implement a queue using a linked list or an array and perform the following operations: 1. enqueue(data): Insert an element into the queue 2. dequeue(): Remove an element from the queue 3. front(): Display the front element 4. Simulate a simple service system using a queue (e.g., ticket counter or process scheduling).
7	Write a program to implement a doubly linked list and perform the following operations: 1. insert_at_beginning(data): Insert a new node with the given data at the beginning of the list. 2. insert_at_end(data): Insert a new node with the given data at the end of the list. 3. traverse_forward (): Traverse the list forward and print the data of each node.
8	Write a program to implement CircularLinkedList and perform the following operations: 1.insert_at_beginning(data): Insert a new node with the given data at the beginning of the list. 2. traverse(): Traverse the list and print the data of each node.

Lab Programs / Practical (continued)	
9	Write a program to create a structure TreeNode to represent a node in a binary tree. Each node should have three attributes: data, left, and right. Then, create a class/structure BinaryTree to represent the binary tree itself. Implement the following operations: 1. insert(data): Insert a new node into the binary tree using simple insertion. 2. in_order_traversal(): Perform an inorder traversal of the binary tree and print the data of each node. 3. pre_order_traversal(): Perform a preorder traversal of the binary tree and print the data of each node. 4. post_order_traversal(): Perform a postorder traversal of the binary tree and print the data of each node.
10	Write a program to implement a Binary Search Tree (BST) with the following operations: 1. Insert a node 2. Search a node 3. Perform inorder traversal of the BST.
11	Write a program to implement AVL Tree insertion. Display the tree using inorder traversal after each insertion and maintain the balance factor using appropriate rotations.
12	Create a structure Graph to represent a graph using an adjacency list. Implement the following operations: 1. add_edge(v, w): Add an edge between vertices v and w.
13	Implement the Depth-First Search(DFS).
14	Implement the Breadth-First Search(BFS).

Semester - **II**			
Course: **OPERATING SYSTEMS**			
Program	**B.Sc. (Hons) CSE**	Category	**Major (Core)**
Course Code	**CS1103**	CIE Marks	**70**
Credits (L:T:P)	**3 (2:0:2)**	SEE Marks	**30**
Hours	**30L + 0T + 30P = 60**	SEE Mode	**Theory**

Course Objectives: students will be able to	
1	Introduce the fundamental concepts, principles, and architecture of modern operating systems.
2	Explore the design and implementation of key operating system components, including process management, memory management, file systems, and input/output systems.
3	Develop problem-solving skills related to concurrency, synchronization, scheduling, and resource allocation.
4	Provide hands-on experience with operating system tools, system calls, and programming exercises for practical understanding of OS design and implementation.

Module - 1	**6 hours**
Introduction to Operating Systems and System Structures- Operating Systems, Types, Functions, Abstract view of OS, System Structures , System Calls, Virtual Machines.	

Module - 2	**6 hours**
Process Management- Process Concepts, Inter process communication, Threads ,Multithreading, Process Scheduling, First-Come, First-Served (FCFS), Shortest Job First, Round Robin (RR), Priority Scheduling.	

Module - 3	**6 hours**
Synchronization and Deadlocks- Synchronization, Semaphores, Monitors Hardware Synchronization, Deadlocks, Methods for Handling Deadlocks.	

Module - 4	**6 hours**
Memory Management- Memory Management Strategies, Contiguous and Non-Contiguous allocation, Virtual memory Management, Demand Paging, Page Placement and Replacement Policies.	

Module - 5	**6 hours**

Secondary Storage Management- File System- Basic concepts , File System design and Implementation, Linux File Systems, Mass Storage Structure, Disk Scheduling- FCFS, SSTF, Disk Management, and SCAN- I/O Systems, System Protection and Security.

Course Outcomes: After completing the course, the students will be able to	
1	Demonstrate the core functions of an operating system and its structural components to effectively manage system resources.
2	Analyze the various process management techniques and scheduling algorithms to ensure efficient execution and resource allocation.
3	Apply synchronization techniques and solutions to prevent or resolve deadlocks in concurrent systems.
4	Apply memory management strategies, virtual memory techniques, file system design principles, disk scheduling algorithms, and security mechanisms to improve performance and resource utilization.

Text Books	
1	Silberschatz, Galvin, Gagne, Operating System Concepts, John Wiley and Sons, 10th edition, 2023, ISBN: 935746056X
2	William Stallings, Operating Systems –Internals and Design Principles, 9/E, Pearson Publications, March 2017, ISBN: 0134670957
3	Richard Petersen, Linux: The Complete Reference, Sixth Edition, Tata McGraw-Hill Publishing Company Limited, New Delhi, 6th Edition 2008, ISBN: 007149247X

Reference Books and Resources	
1	Andrew S. Tanenbaum and Herbert Bos, Modern Operating Systems, 4th Edition, Pearson Publisher, 2015, ISBN-13: 978-0-13-359162-0
2	Michael Kerrisk, The Linux Programming Interface, Publisher: William Pollock, First Editions 2010, ISBN-13: 978-1593272203
3	NPTEL: https://nptel.ac.in/courses/106105214
4	MIT: https://ocw.mit.edu/courses/6-828-operating-system-engineering-fall-2012/
5	NPTEL: https://nptel.ac.in/courses/106106144

Lab Programs / Practical	**30 Hours**
Part A: Lab Programs	
1	Exploring Basic Linux Commands
2	Shell Programming
3	CPU Scheduling Algorithms Simulation
4	Process Creation and Management using fork system call

Lab Programs / Practical (continued)	
5	Thread Creation and Management using POSIX thread library
6	Program for IPC using pipe () function
7	Process synchronization using mutex locks and semaphore
8	Implement the Banker's Algorithm for deadlock avoidance
9	Simulations of Page replacement Algorithms
10	Simulate disk scheduling algorithms, such as FCFS, SSTF, and SCAN, and compare their performance
Part B:	
Design and implement a mini operating system simulator integrating process scheduling, memory management, and disk scheduling; analyse and compare system performance under different workloads using appropriate tools (AI tools permitted).	

Semester - **II**			
Course: **DATABASE MANAGEMENT SYSTEMS**			
Program	**B.Sc. (Hons) CSE**	Category	**Major (Core)**
Course Code	**CS2201**	CIE Marks	**70**
Credits (L:T:P)	**3 (1:1:2)**	SEE Marks	**30**
Hours	**15L + 15T + 30P = 60**	SEE Mode	**Practical**

Course Objectives: students will be able to	
1	Understand the Core Concepts of Database Management Systems
2	Apply Data Modelling Techniques to Design Efficient Databases
3	Develop Proficiency in SQL for Data Manipulation and Querying
4	Demonstrate practical skills in implementing and managing databases while analyzing their role in real-world applications.

Module - 1	**3 hours**
Definition and Importance of Databases, Database system architecture, Characteristics of DBMS vs. File Systems, Advantages & Disadvantages of DBMS, Types of Databases (Hierarchical, Network, Relational), Data Independence, Database Users and Administrators, Three-tier Architecture (External, Conceptual, Internal Levels) Data Models: Hierarchical, Network, Relational, Object-Oriented.	

Module - 2	**3 hours**
Entity Types, Entity Sets, Attributes, and Keys, Relationship Types, Introduction to ER Diagrams, Components of ER Diagrams, Drawing ER Diagrams, Specialization, Generalization & Aggregation, Mapping ER Diagrams to Relational Schema, Integrity constraints, entity integrity, referential integrity, Keys constraints, Domain constraints.	

Module - 3	**3 hours**
Database language and interfaces - DDL, DML, DCL and TCL, Characteristics of SQL, advantages of SQL. SQL data type and literals, Types of SQL commands, SQL operators and their procedure, Tables, views and indexes, Queries and subqueries, Aggregate functions. Insert, update and delete operations, Joins, Unions, Intersection, Minus, Triggers and Procedures in SQL.	

Module - 4	**3 hours**
Introduction to Normalization, Data Anomalies, Functional Dependencies, Types of Normal Forms, 1 Normal Form, 2 Normal Form, 3 Normal Form, BCNF(Boyce & Codd Normal Form.).	

Module - 5	3 hours
Introduction to NoSQL, Types of NoSQL Databases, Advantages & Disadvantages of NoSQL, Basics of MongoDB, Querying in NoSQL (MongoDB)	

Course Outcomes: After completing the course, the students will be able to	
1	Explore core database concepts, including architecture and transactions in DBMS
2	Design and implement database applications using ER diagrams to solve real-world problems.
3	Apply query processing techniques and normalization to optimize database performance and eliminate redundancy.
4	Apply the features of NoSQL databases for handling non-relational data.

Text Books	
1	A. Silberschatz, H. F. Korth, and S. Sudarshan, "Database System Concepts", 7th ed. New York: McGraw-Hill, 2019. ISBN 978-9390727506
2	R. Elmasri and S. B. Navathe, "Fundamentals of Database Systems", 7th ed. Boston: Pearson, 2016. ISBN 978-9332582705
3	P. J. Sadalage and M. Fowler, "NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence". Boston: Addison-Wesley Professional, 2012. ISBN 978-0321826626

Reference Books and Resources	
1	R. Ramakrishnan and J. Gehrke, "Database Management Systems". New York: McGraw-Hill Higher Education, 2002. ISBN 0072465638
2	V. Vaswani, "MySQL: The Complete Reference". New York: Osborne/McGraw-Hill, 2004. ISBN 0072224770
3	D. Sullivan, "NoSQL for Mere Mortals". Boston: Addison-Wesley, 2015. ISBN 0134023218
4	D. Howes, P. Membrey, and E. Plugge, "MongoDB Basics". New York: Apress, 2014. ISBN 148420896X

Tutorials	**15 Hours**
1	Design a University Management System using DDL commands (CREATE, ALTER, and DROP) in MySQL.
2	Perform DML (Data Manipulation Language) operations on a database designed for an online shopping system. This activity covers INSERT, UPDATE, DELETE, and SELECT statements.
3	Apply SQL functions (String, Numeric, Date, and Aggregate functions) using a simple School Database scenario.

Tutorials (continued)	
4	Implement Date and Conversion Functions in a School Database.
5	Implementing Arithmetic, Logical, and Comparison Operators in SQL.
6	Analyze product inventory and customer orders using special operators and set operations.
7	Implement the different types of joins for the given scenario.
8	Implement GROUP BY, HAVING, and ORDER BY in MySQL.
9	Implement a Stored Procedure in MySQL.
10	Implement Triggers in MySQL for INSERT, UPDATE, and DELETE.
11	MongoDB basics to store flexible student data using JSON documents without a fixed schema.
12	MongoDB queries and updates to manage nested fields and arrays in student JSON documents easily.

Lab Programs / Practical	**30 Hours**
Part A: Lab Programs	
1	Implement DDL Commands in SQL: Demonstrate the use of DDL commands with suitable examples, including CREATE TABLE, ALTER TABLE, and DROP TABLE.
2	Implement DML Commands in SQL: Implement DML commands with suitable examples, including INSERT, UPDATE, and DELETE.
3	Implement SQL Functions: Implement various types of functions in SQL with examples, including Number Functions, Aggregate Functions, and Character Functions.
4	Implement SQL Functions: Conversion Functions, and Date Functions.
5	Implement SQL Operators: Demonstrate the use of different types of operators in SQL with examples, including Arithmetic Operators, Logical Operators, Comparison Operators.
6	Implement SQL Operators: Demonstrate the use of different types of operators in SQL with examples Special Operators, and Set Operations(UNION)
7	Implement different types of Joins: Implement various types of joins in SQL, including Inner Join, Outer Join, and Natural Join, with suitable examples.
8	Implement Grouping and Ordering: Study and implement the GROUP BY and HAVING clauses, ORDER BY clause.
9	Implement Procedure.(Basic procedure, procedure for Insert, Update and Delete)
10	Implement Triggers(validation before and after insert, delete and update)

Lab Programs / Practical (continued)	
11	Implement CRUD operations using student profile data in a NoSQL document database.
12	Implement high-frequency student activity logging with time based query to demonstrate non-relational data management.
Part B: Project	
1	To develop a complete database-driven application integrating ER modelling, SQL, normalization, and NoSQL concepts for real-world problem-solving.
2	Students shall undertake a mini project individually or in teams (maximum 3 members) to develop a database-driven application in a chosen domain such as Student Management, Library Management, Online Shopping, Hospital Management, Inventory Management, or any other approved application. Project Components: - Problem Definition and Requirement Analysis: Identification of the problem, objectives, and system requirements. - Data Modelling: Design of ER diagram including entities, attributes, relationships, keys, and constraints. - Relational Schema Design and Normalization: Mapping of ER model to relational schema and application of normalization techniques up to BCNF. - SQL Implementation: Creation of a database using DDL commands and manipulation using DML commands. Implementation of queries using joins, subqueries, and aggregate functions. Use of views and indexes. - Advanced SQL Features: Implementation of stored procedures and triggers. - NoSQL Implementation: Design and implementation of a NoSQL database (MongoDB) to handle semi-structured data, including basic CRUD operations. - Testing and Validation: Verification of database operations and validation of constraints. - Documentation and Demonstration: Preparation of a structured project report and final demonstration of the application.

Semester - **II**			
Course: **ENGLISH COMMUNICATION**			
Program	**B.Sc. (Hons) CSE**	Category	**AEC**
Course Code	**CS1927**	CIE Marks	**70**
Credits (L:T:P)	**2 (0:1:2)**	SEE Marks	**30**
Hours	**0L + 15T + 30P = 45**	SEE Mode	**Practical**

Course Objectives: students will be able to	
1	Improve students' ability to read, understand, and analyse technical and academic texts.
2	Develop effective writing skills for academic, technical, and professional communication.
3	Enhance grammar, vocabulary, and sentence structure for clear and accurate language use.
4	Train students to organise and present ideas logically and confidently in formal written communication.

Module - 1	**6 hours**
Fundamentals of English Comprehension: Introduction to Reading Strategies, Skimming and scanning, identifying main ideas and supporting details, Understanding Context and Inference, Contextual clues, Making inferences, Analysing Text Structures, Recognizing patterns (cause-effect, compare-contrast), Identifying thesis statements and arguments.	

Module - 2	**6 hours**
Vocabulary Building and Usage: Expanding Technical Vocabulary, Common terms in computer science and IT, using glossaries and technical dictionaries, Synonyms, Antonyms, and Homonyms, Enhancing word choice, Avoiding common errors, Contextual Vocabulary Usage, Application in sentences and paragraphs, Precision in technical writing.	

Module - 3	**6 hours**
Writing Skills Development: Sentence Structure and Clarity, Constructing clear and concise sentences Avoiding run-on sentences and fragments, Paragraph Development, Topic sentences and supporting details Coherence and cohesion, Essay and Report Writing, Structure of essays and technical reports, Writing introductions, body paragraphs, and conclusions	

Module - 4	**6 hours**

Technical Writing and Documentation : Writing Technical Descriptions and Definitions, Describing processes and systems, Defining technical terms clearly, Creating User Manuals and Guides, writing instructional content, Using visuals and diagrams effectively, Research and Citation Skills, conducting research for technical writing, Proper citation and avoiding plagiarism.

Module - 5	**6 hours**
Effective Communication Skills: Presentation Skills, Structuring presentations, Effective use of visual aids, Professional Email and Correspondence, Email etiquette, Writing formal and informal emails, Collaborative Writing and Peer Review, Working in teams, Reviewing and editing peers' work.	

Course Outcomes: After completing the course, the students will be able to	
1	Apply reading strategies to comprehend and analyze technical and academic texts, identifying main ideas and inferring meaning.
2	Use technical vocabulary, synonyms, and antonyms accurately to enhance clarity and precision in writing.
3	Construct clear, coherent sentences and well-structured essays, reports, and technical documents.
4	Demonstrate effective communication skills by delivering presentations, writing emails, and collaborating on professional writing tasks.

Text Books	
1	Simon Greenall and Michael Swan, Effective Reading, Cambridge University Press, 1st Edition, ISBN: 978-0521317603.

Reference Books and Resources	
1	Online course from the Hindu Group, the STEP English course.

Semester - **II**			
Course: **ENTREPRENEURIAL MINDSET**			
Program	**B.Sc. (Hons) CSE**	Category	**SEC**
Course Code	**CS1940**	CIE Marks	**70**
Credits (L:T:P)	**2 (1:0:2)**	SEE Marks	**30**
Hours	**15L + 0T + 30P = 45**	SEE Mode	**Jury**

Course Objectives: students will be able to	
1	Analyze market opportunities to gain multiple business model insights
2	Building a business through POCD framework
3	Financing a business through basic understanding of key financial metrics
4	Sources of finance for a startup and calculating value of a startup through a Business model canvas & business plan

Module - 1	**3 hours**
Problem Identification: Techniques for spotting problems worth pursuing and understanding market demand. Market Sizing: Tools and methods for estimating market size and understanding customer willingness to pay. Opportunity Analysis: Case studies on identifying and capitalizing on opportunities within various industries. Value Proposition: Developing compelling value propositions that clearly address identified market needs.	

Module - 2	**3 hours**
Introduction to POCD: Understanding the People, Opportunity, framework. Customer and Consumer Insights: Differentiating between who the customer is and who the consumer is. Case Studies: Examining successful businesses that have effectively utilized the POCD framework	

Module - 3	**3 hours**
Introduction to POCD: Understanding the Context, and Deal framework. Strategic Development: Applying the POCD framework to build robust business strategies. Case Studies: Examining successful businesses that have effectively utilized the Context & Deal framework	

Module - 4	**3 hours**

Financial Literacy: Building comfort with basic financial terms and concepts crucial for entrepreneurs. Key Financial Metrics: Understanding metrics that are essential for evaluating business health. Financial Planning: Tools and techniques for financial forecasting and budgeting. Interactive Workshops: Practical exercises to develop and analyze financial sheets and models.

Module - 5	**3 hours**
Sources of Finance: Overview of different financing options available for startups. Business Model Canvas (BMC): Detailed exploration of the BMC as a tool for articulating business value. Startup Valuation: Methods to calculate the value of a startup, including practical case examples. Funding Strategy: Developing strategies to attract and secure necessary resources for a startup	

Course Outcomes: After completing the course, the students will be able to	
1	Analyze market opportunities to gain multiple business model insights
2	Build and evaluate a business venture using the POCD framework by integrating People, Opportunity, Context, and Deal elements.
3	Financing a business through basic understanding of key financial metrics
4	Sources of finance for a startup and calculating value of a startup through a Business model canvas & business plan

Text Books	
1	The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses by Eric Ries, Crown Business (2011), ISBN: 978-0307887894, 1st Ed.
2	Start with Why: How Great Leaders Inspire Everyone to Take Action by Simon Sinek, Portfolio ,2009, ISBN: 978-1591846444, 1st Ed.

Reference Books and Resources	
1	Freakonomics: A Rogue Economist Explores the Hidden Side of Everything by Steven D. Levitt and Stephen J. Dubner, William Morrow (2005), ISBN: 978-0060731335, 1st Ed.
2	Disciplined Entrepreneurship: 24 Steps to a Successful Startup by Bill Aulet, Wiley (2013), ISBN: 978-1118692288, 1st Ed.

Lab Programs / Practical	**30 Hours**
1	Entrepreneurial Trait Analysis and Mentor Mapping
2	Create an Individual BMC with AI Assistance
3	Numerical Evaluation on Various Case Studies
4	Create a Business Title, Proposition, Finance sheet
5	Create and Present an AI-Enhanced Product Pitch

Lab Programs / Practical (continued)	
6	Startup Idea Formulation and BMC Preparation

Semester - **II**			
Course: **INTRODUCTION TO VERSION CONTROL**			
Program	**B.Sc. (Hons) CSE**	Category	**VAC**
Course Code	**CS1105**	CIE Marks	**70**
Credits (L:T:P)	**2 (0:1:2)**	SEE Marks	**30**
Hours	**0L + 15T + 30P = 45**	SEE Mode	**Practical**

Course Objectives: students will be able to	
1	Develop computational thinking and effective problem-solving abilities
2	Gain hands-on experience with version control, AI tools, and modern technologies.
3	Encourage collaborative learning and peer-to-peer interaction.
4	Practice ethical, responsible, and reflective coding habits.

Module - 1	**3 hours**
Overview of Coding Platforms: LeetCode, Codeforces, HackerRank, CodeChef, GeeksforGeeks. Benefits of practicing on coding platforms Understanding problem-solving and competitive programming.	

Module - 2	**3 hours**
Version Control Basics: What is version control Advantages of version control systems (VCS), Types of VCS: Centralized vs. Distributed. Introduction to Git: What is Git, Installing Git on Windows/Linux/Mac, Git vs other version control tools.	

Module - 3	**3 hours**
Git Repositories: Understanding local and remote repositories, Hosting platforms: GitHub, GitLab, Bitbucket, Creating and managing repositories on GitHub. Collaborating with Git: Forking and cloning repositories, pull requests, and merge conflicts, Understanding. gitignore and branching.	

Module - 4	**3 hours**
Introduction to GitHub Co-Pilot: What is Co-Pilot, Role of AI in coding, and productivity using GitHub Co-Pilot.Setting up Co-Pilot in Visual Studio Code, writing efficient code with AI assistance, Ethical considerations, and limitations of AI-assisted coding.	

Module - 5	**3 hours**

Advanced Git Concepts: Stashing changes (git stash), Rebasing vs merging, Cherry-picking commits, Tagging versions in Git. Version Control Best Practices: Writing meaningful commit messages, structuring repositories effectively, and using branches for feature development.

Course Outcomes: After completing the course, the students will be able to	
1	Understand the purpose, structure, and benefits of centralized and distributed version control systems.
2	Develop problem-solving abilities using coding platforms and real-world programming tasks.
3	Demonstrate proficiency in Git and GitHub for collaborative version control and project management.
4	Apply advanced Git techniques and adopt best practices to enhance workflow efficiency and team productivity.

Text Books	
1	Chacon, S., & Straub, B. (2014). Pro Git (2nd ed.). Apress. ISBN: 978-1484200773

Reference Books and Resources	
1	Loeliger, J., & McCullough, M. (2012). Version control with Git (2nd ed.). O'Reilly Media. ISBN: 978-1449316389
2	Skoulikari, A. (2021). Learning Git: A hands-on and visual guide to the basics of Git. O'Reilly Media. ISBN: 978-1492092513

Tutorials	**15 Hours**
1	Introduction to competitive programming and coding platforms: LeetCode, HackerRank, Codeforces
2	Benefits of practicing on coding platforms; Problem-solving strategies
3	Introduction to Version Control Systems (VCS): Centralized vs. Distributed
4	Installing Git on Windows/Linux/Mac; Setting up user credentials
5	Basic Git operations: git init, git add, git commit, git status, git log
6	Understanding local vs. remote repositories; Git hosting platforms (GitHub, GitLab, Bitbucket)
7	Creating and linking repositories on GitHub; Pushing and pulling changes
8	Forking and cloning repositories; Understanding gitignore
9	Branching in Git: Creating, switching, and deleting branches
10	Merging branches; Identifying and resolving merge conflicts
11	Introduction to GitHub collaboration: Pull requests and code reviews

Tutorials (continued)	
12	Best practices in reviewing and managing pull requests
13	Advanced Git features: git stash, git rebase vs git merge, git tag, git cherry-pick
14	Introduction to GitHub Co-Pilot; Installing and configuring Co-Pilot in VS Code
15	Ethical use of AI in coding; Best practices: commit messages, branching strategies, clean history

Lab Programs / Practical	**30 Hours**
1	Create accounts on LeetCode, HackerRank, and Codeforces and solve two problems on each platform.
2	Focus on array and string problems on LeetCode and reflect on solution strategies.
3	Participate in a Codeforces contest and submit a solution analysis report.
4	Install Git and configure user credentials on your system.
5	Initialize a new Git repository and commit a sample file using basic Git commands.
6	View repository status and commit history using git status and git log.
7	Create and link a local repository with GitHub and push commits using git push.
8	Pull updates from a remote repository and manage repository synchronization.
9	Fork and clone a GitHub repository, create a pull request with changes.
10	Create feature branches, switch between them, and merge them using Git.
11	Simulate and resolve a merge conflict in a collaborative project setup.
12	Use git stash to save and apply temporary changes during development.
13	Rebase a feature branch onto the main branch using git rebase
14	Tag specific commits using git tag and cherry-pick selected commits into another branch.
15	Set up GitHub Co-Pilot in VS Code and generate code suggestions for simple functions.

RV University Vision and Mission

Vision

To be a world-class, tech-driven, global university for liberal education, empowering citizens of tomorrow.

Mission

- Strive for excellence in teaching, research, capacity building, and community engagement, benchmarking against global universities to lead across disciplines.
- Utilise digital and emerging technologies to enhance teaching-learning and research, accessible to all, while fostering a multidisciplinary, inclusive environment that meets evolving learner needs.
- Cultivate a diverse, global academic community through strong national and international collaborations that enrich learning, facilitate mobility, and drive institutional growth.
- Integrate theory with practical application to develop self-driven, empathetic problem-solvers equipped to create meaningful societal impact.

RV UNIVERSITY
Go, change the world

an initiative of RV EDUCATIONAL INSTITUTIONS

RV Vidyanikethan Post, 8th Mile, Mysuru Road, Bengaluru - 560 059
For general enquiry: enquiry@rvu.edu.in | Phone: +91 95136 73778

RV University _RVUniversity_ rv-university RV.University1 rv.university