AC: 02/26

Item No. : 02

The Bombay Salesian Society's
Don Bosco Institute of Technology
(An Autonomous Institute affiliated to University of Mumbai)

Bachelor of Engineering

in

Computer Engineering

(DB25-V1)
Effective from Academic Year 2025 – 2026

1. Preamble

Don Bosco Institute of Technology, Kurla, Mumbai, proudly celebrates the achievement of autonomous status—an academic milestone that reaffirms our steadfast commitment to excellence, holistic development, and student-centric learning. This autonomy empowers us to craft and implement a curriculum that is forward-looking, contextually relevant, and deeply rooted in our institutional values and the aspirations of our nation.

As an autonomous institution affiliated with the University of Mumbai, DBIT embraces the opportunity to restructure its academic framework in alignment with the University Grants Commission (UGC) guidelines and the National Education Policy (NEP) 2020. This curriculum framework outlines the undergraduate engineering programs for the EXTC, COMP, IT, and MECH branches. It reflects NEP's emphasis on multidisciplinary learning, flexibility, and outcome-based education, while staying true to the Don Bosco educational philosophy.

The curriculum adopts a top-down approach, beginning with the institutional Vision and Mission, which guides the definition of Program Educational Objectives (PEOs) and Program Outcomes (POs). These outcomes are used to shape Course Outcomes (COs) and the content and assessment methods of each course. This ensures that all academic efforts remain aligned with the broader goals of transforming learners into technically sound, ethically responsible and socially aware citizens. Importantly, this curriculum has been shaped through extensive consultations with stakeholders, including industry experts, academic peers, alumni, and students—to ensure that it remains aligned with contemporary industry requirements and societal expectations. Their inputs have been instrumental in designing a framework that bridges the gap between academic learning and practical applicability.

Key Objectives in developing syllabus are:

1. **Develop Strong Technical Foundations:** Equip students with robust knowledge and skills in core engineering domains to solve real-world problems through design, analysis, and innovation.

2. **Foster Research, Innovation, and Entrepreneurship:** Cultivate a spirit of inquiry, critical thinking, and entrepreneurial mindset to promote research-based problem-solving and startup culture.

3. **Enhance Interdisciplinary and Industry-Ready Competencies:** Integrate emerging technologies, multidisciplinary learning, and practical exposure to prepare students for dynamic industry requirements and lifelong learning.

4. **Promote Ethical, Sustainable, and Socially Responsible Engineering Practice:** Inculcate ethics, human values, and environmental consciousness to enable students to contribute meaningfully to society and sustainable development.

5. **Empower Communication, Leadership, and Teamwork Abilities:** Strengthen students' soft skills, collaboration, and leadership to perform effectively in diverse professional and global environments.

Academic design includes:

- A Choice-Based Credit System (CBCS) for flexibility
- A range of Minor and Honors options to encourage specialization and research
- Opportunities for field engagement, internships, and experiential learning
- Emphasis on skill enhancement and future workforce needs
- Integration of ethical reasoning, social awareness, and environmental consciousness

As an institution inspired by the values of Saint John Bosco, we strive to create a joyful and inclusive learning

environment that fosters creativity, curiosity, and compassion. Through this curriculum framework, we renew our pledge to produce graduates who are not only professionally competent but also committed to the greater good of society.

2. Vision and Mission

Vision:

DBIT will be known to have an innovative, enjoyable and holistic learning environment that transforms individuals into socially conscious citizens the Don Bosco way, and will lead in research and entrepreneurship in the area of sustainable technologies.

Mission:

1. To create future engineers who work with honesty and integrity and excel in the use of technology for the benefit of the underprivileged.

2. To train engineers to be innovative problem-solvers and entrepreneurs who engage in research and lifelong learning.

3. To provide a diverse and stimulating environment for staff and students to grow holistically.

3. Curriculum Design Philosophy

The curriculum is structured in alignment with the National Education Policy (NEP) 2020 and UGC guidelines. It follows a top-down approach wherein the institutional Vision and Mission guide the Program Educational Objectives (PEOs) and Program Outcomes (POs). These then shape the Course Outcomes (COs) and form the foundation for course structure, delivery, and assessments.

Key design principles include:

- Emphasis on Outcome-Based Education (OBE) with clear mappings of COs to Pos
- Integration of core technical knowledge with interdisciplinary electives
- Inclusion of vocational skills, internships, and community engagement
- Development of entrepreneurship and research aptitude through minor and honors pathways
- Encouragement of ethical, sustainable, and socially responsible engineering practices

This approach ensures that the curriculum remains academically rigorous, industry-relevant, and value-driven.

4. Credit Allocation Guidelines

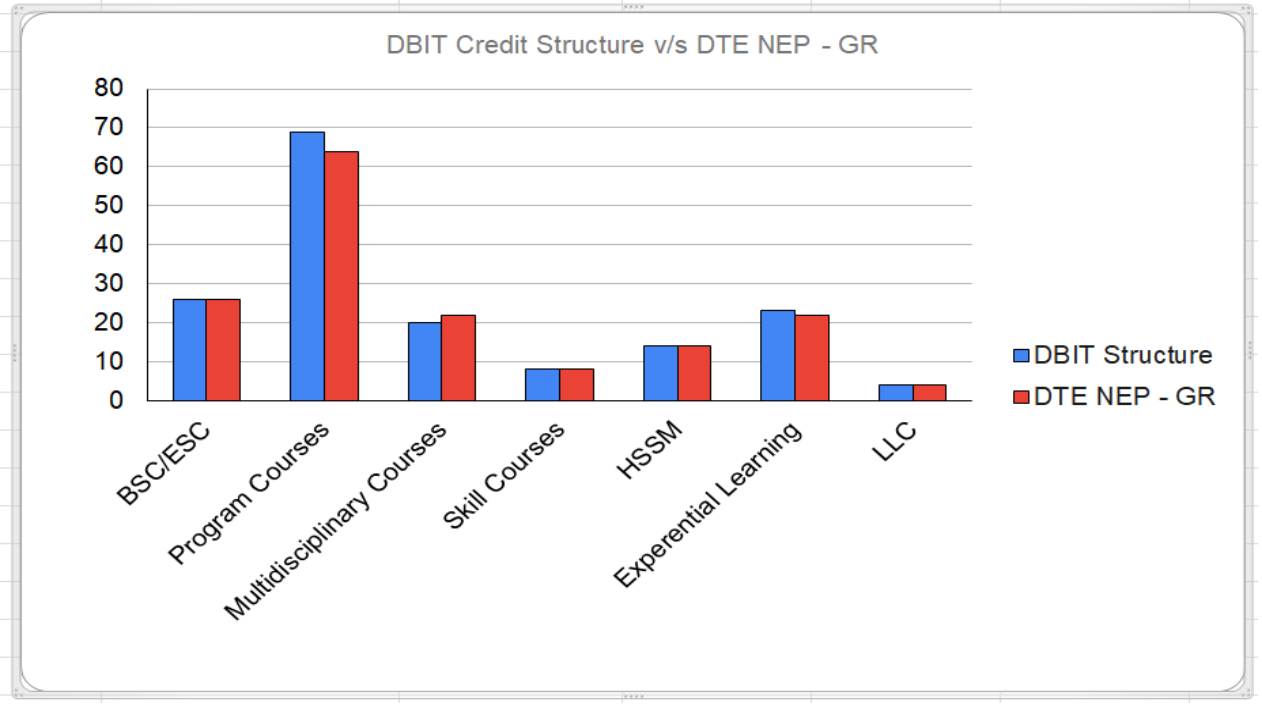
The curriculum is delivered through a structured credit system as follows:

Activity Type	Credit Definition
Theory Course	1 Credit = 15 Teaching Hours
Laboratory / Studio / Workshop	1 Credit = 30 Contact Hours
Internship / Field Work	1 Credit = 40 Hours or 2 weeks
Seminar / Group Discussions	1 Credit = 15 Participation Hours
Community Engagement / Field Project	1 Credit = 30 Contact + 15 Self Hours

DBIT Curriculum Credit Structure: (FE to BE)

Semester		I	II	III	IV	V	VI	VII	VIII	Total Credits	DTE Credits
Basic Science Course	BSC/ESC	9	6							15	14-18
Engineering Science Course		7	4							11	12 - 16
Programme Core Course (PCC)	Program Courses		3	16	14	6	6	6		51	44-56
Programme Elective Course (PEC)						3	3	6	6	18	20
Multidisciplinary Minor (MD M)	Multidisciplinary Courses				3	4	4	3		14	14
Open Elective (OE) Other than a						2	2	2		6	8
Vocational and Skill	Skill Courses	3	3	2						8	8
Ability Enhancement Course (AEC -01,	Humanities Social Science and Management (HSSM)		2			2				4	4
Entrepreneurship/Economics/					2		2			4	4
Indian Knowledge System (IKS)			2							2	2
Value Education Course (VEC)		2		2						4	4
Research Methodology	Experiential Learning Courses					2				2	4
Community. Engagement. Project (CEP)/				1	1	1				3	2
Project							3	3		6	4
Internship/ OJT									12	12	12
Co-curricular Courses (CC)	Liberal		1		1		1		1	4	4
Total Credits (Major)		**21**	**21**	**21**	**21**	**20**	**21**	**20**	**19**	**164**	**160 176**

Graphical Representation:



5. Degree Options and Exit Pathways

Students are offered flexible learning pathways through the following options:

Undergraduate Degree Options:

- B.E (Major) – Minimum 160 credits
- B.E with Double Minor/Honors (Multidisciplinary + Specialization) –178-180 credits
- B.E with Double Minor/Honors with Research – 178 -180 credits

Multiple Entry-Exit Options (Aligned with NEP 2020)

Exit Options	Credits Structure
Certificate after Year 1:	● 42 Credits + 08 credits (04 credit Exit course + 04 Summer internship).
Diploma after Year 2:	● 84 credits + 08 credits (04 credit Exit course + 04 Summer internship).
B. Vocational Degree after Year 3:	● 125 credits + + 08 credits (04 credit Exit course + 04 Summer internship).
Final Degree after Year 4:	● 160 credits

Credits earned are banked in the Academic Bank of Credits (ABC) for lifelong learning flexibility.

Abbreviations

Sr. No	Word / Term / Phrase	Abbreviation
1	The Bombay Salesian Society's Don Bosco Institute of Technology	DBIT
2	Computer Engineering Department	CE
3	Electronics and Telecommunication Department	EC
4	Information Technology Department	IT
5	Mechanical Engineering Department	ME
6	Basic Science Course	BSC
7	Engineering Science Course	ESC
8	Basic Science / Engineering Science Course	BSESC
9	Program Core Course	PCC
10	Program Elective Course	PEC
11	Vocational Skill Enhancement Course	VSEC
12	Multidisciplinary Minore	MDM
13	Open Elective	OE
14	Ability Enhancement Course	AEC
15	Entrepreneurship / Economics / Management Course	EEM
16	Indian Knowledge System	IKS
17	Value Education System	VEC
18	Research Methodology	RM
19	Community Engagement Project / Field Project / Mini – Project	CEP
20	Project	PRJ
21	Internship / OJT	OJT
22	Liberal Learning Course / Co-curricular Course	LLC
23	Modified Bloom's Taxonomy	BT
24	Course Outcome	CO
25	Outcome Based Education	OBE

Preface

Dear Students and Teachers,

It gives us immense pride and joy, as members of the Board of Studies for Computer Engineering at Don Bosco Institute of Technology, Kurla, Mumbai, to present the Computer Engineering syllabus, implemented under our Autonomous status, effective from the Academic Year 2025–26, aligned with the DB25-V1 Scheme.

This curriculum represents a pivotal academic advancement under the autonomy granted to DBIT—enabling us to tailor our academic offerings with greater flexibility, innovation, and responsiveness to emerging global trends. It has been thoughtfully structured in accordance with the University Grants Commission (UGC) guidelines and the vision of the National Education Policy (NEP) 2020, ensuring that students receive a future-ready, multidisciplinary, and outcome-based education.

Computer Engineering remains a highly dynamic and in-demand discipline. This revised syllabus is rooted in the belief that graduates must be well-equipped to tackle challenges in both foundational and cutting-edge domains. In line with this, the curriculum focuses on key verticals that define the future of technology, namely:

- Cybersecurity
- Data Science and Machine Learning
- Business Intelligence
- Immersive and Intelligent Interfaces

These verticals have been carefully integrated into the syllabus through specialized electives, skill-based labs, and project-based learning, providing students with the opportunity to build deep domain expertise and practical proficiency in areas that are transforming industries worldwide.

Key Highlights of the Revised Curriculum:

1. Credit Rationalization: The overall credits have been optimized to 170, giving students a balanced academic load and more time to engage in research, innovation, and extracurricular pursuits.
2. Specialization through Electives: A curated selection of department-level optional courses enables students to pursue focused learning in the above key verticals, nurturing domain-specific capabilities alongside a strong generalist foundation.
3. Skill-Based Labs & Mini Projects: Emphasis on hands-on experimentation, real-world problem solving, and mini projects allows students to build strong portfolios and demonstrate technical creativity and application.
4. MOOCs Integration: Students are encouraged to leverage platforms like SWAYAM for self-paced learning in specialized or emerging topics, promoting lifelong and independent learning.
5. Industry and Outcome-Oriented Design: The curriculum is shaped through extensive consultations with industry experts, alumni, faculty, and students, ensuring its relevance to current professional landscapes and evolving technologies.
6. Ethical and Holistic Development: True to the legacy of Saint John Bosco, this academic framework emphasizes the development of not only technical skills, but also ethical responsibility, social awareness, and environmental stewardship.

Through this forward-looking curriculum, the Department of Computer Engineering continues its mission to cultivate technologically skilled, socially responsible, and globally competent professionals. We are confident that this framework will prepare our students to lead and innovate in a digital world increasingly shaped by intelligent systems and human-centered technology.

We sincerely thank all contributors' faculty, students, industry collaborators, and academic partners who played a vital role in shaping this curriculum. Together, we strive to foster an inclusive, inspiring, and intellectually vibrant learning environment that reflects the core values of our institution.

Warm regards, Board of Studies
Department of Computer Engineering
Don Bosco Institute of Technology, Kurla, Mumbai

UG Second Year Engineering Program

Semester wise Curriculum Structure

Semester- IV

Course Code	Course Vertical	Course Name	Teaching Scheme (Contact Hours)			Credits Assigned			
			T	P	T	T	P	T	Total
25CE4PCC01	PCC	Theoretical Computer Science	3	-	1	3	-	1	4
25CE4PCC02	PCC	Advance data structures and algorithms	3	2	-	3	1	-	4
25CE4PCC03	PCC	Computer Network	3	2		3	1		4
25CE4PCC04	PCC	Operating System		2*+2	-	-	2	-	2
25ILXX4MDM01	MDM	MDM 01@	2	2	-	2	1	-	3
25CE4CEP01	CEP	Community Engagement Project	-	2	-	-	1	-	1
25IL4EEM01	EEM	Entrepreneurship Essentials	2	-	-	2	-	-	2
25IL4LLCXX	LLC	Liberal Learning Course	-	2$	-	-	1	-	1
		Total	**13**	**14**	**1**	**13**	**7**	**1**	**21**

* Two hours of practical class to be conducted as demo/practical/discussion

@ Students must select one Multidisciplinary Minor (MDM) course provided by a different engineering department than their own, from the range offered by the Institute.

$ Two hours of activity to be conducted for full class relative to respective LLC.

Examination Scheme & Assessment Structure

Examination Marking Scheme: Semester IV

Course Code	Course Vertical	Course Name	Examination Marks						
			CA	MSE	ESE	TW	OR	PR	Total
25CE4PCC01	PCC	Theoretical Computer Science	20	30	50	25	-	-	125
25CE4PCC02	PCC	Advance data structures and algorithms	20	30	50	25	-	25	150
25CE4PCC03	PCC	Computer Network	20	30	50	25	25	-	150
25CE4PCC04	PCC	Operating System	50	-	-	25	-	25	100
25ILXX4MDM01	MDM	MDM 01@	20	30	50	25	-	-	125
25CE4CEP01	CEP	Community Engagement Project	-	-	-	25	25	-	50
25IL4EEM01	EEM	Entrepreneurship Essentials	50	-	-	-	-	-	50
25IL4LLCXX	LLC	Liberal Learning Course	50	-	-	-	-	-	50
		Total Marks	**230**	**120**	**200**	**150**	**50**	**50**	**800**

Multidisciplinary Minor (MDM) Courses:

Semester IV to Semester VII (03 Credits)

Course offered by departments	Recipient Department	Name of the Course	Sem IV	SEM V	SEM VI	Sem VII
MDM	**Name**	Industrial IOT Automation	MP MC (3)	Embedded Systems (4)	Wireless Technology (4)	IT Capstone Project (3)
	IT	Data Mgmt	Data Structure & Algorithms	DBMS	BDA	Capstone Project (3)
	Comp	IP	Web Development	Android App Development	Cloud Computing	Capstone Project (3)
	MECH	Industrial Engg & PM	Logistics & SCm	Quality Engg	Planning & Resource Mgmt.	Project Mgmt
	EXTC	Industrial IOT	Embedded System	Networking and Communication Protocols	Industrial IOT & Automation	Capstone Project (3)

LIST OF LIBERAL LEARNING COURSES (LLC)

Track - Performing Arts	Course Code	Course Name
1	**25ILELLC01**	Rhythm & Motion: A Journey Through Dance
2	**25ILELLC02**	Introduction to Dramatics: Exploring Theatre Arts
3	**25ILELLC03**	Swaranjali: Introduction to Vocal Music
4	**25ILELLC04**	Strings & Strokes: An Introduction to Musical Instruments
Track - Visual Arts		
5	**25ILELLC05**	Traditional Rangolis of India
6	**25ILELLC09**	Foundations of Photography
7	**25ILELLC12**	Tradition & Craft: Hands-On Indian Art
Track - Sports		
8	**25ILELLC18**	Sports and Fitness

Assessment Methodology

Type of Course	Assessment Tool	Marks Distribution
Theory	**CA-20**	Certification: NPTEL (20 Marks) (Approved by instructor) OR Any two Pedagogies (10 marks each) MCQ /Class Test Case study/Assignment GATE based Assignment Certification: Udemy/Coursera (Approved by instructor) Open Book Test Working model / simulation of a course-based concept.
Theory (VEC)	**CA-50**	Active Participation = 5 marks MCQ /Class Test= 10 marks Instructor Assessment of the Activity carried out by student = 25 marks Assignment = 10 marks
Workshop	**CA-50**	Active Participation = 5 marks Trade 1# = 15 marks Trade 2# = 15 marks Trade 3# = 15 marks # Based on the performance and satisfactory completion of trade wise tasks.
Liberal Learning Courses (LLC)	**CA-50**	Active Participation = 5 marks Mentor Assessment of the Activity carried out by student = 25 marks Cultural Event (Euphoria) Participation = 10 marks Technical Event (Colosseum) Participation = 10 marks
Theory	**MSE**	Question Paper Pattern is as follows: All Questions are compulsory. Q1 A or B - 10 marks Q2 A or B - 10 marks Q3 A or B - 10 marks For each question, A and B should be based on the same CO. MSE should be based on 50% syllabus. Time: 90 minutes (1 hour 30 minutes) Total Marks: 30

Theory	**ESE**	Question Paper Pattern is as follows: All Questions are compulsory. Q1 A or B - 10 marks Q2 A or B - 10 marks Q3 A or B - 10 marks Q4 A or B - 10 marks Q5 A or B - 10 marks For each question, A and B should be based on the same CO. ESE should be based on 30% syllabus of MSE and 70% syllabus after MSE. Time: 120 minutes (2 hours) Total Marks: 50
Course - Laboratory	**TW- 25**	Active Participation (Lab) = 5 marks Laboratory Report = 10 marks Laboratory performance = 10 marks Based on the performance and satisfactory completion of assigned laboratory work
Mini-project	**TW-25**	Active Participation = 5 marks Project Report = 10 marks Progress presentation (Minimum 02) & demonstration = 10 marks
Tutorial	**TW-25**	Active Participation = 5 marks Tutorial Submission = 20 marks Tutorial based on the entire syllabus
Laboratory	**OR-25**	Oral examination will be based on the entire syllabus.
Laboratory	**PR-25**	Practical examination will be based on the experiments performed by the students during laboratory sessions.

Guideline for All Assessment*

Course Outcomes	Percentage
CO-1, CO-2	20-30
CO-3, CO-4	40-50
CO-5, CO-6	20-30

***Note:** Total Weightage of All CO's should be 100%

Theoretical Computer Science

Course Code	Course Name	Teaching Scheme (Hrs. / Week)			Credits Assigned			
25CE4PCC01	Theoretical Computer science	L	P	T	L	P	T	Total
		3	-	1	3	-	1	4
		Examination Scheme						
			CA	MSE	ESE	TW	OR	PR	Total
		Theory	20	30	50	-	-	-	100
		Tut	-	-	-	25	-	-	25

Pre-requisite: 25DBCE3PCC01: Discrete Structures and Graph Theory

Course Objectives: This course introduces the fundamental concepts of computation through the study of formal languages and automata. It covers basic notions such as strings, alphabets, and languages; deterministic and nondeterministic finite automata and their equivalence; regular expressions, grammars, closure properties, and decision techniques for regular languages. The course further explores context-free grammars, their derivations, ambiguity, and normal forms, along with pushdown automata for recognizing context-free languages. Finally, it introduces Turing machines and the core ideas of decidability and undecidability, providing insight into the capabilities and limitations of computation.

Module 1: Foundations of Computation: This module introduces the basic building blocks of formal language theory, including alphabets, strings, and languages, along with operations on languages. It develops the concept of finite state machines and formal definitions of finite automata, including their structure and functioning. Students learn how strings are processed and accepted by automata, and how languages are recognized. The module lays the groundwork for understanding computational models and includes problem-solving exercises where real-world scenarios are modeled using finite automata.

Module 2: Finite Automata: This module focuses on deterministic and non-deterministic finite automata (DFA and NFA), highlighting their structure, differences, and equivalence in computational power. It introduces methods for converting NFA to DFA and emphasizes the concept of state minimization to obtain efficient automata. The module also covers Mealy and Moore machines, their operational differences, and equivalence, along with applications in real-world systems such as traffic controllers and vending machines. This module strengthens students' ability to design and optimize finite-state systems.

Module 3: Regular Languages: This module explores the theory of regular languages through regular grammars, closure properties, and Arden's theorem. It provides techniques to convert between regular expressions and finite automata, and vice versa. The module also introduces the Pumping Lemma to prove whether a language is regular or not, along with decision properties of regular languages. Students gain both theoretical understanding and practical insight into applications of regular languages in areas such as text processing and pattern matching.

Module 4: Context-Free Grammars: This module extends the study of formal languages to context-free grammars (CFGs), including production systems, derivations, and parse trees. It discusses ambiguity in grammars and techniques to convert ambiguous grammars into unambiguous ones. The module also covers normal forms such as Chomsky Normal Form (CNF) and Greibach Normal Form (GNF), and introduces the Pumping Lemma for context-free languages. Students learn the equivalence between grammars and automata, enabling them to model more complex language structures.

Module 5: Pushdown Automata: This module introduces pushdown automata (PDA) as computational models capable of recognizing context-free languages. It explains the structure and functioning of PDAs, including stack operations and language acceptance methods (by final state and empty stack). The equivalence between PDAs and context-free grammars is also covered. Additionally, students explore instantaneous descriptions (IDs) to trace PDA computations, enhancing their understanding of how stack-based memory extends computational power.

Module 6: Turing Machines and Undecidability: This module presents Turing machines as a general model of computation, capable of solving a wide range of problems. It covers recursive and recursively enumerable languages, techniques for designing Turing machines, and language recognition. The module also introduces advanced topics such as universal Turing machines, variants of Turing machines, and the Halting Problem. It concludes with discussions on decidability, undecidability, Post's Correspondence Problem, and an introduction to NP-completeness, helping students understand the theoretical limits of computation.

Overall, the course equips computer science students with theoretical foundations and analytical skills to model, design, and evaluate computational systems and their limitations.

Course Outcomes: On successful completion, of course, learner/student will be able to:

1	Recall definitions of formal languages, automata, and expressions. (remembering)
2	Explain equivalences and conversions between finite automata models. (understanding)
3	Construct regular expressions, finite automata, and regular grammars and apply Pumping Lemma and closure properties. (applying)
4	Analyze context-free grammars, derivations, ambiguity. (analyzing)
5	Construct and evaluate pushdown automata and use instantaneous descriptions to determine string acceptance. (evaluating)
6	Design Turing machines for language recognition/computation and explain undecidability. (creating)

Module	Detailed Contents	Hours
1	**Foundations of Computation**	**3**
	After completing this module, students will be able to: ● Recall and define alphabets, strings, languages, and operations on languages. ● Explain the structure and components of a finite state machine and finite automaton model.	

	● Illustrate how strings are accepted by finite automata.		
	1.1	Strings, Alphabet, Language, Operations, Finite state machine, definitions, finite automaton model, acceptance of strings, and languages.	
		Self Study: Solve problems as a finite automaton.	
2	**Finite Automata**		**10**
	After completing this module, students will be able to: ● Explain the working and differences between DFA and NFA. ● Demonstrate conversion between NFA and DFA and justify their equivalence. ● Apply techniques to minimize DFA and construct Mealy and Moore machines.		
	2.1	Finite Automata: DFA and NFA, and Equivalence of DFA & NFA,	
	2.2	Minimization of DFA, Mealy and Moore machines and equivalence.	
		Self Learning: Explore how Mealy and Moore machines are used in real-world digital systems like traffic lights, elevators, and vending machines.	
3	**Regular Languages**		**9**
	After completing this module, students will be able to: ● Construct regular expressions, finite automata, and regular grammars for given languages. ● Apply closure properties and Arden's theorem to solve problems. ● Use the Pumping Lemma and decision properties to analyze regular languages.		
	3.1	Regular Grammar, Closure Properties, Arden Theorem, Regular expression to FA, DFA to Regular expression.	
	3.2	Pumping Lemma, Decision Properties of Regular Languages.	
		Self Learning: Practical Applications of Regular Languages,	
4	**Context-Free Grammars**		**9**
	After completing this module, students will be able to: ● Analyze context-free grammars, derivations, and parse trees. ● Identify and resolve ambiguity in CFGs. ● Apply normal forms and Pumping Lemma for CFLs.		
	4.1	Production systems, Right linear grammar and Left Linear Grammar, Equivalence of grammar and finite automata.	
	4.2	Context free grammars, Derivations, Ambiguity, Ambiguous to Unambiguous CFG.	

	4.3	Normal forms, Pumping Lemma for CFLs.	
		Self Learning: Subfamilies of CFL	
5	**Pushdown Automata**		**6**
	After completing this module, students will be able to: ● Construct pushdown automata for given context-free languages. ● Evaluate string acceptance using instantaneous descriptions (IDs). ● Explain the equivalence between PDA and CFG.		
	5.1	Pushdown Automata (PDA), Language Acceptance, Acceptance by final state and empty stack	
	5.2	Equivalence between push- down automata and context-free grammars.	
		Self Learning: Instantaneous descriptions (IDs) to follow PDA computations.	
6	**Turing Machines and Undecidability**		**8**
	After completing this module, students will be able to: ● Design Turing machines for language recognition and computation. ● Explain recursive and recursively enumerable languages. ● Analyze decidability, undecidability, and the Halting Problem.		
	6.1	Recursive & Recursively Enumerable Languages, Turing Machines: Basic Model,Techniques for Turing machine construction, Language Recognition.	
	6.2	Variants of TM, Universal Turing machine	
	6.3	Halting Problem, Decidable and undecidable problems, Post's correspondence problem, Introduction to Theory of NP-completeness.	
		Self Learning: Non- deterministic Turing machine, Rice's theorem	
		TOTAL	**45**

Textbooks:	
1	J. E. Hopcroft, R. Motwani, J. D. Ullman, “Introduction to Automata Theory, Languages, and Computation”, Third Edition, 2006, Pearson Education.
2	K. L. P. Mishra, N. Chandrasekaran, “Theory of Computation”, Third Edition, 2008, PHI Learning Private Limited.
References:	
1	J. L. Mott, A. Kandel, T. P. Baker, “Discrete Mathematics for Computer Scientists and Mathematicians”, Second Edition 1986, Prentice Hall of India.

2	H. R. Lewis, C. H. Papadimitriou, "Elements of the Theory of Computation", Second Edition, 1998, Prentice Hall of India.
3	D. M. Dhamdhere, "Automata Theory", Second Edition, 2008, Tata McGraw Hill Publishing.

Suggested list of Tutorial Questions:	
S. No.	**Tutorial questions**
1	Module 1: Foundations of Computation Objectives: • **Reinforce understanding of basic concepts like alphabets, strings, languages, and regular expressions through examples.** • **Develop ability to relate automata concepts to real-world applications using state diagrams.** Outcomes: • Explain and illustrate the concepts of alphabets, strings, languages, and regular expressions. • Interpret the working of a finite-state machine through real-world applications such as vending machines or traffic light controllers. • Represent system behavior using state diagrams. 1. Illustrate how a finite-state machine is used in a vending machine or traffic light controller. Draw a state diagram for your chosen application. 2. Define and illustrate the basic concepts of alphabets, strings, and regular expressions.
2	Module 2: Finite Automata Objectives: • Provide practice in designing DFA and NFA for given problems. • Introduce practical usage of simulation tools like JFLAP. Outcomes: • Design DFA and NFA for given regular languages. • Apply techniques for NFA to DFA conversion and DFA minimization. • Construct and analyze Mealy and Moore machines. • Use simulation tools like JFLAP to verify automata behavior and language acceptance. 3. Design a Deterministic Finite Automaton (DFA) and Non-deterministic Finite Automaton (NFA) for a given regular language. 4. Practice questions on NFA to DFA, DFA minimization, Moore and Mealy machines. 5. Use JFLAP to simulate DFA, NFA, and their conversions to verify language

	acceptance.
3	Module 3: Properties of Regular Languages Objectives: • Develop problem-solving skills using Arden's Theorem and closure properties. • Encourage exploration of decision properties of regular languages. Outcomes: • Apply Arden's Theorem to derive regular expressions from finite automata. • Use closure properties to construct new regular languages. • Apply the Pumping Lemma to test whether a language is regular. • Examine decision properties of regular languages. 6. Apply Arden's Theorem to derive a regular expression from a given DFA. [Explore: Practice closure properties like union, concatenation, and Kleene star.] 7. Use the Pumping Lemma to prove that a language is not regular. [Explore: Investigate decision properties of regular languages.]
4	Module 4: Context-Free Grammars Objectives: • Enable practice in constructing CFGs for given languages. • Develop skills to convert CFGs into standard normal forms. Outcomes: • Construct context-free grammars (CFGs) for given languages. • Perform transformations of CFGs into Chomsky Normal Form (CNF) and Greibach Normal Form (GNF). 8. Construct a context-free grammar (CFG) for a given language and convert it to Chomsky Normal Form (CNF) and Greibach Normal Form.
5	Module 5: Pushdown Automata Objectives: • Strengthen understanding of CFG–PDA relationships through practice. • Encourage simulation of PDA to observe stack operations. Outcomes: • Design pushdown automata (PDA) for context-free languages. • Demonstrate the conversion between CFG and PDA. • Use tools like JFLAP to simulate PDA and analyze stack-based computations. 9. Design a PDA to accept a context-free language and show conversion between CFG and PDA.

	10. Simulate Pushdown Automata using JFLAP to observe stack operations during language acceptance.
6	Module 6: Turing Machines and Undecidability Objectives: • Develop skills in designing Turing machines for simple problems. • Reinforce understanding of undecidability concepts through examples. Outcomes: • Design Turing machines for language recognition and computation. • Explain concepts of undecidability using problems like the Halting Problem. • Simulate Turing machine execution to understand tape operations and state transitions. 11. Design a Turing Machine to recognize a specific language and demonstrate language acceptance. 12. Explain undecidability using examples and describe the Halting Problem. 13. Simulate Turing Machine execution using JFLAP to understand tape movement and state transitions.
Implementations of automata, grammars, and Turing machines using suitable programming languages to reinforce theoretical concepts.	

Useful Links

1	Introduction to Automata, Languages and Computation https://nptel.ac.in/courses/106105196
2	Formal Language and Automata Theory https://nptel.ac.in/courses/111103016
3	Automata Jeff Ullman Stanford https://www.youtube.com/playlist?list=PLEAYkSg4uSQ33jY4raAXvUT7Bm_S_EGu0

Advance Data Structures and Algorithms

Course Code	Course Name	Teaching Scheme (Hrs./ Week)			Credits Assigned			
		L	P	T	L	P	T	Total
25CE4PCC02	Advance Data Structures and Algorithms	3	2	-	3	1	-	4
		Examination Scheme						
			CA	MSE	ESE	TW	OR	PR	Total
		Theory	20	30	50	-	-	-	100
		Lab	-	-	-	25	-	25	50

Pre-Requisite Courses:	**Pre-requisite:** **1. 25FE1VSEC02: Problem Solving using C Programming** **2. 25FE2PCC01: Data structures** **3. 25CE3PCC02: Analysis of Algorithms**

Course and Module Overview:

The Advanced Data Structures and Algorithms (ADSA) course extends foundational knowledge from Data Structures and Analysis of Algorithms to address efficiency, scalability, and real-world applicability. The course emphasizes amortized analysis, self-balancing trees, multiway storage structures, advanced heaps, graph optimization structures and modern algorithmic paradigms such as approximation and randomized algorithms.

It integrates theoretical rigor with implementation-driven learning through practical experiments and a mini-project. The focus is on understanding trade-offs between different data structures, analyzing performance beyond worst-case scenarios, and designing optimized solutions for large-scale and system-level problems. This course prepares students for industry-level problem solving, competitive programming, and system design contexts.

Module 1: Advanced Cost & Amortized Analysis

This module introduces the limitations of classical worst-case and average-case analysis and motivates the need for amortized analysis in real-world systems. Students learn aggregate, accounting, and potential methods to analyze operations over sequences. The module connects theory with practice through examples like dynamic arrays and stack operations, emphasizing long-term cost behavior.

Module 2: Advanced Balanced Search Trees

This module focuses on self-balancing binary search trees such as AVL, Red-Black, and Splay Trees. It explores their structural properties, balancing mechanisms, and performance guarantees. Students develop a comparative understanding of how different balancing strategies impact search efficiency, update cost, and real-world usage scenarios.

Module 3: Multiway & External Memory Data Structures

This module addresses the need for efficient data structures in secondary storage systems. It introduces B-Trees and B+ Trees, emphasizing their role in minimizing disk access. Students learn how multiway branching improves performance for large datasets and understand applications in databases and file systems, including indexing and range queries.

Module 4: Advanced Heap & Priority Queue Structures
This module explores advanced heap structures including Binary, Binomial, and Fibonacci Heaps. It focuses on their structural design, operational efficiency, and suitability for different applications. Special attention is given to amortized performance improvements and the importance of efficient merge operations in complex systems like network optimization.

Module 5: Advanced Graph Data Structures
This module introduces efficient data structures for solving graph-related problems, particularly Disjoint Set Union (Union-Find) and DAGs. Students learn optimization techniques like path compression and union by rank, and apply them to problems such as dynamic connectivity. The module also highlights applications in scheduling, clustering, and network analysis.

Module 6: Approximation Algorithms & Computational Complexity
This module covers algorithmic strategies for problems where exact solutions are computationally expensive. It introduces approximation algorithms (e.g., Vertex Cover, Set Cover) and evaluates their performance guarantees. Additionally, it explores randomized algorithms (Monte Carlo and Las Vegas), emphasizing probabilistic decision-making and trade-offs between accuracy and efficiency.

	After successful completion of the course, the students will be able to	
Course Outcomes	CO1	Recall and define concepts related to amortized analysis, advanced data structures, and approximation algorithms. *(Remembering*
	CO2	Explain the working principles, structural properties, and performance characteristics of advanced trees, heaps, and graph data structures. *(Understanding)*
	CO3	Apply appropriate advanced data structures and algorithmic techniques to solve complex computational problems efficiently. *(Applying)*
	CO4	Analyze the time and space efficiency of algorithms using amortized, randomized, and approximation-based analysis techniques. *(Analyzing)*
	CO5	Evaluate alternative data structures and algorithmic strategies based on efficiency, scalability, and application-specific constraints. *(Evaluating)*
	CO6	Design optimized and scalable solutions using advanced data structures and algorithmic paradigms for real-world problem scenarios. *(Creating)*

Syllabus:

Module No.	**Unit No.**	**Topics**	**Hours**
1		**Advanced Cost & Amortized Analysis**	**06**
		After completing this module, students will be able to: • Apply amortized analysis techniques (aggregate, accounting, potential) to evaluate algorithm efficiency beyond worst-case scenarios. *(CO3, CO4)* • Analyze and justify performance of dynamic data structures such as arrays, stacks, and queues using amortized bounds. *(CO4)* • Interpret long-term cost behavior of operations in practical systems and libraries. *(CO2, CO5)* • Select appropriate analysis techniques for different computational scenarios. *(CO5)*	

	1.1	• Limitations of worst-case and average-case analysis. • Motivation for amortized analysis in real systems.	
	1.2	• Amortized Analysis: Aggregate Method, Accounting Method and Potential Method. • Amortized analysis of: Dynamic arrays and stack operations. • Practical interpretation of amortized cost in libraries.	
	Self-Learning Topics: Analysis of Basic Data structure: Basic operations of Stack, Queue & Link List		
2	**Advanced Balanced Search Trees**		**08**
	After completing this module, students will be able to: • Explain and implement AVL, Red-Black, and Splay Trees with correct balancing operations. *(CO2, CO3)* • Analyze trade-offs between different balanced trees based on height, rotations, and performance. *(CO4, CO5)* • Evaluate suitability of tree structures for various applications such as ordered data storage and search systems. *(CO5)* • Design efficient solutions using self-balancing trees for dynamic datasets. *(CO6)*		
	2.1	AVL Trees: Height balancing property, Single and double rotations, Insertion and deletion with analysis	
	2.2	Red–Black Trees: Properties and invariants, insert and delete operations, comparison with AVL trees.	
	2.3	Splay Trees: Self-adjusting trees, access patterns and amortized performance	
	Self-Learning Topics: Applications of balanced trees in ordered data management, Comparative study: AVL vs Red-Black vs Splay Trees		
3	**Multiway & External Memory Data Structures**		**08**
	After completing this module, students will be able to: • Explain the structure and working of B-Trees and B+ Trees in the context of external memory. *(CO2)* • Analyze disk access cost and height of multiway trees for large datasets. *(CO4)* • Apply B-Tree/B+ Tree structures in database indexing and file systems. *(CO3)* • Evaluate and justify the choice of indexing structures for efficient data retrieval. *(CO5)*		
	3.1	• Motivation for multiway search trees • B-Trees: Structure and properties, Insertion and deletion, Height and disk access analysis	
	3.2	• B+ Trees: Difference between B-Tree and B+ Tree, Indexing and range queries, Applications in DBMS and file systems • Impact of secondary storage on data structure design	

	Self-Learning Topics: Clustered vs non-clustered indexing		
4	**Advanced Heap & Priority Queue Structures**		**07**
	After completing this module, students will be able to: • Implement and compare Binary, Binomial, and Fibonacci Heaps for priority queue operations. *(CO3)* • Analyze amortized complexities and understand efficiency improvements in advanced heap structures. *(CO4)* • Evaluate heap structures based on operational requirements such as merge, insert, and delete-min. *(CO5)* • Design efficient priority-based systems using suitable heap structures. *(CO6)*		
	4.1	Binary Heaps: Structure and array representation, Heap property (min-heap & max-heap), Insertion, deletion, and heapify operations, Limitations of binary heaps in advanced applications	
	4.2	Binomial Heaps: Structure and operations, Merge operation and complexity	
	4.3	Fibonacci Heaps: Lazy operations, Amortized complexity intuition	
	Self-Learning Topics: Comparative analysis of binary, binomial, and Fibonacci heaps		
5	**Advanced Graph Data Structures**		**08**
	After completing this module, students will be able to: • Apply Disjoint Set Union (Union-Find) with optimizations like path compression and union by rank. *(CO3)* • Analyze amortized complexity of Union-Find operations and justify efficiency improvements. *(CO4)* • Solve real-world problems such as dynamic connectivity, clustering, and scheduling using DSU and DAGs. *(CO3, CO6)* • Evaluate graph-based solutions for performance and scalability. *(CO5)*		
	5.1	• Disjoint Set Union (Union-Find): Union by rank, Path compression, Amortized time complexity • Applications of DSU: Dynamic connectivity, Offline graph queries	
	5.2	Directed Acyclic Graphs (DAG): Properties and applications, Scheduling and dependency resolution	
	Self-Learning Topics: DSU applications in image segmentation and clustering		
6	**Approximation Algorithms & Computational Complexity**		**08**
	After completing this module, students will be able to: • Apply approximation techniques (greedy, local search) to solve NP-hard problems with performance guarantees. *(CO3, CO4)* • Analyze approximation ratios and evaluate solution quality. *(CO4, CO5)* • Differentiate and apply randomized algorithms (Monte Carlo vs Las Vegas) in appropriate scenarios. *(CO2, CO3)* • Design efficient algorithms under constraints of time, uncertainty, or incomplete information. *(CO6)*		

	6.1	• Approximation Algorithms: Motivation for approximation algorithms, Approximation Algorithms: Greedy approximation techniques, Local search–based approximation. • ase studies: Vertex Cover (2-approximation), Set Cover (greedy approach)	
	6.2	• Randomized & Probabilistic Algorithmic Techniques: Role of randomness in algorithm design • Monte Carlo and Las Vegas algorithms	
	Self-Learning Topics: Online Algorithm: Online paging problem		
		TOTAL	**45**

Suggested List of Experiments:

Experiment No.	Title of Experiment
01	Implement a dynamic array supporting insertion and deletion operations with automatic resizing. 1. Track the actual cost of each operation. 2. Compute and compare: Worst-case cost and Aggregate amortized cost Plot or tabulate the cost variation and justify why amortized analysis gives a tighter bound. **Objective:** To implement a dynamic array with automatic resizing and analyze its performance using amortized analysis techniques. **Outcome:** Students will be able to evaluate and justify the efficiency of dynamic data structures by comparing worst-case and amortized costs.
02	To implement AVL Tree supporting insertion and search operations, and to experimentally evaluate their performance by measuring: 1. Tree height 2. Number of rotations 3. Search time For a large set of randomly ordered keys, and to justify the preferable structure under different scenarios. **Objective:** To implement an AVL tree and experimentally analyze its balancing operations and performance metrics. **Outcome:** Students will be able to analyze the impact of height balancing on search efficiency and evaluate AVL tree performance.
03	To implement Red-Black Tree supporting insertion and search operations, and to experimentally evaluate their performance by measuring: 1. Tree height 2. Number of rotations 3. Search time For a large set of randomly ordered keys, and to justify the preferable structure under different scenarios. Compare its performance with AVL tree. **Objective:** To implement a Red-Black tree and compare its performance with AVL trees under various input conditions. **Outcome:** Students will be able to evaluate trade-offs between AVL and Red-Black trees based on rotations, height, and search performance.
04	Implement a **Splay Tree** supporting: • Insert • Search

	Every search operation should splay the accessed node to the root using appropriate rotations. Conduct experiments under different input and access patterns and analyze performance behavior. **Objective:** To implement a Splay tree and analyze its self-adjusting behavior under different access patterns. **Outcome:** Students will be able to analyze amortized performance and justify the efficiency of self-adjusting trees in practical scenarios.
05	Implement a **B-Tree of order m (user-defined)** supporting: • Insert • Search Conduct experiments by inserting large datasets and analyze structural and performance characteristics. Compare performance with a Binary Search Tree (BST). **Objective:** To implement a B-Tree and analyze its structural efficiency for handling large datasets. **Outcome:** Students will be able to evaluate multiway tree structures and justify their superiority over binary trees in external memory contexts.
06	Implement a **B+ Tree** of configurable order (m) supporting the following operations: • Insert • Search • Range Query Insert a large dataset and evaluate the structural and performance characteristics of the B+ Tree. Compare the performance with a Binary Search Tree (BST) or B-Tree. **Objective:** To implement a B+ Tree supporting indexing and range queries and evaluate its performance. **Outcome:** Students will be able to analyze and justify the effectiveness of B+ Trees for database indexing and range query operations.
07	Implement a **priority queue using a Binary Heap** supporting the following operations: • Insert • Delete-Min / Delete-Max • Heapify (Build Heap) • Peek (Find minimum/maximum) Conduct experiments to analyze performance for different input sizes and compare **heap construction using insertion and heapify methods**. **Objective:** To implement a priority queue using a binary heap and analyze different heap construction methods. **Outcome:** Students will be able to compare heap operations and evaluate efficiency differences between insertion-based and heapify-based construction.
08	Implement a **Binomial Heap** supporting the following operations: • Insert • Find-Min • Delete-Min • Merge two heaps Perform experiments to analyze the performance of operations for different input sizes and compare the **merge operation performance with Binary Heap**. **Objective:** To implement a Binomial Heap and analyze its performance, especially for merge operations. **Outcome:** Students will be able to evaluate advanced heap structures and justify the advantages of binomial heaps in merge-intensive applications.

09	Implement a **Fibonacci Heap** supporting the following operations: • Insert • Decrease-Key • Delete-Min • Merge two Fibonacci heaps Perform experiments to analyze performance for different input sizes and compare with Binary Heap operations. **Objective:** To implement a Fibonacci Heap and analyze its amortized performance for priority queue operations. **Outcome:** Students will be able to analyze amortized efficiency and compare Fibonacci heaps with other heap structures for optimized operations.
10	Implement **Disjoint Set Union (Union–Find)** supporting the following operations: • MakeSet • Find • Union Implement three versions: 1. Naïve Union–Find 2. Union by Rank 3. Union by Rank + Path Compression Perform experiments to compare performance of these implementations. **Objective:** To implement and compare different versions of Union-Find with optimization techniques. **Outcome:** Students will be able to analyze the impact of path compression and union by rank on algorithm efficiency.
11	Implement a **Greedy Approximation Algorithm for the Vertex Cover problem** for an undirected graph. The algorithm should: 1. Select an edge (u, v). 2. Add both u and v to the vertex cover. 3. Remove all edges incident on u and v. 4. Repeat until no edges remain. Analyze the size of the vertex cover obtained and compare it with the optimal solution (for small graphs). **Objective:** To implement a greedy approximation algorithm for the Vertex Cover problem and evaluate solution quality. **Outcome:** Students will be able to analyze approximation techniques and evaluate solution accuracy against optimal results.
12	Given a universal set **U** and a collection of subsets **S = {S_1, S_2, S_3, …, S_n}**, implement a **Greedy Approximation Algorithm** to find a minimum number of subsets that cover all elements in U. Greedy strategy: 1. Select the subset that covers the maximum number of uncovered elements. 2. Add that subset to the solution. 3. Remove covered elements from U. 4. Repeat until all elements are covered. Analyze the size of the set cover obtained and compare with optimal solution for small datasets. **Objective:** To implement a greedy approximation algorithm for the Set Cover problem. **Outcome:** Students will be able to evaluate the effectiveness of greedy strategies for NP-hard problems and analyze approximation performance.

13	Implement a **Monte Carlo algorithm** for one of the following problems: • Primality Testing (Monte Carlo method) • Randomized Matrix Multiplication Verification • Randomized Pattern Matching • Estimation of π using random sampling Run the algorithm multiple times and analyze: • Execution time • Probability of correctness Effect of number of trials on accuracy **Objective:** To implement a Monte Carlo randomized algorithm and analyze its probabilistic behavior. **Outcome:** Students will be able to evaluate trade-offs between accuracy and efficiency in randomized algorithms.
14	Implement a **Las Vegas randomized algorithm** for one of the following problems: • Randomized Quick Sort • Randomized Search in an Unsorted List • Randomized Minimum Cut Algorithm • Randomized Selection (Quickselect) The algorithm should always produce the correct output, but execution time may vary due to randomness. Analyze execution time and number of iterations for different input sizes. **Objective:** To implement a Las Vegas randomized algorithm and analyze variability in execution time. **Outcome:** Students will be able to analyze randomized algorithms that guarantee correctness and evaluate performance variability.

Text Books:

1. Introduction to Algorithms by Thomas H Cormen, Charles E. Leiserson, Ronald L Rivest, Clifford Stein, Third Edition.
2. Horowitz, Sahani and Rajsekaran, —Fundamentals of Computer Algorithms‖, Galgotia.
3. "Data Structures using C and C++" by Yedidyah Langsam, Moshe J. Augenstein, Aaron M. Tenenbaum, 2nd edition, Prentice Hall

Reference Books:

1. Data Structures using C", Reema Thareja, Third Edition, Oxford University Press.
2. "Data Structures and Program Design in C++", Robert L. Kruse, Alexander J. Ryba, Prentice-Hall India.
3. "Data Structures and Algorithms in Java", Goodrich and Tamassia, John Wiley and Sons, Sixth Edition 2014. John Wiley & Sons.
4. "Data Structures and Pseudocode approach with C", 2nd Edition by Richard F. Gilberg; Behrouz A Forouzan, Thomson Publishing.

Useful Links:

4. https://nptel.ac.in/courses/106/106/106106131/
5. https://www.coursera.org/specializations/data-structures-algorithms
6. https://visualgo.net
7. www.leetcode.com
8. www.hackerrank.com
9. www.codechef.com
10. https://www.mooc-list.com/tags/algorithms

Assessment Methodology:

Type of Assessment	Assessment Tools
Continuous Assessment (CA) **(20 Marks)**	Certification: NPTEL (20 Marks) (Approved by instructor) **OR** Any 02 Pedagogies (10 marks each) • MCQ /Class Test • Case study/Assignment • GATE based Assignment • Certification Udemy/Coursera (Approved by instructor) • Open Book Test • Working model / Simulation of a course-based concept.
Mid Semester Examination (MSE) **(30 Marks)**	Question Paper Pattern is as follows: All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks • For each question, A and B should be based on the same CO. • MSE should be based on 50% syllabus. • Time: 90 minutes (1 hour 30 minutes) • Total Marks: 30
End Semester Examination (ESE) **(50 Marks)**	Question Paper Pattern is as follows: All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks • Q4 A or B - 10 marks • Q5 A or B - 10 marks • For each question, A and B should be based on the same CO. • ESE should be based on 30% syllabus of MSE and 70% syllabus after MSE. • Time: 120 minutes (02 hours) • Total Marks: 50
Term Work **(25 Marks)**	• Active Participation (Lab) = 05 marks • Laboratory Report / Journal = 10 marks • Laboratory Performance = 10 marks • Based on the performance & satisfactory completion of minimum 10 experiments.
Practical & Oral **(25 Marks)**	• Practical examination will be based on the experiments performed by the students during laboratory sessions.

Computer Network

Course Code	Course Name	Teaching Scheme (Hrs. / Week)			Credits Assigned			
		L	P	T	L	P	T	Total
25CE4PCC03	Computer Network	3	2	-	3	1	-	4
		Examination Scheme						
			CA	MSE	ESE	TW	OR	PR	Total
		Theory	20	30	50	-	-	-	100
		Lab	-	-	-	25	25	-	50

Pre-requisite: Nil

Course and Module Overview:

This course provides a comprehensive and structured understanding of **Computer Networks**, covering concepts from fundamental networking principles to advanced network design and protocol analysis. It begins with an introduction to networking concepts, performance metrics, and layered communication models, building a strong conceptual foundation for understanding how modern communication systems operate.

The course then progresses through the core layers of network architecture, including the **Physical, Data Link, Network, Transport, Session, Presentation, and Application layers**, emphasizing both theoretical concepts and practical implementation aspects. Students learn how data is transmitted, controlled, routed, and managed across networks, along with the mechanisms that ensure reliability, efficiency, and security.

It is especially relevant for Computer Engineering students, as modern computing systems—such as cloud computing platforms, IoT ecosystems, mobile networks, and distributed applications—rely heavily on robust and efficient networking protocols and architectures.

Module 1 builds the foundation by introducing computer networks, their applications, and key performance metrics such as latency, throughput, jitter, and packet loss. It also covers layered communication models like OSI and TCP/IP, helping students understand how complex network functions are organized and interact. This module enables students to analyze network performance and understand real-time communication requirements.

Module 2 focuses on the transmission of data over physical media and reliable node-to-node communication. It covers guided and unguided media, switching techniques, framing, error detection and correction methods, and flow control mechanisms. Medium Access Control protocols such as ALOHA, CSMA/CD, and CSMA/CA are also studied. These concepts are essential for understanding how data is efficiently transmitted over shared communication channels.

Module 3 introduces routing and addressing mechanisms that enable data delivery across multiple networks. It covers IPv4 and IPv6 addressing, subnetting, supernetting, NAT, and routing algorithms such as Distance Vector and Link State. Protocols like ARP, ICMP, and IGMP are discussed along with

congestion control techniques. This module equips students to design scalable and efficient internetworks.

Module 4 deals with end-to-end communication and process-level data delivery. It covers connection-oriented and connectionless protocols, focusing on TCP and UDP, including flow control, error control, congestion control, and timers. Session layer concepts such as Remote Procedure Call (RPC) are also introduced. These concepts are critical for ensuring reliable and efficient data transmission in applications.

Module 5 focuses on data representation and application-level communication. It includes data compression techniques such as Huffman coding, LZW, and image compression methods like JPEG and GIF. It also covers application protocols such as HTTP, FTP, SSH, DHCP, and DNS, enabling students to understand how real-world network services function.

Module 6 integrates all concepts by focusing on practical network design. It introduces VLANs and VPNs and includes a case study for designing a network for an organization, covering IP addressing, routing protocols, and essential network services. This module helps students apply theoretical knowledge to real-world scenarios and develop practical network design skills.

Overall, the course equips students with a strong understanding of networking principles, protocol design, performance analysis, and practical network implementation. It prepares them for advanced studies, industry certifications, and careers in networking, cybersecurity, cloud computing, and distributed systems.

Course Outcomes: On successful completion, of course, learner/student will be able to:

CO1	Recall fundamental networking concepts, models, devices and protocols. (Remembering)
CO2	Explain how different networking layers manage error control, flow control, switching, routing, congestion, and protocol functions. (Understanding)
CO3	Apply IP addressing, subnetting, routing, framing, error control, flow control, channel allocation and performance metrics to solve networking tasks.(Applying)
CO4	Analyze routing, traffic patterns and protocol behavior. (Analyzing)
CO5	Evaluate architectures, techniques, routing and algorithms for suitability and performance of a network.(Evaluating)
CO6	Create and implement a full network solution for a specified case study.(Creating)

Module	Detailed Contents		Hours
1	**Introduction to Networking**		**05**
	After completing this module, students will be able to: • Describe computer networks and their applications • Explain real-time network performance metrics such as latency, jitter, throughput, and packet loss • Analyze network performance using metrics like RTT, congestion, and utilization • Explain the need for layered architecture and compare OSI and TCP/IP models • Illustrate data flow, encapsulation, and interactions between network layers		
	1.1	**Introduction:** Introduction to computer network and network applications	
	1.2	**Real-Time Network Performance Metrics :** Latency(Delay), Jitter, Throughput, Round Trip Time (RTT), Packet Loss, Congestion,Error Rate,Availability, CPU/Buffer utilization, Queue Length, Link Utilization, Re-transmission Rate, Connection Setup Time, numerical analysis of performance metrics	
	1.3	**Layered Communication Models:** Need of layered structure, OSI & TCP/IP models, Encapsulation, data flow, layer interactions, design issues of the layers	
		Self-Learning Topic: Interconnection devices, network topologies, Identify the different devices and topologies used in Network connection of College campus.	
2	**Physical Layer & Data Link Layer**		**10**
	After completing this module, students will be able to: • Describe guided and unguided transmission media and their characteristics • Explain switching techniques such as circuit switching and packet switching • Analyze data link layer design issues including framing, error control, and flow control • Apply error detection and correction techniques like parity, CRC, checksum, and Hamming code • Evaluate elementary data link protocols and sliding window mechanisms • Explain MAC protocols including channel partitioning and random access methods		
	2.1	**Physical layer:** Guided Media, Unguided Media, Switching: Circuit-Switching and Packet Switching, Structure Of a Switch	

	2.2	**Data Link Layer:** DLL Design Issues (Services, Framing, Error Control,Flow Control), Error Detection and Correction (Hamming Code, Parity, CRC, Checksum) , Elementary Data Link protocols : Stop and Wait, Sliding Window (Go Back N,Selective Repeat), Piggybacking	
	2.3	**Medium Access Control Sub-layer:** Channel Partitioning Protocols (TDMA,FDMA, CDMA), Random Access Protocols (ALOHA - Pure/Slotted,CSMA-CSMA/ CD, CSMA/CA)	
		Self-Learning Topic: Differentiate data link layer in IOT network and Normal Network.	
3		**Network Layer**	**12**
	After completing this module, students will be able to: • Explain network layer design issues and communication primitives (unicast, multicast, broadcast) • Apply IP addressing techniques including IPv4, IPv6, sub-netting, and super-netting • Analyze packet forwarding, datagram structure, and NAT concepts • Evaluate routing algorithms such as Distance Vector, Link State, and shortest path (Dijkstra's algorithm) • Describe routing protocols like ARP, RARP, ICMP, and IGMP • Analyze congestion control techniques and QoS mechanisms		
	3.1	**Introduction:** Network Layer design issues, Communication Primitives: Unicast, Multicast, Broadcast.	
	3.2	**The Internet Protocol (IP):** Forwarding and Addressing in the Internet, Datagram Format, IPv4 Addressing (classful and classless),Subnetting, Supernetting design problems, IPv4 Protocol, Network Address Translation (NAT), IPv6 Protocol	
	3.3	**Routing algorithms:** Link state routing, Distance Vector Routing, Shortest path (Dijkastra's Algorithm)	
	3.4	**Routing Protocols:** ARP, RARP, ICMP, IGMP	
	3.5	**Congestion control algorithms:** Open loop congestion control, Closed loop congestion control, QoS parameters, Token & Leaky bucket algorithms.	
		Self-Learning Topic: Routing in the Internet: RIP, OSPF, BGP	
4		**Transport Layer & Session Layer**	**08**
	After completing this module, students will be able to: • Explain transport layer services and compare connection-oriented and connectionless protocols • Analyze the working of UDP and its applications • Describe TCP features including segmentation, connection management, and window mechanisms		

	• Apply flow control, error control, and congestion control techniques in TCP • Explain TCP timers and their role in reliable communication • Describe session layer functions and Remote Procedure Call (RPC) mechanism		
	4.1	**Transport Layer:** Transport Layer Services, Connectionless & Connection-oriented Protocols Transport Layer protocols: User Datagram Protocol: UDP Services, UDP Applications Transmission Control Protocol: TCP Services, TCP Features, Segment, A TCP Connection, Windows in TCP, Flow Control, Error Control, TCP Congestion Control, TCP Timers.	
	4.2	**Session Layer:** Session layer design issues, Session Layer protocol - Remote Procedure Call (RPC)	
		Self-Learning Topic: Berkeley Sockets, List real time example of UDP and TCP	
5	**Presentation Layer & Application Layer**		**06**
	After completing this module, students will be able to: • Explain the role of the presentation layer in data representation and compression • Differentiate between lossy and lossless compression techniques • Apply compression methods such as Huffman coding, LZW, RLE, and image formats (GIF, JPEG) • Describe application layer protocols and their functionalities • Analyze working of client-server protocols like HTTP, FTP, SSH, DHCP, and DNS		
	5.1	**Presentation layer:** Compression: Comparison between Lossy Compression and Lossless Compression, Huffman Coding, Speech Compression:LZW, RLE, Image Compression – GIF,JPEG.	
	5.2	**Application layer:** Standard Client-Server Protocols: HTTP, FTP, SSH, DHCP, Domain Name System (DNS)	
		Self-Learning Topic: Study of Mail Access Protocol -SMTP	
6	**Network Design Concepts**		**04**
	After completing this module, students will be able to: • Explain the concepts and benefits of VLAN and VPN • Design a network for an organization based on given requirements • Select appropriate networking devices, IP addressing schemes, and routing protocols • Identify and configure essential network services such as TELNET, SSH, FTP, web, file, DHCP, and DNS servers • Apply networking concepts to develop efficient and secure network architectures		
	6.1	Introduction to VLAN ,VPN	

	6.2	A case study to design a network for an organization meeting the following guidelines: Networking Devices, IP addressing, Routing Protocols to be used, Services to be used: TELNET, SSH, FTP server, Web server, File server, DHCP server and DNS server.	
		Self-Learning Topic: Study the Network Design of your college campus.	
		Total	**45**

Suggested List of Experiments:

Experiment No.	Title of the Experiment
1	Basic Networking Commands in Linux **Objective:** To study and use fundamental Linux networking commands such as ping, traceroute, nslookup, netstat, ARP, RARP, ip/ifconfig, dig, and route for analyzing network connectivity, troubleshooting, and configuration. **Outcome:** Students will be able to verify network connectivity, analyze routing paths, resolve domain names, inspect active connections, and interpret ARP tables. They will also gain hands-on experience in diagnosing network issues using essential Linux networking tools.
2	Study of Different Connectors and Cables using Packet Tracer **Objective:** To study various types of network cables and connectors in **Cisco Packet Tracer**, including straight-through, cross-over, fiber optic cables, and console connections, and understand their appropriate usage in network setups. **Outcome:** Students will be able to identify different network cables and connectors, select appropriate cable types for connecting devices, and understand their role in establishing reliable network communication.
3	Static Routing Configuration using Packet Tracer **Objective:** To build a simple network topology in Cisco Packet Tracer and configure IP addressing, subnetting, subnet masks, and static routing to enable communication between multiple networks. **Outcome:** Students will be able to design a basic network topology, assign appropriate IP addresses and subnet masks, and configure static routes to establish end-to-end connectivity between different networks.
4	Configuration of Web, E-mail, and DNS Services using Packet Tracer **Objective:**

	To configure and analyze Web (HTTP), E-mail (SMTP/POP3), and DNS services in Cisco Packet Tracer for a functional network setup. **Outcome:** Students will be able to configure servers for web hosting, email communication, and domain name resolution, and verify their operation by accessing web pages, sending/receiving emails, and resolving domain names within the network.
5	Network Discovery using Nmap **Objective:** To perform network discovery using the Nmap tool for identifying active hosts, open ports, and services in a network. **Outcome:** Students will be able to scan networks to detect live hosts, analyze open ports and running services, and interpret scan results for basic network security assessment and troubleshooting.
6	Traffic Analysis using Wireshark **Objective:** To analyze network traffic flow of different protocols using network monitoring tools like Wireshark. **Outcome:** Students will be able to capture and analyze packets, identify different network protocols, interpret protocol behavior, and understand data flow across the network for troubleshooting and performance analysis.
7	Network Simulation using TCL Scripts in NS2 **Objective:** To write TCL scripts for creating network topologies, generating traffic, and analyzing performance using the NS2 simulator. **Outcome:** Students will be able to design network topologies through scripting, simulate different types of traffic, and analyze simulation results such as throughput, delay, and packet loss for performance evaluation.
8	Traffic Simulation and Visualization using NS2 and NAM **Objective:** To write TCL scripts for creating network topologies, simulate traffic using TCP and UDP protocols, visualize the simulation using NAM, and plot performance graphs using NS2 outputs. **Outcome:** Students will be able to simulate TCP and UDP traffic, visualize packet flow through graphical animation, and analyze performance metrics by plotting graphs such as throughput, delay, and packet loss.
9	Implementation of Routing Protocols using NS2 / Packet Tracer

	Objective: To implement and analyze distance vector and link state routing protocols using NS2 or Cisco Packet Tracer. **Outcome:** Students will be able to configure and simulate routing protocols, understand the working of distance vector and link state algorithms, and analyze routing behavior and convergence in a network.
10	Simulation of ALOHA and CSMA/CD Protocols **Objective:** To use network simulators such as NS2 or Cisco Packet Tracer to understand the working of ALOHA and CSMA/CD protocols. **Outcome:** Students will be able to simulate and analyze random access protocols, understand collision handling mechanisms, and compare the performance of ALOHA and CSMA/CD in terms of efficiency and throughput.
11	IP Address Classification and Subnet Analysis **Objective:** To write a program that determines the class of a given IP address and computes the subnet mask, as well as the first and last IP addresses of the corresponding network block. **Outcome:** Students will be able to classify IP addresses into different classes, calculate subnet masks, and determine network range including the first (network) and last (broadcast) addresses for a given IP block.
12	VPN Design and Routing Configuration using Packet Tracer **Objective:** To design a Virtual Private Network (VPN) and configure routing protocols such as RIP and OSPF in Cisco Packet Tracer for secure and efficient network communication. **Outcome:** Students will be able to implement VPN connectivity, configure and compare RIP and OSPF routing protocols, and analyze secure data transmission and routing behavior across networks.
13	Socket Programming using UDP **Objective:** To implement socket programming using the User Datagram Protocol (UDP) for establishing connectionless communication between client and server systems. **Outcome:** Students will be able to develop UDP-based client-server programs, understand connectionless data transmission, and analyze the behavior of datagram communication in terms of speed and reliability.

14	Socket Programming using TCP **Objective:** To implement socket programming using the Transmission Control Protocol (TCP) for establishing reliable, connection-oriented communication between client and server systems. **Outcome:** Students will be able to develop TCP-based client-server programs, understand connection establishment and termination, and analyze reliable data transmission mechanisms such as sequencing, acknowledgment, and flow control.
15	Remote Login using FTP Server in Packet Tracer **Objective:** To configure and perform remote login and file transfer using an FTP server in Cisco Packet Tracer. **Outcome:** Students will be able to configure an FTP server, establish remote connections using FTP, and perform file upload and download operations between client and server systems.
16	Error Detection and Correction Codes Implementation **Objective:** To write a program for implementing error detection and correction techniques such as Hamming Code, Parity Check, CRC, and Checksum. **Outcome:** Students will be able to implement and compare different error detection and correction methods, identify errors in transmitted data, and understand techniques for ensuring data integrity in communication systems.
17	Congestion Control Algorithms Implementation **Objective:** To write a program to implement congestion control algorithms such as Leaky Bucket and Token Bucket for regulating network traffic flow. **Outcome:** Students will be able to understand and implement traffic shaping mechanisms, analyze the behavior of Leaky Bucket and Token Bucket algorithms, and evaluate their effectiveness in controlling congestion and ensuring smooth data transmission.
18	Implementation of Framing Methods **Objective:** To write a program to implement different framing techniques such as character count, byte stuffing, and bit stuffing for data transmission. **Outcome:**

	Students will be able to implement various framing methods, understand how data is encapsulated for transmission, and analyze the effectiveness of each technique in ensuring proper data delimitation and synchronization.
19	Network Design and Simulation for an Organization **Objective:** To design and simulate a network for an organization based on given specifications using tools such as Cisco Packet Tracer, including selection of appropriate topology, IP addressing scheme, and configuration of network services. **Outcome:** Students will be able to analyze network requirements, design an efficient and scalable network architecture, implement necessary configurations, and evaluate network performance to meet organizational objectives.
20	Smart Campus Network Design and Simulation **Objective:** To design and simulate a virtual "Smart Campus" network using Cisco Packet Tracer, incorporating VLANs, subnetting, and integration of IoT devices such as CCTV cameras, sensors, and smart boards. **Outcome:** Students will be able to create segmented networks using VLANs, implement subnetting for efficient IP management, integrate IoT devices into the network, and analyze secure and scalable communication within a smart campus environment.

Text Books:

1. A.S. Tanenbaum, Computer Networks, 6th edition Pearson Education, 2020
2. B.A. Forouzan, Data Communications and Networking with TCP/IP Protocol Suite, 6 th edition, TMH, 2022
3. James F. Kurose, Keith W. Ross, Computer Networking, A Top-Down Approach Featuring the Internet, 6 th edition, Pearson, 2017

Reference Books:

1. Behrouz A. Forouzan, Firouz Mosharraf, Computer Networks: A Top-Down Approach,Mc Graw Hill, 2023
2. S. Keshav, An Engineering Approach to Computer Networks, 2nd Edition, Pearson Education.
3. Khalid Sayood, Introduction to Data Compression, Third Edition, Morgan Kaufman.

Useful Links:

1. Computer Networks And Internet Protocol:

https://onlinecourses.nptel.ac.in/noc26_cs35/preview

2. Basics of Computer Networks:
   https://olympus.mygreatlearning.com/courses/88863?th=b&pb_id=581
3. Introduction to Computer Networking Basics:
   https://www.simplilearn.com/free-computer-networking-course-skillup

Assessment Methodology:

Type of Assessment	Assessment Tools
Continuous Assessment (CA) **(20 Marks)**	Certification: NPTEL (20 Marks) (Approved by instructor) **OR** Any 02 Pedagogies (10 marks each) • MCQ /Class Test • Case study/Assignment • GATE based Assignment • Certification Udemy/Coursera (Approved by instructor) • Open Book Test • Working model / Simulation of a course-based concept.
Mid Semester Examination (MSE) **(30 Marks)**	Question Paper Pattern is as follows: All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks • For each question, A and B should be based on the same CO. • MSE should be based on 50% syllabus. • Time: 90 minutes (1 hour 30 minutes) • Total Marks: 30
End Semester Examination (ESE) (50 Marks)	Question Paper Pattern is as follows: All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks • Q4 A or B - 10 marks • Q5 A or B - 10 marks • For each question, A and B should be based on the same CO. • ESE should be based on 30% syllabus of MSE and 70% syllabus after MSE. • Time: 120 minutes (02 hours) • Total Marks: 50

Term Work (25 Marks)	• Active Participation (Lab) = 05 marks • Laboratory Report / Journal = 10 marks • Laboratory Performance = 10 marks • Based on the performance & satisfactory completion of minimum 10 experiments.
Oral (25 Marks)	• Oral examination will be based on the entire syllabus.

Operating System

Course Code	Course Name	Teaching Scheme (Contact Hours)			Credits Assigned			
		Theory	Pract.	Tut.	Theory	Pract.	Tut.	Total
25CE4PCC04	Operating System	2	2		2	1		2

Course Code	Operating System	Examination Scheme							
		Theory					Term Work	Pract & oral	Total
		CA	CA	MSE	ESE				
25CE4PCC04		20	30				25	25	100

Pre-requisite Courses:
1. 25FE1VSEC02: Problem Solving using C Programming
2. 25CE3PCC03: Computer Organization and Architecture

Course Overview

This comprehensive Operating Systems course spans 15 weeks with 30 theory lectures and 15 practical sessions, structured around six modules covering OS fundamentals, process management, synchronization/deadlocks, memory management, file systems, and I/O/advanced topics. The theory component explores architectures, algorithms, and system concepts while practical labs provide hands-on experience with Linux programming, system calls, algorithm implementation, and kernel exploration using C and system tools. The integrated approach develops essential skills for system programming, cloud computing, and embedded systems through balanced conceptual understanding and practical application.

Module 1: Operating System Overview: Students will understand OS basics, structures, Linux architecture, system calls, and types of operating systems.

Module 2: Process Management: Students will learn process and thread concepts and implement various CPU scheduling algorithms.

Module 3: Process Synchronization & Deadlocks: Students will understand synchronization techniques, critical section problems, and deadlock handling methods.

Module 4: Memory Management: Students will learn memory allocation, paging, segmentation, virtual memory, and page replacement algorithms.

Module 5: File Management: Students will understand file systems, directory structures, and file allocation methods.

Module 6: I/O Management & Advanced Topics: Students will learn disk scheduling, I/O management, and basics of virtualization and security.

Course Outcomes: On successful completion, of course, learner/student will be able to:

1	Identify and recall OS architectures, components, and system call mechanisms from different operating systems.
2	Explain the working principles of scheduling algorithms, synchronization mechanisms, and memory management techniques in operating systems.
3	Implement CPU scheduling algorithms, process synchronization solutions, and memory management techniques using programming languages and simulation tools.
4	Compare and contrast the performance characteristics of different OS algorithms, evaluating their efficiency, throughput, and resource utilization.
5	Assess various OS design approaches, making informed judgments about their trade-offs in terms of performance, complexity, and suitability for different computing environments.
6	Design and develop system-level programs, synchronization solutions, and simple OS components to address concurrency, deadlock, and memory allocation challenges.

Module	Detailed Contents		Hours
1	**MODULE 1: OPERATING SYSTEM OVERVIEW** After completion of this module, students will be able to: • Understand the basic concepts, functions, and evolution of operating systems • Identify key components and structures of an operating system • Explain the architecture of Linux and the role of the Linux kernel • Understand the concept and functioning of system calls • Analyze different types of operating systems and their characteristics • Differentiate between various operating system structures and types		4
	1.1	Introduction, Objectives, Functions and Evolution of OS	
	1.2	OS Structures: Layered, Monolithic, Microkernel	
	1.3	Linux Kernel Architecture, Shell and System Calls	
	1.4	Types of OS: Batch, Time-sharing, Distributed, Real-time	

		Self-Learning 1. Compare the architecture of Windows NT, Linux Kernel, and macOS Darwin 2. Install Linux VM and explore kernel version using `uname -a`	
2	**MODULE 2: PROCESS MANAGEMENT** After completion of this module, students will be able to: • Explain the process lifecycle and the concept of threads • Understand and compare different CPU scheduling techniques • Apply and implement various scheduling algorithms • Evaluate the performance of scheduling algorithms using appropriate criteria		**8**
	2.1	Process Concept, States, PCB, Process Operations	
	2.2	Threads: Types, Models, Multithreading	
	2.3	CPU Scheduling: Types, Performance Criteria	
	2.4	Scheduling Algorithms: FCFS, SJF, Priority	
	2.5	Advanced Scheduling: Round Robin, Multi-level Queue	
		Self-Learning 1. Programming Task: Implement process tree visualization using C/Python 2. Online Resource: Complete scheduling algorithms module on GeeksforGeeks CPU Scheduling in Operating Systems - GeeksforGeeks	
3	**MODULE 3: PROCESS SYNCHRONIZATION & DEADLOCKS** After completion of this module, students will be able to: • Understand synchronization mechanisms and their importance • Analyze the critical section problem and its solutions • Apply semaphores and other synchronization techniques • Solve classical synchronization problems • Analyze deadlock conditions and methods for handling deadlocks		**6**
	3.1	Concurrency, Critical Section, Mutual Exclusion	
	3.2	Semaphores, Monitors, Classical Problems	

	3.3	Deadlock: Conditions, Prevention, Avoidance	
	3.4	Banker's Algorithm, Detection, Recovery	
		Self-Learning 1. Programming Assignment: Solve producer-consumer problem with three approaches 2. ResearchPaper: Dining Philosophers Theory and Concept in Operating System Scheduling. Dining-Philosophers-Theory-and-Concept-in-Operating-System-Scheduling.pdf 3. Online Practice: Complete synchronization problems on LeetCode/HackerRank	
4	**MODULE 4: MEMORY MANAGEMENT** After completion of this module, students will be able to: • Understand memory allocation techniques and their applications • Explain paging, segmentation, and virtual memory concepts • Apply various memory management methods • Analyze different page replacement algorithms • Understand page faults and evaluate memory usage efficiency		**7**
	4.1	Memory Allocation: Fixed/Dynamic Partitioning, Fragmentation	
	4.2	Paging: Basic, Multi-level, Hashed Page Tables	
	4.3	Segmentation, Segmentation with Paging	
	4.4	Virtual Memory: Demand Paging, Page Fault Handling	
	4.5	Page Replacement Algorithms: FIFO, Optimal, LRU	
		Self-Learning 1. Simulation Project: Create memory allocation visualizer 2. Programming: Implement page replacement algorithms with statistics	
5	**MODULE 5: FILE MANAGEMENT** After completion of this module, students will be able to: • Understand file system concepts and their structure • Explain file operations and directory structures		**3**

	• Compare different file allocation techniques • Understand various file storage and allocation methods		
	5.1	File System Interface: Concepts, Operations, Types	
	5.2	Directory Structure: Single-level, Tree-structured, Acyclic Graph	
	5.3	File Allocation Methods: Contiguous, Linked, Indexed	
		Self-Learning Case Study: Analyze Google File System (GFS) architecture	
6	**MODULE 6: I/O MANAGEMENT & ADVANCED TOPICS** After completion of this module, students will be able to: • Understand I/O operations and disk scheduling techniques • Apply various disk scheduling algorithms • Explain the basics of virtualization • Understand fundamental security concepts in operating systems • Gain foundational knowledge of virtualization and system security		**2**
	6.1	I/O Organization, Disk Structure, Disk Scheduling	
	6.2	Introduction to Virtualization and Security	
		Self-Learning Project: Create simple virtualization simulation	

Textbooks:

1.

1. William Stallings, Operating System: Internals and Design Principles, Prentice Hall, 8th Editic
2. Abraham Silberschatz, Peter Baer Galvin and Greg Gagne, Operating System Concepts, John Wiley & Sons, 9th Edition

References:

1. Achyut Godbole and Atul Kahate, Operating Systems, McGraw Hill Education, 3rd Edition
2. Andrew Tannenbaum, Operating System Design and Implementation, Pearson, 3rd Edition
3. Maurice J. Bach, "Design of UNIX Operating System", PHI

	Experiments List: (Suggested)
1	**Linux Environment & Basic Commands** **Objective:** To understand the Linux operating system environment and learn basic commands for file handling, directory management, and system navigation. **Outcome:** Students will be able to efficiently use Linux commands like ls, cd, pwd, cp, mv, rm, and understand the directory structure.
2	**Shell Scripting** **Objective:** To learn how to write and execute shell scripts for automating tasks in Linux. **Outcome:** Students will be able to create scripts using variables, loops, and conditional statements to automate repetitive tasks.
3	**System Calls Implementation** **Objective:** To understand and implement basic system calls like fork(), exec(), wait(). **Outcome:** Students will gain knowledge of how user programs interact with the OS kernel and manage processes.
4	**Process Creation & Management** **Objective:** To study process creation, execution, and termination in an operating system. **Outcome:** Students will be able to create and manage processes and understand process states and lifecycle.
5	**CPU Scheduling - Non-preemptive Algorithms** **Objective:** To implement and analyze non-preemptive CPU scheduling algorithms. **Outcome:** Students will understand scheduling metrics like waiting time and turnaround time and compare algorithm performance.

6	**CPU Scheduling - Preemptive Algorithms** **Objective:** To implement preemptive scheduling techniques and evaluate their efficiency. **Outcome:** Students will learn time-sharing concepts and how context switching affects performance.
7	**Process Synchronization** **Objective:** To understand synchronization problems and implement solutions using semaphores. **Outcome:** Students will be able to solve problems like Producer-Consumer and avoid race conditions.
8	**Deadlock Handling** **Objective:** To study deadlock conditions and implement deadlock prevention, avoidance (Banker's Algorithm), and detection. **Outcome:** Students will understand how deadlocks occur and how to handle them effectively.
9	**Memory Management - MVT & MFT** **Objective:** To study fixed and variable partition memory allocation techniques. **Outcome:** Students will understand internal and external fragmentation and memory utilization.
10	**Dynamic Partitioning Algorithms** **Objective:** To implement dynamic memory allocation techniques. **Outcome:** Students will be able to compare different allocation strategies and their efficiency.
11	**Virtual Memory - Demand Paging** **Objective:** To understand the concept of virtual memory and implement demand paging. **Outcome:** Students will learn how pages are loaded on demand and how page faults occur.
12	**Page Replacement Algorithms** **Objective:** To implement and compare page replacement techniques.

	Outcome: Students will understand page fault handling and evaluate algorithm performance.
13	**File Allocation Strategies** **Objective:** To study different file allocation methods used in operating systems. **Outcome:** Students will understand how files are stored and accessed on disk.
14	**Disk Scheduling Algorithms** **Objective:** To implement disk scheduling algorithms and analyze seek time. **Outcome:** Students will be able to compare performance and efficiency of disk scheduling techniques.
15	**Mini Project & Comprehensive Viva** **Objective:** To integrate and apply operating system concepts in a real-world mini project. **Outcome:** Students will demonstrate practical understanding of OS concepts and improve problem-solving, coding, and presentation skills.

Useful Links	
1	https://nptel.ac.in/courses/106/106/106106144/ (NPTEL OS Course)
2	https://swayam.gov.in/nd1_noc19_cs47/preview (SWAYAM OS Course)

Type of Course	**Assessment Tool**	**Marks Distribution**
Theory	CA-50	Any two Pedagogies (10 marks each) • MCQ /Class Test • Case study/Assignment • GATE based Assignment • Certification: Udemy/Coursera (Approved by instructor) • Open Book Test Working model / simulation of a course-based concept (30 Marks).
Course - Laboratory	TW- 25	• Active Participation (Lab) = 5 marks • Laboratory Report = 10 marks • Laboratory performance = 10 marks Based on the performance and satisfactory completion of assigned laboratory work
Laboratory	PR-25	Practical examination will be based on the experiments performed by the students during laboratory sessions.

Course Code	**Course Name**	**Teaching Scheme (Hrs. / Week)**	**Credits Assigned**

25IL4EEM01	**Entrepreneurship Essentials**	**L**	**P**	**T**	**L**	**P**	**T**	**Total**	
		2	-	-	2	-	-	**2**	
		Examination Scheme							
			CA	**MSE**	**ESE**	**TW**	**OR**	**PR**	**Total**
		Theory	50	-	-	-	-	-	**50**
		Lab/Tut	-	-	-	-	-	-	-
		Total	**50**						

Pre-Requisite Courses:	Basic knowledge of management functions, leadership, and organizational behavior.
	Ability to interpret simple financial data, manage budgets, and understand basic accounting principles & Proficiency in using digital tools line Excel.
	Communication, presentation, and interpersonal skills & critical thinking, creativity, problem-solving skills, and an interest in innovation and real-world business scenarios.

Course and Module Overview:

The course aims to develop an understanding of entrepreneurship and its role in economic development. It focuses on entrepreneurial behavior, theories, types of entrepreneurs, business idea generation, project planning, venture development, and the impact of external environments including government policies and global challenges. The course also encourages self-learning through case studies and real-world entrepreneurial practices in India.

Module 1: Foundations of Entrepreneurship Development

This module introduces the fundamentals of entrepreneurship and its significance in economic growth. It covers the concept and need for entrepreneurship development, definitions of entrepreneurship and entrepreneurs, and key characteristics required for entrepreneurial success. Case studies of entrepreneurs from small towns in India provide practical insights into grassroots entrepreneurship.

Module 2: Theories of Entrepreneurship & External Influences

This module explores major entrepreneurship theories proposed by Schumpeter, McClelland, Leibenstein, Knight, and Hagen. It examines how sociocultural, political, economic, and personal factors influence entrepreneurial development. The role of entrepreneurial culture in fostering innovation and enterprise creation is emphasized.

Module 3: Types & Classification of Entrepreneurs

This module focuses on different types of entrepreneurs, including intrapreneurs, women entrepreneurs, and social entrepreneurs. It discusses challenges faced by women entrepreneurs and the role of Self-Help Groups (SHGs). The module also highlights social entrepreneurship in India and the importance of NGOs in social responsibility and development.

Module 4: Entrepreneur Project Development & Business Idea

This module emphasizes innovation, creativity, and idea generation. Students learn about identifying

business opportunities through environmental scanning and change. The entrepreneurship development cycle and SWOT analysis help learners understand how to convert ideas into viable entrepreneurial ventures.

Module 5: Business Plan

This module covers the structure and objectives of a business plan. It includes market analysis, feasibility analysis, and planning for marketing, finance, organization, and management. Case studies on mergers, acquisitions, and takeovers of start-ups provide exposure to strategic business growth and expansion.

Module 6: Venture Development

This module explains the steps involved in starting a venture, institutional support systems, venture funding, and sources of finance. It discusses legal requirements, marketing channels, and challenges in venture setup. The module also analyzes the impact of COVID-19 on MSMEs in India and government relief measures under the Atma Nirbhar Bharat Abhiyan.

Summary

The course offers a comprehensive blend of theory, practical insights, and self-learning components to equip students with entrepreneurial knowledge and skills. It prepares learners to identify opportunities, develop business plans, and understand the challenges and support mechanisms for successful entrepreneurial ventures.

Course Outcomes	**After successful completion of the course, the students will be able to**	
	CO1	Recall fundamental concepts of entrepreneurship, identify types and characteristics of entrepreneurs, and list major entrepreneurship theories and government support initiatives in India.
	CO2	Explain the role of entrepreneurship in economic and social development and describe the influence of theories, innovation, creativity, and external environmental factors on entrepreneurial growth.
	CO3	Apply idea generation techniques, SWOT and feasibility analysis, and knowledge of finance, institutional support, and legal requirements to plan entrepreneurial ventures.
	CO4	Analyze market conditions, customer needs, challenges faced by women entrepreneurs, MSMEs and social entrepreneurs, and the impact of policies and external factors on ventures.
	CO5	Evaluate business plans, government support schemes, and entrepreneurial case studies to assess venture viability and development outcomes.
	CO6	Design innovative business ideas, develop comprehensive business plans, and create sustainable entrepreneurial solutions to address socio-economic and environmental challenges.

Syllabus:

Module No.	Unit No.	Topics	Hours
1		**Review of Module 1: Foundations of Entrepreneurship Development** After completing this module, students will be able to: • Explain the concept and need for entrepreneurship development. • Identify the characteristics and qualities of successful entrepreneurs. • Describe the importance of entrepreneurship in economic growth.	6
	1.1	Concept and need of entrepreneurship development.	
	1.2	Definition of entrepreneur, entrepreneurship importance and significance of growth of entrepreneurial activities.	
	1.3	Characteristics and qualities of an entrepreneur.	
		Self-Learning Topics: Case studies of entrepreneurs from small towns in India.	
2		**Review of Module 2: Theories of Entrepreneurship & External Influences on Entrepreneurship Development** After completing this module, students will be able to: • Explain major theories of entrepreneurship proposed by Schumpeter, McClelland, Leibenstein, Knight, and Hagen. • Examine the influence of sociocultural, political, economic, and personal factors on entrepreneurship development. • Assess the role of entrepreneurial culture in promoting entrepreneurial activities.	6
	2.1	Theories by: Schumpeter, McClelland, Leibenstein, Knight & Hagen.	
	2.2	External influences: Sociocultural, Political, Economic, Personal.	
	2.3	Role of entrepreneurial culture in entrepreneurship development.	
		Self-Learning Topics: Factors affecting entrepreneurship development.	
3		**Review of Module 3: Types & Classification of Entrepreneurs** After completing this module, students will be able to: • Distinguish between different types of entrepreneurs including intrapreneurs, women entrepreneurs, and social entrepreneurs. • Analyze the challenges faced by women entrepreneurs and the role of Self-Help Groups (SHGs).	6

	• Explain the importance of social entrepreneurship and the role of NGOs in social responsibility.		
	3.1	Intrapreneur: Types & classification, concept & development of intrapreneurship.	
	3.2	Women Entrepreneur: concept, development and problems faced by women entrepreneurs, development of women entrepreneurs with reference to the Self Help Group (SHG).	
	3.3	Social Entrepreneurship: concept, development of social entrepreneurship in India. Importance and Social responsibility of NGOs.	
	Self-Learning Topics: Entrepreneurial Development Program (EDP): concept & factors influencing EDP.		
4	**Review of Module 4: Entrepreneur Project Development & Business Idea** After completing this module, students will be able to: • Differentiate between innovation, invention, and creativity in the context of entrepreneurship. • Apply idea generation techniques and environmental scanning to identify business opportunities. • Explain the entrepreneurship development cycle and use SWOT analysis for project evaluation.		6
	4.1	Innovation, Invention, Creativity, Business idea, opportunities through change.	
	4.2	Idea generation: Sources, development of product or idea, environmental scanning.	
	4.3	Creating entrepreneurial ventures: Entrepreneurship development cycle.	
	Self-Learning Topics: SWOT analysis.		
5	**Review of Module 5: Business Plan** After completing this module, students will be able to: • Identify the elements and objectives of a business plan. • Analyze market and feasibility aspects of entrepreneurial ventures. • Evaluate marketing, financial, organizational, and ownership structures of business plans.		6
	5.1	Elements of business plan, objectives of business plan.	
	5.2	Market analysis and feasibility analysis.	
	5.3	Marketing, Finance, Organization & Management, Ownership.	
	Self-Learning Topics: Case study on takeover, mergers and acquisitions of start-ups in India & global.		

6	**Review of Module 6: Venture Development** After completing this module, students will be able to: ● Explain the steps involved in starting a venture and sources of finance. ● Identify legal requirements and marketing channels for establishing a new business unit. ● Analyze the impact of COVID-19 and government policy responses on MSMEs in India.		6
	6.1	Steps involved in starting a venture, institutional support to an entrepreneur, venture funding, requirements of capital (Fixed and working), sources of finance, problem of venture set-up and prospects.	
	6.2	Legal requirements for establishment of a new unit, Marketing: methods and channel.	
	6.3	Impact of COVID-19 on micro, small and medium enterprises in India, pandemic shock of COVID-19 and policy response.	
	Self-Learning Topics: Self Learning topics: Government financing support programme for businesses Covid 19, relief measures to small businesses in India (Atma Nirbhar Bharat Abhiyan).		
		TOTAL	36

Text Books:

1. Business Planning and Entrepreneurial Management by Michael Vaz & Meeta Seta, Publication: Manan Prakashan (2023 June, 7th edition), ISBN: 978-93-5750-083-8.
2. Business Planning and Entrepreneurial Management by Dr. Rinkesh Chheda & Ms. Falguni Mathews, Publication: Himalaya Publishing House (2019), ISBN: 978-93-5367-613-1.
3. Business Planning and Entrepreneurial Management by Veena Prasad & Deepali Kamle, Publication: Himalaya Publishing House (2018), ISBN: 978-93-5202-078-2.

Reference Books:

1. Corporate Entrepreneurship and Innovation by Paul Burns, Publication: Bloomsbury Academic (2025 June, 5th edition), ISBN (Paperback): 9781350384071, ISBN (eBook EPUB/MOBI): 9781350384095, ISBN (eBook PDF): 9781350384101.
2. Dynamics of Entrepreneurial Development & Management by Dr. Desai Vasant, Publication: Himalaya Publishing House (2019, 6th edition), ISBN: 978-93-5750-083-8.

3. Online Book, Institute of Distance & Open Learning (IDOL) University of Mumbai, https://old.mu.ac.in/wp-content/uploads/2014/04/Management-PAPER-V-ENTREPRENEURSHIP-Management-final-book.pdf

4. Entrepreneurship in the New Millennium by Donald F. Kuratko & Richard M. Hodgetts, Publication: Cengage learning South-Western Cengage Learning India (2008 Jan), ISBN: 978-8131505618 (typical Indian edition).

5. Business Planning: A Guide to Business Start-Up by Butler David, Publication: Taylor & Francis Ltd / Butterworth-Heinemann (2000). ISBN-13: 978-0-7506-4706-9

Useful Links:

1. https://www.ahlawatassociates.com/blog/legal-requirements-for-starting-a-business-in-india
2. https://smallbusiness.chron.com/business-plans-fail-projects-fail-10901.html
3. https://rcic.in/acquisitions/mergers-acquisitions-case-studies-india/
4. http://ecoursesonline.iasri.res.in/mod/page/view.php?id=1049

Assessment Methodology:

Type of Assessment	Assessment Tools
Continuous Assessment (CA) (50 Marks)	• Certification NPTEL: 20 Marks (Approved by instructor) ***And / Or*** **Any 05 Pedagogies (10 marks each)** • Assignment • Case Study Analysis (individual / group) • Certification: Udemy / Coursera (Approved by instructor) • Class Test • Discussion & Reflective Learning • Idea Generation & Opportunity Identification • MCQ Test • Open Book Test • Project Report: Venture Development Plan • Simulation of a course based concept.

Community Engagement Project

Course Code	Course Name	Teaching Scheme (Contact Hours)			Credits Assigned			
		Theory	Pract.	Tut.	Theory	Pract.	Tut.	Total
25CE4CEP02	Community Engagement Project		2			1		1

Course Code	Community Engagement Project	Examination Scheme							
		Theory					Term Work	Pract & oral	Total
		Internal Assessment			End Sem. Exam	Exam. Duration (in Hrs)			
		CA	MSE	ESE					
25CE4CEP02		-	-	-	-	-	25	25	50

Pre-requisite: Basic programming knowledge, Basic communication and teamwork abilities

Course Overview: This hands-on, community-based capstone course immerses students in the end-to-end process of addressing real-world challenges through applied design and technology. Grounded in Design Thinking, learners collaborate with community stakeholders to identify and refine meaningful problems, select appropriate technical stacks, and develop functional software, hardware, or IoT prototypes. The course emphasizes iterative development through user testing, community feedback, and validation, supported by structured planning and professional documentation. By the end of the course, students deliver a future-ready, industry-aligned solution that demonstrates applied innovation and prepares them to translate ideas into impactful outcomes for communities, industry, entrepreneurship, or advanced research.

Module 1: Project Refinement & Advanced Planning: This module will refine problem statements based on feedback and develop students' ability to plan projects effectively using tools like Gantt charts, PERT, and CPM, along with basic risk and resource management.

Module 2: Technical Stack Selection & Environment Setup: This module will help students select appropriate technical tools, software stacks, and platforms, and guide them in setting up a proper development environment using version control systems and modern development tools.

Module 3: Advanced Prototyping & Development: his module will focus on developing complete software and hardware prototypes by applying full-stack development concepts, integrating APIs, and working with IoT platforms and cloud services.

Module 4: Testing, Validation & Iteration: This module will teach students how to perform different types of testing, analyze user feedback, and improve system performance through iterative development and validation techniques.

Module 5: Documentation & Presentation Preparation: This module will develop students' ability to prepare professional technical documentation, including reports, user manuals, and presentations, along with creating demo videos for project demonstration.

Module 6: Final Showcase & Professional Development: This module will enhance students' presentation skills, help them build professional portfolios, and prepare them for project showcasing in competitions, exhibitions, or industry platforms.

Course Outcomes: On successful completion, of course, learner/student will be able to:

1	Explain key Design Thinking principles and project planning concepts relevant to prototype development.
2	Interpret user needs and feedback to refine problem statements and define effective design requirements.
3	Apply project planning, scheduling, and time management techniques to organize and execute mini-project activities.
4	Analyze user feedback, technical constraints, and feasibility studies to improve solution approaches and design decisions.
5	Evaluate prototype performance using test results and user validation data to determine effectiveness and areas for improvement.
6	Design, develop, and present a functional prototype supported by appropriate technical documentation and professional communication.

Module	Detailed Contents		Hours
1	**Module 1: Project Refinement & Advanced Planning** After completion of this module, students will be able to: • Analyze feedback to refine and improve problem statements • Apply project planning tools such as Gantt charts, PERT, and CPM • Improve problem definition based on iterative feedback • Apply planning and scheduling techniques for effective project execution		**4**
	1.1	Feedback analysis from Mini Project I; revising problem statements with technical scope	
	1.2	Advanced project planning: Gantt, PERT, CPM with risk and resource management	

2	**Module 2: Technical Stack Selection & Environment Setup** After completion of this module, students will be able to: • Select suitable technologies and tools for project development • Choose an appropriate technology stack based on requirements • Set up a proper development environment • Use version control systems such as Git for project management		**4**
	2.1	Technical Stack Selection & Environment Setup	
	2.2	Technology evaluation: software stacks, hardware platforms, IoT frameworks	
	2.3	Version control with Git/GitHub; collaborative workflows	
	2.4	Development environment setup: Docker, IDEs, debugging tools	
3	**Module 3: Advanced Prototyping & Development** **Objectives:** • Develop software/hardware prototypes • Learn APIs and cloud integration **Outcomes:** • Build functional prototypes • Integrate APIs and backend services		**8**
	3.1	Full-stack development basics (frontend, backend, database)	
	3.2	Hardware/IoT prototyping (Arduino, Raspberry Pi, sensors)	
	3.3	API integration (REST, MQTT) and cloud services	
4	**Module 4: Testing, Validation & Iteration** **Objectives:** • Learn testing methods • Analyze user feedback **Outcomes:** • Perform testing and validation • Improve system through iteration		**6**
	4.1	Test planning: unit, integration, usability testing	

	4.2	User validation methods and feedback analysis	
	4.3	Iterative development and prototype refinement	
5	**Module 5: Documentation & Presentation Preparation** **Objectives:** • Learn technical documentation • Prepare presentations and demo **Outcomes:** • Create proper project documentation • Present project effectively		**4**
	5.1	Technical writing: LaTeX/Markdown, API docs, user manuals	
	5.2	Creating demo videos and presentation slides	
6	**Module 6: Final Showcase & Professional Development** **Objectives:** • Develop presentation and portfolio skills • Prepare for professional exposure **Outcomes:** • Deliver final project presentation • Build portfolio and reflect on learning		**4**
	6.1	Final presentation skills and portfolio building	
	6.2	Submission to hackathons/exhibitions and project reflection	

Textbooks:	
	1. **Brown, T. (2009).** *Change by Design: How Design Thinking Transforms Organizations and Inspires Innovation.* Harper Business. 2. **Ulrich, K. T., & Eppinger, S. D. (2016).** *Product Design and Development* (6th Edition). McGraw-Hill Education. 3. **Pressman, R. S., & Maxim, B. R. (2019).** *Software Engineering: A Practitioner's Approach* (9th Edition). McGraw-Hill Education.
References:	

	1. **Stickdorn, M., Hormess, M., Lawrence, A., & Schneider, J. (2018).** *This Is Service Design Doing.* O'Reilly Media. 2. **Ries, E. (2011).** *The Lean Startup.* Crown Business. 3. **Martin, R. L. (2009).** *The Design of Business: Why Design Thinking Is the Next Competitive Advantage.* Harvard Business Press. .

Experiments List: (Suggested)

Exp. No.	Experiment Title	Tools/Platforms
1	**Revise problem statement and create detailed project plan** **Objectives:** • Refine problem statement • Create project plan **Outcomes:** • Define clear project scope • Develop structured project plan	Miro, Notion, GitHub Projects
2	**Set up version control and development environment** **Objectives:** • Learn Git and environment tools • Set up development workspace **Outcomes:** • Use Git for collaboration • Configure development tools	Git, Docker, VS Code
3	**Develop a functional software module with API integration** **Objectives:** • Develop software component • Integrate APIs **Outcomes:** • Build working module	React/Node.js, Postman, Firebase

	• Perform API integration	
4	**Build and test a hardware/IoT prototype** **Objectives:** • Build hardware prototype • Test sensors/devices **Outcomes:** • Develop IoT-based system • Perform hardware testing	Arduino/Raspberry Pi, PlatformIO
5	**Conduct user testing and analyze feedback** **Objectives:** • Conduct testing with users • Collect feedback **Outcomes:** • Analyze user feedback • Improve system design	UserTesting, Google Forms
6	**Prepare final documentation and demo video** **Objectives:** • Prepare reports and demo • Create presentation **Outcomes:** • Produce final documentation • Deliver effective demo	LaTeX/Overleaf, OBS Studio, Canva

Useful Links

1. IDEO Design Thinking Toolkit
   https://www.designkit.org
2. Stanford d.school – Design Thinking Resources
   https://dschool.stanford.edu/resources
3. Interaction Design Foundation – Design Thinking
   https://www.interaction-design.org/literature/topics/design-thinking
4. Project Ideas Repository
   https://nevonprojects.com/project-ideas/

5. Developer Roadmaps & Project Guides
   https://roadmap.sh/projects
6. GitHub Learning Lab
   https://lab.github.com

Type of Course	**Assessment Tool**	**Marks Distribution**
Mini-Project	TW- 25	Active Participation (Lab) = 5 marks Project Report = 10 marks Progress Presentations (Minimum – 02) & Demonstration = 10 Marks
	PR-25	Practical examination will be based on the experiments performed by the students during laboratory sessions.

The Bombay Salesian Society's

Don Bosco Institute of Technology, Mumbai

(An Autonomous Institute Affiliated to the University of Mumbai)

CURRICULUM STRUCTURE

Multi-Disciplinary Minor (MDM) Courses

(As Per NEP 2020)

Scheme: DB25-V1

(w.e.f. AY 2025-2026)

Preface

The evolving landscape of higher education emphasizes the need for students to acquire knowledge that extends beyond the boundaries of a single discipline. In alignment with the broader vision of the National Education Policy (NEP) 2020, engineering education increasingly encourages students to explore learning opportunities that integrate concepts from multiple domains. The Multi-Disciplinary Minor (MDM) Courses offered at Don Bosco Institute of Technology (DBIT), Mumbai, reflect this academic approach by providing structured pathways for students to develop competence in areas outside their primary field of study.

Multi-Disciplinary Minor courses enable students to broaden their academic exposure while pursuing their core engineering program. Through these courses, learners are encouraged to engage with complementary areas of knowledge that enhance their analytical ability, problem-solving skills, and intellectual curiosity. Such interdisciplinary learning supports the development of well-rounded professionals who are capable of understanding complex real-world challenges from multiple perspectives.

The MDM framework allows students to pursue a focused set of courses in a chosen domain, thereby enabling deeper engagement with emerging technologies, interdisciplinary applications, and contemporary areas of study. This approach encourages learners to build additional competencies that complement their primary discipline and strengthen their academic and professional profile.

These courses emphasize conceptual understanding, application-oriented learning, and collaborative exploration of ideas. Students are encouraged to participate in discussions, case studies, project-based learning, and practical activities that promote critical thinking and innovation. Such learning experiences help develop adaptability, creativity, and interdisciplinary awareness—skills that are increasingly essential in modern professional environments.

Multi-Disciplinary Minor courses also play an important role in fostering intellectual flexibility among students. By interacting with concepts from diverse academic fields, learners develop the ability to connect ideas, integrate knowledge, and apply learning in varied contexts. This exposure strengthens their readiness to work in multidisciplinary teams and address complex societal and technological challenges.

The following pages present the syllabi and structure of the Multi-Disciplinary Minor courses offered as part of the academic curriculum at DBIT. The document outlines the objectives, learning outcomes, and academic components associated with these courses. It serves as a reference for students and faculty members to understand the scope and academic expectations of the MDM framework.

The contents of this document may be reviewed and updated periodically by the Academic Council and other academic bodies of the institute in accordance with evolving academic practices and institutional priorities.

Through the Multi-Disciplinary Minor courses, DBIT aims to nurture engineers who possess not only strong disciplinary knowledge but also the interdisciplinary perspective required to contribute effectively to innovation, research, and societal development.

Web Development

Course Code	Course Name	Teaching Scheme (Contact Hours)			Credits Assigned			
		Theory	Pract.	Tut.	Theory	Pract.	Tut.	Total
25ILXX4MDM 01	Web Development	2	2		2	1		3

Course Code	Course Name	Examination Scheme							
		Theory					Term Work	Pract & oral	Total
		Internal Assessment			End Sem. Exam	Exam. Duration (in Hrs)			
		CA 1	CA2	MSE					
25ILXX4MDM 01	Web Development	10	10	30	50	–	25	-	125

Pre-requisite: 25FE1VSEC02 - Problem Solving using C programming.

Course and Module Overview: This course provides a structured and comprehensive introduction to web development, covering both client-side and server-side technologies along with database integration. The course begins with fundamental concepts of web communication, including clients, servers, HTTP protocols, and the structure of web applications. It then progresses to front-end development using HTML, CSS, and JavaScript, enabling students to design and develop interactive

and responsive web pages.

Further, the course introduces modern full-stack development using the **MERN stack (MongoDB, Express.js, React.js, Node.js)**. Students learn how different components of a web application interact, including user interface design, server-side scripting, API development, and database operations. The course emphasizes practical implementation, data flow analysis, and performance considerations, culminating in the development of a simple full-stack web application.

Module 1 Introduces fundamentals of web development including HTTP, clients, servers, HTML, and CSS. Helps students understand webpage structure and styling.

Module 2 Covers JavaScript, DOM, and event handling for adding interactivity. Enables dynamic content manipulation and basic form validation.

Module 3 Introduces MERN stack components and basics of Node.js and Express.Helps in understanding backend development and simple API creation.

Module 4 Focuses on React.js including components, props, state, and hooks.
Enables development of reusable and interactive user interfaces.

Module 5 Covers MongoDB operations and Express.js backend development.Helps in building APIs and connecting applications with databases.

Module 6 Integrates React, Node, Express, and MongoDB into a full-stack application. Provides hands-on experience in building and connecting complete web systems.

Overall, the course equips students with fundamental and practical skills in full-stack web development, enabling them to design, develop, and deploy basic web applications. It builds a strong foundation for advanced topics such as cloud computing, microservices, and scalable web architectures, and prepares students for internships and careers in software and web development.

Course Outcomes: On successful completion, of course, learner/student will be able to:

CO1	Identify basic concepts of web technologies, front-end tools, server-side components, databases, and the MERN stack.
CO2	Explain how web pages, scripts, servers, APIs, and databases work together in a complete web application.
CO3	Use HTML, CSS, JavaScript, React, Node.js, Express, and MongoDB to build functional parts of a web application.
CO4	Analyze data flow between frontend, backend, and database, and break down how different MERN components interact.

CO5	Evaluate web application performance, select suitable web technologies, and validate user input, API responses, and database operations.
CO6	Design and develop a simple full-stack MERN application integrating user interface, server logic, and database operations.

Module	Detailed Contents		Hours
1	**Introduction to Web Development**		**06**
	After completing this module, students will be able to: ● Describe basic web concepts including clients, servers, and communication protocols ● Explain HTTP request-response cycle and working of web applications ● Understand HTML structure and semantic elements for webpage design ● Apply CSS for styling web pages using different selectors and properties ● Analyze how web pages are structured and rendered in browsers		
	1.1	Web Essentials: Clients, Servers and Communication, The Internet, Basic Internet protocols, World wide web, HTTP Request Message, HTTP Response Message, Web Clients, Web Servers	
	1.2	HTML: fundamental syntax and semantics, Tables, Lists, Image, HTML5 control elements, Semantic elements, Drag and Drop.	
	1.3	**CSS3:** Inline, embedded and external style sheets – Rule cascading,Syntax, Inclusion, Color, Background, Fonts, Tables, lists, CSS3 selectors.	
Self Learning		HTML5 Audio Video controls	
2	**Front End Development**		**04**
	After completing this module, students will be able to:		

	● Describe basic concepts of JavaScript including variables, operators, and functions ● Apply conditional statements and functions to solve simple problems ● Explain the Document Object Model (DOM) and its role in web development ● Implement form validation using regular expressions ● Analyze and handle events to create interactive web pages		
	2.1	**Java Script:** Introduction to JavaScript: Variables, Operators, Conditional Statements, Functions.	
	2.2	Document Object Model- Introduction to the DOM, Defining the DOM,, Dom Tree, Simple form validation-Regular Expressions—patterns, flags, matching, validation applications.	
	2.3	Event Handling- Events, Fetch & Callbacks: Event Flow, Event Handlers/Listeners, The Event Object, Types of Events.	
Self Learning		Date Object - Getting and setting date & time values	
3	**Introduction to MERN Stack**		**05**
	After completing this module, students will be able to: ● Describe the components of the MERN stack and their roles ● Install and set up Node.js and MongoDB environments ● Explain the concept of npm and package management ● Develop basic server-side programs using Node.js ● Create simple APIs using Express and understand JSON data format		
	3.1	What is MERN-MongoDB, Express, React, Node, Installing Node.js & MongoDB	
	3.2	Understanding npm and packages, Basics of Node.js: running a simple JS program, Creating a very simple API using Express (GET request)	
Self Learning		Understanding JSON Data Format	
4	**Web Programming using React JS**		**05**

	After completing this module, students will be able to: ● Describe the fundamentals of React and its component-based architecture ● Develop UI components using React elements and components ● Explain and use state and props for dynamic data handling ● Implement event handling and hooks in React applications ● Display and manage lists of data in user interfaces		
	4.1	**React Framework:** Introduction to React JS, Components and Elements of React.	
	4.2	React State and Props, Handling events in React Events, Hooks Displaying list of items in UI.	
Self Learning		Passing props between components	
5	**MongoDB-Database Operations and Express.js**		**05**
	After completing this module, students will be able to: ● Describe NoSQL databases and their advantages over traditional databases ● Perform CRUD operations using MongoDB ● Connect MongoDB with Node.js applications ● Develop backend services using Express framework and routing ● Explain middleware and handle HTTP request–response objects		
	5.1	Introduction to NoSQL databases, CRUD operations in MongoDB Connecting MongoDB with Node.js.	
	5.2	Introduction to Express framework,Creating routes (GET, POST, PUT, DELETE),Middleware concept, Working with request & response objects	
Self Learning		Difference Between SQL and NoSQL Databases	
6	**MERN Integration**		**05**
	After completing this module, students will be able to:		

		• Explain the integration of frontend, backend, and database in MERN stack • Connect React applications with Express backend APIs • Perform basic CRUD operations using MongoDB • Fetch and display API data in React applications • Develop a simple full-stack MERN application	
	6.1	Connecting React frontend with Express backend, CRUD app using MERN, Simple MongoDB data read/write, Basic Create & Read operations, Fetching and displaying API data in React	
Self Learning		Testing APIs in Postman	

Textbooks:	
1	J. Duckett, *HTML and CSS: Design and Build Websites*, 1st edition, John Wiley & Sons, 2011.
2	J. Duckett, *JavaScript and JQuery: Interactive Front-End Web Development*, 1st edition, John Wiley & Sons, 2014.
3	M. Haverbeke, *Eloquent JavaScript: A Modern Introduction to Programming*, 3rd edition, No Starch Press, 2018.
4	A. Banks and E. Porcello, *Learning React: Modern Patterns for Developing React Apps*, 2nd edition, O'Reilly Media, 2020.
References:	
1	E. Brown, *Learning JavaScript*, 3rd edition, O'Reilly Media, 2016.
2	M. Casciaro and L. Mammino, *Node.js Design Patterns*, 3rd edition, Packt Publishing, 2020.
3	K. Chodorow, *MongoDB: The Definitive Guide*, 3rd edition, O'Reilly Media, 2019.

Experiments List: (Suggested)

1	Design a static web page using headings, paragraphs, lists, tables, images, and semantic elements. **Objective:** To identify and apply basic concepts of web technologies and use HTML elements to design a structured static web page. **Outcome:** Students will be able to create structured web pages using HTML and demonstrate understanding of fundamental web concepts and front-end components .
2	Create a web form using HTML5 input types and form elements. **Objective:** To understand how web pages collect user input and use HTML5 form elements to

	design interactive forms . **Outcome:** Students will be able to design forms that capture user data effectively and explain how form data is used in web applications.
3	Apply inline, internal, and external CSS for colors, fonts, layout, and basic responsiveness. **Objective:** To identify front-end styling techniques and apply CSS to improve layout, design, and responsiveness of web pages . **Outcome:** Students will be able to style web pages using different CSS methods and evaluate suitable styling approaches for better UI design.
4	Implement JavaScript programs using variables, operators, conditions, and functions. **Objective:** To understand scripting in web applications and implement JavaScript programs using basic programming constructs. **Outcome:** Students will be able to develop scripts that add dynamic behavior to web pages and explain how scripts interact with web content.
5	Implement mouse and keyboard events such as click, hover, and keypress on a web page. **Objective:** To analyze user interaction with web pages and implement event handling using JavaScript. **Outcome:** Students will be able to create interactive web applications by handling events and analyzing how user actions affect application behavior.
6	Validate user inputs using DOM manipulation and regular expressions. **Objective:** To evaluate input validation techniques and implement validation using DOM manipulation and regular expressions. **Outcome:** Students will be able to validate user inputs effectively and ensure data correctness in web applications.
7	Develop a basic server and implement GET and POST APIs using Express.js. **Objective:**

	To understand server-side components and APIs and develop a basic server using Express.js. **Outcome:** Students will be able to build server-side applications and explain how APIs handle client-server communication.
8	Perform Create and Read operations in MongoDB and connect the database with Node.js. **Objective:** To identify database concepts and perform Create and Read operations while connecting MongoDB with Node.js. **Outcome:** Students will be able to interact with databases, manage data, and analyze data flow between server and database.
9	Create a React application using components, props, state, and hooks to render data. **Objective:** To understand front-end frameworks and develop dynamic user interfaces using React components, props, state, and hooks . **Outcome:** Students will be able to build modular UI components and analyze how data flows within a React application.
10	Develop a simple full-stack MERN application integrating React frontend, Express backend, and MongoDB database. **Objective:** To design and integrate frontend, backend, and database components of a MERN stack application and analyze interaction between them. **Outcome:** Students will be able to develop a complete full-stack web application and evaluate the performance and integration of different MERN components.

Useful Links	
1	**Web Development Tutorials (HTML, CSS, JavaScript, React, Node, MongoDB)**: https://www.w3schools.com/
2	**MongoDB Tutorial (Beginner to Advanced)**: https://www.w3schools.com/mongodb/

3	**Express.js & Node.js Web Development Guide**: https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Server-side/Express_Nodejs/Introduction
4	**JavaScript Programming with React, Node & MongoDB (Coursera Specialization): https://www.coursera.org/specializations/javascript-programming-with-react-node-mongodb**

Course Code	**Course Name**	**Teaching Scheme (Hrs. / Week)**			**Credits Assigned**			
25ILIT4MDM01	**Data Structure and Algorithm**	**L**	**P**	**T**	**L**	**P**	**T**	**Total**
		2	1	-	2	1	-	**3**
		Examination Scheme						
			CA	**MSE**	**ESE**	**TW**	**OR**	**PR**	**Total**
		Theory	20	30	50	-	-	-	**100**
		Lab	-	-	-	25	-	–	**25**

Pre-Requisite Courses:	25FE1VESC02 - Problem Solving using C programming

Course and Module Overview:

The course Data Structure and Algorithm is designed to provide students with a strong foundation in organizing, managing, and processing data structure efficiently using appropriate data structures and algorithms. The course introduces fundamental concepts such as Abstract Data Types (ADTs), algorithm design principles, and complexity analysis, enabling students to evaluate algorithmic efficiency using asymptotic notations.Emphasis is placed on both linear and non-linear data structures, along with their practical implementations using arrays and linked representations. Students gain hands-on experience in applying stacks, queues, trees, graphs, searching, sorting, and hashing techniques to solve real-world computational problems.

Module 1: Introduction to Data Structures and Algorithms

This module introduces the basic concepts and classifications of data structures, highlighting the need for structured data organization in programming. Students explore primitive and non-primitive data structures, linear and non-linear structures, and Abstract Data Types (ADTs). The module also covers algorithm fundamentals, including properties, design techniques, and performance evaluation. A strong focus is placed on asymptotic notations (Big O, Omega, Theta) to analyze time and space complexity, enabling students to compare algorithm efficiency.

Module 2: Linear Data Structure – LISTS

This module focuses on Lists as an Abstract Data Type, covering both array-based and linked list

implementations. Students learn about different types of linked lists, including singly, doubly, and

circular linked lists. All fundamental operations such as insertion, deletion, traversal, merging, updating, and copying are studied along with their performance analysis. Practical applications like Polynomial arithmetic helps students understand real-life use cases of linked lists.

Module 3: Linear Data Structure – STACKS & QUEUES

This module introduces Stacks and Queues as ADTs, emphasizing their operations and implementations using arrays and linked lists. Students explore stack applications such as reversing data and expression conversion. Queue concepts include linear queues, circular queues, priority queues, and their implementations. The module also highlights expression evaluation techniques and double-ended queues, strengthening students' understanding of sequential data processing

Module 4: Non-Linear Data Structure – TREES

This module introduces Stacks and Queues as ADTs, emphasizing their operations and implementations using arrays and linked lists. Students explore stack applications such as reversing data and expression conversion. Queue concepts include linear queues, circular queues, priority queues, and their implementations. The module also highlights expression evaluation techniques and double-ended queues, strengthening students' understanding of sequential data processing

Module 5:Non-Linear Data Structure – GRAPHS

This module introduces graphs as a data structure for modeling networks and relationships. Students learn graph terminologies and representation methods using adjacency matrices and adjacency lists. Graph traversal techniques such as Breadth First Search and Depth First Search are studied along with applications like topological sorting. Real-world applications of graphs in networking are emphasized.

Module 6: Searching, Sorting & Hashing

This module focuses on efficient data retrieval and organization techniques. Students study linear and binary search algorithms, followed by basic sorting techniques such as selection sort, insertion sort, and bubble sort. The module also introduces hashing concepts, including hash functions, collision handling, and resolution techniques like chaining and open addressing. Advanced sorting techniques such as merge sort are included as self-learning topics.

Course	**After successful completion of the course, the students will be able to**	
	CO1	Identify fundamental concepts of data structures, algorithms, Abstract Data Types (ADTs), and asymptotic notations.
	CO2	Explain the functioning, implementation, and applications of linear and non-linear data structures, including arrays, linked lists, stacks, queues, trees, and graphs.
	CO3	Select appropriate data structures and ADT operations (insertion, deletion, traversal, etc.) to solve computational and real-world problems efficiently.

Outcomes	CO4	Analyze the time and space complexity of array-based and linked-list-based implementations, and evaluate algorithm performance using asymptotic techniques.
	CO5	Evaluate the searching, sorting, hashing, and expression processing techniques (infix, postfix, evaluation) using suitable data structures.
	CO6	Design and develop solutions for real-life problems using data structures .

Syllabus:

Module No.	Unit No.	Topics	Hours
1		**Introduction to Data Structures and Algorithms** After completing this module, students will be able to: 1. Explain the need for data structures and classify them into primitive and non-primitive types. 2. Distinguish between linear and non-linear data structures and their use cases. 3. Define Abstract Data Types (ADTs) and relate them to practical implementations. 4. Describe fundamental algorithm properties and common design techniques. 5. Analyze time and space complexity using asymptotic notations (Big-O, Ω, Θ). 6. Compare algorithms based on efficiency and performance metrics.	**05**
	1.1	Data Structures concepts: Definition, classification, and need for data structures. Types of data structures: primitive, non-primitive, linear, and non-linear, Abstract Data Types (ADT)	
	1.2	Concept of algorithms: properties, design techniques, and performance analysis.Asymptotic notation: Big O, Omega, Theta with examples	
		Self-Learning Topics: Self-Learning: Comparative growth analysis of functions	

2	**Linear Data Structure – LISTS**		
	After completing this module, students will be able to: 1. Describe Lists as an Abstract Data Type and explain their characteristics. 2. Implement array-based and linked list representations. 3. Differentiate between singly, doubly, and circular linked lists. 4. Perform fundamental operations such as insertion, deletion, traversal, merging, updating, and copying. 5. Analyze the performance of list operations in different implementations. 6. Apply linked lists to solve problems such as polynomial arithmetic.		
	2.1	List as an ADT, Array-based implementation, Linked List implementation.	
	2.2	Types of Linked List- Singly linked lists, doubly linked lists and circular linked lists.	
	2.3	All operations (Insertion, Deletion, Merge, Traversal, update, copying etc.) with singly linked lists, doubly linked lists and their analysis.	**04**
	Self-Learning Topics:Reversing a singly linked list, Applications of linked lists - Polynomial arithmetic		
3	**Linear Data Structure – STACKS & QUEUES**		
	After completing this module, students will be able to: 1. Explain Stacks and Queues as Abstract Data Types. 2. Implement stacks and queues using arrays and linked lists. 3. Apply stack operations to problems such as data reversal and expression conversion. 4. Describe different queue types including linear, circular, priority, and double-ended queues. 5. Implement expression evaluation techniques using stacks. 6. Analyze the efficiency of stack and queue operations in sequential data processing.		**06**
	3.1	Introduction to Stack, Stack as ADT, ADT Operations on Stack, Array and Linked List representation of Stack, Applications – Reversing data, Conversion of Infix to prefix and postfix expression.	

	3.2	Introduction to Queue, Queue as an ADT, operations on Queue, Implementation of Linear Queue, Circular and Priority Queue using arrays and Linked List.	
	Self-Learning Topics: Evaluation of postfix and prefix expressions, Double Ended Queue		
4	**Non-Linear Data Structure – TREES**		**05**
	After completing this module, students will be able to: 1. Define trees and explain basic tree terminologies. 2. Differentiate between various tree structures and representations. 3. Perform tree traversal techniques such as preorder, inorder, and postorder. 4. Implement binary trees and binary search trees. 5. Analyze the time complexity of tree operations. 6. Apply tree-based structures to solve hierarchical data problems.		
	4.1	Tree Terminologies, Tree as an ADT, Binary Tree - Operations, Tree Traversals, Binary Search Tree (BST) - Operations	
	Self-Learning Topics: AVL Tree, Applications		
5	**Non-Linear Data Structure – GRAPHS**		**03**
	After completing this module, students will be able to: 1. Explain graph concepts, terminologies, and applications. 2. Represent graphs using adjacency matrices and adjacency lists. 3. Implement graph traversal techniques such as Breadth First Search (BFS) and Depth First Search (DFS). 4. Apply graph algorithms to problems such as topological sorting. 5. Analyze the computational complexity of graph traversal algorithms. 6. Relate graph concepts to real-world applications such as networking and routing.		
	5.1	Graph Terminologies, Graph representation: adjacency matrix and list	
	5.2	Graph traversal: BFS, DFS with applications, Applications of Graphs -Topological sorting.	
	Self-Learning Topics: Graph applications in networking		
	Searching, Sorting & Hashing		

6	After completing this module, students will be able to: 1. Implement linear and binary search algorithms. 2. Compare searching techniques based on time complexity and applicability. 3. Implement basic sorting algorithms such as selection, insertion, and bubble sort. 4. Explain hashing concepts, hash functions, and collision resolution techniques. 5. Apply hashing methods such as chaining and open addressing. 6. Explore advanced sorting techniques like merge sort as a self-learning component.		**07**
	6.1	Searching:Linear Search and Binary Search: Concepts and Implementation	
	6.2	Sorting: Selection Sort, Insertion Sort, Bubble Sort	
	6.3	Hashing: Hash Functions, Overflow handling, Collision & Collision Resolution Techniques, Linear hashing, Hashing with chaining, Separate Chaining, Open Addressing.	
	Self-Learning Topics: Merge Sort.		
		TOTAL	**30**

Suggested List of Experiments:

Experiment No.	**Title of the Experiment (Perform any 10 of the following)**
1	Implementation of Insertion and deletion in a specific position in an Array using Function. **Objective:** To understand and implement insertion and deletion operations at specific positions in an array using functions. **Outcome:** Students will be able to perform and analyze array operations and understand their time complexity.

2	Implementation of recursive programs using functions. **Objective:** To understand recursion and implement basic recursive algorithms using functions. **Outcome:** Students will be able to design recursive solutions and compare them with iterative approaches.
3	Array Implementation of Stack **Objective:** To implement stack operations (push, pop, peek) using arrays. **Outcome:** Students will be able to use stack ADT and understand the LIFO principle in problem solving.
4	Array Implementation of Linear and Circular Queue. **Objective:** To implement creation and operations (insertion, deletion, traversal) on singly linked lists. **Outcome:** Students will be able to differentiate between linear and circular queues and apply FIFO principle efficiently.
5	Implementation of Singly Linked List

	Objective: To implement creation and operations (insertion, deletion, traversal) on singly linked lists. **Outcome:** Students will be able to dynamically manage data using linked list structures.
6	Implementation of Doubly Linked List **Objective:** To implement doubly linked lists and perform bidirectional traversal and operations. **Outcome:** Students will be able to efficiently perform operations using forward and backward links.
7	Implementation of Stack using Linked List **Objective:** To implement stack ADT using linked list representation. **Outcome:** Students will be able to overcome array limitations and implement dynamic stack operations.
8	Implementation of Binary Search Tree and Traversals **Objective:** To implement Binary Search Tree (BST) and perform traversal techniques (inorder, preorder, postorder). **Outcome:** Students will be able to organize hierarchical data and apply traversal methods.

9	Reversing a List using Stack **Objective:** To use stack for reversing elements of a list. **Outcome:** Students will be able to apply stack concepts in solving practical problems.
10	Infix to Postfix Conversion using Stack **Objective:** To convert infix expressions to postfix using stack ADT. **Outcome:** Students will be able to understand operator precedence and expression handling.
11	Evaluation of Postfix Expression using Stack **Objective:** To evaluate postfix expressions using stack operations. **Outcome:** Students will be able to implement expression evaluation algorithms.
12	Implementation of Deque using Linked List **Objective:** To implement a double-ended queue (deque) using linked lists. **Outcome:** Students will be able to perform insertion and deletion at both ends efficiently.

Text Books:

1. Reema Thareja, “Data Structures using C”, 3rd Edition, Oxford, 2023.
2. Ellis Horowitz, Sartaj Sahni and Susan Anderson-Freed, “Fundamentals of Data Structures in C”, 2nd Edition, W. H. Freeman and Company, 2008.
3. “Introduction to Algorithms” – Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (CLRS), 4th Edition (2022).

Reference Books:

1. “Data Structures and Algorithms Made Easy” – Narasimha Karumanchi, Latest (2022) – CareerMonk Publications.
2. “Data Structures and Algorithm Analysis in C++” – Mark Allen Weiss, 4th Edition.
3. Aaron M Tenenbaum, Yedidyah Langsam, Moshe J Augenstein, “Data Structures Using C”, Pearson Publication, 2nd edition, 2015.

Useful Links:

1. Use visualization tools like Visualgo or Pythontutor.
2. https://nptel.ac.in/courses/106/102/106102064/
3. Data Structure using C Programming - Course (swayam2.ac.in)

Assessment Methodology:

Type of Assessment	Assessment Tools
Continuous Assessment (CA) **(20 Marks)**	Certification: NPTEL (20 Marks) (Approved by instructor) **OR** Any 02 Pedagogies (10 marks each) ● MCQ /Class Test ● Case study/Assignment ● GATE based Assignment ● Certification Udemy/Coursera (Approved by instructor) ● Open Book Test ● Working model / Simulation of a course-based concept.
Mid Semester Examination (MSE) **(30 Marks)**	Question Paper Pattern is as follows: All Questions are compulsory. ● Q1 A or B - 10 marks ● Q2 A or B - 10 marks ● Q3 A or B - 10 marks ● For each question, A and B should be based on the same CO. ● MSE should be based on 50% syllabus. ● Time: 90 minutes (1 hour 30 minutes) ● Total Marks: 30

End Semester Examination (ESE) **(50 Marks)**	Question Paper Pattern is as follows: All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks • Q4 A or B - 10 marks • Q5 A or B - 10 marks • For each question, A and B should be based on the same CO. • ESE should be based on 30% syllabus of MSE and 70% syllabus after MSE. • Time: 120 minutes (02 hours) • Total Marks: 50
Term Work (25 Marks)	• Active Participation (Lab) = 05 marks • Laboratory Report / Journal = 10 marks • Laboratory Performance = 10 marks

The Bombay Salesian Society's

Don Bosco Institute of Technology, Mumbai

(An Autonomous Institute Affiliated to the University of Mumbai)

CURRICULUM STRUCTURE

LIBERAL LEARNING COURSES (LLC)

(As Per NEP 2020)

Scheme: DB25-V1

(w.e.f. AY 2025-2026)

Preface

Don Bosco Institute of Technology (DBIT), Kurla, Mumbai, presents the Liberal Learning Courses (LLC) as part of its academic curriculum under the autonomous framework aligned with the National Education Policy (NEP) 2020. The policy emphasizes holistic and multidisciplinary education that enables students to explore learning beyond the boundaries of their primary discipline. In line with this educational vision, Liberal Learning Courses form an important component of the undergraduate curriculum, providing opportunities for students to engage with diverse areas of knowledge and creativity alongside their engineering studies.

Engineering education today requires more than technical competence alone. It demands creativity, communication skills, adaptability, emotional intelligence, and the ability to appreciate perspectives beyond one's core discipline. Liberal Learning Courses aim to address this broader educational need by encouraging students to engage with creative, cultural, and aesthetic domains that enrich their intellectual and personal development.

The LLC framework provides students with opportunities to explore various forms of artistic expression, cultural practices, and creative activities. Such exposure enables learners to develop imagination, aesthetic sensibility, confidence, and collaborative skills. Participation in these courses encourages students to step beyond conventional classroom learning and discover new interests and abilities that contribute to their overall personality development.

The courses emphasize experiential learning through participative and activity-based approaches. Students learn through hands-on engagement, collaborative practice, creative projects, demonstrations, and peer interaction. These learning experiences foster teamwork, communication, leadership, and self-expression while cultivating respect for diverse cultural traditions and forms of artistic creativity.

In addition to promoting creativity and cultural awareness, Liberal Learning Courses contribute to students' emotional well-being and balance during their academic journey. Engaging in creative pursuits provides an avenue for expression and reflection, helping students develop resilience and maintain a healthy perspective amidst the demands of rigorous technical education.

The following pages present the syllabi and structure of the Liberal Learning Courses offered as part of the academic curriculum at DBIT. This document provides an overview of the course objectives, learning outcomes, and learning activities associated with these courses. It serves as a reference for students and faculty members to understand the scope and implementation of Liberal Learning Courses within the curriculum.

The contents of this document may be reviewed and updated periodically by the Academic Council and other academic bodies of the institute in accordance with evolving educational guidelines and institutional priorities. Feedback from students and faculty will continue to play an important role in strengthening the effectiveness and relevance of these courses.

Through the Liberal Learning Courses, DBIT aims to contribute to the development of engineers who are not only technically competent but also creative, culturally aware, confident, and socially responsible individuals capable of contributing meaningfully to society.

Course Code	Course Name	Teaching Scheme (Contact Hours)			Credits Assigned			
		Theory	Pract.	Tut.	Theory	Pract.	Tut.	Total
25IL4LLC01	Rhythm & Motion: A Journey Through Dance		2			1		1

	Course Name	Assessment Methods				Total Marks	Total Credits
		Mentor Assessment	Course Attendance	Euphoria Participation	Colosseum Participation		
	Rhythm & Motion: A Journey Through Dance	30	5	10	5	50	1

Course Objectives:

To introduce students to the basic elements and techniques of Indian & contemporary dance forms.

- To foster collaborative learning through peer-led instruction and group choreography.
- To enhance students' body rhythm, coordination, expression, and stage confidence through regular practice and performance.
- To provide a creative platform for self-expression, teamwork, and appreciation of cultural diversity through dance.

Course Outcomes:

- **CO1:** Identify and describe the basic elements and cultural context of selected Indian and contemporary dance forms.
- **CO2:** Perform foundational movements and rhythm patterns of at least one dance style with correct posture and coordination.
- **CO3:** Design and choreograph a short group dance performance using acquired skills and creativity.
- **CO4:** Engage in effective peer collaboration, contributing ideas, giving and receiving feedback, and working towards a shared goal.
- **CO5:** Document the learning process, including practice routines, group reflections, and performance insights in a learning log.
- **CO6:** Demonstrate confidence, stage presence, and expressive ability through a final group performance.

Sr. No.	Name of Module	Detailed Content	Hours
1	Foundations of Dance and Body Awareness	Understanding the role of dance in culture Importance of body posture, balance, and rhythm Warm-up techniques and movement preparation	02
2	Introduction to Indian and Contemporary Dance Forms	Basic steps and hand gestures (mudras) from Indian semi-classical styles Folk and contemporary forms (Garba, Bhangra, Bollywood freestyle, etc.) Practice and demonstration under teacher guidance	02
3	Peer Group Formation and Planning for Choreography	Formation of student groups Selection of dance form(s) for performance Setting group goals and distributing roles (lead, scribe, music, etc.)	02
4	Choreography, Practice & Feedback	Step-by-step choreography building through peer learning Weekly practice and feedback loops Focus on synchronization, formations, and transitions	12
5	Performance Rehearsal and Expression Techniques	Integration of expression (bhava), facial movements, and stage presence Full performance rehearsals Guidance and critiques from teacher-in-charge and peers	08
6	Final Performance and Reflective Practice	Group performances (3–5 minutes per group) Reflective presentations on the learning journey and group collaboration Submission of group logbooks and performance details	04

Suggested activities for Rhythm & Motion: A Journey Through Dance

1. Celebration of Culture
- Fusion of Indian folk dances from different states
- Depiction of festivals through dance (e.g., Holi, Navratri, Onam)

2. Unity in Diversity
- Blend of classical and contemporary forms (e.g., Bharatanatyam + Hip-hop)
- Represent different states/languages/cultures in a seamless performance

3. Seasons of Life
- Portray different stages: childhood, youth, maturity
- Express through changing moods and music tempos

4. Nature and Elements

- Themes like rain, sun, wind, or forest
- Use movement to express fluidity, calmness, or energy

5. Women Empowerment / Social Change

- Portray strength, transformation, or voice of change through expressive dance
- Depict social messages: education, freedom, equality

6. Friendship and Togetherness

- Dance to illustrate bonding, celebration, or emotional connection
- Use duets and group motifs creatively

7. Bollywood through the Decades

- Mix iconic dance styles and songs from 70s to present
- Highlight evolution of movement and costume

8. Patriotic Spirit / India through Dance

- Depict freedom movement, unity, or symbols of national pride
- Use flag colours, folk styles, or instrumental music

9. Time Travel in Dance

- Present past, present, and future through costumes, styles, and transitions
- Explore how dance evolves across time

10. Storytelling Without Words

- Select a theme like a journey, dream, or emotional arc
- Tell a story only through expression, posture, and movement

Introduction to Dramatics: Exploring Theatre Arts

Course Code	Course Name	Teaching Scheme (Contact Hours)			Credits Assigned			
		Theory	Pract.	Tut.	Theory	Pract.	Tut.	Total
25IL4LLC02	Introduction to Dramatics: Exploring Theatre Arts		2			1		1

	Course Name	Assessment Methods				Total Marks	Total Credits
		Mentor Assessment	Course Attendance	Euphoria Participation	Colosseum Participation		
	Introduction to Dramatics: Exploring Theatre Arts	30	5	10	5	50	1

Course Objectives:

- To introduce students to the fundamental elements of drama and theatre performance.
- To build confidence, voice modulation, and body language through theatrical expression.
- To encourage collaborative learning through peer-group script development and dramatization.
- To provide a platform for creativity, empathy, and reflective thinking through stage performance.

Course Outcomes: ***By the end of the course, students will be able to:***

- **CO1:** Identify key elements of theatre including character, dialogue, movement, and space.
- **CO2:** Demonstrate basic acting techniques such as voice projection, improvisation, and body language.
- **CO3:** Collaboratively develop and rehearse a short play or dramatic piece.
- **CO4:** Reflect on dramatic themes, character motivation, and audience engagement.
- **CO5:** Engage in peer learning by giving and receiving feedback during rehearsals.
- **CO6:** Perform a scripted or devised scene on stage as part of a team.

Sr. No.	Name of Module	Detailed Content	Hours
1	Stagecraft and Self-Awareness	Theatre basics, warm-ups, body/voice awareness	02
2	Acting Essentials and Theatre Forms	Voice projection, improvisation, intro to theatre styles	04
3	Script to Stage: Forming Dramatic Teams	Peer group formation, script selection/creation	02
4	Rehearse, Reflect, Repeat	Blocking, dialogues, emotions, peer feedback	12
5	Character, Costume, and Confidence	Character work, stage elements, rehearsal polishing	06
6	Curtains Up: Performance and Reflection	Final performance + reflective presentations	04

Suggested Themes/Genres for Performance
Students may choose or create scenes around:

- Social issues (e.g., gender, education, environment)
- Adapted mythology or folk tales
- Short comedies or farces
- Emotional/dramatic scenes (1-act plays, monologues)

Course Code	Course Name	Teaching Scheme (Contact Hours)			Credits Assigned			
		Theory	Pract.	Tut.	Theory	Pract.	Tut.	Total
25IL4LLC03	Swaranjali: Introduction to Vocal Music (Singing)		2			1		1

	Course Name	Assessment Methods				Total Marks	Total Credits
		Mentor Assessment	Course Attendance	Euphoria Participation	Colosseum Participation		
	Swaranjali: Introduction to Vocal Music (Singing)	30	5	10	5	50	1

Course Objectives:

- To introduce students to the fundamentals of Indian vocal music.
- To build foundational skills in singing through practice of swaras, alankars, and simple compositions.
- To encourage collaborative learning, creativity, and confidence through group performances.
- To develop appreciation for musical expression as a form of self-exploration and emotional well-being.

Course Outcomes:

By the end of the course, students will be able to:

- **CO1:** *Identify and explain basic elements of Indian vocal music such as swaras, taal, and raag.*
- **CO2:** Demonstrate swara practice, pitch accuracy, and basic vocal exercises.
- **CO3:** Collaboratively learn and rehearse selected compositions in peer groups.
- **CO4:** Reflect on personal growth, voice improvement, and peer collaboration.
- **CO5:** Participate in a group musical presentation with proper rhythm and expression.
- **CO6:** Compose or creatively adapt a short group performance based on learned concepts.

Sr. No.	Name of Module	Detailed Content	Hours
1	Basics of Indian Vocal Music	Sound, pitch (swar), rhythm (taal), laya, introduction to saptak (scale), shruti, swaras.	02

2	Voice Culture & Alankars	Breathing, pitch practice, alankars (note patterns), vocal warm-ups, intro to raag-based practice.	04
3	Raag & Taal Practice	Simple raags like Bhupali/Yaman, Teen Taal, Dadra; clapping cycles, rhythm coordination.	02
4	Song Practice in Peer Groups	Group division; learning bhajans, folk songs, patriotic songs, or classical compositions.	12
5	Expression, Bhava & Presentation Skills	Understanding meaning, emotion (bhava), and improving stage confidence, posture, and projection.	06
6	Group Performance & Reflection	Final group performance (3–5 min); sharing experiences; submission of logbooks/journals.	04

Suggested list of activities for Swaranjali: Introduction to Vocal Music (Singing)

- "Raag Rang: Colors of Melody": A performance based on a single or combination of simple raags (e.g., Bhupali, Yaman), showcasing how mood and emotion can be conveyed through melody.
- "Voices of Unity": Group performance using patriotic or unity-based songs (e.g., Vande Mataram, Desh Mera Rangeela) to represent harmony and national spirit.
- "Bhakti & Bhava": Present devotional or bhajan-based compositions that emphasize expression (bhava), simplicity, and spiritual connection.
- "Folk Fusion": Blend two or more regional Indian folk songs (e.g., Rajasthani, Marathi, Bengali) with a shared rhythm or melody line to showcase cultural diversity.
- "Seasons in Song": A musical expression of seasons (spring, monsoon, winter) through selected compositions or original adaptations, using changes in tempo and pitch to reflect mood.
- "Swar Se Shanti" (Peace Through Voice): Soothing compositions or raag-based presentations that focus on calmness, wellness, and introspection—ideal for a soft, expressive performance.
- "Story Through Song": A mini musical narrative told through 2–3 linked compositions expressing a journey or message (e.g., struggle to peace, darkness to light).

Course Code	Course Name	Teaching Scheme (Contact Hours)			Credits Assigned			
		Theory	Pract.	Tut.	Theory	Pract.	Tut.	Total
25IL4LLC04	**Strings & Strokes: An Introduction to Musical Instruments**		**2**			**1**		**1**

	Course Name	Assessment Methods				Total Marks	Total Credits
		Mentor Assessment	Course Attendance	Cultural Fest Participation	Technical Fest Participation		
	Strings & Strokes: An Introduction to Musical Instruments	**30**	**5**	**10**	**5**	**50**	**1**

Course Objectives:

- To introduce students to the fundamentals of rhythm, melody, and musical notation.
- To provide hands-on learning in playing selected basic musical instruments.
- To foster collaborative learning through peer practice, ensemble formation, and group performance.
- To develop listening skills, coordination, and appreciation for different music cultures.
- To build self-confidence through stage performance and group expression.

Course Outcomes:

By the end of the course, students will be able to:

- **CO1:** Identify the basic components, history, and playing techniques of selected musical instruments.
- **CO2:** Demonstrate foundational skills in playing at least one melodic or rhythmic instrument.
- **CO3:** Interpret simple musical patterns and rhythms using basic notation or auditory learning.
- **CO4:** Collaboratively compose or practice a short ensemble performance in a peer-learning group.
- **CO5:** Reflect on their own learning journey and peer group experiences through documentation.
- **CO6:** Participate confidently in a group musical performance.

Sr. No.	Name of Module	Detailed Content	Hours
1	Musical Foundations: Sound, Rhythm & Melody	Introduction to sound, rhythm, pitch; clapping exercises, ear training	02
2	Instrument Basics: Form, Function & Playing Technique	Introduction to Instruments: tabla, djembe, harmonium, keyboard, flute, ukulele, etc. (Instruments will vary based on student's choice)	04
3	Group Formation & Instrument Selection	Peer grouping, instrument choice, practice plan	02
4	Practice, Play & Peer Learning	Guided group practice, simple compositions, internal peer reviews	12
5	Building an Ensemble: Sound & Synchrony	Coordination of parts, ensemble play, polishing performance	06
6	The Final Note: Performance & Reflection	Final group performance, presentation, peer feedback	04

Suggested Performance Themes

- "Rhythms of India" - A medley combining classical (e.g., tabla, harmonium), folk, or regional musical patterns from different parts of India.
- "Sounds Without Borders" - A fusion of instruments or rhythms inspired by global cultures—e.g., African djembe + Indian flute + Western ukulele.
- "Nature's Symphony" - Use instruments and sounds to depict elements of nature—rain, breeze, thunder, sunrise, birdsong.
- "Music & Mood" - A performance that explores different emotions—joy, calm, sadness, excitement—through changes in melody, rhythm, and tempo.
- "India in Harmony" - Celebrate unity in diversity through a piece that includes instruments and musical motifs representing India's multilingual, multicultural richness.
- "Cinematic Soundtrack" - Create a simple instrumental piece based on popular Indian or world cinema themes (e.g., folk version of a Bollywood classic or instrumental theme from a famous film).
- "Soulful Strings: Music for Peace" - A meditative or soft instrumental piece designed to calm, soothe, and create emotional connection.
- "Journey Through Time" - Showcase evolution in music—from folk to contemporary—by starting with traditional sounds and gradually shifting to modern ones.

Course Code	Course Name	Teaching Scheme (Contact Hours)			Credits Assigned			
		Theory	Pract.	Tut.	Theory	Pract.	Tut.	Total
25IL4LLC05	Traditional Rangolis of India		2			1		1

	Course Name	Assessment Methods				Total Marks	Total Credits
		Mentor Assessment	Course Attendance	Cultural Fest Participation	Technical Fest Participation		
	Traditional Rangolis of India	30	5	10	5	50	1

Course Objectives:

- To introduce students to the traditional art of Kolams and Rangolis from Tamil Nadu and other states of India.
- To understand the cultural, spiritual, and social significance of Rangolis.
- To develop skills in creating various types of Rangoli patterns.
- To appreciate the geometrical symmetry and aesthetic value of Rangoli designs.
- To encourage creativity and imagination in designing new Rangoli patterns.

Course Outcomes:

- CO1: Recall different types of Rangolis and the materials used to create them. (Remember)
- CO2: Explain the traditions and cultural significance behind Rangolis and Kolams. (Understand)
- CO3: Apply basic Rangoli techniques to create structured and larger designs. (Apply)
- CO4: Analyse the geometric patterns and symmetries present in various Rangoli forms. (Analyse)
- CO5: Evaluate different Rangoli styles based on creativity, symmetry, and theme relevance. (Evaluate)
- CO6: Create original Rangoli designs integrating traditional elements with innovative themes. (Create)

Syllabus

Sr. No.	Name of Module	Detailed Content	Hours
1	Introduction to Kolams	What are Kolams or Rangolis and their traditions, Kolams during festivals and religious functions, Kolams inside the puja room (Hridaya Kamalam, Aishwaryam), Health benefits of early morning Rangoli during Margasheersha.	04
2	Dot-Based Kolams	Basic square Kolams (3 dots to 10 dots), Dot Kolams with straight dots, Dot Kolams with interspread dots, Practice and creation of self-designed dot Kolam pattern.	06
3	Sikku Kolam (Twisted Loops)	Basic Sikku Kolams (3 dots to 7 dots), Sikku Kolams with straight dots, Sikku Kolams with interspread dots, Creation of self-designed Sikku Kolam pattern.	06
4	Kambi Kolam (Line Patterns)	Basic Kambi Kolam with 2–3 layers, Kambi Kolam with 4–6 layers, Kambi Kolam with more than 6 layers, Creation of self-designed Kambi Kolam pattern.	04
5	Flower Rangoli and Thematic Designs	Basic Rangoli patterns with flowers, Rangoli with themes, Rangoli during festivals, Creation of theme-based Rangoli design.	03
6	Kolams and Society	Community Kolams, Kolams during marriages, Kolams during Margasheersha month, Social and cultural relevance of Rangolis.	03

Suggested Activities for Traditional Rangolis of India Course

- Create Hridaya Kamalam or Aishwaryam Kolam.
- Design a 5-dot to 5-dot Kolam pattern.
- Create a Sikku Kolam using 7 dots.
- Design a Kambi Kolam with 6 layers.
- Create a sustainability-themed Rangoli.
- Design decorative border Kolams for community display.

Course Code	Course Name	Teaching Scheme (Contact Hours)			Credits Assigned			
		Theory	Pract.	Tut.	Theory	Pract.	Tut.	Total
25IL4LLC06	Foundations of Photography		2			1		1

	Course Name	Assessment Methods				Total Marks	Total Credits
		Mentor Assessment	Course Attendance	Cultural Fest Participation	Technical Fest Participation		
	Foundations of Photography	30	5	10	5	50	1

Course Objectives:

- To introduce students to the evolution and foundational concepts of digital photography.
- To develop proficiency in camera operations, exposure settings, and essential photographic gear.
- To enable students to understand and apply exposure principles using the exposure triangle (ISO, aperture, shutter speed).
- To provide hands-on experience in post-processing using Photoshop and Lightroom.
- To cultivate technical skills and creative vision in digital image-making.

Course Outcomes:

- CO1: Recall the evolution of photography, types of cameras, photographic styles, and basic digital photography concepts. (Remember)
- CO2: Explain camera components, lens types, sensors, and exposure settings used in photography. (Understand)
- CO3: Apply composition rules and exposure triangle principles to capture well-composed photographs. (Apply)
- CO4: Analyze lighting conditions, exposure settings, and subject positioning to make informed shooting decisions. (Analyse)
- CO5: Evaluate and refine photographs using Photoshop and Lightroom tools. (Evaluate)
- CO6: Create compelling photo compositions and digital stories integrating technical skills and creativity. (Create)

Sr. No.	Name of Module	Detailed Content	Hours
1	Introduction to Digital Photography	History of photography, Photographic styles, Rule of thirds, Basic DSLR settings, Necessary gears, Sensors and	05

		mirrors in cameras, Full frame vs crop sensor.	
2	Working with Your Camera	Commanding the mode dial, Lens structure and types, Prime lenses, cleaning lenses, Introduction to shutter speed, aperture and ISO, White balance, Introduction to light.	05
3	Balancing Light: The Exposure Triangle	Detailed understanding of shutter speed, ISO, and aperture, Practical application of exposure triangle, Managing lighting conditions for better image capture.	05
4	Photoshop – Interface and Post-Production	Opening files in Photoshop, Understanding the Photoshop interface, Basic post-production techniques, Color correction and exposure adjustments.	05
5	Photoshop – Tools and Editing Techniques	Different tools used in Photoshop, Image enhancement techniques, Cropping, retouching, layering basics.	05
6	Lightroom – Editing and Enhancement	Lightroom interface overview, Using filters to enhance photographs, Basic workflow for organizing and refining images.	05

Suggested Activities for Photography Course

- Rule of Thirds Challenge – Take 3 photos applying the rule of thirds; peer review and discuss composition.
- Photography challenge - Take photos in different types of photography explained.
- Sensor Showdown – Comparative presentation or discussion on full-frame vs. crop sensor
- DSLR Demo Day – Hands-on exploration of camera settings
- Clean It Right – video on safe lens cleaning
- Before & After Edits – Edit an image using subtle adjustments (brightness, contrast, hue) and present a before/after comparison.
- Mini Retouch Project – Use crop, clone, healing brush, and selection tools to improve an image.
- Creative Edits – Use various tools to apply a creative twist (like turning a daytime photo into night).
- Filter Fun – Apply 3 different Lightroom filters to one image and explain the effects.
- Mood Edit Challenge – Choose a photo and use Lightroom adjustments to convey a specific emotion (e.g., warmth, mystery)

Course Code	Course Name	Teaching Scheme (Contact Hours)			Credits Assigned			
		Theory	Pract.	Tut.	Theory	Pract.	Tut.	Total
25IL4LLC07	Tradition & Craft: Hands-On Indian Art		2			1		1

	Course Name	Assessment Methods				Total Marks	Total Credits
		Mentor Assessment	Course Attendance	Cultural Fest Participation	Technical Fest Participation		
	Tradition & Craft: Hands-On Indian Art	30	5	10	5	50	1

Course Objectives:

- To introduce students to diverse Indian traditional and contemporary art forms.
- To develop hands-on skills in craft techniques such as Tanjore, Lipan, Quilling, Clay Modeling, and DIY décor.
- To cultivate design thinking, creativity, and aesthetic sensibility among students.
- To promote sustainable and mindful crafting as a stress-relieving and enriching activity.
- To encourage integration of traditional craftsmanship with contemporary applications.

Course Outcomes:

- CO1: Recall the historical and cultural origins of traditional Indian art forms such as Tanjore and Lipan art. (Remember)
- CO2: Explain the tools, materials, and techniques used in Tanjore, Lipan, Quilling, and Clay Modeling crafts. (Understand)
- CO3: Apply techniques to create basic forms in clay modelling, quilling, mirror work, and Tanjore relief work. (Apply)
- CO4: Analyse differences between traditional and contemporary art styles and their contribution to modern décor. (Analyse)
- CO5: Evaluate the aesthetic and functional aspects of handmade art pieces for home décor. (Evaluate)
- CO6: Create a unique DIY home décor project by integrating multiple art forms learned in the course. (Create)

Sr. No.	Name of Module	Detailed Content	Hours
1	Introduction to Tanjore art	History and origin (Thanjavur, Tamil Nadu), Traditional themes and subjects, Tools and materials: MDF board/wooden board, chalk powder and gum mixture, gold foil, stones, brushes, Sketching and layout transfer	06

		techniques, Gesso (relief) work, Stone and gold foil application, Painting and finishing techniques.	
2	Lipan Art (Mud & Mirror Work)	Origin and cultural significance of Lipan art (Kutch, Gujarat), Tools and materials: clay/M-seal, mirrors, MDF/canvas base, adhesives, Basic motifs and symmetry patterns, Clay application techniques, Mirror embedding techniques, Painting and finishing methods.	06
3	Quilling Art – Paper Filigree	Introduction to quilling art and its applications, Tools and materials: quilling strips, slotted tool, board, glue, Basic coil techniques: tight coil, loose coil, teardrop, marquise, scrolls, Practice of shapes and patterns, Jewelry or decorative craft creation.	06
4	Creative Clay Modeling	Introduction to clay art and types of clay (natural, air-dry, polymer), Tools and safety practices, Basic clay techniques: rolling, pinching, coiling, slab method, joining techniques (scoring and slip), Creating miniature objects, Painting and surface finishing.	04
5	DIY Home Décor	Basics of home décor and design principles, Traditional vs modern décor elements, Wall art and hangings, Decorative lighting concepts, Tabletop and shelf décor ideas, Sustainable and recycled craft applications.	04
6	Creative Hands: Traditional Meets Contemporary	Integration of traditional crafts with modern aesthetics, Mixed-media approach combining Tanjore-style elements, Lipan patterns, Quilling, and Clay modelling, Design planning and execution of final DIY project, Peer feedback and refinement.	04

Suggested Activities for Tradition & Craft: Hands-On Indian Art Course

- Create a small Tanjore-style relief artwork on MDF board using gesso work and gold foil application.
- Design and execute a decorative Lipan art wall plaque incorporating clay work and mirror embedding.
- Craft a quilled jewellery set (earrings and pendant) using basic quilling shapes and proper finishing techniques.
- Create a clay nameplate or wall hanging using basic techniques like rolling, pinching, and coiling.
- Design a functional decorative item (wall art or tabletop décor) using recycled or sustainable materials.
- Develop a mixed-media DIY décor project integrating at least three techniques learned in the course.

Course Code	Course Name	Teaching Scheme (Contact Hours)			Credits Assigned			
		Theory	Pract.	Tut.	Theory	Pract.	Tut.	Total
25IL4LLC08	LLC – Sports & Fitness		2			1		1

	Course Name	Assessment Methods				Total Marks	Total Credits
		Mentor Assessment	Course Attendance	Cultural Fest Participation	Technical Fest Participation		
	LLC – Sports & Fitness	30	5	10	5	50	1

Course Objectives:

- To encourage active participation in indoor and outdoor sports for physical and mental well-being.
- To develop teamwork, leadership, and sportsmanship through group sports activities.
- To promote a healthy and stress-free lifestyle through regular physical activity.
- To create awareness about basic fitness practices and recreational sports.

Course Outcomes:

- CO1: Recall the importance of physical fitness and healthy lifestyle habits. (Remembering)
- CO2: Explain basic rules of selected indoor and outdoor games. (Understanding)
- CO3: Apply basic fitness exercises and warm-up routines. (Applying)
- CO4: Demonstrate teamwork and active participation in sports activities. (Analyzing)
- CO5: Evaluate personal fitness through continuous participation. (Evaluating)
- CO6: Develop lifelong interest in sports and fitness. (Creating)

Syllabus

Sr. No.	Name of Module	Detailed Content	Hours
1	Foundations of Physical Fitness & Team Formation	Introduction to Sports & Physical Fitness – Importance of sports, fitness components, warm-up and team formation.	05
2	Basic Fitness Conditioning & Injury Prevention	Basic Fitness & Conditioning – Stretching, endurance activities, light strength exercises, safety measures. (Hours: 04)	04
3	Skill Development in Indoor Sports	Indoor Sports Participation – Practice of badminton, table tennis, chess, carrom and basic rules. (Hours: 06)	06
4	Skill Development in Outdoor Sports	Outdoor Sports Participation – Practice of cricket, football, volleyball, basketball, kho-kho and kabaddi. (Hours: 06)	06
5	Team Dynamics, Leadership & Sports Ethics	Teamwork, Leadership & Sports Ethics – Role of captain, discipline, fair play, managing wins and losses. (Hours: 05)	05
6	Recreational Sports, Wellness & Lifelong Fitness	Recreational Sports & Wellness – Sports for stress management and semester-end sports activity. (Hours: 05)	05

Suggested Activities for the Course

- Weekly sports participation
- Team practice matches
- Basic fitness routine before games
- Inter-team matches
- Semester-end sports event