

Amrutvahini Sheti and Shikshan Vikas Sanstha's

AMRUTVAHINI COLLEGE OF ENGINEERING, SANGAMNER

Amrutnagar, Ghulewadi, Sangamner - 422608



(An Autonomous Institute Approved by AICTE and Affiliated to SPPU, Pune)

Curriculum Structure and Syllabus
of
Second Year B. Tech. Computer Engineering



(Effective from Academic Year 2026-27)

Sanjay

Chairman

BoS, Computer Engineering
Amrutvahini College of Engineering, Amrutnagar
Sangamner-422608



R. Ganesh

Chairman
Academic Council

Amrutvahini College of Engineering, Amrutnagar
Sangamner-422608

INDEX

1	Institute Vision	i
2	Institute Mission	i
3	Department Vision	i
4	Department Mission	i
5	EOMS Policy	ii
6	Knowledge and Attitude Profile	ii
7	Program Outcomes (POs)	iv
8	Program Education Objectives (PEOs)	v
9	Program Specific Outcomes (PSOs)	v
10	Preface	vi
11	Program Codes and Subject Code Policy	vii
12	Guidelines for Examination Scheme	viii
13	Syllabus Structure – Semester III & IV	x
14	Course Approval Summary	xii
15	Data Structures	1
16	Digital Logic and Microprocessors	5
17	Object Oriented Programming and Computer Graphics	9
18	Probability and Statistics	13
19	Economics for Engineer	18
20	Universal Human Values & Ethics	23
21	Data Structures Lab	27
22	Object Oriented Programming and Computer Graphics Lab	41
23	Community Engagement Project	51
24	Syllabus Structure – Semester IV	58
25	Applied Discrete Mathematics	59
26	Operating System	64
27	Database Management Systems	67
28	Embedded Systems & Internet of Things	71
29	Programming in Java	75
30	Modern Indian Language and Technical Writing	85
31	Entrepreneurship and Startup Fundamentals	91
32	Operating System Lab	95
33	Database Management Systems Lab	102
34	Sustainable Technology and Green Computing	107

Abbreviation

AEC - Ability Enhancement Course

AICTE - All India Council for Technical Education

CEP - Community Engagement Project

EEM - Entrepreneurship/Economics/Management Courses

MDM - Multidisciplinary Minor

MOOC - Massive Open Online Course

NEP - National Education Policy

NPTEL - National Programme on Technology Enhanced Learning

OE - Open Elective

PCC - Program Core Course

CCC – Cocurricular Course

PEO - Programme Educational Objectives

PSO - Program Specific Outcomes

SWAYAM - Study Webs of Active-learning for Young Aspiring Minds

UGC - University Grants Commission

VEC - Value Education Course

VSE - Vocational and Skill Enhancement Course

WK - Knowledge and Attitude Profile

CCE - Comprehensive and Continuous Evaluation

ISO - International Organization for Standardization

EOMS - Educational Organization Management System

Institute Vision

To create opportunities for rural students to become able engineers and technocrats through continual excellence in engineering education.

Institute Mission

Our mission is to create self-disciplined, physically fit, mentally robust and morally strong engineers and technocrats with high degree of integrity and sense of purpose who are capable to meet challenges of ever advancing technology for the benefit of mankind and nature.

We, the management, the faculty and staff, therefore promise to strive hard and commit ourselves to achieve this objective through a continuous process of learning and appreciation of needs of time.

Department Vision

To serve the needs of society through competent technical education, promoting Industry-Institute Interaction and develop competent and cultured computer engineers.

Department Mission

M1: The Mission of computer Engineering Department is to effectively develop and channelize creative talents of students to face challenges posed by ever- changing computing industry with enhancement in Industry-Institute interaction.

M2: We, the faculty & the staff of Computer Engineering department commit ourselves to inculcate good moral values and shape a strong academic background of the pupil so important to work towards the betterment of the society as able technocrats & responsible citizens.

ISO 21001:2018 - EOMS Policy

We, at AVCOE Sangamner, are committed to:

- Provide quality technical education to the rural students
- Provide conducive environment for all round development of the students.
- Develop as a premier institute for technical education & research as per the needs and expectations of all stake holders.
- Comply all applicable requirements.
- Develop students into technocrats having social responsibility.
- Keep abreast about the advances in technology and innovation.
- Enhance industry-institute interaction.
- Managing intellectual property.
- Maintain hygiene, health and safety for applicable requirements.
- Achieve continual improvement in academic and administrative processes.
- We shall strive to maintain Educational Organizational Management System.

Knowledge and Attitude Profile

A Knowledge and Attitude Profile (KAP), often represented as WK (Knowledge and Attitude Profile) in some contexts, is a framework or assessment tool used to evaluate an individual's knowledge and attitudes related to a specific area, topic, or domain.

WK1	A systematic, theory-based understanding of the natural sciences applicable to the discipline and awareness of relevant social sciences.
WK2	Conceptually-based mathematics, numerical analysis, data analysis, statistics and formal aspects of computer and information science to support detailed analysis and modelling applicable to the discipline.
WK3	A systematic, theory-based formulation of engineering fundamentals required in the engineering discipline.
WK4	Engineering specialist knowledge that provides theoretical frameworks and bodies of knowledge for the accepted practice areas in the engineering discipline; much is at the forefront of the discipline.
WK5	Knowledge, including efficient resource use, environmental impacts, whole- life cost, re-use of resources, net zero carbon, and similar Concepts, that supports engineering design and operations in a practice area.
WK6	Knowledge of engineering practice (technology) in the practice areas in the engineering discipline.

WK7	Knowledge of the role of engineering in society and identified issues in engineering practice in the discipline, such as the professional responsibility of an engineer to public safety and sustainable development.
WK8	Engagement with selected knowledge in the current research literature of the discipline, awareness of the power of critical thinking and creative approaches to evaluate emerging issues.
WK9	Ethics, inclusive behavior and conduct. Knowledge of professional ethics, responsibilities, and norms of engineering practice. Awareness of the need for diversity by reason of ethnicity, gender, age, physical ability etc. with mutual understanding and respect, and of inclusive attitudes.

Program Outcomes (POs)

PO1	Engineering knowledge	Engineering Knowledge: Apply knowledge of mathematics, natural science, computing, engineering fundamentals and an engineering specialization as specified in WK1 to WK4 respectively to develop to the solution of complex engineering problems.
PO2	Problem analysis	Problem Analysis: Identify, formulate, review research literature and analyze complex engineering problems reaching substantiated conclusions with consideration for sustainable development. (WK1 to WK4)
PO3	Design / Development of Solutions	complex engineering problems and design/develop systems/components/processes to meet identified needs with consideration for the public health and safety, whole-life cost, net zero carbon, culture, society and environment as required. (WK5)
PO4	Conduct Investigations of Complex Problems	Conduct investigations of complex engineering problems using research-based knowledge including design of experiments, modelling, analysis & interpretation of data to provide valid conclusions. (WK8).
PO5	Engineering Tool Usage	Create, select and apply appropriate techniques, resources and modern engineering & IT tools, including prediction and modelling recognizing their limitations to solve complex engineering problems. (WK2 and WK6)
PO6	The Engineer and The World	Analyze and evaluate societal and environmental aspects while solving complex engineering problems for its impact on sustainability with reference to economy, health, safety, legal framework, culture and environment. (WK1, WK5, and WK7).
PO7	Ethics	Apply ethical principles and commit to professional ethics, human values, diversity and inclusion; adhere to national & international laws. (WK9)
PO8	Individual and Collaborative Team work	Function effectively as an individual, and as a member or leader in diverse/multi-disciplinary teams.
PO9	Communication	Communicate effectively and inclusively within the engineering community and society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations considering cultural, language, and learning differences
PO10	Project Management and Finance	Apply knowledge and understanding of engineering management principles and economic decision-making and apply these to one's own work, as a member and leader in a team, and to manage projects and in multidisciplinary environments.
PO11	Life-Long Learning	Recognize the need for, and have the preparation and ability for i) Independent and life-long learning ii) Adaptability to new and emerging technologies and iii) Critical thinking in the broadest context of technological change. (WK8)

Program Educational Objectives (PEOs):

PEO1: Attainment of key principles and practices of computation, mathematics and basic principles of engineering to ensure that graduates are able to apply their software development skills in design and implementation of practical systems consisting of software and/or hardware components.

PEO2: Analyze real-life problems and impart science-based engineering education to develop professional skills that will prepare the students for immediate employment in the industry.

PEO3: Imbibe lifelong learning, professional and ethical attitude for embracing global challenges and make positive impact on environment and society.

Program Specific Outcomes (PSOs):

A graduate of the Computer Engineering Program will demonstrate-

PSO1: Professional Skills - The ability to understand, analyse and develop computer programs in the areas related to algorithms, system software, multimedia, web design, big data analytics, and networking for efficient design of computer-based systems of varying complexities.

PSO2: Problem-Solving Skills - The ability to apply standard practices and strategies in software project development using open-ended programming environments to deliver a quality product for business success.

PSO3: Successful Career and Entrepreneurship- The ability to employ modern computer languages, environments, and platforms in creating innovative career paths to be an entrepreneur, and a zest for higher studies.

Preface

It gives me immense pleasure to present the autonomous syllabus of the Second Year B.Tech. in Computer Engineering, introduced for the first time by the department. This curriculum has been carefully designed to meet the changing needs of industry, higher education, and emerging technologies while strengthening the academic foundation of students.

The syllabus emphasizes conceptual understanding, practical learning, skill development, innovation, and problem-solving abilities. It includes modern and industry-relevant subjects that encourage students to explore advanced areas of computing and interdisciplinary technologies. Special focus has been given to outcome-based education, hands-on learning, and project-oriented approaches to prepare students for professional challenges and lifelong learning.

I sincerely appreciate the valuable efforts of all faculty members, subject experts, and Board of Studies members who contributed to the successful framing of this curriculum. I am confident that this syllabus will provide students with strong technical knowledge, professional competence, and opportunities for overall growth in the field of Computer Engineering.

I extend my best wishes to all students and faculty members for the successful implementation of this autonomous curriculum.

Dr. S. K. Sonkar

Head, Department of Computer Engineering

Amrutvahini College of Engineering, Sangamner

(An Autonomous Institute affiliated to Savitribai Phule Pune University, Pune)

Program codes:

The institute has assigned a unique program code to each branch.

Table 1 – Program codes

Sr. No.	Program code	Code
UG Programs		
1	Computer Engineering	CO
PG Programs		
2	M. Tech Computer Engineering	MCO

Subject code nomenclature:

The following nomenclature demonstrates the subject codes:

- **R-Year** (of designing/revising syllabus) – **Program code** (as given in table 1) - **Course Type** (as per NEP) – **Course Number** (101 – FY B. Tech Subject 1, 201 - SY B. Tech Subject 1, 301 - TY B. Tech Subject 1, 401 - B. Tech Subject 1 etc.)

Guidelines for Examination Scheme:

The evaluation of students shall be based on continuous academic performance in a course based on Comprehensive Continuous Evaluation (CCE) and End Semester Examination (ESE).
Components of the Examination System:

1. Comprehensive Continuous Evaluation (CCE):

- CCE of 40 marks based on all the Units of course.
- To design a Comprehensive Continuous Evaluation scheme for a theory subject of 40 marks with the specified parameters, the allocation of marks and the structure can be detailed as follows:

Table 2: Allocation of marks & structure for 3-Credit Subject

Sr. No.	Examination	Conduction Marks	Marks	Coverage of Units
1	CCE –I (Class test – 1)	32	16	Units 1 & Unit 2 (8 Marks/Unit)
2	CCE –II (Assessment tools: Case Study, presentation, Open book test etc.)	32	16	Units 3 & Unit 4 (8 Marks/Unit)
3	CCE –III (Assessment tools: Assignments, seminar, quiz, course project, field visit etc.)	08	08	Unit 5

The distribution of marks across various assessment tools and their mapping to COs is detailed in the table below.

Table 3: Allocation of marks & pattern for 3-Credit subjects

Exam	Assessment	CO1	CO2	CO3	CO4	CO5
CCE1	Written Test (Units 1 & 2)	8	8	–	–	–
CCE2	Continuous Assessment*	–	–	8	8	–
CCE3	Continuous Assessment*	–	–	–	–	8
ESE	Written Examination	12	12	12	12	12
Total	100 Marks	20	20	20	20	20

Table 4: Allocation of marks & pattern for 2-Credit subjects

Exam	Assessment	CO1	CO2	CO3	CO4
CCE1	Written Test (Units 1 & 2)	10	10	–	–
CCE2	Continuous Assessment*	–	–	10	–
CCE3	Continuous Assessment*	–	–	–	10
ESE	Written Examination	15	15	15	15
Total	100 Marks	25	25	25	25

***Continuous Assessment:** The course instructor may use any suitable assessment tool such as Assignment, Quiz, Seminar, Case Study, Open Book Test, Course Project, Presentation, Field Visit, or Viva-Voce for CCE2 and CCE3.

Evaluation and Feedback:

- **Unit Test:** Student answer papers will be evaluated promptly, and constructive feedback will be provided highlighting strengths and areas for improvement.
- **Assignments / Case Study:** Student submissions will be assessed as per the given rubric, and meaningful feedback will be provided to help students understand their performance and scope for enhancement.
- **Seminar Presentation:** Students will be evaluated based on content accuracy, delivery, and responses during the Q&A session. Feedback will be shared on communication skills and conceptual understanding.
- **Open Book Test:** Evaluation will focus on the depth of analysis and the ability to apply concepts. Feedback will emphasize students' critical thinking and problem-solving skills.
- **Mini Project:** Students will be assessed on problem identification, design approach, implementation quality, and documentation. Innovation, teamwork, and practical application of concepts will be strongly considered.
- **Field Visit:** Evaluation will be based on the student's report and reflection, focusing on understanding of real-world practices and their relevance to the course outcomes. Marks will reflect clarity, insight, and active participation.

2. End Semester Examination (ESE):

The ESE is a compulsory examination for all students. The ESE examination will be conducted at the end of the semester by the institute Department of Examination and Evaluation (DEE). The ESE will be based on 100% syllabus (Unit 1 to Unit 5). The ESE shall be of 60 marks (12 Marks / Unit) conducted for all theory courses as mentioned in the programme structure.

Syllabus Structure

Course Code	Course Type	Name of the Subject	Teaching Scheme (Hrs./week)					Examination Scheme and Marks						Credit			
			TH	PR	TUT	SL	Total	CCE	ESE	TW	PR	OR	TOTAL	TH	PR	TUT	TOTAL
Semester-III																	
R25-CO-PCC-201	Program Core course - 1	Data Structures	3	-	-	45	90	40	60	-	-	-	100	3	-	-	3
R25-CO-PCC-202	Program Core course - 2	Digital Logic and Microprocessors	3	-	-	45	90	40	60	-	-	-	100	3	-	-	3
R25-CO-PCC-203	Program Core course - 3	Object Oriented Programming and Computer Graphics	3	-	-	45	90	40	60	-	-	-	100	3	-	-	3
R25-CO-MDM-204	Multidisciplinary Course-1	Probability and Statistics	3	-	-	45	90	40	60	-	-	-	100	3	-	-	3
R25-CO-OEC-205	Open Elective - 1	(To be chosen from OE-1 basket finalized at college level)	2	-	-	30	60	40	60	-	-	-	100	2	-	-	2
R25-CO-EEM-206	Entrepreneurship/Economics/Management course	Economics for Engineers	1	2	-	15	60	25	-	25	-	-	50	1	1	-	2
R25-CO-VEC-207	Value Education Course	Universal Human Values & Ethics	2	-	-	30	60	50	-	-	-	-	50	2	-	-	2
R25-CO-PCC-208	Program Core Course- Lab	Data Structures Lab	-	2	-	-	30	-	-	25	25	-	50	-	1	-	1
R25-CO-PCC-209	Program Core Course Lab / Multidisciplinary Course-Lab	Object Oriented Programming and Computer Graphics Lab	-	2	-	-	30	-	-	25	25	-	50	-	1	-	1
R25-CO-CEP-210	Community Engagement Project / Field Project	Community Engagement Project	-	4	-	-	60	-	-	50	-	-	50	-	2	-	2
			17	10	0	255	660	275	300	125	50	0	750	17	5	0	22

Semester-IV


Chairman
 BoS, Computer Engineering
 Amrutvahini College of Engineering, Amrutnagar
 Sangamner-422608


Chairman
 Academic Council
 Amrutvahini College of Engineering, Amrutnagar
 Sangamner-422608

Course Code	Course Type	Name of the Subject	Teaching Scheme (Hrs./week)					Examination Scheme and Marks						Credit			
			TH	PR	TUT	SL	Total	CCE	ESE	TW	PR	OR	TOTAL	TH	PR	TUT	TOTAL
R25-CO-PCC-211	Program Core course - 4	Applied Discrete Mathematics	3	-	-	45	90	40	60	-	-	-	100	3	-	-	3
R25-CO-PCC-212	Program Core course - 5	Operating System	2	-	-	30	60	40	60	-	-	-	100	2	-	-	2
R25-CO-PCC-213	Program Core course - 6	Database Management Systems	3	-	-	45	90	40	60	-	-	-	100	3	-	-	3
R25-CO-MDM-214	Multidisciplinary Course - 2	Embedded Systems & Internet of Things	2	-	-	30	60	40	60	-	-	-	100	2	-	-	2
R25-CO-OEC-215	Open Elective - 2	(To be chosen from OE-2 basket finalized at the college level)	2	-	-	30	60	40	60	-	-	-	100	2	-	-	2
R25-CO-VSC-216	Vocational Skill Course	Programming in Java	1	2	-	15	60	-	-	25	25	-	50	1	1	-	2
R25-CO-AEC-217	Ability Enhancement Course	Modern Indian Language and Technical Writing	1	2	-	15	60	-	-	50	-	-	50	1	1	-	2
R25-CO-EEM-218	Entrepreneurship/Economics/Management course	Entrepreneurship and Startup Fundamentals	2	-	-	30	60	50	-	-	-	-	50	2	-	-	2
R25-CO-PCC-219	Program Core Course- Lab	Operating System Lab	-	2	-	-	30	-	-	-	25	-	25	-	1	-	1
R25-CO-PCC-220	Program Core Course Lab	Database Management Systems Lab	-	2	-	-	30	-	-	-	25	-	25	-	1	-	1
R25-CO-VEC-221	Value Education Course	Sustainable Technology and Green Computing	2	-	-	30	60	50	-	-	-	-	50	2	-	-	2
			18	8	0	270	660	300	300	75	75	0	750	18	4	0	22



Sr. No.	Name of the course	Course Code	Syllabus Design Teams
Semester - III			
1	Data Structures	R25-CO-PCC-201	Dr. S. B. Bhonde - <i>SBB</i> Prof. K. U. Rahane - <i>KUR</i> Prof. S. N. Shingote - <i>SN</i>
2	Digital Logic and Microprocessors	R25-CO-PCC-202	Prof. Y. S. Gunjal - <i>YS</i> Prof. J. N. Ganthade - <i>JN</i> Prof. A. R. Nehe - <i>ARN</i>
3	Object Oriented Programming and Computer Graphics	R25-CO-PCC-203	Dr. G. D. Puri - <i>GDP</i> Prof. A. R. Nehe - <i>ARN</i> Prof. S. B. Phatangare - <i>SBP</i>
4	Probability and Statistics	R25-CO-MDM-204	Prof. K. U. Rahane - <i>KUR</i> Dr. A. S. Thorat - <i>AST</i> Dr. S. K. Sonkar - <i>SKS</i>
5	Financial Literacy and Investment Planning	R25-CO-OEC-205	Dr. M. A. Wakchaure - <i>MAW</i> Prof. Y. S. Gunjal - <i>YS</i> Dr. M. B. Gunjal - <i>MBG</i>
6	Economics for Engineers	R25-CO-EEM-206	Prof. P. A. Thorat - <i>PAT</i> Dr. M. A. Wakchaure - <i>MAW</i>
7	Universal Human Values & Ethics	R25-CO-VEC-207	Prof. R. F. Shaikh - <i>RFS</i> Prof. V. K. Abhang - <i>VKA</i>
8	Data Structures Lab	R25-CO-PCC-208	Prof. S. N. Shingote - <i>SN</i> Dr. S. B. Bhonde - <i>SBB</i> Prof. K. U. Rahane - <i>KUR</i>
9	Object Oriented Programming and Computer Graphics Lab	R25-CO-PCC-209	Prof. A. R. Nehe - <i>ARN</i> Dr. G. D. Puri - <i>GDP</i> Prof. S. B. Phatangare - <i>SBP</i>
10	Community Engagement Project	R25-CO-CEP-210	Prof. Y. A. Shinde - <i>YAS</i> Prof. A. N. Inamdar - <i>ANI</i> Prof. R. F. Shaikh - <i>RFS</i>

*Sonkar***Chairman**BoS, Computer Engineering
Amrutvahini College of Engineering, Amrutnagar
Sangamner-422608*R. Gunimhan***Chairman**Academic Council
Amrutvahini College of Engineering, Amrutnagar
Sangamner-422608



Sr. No.	Name of the course	Course Code	Syllabus Design Teams
Semester - IV			
1	Applied Discrete Mathematics	R25-CO-PCC-211	Prof. K. U. Rahane – <i>K. U. Rahane</i> Dr. A. S. Thorat – <i>A. S. Thorat</i> Dr. S. K. Sonkar – <i>S. K. Sonkar</i>
2	Operating System	R25-CO-PCC-212	Dr. D. R. Patil – <i>D. R. Patil</i> Prof. A. N. Inamdar – <i>A. N. Inamdar</i> Dr. M. B. Gunjal – <i>M. B. Gunjal</i>
3	Database Management Systems	R25-CO-PCC-213	Dr. R. S. Gaikwad – <i>R. S. Gaikwad</i> Dr. A. S. Thorat – <i>A. S. Thorat</i> Dr. S. R. Wakchaure – <i>S. R. Wakchaure</i>
4	Embedded Systems & Internet of Things	R25-CO-MDM-214	Dr. S. B. Bhonde – <i>S. B. Bhonde</i> Dr. S. R. Wakchaure – <i>S. R. Wakchaure</i>
5	Digital Governance and Information System	R25-CO-OEC-215	Dr. S. K. Sonkar – <i>S. K. Sonkar</i> Prof. Y. A. Shinde – <i>Y. A. Shinde</i> Prof. P. A. Thorat – <i>P. A. Thorat</i>
6	Programming in Java	R25-CO-VSC-216	Prof. V. K. Abhang – <i>V. K. Abhang</i> Dr. S. B. Bhonde – <i>S. B. Bhonde</i> Prof. K. U. Rahane – <i>K. U. Rahane</i>
7	Modern Indian Language and Technical Writing	R25-CO-AEC-217	Dr. S. B. Bhonde – <i>S. B. Bhonde</i> Prof. V. K. Abhang – <i>V. K. Abhang</i>
8	Entrepreneurship and Startup Fundamentals	R25-CO-EEM-218	Dr. M. B. Gunjal – <i>M. B. Gunjal</i> Prof. P. A. Thorat – <i>P. A. Thorat</i> Prof. S. N. Shingote – <i>S. N. Shingote</i>
9	Operating System Lab	R25-CO-PCC-219	Prof. A. N. Inamdar – <i>A. N. Inamdar</i> Dr. D. R. Patil – <i>D. R. Patil</i> Dr. M. B. Gunjal – <i>M. B. Gunjal</i>
10	Database Management Systems Lab	R25-CO-PCC-220	Dr. A. S. Thorat – <i>A. S. Thorat</i> Dr. R. S. Gaikwad – <i>R. S. Gaikwad</i> Dr. S. R. Wakchaure – <i>S. R. Wakchaure</i>

*S. K. Sonkar***Chairman**BoS, Computer Engineering
Amrutvahini College of Engineering, Amrutnagar
Sangamner-422608*P. A. Thorat***Chairman**Academic Council
Amrutvahini College of Engineering, Amrutnagar
Sangamner-422608



11	Sustainable Technology and Green Computing	R25-CO-VEC-221	Prof. K. U. Rahane – Dr. S. B. Bhonde – Prof. V. K. Abhang –
----	--	----------------	--

Dr. S. K. Sonkar
Head of Department

Dr. M. A. Venkatesh
Chairman, Academic Council

Chairman
BoS, Computer Engineering
Amrutvahini College of Engineering, Amrutnagar
Sangamner-422608



Chairman
Academic Council
Amrutvahini College of Engineering, Amrutnagar
Sangamner-422608

Semester - III



Program:		S. Y. B. Tech (Computer Engineering)		Semester:		III	
Course:		Data Structures		Course Code:		R25-CO-PCC-201	
Teaching Scheme (Hrs. / week)			Credits (TH)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
3	-	-	3	40	60	-	-
Course Description: This course introduces fundamental and advanced data structures used for efficient data organization, storage, and processing. It covers algorithmic foundations, linear and non-linear data structures, and core techniques such as searching, sorting, hashing, and traversal methods, enabling students to develop efficient and scalable computational solutions							
Course Relevance: Data Structures form the foundation of computer science and software development. The course equips students with essential problem-solving skills required for algorithm design, competitive programming, system development, and advanced computing domains such as artificial intelligence, machine learning, data analytics, and distributed systems.							
Prerequisite: Basic knowledge of programming concepts using C or Python, familiarity with variables, control structures, functions, and basic problem-solving techniques.							
Bridge Content: Revision of programming fundamentals, data types, control flow statements, functions, arrays, pointers or references, and basic recursion concepts to ensure uniform preparedness among learners.							
Course Objectives: • To introduce the fundamental concepts and classification of data structures and algorithms. • To develop the ability to analyze algorithm efficiency using time and space complexity measures. • To enable implementation of linear data structures and their applications. • To apply searching, sorting, and hashing techniques for efficient data processing. • To understand and use stack, queue, tree, heap, and graph data structures in solving computational problems.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome					Bloom's Level	
CO1	DESCRIBE fundamental data structure concepts and algorithm design principles.					2	
CO2	IMPLEMENT arrays and linked lists for solving basic computational problems					3	
CO3	APPLY searching, sorting, and hashing techniques to suitable problem scenarios					3	
CO4	DEMONSTRATE stacks and queues in recursion handling and scheduling applications.					3	
CO5	ANALYZE tree, heap, and graph data structures for hierarchical and network-based problems.					4	



Course Contents			
Unit 1	Introduction to Data Structures and Algorithmic Foundations	9 Hrs.	CO1
Data structures, data and data types, abstract data types (ADT), classification of data structures, primitive and non-primitive data structures, linear and non-linear data structures, static and dynamic data structures, algorithms and their characteristics, algorithm design principles, asymptotic notations (Big-O, Big-Ω, Big-Θ), time complexity, space complexity, recursion, advantages and limitations of recursion			
Exemplars / Practical Applications and usage: • Use of arrays and ADTs in data management systems • Application of linear and non-linear structures in scheduling and hierarchical data • Performance analysis of algorithms using asymptotic notations • Use of recursion in problems like factorial computation and tree traversal			
Unit 2	Linear Data Structures – Arrays and Linked Lists	9 Hrs.	CO2
Arrays, one-dimensional arrays, multi-dimensional arrays, array representation, array operations (traversal, insertion, deletion, searching), applications of arrays, linked lists, singly linked list, doubly linked list, circular linked list, linked list operations, applications of linked lists, comparison of arrays and linked lists			
Exemplars / Practical Applications and usage: • Arrays are used for storing structured and indexed data such as marks and matrices. • Linked lists are applied in dynamic data handling like playlists and browser navigation. • Circular linked lists support cyclic processes such as CPU scheduling. • Comparison of arrays and linked lists helps in selecting efficient data storage methods.			
Unit 3	Searching, Sorting and Hashing Techniques	9 Hrs.	CO3
Searching techniques, linear search, binary search, comparison of searching techniques, sorting techniques, bubble sort, selection sort, insertion sort, merge sort, quick sort, hashing, hash tables, hash functions, collision handling techniques, linear probing, chaining, and applications of hashing.			
Exemplars / Practical Applications and usage: • Searching techniques are used in contact lists, databases, and file systems. • Sorting techniques help organize data in ranking systems, reports, and transaction logs. • Hashing is widely applied in password storage, data indexing, and fast data retrieval systems. • Collision handling techniques ensure efficiency in large-scale data storage applications.			
Unit 4	Stack and Queue	9 Hrs.	CO4
Stack abstract data type, stack operations, stack implementation using arrays, stack implementation using linked lists, applications of stack, expression evaluation, infix to postfix conversion, recursion handling, queue abstract data type, queue operations, linear queue, circular queue, dequeue, priority queue, applications of queues			

**Exemplars / Practical Applications and usage:**

- Stacks are used in function calls, expression evaluation, and undo/redo operations.
- Recursion handling relies on stack memory during program execution.
- Queues are applied in task scheduling, printer spooling, and service request systems.
- Priority queues support emergency handling and CPU scheduling.

Unit 5	Trees, Heaps and Graphs	9 Hrs.	CO5
Trees, tree terminology, properties of trees, binary tree, binary search tree, tree traversals (inorder, preorder, postorder), applications of trees, heaps, min heap, max heap, heap operations, applications of heaps, graphs, graph terminology, graph representations, adjacency matrix, adjacency list, graph traversals, breadth first search (BFS), depth first search (DFS), applications of graphs			
Exemplars / Practical Applications and usage: • Trees are used in file systems, XML/HTML parsing, and hierarchical data representation. • Binary search trees support efficient searching in databases and indexing systems. • Heaps are applied in priority scheduling and shortest-path algorithms. • Graphs model networks such as social media, transportation, and communication systems.			
Textbooks: 1. Seymour Lipschutz, Data Structures, Schaum's Outline Series, McGraw-Hill Education, Latest Edition. 2. Ellis Horowitz, Sartaj Sahani, Susan Anderson-Freed, Fundamentals of Data Structures in C, Universities Press, Latest Edition. 3. Reema Thareja, Data Structures Using C, Oxford University Press, Latest Edition. 4. Mark Allen Weiss, Data Structures and Algorithm Analysis in C, Pearson Education, Latest Edition. 5. Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman, <i>Data Structures and Algorithms</i>, Pearson Education, Latest Edition.			
Reference Books: 1. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, Introduction to Algorithms, MIT Press, Latest Edition. 2. Robert Sedgewick, Kevin Wayne, Algorithms, Pearson Education, Latest Edition. 3. Yedidyah Langsam, Moshe J. Augenstein, Aaron M. Tenenbaum, Data Structures Using C and C++, Pearson Education, Latest Edition. 4. Mark Allen Weiss, Data Structures and Algorithm Analysis, Pearson Education. 5. Narasimha Karumanchi, Data Structures and Algorithms Made Easy, Career Monk Publications, Latest Edition.			
MOOCs Links and additional reading material: 1. Programming, Data Structures and Algorithms Using Python (NPTEL – IIT Madras) https://nptel.ac.in/courses/106/106/106106145 2. Data Structures and Algorithms Specialization (UC San Diego – Coursera) https://www.coursera.org/specializations/data-structures-algorithms 3. Algorithms, Part I & II (Princeton University – Coursera) https://www.coursera.org/specializations/algorithms			



4. Introduction to Algorithms and Data Structures (edX)

<https://www.edx.org/course/algorithms-and-data-structures>

Strength of CO-PO-PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	3	2	1	1	–	–	–	–	–	–	2	3	2	1
CO2	3	3	2	2	1	–	–	–	–	–	2	3	3	1
CO3	3	3	3	2	2	–	–	–	–	–	2	3	3	2
CO4	3	2	2	2	2	–	–	–	–	–	2	3	2	2
CO5	3	3	3	3	2	–	–	–	–	–	2	3	3	3
Avg.	3.0	2.6	2.2	2.0	1.8	–	–	–	–	–	2.0	3.0	2.6	1.8

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

Assessment Pattern

BT Level	CCE					ESE				
	CO1	CO2	CO3	CO4	CO5	CO1	CO2	CO3	CO4	CO5
	8	8	8	8	8	12	12	12	12	12
L1	√	√	√	√	√	√	√	√	√	√
L2	√	√	√	√	√	√	√	√	√	√
L3		√	√	√	√		√	√	√	√
L4	–	–	–	–	√	–	–	–	–	√
L5	–	–	–	–	–	–	–	–	–	–
L6	–	–	–	–	–	–	–	–	–	–

This course serves as a prerequisite for / maps with the following future courses:

This course provides a strong foundation for advanced computing subjects. The concepts of data structures, algorithms, and complexity analysis introduced in this course directly support future courses such as Advanced Data Structures, Design and Analysis of Algorithms, Database Management Systems, Operating Systems, Computer Networks, Software Engineering, and emerging areas like Machine Learning and Artificial Intelligence.

Job Mapping:

- Programmer Analyst Trainee
- Junior Software Engineer
- Associate Software Engineer
- Coding Associate
- Software Developer Trainee



Program:		S. Y. B. Tech (Computer Engineering)		Semester:		III	
Course:		Digital Logic and Microprocessor		Course Code:		R25-CO-PCC-202	
Teaching Scheme (Hrs. / week)			Credits (TH)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
3	-	-	3	40	60	-	-
Course Description: This course introduces the fundamental principles of digital logic and microprocessor systems used in electronic and computer-based applications. The course covers number systems, Boolean algebra, logic gates, combinational logic circuits, and sequential logic circuits for digital system design. It also includes the architecture, instruction sets, programming techniques, and interfacing concepts of Intel 8086 and Intel 80386 microprocessors.							
Course Relevance: The course helps students understand digital circuit operation and microprocessor-based system design, forming a strong foundation for advanced subjects in computer engineering. It develops the analytical and programming skills required for embedded systems, computer architecture, hardware interfacing, and real-world electronic system applications.							
Prerequisite: Basic Electrical and Electronics Engineering							
Bridge Content: -							
Course Objectives: • To analyze and simplify Boolean expressions using Karnaugh Maps (K-Maps) and other minimization techniques. • To study number systems and develop skills for design and implementation of combinational logic circuits. • To explore number systems and develop the ability to design and implement sequential circuits. • To acquire knowledge of the functional architecture, addressing modes, and assembly language programming of the Intel 8086 microprocessor. • To examine the internal organization, memory management, and instruction processing of the Intel 80386 microprocessor.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome						Bloom's Level
CO1	EXPLAIN number systems, Boolean algebra, and logic gates used in digital electronic systems.						2
CO2	APPLY combinational logic circuits such as adders, subtractors, multiplexers, and decoders.						3
CO3	DEMONSTRATE the sequential circuits including flip-flops, registers, and counters.						3
CO4	DESCRIBE the architecture, addressing modes, instruction set, and programming concepts of the Intel 8086 microprocessor.						2
CO5	ILLUSTRATE the architecture, memory organization, operating modes, and instruction processing of the Intel 80386 microprocessor.						2



Course Contents			
Unit 1	Minimization Techniques	9 Hrs.	CO1
Logic Design Minimization Technique -: Minimization of Boolean function using K-map (up to 4 variables) Representation of signed number- sign magnitude representation ,1's complement and 2's complement form, Sum of product and Product of sum form, Minimization of SOP and POS using K-map. Exemplars / Practical Applications and usage: - The concepts of Boolean function minimization and number representation are widely used in the design and optimization of digital systems. These techniques help reduce hardware complexity, improve speed, lower power consumption, and increase system efficiency - Applications of Boolean function minimization are Simplification of digital circuits in computers and embedded systems, Designing efficient control circuits for automation systems. Digital locks using logic gates.			
Unit 2	Combinational Logic Design	9 Hrs.	CO2
Code converter -: BCD, Excess-3, Gray code, Binary Code. Half- Adder, Full Adder, Half Subtractor, Full Subtractor, BCD adder, Multiplexers (MUX): MUX (IC 74153, 74151), Cascading multiplexers, Demultiplexers (DEMUX), Implementation of SOP and POS using MUX, DMUX Comparators (2 bit), Parity generators and Checker. Exemplars / Practical Applications and usage: - Adders and Subtractors are used in Arithmetic operations in computers and calculators, Arithmetic Logic Unit design. Half Adder and Full Adder perform binary addition. Half Adder and Full Adder, Half Subtractor and Full Subtractor are used in Computers, Mobile processors, Digital calculators. - Multiplexers (MUX) are used in Data selection and communication systems, Signal routing in processors and networks, Resource sharing systems. It is used in CPU data buses , Communication switching systems.			
Unit 3	Sequential Logic Design	9 Hrs.	CO3
Flip-Flop: SR, JK, D, T; Preset & Clear, Master-Slave JK Flip Flops, Truth Tables and Excitation tables, Conversion from one type to another type of Flip-Flop. Registers: SISO, SIPO, PISO, PIPO, Shift Registers, Bidirectional Shift Register, Ring Counter , Universal Shift Register Counters: Asynchronous Counter, Synchronous Counter, BCD Counter, Synchronous Sequential Circuit Design :Models- Moore and Mealy. Exemplars / Practical Applications and usage: - SR Flip-Flop are used to Stores binary information using Set (S) and Reset (R) inputs, Simple memory storage ,Control circuits, Switch debouncing . - JK Flip-Flop are used in Digital clocks, Binary counters . - T Flip-Flop are used in Toggle operations, Counter design. - Master-Slave JK Flip-Flop are used in Synchronous systems, Race-around condition elimination. Electronic Voting Machine (EVM).			
Unit 4	Introduction to 8086	9 Hrs.	CO4
Basic concepts of Microprocessor, evolution of microprocessor, Architecture of Intel 8086, Pin Configuration and Functional Blocks, Memory Segmentation, Addressing Mode, Instruction			



Set, Assembly Language Programming, Interrupt Structure.			
Exemplars / Practical Applications and usage: The Intel 8086 microprocessor is used in embedded systems, industrial automation, communication systems, and control applications.			
Unit 5	Intel 80386 Fundamentals	9Hrs	CO5
Architecture of Intel 80386, Operating Modes, Register Organization, Memory Management, Paging and Virtual Memory, Instruction Set Features, Applications of 80386 Microprocessor, Pipelining and Performance Enhancement, Introduction to Advanced Processor Technologies.			
Exemplars / Practical Applications and usage: The Intel 80386 microprocessor is used in computer systems, embedded applications, industrial automation, and control systems. Its 32-bit architecture supports multitasking, memory management, and efficient data processing in advanced computing applications.			
Textbooks:			
- Modern Digital Electronics by R.P.Jain, 4th Edition, ISBN 978-0-07-06691-16 Tata McGraw Hill - Digital Logic and Computer Design by Moris Mano, Pearson, ISBN 978-93-325-4252-5 - Digital Principles and Applications by Donald P. Leach and Albert Paul Malvino - Ramesh S. Gaonkar”, Microprocessor Architecture,” Programming and Applications with 8085”, Penram International, 2002 - Ajoy Kumar Ray, K. M. Bhurchandi,” Advanced Microprocessors and Peripherals” McGraw-Hill,2006 - Barry B. Brey,”Intel Microprocessors”, Pearson Education,2009.			
Reference Books:			
- John Yarbrough, —Digital Logic applications and Design, Cengage Learning, ISBN – 13: 978-81- 315-0058- D. Leach, Malvino, Saha, —Digital Principles and Applicationsl, Tata McGraw Hill, ISBN – 13:978-0-07-014170-4 - Anil Maini, —Digital Electronics: Principles and Integrated Circuitsl, Wiley India Ltd, ISBN:978- 81-265-1466-3. - Liu, Gibson, “Microcomputer Systems: The 8086/88 Family”, 2 nd Edition, PHI,2005. - Kenneth Ayala, “The 8086 Microprocessor: Programming & Interfacing the PC			
MOOCs Links and additional reading material:			
- Digital Circuits and Systems – Prof. S. Srinivasan, NPTEL https://nptel.ac.in/courses/117106086 - Digital Circuits – Prof. Santanu Chattopadhyay, SWAYAM-NPTEL https://onlinecourses.nptel.ac.in/noc20_ee70/preview - Microprocessors and Interfacing – Prof. Shaik Rafi Ahamed, IIT Guwahati, NPTEL https://archive.nptel.ac.in/courses/108/103/108103157/ 8086 Microprocessor Architecture (Hindi) – Bharat Acharya https://www.youtube.com/watch?v=DmwOSdwzZ3E - Microprocessor 8086 – YouTube Playlist https://www.youtube.com/playlist?list=PLgwJf8NK2e4oAeDid0hwuiol_RJds			

**Strength of CO-PO-PSO Mapping**

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	3	2	2	-	1	-	-	-	-	-	-	3	2	1
CO2	3	3	3	2	2	-	-	-	1	-	-	3	3	2
CO3	3	3	2	2	2	-	-	-	1	-	-	2	2	3
CO4	3	2	2	1	2	-	-	-	-	1	-	3	2	2
CO5	2	2	2	2	2	-	-	-	-	1	-	3	2	2
Avg.	2.8	2.4	2.2	1.75	1.8	-	-	-	1	1	-	2.8	2.2	2

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

Assessment Pattern

BT Level	CCE					ESE				
	CO1	CO2	CO3	CO4	CO5	CO1	CO2	CO3	CO4	CO5
	8	8	8	8	8	12	12	12	12	12
L1	√	√	√	√	√	√	√	√	√	√
L2	√	√	√	√	√	√	√	√	√	√
L3	—	√	√	—	—	—	√	√	—	—
L4	—	—	—	—	—	—	—	—	—	—
L5	—	—	—	—	—	—	—	—	—	—
L6	—	—	—	—	—	—	—	—	—	—

This course serves as a prerequisite for / maps with the following future courses:

This course serves as a foundation for advanced subjects related to digital electronics, computer hardware, and embedded system design. The concepts of digital logic, combinational and sequential circuits, and microprocessor architecture support further study in Computer Organization and Architecture, Embedded Systems, Microcontrollers, Digital System Design, VLSI Design, and Advanced Microprocessors. The course also provides the necessary background for hardware interfacing, robotics, automation, IoT applications, and embedded programming.

Job Mapping:

This course supports career opportunities in the fields of digital electronics, computer hardware, and embedded system design. The knowledge of digital logic and microprocessors is applicable to roles such as Embedded System Engineer, Hardware Design Engineer, Electronics Engineer, FPGA/VLSI Engineer, Microprocessor Programmer, Automation Engineer, and Technical Support Engineer. The course also provides foundational skills required in industries related to computer hardware, industrial automation, robotics, communication systems, and embedded applications.



Program:		S. Y. B. Tech (Computer Engineering)		Semester:		III	
Course:		Object Oriented Programming and Computer Graphics		Course Code:		R25-CO-PCC-203	
Teaching Scheme (Hrs./week)			Credits (TH)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
03	---	---	03	40	60	--	---
Course Description: The OOPCG course is designed to provide a comprehensive introduction to object-oriented programming methodologies and foundational computer graphics concepts.							
Course Relevance: The course description for Object-Oriented Programming and Computer Graphics (OOPCG) covers fundamental OOP principles and their application in creating software and implementing graphics algorithms. Its relevance lies in providing essential skills for modern software development, designing modular programs, and developing graphical applications.							
Prerequisite: Fundamentals of Programming Languages, Programming & Problem Solving							
Bridge Content: Revision of structure in C, OOP concepts in Python							
Course Objectives: 1. To explore & understand the principles of Object-Oriented Programming (OOP) 2. To learn how inheritance and virtual functions implement dynamic binding with polymorphism. 3. To learn how to design and implement exception handling in C++ programs. 4. To get familiar with the mathematical foundation of the concepts of computer graphics 5. To lay a foundation for computer graphics concepts and implementation.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome						Bloom's Level
CO1	APPLY fundamental programming constructs, object-oriented constructs in C++ for Implementing an application						3
CO2	SOLVE object-oriented problems for small systems involving multiple objects.						3
CO3	ANALYZE object-oriented features like Inheritance, Polymorphism, Dynamic binding, Exception handling, for implementing an application						4
CO4	IDENTIFY the basic terminologies of Computer Graphics and interpret the mathematical foundation of the concepts of computer graphics.						2
CO5	EXAMINE the concepts of windowing and clipping and apply various algorithms to fill and clip polygons.						3



Course Contents			
Unit 1	Introduction to OOP Concepts	9 Hrs.	CO1
Introduction to object-oriented programming, need of object-oriented programming, Fundamentals of object-oriented programming: Namespaces, objects, classes, data members, methods, messages, data encapsulation, data abstraction and information hiding, inheritance, polymorphism. Benefits of OOP, C++ as object-oriented programming language. C++ Programming C++ programming Basics, Data Types, Structures, Enumerations, control structures, Arrays and Strings, Class, Object, class and data abstraction, Access specifiers, separating interface from implementation. Functions- Function, function prototype, accessing function and utility function, Constructors and destructor, Types of constructor, Objects and Memory requirements, Static members: variable and functions, inline function, friend function. Exemplars / Practical Applications and usage: Case Study: Study of C++ invention by Bjarne Stroustrup			
Unit 2	Inheritance and Pointers	9 Hrs.	CO2
Inheritance- Base Class and derived Class, protected members, relationship between base Class and derived Class, Constructor and destructor in Derived Class, Overriding Member Functions, Class Hierarchies, Public and Private Inheritance, Types of Inheritance, Ambiguity in Multiple Inheritance, Virtual Base Class, Abstract class, Friend Class, Nested Class. Pointers: declaring and initializing pointers, indirection Operators, Memory Management: new and delete, Pointers to Objects, this pointer, Pointers Vs Arrays, accessing Arrays using pointers, Arrays of Pointers, Function pointers, Pointers to Pointers, Pointers to Derived classes, Passing pointers to functions, Return pointers from functions, Null pointer, void pointer. Exemplars / Practical Applications and usage: Case Studies: Know about Firefox and Thunderbird as one of the popular software developed using C++			
Unit 3	Polymorphism Exception Handling	9 Hrs.	CO3
Polymorphism- Introduction to Polymorphism, Types of Polymorphism, Operator Overloading-concept of overloading, operator overloading, Overloading Unary Operators, Overloading Binary Operators, Data Conversion, Type casting (implicit and explicit), Pitfalls of Operator Overloading and Conversion, Keywords explicit and mutable. Function overloading, Run Time Polymorphism Pointers to Base class, virtual function and its significance in C++, pure virtual function and virtual table, virtual destructor, abstract base class. Exception handling: fundamentals, Exception Types, Using try-catch, Multiple try-catch clauses, Nested try statements, throw, throws, finally, Built-in Exceptions. Exemplars / Practical Applications and usage: Case Study: Study about use of C++ SDK wrappers for Java and .net			
Unit 4	Graphics Primitives, Scan Conversion	9 Hrs.	CO4
Introduction: graphics primitives - pixel, resolution, aspect ratio, frame buffer. Display devices, applications of computer graphics, GPU, Shader and Shader Programming, OpenGL introduction Scan conversion: Line drawing algorithms: Digital Differential Analyzer (DDA), Bresenham.			



Circle drawing algorithms: DDA, Bresenham, and Midpoint.			
Exemplars / Practical Applications and usage: Case Study: OpenGL 3D Viewing Pipeline.			
Unit 5	Windowing, clipping and Animation	9 Hrs.	CO5
Polygons: Introduction to polygon, types: convex, concave and complex. Inside test. Polygon Filling: flood fill, seed fill, scan line fill. Windowing and clipping: viewing transformations, 2-D clipping: Cohen-Sutherland algorithm line Clipping algorithm, Sutherland Hodgeman polygon clipping algorithm. Animation: Introduction, Conventional and computer based animation, Design of animation sequences, Animation languages, Key- frame, Morphing, Motion specification.			
Exemplars / Practical Applications and usage: Case Studies - Study of any open source tools Unity/ Maya/blender			
Textbooks:			
1. Deitel, "C++ How to Program", 4th Edition, Pearson Education, ISBN:81-297-0276-2 2. Robert Lafore, "Object-Oriented Programming in C++ l", fourth edition, Sams Publishing, ISBN:0672323087 (ISBN 13: 9780672323089 3. S. Harrington, "Computer Graphics", 2ndEdition, McGraw-Hill Publications, 1987, ISBN 0 – 07–100472 – 6. 4. Donald D. Hearn and Baker, "Computer Graphics with OpenGL", 4thEdition, ISBN-13: 9780136053583			
Reference Books:			
1. Herbert Schildt, "C++-The complete reference" , Eighth Edition, McGraw Hill Professional, 2011, ISBN:978-00-72226805 2. Matt Weisfeld, "The Object-Oriented Thought Process", Third Edition Pearson ISBN-13:075-2063330166 3. E.Balagurusamy, "Object-Oriented Programming with C++", 7th edition, Graw-Hill Publication, ISBN 10: 9352607996 ISBN 13: 9789352607990 4. Cox Brad, Andrew J. Novobilski, "Object -Oriented Programming: An Evolutionary Approach" , Second Edition, Addison-Wesley, ISBN:13:978-020-1548341 5. Dr. Samit Bhattacharya, Computer Graphics, Oxford University Press, ISBN-13: 978-0-19-809619-1; ISBN-10: 0-19-809619-4. 6. D. Rogers, "Procedural Elements for Computer Graphics", 2ndEdition, Tata McGraw-Hill Publication, 2001, ISBN 0 – 07 – 047371 – 4.			
E-books:			
1. An Introduction to Object-Oriented Programming in C++ with Applications in Computer Graphics, Textbook,© 2001,2nd edition, https://www.springer.com/gp/book/9781852334505 2. Introduction to Computer Graphics, David J. Eck, Hobart and William Smith Colleges, Copyright Year: 2016,Publisher: David J. Eck, https://open.umn.edu/opentextbooks/textbooks/introduction-to-computer-graphics			

**MOOCs Links and additional reading material:**

1. Programming in Modern C++, IIT Kharagpur
<https://nptel.ac.in/courses/106105234>
2. Fundamentals of Object Oriented Programming , IIT Roorkee
https://onlinecourses.nptel.ac.in/e-learning/preview/noc25_cs34
3. Programming in C++, IIT Kharagpur,
https://onlinecourses.nptel.ac.in/e-learning/preview/noc21_cs02
4. Computer Graphics, IIT Guwahati,
https://onlinecourses.nptel.ac.in/e-learning/preview/noc20_cs90
5. Introduction to Computer Graphics, IIT Delhi
<https://nptel.ac.in/courses/106102065>

Strength of CO-PO-PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	3	2	3	2	-	-	-	-	-	-	3	3	2	2
CO2	3	2	3	2	-	-	-	-	-	-	3	3	3	2
CO3	3	2	3	2	-	-	-	-	-	-	3	3	3	2
CO4	3	2	2	2	-	-	-	-	-	-	3	3	3	2
CO5	3	2	2	2	-	-	-	-	-	-	3	3	3	2
Avg.	3	2	2.6	2	-	-	-	-	-	-	3	3	3	2

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

Assessment Pattern

BT Level	CCE					ESE				
	CO1	CO2	CO3	CO4	CO5	CO1	CO2	CO3	CO4	CO5
	8	8	8	8	8	12	12	12	12	12
L1	√	√	√	√	√	√	√	√	√	√
L2	√	√	√	√	√	√	√	√	√	√
L3	√	√	√	—	√	√	√	√	—	√
L4	—	—	√	—	—	—	—	√	—	—
L5	—	—	—	—	—	—	—	—	—	—
L6	—	—	—	—	—	—	—	—	—	—

This course serves as a prerequisite for / maps with the following future courses:

- OOMD
- Java Programming

Job Mapping

- Software Engineer/Developer
- System Software Engineer
- Game Developer
- Graphics Programmer/Engineer



Program:	S. Y. B. Tech (Computer Engineering)			Semester:	III		
Course:	Probability and Statistics			Course Code:	R25-CO-MDM-204		
Teaching Scheme (Hrs./week)			Credits (TH)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
3	-	-	3	40	60	-	-

Course Description:

This course introduces the fundamental concepts of probability theory and statistical methods required for engineering problem-solving and decision-making under uncertainty. The course covers probability laws, random variables, probability distributions, descriptive statistics, sampling theory, estimation, hypothesis testing, correlation, regression, and analysis of variance. Students will develop analytical and quantitative skills for solving real-world engineering and research problems.

Course Relevance:

Probability and Statistics provide the mathematical foundation for engineering analysis, scientific research, quality control, reliability assessment, and intelligent decision-making. The subject is highly relevant in:

- Computer Engineering
- Artificial Intelligence and Machine Learning
- Data Science
- Software Reliability Engineering
- Operations Research
- Quality Control
- Risk Analysis
- Scientific Research
- Business Analytics

Prerequisite:

Students should have basic knowledge of:

- Elementary Mathematics
- Algebra and Functions
- Basic Calculus
- Sets and Logic

Bridge Content:

To help students from diverse academic backgrounds, the following bridge topics will be covered:

- Basics of Mathematical Operations
- Introduction to Data and Information
- Basics of Graphs and Charts
- Introduction to Python Programming

**Course Objectives:**

The course aims to:

1. Understand the fundamental concepts of probability and statistics.
2. Apply probability laws and distributions to engineering problems.
3. Analyze and summarize data using descriptive statistical methods.
4. Perform statistical inference through estimation and hypothesis testing.
5. Apply correlation, regression, and variance analysis for data interpretation and prediction.

Course Outcomes: After learning the course, students will be able to:

CO	Course Outcome	Bloom's Level
CO1	APPLY basic probability concepts to solve engineering problems	2
CO2	USE random variables and probability distributions to model uncertainty.	3
CO3	ANALYZE datasets using descriptive statistical techniques.	3
CO4	PERFORM statistical inference using estimation and hypothesis testing methods.	4
CO5	APPLY correlation, regression, and ANOVA techniques for prediction and decision-making.	3

Course Contents

Unit 1	Fundamentals of Probability	9 Hrs.	CO1
--------	-----------------------------	--------	-----

Introduction to Probability, Need of Probability in Computer Engineering, Random Experiments and Outcomes, Sample Space and Events, Types of Events: Simple Event, Compound Event, Mutually Exclusive Events, Independent Events, Basic Probability Rules, Addition Rule of Probability, Multiplication Rule of Probability, Conditional Probability, Bayes' Theorem (Basic Concept and Applications), Probability in Decision Making and Prediction

Exemplars / Practical Applications and usage:

- Probability-based Decision Making in Artificial Intelligence Systems
- Risk assessment in software projects
- Medical diagnosis using Bayes' theorem
- Network failure probability estimation
- Routing Decisions in Computer Networks

Unit 2	Random Variables and Probability Distributions	9 Hrs.	CO2
--------	--	--------	-----

Introduction to Random Variables, Discrete and Continuous Random Variables, Probability Mass Function (PMF), Probability Density Function (PDF), Cumulative Distribution Function (CDF), Mean (Expectation), Variance and Standard Deviation, Introduction to Probability Distributions, Bernoulli Distribution, Binomial Distribution, Poisson Distribution, Normal Distribution, Applications of Probability Distributions in Computing and Data Analytics



Exemplars / Practical Applications and usage:			
- Modeling software defect occurrences using Poisson distribution. - Analysis of network packet arrivals and traffic patterns. - Performance evaluation of computer systems and servers. - Machine Learning data modeling using Normal distribution. - Quality analysis of software testing processes.			
Unit 3	Descriptive Statistics and Data Analysis	9 Hrs.	CO3
Data Collection and Classification, Frequency Distribution, Tabulation of Data, Graphical Representation of Data, Measures of Central Tendency: Mean, Median, Mode, Geometric Mean and Harmonic Mean, Measures of Dispersion. Quartiles and Percentiles, Coefficient of Variation, Measures of Skewness, Kurtosis, Interpretation and Applications of Descriptive Statistical Measures.			
Exemplars / Practical Applications and usage:			
- Student performance analysis - Population survey analysis - Examination result interpretation - Industrial production analysis - Customer satisfaction surveys			
Unit 4	Sampling Theory and Statistical Inference	9 Hrs.	CO4
Population and Sample, Sampling Techniques, Sampling Distributions, Central Limit Theorem, Point Estimation, Interval Estimation, Confidence Intervals, Hypothesis Testing, Type I and Type II Errors, z-Test, t-Test, Chi-Square Test, Applications of Statistical Inference in Decision Making and Data Analysis.			
Exemplars / Practical Applications and usage:			
- Product quality testing - Clinical trial analysis - Software performance evaluation - Industrial process monitoring - Research data interpretation			
Unit 5	Correlation, Regression and Analysis of Variance	9 Hrs.	CO5
Correlation Analysis, Karl Pearson Correlation Coefficient, Spearman Rank Correlation, Simple Linear Regression, Regression Lines, Introduction to Multiple Regression, Analysis of Variance (ANOVA), Statistical Decision Making, Applications of Correlation, Regression and ANOVA in Data Analysis and Predictive Modeling.			
Exemplars / Practical Applications and usage:			
- Sales forecasting - Temperature versus performance analysis - Software reliability prediction - Academic performance analysis - Trend forecasting			
Textbooks:			
S. C. Gupta and V. K. Kapoor, Fundamentals of Mathematical Statistics, Sultan Chand. - Jay L. Devore, Probability and Statistics for Engineering and the Sciences, Cengage.			



3. Sheldon Ross, A First Course in Probability, Pearson.
4. Ronald E. Walpole et al., Probability and Statistics for Engineers and Scientists, Pearson.
5. William Mendenhall, Robert Beaver, Barbara Beaver, Introduction to Probability and Statistics, Cengage.

Reference Books:

1. Douglas C. Montgomery and George C. Runger, Applied Statistics and Probability for Engineers, Wiley India.
2. Richard A. Johnson, Irwin Miller and John Freund, Probability and Statistics for Engineers, Pearson.
3. Murray R. Spiegel, John Schiller and R. Alu Srinivasan, Schaum's Outline of Probability and Statistics, McGraw Hill.
4. N. G. Das, Statistical Methods, McGraw Hill Education.
5. Hogg, McKean and Craig, Introduction to Mathematical Statistics, Pearson.

MOOCs Links and additional reading material:**NPTEL Courses**

1. Probability and Statistics - <https://nptel.ac.in/courses/111106112>
2. Introduction to Probability and Statistics - <https://onlinecourses.nptel.ac.in>
3. Statistical Inference - <https://nptel.ac.in/courses/111105090>
4. Probability Theory - <https://nptel.ac.in/courses/111106148>
5. Engineering Statistics - <https://nptel.ac.in/courses/111105091>

Coursera Courses

1. Introduction to Probability and Data - <https://www.coursera.org/learn/probability-intro>
2. Statistics with Python Specialization - <https://www.coursera.org/specializations/statistics-with-python>
3. Introduction to Statistics - <https://www.coursera.org/learn/stanford-statistics>
4. Basic Statistics - <https://www.coursera.org/learn/basic-statistics>

Strength of CO-PO-PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	3	2	1	1	2	-	-	-	-	-	1	3	2	1
CO2	3	2	2	1	2	-	-	-	-	-	1	3	2	1
CO3	3	2	3	2	3	-	-	-	-	-	2	3	2	2
CO4	2	3	2	3	2	-	-	-	-	-	2	2	3	2
CO5	2	3	2	3	2	-	-	-	-	-	2	2	3	2
Avg.	3	3	2	2	2	-	-	-	-	-	2	3	2	2

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)



Assessment Pattern										
BT Level	CCE					ESE				
	CO1	CO2	CO3	CO4	CO5	CO1	CO2	CO3	CO4	CO5
	8	8	8	8	8	12	12	12	12	12
L1	√	√	√	√	√	√	√	√	√	√
L2	√	√	√	√	√	√	√	√	√	√
L3	–	√	√	√	√	–	√	√	√	√
L4	–	–	–	√	–	–	–	–	√	–
L5	–	–	–	–	–	–	–	–	–	–
L6	–	–	–	–	–	–	–	–	–	–

This course serves as a prerequisite for / maps with the following future courses:

This course provides a strong foundation in probability theory, statistical analysis, sampling techniques, hypothesis testing, correlation, regression, and quantitative decision-making. It develops analytical and problem-solving skills required for modeling uncertainty, interpreting data, and making informed decisions in engineering and computing applications. The course serves as a prerequisite for advanced subjects such as Data Science, Machine Learning, Artificial Intelligence, Business Analytics, Operations Research, Research Methodology, Predictive Analytics, and Quantitative Decision Support Systems.

Job Mapping:

The knowledge gained from Probability and Statistics equips students with quantitative reasoning, statistical analysis, data interpretation, and decision-making skills required in engineering, research, business, and computing domains.

Graduates are suitable for roles such as:

- Data Analyst
- Statistical Analyst
- Business Analyst
- Operations Analyst
- Market Research Analyst
- Risk Analyst
- Data Visualization Associate



Program:		S. Y. B. Tech (Computer Engineering)		Semester:		III	
Course:		Economics for Engineer		Course Code:		R25-CO-EEM-206	
Teaching Scheme (Hrs./week)			Credits (TH+PR)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
1	2	-	1+1	25	-	25	-
Course Description : This course provides engineering students with fundamental economic principles and analytical tools for evaluating costs, markets, investments, and business decisions in technology-driven environments.							
Course Relevance: This course helps engineering students understand economic principles and apply them to technology, business, investment, and decision-making in professional practice.							
Prerequisite: Basic Mathematics							
Course Objectives: The course aims to: 1. Understand basic economic concepts and their applications in engineering and technology. 2. Apply cost analysis and demand forecasting for decision-making in IT and software projects. 3. Learn digital market structures, pricing strategies, and technology business models. 4. Use investment evaluation techniques for analyzing engineering and technology projects.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome					Bloom's Level	
CO1	EXPLAIN basic economic concepts related to engineering and technology					2	
CO2	APPLY cost analysis and demand forecasting in IT and software projects					3	
CO3	INTERPRET the impact of pricing strategies and business models on digital market performance.					4	
CO4	ANALYZE investment options and the impact of economic factors on technology businesses.					4	
Course Contents							
Unit 1	Fundamentals of Economics for Computer Engineers				3 Hrs.	CO1	
Introduction to Engineering Economics and its relevance to Computer Engineering, Basic Economic Concepts: Utility, Scarcity, Choice, Opportunity Cost, Economic Systems and Firm Objectives, Laws of Demand and Supply, Elasticity of Demand, Market Equilibrium and Price Determination, Time Value of Money: Present Value, Future Value, and Annuity (Basics).							
Exemplars / Practical Applications and usage: Understanding scarcity and opportunity cost helps engineers choose the best use of limited resources in projects.							



Unit 2	Cost Analysis and Decision-Making in IT Industries	4 Hrs.	CO2
Cost Concepts: Fixed, Variable, Marginal, and Total Costs ,Cost-Volume-Profit Analysis and Break-Even Analysis, Demand Forecasting Techniques for Software and IT Products, Decision Making under Uncertainty, Economies of Scale in Software Development and Cloud Computing, Cost Estimation Models in Software Projects			
Exemplars / Practical Applications and usage: Calculate project costs, forecast software demand, and perform break-even analysis for IT projects. Apply cost estimation and decision-making techniques to improve profitability in software and cloud-based businesses			
Unit 3	Economics of Digital and Software Markets	4 Hrs.	CO3
Market Structures: Perfect Competition, Monopoly, Monopolistic Competition, Oligopoly, Digital Economy and Platform-Based Business Models, Pricing Strategies for Software Products and SaaS Services, Product Life Cycle Costing for Software Systems, Basics of Game Theory and Strategic Behavior in Technology Markets, Open-Source vs. Proprietary Software: Economic Perspectives			
Exemplars / Practical Applications and usage: Analyze pricing strategies, digital business models, and market competition in software and SaaS industries. Compare open-source and proprietary software and study strategic decisions in technology markets.			
Unit 4	Investment Analysis and Macroeconomic Environment	4 Hrs.	CO4
Capital budgeting techniques such as Payback Period, NPV, and IRR for technology investment decisions. Macroeconomic indicators including GDP, inflation, and interest rates. Impact of economic policies on IT and digital industries. Intellectual Property Rights (IPR), innovation, and sustainability in technology businesses			
Exemplars / Practical Applications and usage: Evaluate technology investments using capital budgeting techniques and analyze the impact of economic factors on IT businesses. Study innovation, IPR, sustainability, and green computing practices for responsible technology development.			

List of Practical:			
Experiment No.1	Case Study on Scarcity, Choice, and Opportunity Cost	CO1	2 Hrs.
Objective: To understand the concepts of scarcity, choice, and opportunity cost using real-life examples. Associated Tasks: 1. Identify limited resources in daily life or industries. 2. Compare different choices in engineering situations. 3. Analyze opportunity cost with simple examples. Exemplars and Utility: Helps students understand decision-making and efficient use of resources in engineering projects and industries.			
Experiment No.2	Case Study on Production Process and Factors of Production	CO1	2 Hrs.

**Objective:**

To understand the production process and the role of land, labor, capital, and entrepreneurship.

Associated Tasks:

1. Identify factors of production in an organization or industry.
2. Analyze the production process of a selected product or service.
3. Study the role of entrepreneurs in managing production activities.

Exemplars and Utility:

Provides practical insight into resource utilization and production management.

Experiment No.3	Case Study on Cost Analysis of a Software or Manufacturing Project	CO2	2 Hrs.
------------------------	---	------------	---------------

Objective: To calculate and analyze various costs involved in project execution.

Associated Tasks:

1. Identify fixed and variable costs.
2. Prepare a cost sheet for the selected project.
3. Analyze cost components and their impact on profitability.

Exemplars and Utility:

Helps students understand project budgeting and financial planning.

Experiment No.4	Case Study on Demand Forecasting for a Technology Product or Service	CO2	2 Hrs.
------------------------	---	------------	---------------

Objective: To apply basic demand forecasting techniques using available data.

Associated Tasks:

1. Collect historical sales or user data.
2. Perform trend analysis.
3. Forecast future demand and interpret results.

Exemplars and Utility:

Supports business planning, inventory control, and product management.

Experiment No.5	Case Study on Market Structures in the Technology Industry	CO3	2 Hrs.
------------------------	---	------------	---------------

Objective: To analyze different market structures and their characteristics.

Associated Tasks:

1. Identify technology companies operating in different market structures.
2. Compare pricing and competition strategies.
3. Analyze market behavior and business performance.

Exemplars and Utility:

Develops understanding of competitive strategies and market dynamics

Experiment No.6	Case Study on Digital Economy and Platform-Based Business Models	CO3	2 Hrs.
------------------------	---	------------	---------------

Objective: To examine digital platforms and their business models.

Associated Tasks:

1. Study a selected digital platform.
2. Identify revenue generation mechanisms.
3. Compare platform-based and traditional business approaches.

Exemplars and Utility:

Enhances understanding of modern digital enterprises and innovation-driven markets.

Experiment No.7	Case Study on Investment Analysis of a Technology Project	CO4	2 Hrs.
------------------------	--	------------	---------------

Objective: To evaluate investment decisions using financial analysis techniques.

Associated Tasks:

1. Collect project investment and return data.
2. Calculate Payback Period and ROI.



3. Interpret investment feasibility.

Exemplars and Utility:

Supports informed investment and project evaluation decisions

Experiment No.8	Case Study on the Impact of Macroeconomic Factors on Technology Businesses	CO4	2 Hrs.
------------------------	---	------------	---------------

Objective: To analyze the influence of GDP, inflation, and unemployment on technology industries.

Associated Tasks:

1. Collect macroeconomic data from reliable sources.
2. Analyze recent economic trends.
3. Evaluate their impact on technology companies and engineering projects.

Exemplars and Utility:

Helps students understand the relationship between economic conditions and business performance.

Textbooks:

1. N. D. Vohra, Engineering Economics, McGraw Hill Education, New Delhi.
2. S. K. Bhattacharya, Engineering Economics and Costing, Pearson Education, New Delhi..

Reference Books:

1. K. Dewett and J. D. Varma, Elementary Economic Theory, Revised Edition, S. Chand & Company Ltd., New Delhi.
2. Suma Damodaran, Managerial Economics, 2nd Edition, Oxford University Press, New Delhi.
3. N. Gregory Mankiw, Principles of Economics, 9th Edition, Cengage Learning.
4. Dominick Salvatore, Managerial Economics in a Global Economy, 9th Edition, Oxford University Press.
5. Ramesh Singh, Indian Economy, Latest Edition, McGraw Hill Education, New Delhi.

MOOCs Links and additional reading material:

1. NPTEL – Managerial Economics – <https://nptel.ac.in/courses>
2. Introduction to Economics (Coursera) – <https://www.coursera.org/learn/microeconomics>
3. Principles of Economics (MIT OpenCourseWare) – <https://ocw.mit.edu/courses/economics>
4. Managerial Economics and Business Analysis (SWAYAM) – <https://swayam.gov.in>
5. Economics for Managers (edX) – <https://www.edx.org/learn/economics>
6. Microeconomics and Macroeconomics Fundamentals (Khan Academy) – <https://www.khanacademy.org/economics-finance-domain>
7. Digital Economy and Platform Business Models (FutureLearn) – <https://www.futurelearn.com>
8. Investment Analysis and Portfolio Management (NPTEL) – <https://nptel.ac.in/courses/110105146>
9. Macroeconomics for Business (Coursera) – <https://www.coursera.org/browse/business/economics>
10. OpenStax Economics Textbook – <https://openstax.org/details/books/principles-economics-3e>



Strength of CO-PO-PSO Mapping														
CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	2	1	–	–	–	–	–	–	–	–	1	1	–	–
CO2	2	2	1	–	1	–	–	–	–	–	1	1	1	–
CO3	2	2	2	–	1	–	–	–	–	–	2	–	1	–
CO4	2	2	1	1	–	1	–	–	–	–	2	–	–	1
AVG	2	2	1	1	1	1	–	–	–	–	2	1	1	1

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

Assessment Pattern				
BT Level	CCE			
	CO1	CO2	CO3	CO4
	6	6	6	7
L1	√	√	√	√
L2	√	√	√	√
L3	–	√	√	√
L4	–	–	√	√
L5	–	–	–	–
L6	–	–	–	–

This course serves as a prerequisite for / maps with the following future courses:

This course provides a foundation for advanced studies in Entrepreneurship Development, Project Management, Software Engineering, Business Analytics, Technology Management, Innovation and Startup Management, Financial Management, Operations Management, Strategic Management, Digital Business and E-Commerce, and Engineering Management. The knowledge gained also supports industry internships, product development activities, and capstone/major projects by enabling students to make informed economic and business decisions in technology-driven environments.

Job Mapping:

- Software Developer
- System Analyst
- IT Project Manager
- Business Analyst
- Technology Consultant
- Product Manager
- Entrepreneur
- Financial Analyst
- Operations Manager
- Data Analyst
- Digital Business Manager



Program:		S. Y. B. Tech (Computer Engineering)		Semester:	III		
Course:		Universal Human Values & Ethics		Course Code:	R25-CO-VEC-207		
Teaching Scheme (Hrs./week)			Credits (TH)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
2	-	-	2	50	-	-	-
Course Description: This course introduces the concept of human values and professional ethics through self-exploration and right understanding. It focuses on harmony within the individual, family, society, and nature. The course emphasizes ethical living, value-based professional practices, and sustainable, human-centered development.							
Course Relevance: The course helps future professionals integrate ethical values with technical competence, enabling responsible decision-making, sustainable development, and holistic well-being in personal and professional life.							
Prerequisite: Student Induction Program (SIP)							
Bridge Content: It supports students in making value-based decisions in personal life and professional practice.							
Course Objectives: 1. To develop understanding of universal human values and ethical living. 2. To promote harmony within self, family, society, and nature. 3. To encourage self-exploration and value-based decision making. 4. To inculcate professional ethics and social responsibility among students.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome					Bloom's Level	
CO1	DESCRIBE the significance of universal human values in personal and professional life.					2	
CO2	DEMONSTRATE harmony between self and body through self-regulation and healthy practices.					3	
CO3	APPLY human values to maintain harmonious relationships in family and society.					3	
CO4	ANALYZE professional and environmental responsibilities based on ethical and sustainable practices.					4	
Course Contents							
Unit 1	Introduction to Value Education				8 Hrs.	CO1	
Understanding Value Education: Need and significance of value education, Role of values in personal and professional life, Self-Exploration as the Process for Value Education: Meaning and process of self-exploration, Natural acceptance and experiential validation, Basic Human Aspirations: Continuous happiness and prosperity, Understanding fulfilment of human aspirations, Dimensions of Human Life: Right understanding, Relationship, Physical facilities, Current Scenario of Happiness and Prosperity: Challenges in modern society, Materialism and imbalance in life, Methods to Fulfil Human Aspirations: Value-based living, Holistic development approach.							

**Exemplars / Practical Applications and usage:**

Usage: Helps students distinguish between right and wrong in practical life and develop critical thinking based on values.

Unit 2	Harmony in the Human Being	7 Hrs.	CO2
---------------	-----------------------------------	---------------	------------

Human Being as Co-existence of Self and Body: Understanding ‘Self’ (I) and Body, Their relationship and functions, **Needs of the Self and the Body:** Distinguishing physical and psychological needs, Priority and balance among needs, **Body as an Instrument of the Self:** Proper utilization and care of the body, Importance of health and discipline, **Understanding Harmony in the Self:** Self-regulation, Inner harmony and clarity, **Harmony between Self and Body:** Integration of thoughts, behavior, and health, Emotional and physical well-being, **Program for Self-regulation and Health:** Healthy lifestyle practices, Value-based personal development.

Exemplars / Practical Applications and usage:

Usage: Helps students recognize that fulfilling only bodily needs does not ensure true happiness, fostering conscious decision-making.

Unit 3	Harmony in the Family, Society and Nature	7 Hrs.	CO3
---------------	--	---------------	------------

Family as the Basic Unit of Human Interaction: Importance of family in value development, Harmony and mutual fulfilment in relationships, **Values in Human-to-Human Relationships:** Affection, care, guidance, gratitude, reverence, and love, Justice in relationships, **Harmony in Society:** Social harmony and coexistence, Ethical social participation, **Understanding Harmony in Nature:** Relationship between human beings and nature, Ecological balance and sustainability, **Interconnectedness and Mutual Fulfilment among Four Orders of Nature:** Material order, Plant/bio order, Animal order, Human order

Exemplars / Practical Applications and usage:

Usage: Students experience and reinforce trust, which forms the foundation of healthy relationships.

Unit 4	Professional Ethics	8 Hrs.	CO4
---------------	----------------------------	---------------	------------

Professional Ethics in the Light of Right Understanding: Ethical responsibilities of professionals, Human values in engineering and technology, **Strategies for Transition towards a Value-based Life and Profession:** Value-based decision making, Ethical leadership and responsible citizenship, Code of Ethics for Computing professionals, Whistleblowing and Ethical responsibilities in the workplace, Ethical considerations in IT Project management.

Exemplars / Practical Applications and usage:

Usage: Students develop awareness of harmony in nature and to make ethical decisions by understanding the underlying principles rather than just following rules.

Textbooks:

1. R. R. Gaur, R. Asthana, G. P. Bagaria, 2023, A Foundation Course in Human Values and Professional Ethics, 3rd revised edition, UHV Publications, ISBN: 978-81-957703-7-3 (Printed Copy), 978-81-957703-6-6 (e-book).
2. R. R. Gaur, R. Asthana, G. P. Bagaria, 2023, Teachers’ Manual for A Foundation Course in Human Values and Professional Ethics, 3rd revised edition, UHV Publications, ISBN: 978-81-957703-5-9 (Printed Copy), 978-81-957703-0-4 (e-book).

Reference Books:

1. P. L. Dhar, R. R. Gaur, 1990, Science and Humanism, Commonwealth Publishers.



2. A. Nagaraj, 1999, Jeevan Vidya: Ek Parichaya, Jeevan Vidya Prakashan, Amarkantak
3. B. P. Banerjee, 2005, Foundations of Ethics and Management, Excel Books.
4. A. N. Tripathy, 2003, Human Values, New Age International Publishers.
5. E. G. Seebauer & Robert L. Berry, 2000, Fundamentals of Ethics for Scientists & Engineers, Oxford University Press
6. B. L. Bajpai, 2004, Indian Ethos and Modern Management, New Royal Book Co., Lucknow. Reprinted 2008.
7. M. Govindrajran, S Natrajan & V.S. Senthil Kumar, Engineering Ethics and Human Values, East- ern Economy Edition, Prentice Hall of India Ltd.

MOOCs Links and additional reading material:

1. Swayam Course on “Understanding Human Being Nature and Existence Comprehensively” by Dr. Kumar Sambhav, Director, UP Institute of Design (UPID), Noida. <https://onlinecourses.swayam2.ac>
2. NPTEL Course on “Exploring Human Values: Visions of Happiness and Perfect Society” by Prof. K. Sharma, Department of Humanities and Social Sciences, IIT Kanpur. <https://nptel.ac.in/courses/1>

Strength of CO-PO-PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	1	-	-	-	-	1	2	1	-	-	1	1	-	1
CO2	2	1	-	-	-	1	2	1	1	-	1	2	1	1
CO3	1	2	-	-	-	1	3	3	2	-	1	1	2	2
CO4	1	2	1	-	-	2	3	2	2	1	2	1	2	3
Avg.	1.25	1.67	1	-	-	1.25	2.50	1.75	1.67	1	1.25	1.25	1.67	1.75

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

Assessment Pattern

BT Level	CCE			
	CO1	CO2	CO3	CO4
	12	13	12	13
L1	√	√	√	√
L2	√	√	√	√
L3	–	√	√	√
L4	–	–	–	√
L5	–	–	–	–
L6	–	–	–	–



This course serves as a prerequisite for / maps with the following future courses:

This course serves as a foundation for future courses related to Professional Ethics, Human Resource Management, Leadership Development, Environmental Studies, Corporate Social Responsibility, and Entrepreneurship Development. It helps students develop ethical values, professional behaviour, emotional intelligence, teamwork, and social responsibility required for personal growth and successful professional careers.

Job Mapping:

- Social Impact Associate
- Corporate Communication Executive
- Ethical Compliance Associate
- Community Development Officer
- Public Relations Executive



Program:		S. Y. B. Tech (Computer Engineering)		Semester:		III	
Course:		Data Structures Lab		Course Code:		R25-CO-PCC-208	
Teaching Scheme (Hrs./week)			Credits (PR)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
-	2	-		-	-	25	25
Course Description: This laboratory course emphasizes hands-on implementation of fundamental data structures such as arrays, linked lists, stacks, queues, trees, and graphs. Through practical exercises, students develop strong problem-solving and programming skills necessary for efficient algorithm design and analysis.							
Course Relevance: This course provides a crucial foundation for understanding complex data organization and manipulation techniques, which are vital for advanced computer science topics and real-world software development. Mastery of these concepts enables students to write optimized code and solve computational problems effectively.							
Prerequisite: Basics of object-oriented programming, Basic knowledge of programming concepts using C or Python, familiarity with variables, control structures, functions, and basic problem-solving techniques.							
Bridge Content: This lab course offers practical experience with essential data structures like arrays, linked lists, stacks, and queues. It helps students connect theoretical knowledge with real programming implementation. Through hands-on exercises, students learn to design and analyze efficient data structures, building a strong foundation for advanced computer science topics and problem-solving skills							
Course Objectives: 1. To provide hands-on experience with basic and advanced data structures. – 2. To understand various data searching and sorting methods with pros and cons. 3. To apply Data Structures for real-world problem solving. 4. To strengthen logical thinking by implementing recursive and iterative solutions. 5. To promote good programming practices , including modularity and documentation.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome						Bloom’s Level
CO1	APPLY fundamental data structures, recursion, and basic algorithm analysis techniques to solve simple computational problems						3
CO2	IMPLEMENT basic operations on arrays and linked lists, including traversal, insertion, deletion, and searching.						3
CO3	DEMONSTRATE the use of recursion and stacks/queues in problem-solving and process scheduling.						3
CO4	ILLUSTRATE the use of stacks and queues in expression evaluation, recursion handling, and task scheduling.						3
CO5	ANALYZE the time and space efficiency of implemented data structure algorithms through empirical testing.						4
Course Contents							
Guidelines for Instructor’s Manual							
The instructor’s manual/Lab Manual is to be developed as a hands-on resource and reference.							



The instructor's manual needs to include prologue (about university/program/ institute/ department/ foreword/ preface), curriculum of course, conduction and Assessment guidelines, topics under consideration-concept, objectives, outcomes, set of typical applications/assignments/guidelines, references.

Guidelines for Laboratory/Term Work Assessment

Continuous assessment of laboratory work should be done based on overall performance and Laboratory assignments performance of student. Each Laboratory assignment assessment should be assigned grade/marks based on parameters with appropriate weightage. Suggested parameters for overall assessment as well as each Laboratory assignment assessment include timely completion performance, innovation, efficient codes, punctuality, and neatness.

Guidelines for Student's Laboratory Journal

The laboratory assignments are to be submitted by student in the form of journal. Journal consists of prologue, Certificate, table of contents, and handwritten write-up of each assignment (Title, Objectives, Problem Statement, Outcomes, software and Hardware requirements, Date of Completion, Assessment grade/marks and assessor's sign, Theory Concept in brief, algorithm, flowchart, test cases, Test Data Set (if applicable), mathematical model (if applicable), conclusion/analysis. Program codes with sample output of all performed assignments are to be submitted as softcopy.

As a conscious effort and little contribution towards Green IT and environment awareness, attaching printed papers as part of write-ups and program listing to journal may be avoided. Students' programs maintained on cloud or college server by Laboratory In-charge is highly encouraged. For reference one or two journals may be maintained with program prints at Laboratory for accreditation purpose.

Guidelines for Laboratory Conduction

The instructor is expected to frame the assignments by understanding the prerequisites, technological aspects, utility and recent trends related to the topic. The assignment framing policy needs to address the average students and inclusive of an element to attract and promote the intelligent students. The instructor may set multiple sets of assignments and distribute among batches of students. It is appreciated if the assignments are based on real world problems/applications. Encourage students for appropriate use of Hungarian notation, proper indentation, and comments. Use of open-source software is to be encouraged. In addition to these, instructor may assign one real life application in the form of a mini-project based on the concepts learned. Instructor may also set one assignment or mini-project that is suitable to respective branch beyond the scope of syllabus.

Set of suggested assignment list is provided in groups- A, B, C, D, and E. Each student must perform at least 11 assignments as at least 2 from group A, 2 from group B, 2 from group C, 2 from group D and 2 from group E and one mandatory mini project based on data structure.

All group assignments should be implemented in C++ without using built-in methods for major functionality of assignment.

Operating System recommended: - 64-bit Open-source Linux or its derivative

Programming tools recommended: - Open Source, Programming tool like G++ compiler, Eclipse IDE, or Visual Studio IDE.



Guidelines for Practical Examination			
Both examiners should set the problem statements together, and assessment should focus mainly on correct implementation. Supplementary questions may be asked to check deeper understanding and efficiency. A fair, transparent, and encouraging evaluation process will build student confidence and support a strong academic start.			
List of Practical:			
Group A: Perform any 2 experiments.			
Experiment No. A1	Abstract Data Types	2 Hrs.	CO1
Create a Student ADT in C++ to manage student details and compare two students based on their marks.			
Objectives: - To understand the concept of Abstract Data Types (ADT) using C++ classes. - To implement a Student ADT with encapsulation and appropriate data types. - To perform basic operations such as creating, displaying, and updating student data. - To compare two Student objects based on their marks.			
Associated Task: - Define a student class representing the Student ADT. - Declare the data members: rollNo, name, and marks. - Implement the createRecord() function to initialize student details. - Implement the displayRecord() function to print student details. - Implement the updateMarks() function to modify the marks of a student. - Implement the compareStudents() function to compare two students based on marks. - In main (), create at least two Student objects. - Call the functions to create, display, update, and compare student records.			
Exemplars: - Create two Student objects: s1 and s2 - Input their roll number, name, and marks - Display both records - Update marks of one student - Compare marks of s1 and s2 and show who scored more			
Utility: Helps students understand ADT and encapsulation in real-world scenarios.			
Experiment No. A2	Recursive function	2 Hrs.	CO1
Write a recursive function to find the factorial of a number (e.g factorial of 5 = 120).			
Objective: - To understand the concept of recursion where a function calls itself. - To observe how problems can be broken into smaller subproblems. - To analyse the efficiency of recursive algorithms.			
Associated Task: - Define a function factorial(n) that calls itself until n == 1. - Identify and implement the base case (n = 0 or 1).			



- Test the function with different inputs.
- Observe the call flow (stack behaviour).

Exemplars:

Recursion is when a function calls itself to solve a problem step by step.

Factorial example:

- $5! = 5 \times 4!$
- $4! = 4 \times 3!$
- ... continues until base case $1! = 1$

This repeated self-calling continues until the smallest case is reached.

Utility:

- Recursion simplifies problems that have repetitive or nested structure (trees, graphs, puzzles).
- Helps understand stack memory and function call behaviour.
- Used in algorithms like DFS, backtracking, and divide-and-conquer methods.

Experiment No. A3	Static vs Dynamic Array Implementation	2 Hrs.	CO1
Write a C++ program to create a static array and a dynamic array of integers. Perform basic operations on both arrays such as inserting, deleting, and searching elements. Display the array after each operation and show how static and dynamic arrays differ in memory usage and flexibility. Objective: To understand static and dynamic arrays by implementing basic operations like insertion, deletion, and searching in C++. Associated Task: • Create a static array and a dynamic array. • Perform insert, delete, search, and display operations on both. • Compare both arrays in terms of memory, flexibility, resizing, and efficiency. Exemplars: A good implementation should: • Use proper static and dynamic array initialization • Perform insertion, deletion, and search with correct index handling • Show output clearly after each operation • Include basic error checking (wrong index, empty array, etc.) • Use meaningful comments and simple code structure Utility: • Helps students understand how memory is managed, how arrays work internally, and why dynamic arrays are more flexible than static ones			
Group B: Perform any 2 experiments.			
Experiment No. B1	Array	2 Hrs	CO2
Design a C++ program to perform basic operations on arrays. The program should first use a 1D array to insert, delete, search, and update elements. It should then use a two-dimensional array to store the marks of students in multiple subjects, display them in a table, and calculate			



each student's total marks and the average marks for each subject.

Objective:

- To understand and implement basic operations (insert, delete, search, update) using a 1D array.
- To use a two-dimensional array to store and display student marks and compute totals and averages.

Associated Task:

- Create a 1D array and perform basic operations such as insertion, deletion, search, and update.
- Display the final contents of the array after each operation.
- Create a two-dimensional array to store marks of students in multiple subjects.
- Input and display the marks in a simple table format.
- Calculate each student's total marks and the average marks for each subject.

Exemplars:

- A **1D array** stores elements in contiguous memory and supports operations like:
 - Insertion
 - Deletion
 - Search
 - Update
- A **2D array** represents data in rows and columns, useful for tabular information like:
 - Student marks across subjects
 - Displaying structured tables
 - Performing calculations on rows (totals) and columns (averages)

Utility:

Helps students understand how real-life tabular data (mark sheets, attendance tables, matrices) are stored and processed.

Experiment No. B2	Singly linked list	2 Hrs	CO2
Create a C++ program to implement a singly linked list with operations for insertion, deletion, traversal and searching using dynamic memory allocation.			
Objective: To Design and implement a program to perform various operations on a singly linked list , a dynamic linear data structure. The implementation should support insertion, deletion, traversal, and searching operations using dynamically allocated nodes and pointer manipulation. Students must analyse efficiency and justify the use of linked lists where dynamic storage is required.			
Associated Tasks: 1. Create a singly linked list using nodes. 2. Insert a node at: ➤ Beginning ➤ End ➤ Specific position 3. Delete a node from: ➤ Beginning			



- End
- Specific value/key
- 4. Traverse and display all nodes.
- 5. Search for a node containing a given value.
- 6. Analyse algorithmic efficiency.

Exemplars:

A **linked list** is a dynamic linear data structure made up of **nodes**, each containing:

- **Data**
- **Pointer** to the next node

Utility

- Suitable for applications where **frequent insertions and deletions** occur.
- Ideal for implementing stacks, queues, adjacency lists in graphs, and dynamic memory structures.

Experiment No. B3	Doubly Linked List	2 Hrs	CO2
Design and implement a Doubly Linked List in C++ that supports insertion, deletion, traversal (both forward and backward), and searching operations. The program should demonstrate effective pointer manipulation and allow comparison of the efficiency and flexibility of doubly linked lists with singly linked lists and arrays. Objective: To implement a Doubly Linked List in C++ with insertion, deletion, traversal, and searching operations. Associated Tasks: • Create a doubly linked list using dynamic memory. • Perform insertion at the beginning, end, and specific position. • Perform deletion from the beginning, end, or by key. • Traverse the list in forward and backward directions. • Search for a node by value. • Compare efficiency with a singly linked list Exemplars:. Structure: A doubly linked list stores data with pointers to both the next and previous nodes. Efficiency: • Traversal: $O(n)$ • Insertion/Deletion (with known pointer): $O(1)$ • Search: $O(n)$ Utility • Useful when bi-directional movement through elements is required. • Suitable for implementing: ➤ Browsing history (back/forward navigation) ➤ Undo/redo operations in editors			



Group C: Perform any 2 experiments.			
Experiment No. C1	Linear Search and Binary Search algorithms	2 Hrs	CO3
Write a C++ program to search for a specific customer account ID in a customer database. Implement both Linear Search and Binary Search algorithms to compare their performance and usability on different types of lists, demonstrating their practical applications in e-commerce systems. Objective: • To check if a given customer account ID exists in the database. • To compare how Linear Search and Binary Search perform on different lists. • To understand real-world uses of search algorithms, such as in e-commerce systems. Associated Task: • Create a list of customer account IDs (numeric or alphanumeric). • Implement Linear Search to sequentially check each ID. • Implement Binary Search on a sorted list to quickly locate an ID. • Take user input for the account ID to search. • Display appropriate output indicating whether the ID was found and which search method was used. • Compare both methods based on: ➤ Number of comparisons ➤ Efficiency ➤ Suitability for large datasets Exemplars: Example 1: Linear Search • Input: List = [102, 205, 309, 412, 578] Search ID = 309 • Process: Check each element one by one ; Found at position 3 • Output: “Customer ID 309 found using Linear Search.” Example 2: Binary Search • Input: Sorted List = [102, 205, 309, 412, 578] Search ID = 412 • Process: Compare mid → move to appropriate half → found • Output: “Customer ID 412 found using Binary Search.” Utility: • Helps authenticate customer accounts during login. • Validates account existence during checkout or service requests.			



• Demonstrates fundamental searching techniques used in databases and backend systems. • Shows efficiency differences: ➤ Linear Search : works on unsorted lists, simple but slower for large data ➤ Binary Search :very fast but requires a sorted list			
Experiment No. C2	Sorting	2 Hrs	CO3
Design a C++ program to sort employee salaries using both Selection Sort and Bubble Sort. The program should arrange the salaries in ascending order using each algorithm separately and then display the top five highest salaries from the sorted list. Objective: • To implement Selection Sort and Bubble Sort to arrange employee salaries in ascending order. • To identify and display the top five highest salaries after sorting using both algorithms. Associated Task: • Accept the number of employees and their salary values. • Implement Selection Sort to sort the salary list. • Implement Bubble Sort to sort the salary list. • Display the sorted list after each algorithm. • Extract and display the top five highest salaries. • Compare the results produced by both sorting methods. Exemplars: • Selection Sort: Repeatedly selects the smallest element and places it at the correct index. Time Complexity: $O(n^2)$ • Bubble Sort: Repeatedly compares adjacent elements and swaps them if needed. Time Complexity: $O(n^2)$ Utility: • Helps understand the working and differences between two basic sorting algorithms. • Useful in real-world scenarios like analysing employee salary data.			
Experiment No. C3	Hashing	2 Hrs	CO3
Develop a C++ program to implement a hash table of size 10 using the division method as the hash function. Handle collisions using chaining and support basic operations: Insert, Search, Delete, and Display key-value pairs. Objective: • To Implement a hash table of size 10 and use the division method as a hash function. In case of a collision, use chaining. Implement the following operations: ➤ Insert(key): Insert key-value pairs into the hash table. ➤ Search(key): Search for the value associated with a given key. ➤ Display (): Displays the hash table. ➤ Delete(key): Delete a key-value pair from the hash table. • To understand how hashing works using the division method hash function. • To implement collision handling using chaining. • To demonstrate efficient data storage and retrieval using a hash table. • To perform basic dictionary-like operations: Insert, Search, Delete, and Display.			

**Associated Task:**

- Create a hash table of size 10 using chaining (linked list or vector).
- Define hash function: $h(\text{key}) = \text{key} \% 10$.
- Insert key–value pairs by computing index and adding to the chain; handle collisions by appending.
- Search a key by computing its index and traversing the chain to find the value.
- Delete a key by locating and removing it from the chain.
- Display the hash table with all indices and stored chains.

Exemplars:**Example 1: Insert Operations**

Insert the following key-value pairs:

(15, “A”), (25, “B”), (35, “C”), (7, “D”)

Hash calculation:

$$15 \% 10 = 5$$

$$25 \% 10 = 5$$

$$35 \% 10 = 5$$

$$7 \% 10 = 7$$

Resulting Hash Table:

Index 5 $\rightarrow$ (15, A) $\rightarrow$ (25, B) $\rightarrow$ (35, C)

Index 7 $\rightarrow$ (7, D)

Example 2: Search Operation

Search for key = 25

$$25 \% 10 = 5$$

Check chain at index 5 $\rightarrow$ Found value “B”

Output:

Key 25 found with value B

Example 3: Delete Operation

Delete key = 25

Updated chain at index 5:

(15, A) $\rightarrow$ (35, C)

Example 4: Display Operation

0:

1:

2:

3:

4:

5: (15, A) $\rightarrow$ (35, C)

6:

7: (7, D)

8:

9:

Utility: Essential for learning hash-based structures such as unordered map, dictionaries, and hash sets.



Group D: Perform any 2 experiments.			
Experiment No. D1	Stack	2 Hrs.	CO4
Design a C++ menu-driven program to implement a Stack using an array. The program should allow users to perform operations such as push, pop, peek, and display. This will help students understand LIFO behavior and stack operations through interactive execution. Objective: - To implement Stack ADT using arrays. - To perform push, pop, peek, and display operations. - To learn LIFO order with real-time interaction. - To handle overflow and underflow conditions. Associated Task: - Declare an array-based stack with MAX size. - Implement push (), pop (), peek (), and display () functions. - Create a menu to choose required operations. - Handle exceptions like overflow and underflow. - Demonstrate all operations in the main () function. Exemplars: STACK MENU 1. Push 2. Pop 3. Peek 4. Display 5. Exit Enter your choice: Example Operation - Push 10 → Stack: [10] - Push 25 → Stack: [10, 25] - Pop → removes 25 Utility: Helps understand LIFO order in function calls, recursion, undo operations, and expression evaluation.			
Experiment No. D2	Stack Real world scenario	2 Hrs	CO4
Develop a C++ program that simulates a text editor by using two stacks to perform Undo and Redo operations based on the LIFO principle, allowing the user to track, revert, and reapply recent changes. Objective: - To understand & apply the LIFO (Last In First Out) principle of stacks in a real-time text editing scenario. - To implement Undo and Redo functionalities using two independent stacks to manage document changes.			



- To develop the ability to track, revert, and reapply changes efficiently in software applications.

Associated Task:

- Create two stacks: **Undo stack** for storing actions and **Redo stack** for undone actions.
- On making a change, push current state to Undo stack and clear Redo stack.
- For Undo, move the last state from Undo stack to Redo stack and restore previous state.
- For Redo, move state from Redo stack back to Undo stack and apply it.
- Display the current document state after each operation.

Exemplars:

- Students create a text editor simulation where typing “Hello” and then “World” results in stacked states:
 - Change 1: Hello
 - Change 2: Hello World
- Undo Example:

Undo → Reverts from Hello World to Hello.
- Redo Example:

Redo → Reapplies World, returning to Hello World.
- Multiple Operations Example:
 - User makes three changes, performs two Undo’s, and then types something new.
 - Students observe how the redo stack gets cleared after a new change.
- Document Display Example:

After each operation, the current state (e.g., "Hello") is printed to the console.

Utility:

- Practical Application:

Undo/Redo is widely used in text editors, image editing tools, IDEs, and design software.
- Demonstrates Stack Relevance:

Shows how LIFO ordering naturally supports step-by-step reversal of changes.

Experiment No. D3	Queue	2 Hrs	CO4
Write a C++ program to implement a queue using the FIFO principle, supporting basic operations such as enqueue, dequeue, and traversal to display the queue elements.			
Objective: • To implement queue operations based on the First In First Out (FIFO) principle. • To understand enqueue, dequeue, and traversal.			
Associated Task: • Implement enqueue, dequeue, and display functions. • Handle queue overflow and underflow conditions.			

**Exemplars:**

Simulate a printer queue:

- Add print jobs (enqueue).
- Process and remove jobs (dequeue).
- Display all pending print jobs.

Utility:

- Queues follow FIFO, making them suitable for ordered task processing.
- Used in scheduling, printing, CPU tasks, BFS, and network data handling.
- Ensure fair processing based on first-come, first-served principle.

Group E: Perform any 2 experiments.

Experiment No. E1	Tree data structure	2 Hrs	CO5
Develop a C++ program to represent an organization's hierarchy using a tree structure and display the hierarchy using preorder traversal. Objective: Represent an organization hierarchy (CEO → Managers → Employees) using a Tree and display it using preorder traversal. Associated Task: • Construct a tree with multiple child nodes. • Store employee names and their reporting structure. • Perform preorder traversal to display hierarchy. Exemplars: CEO • Manager A ➤ Employee 1 ➤ Employee 2 • Manager B ➤ Employee 3 Utility: Helps visualize hierarchical relationships.			
Experiment No. E2	Construct an Expression Tree	2 Hrs	CO5
Construct an Expression Tree from a given prefix expression, perform a non-recursive post-order traversal to display its postfix form, and then delete the tree to free memory. Objective: To construct an expression tree from a prefix expression, perform non-recursive post-order traversal to generate the postfix expression, and delete the tree to free memory. Associated Task: • Read the prefix expression from the user. • Construct an Expression Tree by scanning the prefix string from right to left. • Use a stack to build the tree (operand → node, operator → combine nodes). • Perform post-order traversal without recursion using two stacks. • Delete all nodes from the tree using post-order deletion.			

**Exemplars:****For prefix: +--a*bc/def**Constructed tree will represent the expression: $((a - (b * c)) + (d / e - f))$

After traversal → delete all nodes → tree becomes empty.

Utility:

- Expression trees are widely used in compilers, calculators, and symbolic mathematics systems.
- Helps understand tree construction, expression evaluation, and stack-based algorithms.

Experiment No.3	Graph Real World Scenario	2 Hrs	CO5
Write a program to represent flight connections between cities using a graph stored as an adjacency matrix and determine whether the graph is connected. Objective: Represent flight connections between cities using a graph, where each city (airport) is a node and each flight path is an edge with a cost (time or fuel). Store the graph using an Adjacency Matrix, and check whether the graph is connected or not. Associated Task: • Accept the list of cities (nodes). • Read flight connections and store them in an adjacency matrix. • If a flight exists between City A and City B, store its cost in the matrix; otherwise store 0. • Perform BFS or DFS to check graph connectivity. • Print whether all cities are reachable from a starting city. • Explain why adjacency matrix is suitable for this problem. Exemplars: Cities: A, B, C, D Adjacency Matrix (cost values): <pre> A B C D A 0 2 0 5 B 2 0 4 0 C 0 4 0 3 D 5 0 3 0 </pre> • If traversal covers all nodes → Graph is connected. • If some city is unreachable → Graph is not connected. Utility: • Useful for representing fixed-size networks like airports. • Adjacency matrix is simple, easy to implement, and efficient for graphs with frequent edge lookups. • Good for flight networks because checking if two airports are connected is $O(1)$.			

**Mini Project (Mandatory):**

Sample topics are listed here for reference. Students may choose any one topic from the given list or select a similar topic that covers the concepts learned in this subject.

1. College Event Registration & Participant Management System
2. Intelligent Chat Suggestion System (Using Trie + Queue)
3. Vehicle Parking Management System
4. Inventory & Stock Management System for a Store
5. Hospital Patient Appointment System
6. Attendance Tracking System
7. Online Food Ordering Simulation
8. City Map Navigation System (Graph Based)
9. Banking Transaction History Management System
10. Student Academic Record Management System
11. Sentiment Word Analyzer (Using Hash Table + Linked List)
12. AI-Based Spam Message Filter (Using Array + Stack + Hashing)
13. AI-Based Recommendation Helper (Using Linked List + Tree)
14. Rule-Based Expert System for Disease Prediction (Using Linked Lists + Trees)

Strength of CO-PO-PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	3	2	1	1	1	-	-	-	-	-	2	3	2	2
CO2	3	2	1	1	2	-	-	-	-	-	2	3	2	2
CO3	2	2	1	1	2	3	3	-	-	-	2	3	2	2
CO4	2	2	1	1	2	-	-	-	-	-	2	3	2	2
CO5	3	2	1	1	2	3	-	-	-	-	2	3	2	2
Avg.	3	2	1	1	2	3	3	-	-	-	2	3	2	2

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

This course serves as a prerequisite for / maps with the following future courses:

This course serves as a prerequisite for and aligns with future courses such as the Data Structures Laboratory, where students apply programming concepts to implement and analyze various data structures and algorithms.

Job Mapping:

- Programmer Trainee
- Junior Software Developer
- Coding Instructor / Programming Trainer



Program:	S. Y. B. Tech (Computer Engineering)	Semester:	III
Course:	Object Oriented Programming and Computer Graphics Laboratory	Course Code:	R25-CO-PCC-209
Teaching Scheme (Hrs./week)			Credits
Lecture	Practical	Tutorial	(PR)
-	02	-	01

Examination Scheme

CCE

ESE

TW

PR

-

-

25

25

Course Description:

The OOPCG course is designed to provide a comprehensive introduction and hands-on experience in implementing core concepts of Object-Oriented Programming and fundamental Computer Graphics algorithms. Students will be able to design and develop programs using object-oriented principles, handle runtime errors and concurrency, and implement fundamental computer graphics algorithms for drawing, filling, and clipping graphical objects.

Course Relevance:

The Object-Oriented Programming and Computer Graphics Laboratory is highly relevant to modern computing and software development as it equips students with essential practical skills required in the software industry. It enhances problem-solving ability, logical thinking, and programming proficiency.

Prerequisite: Understanding of basic programming and problem-solving concepts, including familiarity with variables, data types, control structures, and functions, to effectively implement object-oriented programming and computer graphics laboratory experiments.

Bridge Content: Revision of programming fundamentals with emphasis on object-oriented concepts (classes, objects, inheritance basics), control structures, arrays, functions, references and introductory graphics concepts.

Course Objectives: The course aims to:

1. To Understand and apply the fundamental principles and concepts of Object-Oriented Programming (OOP).
2. To develop programs using inheritance, polymorphism, and exception handling.
3. To Gain practical exposure by applying pointer concepts and memory management.
4. To develop a strong foundation in computer graphics concepts and algorithms through hands-on laboratory experiments.

Course Outcomes: After learning the course, students will be able to:

CO	Course Outcome	Bloom's Level
CO1	INTERPRET fundamental programming constructs in OOP for implementing applications.	2
CO2	APPLY object-oriented programming constructs like inheritance and pointers to solve complex problems.	3
CO3	DEMONSTRATE the concepts of polymorphism and exception handling for developing small systems involving multiple objects.	3
CO4	IMPLEMENT object-oriented programming principles to develop basic computer graphics concepts.	3



CO5	USE windowing and clipping concepts using algorithms to fill and clip polygons.	3
Course Contents		
Guidelines for Instructor's Manual		
The instructor's manual/Lab Manual is to be developed as a hands-on resource and reference. The instructor's manual needs to include prologue (about university/program/ institute/ department/ foreword/ preface), curriculum of course, conduction and Assessment guidelines, topics under consideration-concept, objectives, outcomes, set of typical applications/assignments/guidelines, references.		
Guidelines for Laboratory/Term Work Assessment		
Continuous assessment of laboratory work should be done based on overall performance and Laboratory assignments performance of student. Each Laboratory assignment assessment should be assigned grade/marks based on parameters with appropriate weightage. Suggested parameters for overall assessment as well as each Laboratory assignment assessment include timely completion performance, innovation, efficient codes, punctuality, and neatness.		
Guidelines for Student's Laboratory Journal		
The laboratory assignments are to be submitted by student in the form of journal. Journal consists of prologue, Certificate, table of contents, and handwritten write-up of each assignment (Title, Objectives, Problem Statement, Outcomes, software and Hardware requirements, Date of Completion, Assessment grade/marks and assessor's sign, Theory Concept in brief, algorithm, flowchart, test cases, Test Data Set (if applicable), mathematical model (if applicable), conclusion/analysis. Program codes with sample output of all performed assignments are to be submitted as softcopy. As a conscious effort and little contribution towards Green IT and environment awareness, attaching printed papers as part of write-ups and program listing to journal may be avoided. Students' programs maintained on cloud or college server by Laboratory In-charge is highly encouraged. For reference one or two journals may be maintained with program prints at Laboratory for accreditation purpose.		
Guidelines for Laboratory Conduction		
The instructor is expected to frame the assignments by understanding the prerequisites, technological aspects, utility and recent trends related to the topic. The assignment framing policy needs to address the average students and inclusive of an element to attract and promote the intelligent students. The instructor may set multiple sets of assignments and distribute among batches of students. It is appreciated if the assignments are based on real world problems/applications. Encourage students for appropriate use of Hungarian notation, proper indentation, and comments. Use of open-source software is to be encouraged. In addition to these, instructor may assign one real life application in the form of a mini-project based on the concepts learned. Instructor may also set one assignment or mini-project that is suitable to respective branch beyond the scope of syllabus. Set of suggested assignment list is provided in groups- A, B, and C. Each student must perform at least 10 assignments as at least 5 from group A, 5 from group B and one mandatory mini project based on data structure.		



All group assignments should be implemented in C++ without using built-in methods for major functionality of assignment.

Operating System recommended: - 64-bit Open-source Linux or its derivative

Programming tools recommended: - Open Source, Programming tool like G++ compiler, Eclipse IDE, or Visual Studio IDE.

Guidelines for Practical Examination

Both examiners should set the problem statements together, and assessment should focus mainly on correct implementation. Supplementary questions may be asked to check deeper understanding and efficiency. A fair, transparent, and encouraging evaluation process will build student confidence and support a strong academic start.

Guidelines for Oral Examination

Oral examination gauge students' knowledge and skills based on the spoken word, typically guided by questions or small tasks. A pair of examiners must design appropriate questions for each learning outcome. They should focus on depth rather than breadth. They should include potential follow-up questions and prompts based on different types of answers. Examiners should standardize the number of questions, difficulty of questions, and the time allotted. Questions should be based on the practical assignments performed in the term work and not on the entire syllabus.

List of Practical:

Group A - Any Five (from 1 to 6)

Experiment No.1	Implement a class Complex which represents the Complex Number data type. Implement the following	2 Hrs.	CO1
	• Constructor (including a default constructor which creates the complex number $0+0i$). • Overload operator + to add two complex numbers. • Overload operator * to multiply two complex numbers. • Overload operators << and >> to print and read Complex Numbers.		

Objective: To understand and implement operator overloading in C++ by creating a Complex class for performing arithmetic operations on complex numbers.

Associated Task: Implement a class Complex which represents the Complex Number data type. Implement the following:

1. Constructor (including a default constructor which creates the complex number $0 + 0i$)
2. Overload operator + to add two complex numbers.
3. Overload operator * to multiply two complex numbers.
4. Overload operators << and >> to print and read Complex Numbers.

Exemplars:

- Demonstrates creation of a user-defined data type using class in C++
- Demonstrates operator overloading for arithmetic operations
- Improves understanding of Object-Oriented Programming concepts

**Utility:**

- Helps in understanding **Object Oriented Programming concepts** like classes and objects.
- Demonstrates **operator overloading** in C++.

Experiment No.2	Develop a program in C++ to create a database of student's information system containing the following information: Name, Roll number, Class, Division, Date of Birth, Blood group, Contact address, Telephone number, Driving license no. and other. Construct the database with suitable member functions. Make use of constructor, default constructor, copy constructor, destructor, static member functions, friend class, this pointer, inline code and dynamic memory allocation operators-new and delete.	2Hrs.	CO2
------------------------	---	--------------	------------

Objective: To understand and implement advanced Object-Oriented Programming concepts in C++ by developing a Student Information Management System.

Associated Task:

Construct the database with suitable member functions and implement the following concepts: Constructor, Destructor, Static Data Members and Static Member Functions, Inline Functions, Dynamic Memory Allocation using new and delete.

Exemplars:

- Uses classes and objects for storing student records
- Implements default and parameterized constructors
- Uses inline functions for small utility methods
- Implements dynamic memory allocation using new and delete

Utility:

- Improves understanding of constructors and destructors
- Useful for developing student record and database management systems
- Improves understanding of constructors and destructors

Experiment No.3	Imagine a publishing company which does marketing for book and audio cassette versions. Create a class publication that stores the title (a string) and price (type float) of publications. From this class derive two classes: book which adds a page count (type int) and tape which adds a playing time in minutes (type float). Write a program that instantiates the book and tape class, allows user to enter data and displays the data members..	2Hrs.	CO2
------------------------	--	--------------	------------

Objective: To understand and implement the concept of inheritance in C++ by creating a base class and derived classes for managing publication details.

Associated Task:

- Create a class publication that stores the title (string) and price (float) of publications. Derive two classes from it: book – adds page count (int), tape – adds playing time in minutes (float).
- Write a program that: Instantiates objects of book and tape.
- Accepts data from the user.



- Displays the entered data members. Exemplars: - Demonstrates single inheritance in C++ - Illustrates code reusability and hierarchical classification - Uses objects of derived classes to access inherited properties Utility: - Helps in understanding inheritance and reusability in Object-Oriented Programming. - Enhances skills in creating hierarchical class structures - Forms the basis for advanced OOP concepts like multilevel and hybrid inheritance			
Experiment No.4	Develop a program in C++ to create an Employee Payroll System using multilevel inheritance. Create a base class Employee to store employee details. Derive a class Salary to store salary information and allowances. Derive another class Payroll from Salary to calculate and display gross salary. Accept employee information from the user and display complete payroll details..	2 Hrs.	CO2
Objective: To understand and implement multilevel inheritance in C++ by developing an Employee Payroll System for storing employee details and calculating gross salary.. Associated Task: - Develop a program in C++ to create an Employee Payroll System using multilevel inheritance. - Create a base class Employee to store employee details. - Derive a class Salary to store salary information and allowances. - Derive another class Payroll from Salary to calculate and display gross salary. - Accept employee information from the user and display complete payroll details. Exemplars: - Demonstrates multilevel inheritance in C++. - Represents real-world employee payroll management system - Console-based interactive C++ application Utility: - Useful for developing payroll and employee management systems - Demonstrates hierarchical relationship between classes - Reduces code redundancy through inheritance			
Experiment No.5	Create a base class Shape and derive the following classes from it: Circle, Rectangle, Triangle The base class should contain a virtual function calculateArea() and a display function. Each derived class should override the virtual function to calculate and display the area of the respective shape.	2 Hrs.	CO3
Objective: To understand and implement runtime polymorphism in C++ using virtual functions and inheritance for calculating the area of different shapes. Associated Task:			



- Creates a base class Shape
- Derives classes Circle, Rectangle, and Triangle
- Overrides the virtual function calculateArea() in each derived class
- Accepts dimensions from the user
- Calculates and displays the area using base class pointers

Exemplars:

- Demonstrates runtime polymorphism.
- Uses virtual functions for dynamic binding.
- Uses base class pointers to access derived class objects.

Utility:

- Helps in understanding runtime polymorphism and virtual functions.
- Demonstrates dynamic method dispatch.
- Improves understanding of inheritance and method overriding.

Experiment No.6	Develop a program for simulation of any real time application with required functionalities. For e.g. ATM machine with functionalities like checking account balance, withdrawing, and depositing money. Use try, catch, and finally blocks to handle potential exceptions such as insufficient funds(throwing ArithmeticException) and invalid input (throwing IllegalArgumentException). Ensure that the application continues to run smoothly after handling exceptions.	2 Hrs.	CO3
------------------------	---	---------------	------------

Objective:

To understand and implement Exception Handling in C++ by developing a real-time application simulation such as an ATM Machine with proper error handling mechanisms.

Associated Task:

- Develop a C++ program for simulation of a real-time application such as an ATM Machine with the following functionalities: Check account balance, Withdraw money, Deposit money.
- Use exception handling concepts: try-throw-catch
- Handle exceptions for: Insufficient balance & Invalid input or negative amount.
- Ensure that the application continues execution smoothly after handling exceptions.

Exemplars:

- ATM Machine simulation using C++
- Balance inquiry functionality
- Money withdrawal and deposit operations
- Exception handling using try, throw, and catch
- Handling insufficient funds exception
- Handling invalid amount input exception

**Utility:**

- Helps in understanding Exception Handling in C++
- Prevents abnormal program termination
- Enhances debugging and error-handling skills
- Encourages development of fault-tolerant applications

Group B: Perform any Five experiments

Experiment No.1	Write a C++ program to draw the following pattern using (a) the DDA line drawing algorithm for both rectangles with Dotted, Thick line style and (b) Bresenham's line drawing algorithm for a diamond shape with Dashed, Solid line style.	2 Hrs.	CO4
------------------------	--	---------------	------------

Objective:

To implement line drawing algorithms in computer graphics by using the DDA algorithm to draw rectangles with dotted and thick line styles and the Bresenham's algorithm to draw a diamond shape with dashed and solid line styles.

Associated Task:

- Implement the DDA line drawing algorithm
- Draw a rectangle using dotted and thick line styles
- Implement Bresenham's line drawing algorithm
- Draw a diamond shape using dashed and solid line styles
- Display the output using a graphics library

Exemplars:

- DDA Algorithm: Rectangle drawn with:
 - Dotted line style
 - Thick line style
- Bresenham Algorithm: Diamond shape drawn with:
 - Dashed line style
 - Solid line style

Utility:

- Helps understand fundamental computer graphics algorithms
- Demonstrates pixel-level line drawing techniques
- Forms the foundation for advanced rendering techniques

Experiment No.2	Write a menu driven program in C++ to draw circle using DDA, Bresenham's Midpoint circle drawing algorithm with different styles as solid, dotted and dashed circles.	2Hrs.	CO4
------------------------	---	--------------	------------

Objective:

To implement and compare different circle drawing algorithms using a menu-driven program and display circles with various styles such as solid, dotted, and dashed.

Associated Task:

- Implement DDA, Bresenham's, and Midpoint circle drawing algorithms
- Create a menu-driven program for user selection
- Draw circles with solid, dotted, and dashed styles
- Display graphical output using a graphics library

**Exemplars:**

- Dotted circle drawn using DDA algorithm
- Solid circle drawn using Bresenham's algorithm

Utility:

- Helps understand and compare circle drawing algorithms
- Demonstrates pixel plotting and symmetry in graphics

Experiment No.3	Write a menu driven program in C++ to draw a concave polygon and fill it with the desired color using the scan fill algorithm; flood fill and seed fill algorithms.	2 Hrs.	CO4
------------------------	---	---------------	------------

Objective: To develop a menu driven program in C/C++/Java to:

- Draw a concave polygon on the graphics screen
- Fill the polygon with a desired color using:
 - Scan Line Fill Algorithm
 - Flood Fill Algorithm
 - Seed Fill Algorithm

Associated Tasks:

1. Create graphics window and draw a concave polygon
2. Implement scan line algorithm to fill interior pixel by pixel
3. Implement flood fill using boundary color detection
4. Implement seed fill using starting interior point
5. Allow user to choose desired color and method
6. Display final filled polygon on screen

Exemplars:**1. Scan Fill Output**

- Concave polygon filled using horizontal scan lines
- Color applied between intersection points
- Demonstrates edge table and active edge table concept

2. Flood Fill Output

- Filling from interior seed point
- Color spreads until boundary is reached
- 4-connected / 8-connected approach

3. Seed Fill Output

- Recursive pixel replacement
- Filling only inside region
- Demonstrates stack based region filling

Utility: Used in paint and drawing applications

- Helps understand:
 - Region filling techniques
 - Pixel connectivity
 - Raster graphics concepts



Experiment No.4	Write a C++ program to implement the Cohen-Sutherland line clipping algorithm.	2 Hrs.	CO5
Objective: To implement the Cohen–Sutherland line clipping algorithm to clip a line against a rectangular clipping window and display the clipped line. Associated Task: • Define a rectangular clipping window • Assign region codes to line endpoints • Apply the Cohen–Sutherland line clipping logic • Display the original line and the clipped line • Demonstrate acceptance or rejection of the line Exemplars: Before Clipping: A line partially outside the rectangular window is drawn in red After Clipping: The visible portion of the line inside the window is drawn in green Utility: • Helps understand line clipping techniques in computer graphics • Graphics pipelines and rendering • Improves efficiency by displaying only visible objects			
Experiment No.5	Write a menu driven program in C++ to draw a concave polygon and fill it with the desired color using the scan fill algorithm; flood fill and seed fill algorithms.	2 Hrs.	CO5
Objective: To develop a menu-driven program that draws a concave polygon and fills it with a desired color using scan line fill, flood fill, and seed fill algorithms Associated Task: • Draw a concave polygon on the screen • Implement the scan line fill algorithm • Implement flood fill algorithm • Implement seed fill algorithm • Allow the user to choose the filling technique using a menu Exemplars: Scan Line Fill, Flood Fill, Seed Fill Utility: • Demonstrates filling techniques for concave polygons • Useful in: Computer graphics systems, Paint and drawing applications, CAD and animation software • Helps understand region filling concepts and pixel connectivity			
Experiment No.6	Write a C++ program to implement bouncing ball using sine wave form OR program to draw Sun Rise and Sunset OR Moving car animation OR Water wave animation using sine curve	2 Hrs.	CO5
Objective: To understand and implement basic computer graphics concepts in C++ by creating animations and graphical scenes such as a bouncing ball using sine wave motion or drawing Sunrise and Sunset.			

**Associated Task:**

- Use graphics functions, loops, mathematical equations, and animation techniques for real-time visualization.
- Make use of maximum features of Object Oriented Programming.

Exemplars:

- Use of graphics library functions in C++
- Animation using loops and delay functions
- Real-time graphical rendering

Utility:

- Enhances visualization and animation design skills
- Useful in game development and simulation applications
- Demonstrates real-time graphics rendering techniques

Group C - Mini Project

1. Mini-Project using maximum features of Object-Oriented programming to develop solutions for topics like mentioned below:
 - Inventory Management System
 - Vehicle Management System
 - Online Examination System
 - Railway Reservation System
 - Shopping Cart/Billing System
 - E-Learning Management System
2. Design and implement game / animation clip / Graphics Editor using open source graphics library.

Strength of CO-PO-PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	2	3	3	2	1	-	-	2	1	2	-	2	2	3
CO2	2	3	3	3	1	-	-	2	1	2	-	2	2	3
CO3	2	3	3	3	1	-	-	2	1	2	-	2	2	3
CO4	2	3	2	2	1	-	-	2	1	2	-	2	2	3
CO5	2	3	3	2	1	-	-	2	1	2	-	2	2	3
Avg.	2	3	2.8	2.4	1	-	-	2	1	2	-	2	2	3

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

This course serves as a prerequisite for / maps with the following future courses:

This course serves as a prerequisite for and maps with future courses including Advanced Java Programming, Web Technologies, Computational Modelling and Simulation, Robotics Programming, and AI & ML, as it equips students with fundamental object-oriented programming skills and graphical computing concepts required for these advanced domains.

Job Mapping:

After completing this course, students will acquire the ability to build robust object-oriented systems, design graphical solutions, and pursue entry-level roles such as: Java Developer, C++ Developer, Software Developer, Graphics Programmer, UI Designer, Game Developer.



Program:	S. Y. B. Tech (Computer Engineering)			Semester:	III		
Course:	Community Engagement Project			Course Code:	R25-CO-CEP-210		
Teaching Scheme (Hrs./week)			Credits (PR)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
-	04	-	2	-	-	50	-
Course Description: This course offers experiential learning through supervised group-based community projects carried out in collaboration with local authorities or organizations, focusing on social responsibility, practical engagement, and real-world problem solving.							
Course Relevance: The course fosters social awareness, civic responsibility, and practical problem-solving skills by enabling students to apply their academic knowledge to real-world community challenges through direct engagement and teamwork.							
Prerequisite: Students should have prior knowledge of							
1. Awareness of social, ethical, and civic responsibilities in addressing community-based issues							
2. Ability to collaborate effectively in teams and communicate clearly with diverse stakeholders							
3. Application of problem-solving skills and basic project planning techniques to real-world challenges							
4. Proficiency in communication in the local language to facilitate meaningful community engagement							
Bridge Content: Orientation on social and ethical responsibilities, basics of community interaction, teamwork and communication skills, introductory project planning, and use of basic engineering and digital tools for data collection, analysis, and reporting.							
Course Objectives:							
5. BUILD a mutually beneficial relationship between the institution and the local community.							
6. PROVIDE opportunities for community engagement to develop empathy, teamwork, and problem-solving skills.							
7. ENHANCE understanding of community challenges and the role of engineering in addressing them.							
8. APPLY technical knowledge and skills to design and implement solutions with positive social impact.							
9. DEVELOP the skills to evaluate and analyze engagement outcomes, drawing actionable insights for sustainable impact.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome						Bloom's Level
CO1	IDENTIFY and explain community needs, challenges, and stakeholder roles.						2
CO2	ANALYZE real-world community problems considering social and technical aspects.						4
CO3	APPLY teamwork, communication, and planning skills in community-based projects.						3
CO4	IMPLEMENT context-specific solutions using engineering principles.						3
CO5	EXAMINE the impact of community interventions and present reflective outcomes.						4



Course Contents			
List of Practical			
Experiment No. 1	Community Orientation Visit and Stakeholder Interaction	CO1	4 Hrs.
Objective: To build a mutually beneficial relationship between the institution and the local community through stakeholder interaction and field engagement. Associated Tasks: 1. Study the demographic, social, economic, and infrastructural profile of the selected community. 2. Identify key stakeholders and analyze their roles in community development activities. 3. Conduct structured interactions with community representatives to understand local issues and expectations. 4. Document community resources, facilities, and existing development initiatives. 5. Examine opportunities for institutional participation in community development. 6. Prepare a community profile report highlighting stakeholder observations and engagement opportunities. Exemplars and Utility: Helps students understand community structure, stakeholder roles, and local challenges while improving communication, observation, and interaction skills.			
Experiment No. 2	Community Survey and Need Assessment	CO1	4 Hrs.
Objective: To identify community needs, challenges, and stakeholder expectations through surveys and community interactions. Associated Tasks: • Design survey questionnaires. • Conduct household and stakeholder surveys. • Collect demographic, educational, economic, environmental, and technological data. • Compile and organize collected information. • Analyze survey findings. Exemplars and Utility: Develops survey design, data collection, analysis, and documentation skills required for community-based projects.			
Experiment No. 3	Analysis of Community Problems and SDG	CO2	4 Hrs.
Objective: To enhance understanding of community challenges and analyze real-world problems from social and technical perspectives while mapping them to relevant SDGs. Associated Tasks: 1. Analyze survey and field data. 2. Identify root causes of community problems. 3. Classify issues into social, environmental, and technological categories. 4. Map identified problems to relevant SDGs. 5. Prioritize issues based on impact and feasibility. 6. Prepare SDG mapping documentation. Exemplars and Utility: Promotes sustainable thinking and analytical skills by linking local issues to global development goals.			
Experiment No.4	Community Project Planning and Team Management	CO3	4 Hrs.
Objective: To develop empathy, teamwork, communication, and problem-solving skills through			



collaborative planning of community-based projects.

Associated Tasks:

1. Form project teams and assign responsibilities.
2. Define project objectives and expected outcomes.
3. Prepare activity schedules and timelines.
4. Identify required resources and stakeholders.
5. Develop implementation and monitoring plans.
6. Obtain mentor approval.

Exemplars and Utility: Strengthens teamwork, leadership, planning, documentation, and project management skills.

Experiment No.5	Implementing solutions to identified community problems.	CO4	4 Hrs.
------------------------	---	------------	---------------

Objective: To apply technical knowledge and engineering skills to design and implement context-specific solutions with positive social impact.

Associated Tasks:

1. Execute approved action plans.
2. Conduct awareness campaigns, training sessions, or demonstrations.
3. Implement simple technological solutions or prototypes.
4. Coordinate with community members and local authorities.
5. Document implementation activities.
6. Monitor participation and effectiveness.

Exemplars and Utility: Enhances leadership, communication, technical implementation, and community engagement skills through real-world interventions.

Experiment No.6	Evaluate effectiveness and impact assessment.	CO5	4 Hrs.
------------------------	--	------------	---------------

Objective: To evaluate community engagement outcomes and derive actionable insights for sustainable community development and continuous improvement.

Associated Tasks:

1. Collect feedback from beneficiaries and stakeholders.
2. Measure project outcomes against objectives.
3. Analyze social, environmental, and technical impacts.
4. Document challenges, lessons learned, and recommendations.
5. Prepare final report and presentation.
6. Reflect on personal and professional learning outcomes.

Exemplars and Utility: Develops evaluation, critical thinking, report writing, presentation, and reflective learning skills while assessing community impact.

Course Implementation

- Students will work in groups of 3 to 4, assigned to a village, municipal ward, or local community close to the college or their residence.
- Each group will be guided by a faculty mentor throughout the project.
- For a class of 60 students, three batches of approximately 20 students each will be formed. Each batch will have a practical engagement of 4 hours.
- The CEP must focus on unique community activities, separate from existing programs



like NSS, NCC, Green Club, or other hobby/special interest clubs.

- Each student is required to maintain an activity log documenting their participation and learning, to be verified by the mentor or HoD.
- An internal evaluation will be conducted by a committee appointed by the HoD. Assessment will consider participation, contribution, and overall engagement.
- The Oral/Presentation examination will include presentations and demonstrations of the project outcomes.

Mode of Evaluation

- Evaluation shall be conducted internally through **two reviews**, each carrying **50% weightage**.
- **Review I (Mid-Semester):** Assessment of community study, need assessment, problem identification, SDG mapping, and project planning.
- **Review II (End-Semester):** Assessment of project implementation, impact evaluation, documentation, presentation, and overall community engagement outcomes.
- A **final report with plagiarism check** must be submitted. Each student shall submit a **report** clearly indicating their **contribution**.

Suggestive list of topics under Community Engagement Project:

- Entrepreneurship initiatives
- Civic awareness
- Women's education and empowerment
- Healthcare practices
- Road safety awareness
- Waste management and recycling
- Herbal and nutrition awareness
- Government schemes awareness
- Yoga and fitness programs
- Water access and conservation
- Food safety and adulteration
- Mental health and stress management
- Renewable energy initiatives
- Climate change awareness
- Health and hygiene awareness
- Cultural preservation
- Tree plantation and soil conservation
- Chronic disease awareness
- Digital literacy
- Safe use of mobile phones and social media
- Healthy eating habits
- Waterbody cleaning and restoration
- Career guidance and skill development
- Disaster preparedness
- Cybersecurity awareness
- E-governance awareness



- IoT applications for villages
- Tech solutions for environmental monitoring
- STEM activities for school children

Proposal Submission and Report

Each CEP group shall submit a project proposal at the beginning of the semester containing:

- Project Title
- Aim, Objectives, and Expected Outcomes
- Execution Plan and Timeline
- Community/Location Details
- Required Resources
- Faculty Mentor Approval

At the end of the semester, each group shall submit a final report containing:

- Abstract
- Introduction
- Objectives
- Methodology
- Activities Performed
- Results and Impact Assessment
- Conclusion
- References and Appendices

The project outcomes shall be presented through an A3 poster and presentation.

Textbooks:

1. Waterman, A. *Service-Learning: A Guide to Planning, Implementing, and Assessing Student Projects*. Routledge, 1997.
2. Beckman, M., & Long, J. F. *Community-Based Research: Teaching for Community Impact*. Stylus Publishing, 2016.
3. IDEO Press. *Design Thinking for Social Innovation*. 2015.
4. Dostilio, L. D., et al. *The Community Engagement Professional's Guidebook: A Companion to The Community Engagement Professional in Higher Education*. Stylus Publishing, 2017.

MOOCs Links and additional reading material:

1. NPTEL Course: *Ecology and Society* – https://onlinecourses.nptel.ac.in/noc20_hs77/preview

Web Links

1. UNESCO: *Education for Sustainable Development* – <https://www.unesco.org>
2. EPICS (Engineering Projects in Community Service) – <https://engineering.purdue.edu/EPICS>
3. Ashoka: *Innovators for the Public* – <https://www.ashoka.org>
4. Design for Change – <https://www.dfeworld.com>

**Strength of CO-PO-PSO Mapping**

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	2	1	–	–	2	3	2	2	2	2	1	2	1	1
CO2	3	3	2	2	2	3	3	–	–	–	2	3	2	1
CO3	–	–	–	–	2	2	–	2	3	3	2	1	2	2
CO4	3	3	3	2	3	–	2	–	–	–	2	3	3	2
CO5	2	2	–	2	2	2	3	3	–	3	3	2	2	3
Avg.	2.5	2.25	2.5	2.0	2.2	2.5	2.5	2.33	2.5	2.67	2.0	2.2	2.0	1.8

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

This course serves as a prerequisite for / maps with the following future courses:

The Community Engagement Project provides foundational exposure to real-world problem identification, teamwork, ethical responsibility, and project planning. The competencies developed in this course support and strengthen learning in the following future courses:

- Mini Project
- Major Project / Capstone Project
- Internship / Industry-based Learning
- Project Management
- Professional Practice and Ethics
- Entrepreneurship Development
- Socially Relevant Engineering Projects

Job Mapping:

Job opportunities that one can get after learning this course:

1. Project Coordinator
2. CSR Executive
3. Sustainability Associate
4. Community Development Officer



Semester - IV

Course Code	Course Type	Name of the Subject	Teaching Scheme (Hrs./week)			SL	Total	Examination Scheme and Marks						Credit			
			TH	PR	TUT			CCE	ESE	TW	PR	OR	TOTAL	TH	PR	TUT	TOTAL
R25-CO-PCC-211	Program Core course - 4	Applied Discrete Mathematics	3	-	-	45	90	40	60	-	-	-	100	3	-	-	3
R25-CO-PCC-212	Program Core course - 5	Operating System	2	-	-	30	60	40	60	-	-	-	100	2	-	-	2
R25-CO-PCC-213	Program Core course - 6	Database Management Systems	3	-	-	45	90	40	60	-	-	-	100	3	-	-	3
R25-CO-OEC-214	Multidisciplinary Course - 2	Embedded Systems & Internet of Things	2	-	-	30	60	40	60	-	-	-	100	2	-	-	2
R25-CO-MDM-215	Open Elective - 2	Digital Governance & Management Information System	2	-	-	30	60	40	60	-	-	-	100	2	-	-	2
R25-CO-VSC-216	Vocational Skill Course	Programming in Java	1	2	-	15	60	-	-	25	25	-	50	1	1	-	2
R25-CO-AEC-217	Ability Enhancement Course	Modern Indian Language and Technical Writing	1	2	-	15	60	-	-	50	-	-	50	1	1	-	2
R25-CO-EEM-218	Entrepreneurship/Economics/Management course	Entrepreneurship and Startup Fundamentals	2	-	-	30	60	50	-	-	-	-	50	2	-	-	2
R25-CO-PCC-219	Program Core Course- Lab	Operating System Lab	-	2	-	-	30	-	-	-	25	-	25	-	1	-	1
R25-CO-PCC-220	Program Core Course Lab	Database Management Systems Lab	-	2	-	-	30	-	-	-	25	-	25	-	1	-	1
R25-CO-VEC-221	Value Education Course	Sustainable Technology and Green Computing	2	-	-	30	60	50	-	-	-	-	50	2	-	-	2
			18	8	0	270	660	350	300	25	75	0	750	18	4	0	22

Sunkar

R. Ganesan



Program:		S. Y. B. Tech (Computer Engineering)		Semester:		IV	
Course:		Applied Discrete Mathematics		Course Code:		R25-CO-PCC-211	
Teaching Scheme (Hrs./week)			Credits (TH)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
3	-	-	3	40	60	-	-
Course Description: This course introduces fundamental concepts of discrete mathematics required in computer engineering. The course covers logic, sets, relations, functions, counting techniques, Boolean algebra, graphs, and trees. The subject focuses on developing logical reasoning and problem-solving skills used in programming, algorithms, databases, computer networks, and digital systems.							
Course Relevance: • Develops logical and analytical thinking required for programming and algorithms. • Helps students model computing problems using discrete mathematical structures. • Provides foundation for subjects like Data Structures, Algorithms, DBMS, Digital Logic, and Computer Networks. • Enhances problem-solving ability in software and computing applications.							
Prerequisite: Students should have basic knowledge of: • Elementary algebra and equations • Basic set concepts • Simple mathematical reasoning • Basic understanding of computer applications							
Bridge Content: Before the course begins or during initial lectures: • Review of basic algebraic operations • Basics of Boolean expressions							
Course Objectives: Students will be able to: 1. Understand and apply logical reasoning techniques in computing problems. 2. Use sets, relations, and functions for mathematical modeling of systems. 3. Apply counting principles and recurrence relations in problem-solving. 4. Understand Boolean algebra and discrete structures used in digital systems. 5. Represent and analyze graphs and trees used in computer science applications.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome					Bloom's Level	
CO1	APPLY propositional logic and proof techniques in problem-solving.					3	
CO2	USE sets, relations, and functions to model computational systems.					3	
CO3	SOLVE counting and recurrence problems related to programming and algorithms.					4	



CO4	EMPLOY Boolean algebra concepts in digital logic and decision systems.	3	
CO5	ANALYZE graphs and trees for computing applications.	4	
Course Contents			
Unit 1	Logic and Mathematical Reasoning	9 Hrs.	CO1
Introduction to propositions and logical connectives; compound statements; truth tables; logical equivalence; predicates and quantifiers; basic proof techniques: direct proof and proof by contradiction; mathematical induction; applications of logic in computer programs and decision-making systems.			
Exemplars / Practical Applications and usage:			
• Logical conditions used in login systems and eligibility checking. • Truth tables used in program decision making. • Mathematical reasoning applied in algorithm verification and program correctness.			
Unit 2	Sets, Relations & Functions	9 Hrs.	CO2
Definition of sets, subsets, universal sets, power sets, set operations and laws; relations and their types; equivalence relations; partial ordering; functions and types of functions; one-one, onto and bijective functions; inverse and composition of functions; applications in databases and computing systems.			
Exemplars / Practical Applications and usage:			
• Sets used in grouping users and data. • Relations used in database management systems. • Functions used in input-output mappings in programs.			
Unit 3	Counting Techniques & Recurrence Relations	9 Hrs.	CO3
Basic counting principles; addition and multiplication principles; permutations and combinations; combinations with repetition; pigeonhole principle; inclusion–exclusion principle; introduction to recurrence relations; recursive sequences; solving simple recurrence relations using iteration method.			
Exemplars / Practical Applications and usage:			
• Counting techniques used in password generation and scheduling. • Recurrence relations used in recursive algorithms. • Pigeonhole principle applied in collision detection.			
Unit 4	Boolean Algebra & Discrete Structures	9 Hrs.	CO4
Introduction to Boolean algebra; Boolean operations and laws; De Morgan’s laws; simplification of Boolean expressions; logic gates and switching circuits; introduction to lattices and discrete structures; applications in digital systems and computing.			
Exemplars / Practical Applications and usage:			
• Boolean algebra used in digital circuits. • Logic gates used in computer hardware. • Boolean expressions applied in search and filtering systems.			
Unit 5	Graphs & Trees	9 Hrs.	CO5
Introduction to graphs; directed and undirected graphs; graph representation using adjacency matrix and adjacency list; paths, cycles, and connectivity; graph traversal concepts: BFS and DFS; introduction to trees; properties of trees; rooted trees and spanning trees; applications in			



computer science.

Exemplars / Practical Applications and usage:

- Graphs used in social networks and maps.
- Trees used in file systems and databases.
- BFS and DFS used in searching and navigation systems.

Textbooks:

1. Kenneth H. Rosen, Discrete Mathematics and Its Applications, 8th Edition, McGraw-Hill Education.
2. J. P. Tremblay and R. Manohar, Discrete Mathematical Structures with Applications to Computer Science, McGraw-Hill.
3. C. L. Liu, Elements of Discrete Mathematics, Tata McGraw-Hill.
4. Ralph P. Grimaldi, Discrete and Combinatorial Mathematics, Pearson.

Reference Books:

1. Norman L. Biggs, *Discrete Mathematics*, Oxford University Press.
2. Ronald L. Graham, Donald E. Knuth, Oren Patashnik, *Concrete Mathematics*, Addison-Wesley.
3. Sharma J. K., *Business Statistics*, Pearson Education (for statistical inference & regression topics).
4. Sheldon Ross, *A First Course in Probability*, Pearson.
5. Seymour Lipschutz, *Discrete Mathematics*, Schaum's Outline Series.
6. Susanna S. Epp, *Discrete Mathematics with Applications*, Cengage Learning.
7. Tremblay and Manohar, *Discrete Mathematical Structures*, McGraw-Hill.

MOOCs Links and additional reading material:

1. Discrete mathematics for computer science: proofs, combinatorics, graphs, number theory, algebra. NPTEL: Discrete Mathematics for CS (IIT Kanpur)
<http://nptel.ac.in/courses/106104573>
2. Sets, logic, relations, functions, counting, graph theory, recurrence relations etc.
onlinecourses - http://nptel.ac.in/noc25_cs105/preview
3. Axiomatic probability, random variables, distributions, sampling, estimation, hypothesis testing, correlation and regression. NPTEL: Introduction to Probability Theory and Statistics,
<http://nptel.ac.in/courses/111102160>
4. Introductory probability and statistics for engineering students. NPTEL: Probability and Statistics (IIT Kharagpur) - <http://nptel.ac.in/courses/111105090>
5. Discrete mathematics for CS: logic, sets, relations, induction, recurrences, counting, graphs, discrete probability. MIT OCW: Mathematics for Computer Science.
<http://ocw.mit.edu/courses/6-1200j-mathematics-for-computer-science-spring-2024/>
6. Free interactive version with problems and solutions. MIT Open Learning Library: Mathematics for Computer Science (Interactive).
<http://openlearninglibrary.mit.edu/courses/course-v1:OCW+6.042J+2T2019/about>
7. Free book covering sets, logic, combinatorics, relations, graphs and more (good for Units 1–



3), Applied Discrete Structures.

<http://discrete.openmathbooks.org/pdfs/dmoi-tablet.pdf>

8. Open-access textbook for probability and statistics with examples and interactive tools (Units 4–5), Introduction to Probability, Statistics, and Random Processes – Hossein Pishro-Nik,
<http://www.probabilitycourse.com>

9. Free e-book: Introduction to Probability, Statistics, and Random Processes,
<http://www.e-booksdirectory.com/details.php?ebook=10166>

Strength of CO-PO-PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	3	2	1	1	2	-	-	-	-	-	2	3	2	2
CO2	3	3	2	1	2	-	-	-	-	-	2	3	2	2
CO3	3	2	3	2	3	-	-	-	-	-	3	3	2	2
CO4	2	3	2	3	2	-	-	-	-	-	3	2	2	2
CO5	2	3	2	3	2	-	-	-	-	-	3	3	2	2
Avg.	3	3	2	2	2	-	-	-	-	-	3	3	2	2

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

zAssessment Pattern

BT Level	CCE					ESE				
	CO1	CO2	CO3	CO4	CO5	CO1	CO2	CO3	CO4	CO5
	8	8	8	8	8	12	12	12	12	12
L1	√	√	√	√	√	√	√	√	√	√
L2	√	√	√	√	√	√	√	√	√	√
L3	√	√	√	√	√	√	√	√	√	√
L4	—	—	√	—	√	—	—	√	—	√
L5	—	—	—	—	—	—	—	—	—	—
L6	—	—	—	—	—	—	—	—	—	—

This course serves as a prerequisite for / maps with the following future courses:

This course provides mathematical and logical foundations required for advanced computer engineering subjects such as:

- Data Structures
- Design and Analysis of Algorithms
- Database Management Systems



- Theory of Computation
- Artificial Intelligence
- Computer Networks
- Digital Logic Design

Job Mapping:

The knowledge gained from Applied Discrete Mathematics helps students develop analytical and logical problem-solving skills required in software and computing industries. The course supports careers such as:

- Software Developer
- Backend Developer
- Database Programmer
- System Analyst
- QA Engineer
- Network Engineer
- Junior AI/ML Engineer



Program:		S. Y. B. Tech (Computer Engineering)		Semester:	IV		
Course:		Operating System		Course Code:	R25-CO-PCC-212		
Teaching Scheme (Hrs./week)			Credits (TH+PR)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
03	-	-	2	40	60	-	-
Course Description: This course provides a comprehensive introduction to the design and implementation of modern Operating Systems (OS).							
Course Relevance: In the era of Cloud Computing, Multi-core processors, and Mobile technology, understanding the Operating System is vital for every software engineer. This course is highly relevant for: • System Programming & Security: Understanding kernel-level operations to build secure and robust software. • DevOps & Cloud Infrastructure: Mastering resource isolation, virtualization, and containerization (like Docker). • High-Performance Computing: Optimizing application performance through efficient memory management and process scheduling. • Embedded Systems: Applying Real-Time Operating System (RTOS) concepts to IoT and automotive technology.							
Prerequisite: Data Structure, Digital Electronics							
Bridge Content:							
Course Objectives: The course aims to: 1. To understand the fundamental concepts, types, and structures of Operating Systems. 2. To understand and analyze process management Concurrency & Deadlock. 3. To Explore the various techniques of memory management. 4. To Understand I/O management, disk scheduling, and file systems.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome					Bloom's Level	
CO1	ANALYZE the fundamentals of Operating Systems, including types, structures, system calls, and basic Linux commands.					4	
CO2	APPLY process scheduling and identify the mechanism for dealing with deadlocks and concurrency concerns.					3	
CO3	APPLY techniques of memory management to solve memory management problems.					3	
CO4	ILLUSTRATE I/O and file management policies.					2	
Course Contents							
Unit 1	Introduction to Operating System				07 Hrs	CO1	
Basics of Operating Systems: Objectives & Functions, Evolution of OS, Types of Operating Systems, OS Service, System Calls: Introduction, Types of System Calls OS structure: Layered Approach, Monolithic, Microkernel Operating Systems Introduction to Linux OS: Components of Linux system, Basic Shell commands. Virtualization and Containers (Docker vs. Vms), Monolithic (Linux) and Hybrid (Windows/macOS) kernels.							
Exemplars / Practical Applications and usage: Smartphones vs. Mainframes.							



Unit 2	Process Management, Concurrency & Deadlock	08 Hrs.	CO2
Process management: Definition, types of process. Process States and Transitions diagram, Process Control Block (PCB), context switching. Process Scheduling. Types of Schedulers: long term, short term, middle term, Threads: Concept of thread, Multithreading, Scheduling Algorithms: Preemptive Scheduling vs Non-preemptive Scheduling, FCFS, SJF, RR. Concurrency: Critical section problem, Synchronization primitives (Semaphores, Mutex, Monitors) Synchronization Problems: Producer-Consumer, Inter-Process Communication (IPC): Message passing, Shared memory Deadlocks: Conditions, Prevention, Avoidance (Banker's Algorithm), Detection, Recovery Case Studies - 1) Deadlock in Railway Scheduling Systems			
Exemplars / Practical Applications and usage: Online Ticket Booking (Critical Section), Video Streaming (YouTube/Netflix).			
Unit 3	Memory Management	07 Hrs.	CO3
Introduction, Contiguous and non-contiguous, Fragmentation: Internal and External fragmentation Memory allocation strategies: First Fit, Best Fit, Worst Fit, Memory Partitioning: Fixed and Dynamic partitions Paging: Structure of page table, Segmentation. Virtual Memory: Background, Demand Paging Page Replacement: FIFO, LRU, Optimal Thrashing. Case study : Intel Premium, Translation Lookaside Buffer (TLB).			
Exemplars / Practical Applications and usage: Streaming Services (e.g., Netflix)			
Unit 4	File and Disk Management	08 Hrs.	CO4
File Management: File operation, Directory structure, File System structure, File Organization and Access, File Directories, File Allocation Methods, Secondary Storage Management, File Systems in Operating System (FAT, NTFS, EXT, and HFS). Disk Management: Secondary-Storage Structure - Disk structure, Flash Memory/SSD management, Disk scheduling algorithm (FIFO, SSTF, SCAN, C-SCAN, LOOK, C-LOOK), Disk reliability, Disk formatting, Boot-block, Bad blocks. Cloud Storage or HDFS (Hadoop Distributed File System) Case Studies - 1) Study of Android File System.			
Exemplars / Practical Applications and usage: Google Drive/Dropbox.			
Textbooks: 1. William Stallings, Operating System: Internals and Design Principles, Prentice Hall, ISBN-10: 0-13-380591-3, ISBN-13: 978-0-13-380591-8, 8th Edition 2. Abraham Silberschatz, Peter Baer Galvin and Greg Gagne, Operating System Concepts, WILEY, ISBN 978-1-118-06333-0, 9th Edition 3. Arnold Robbins, Nelson H. F. Beebe, Classic Shell Scripting, O'Reilly Media, Inc., 2005, ISBN 9780596005955.			
Reference Books: 1. Tom Adelstein and Bill Lubanovic, Linux System Administration, O'Reilly Media, ISBN-10: 0596009526, ISBN-13: 978-0596009526. 2. Harvey M. Deitel, Operating Systems, Prentice Hall, ISBN-10: 0131828274, ISBN-13: 978-0131828278. 3. Thomas W. Doeppner, Operating System in depth: Design and Programming, WILEY, ISBN: 978-0-471-68723-8. 4. Mendel Cooper, Advanced Shell Scripting, Linux Documentation Project. 5. Andrew S. Tanenbaum & Herbert Bos, Modern Operating System, Pearson, ISBN-13: 9780133592221, 4th Edition.			
MOOCs Links and additional reading material: 1. https://repository.dinus.ac.id/docs/ajar/Operating_System.pdf 2. https://onlinecourses.nptel.ac.in/noc24_cs108/preview			

**Strength of CO-PO-PSO Mapping**

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	3	2	-	-	2	-	-	-	-	-	-	3	3	2
CO2	3	3	2	2	-	-	-	-	-	-	-	3	2	2
CO3	3	3	2	2	-	-	-	-	-	-	-	2	2	3
CO4	2	2	1	-	-	-	-	-	-	-	-	2	3	3
Avg.	2.7	2.5	1.6	2.0	2.0	-	-	-	-	-	-	2.5	2.5	2.5

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

Assessment Pattern

BT Level	CCE				ESE			
	CO1	CO2	CO3	CO4	CO1	CO2	CO3	CO4
	10	10	10	10	15	15	15	15
L1	√	√	√	√	√	√	√	√
L2	√	√	√	√	√	√	√	√
L3	√	√	√	—	√	√	√	—
L4	√	—	—	—	√	—	—	—
L5	—	—	—	—	—	—	—	—
L6	—	—	—	—	—	—	—	—

This course serves as a prerequisite for / maps with the following future courses:**Direct Prerequisites for Core Academic Courses**

- System Programming
- Distributed Systems
- Computer Networks
- Advanced Computer Architecture

Mapping with Specialization Tracks

- Cloud Computing .
- Cyber Security
- Embedded Systems & IoT



Program:	S. Y. B. Tech (Computer Engineering)			Semester:	IV		
Course:	Database Management Systems			Course Code:	R25-CO-PCC-213		
Teaching Scheme (Hrs./week)			Credits (TH)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
3	-	-		40	60	-	-
Course Description: The Database Management Laboratory provides hands-on experience in database design and implementation using relational and NoSQL databases. Students practice ER modeling, SQL programming, transaction control, database objects, and MongoDB operations, and apply these concepts in a mini-project following SDLC principles.							
Course Relevance: This course equips students with essential database skills required for modern software applications. It strengthens backend development capabilities, supports data-driven decision-making, and prepares students for industry, internships, and advanced studies.							
Prerequisite: Discrete Mathematics, Data Structures and Algorithms							
Bridge Content:							
Course objectives: 1. To understand database concepts, design principles, and ER/ERR modelling. 2. To develop SQL and PL/SQL skills for efficient database operations and procedural programming. 3. To apply normalization techniques for designing well-structured relational databases. 4. To explore database transactions, concurrency control methods, and recovery mechanisms. 5. To analyse NOSQL database models and their role in managing unstructured data.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome						Bloom's Level
CO1	DESIGN database management system using ER model by understanding fundamentals of Database Systems.						3
CO2	IMPLEMENT SQL and PL/SQL programs to manage and manipulate relational data.						3
CO3	APPLY normalization techniques to improve database design and ensure data integrity.						3
CO4	DEMONSTRATE transaction management concepts and concurrency control techniques for reliable database systems						3
CO5	ANALYZE NoSQL database types and explain their suitability for handling unstructured data.						5



Course Contents			
Unit 1	Introduction to Database Management System	9 Hrs.	CO1
Introduction to Database Management Systems, Purpose of Database Systems, Database-System Applications, View of Data, Database Languages, Database System Structure, Enterprise Constraints Data Models, Database Design and ER Model: Entity, Attributes, Relationships, Constraints, Keys, Design Process, Entity Relationship Model, ER Diagram, Design Issues, Extended E-R Features, Converting E-R & EER diagram into tables.			
Exemplars / Practical Applications and usage: 1. A centralized College ERP system used to manage students, faculty, courses, examinations, attendance, and results. 2. A secure Banking Database System handling customer accounts, transactions, loans, and ATM operations. 3. Study of Architecture of any DBMS like Oracle or MySQL. Design a database schema for any problem given in previous Question Papers.			
Unit 2	SQL and PL/SQL	9 Hrs.	CO2
SQL: DDL, DML, Select Queries, String, Date and Numerical Functions, Aggregate Functions, View, Indexes, Group by and Having Clause, Join Queries, Set, Set operation, Set membership, Nested queries, DCL, TCL			
PL/SQL: Control Statement, Cursor, Stored Procedure and Function, Trigger			
Exemplars / Practical Applications and usage: 1. A centralized database system used to manage students, courses, examinations, results, and faculty access. 2. A secure relational database supporting customer accounts, transactions, loans, and ATM/online banking services.			
Unit 3	Relational Database Design	9 Hrs.	CO3
Relational Model: Basic concepts, Attributes and Domains, CODD's Rules, Relational Integrity, Referential Integrities, Database Design: Features of Good Relational Designs, Normalization, Atomic Domains and First Normal Form, Decomposition using Functional Dependencies, 2NF, 3NF, BCNF.			
Exemplars / Practical Applications and usage: 1. A college database uses the relational model to store student, faculty, course, and examination data in separate tables. 2. A banking database stores customer, account, loan, and transaction details using normalized relational tables.			
Unit 4	Database Transactions	9 Hrs.	CO4
Basic concept of a Transaction, Transaction Management, Properties of Transactions, ACID Properties, Concept of Schedule, Serial Schedule, Serializability: Conflict and View, Cascaded Aborts, Recoverable and Non-recoverable Schedules, Concurrency Control: Need, Locking Methods, Deadlocks, Time-stamping Methods, Recovery methods: Shadow-Paging and Log-Based Recovery, Checkpoints, Introduction to Query Processing and Query Optimization.			



Exemplars / Practical Applications and usage: Case study: Design Online Shopping Cart Transaction Management. In an e-commerce platform, multiple users simultaneously add, update, and purchase products. To ensure data consistency and reliability, the system must handle concurrent transactions effectively.

Unit 5	NoSQL Database	9 Hrs.	CO5
Introduction to NoSQL Database, NoSQL data models, CAP theorem and BASE Properties, Comparative study of SQL and NoSQL, MongoDB: CRUD Operations, Indexing and Aggregation.			
Exemplars / Practical Applications and usage: 1. Online Banking & Fund Transfer System 2. Online Railway / Airline Reservation System			
Textbooks:			
1. Silberschatz A., Korth H., Sudarshan S., "Database System Concepts", McGraw Hill Publishers, ISBN 0-07-120413-X, 6th edition 2. Connally T., Begg C., "Database Systems", 4th Edition, Pearson Education, 2002, ISBN 8178088614 3. D T Editorial Services "BIG DATA Black Book", Dreamtech Press ISBN 13: 9789351199311			
Reference Books:			
1. C J Date, "An Introduction to Database Systems", Addison-Wesley, ISBN: 0201144719 2. S.K.Singh, "Database Systems: Concepts, Design and Application", Pearson Education, ISBN 978-81-317-6092-5 3. Kristina Chodorow, Michael Dierolf, "MongoDB: The Definitive Guide", O Reilly Publications, ISBN: 978-1-449-34468-9 4. Adam Fowler, "NoSQL For Dummies", John Wiley & Sons, ISBN-1118905628 5. Kevin Roebuck, "Storing and Managing Big Data - NoSQL, HADOOP and More", Emerepty Limited, ISBN: 1743045743, 9781743045749 6. Joy A. Kreibich, "Using SQLite", O'REILLY, ISBN: 13:978-93-5110-934-1 7. Ivan Bayross, "SQL, PL/SQL the Programming Language of Oracle", BPB Publications ISBN: 9788176569644, 9788176569644			
MOOCs Links and additional reading material:			
1. NPTEL – Database Management System - https://nptel.ac.in/courses/106106220 2. NPTEL – Introduction to Database Systems - https://nptel.ac.in/courses/106105175 3. MongoDB – NoSQL Explained https://www.mongodb.com/resources/basics/databases/nosql-explained 4. Microsoft Learn – Data Modeling in NoSQL (Azure Cosmos DB) https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/modeling-data			



Strength of CO-PO-PSO Mapping														
CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	3	2	2	1	1	–	–	–	–	–	1	3	2	1
CO2	3	2	3	1	3	–	–	–	1	–	1	3	3	2
CO3	3	3	2	1	2	–	–	–	–	–	1	3	2	1
CO4	3	3	2	2	2	1	1	–	–	1	2	2	3	2
CO5	2	3	2	2	3	1	–	–	1	1	3	2	3	3
Avg.	2.8	2.6	2.2	1.4	2.2	1	1	-	1	1	1.6	2.6	2.6	1.8

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

Assessment Pattern										
BT Level	CCE					ESE				
	CO1	CO2	CO3	CO4	CO5	CO1	CO2	CO3	CO4	CO5
	8	8	8	8	8	12	12	12	12	12
L1	√	√	√	√	√	√	√	√	√	√
L2	√	√	√	√	√	√	√	√	√	√
L3	–	√	√	√	√	–	√	√	√	√
L4	–	–	–	√	√	–	–	–	√	√
L5	–	–	–	–	√	–	–	–	–	√
L6	–	–	–	–	–	–	–	–	–	–

This course serves as a prerequisite for / maps with the following future courses:

This course serves as a prerequisite and provides foundational knowledge for advanced courses such as Database Management Systems, Operating Systems, Data Analytics, Big Data Technologies, Cloud Computing, Artificial Intelligence, Machine Learning, Software Engineering, and Web/Application Development. The concepts learned in this course also support practical implementation in backend systems, data-driven applications, and intelligent computing solutions.

Job Mapping:

- SQL Developer
- Backend Developer
- Database Administrator (DBA) Trainee
- Data Analyst Trainee



Program:	S. Y. B. Tech (Computer Engineering)			Semester:	IV		
Course:	Embedded Systems & Internet of Thing			Course Code:	R25-CO-MDM-214		
Teaching Scheme (Hrs. / week)			Credits (TH)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
2	-	-		40	60	-	-

Course Description: Embedded Systems and Internet of Things (IoT) are key technologies driving modern smart devices, automation, and connected systems. This course provides students with foundational knowledge of embedded hardware, sensors, microcontrollers, communication protocols, and IoT architectures required for designing intelligent real-time applications. The course enables learners to develop practical skills for smart systems development used in domains such as healthcare, agriculture, industrial automation, smart homes, transportation, and wearable technologies, preparing them for multidisciplinary and industry-oriented innovation.

Course Relevance: Embedded Systems and Internet of Things (IoT) play a vital role in the development of smart and connected technologies used in modern industries and daily life. This course helps students understand the integration of hardware and software for designing intelligent real-time systems using sensors, microcontrollers, and communication technologies. The knowledge gained through this multidisciplinary minor supports applications in smart homes, healthcare, industrial automation, agriculture, transportation, and wearable devices, enhancing students' practical and innovation-oriented skills for emerging technology domain

Prerequisite: Students should have basic knowledge of computer fundamentals, programming concepts, electronic components, and digital systems. Familiarity with C programming, microcontrollers, sensors, and basic networking concepts will help students understand embedded systems and IoT application development effectively.

Bridge Content: Basics of C Programming, Fundamentals of Electronics and Digital Logic

Course Objectives:

- To understand the fundamentals of Embedded Systems and Internet of Things (IoT).
- To study embedded hardware platforms, processors, sensors, and IoT devices.
- To analyze IoT communication protocols, connectivity technologies, and cloud integration.
- To develop smart IoT-based solutions for real-world applications.

Course Outcomes: After learning the course, students will be able to:

CO	Course Outcome	Bloom's Level
CO1	DESCRIBE fundamentals of Embedded Systems, ARM architecture, and real-time systems	2
CO2	DEMONSTRATE the interfacing and utilization of IoT components, development boards, sensors and actuators.	3
CO3	ANALYZE IoT communication protocols, wireless technologies, and connectivity models for data exchange	4



CO4	DEVELOP secure IoT-based applications using cloud platforms and embedded devices for real-world problems.	3	
Course Contents			
Unit 1	Fundamentals of Embedded Systems	7 Hrs.	CO1
Introduction to Embedded Systems, Characteristics and Application Domains of Embedded Systems, Classification of Embedded Systems, Real-Time Systems and Real-Time Scheduling, Embedded Hardware and Software, Processor Basics and System-On-Chip (SoC), Introduction to ARM Processor and ARM Architecture, Design Constraints in Embedded Systems, Embedded System Development Tools, Emerging Trends in Embedded and Smart Systems.			
Exemplars / Practical Applications and usage:			
- Smart home automation system - ARM-based embedded controller applications - Real-time traffic control and monitoring systems - Industrial automation and IoT-based smart devices			
Unit 2	IoT Fundamentals and Smart Devices	8 Hrs.	CO2
Introduction to IoT, Definition and Characteristics of IoT, Evolution and Vision of IoT, IoT Functional Blocks and Architecture, Physical and Logical Design of IoT, Things in IoT, IoT Communication Models and APIs, Sensors and Actuators, IoT Development Boards: Arduino, Raspberry Pi, NodeMCU and ESP32, IoT Enabling Technologies, IoT Applications.			
Exemplars / Practical Applications and usage:			
- Smart lighting system - Smart irrigation system - Health monitoring devices - IoT-enabled home appliances			
Unit 3	IoT Communication Protocols and Connectivity	8 Hrs.	CO3
IoT Access Technologies, Wireless Sensor Networks, RFID and M2M Communication, IoT Communication Protocols, IEEE 802.15.4, ZigBee, Bluetooth and BLE, Wi-Fi and LoRaWAN, IPv4 and IPv6, 6LoWPAN, MQTT and CoAP Protocols, IoT Network and Application Layer Protocols, Challenges in IoT Communication.			
Exemplars / Practical Applications and usage:			
- Smart parking system - Wearable fitness tracker - Environmental monitoring system - Connected industrial sensors			
Unit 4	Cloud Integration, Security and IoT Applications	7 Hrs.	CO4
Cloud Computing for IoT, IoT Data Storage and Analytics, IoT Cloud Platforms, Edge and Cloud Concepts in IoT, IoT Security and Privacy Issues, Threats and Challenges in IoT, Smart Cities and Smart Healthcare, Industrial IoT Applications, Agriculture and Environmental Monitoring Applications, Emerging Trends in IoT and Smart Systems.			
Exemplars / Practical Applications and usage:			
- Air pollution monitoring system - Smart healthcare monitoring - Industrial IoT dashboard - Smart city surveillance system			

**Textbooks:**

- Raj Kamal, Internet of Things: Architecture and Design Principles, McGraw-Hill Education, Latest Edition.
- Arshdeep Bahga and Vijay Madisetti, Internet of Things – A Hands-On Approach, Universities Press, Latest Edition.
- David Hanes, Gonzalo Salgueiro, Patrick Grossetete, Rob Barton and Jerome Henry, IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things, Cisco Press, Latest Edition.
- Perry Lea, Internet of Things for Architects, Packt Publishing, Latest Edition.
- Tammy Noergaard, Embedded Systems Architecture: A Comprehensive Guide for Engineers and Programmers, Elsevier/Morgan Kaufmann, Latest Edition.

Reference Books:

1. Adrian McEwen and Hakim Cassimally, Designing the Internet of Things, Wiley Publications, Latest Edition.
2. Frank Vahid and Tony Givargis, Embedded System Design: A Unified Hardware/Software Introduction, Wiley India, Latest Edition.
3. Shibu K. V., Introduction to Embedded Systems, McGraw-Hill Education, Latest Edition.
4. David E. Simon, An Embedded Software Primer, Pearson Education, Latest Edition.

MOOCs Links and additional reading material:

- NPTEL: Introduction to Internet of Things
https://onlinecourses.nptel.ac.in/noc22_cs53/preview
- Coursera: Introduction to the Internet of Things and Embedded Systems
<https://www.coursera.org/learn/iot>
- edX: Embedded Systems Essentials with Arm
<https://www.edx.org/learn/embedded-systems>
- Arduino Tutorials and Embedded Programming Resources
<https://www.arduino.cc/en/Tutorial/HomePage>
- Raspberry Pi Documentation and IoT Projects
<https://www.raspberrypi.com/documentation/>

Strength of CO-PO-PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	3	2	1	1	2	-	-	-	-	-	1	3	1	1
CO2	2	2	3	2	3	-	-	-	-	-	1	3	2	1
CO3	2	3	2	3	2	-	-	-	-	-	2	3	2	1
CO4	2	2	3	2	3	-	-	-	-	-	2	3	2	2
Avg.	2.25	2.25	2.25	2.00	2.50	-	-	-	-	-	1.50	3.00	1.75	1.25

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)



Assessment Pattern								
BT Level	CCE				ESE			
	CO1	CO2	CO3	CO4	CO1	CO2	CO3	CO4
	10	10	10	10	15	15	15	15
L1	√	√	√	√	√	√	√	√
L2	√	√	√	√	√	√	√	√
L3	–	√	√	√	–	√	√	√
L4	–	–	√	–	–	–	√	–
L5	–	–	–	–	–	–	–	–
L6	–	–	–	–	–	–	–	–

This course serves as a prerequisite for / maps with the following future courses:

This course serves as a foundation for advanced courses related to Embedded Systems, Internet of Things (IoT), Smart Communication Technologies, Edge Computing, Cyber Physical Systems, Intelligent Autonomous Systems, Industrial Automation, Wireless Sensor Networks, Cloud and IoT Integration, and AI-enabled Smart Systems. The course also supports future learning in real-time embedded application development, smart device programming, and interdisciplinary innovation projects.

Job Mapping:

- IoT Developer
- Embedded Software Developer
- IoT Application Developer
- Automation and Control Engineer



Program:		S. Y. B. Tech (Computer Engineering)		Semester:		IV	
Course:		Programming in Java		Course Code:		R25-CO-VSC-216	
Teaching Scheme (Hrs./week)			Credits (TH+PR)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
1	2	-	1+1	-	-	25	25
Course Description: Java Programming is a comprehensive course designed to provide students with strong foundations in object-oriented programming and enterprise application development using Java technologies. The course covers fundamental programming concepts such as classes, objects, inheritance, polymorphism, exception handling, multithreading, file handling, GUI programming, and database connectivity. It also introduces advanced Java technologies including Servlets, JSP, and Struts 2.0 framework for developing dynamic web applications based on MVC architecture. Through theoretical concepts and practical implementation, students gain hands-on experience in developing standalone, database-driven, and web-based applications using modern development tools and frameworks.							
Course Relevance: Java is one of the most widely used programming languages in software industries due to its platform independence, robustness, security, and scalability. This course enables students to develop problem-solving abilities and application development skills required in modern software engineering. The subject forms the foundation for advanced technologies such as Spring Framework, Hibernate, Android Development, Enterprise Application Development, Full Stack Java Development, and Cloud-based applications. It also enhances employability by preparing students for industry-oriented programming, web technologies, and enterprise software development practices.							
Prerequisite: • Basic knowledge of programming concepts using C language • Understanding of variables, data types, operators, and control statements • Familiarity with functions, arrays, and basic problem-solving techniques • Basic understanding of object-oriented programming concepts Fundamentals of database systems and HTML							
Bridge Content: The content focuses on refreshing fundamental programming concepts required for learning Java Programming effectively. It includes revision of algorithms, flowcharts, data types, operators, decision-making statements, loops, functions, arrays, and basic object-oriented concepts such as classes and objects. Students are also introduced to basic HTML and database concepts to support advanced Java topics like JDBC, Servlet, and JSP development.							
Course Objectives: 1. To understand the fundamentals of object-oriented programming using Java. 2. To develop Java applications using classes, inheritance, packages, interfaces, exception handling, multithreading, and file handling. 3. To implement GUI-based and database-oriented Java applications using Java APIs. 4. To design web-based enterprise applications using Servlet, JSP, and Struts 2.0 framework.							
Course Outcomes: After learning the course, students will be able to:							



CO	Course Outcome	Bloom's Level	
CO1	APPLY Java programming fundamentals, object-oriented programming (OOP) concepts, operators, control statements, arrays, and classes to develop simple Java progra	3	
CO2	DEVELOP Java programs using inheritance, interfaces, packages, exception handling, multithreading, and file handling.	3	
CO3	BUILD GUI and database applications using Java Swing, JDBC, and event handling mechanisms.	3	
CO4	DEVELP dynamic web applications using Servlet, JSP, and Struts 2.0 framework.	4	
Course Contents			
Unit 1	Introduction to Java & Object-Oriented Programming	4 Hrs.	CO1
Introduction to Java, Features of Java, Java Virtual Machine (JVM), JDK, JRE, Structure of Java Program, Data Types, Variables, Constants, Operators and Expressions, Control Statements, Selection Statements, Looping Statements, Jump Statements, Arrays and Strings, Object-Oriented Concepts, Class and Object, Methods and Constructors, Method Overloading, Static Keyword, Command Line Arguments, Wrapper Classes, Introduction to Packages			
Exemplars / Practical Applications and usage:			
• Java programs used for calculator applications, student grading systems, and menu-driven applications. • Arrays and strings applied in searching student records and text processing. • Classes and objects used in modeling real-world entities such as students, employees, and bank accounts. • Method overloading used in developing flexible applications with multiple input options. • Packages used for modular software development.			
Unit 2	Advanced Object-Oriented Programming in Java	4 Hrs.	CO2
Inheritance- Types of Inheritance, Method Overriding, Super Keyword, Abstract Classes and Interfaces, Packages and Access Specifiers, Exception Handling- try, catch, throw, throws, finally, User Defined Exceptions, Multithreading- Thread Life Cycle, Thread Class, Runnable Interface, Synchronization, File Handling- Byte Stream and Character Stream, File Class, Serialization and Deserialization			
Exemplars / Practical Applications and usage:			
• Inheritance used in banking and employee management systems. • Interfaces applied in payment gateway and multiple-service systems. • Exception handling used in secure software applications. • Multithreading used in gaming, chat applications, and background processing. • File handling used in report generation and data storage systems.			



Unit 3	GUI Programming and Database Connectivity	4 Hrs.	CO3
Event Driven Programming, AWT Components, Swing Components- JFrame, JLabel, JTextField, JButton, JTable, Layout Managers, Event Handling Mechanism- ActionListener, MouseListener, KeyListener, JDBC- Architecture, JDBC Drivers, Database Connectivity using JDBC- Connection, Statement, PreparedStatement, ResultSet, CRUD Operations using Java and MySQL, Introduction to Networking in Java- Socket Programming Basics			
Exemplars / Practical Applications and usage: • GUI applications used in billing systems and registration forms. • Event handling used in button-click and keyboard interaction systems. • JDBC used in student databases and inventory systems. • Socket programming used in chat applications and client-server communication.			
Unit 4	Advanced Java Programming	4 Hrs.	CO4
Servlet- Introduction to Web Applications, Servlet Architecture and Life Cycle, Generic Servlet and HTTP Servlet, Handling Client Requests and Responses, Session Management, Cookies and URL Rewriting, JSP- Introduction to JSP, JSP Life Cycle, JSP Directives, JSP Scriptlets, JSP Implicit Objects, Expression Language (EL), JSTL Basics Struts 2.0 Framework- Introduction to MVC Architecture, Introduction to Struts 2.0, Struts Architecture, Configuration Files, Actions and Interceptors, Validation Framework, Developing Simple Web Applications using Struts 2.0			
Exemplars / Practical Applications and usage: • Servlets used in online form processing and authentication systems. • JSP used in dynamic web page generation. • Session management applied in e-commerce websites. • Struts framework used in enterprise web applications following MVC architecture.			
Textbooks:			
1. Java: The Complete Reference, Herbert Schildt, published by McGraw Hill Education, 13 th Edition, 2023. 2. Programming with Java , E. Balagurusamy, published by McGraw Hill Education India, 7th Edition, 2020. 3. Head First Java, Kathy Sierra and Bert Bates, published by O'Reilly Media, 3rd Edition, 2022.			
Reference Books:			
1. Paul Deitel and Harvey Deitel, <i>Java How to Program</i> , published by Pearson Education, 11th Edition, 2018. 2. Patrick Naughton, <i>The Java Handbook</i> , published by McGraw Hill Education, 1st Edition, 1996. 3. Jim Keogh, <i>J2EE: The Complete Reference</i> , published by McGraw Hill Education, 1st Edition, 2002.			
MOOCs Links and additional reading material:			
1. NPTEL – Programming in Java https://onlinecourses.nptel.ac.in/noc26_cs47/preview 2. Coursera – Java Programming and Software Engineering Fundamentals Specialization https://www.coursera.org/specializations/java-programming			



3. Oracle Java Tutorials (Official Resource) - <https://docs.oracle.com/javase/tutorial/>
4. GeeksforGeeks – Java Programming Language - <https://www.geeksforgeeks.org/java/>
5. Baeldung – Java Tutorials - <https://www.baeldung.com/java-tutorial>

List of Practical:**Group A: Introduction to Java and Object-Oriented Programming****Perform any 2 experiments.**

Experiment No. 1	Write Java programs to demonstrate basic syntax, data types, operators, command line arguments, and control statements.	2 Hrs.	CO1
-------------------------	---	---------------	------------

Objective:

To understand Java's basic syntax, data types, operators, command-line arguments, and control statements.

Associated Task:

Develop Java programs using decision-making and looping statements for solving simple computational problems.

Exemplars:

- Student grade calculator
- Simple calculator
- Even-odd number checker
- Menu-driven arithmetic operations

Utility:

Helps students understand the foundation of Java programming and develop logical problem-solving skills required for software development.

Experiment No.2	Develop Java programs using arrays and strings for searching, sorting, and string manipulation operations.	2 Hrs.	CO1
------------------------	--	---------------	------------

Objective:

To implement searching, sorting, and string manipulation operations using arrays and strings in Java.

Associated Task:

Develop Java applications for processing and manipulating numerical and textual data.

Exemplars:

- Linear search and binary search
- Bubble sort and selection sort
- String reverse and palindrome checker
- Word counting application

Utility:

Useful in data processing, text handling, and algorithm implementation used in software applications.

Experiment No.3	Implement class, object, constructor, method overloading, and static keyword concepts using suitable applications.	2 Hrs.	CO1
------------------------	--	---------------	------------

**Objective:**

To implement classes, objects, constructors, method overloading, and static keyword concepts in Java.

Associated Task:

Develop object-oriented applications representing real-world entities.

Exemplars:

- Student information system
- Employee payroll system
- Banking application
- Area calculation using method overloading

Utility:

Provides practical understanding of object-oriented programming concepts used in enterprise software development.

Group B: Advanced Object-Oriented Programming in Java

Perform any 3 experiments.

Experiment No.4	Develop a Java application demonstrating inheritance, method overriding, super keyword, and runtime polymorphism.	2 Hrs.	CO2
------------------------	---	---------------	------------

Objective:

To implement inheritance, method overriding, super keyword, and runtime polymorphism in Java.

Associated Task:

Develop Java applications demonstrating hierarchical relationships among classes.

Exemplars:

- Employee management hierarchy
- Vehicle management system
- Shape area calculation using polymorphism

Utility:

Helps students design reusable and maintainable software systems using inheritance and polymorphism.

Experiment No.5	Write Java programs using abstract classes, interfaces, and packages for real-world problem solving.	2 Hrs.	CO3
------------------------	--	---------------	------------

Objective:

To implement abstract classes, interfaces, and packages in Java applications.

Associated Task:

Develop modular Java applications using abstraction and interface-based programming.

Exemplars:

- Payment gateway interface
- Banking transaction system
- Package-based utility library

Utility:



Improves modular programming and supports large-scale software development practices.

Experiment No.6	Implement exception handling mechanisms including user-defined exceptions for banking or student management systems.	2 Hrs.	CO2
------------------------	--	---------------	------------

Objective:

To implement exception handling and user-defined exceptions in Java.

Associated Task:

Develop robust Java applications capable of handling runtime errors.

Exemplars:

- ATM withdrawal validation
- Student marks validation
- Invalid age exception handling

Utility:

Helps build secure and reliable software systems with proper error handling mechanisms.

Experiment No.7	Create Java programs for file handling operations using File class, byte streams, character streams, and serialization.	2 Hrs.	CO2
------------------------	---	---------------	------------

Objective:

To develop multithreading and file handling applications in Java.

Associated Task:

Implement concurrent programming and file processing operations.

Exemplars:

- Ticket reservation system
- File copy utility
- Serialization and deserialization program
- Producer-consumer simulation

Utility:

Provides practical exposure to concurrent execution and permanent data storage techniques

Experiment No.8	Develop multithreading applications using Thread class, Runnable interface, synchronization, and thread communication.	2 Hrs.	CO2
------------------------	--	---------------	------------

Objective:

To develop multithreading applications using Thread class, Runnable interface, synchronization, and thread communication.

Associated Task:

Implement concurrent Java applications using multiple threads and synchronization mechanisms.

Exemplars:

- Railway ticket reservation system
- Producer-consumer problem
- Traffic signal simulation



- Multi-user chat simulation

Utility:

Helps students understand concurrent execution, resource sharing, synchronization, and multitasking concepts used in operating systems and enterprise applications.

Group C: GUI Programming and Database Connectivity

Perform any 3 experiments.

Experiment No.9	Design GUI applications using AWT components and event handling mechanisms.	2 Hrs.	CO3
------------------------	---	---------------	------------

Objective:

To design GUI applications using AWT and Swing components.

Associated Task:

Develop interactive desktop applications with graphical user interfaces.

Exemplars:

- Student registration form
- Login form
- Calculator using Swing

Utility:

Enables development of user-friendly desktop applications with interactive interfaces.

Experiment No.10	Implement event-driven applications using ActionListener, MouseListener, and KeyListener interfaces.	2 Hrs.	CO3
-------------------------	--	---------------	------------

Objective:

To implement event-driven programming using event listener interfaces.

Associated Task:

Develop Java applications responding to user-generated events.

Exemplars:

- Button click event handling
- Keyboard event detection
- Mouse event tracking

Utility:

Helps create interactive and responsive GUI applications.

Experiment No.11	Create client-server communication applications using basic socket programming in Java.	2 Hrs.	CO3
-------------------------	---	---------------	------------

Objective:

To implement JDBC connectivity and socket programming in Java.

Associated Task:

Develop database-oriented and client-server applications.

Exemplars:

- Student database CRUD operations
- Employee record management



- Chat application using sockets

Utility:

Useful in enterprise application development and network-based software systems.

Experiment No.12	Develop Swing-based GUI applications using JFrame, JButton, JLabel, JTextField, JTable, and layout managers.	2 Hrs.	CO3
-------------------------	--	---------------	------------

Objective:

To develop Swing-based GUI applications using JFrame, JButton, JLabel, JTextField, JTable, and layout managers.

Associated Task:

Design user-friendly desktop applications with graphical interfaces and event handling.

Exemplars:

- Student registration form
- Employee information system
- Library management GUI
- Calculator application

Utility:

Provides practical exposure to desktop application development and graphical UI design.

Experiment No.13	Develop JDBC applications to perform database connectivity and CRUD operations using MySQL.	2 Hrs.	CO3
-------------------------	---	---------------	------------

Objective:

To develop JDBC applications for database connectivity and CRUD operations using MySQL.

Associated Task:

Implement Java database applications using JDBC API and MySQL database.

Exemplars:

- Student database management system
- Employee record management
- Library database application
- Product inventory system

Utility:

Helps students understand database-driven application development and integration of Java applications with relational databases.

Group D: Advanced Java Programming

Perform any 2 experiments.

Experiment No.15	Develop web applications using Servlet for handling client requests, session management, cookies, and form processing. (Servlet+Mysql)	2 Hrs.	CO4
-------------------------	--	---------------	------------

Objective:

To develop web applications using Servlet technology.

Associated Task:



Create Java web applications for handling client requests and session management.

Exemplars:

- User login system
- Online feedback form
- Session tracking application

Utility:

Provides understanding of server-side web application development using Java technologies.

Experiment No.15	Create dynamic web applications using JSP and Struts 2.0 framework implementing MVC architecture. (JSP+Mysql)	2 Hrs.	CO4
-------------------------	---	---------------	------------

Objective:

To create dynamic web applications using JSP and Struts 2.0 framework.

Associated Task:

Develop MVC-based enterprise web applications.

Exemplars:

- Student management system
- Product management application
- Online registration portal

Utility:

Helps students understand enterprise web application architecture and dynamic content generation.

Strength of CO-PO-PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	3	2	1	-	2	-	-	-	-	1	-	3	-	-
CO2	3	3	2	1	2	-	-	-	1	1	-	3	1	1
CO3	2	3	3	2	3	-	-	-	1	1	-	2	3	2
CO4	2	2	3	2	3	1	-	-	1	2	1	2	3	3
Avg.	2	2	2	2	2	1	-	-	1	1	1	2	2	2

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

This course serves as a prerequisite for / maps with the following future courses:

This practical-oriented Java Programming course provides hands-on programming, software development, database connectivity, GUI development, and web application development skills required for advanced computing subjects and industry-oriented technologies. The course supports understanding and implementation of modern software engineering concepts using Java technologies.

**Job Mapping:**

The knowledge and practical skills gained from Programming in Java enable students to develop industry-oriented software applications, desktop applications, database-driven systems, and web applications. The course prepares students for software development, enterprise application development, and full-stack programming roles in IT industries.

The course supports careers such as:

- Java Developer
- Software Developer
- Backend Developer
- Web Application Developer
- Desktop Application Developer



Program:	S. Y. B. Tech (Computer Engineering)			Semester:	IV		
Course:	Modern Indian Language & Technical Writing			Course Code:	R25-CO-PCC-217		
Teaching Scheme (Hrs. / week)			Credits (TH+PR)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
1	2	-	1+1	-	-	50	-
Course Description: Modern Indian Language & Technical Writing focuses on developing effective communication, technical writing, and professional documentation skills. The course enhances students’ reading, writing, speaking, and presentation abilities required in academic and professional environments while promoting expression through Indian languages and cultural understanding.							
Course Relevance: The course improves communication proficiency, technical documentation, and workplace interaction skills. It supports employability, professional communication, report writing, and overall personality development for engineering students.							
Prerequisite: Basic reading, writing, and communication skills in English and/or any Modern Indian Language.							
Bridge Content: Fundamentals of language communication, grammar basics, vocabulary enhancement, and introduction to professional writing practices.							
Course Objectives: - To develop effective communication skills in Modern Indian Language and English for academic and professional use. - To enhance technical writing, documentation, and professional correspondence skills. - To prepare students for software project documentation, presentations, and workplace communication. - To develop digital professionalism, presentation abilities, and awareness of modern collaborative tools.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome					Bloom’s Level	
CO1	DEMONSTRATE effective oral and written communication skills in Modern Indian Language and professional environments.					3	
CO2	APPLY principles of technical and professional communication for preparing emails, resumes, reports, and presentations.					3	
CO3	DEVELOP technical content and documentation such as project abstracts, manuals, blogs, and software-related documents.					3	
CO4	PREPARE software project documents and use digital collaboration tools for professional and academic activities.					3	



Course Contents			
Unit 1	Expressing Ourselves in Mother Tongue	3 Hrs.	CO1
Effective oral and written communication in mother tongue, Self-introduction, Group discussion and public speaking, Formal letters, Notices and applications, Vocabulary for science and technology, Translation of common technical terms into mother tongue, Importance of regional language in professional and social life			
Exemplars: • Writing formal applications and notices in mother tongue • Delivering self-introduction and public speeches • Translation of technical terms into regional language • Preparing vocabulary lists related to science and technology Case Studies: • Case study on effective communication in regional language during social interaction • Case study on the importance of mother tongue in education and professional life			
Unit 2	Fundamentals of Technical and Professional Communication	4 Hrs.	CO2
Principles of technical writing, Clarity, accuracy, brevity and coherence, Grammar, punctuation and sentence construction, Official email writing and correspondence, Resume/CV writing, Interview skills and group discussion techniques, Professional profile creation on LinkedIn.			
Exemplars: • Drafting professional emails and official correspondence • Preparing resumes/CVs and LinkedIn profiles • Conducting mock interviews and group discussions • Creating professional presentation slides Case Studies: • Case study on professional communication in workplace environments • Case study on resume screening and interview performance techniques			
Unit 3	Technical Writing and Content Development	4 Hrs.	CO3
Project abstract writing, technical report writing, User manuals and installation guides, writing code comments and README files, Preparation of meeting agenda and minutes, Documentation standards in projects, Blog writing on technology topics, writing for websites and mobile apps, Proposal writing for startup ideas and innovation pitching			
Exemplars: • Writing project abstracts and technical reports • Preparing user manuals and installation guides • Writing code comments and README documentation • Creating blogs on emerging technology topics Case Studies: • Case study on technical documentation practices in software companies • Case study on startup proposal writing and innovation pitching			
Unit 4	Software Project Documentation and Digital Presentation Skills	4 Hrs.	CO4



Need of documentation in software projects, Overview of SDLC, Software Requirement Specification (SRS) basics, Project proposal writing, Bug report writing, Test report writing, Release notes and version documentation, Report writing tools and formatting techniques, PPT design principles and effective slide creation, Seminar presentation skills, Digital etiquette and online professionalism, Responsible use of AI-assisted writing tools, Introduction to GitHub, Basics of Competitive Programming, Flex/Logo/Poster design basics

Exemplars:

- Preparing Software Requirement Specification (SRS) documents
- Writing bug reports, test reports, and release notes
- Designing technical posters, logos, and presentations
- Creating GitHub repositories with documentation

Case Studies:

- Case study on documentation workflow in software development projects
- Case study on collaborative project management using GitHub
- Case study on responsible use of AI-assisted writing tools in academics and industry

List of Practical

Experiment no. 1	Effective Oral Communication and Public Speaking in Mother Tongue	2 Hrs.	CO1
Problem Statement: Prepare and deliver a self-introduction and short speech in mother tongue on a technical or social topic. Associated Task: Deliver self-introduction and short speech in regional language Exemplars: Public speaking, verbal communication Utilities/Tools: PPT, microphone			
Experiment no. 2	Formal Application, Notice, and Letter Drafting	2 Hrs.	CO1
Problem Statement: Draft formal applications, notices, and official letters for academic and professional communication. Associated Task: Prepare formal written communication documents Exemplars: Applications, notices, official letters Utilities/Tools: MS Word, Google Docs			
Experiment no. 3	Professional Resume and LinkedIn Profile Development	2 Hrs.	CO2
Problem Statement: Create a professional resume/CV and LinkedIn profile for internship and placement opportunities. Associated Task: Create professional profile documents Exemplars: Resume/CV, LinkedIn profile Utilities/Tools: MS Word, LinkedIn			
Experiment no. 4	Official Email Writing and Group Discussion Practice	2 Hrs.	CO2
Problem Statement: Draft professional emails and participate in group discussion activities on technical topics. Associated Task: Draft professional emails and participate in GD activity Exemplars: Email communication, group discussion Utilities/Tools: Gmail, presentation tools			
Experiment no. 5	Project Abstract and Technical Report Preparation	2 Hrs.	CO3



Problem Statement: Prepare a project abstract and technical report for a mini project using standard documentation practices. Associated Task: Prepare technical project documentation Exemplars: Project abstract, technical report Utilities/Tools: MS Word, report templates			
Experiment no. 6	UI/UX Design Prototype Creation using tools	2 Hrs.	CO3
Problem Statement: Design a simple user interface prototype for a website or mobile application using UI/UX design principles. Associated Task: Create wireframes and interface prototypes for digital applications Exemplars: Login page design, mobile app interface, dashboard prototype Utilities/Tools: Figma, Canva, Adobe XD or similar UI/UX design tools			
Experiment no. 7	Technology Blog and Technical Content Writing	2 Hrs.	CO3
Problem Statement: Write a technical blog/article on emerging technologies for digital platforms. Associated Task: Write blog/article on emerging technology topics Exemplars: Technical blog, web content Utilities/Tools: Blogger, Word processor			
Experiment no. 8	Software Requirement Specification and Project Proposal Writing	2 Hrs.	CO4
Problem Statement: Prepare a Software Requirement Specification (SRS) document and project proposal for a software system. Associated Task: Prepare SRS and software project proposal Exemplars: SRS document, proposal report Utilities/Tools: Documentation templates			
Experiment no. 9	Bug Reporting and Software Test Documentation	2 Hrs.	CO4
Problem Statement: Create bug reports and software test reports for identifying and documenting software issues. Associated Task: Create testing and maintenance reports Exemplars: Bug report, test report Utilities/Tools: Testing/report tools			
Experiment no. 10	Technical Presentation and GitHub Repository Documentation	2 Hrs.	CO4
Problem Statement: Create a technical presentation and maintain project documentation using GitHub repository tools. Associated Task: Create presentation and upload project documentation Exemplars: PPT presentation, GitHub repository Utilities/Tools: PowerPoint, GitHub, Canva			
Experiment no. 11	AI-Assisted Technical Writing and Documentation	2 Hrs.	CO4
Problem Statement: Exploring the Use of Artificial Intelligence Tools for Effective Technical Writing and Professional Documentation Associated task: - Select any one AI tool such as ChatGPT, Google Gemini, or Grammarly. - Choose any engineering-related topic of your interest.			



- Use the selected AI tool to:
 - Generate technical content
 - Improve grammar and sentence formation
 - Organize the document professionally
 - Create headings, summaries, or conclusions
 - Prepare a technical report containing:
 - Introduction to the AI tool
 - Engineering topic details
 - Screenshots/prompts used with the AI tool
 - Advantages and limitations of the tool
 - Ethical considerations in AI usage
 - Conclusion and learning outcomes
 - Submit the final report in PDF format.

Exemplars: AI-generated technical report, AI-assisted content editing, professional documentation using AI tools

Utilities/Tools: ChatGPT, Google Gemini, Grammarly, MS Word/PDF editor

Textbooks:

1. M. Ashraf Rizvi, Priyadarshi Patnaik, Effective Technical Communication, 3rd Edition, McGraw Hill Education, 2024.
2. Meenakshi Raman, Sangeeta Sharma, Technical Communication: Principles and Practice, 3rd Edition, Oxford University Press, 2015.
3. Meenakshi Raman, Sangeeta Sharma, Fundamentals of Technical Communication, Oxford University Press, 2011.

Reference Books:

1. Gerald J. Alfred, Charles T. Brusaw, Walter E. Oliu, *Handbook of Technical Writing*, 12th Edition, Bedford/St. Martin's, 2019.
2. William Zinsser, *On Writing Well*, 30th Anniversary Edition, HarperCollins Publishers, 2006.
3. Phillip A. Laplante, *Technical Writing: A Practical Guide for Engineers and Scientists*, 2nd Edition, CRC Press, 2012.
4. Markel M. H., *Technical Communication*, 11th Edition, Bedford/St. Martin's, 2015.

MOOCs Links and additional reading material:

1. Technical Communication for Engineers – www.onlinecourses.nptel.ac.in/noc24_ge37/preview
2. Technical English for Engineers – www.onlinecourses.nptel.ac.in/noc24_hs175/preview
3. Introduction to Technical Writing – www.coursera.org/learn/technical-writing-introduction
4. Technical Communication Specialization – www.coursera.org/specializations/technical-communication
5. Writing Effective Technical Documentation – www.coursera.org/learn/writing-effective-technical-documentation-course
6. Google Technical Writing Courses – www.developers.google.com/tech-writing
7. SWAYAM Online Courses – www.swayam.gov.in
8. GitHub Documentation and README Examples – www.github.com
9. LinkedIn Learning – Professional Communication Courses – www.linkedin.com/learning
10. Technical Writing and Programming Articles – www.geeksforgeeks.org



11. Purdue OWL Writing Resources – www.owl.purdue.edu
 12. Atlassian Documentation Guidelines –
www.atlassian.com/software/confluence/resources/guides/documentation

Strength of CO-PO-PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	1	1	–	–	1	–	1	2	3	1	1	3	1	1
CO2	1	2	1	1	2	–	1	2	3	2	1	3	2	2
CO3	1	2	2	1	2	–	1	2	2	2	1	3	2	2
CO4	1	2	2	2	3	–	1	2	2	2	2	3	3	2
Avg.	1	1.75	1.67	1.33	2	–	1	2	2.5	1.75	1.25	3	2	1.75

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

This course serves as a prerequisite for / maps with the following future courses:

This course provides foundational communication, technical writing, documentation, and presentation skills required for advanced courses such as Software Engineering, Web Development, UI/UX Design, Project Work, and Technical Seminar. It also supports future learning in software documentation, professional communication, research, entrepreneurship, internship training, and collaborative software development practices.

Job Mapping:

- Technical Content Writer
- Documentation Assistant
- Technical Support Executive
- Digital Content Creator



Program:		S. Y. B. Tech (Computer Engineering)		Semester:		IV	
Course:		Entrepreneurship and Startup Fundamentals		Course Code:		R25-CO-EEM-218	
Teaching Scheme (Hrs./week)			Credits (TH)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
2	-	-	2	50	-	-	-
Course Description: The course Entrepreneurship and Startup Fundamentals introduces students to the basics of entrepreneurship, innovation, and startup development. It helps students develop creativity, opportunity recognition, business modeling, and problem-solving skills. The course covers startup ecosystems, marketing, branding, funding, and business communication through practical applications and case studies. It prepares students to identify business opportunities and develop innovative startup ideas.							
Course Relevance: Prepares students to start and grow new ventures, driving innovation and economic growth.							
Prerequisite: Basic understanding of business, communication, and problem-solving concepts.							
Bridge Content: Overview of business types, digital awareness, basic economics , and market analysis fundamentals.							
Course Objectives: • To INTRODUCE students to the fundamentals of entrepreneurship and startup ecosystems. • To DEVELOP creativity, innovation, and opportunity recognition skills among students • To enable students to DESIGN and validate basic business models for startups. • To FAMILIARIZE students with startup marketing, branding, financial planning, and business communication.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome					Bloom's Level	
CO1	EXPLAIN the concepts of entrepreneurship, entrepreneurial traits, and startup ecosystems..					2	
CO2	APPLY creativity and opportunity recognition techniques to generate innovative business ideas.					3	
CO3	DEVELOP business models using market research and validation techniques.					6	
CO4	DEMONSTRATE startup marketing, branding, financial planning, and business communication skills.					3	
Course Contents							
Unit 1	Introduction to Entrepreneurship & Startup ecosystem			8 Hrs.		CO1	
Introduction to entrepreneurship, importance in economic development, innovation, employment generation, and GDP growth. Characteristics of entrepreneurs: risk-taking, creativity, leadership, decision-making, perseverance, and resilience. Types of entrepreneurship: startup, social, intrapreneurship, and small business entrepreneurship. Forms of business organization: sole proprietorship, partnership, private limited company, and public limited company.							



Entrepreneurial mindset: growth mindset, adaptability, initiative-taking, and problem-solving skills. Overview of startup ecosystem: incubators, accelerators, angel investors, venture capitalists, and government startup initiatives.			
Exemplars / Practical Applications and usage/Case Studies:			
• Case Study of successful startups and entrepreneurs in India.			
Unit 2	Idea Generation & Opportunity Recognition	8 Hrs.	CO2
Importance of creativity and innovation in entrepreneurship. Entrepreneurial thinking and idea generation techniques : brainstorming, mind mapping, observation methods, and basics of design thinking. Understanding customer needs: identification of customer pain points, problem-to-solution approach, and opportunity identification. Opportunity evaluation : feasibility analysis, market demand analysis, and profitability analysis. Business idea validation tools: SWOT analysis, market trend analysis, and competitor analysis.			
Exemplars / Practical Applications and usage/Case Study:			
Case Study: Zepto solved the problem of local delivery and daily task management by offering quick hyperlocal delivery services through a mobile application.			
Unit 3	Business Model Development and Startup Validation	8 Hrs.	CO3
Introduction to business models and their importance for startups. Business Model Canvas (BMC) components : value proposition, customer segments, revenue streams, channels, customer relationships, and cost structure. Fundamentals of market research : customer surveys, interviews, feedback collection, and data interpretation. Startup validation : Minimum Viable Product (MVP), prototype development, feedback loops, and iteration. Introduction to financial planning : basic costing concepts, fixed and variable costs, break-even analysis, and revenue planning.			
Exemplars / Practical Applications and usage/ Case study:			
• Design a simple Business Model Canvas for a startup idea. • Conduct and investigate customer surveys to validate business ideas.			
Unit 4	Startup Marketing, Branding and Business Communication	8 Hrs.	CO4
Introduction to branding and positioning, Brand identity: name, logo, brand voice, and values. Positioning strategies and building customer trust and brand image. Digital marketing concepts: traditional vs digital marketing, social media platforms for startups, content marketing, organic and paid marketing, and basics of SEO. Customer acquisition strategies : customer journey, referral marketing, influencer marketing, email marketing, and lead generation. Business communication and pitching : elevator pitch, storytelling, verbal and non-verbal communication skills. Introduction to startup funding methods : bootstrapping, angel investment, venture capital, and crowdfunding.			
Exemplars / Practical Applications and usage/Case Study: Design a logo and brand identity for a startup.			
Textbooks:			
1. Bygrave, W.D., Zacharakis, A., & Corbett, A.C. Entrepreneurship, 6th Edition, Wiley, 2025. ISBN: 9781394262809. 2. Osterwalder, Alexander & Pigneur, Yves. Business Model Generation: A Handbook for Visionaries,			



Game Changers, and Challengers, 1st Edition, Wiley, 2010. ISBN: 9780470876411.

Reference Books:

- Ries, Eric. The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses, 1st Edition, Crown Business, 2011. ISBN: 9780307887894.
- Kuwasaki, Guy. The Art of the Start 2.0: The Time-Tested, Battle-Hardened Guide for Anyone starting Anything, Portfolio (Penguin Random House), 2015. ISBN: 9781591847847.

MOOCs Links and additional reading material:

1. Entrepreneurship Essentials By Prof. Manoj Kumar Mondal IIT Kharagpur
https://onlinecourses.nptel.ac.in/noc20_ge08/preview
2. Entrepreneurship By Prof. C Bhaktavatsala Rao IIT Madras
https://onlinecourses.nptel.ac.in/noc21_mg70/preview
1. NPTEL Course: Entrepreneurship and IP Strategy
https://onlinecourses.nptel.ac.in/noc20_mg35
2. A Guide to Entrepreneurship for Beginners - <https://www.coursera.org/learn/entrepreneur-guide-beginners>

Strength of CO-PO-PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	1	2	-	-	-	2	1	-	-	-	-	1	1	3
CO2	1	3	2	2	-	2	-	-	-	-	-	2	3	3
CO3	2	3	3	2	2	-	-	-	-	2	3	2	3	3
CO4	1	2	2	-	2	-	1	-	2	3	2	2	2	3
Avg.	1	2.5	2.3	2	2	2	1	-	1	2.5	2.5	1.7	2.25	3

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

Assessment Pattern

BT Level	CCE			
	CO1	CO2	CO3	CO4
	12	13	12	13
L1	√	√	√	√
L2	√	√	√	√
L3	—	√	√	√
L4	—	—	√	—
L5	—	—	√	—
L6	—	—	√	—



This course serves as a prerequisite for / maps with the following future courses:

This course serves as a foundation for advanced courses related to entrepreneurship, innovation, business management, and startup development. It supports future learning in areas such as Entrepreneurship Development, Startup and Innovation Management, Digital Marketing, Business Planning, Financial Management, Design Thinking, Business Analytics, Strategic Management, and Project Management. The course also prepares students for entrepreneurship labs, incubation activities, and startup-based interdisciplinary projects.

Job Mapping:

- Entrepreneur / Startup Founder
- Business Development Executive
- Innovation Associate
- Startup Coordinator
- Digital Marketing Executive



Program:	S. Y. B. Tech (Computer Engineering)			Semester:	IV		
Course:	Operating System Lab			Course Code:	R25-CO-PCC-219		
Teaching Scheme (Hrs./week)			Credits (PR)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
-	2	-	1	40	60	-	25

Course Description :

The Operating System Laboratory course provides hands-on experience with fundamental concepts of operating systems through the implementation of various algorithms and system programs. The course focuses on practical understanding of CPU scheduling, process management, process synchronization, deadlock handling, memory management, file system operations, and disk scheduling techniques.

Students implement both preemptive and non-preemptive scheduling algorithms, explore process creation and control using system calls, and solve classical synchronization problems such as Producer–Consumer, Reader–Writer, and Dining Philosophers. The course also covers deadlock avoidance using Banker’s algorithm, memory allocation strategies, page replacement algorithms, and directory operations in Linux environments.

Through these practical exercises, students develop the ability to analyze operating system mechanisms, implement system-level programs, and understand resource management techniques used in modern computing systems.

Course Relevance:

The Operating System Laboratory course is essential for understanding how computer systems manage hardware and software resources efficiently. It provides practical knowledge of key operating system concepts such as process management, CPU scheduling, synchronization, deadlock handling, memory management, file systems, and disk scheduling.

Through implementation of various algorithms and system calls, students gain hands-on experience in developing and analyzing system-level programs. This course helps students understand the internal working of operating systems and enhances their problem-solving and programming skills.

The knowledge gained from this course is highly relevant for software development, system programming, networking, cloud computing, and operating system design. It also prepares students for advanced studies, research, and careers in the field of computer science and engineering.

Prerequisite: Data Structure, Digital Electronics

Bridge Content:**Course Objectives: The course aims to:**

- To understand the fundamental concepts of operating systems including process management, memory management, and file systems.
- To study CPU scheduling algorithms and analyze their performance using preemptive and non-preemptive techniques.
- To learn process creation, synchronization, and inter-process communication mechanisms in operating systems.



- To understand deadlock concepts and techniques for prevention and avoidance using algorithms like Banker's algorithm and synchronization problems.
- To implement memory management, page replacement, disk scheduling, and directory management techniques through programming.

Course Outcomes: After learning the course, students will be able to:

CO	Course Outcome	Bloom's Level
CO1	EXPLAIN the concepts of process management, CPU scheduling, and operating system services	2
CO2	IMPLEMENT preemptive and non-preemptive CPU scheduling algorithms and analyze their performance	3
CO3	DEVELOP programs for process creation, threads, and synchronization problems such as Producer-Consumer, Reader-Writer, and Dining Philosophers	3
CO4	ANALYZE deadlock conditions and apply Banker's algorithm for deadlock avoidance	4
CO5	IMPLEMENT memory allocation, page replacement, disk scheduling, and directory operations in an operating system environment.	3

List of Practical:

Group A: Perform any 3 experiments.

Experiment No.1	Write a program to perform basic shell scripting operations in Linux	2 Hrs.	CO1
-----------------	--	--------	-----

Objective:

To understand and implement basic shell scripting using different Linux shells such as Bash, sh, zsh, and ksh.

Associated Task:

Write and execute shell scripts using different shell environments to perform basic operations, decision making, looping, and file handling.

Exemplars:

- Write and execute a simple "Hello World" script in Bash, sh, zsh, and ksh
- Perform arithmetic operations using shell scripts
- Use input and output commands in shell scripting
- Implement conditional statements using if-else
- Implement loops using for and while
- Create and manipulate files/directories using shell commands
- Compare execution of scripts in different shells

Utility:

Helps in understanding:

- Basics of Linux shell environments
- Shell script execution and syntax
- Automation of system tasks
- Command-line operations in Linux
- Differences between Bash, sh, zsh, and ksh shells



Operating system service interaction through scripting			
Experiment No.2	Write a program to simulate CPU scheduling using preemptive and non-preemptive scheduling algorithms	2 Hrs.	CO2
Objective: To simulate CPU scheduling using both preemptive and non-preemptive scheduling algorithms. Associated Task: Implement different CPU scheduling algorithms and calculate: - Waiting Time - Turnaround Time - Average Waiting Time - Average Turnaround Time Exemplars: Non-Preemptive Scheduling Algorithms: - First Come First Serve (FCFS) - Non-Preemptive Shortest Job First (SJF) - Non-Preemptive Priority Scheduling Preemptive Scheduling Algorithms: - Preemptive Shortest Job First (SRTF) - Round Robin Scheduling Algorithm - Preemptive Priority Scheduling Utility: Helps in understanding: - CPU process scheduling concepts - Difference between preemptive and non-preemptive execution - Context switching and CPU allocation - Process execution order and response handling - Performance analysis of scheduling algorithms using waiting time and turnaround time calculations.			
Experiment No.3	Write a program to implement process control and thread detection in Linux/Java	2 Hrs.	CO3
Objective: To implement process control and thread detection using Java . Associated Task: - Create and manage processes/threads - Detect and display active threads - Demonstrate basic process and thread control mechanisms Exemplars: - Process creation using ProcessBuilder - Thread creation using Thread class - Thread detection using Thread.activeCount() and Thread.enumerate() Utility: Helps in understanding process management, multithreading, and concurrency control, which are essential concepts in operating systems and Java programming.			
Experiment No.4	Write a program to demonstrate the creation of parent and child processes using the fork()	2Hrs.	CO3



system call			
Objective: To demonstrate the creation of parent and child processes using the fork() system call . Associated Task: • Create a child process using fork() • Identify parent and child processes using process IDs • Display execution of both processes Exemplars: • Process creation using fork() • Parent–Child relationship in Linux • Process identification using getpid() and getppid() Utility: Helps in understanding process creation, process hierarchy, and basic process control in Linux operating systems.			
Group B: Perform any 3 experiments.			
Experiment No.1	Write a program to implement the Producer–Consumer problem using process synchronization techniques	2 Hrs.	CO3
Objective: To implement the Producer–Consumer problem using process synchronization techniques to avoid race conditions Associated Task: • Implement producer and consumer processes • Use synchronization mechanisms (semaphores) • Ensure mutual exclusion while accessing the shared buffer Exemplars: • Synchronization using semaphores • Critical section handling • Inter-process communication Utility: Helps in understanding process synchronization, concurrency control, and prevention of race conditions in operating systems.			
Experiment No.2	Write a program to implement the Reader–Writer problem using synchronization mechanisms	2 Hrs.	CO3
Objective: To implement the Reader–Writer problem using synchronization mechanisms to manage concurrent access to shared data. Associated Task: • Implement multiple reader and writer processes • Allow concurrent reading • Ensure exclusive access for writers using synchronization primitives Exemplars: • Synchronization using semaphores • Handling critical sections • Solving concurrency problems in operating systems			



Utility: Helps in understanding process synchronization, mutual exclusion, and concurrent data access control, which are fundamental concepts in operating systems.

Experiment No.3	Write a program to implement the Dining Philosophers problem to demonstrate deadlock and synchronization	2 Hrs.	CO3
------------------------	---	---------------	------------

Objective: To implement the Dining Philosophers problem to demonstrate deadlock and the use of synchronization mechanisms for resource sharing.

Associated Task:

- Create multiple philosopher threads
- Control access to shared resources (chopsticks)
- Prevent deadlock using synchronization techniques

Exemplars:

- Synchronization using mutex locks
- Critical section handling
- Deadlock prevention in concurrent systems

Utility: Helps in understanding deadlock, mutual exclusion, resource sharing, and synchronization concepts in operating systems

Experiment No.4	Write a program to implement deadlock avoidance using Banker's Algorithm.	2 Hrs.	CO4
------------------------	--	---------------	------------

Objective: To implement deadlock avoidance in a multi-process system using Banker's Algorithm and ensure safe resource allocation.

Associated Task:

- Take input of processes, resources, maximum demand, and allocation
- Compute the need matrix
- Determine if the system is in a safe state
- Allocate resources safely to avoid deadlock

Exemplars:

- Banker's Algorithm for deadlock avoidance
- Resource allocation and safe state determination
- Process and resource management in operating system

Utility: Helps in understanding deadlock avoidance, safe state determination, and ensures reliable resource allocation in operating systems.

Group C: Perform any 3 experiments.

Experiment No.1	Write a Program to Simulate Memory Allocation Techniques: First Fit, Best Fit, and Worst Fit	2 Hrs.	CO5
------------------------	---	---------------	------------

Objective: To simulate memory allocation techniques (First Fit, Best Fit, Worst Fit) for processes in main memory.

Associated Task:



- Input the number of memory blocks and processes
- Allocate memory to processes using different strategies
- Display allocated block for each process and fragmentation

Exemplars:

- First Fit: Allocate the first block that is large enough
- Best Fit: Allocate the smallest block that fits the process
- Worst Fit: Allocate the largest available block

Utility: Helps in understanding memory management, fragmentation, and efficiency of different allocation strategies in operating systems

Experiment No.2	Write a Program to Stimulate Page Replacement Algorithms: FIFO, LIFO, and Optimal	2 Hrs.	CO5
------------------------	--	---------------	------------

Objective: To simulate page replacement algorithms (FIFO, LIFO, Optimal) and understand page faults in memory management

Associated Task:

- Take page reference string input and number of frames
- Implement FIFO, LIFO, and Optimal page replacement
- Calculate and display page faults for each algorithm

Exemplars:

- Take page reference string input and number of frames
- Implement FIFO, LIFO, and Optimal page replacement
- Calculate and display page faults for each algorithm

Utility: Helps in understanding virtual memory management, page replacement policies, and efficiency of memory utilization in operating systems

Experiment No.3	Write a Program to Stimulate Directory Structure Operations using mkdir(), rmdir(), opendir(), and readdir()	2 Hrs.	CO5
------------------------	---	---------------	------------

Objective: To demonstrate directory structure operations in Linux using mkdir(), rmdir(), opendir(), and readdir() system calls

Associated Task:

- Create and remove directories using mkdir() and rmdir()
- Open directories and list their contents using opendir() and readdir()

Exemplars:

- mkdir(): Create a new directory
- rmdir(): Remove an empty directory
- opendir(): Open a directory stream
- readdir(): Read entries (files/directories) from a directory stream

Utility: Helps in understanding file system management, directory navigation, and basic Linux filesystem operations, which are essential for operating system concepts and system programming.

Experiment No.4	Write a Program to Stimulate Disk Scheduling Algorithms: FIFO, SSTF, SCAN, C-SCAN, LOOK, and C-LOOK	2 Hrs.	CO5
------------------------	--	---------------	------------



Objective: To simulate disk scheduling algorithms (FIFO, SSTF, SCAN, C-SCAN, LOOK, and C-LOOK) and calculate total head movement for servicing disk requests.

Associated Task:

- Take input of disk requests and initial head position
- Implement different disk scheduling algorithms
- Calculate and display total head movement for each algorithm

Exemplars:

- FIFO (First In First Out): Serve requests in the order they arrive
- SSTF (Shortest Seek Time First): Serve the request closest to the current head
- SCAN (Elevator Algorithm): Head moves in one direction servicing requests until end, then reverses
- C-SCAN (Circular SCAN): Head moves in one direction; jumps to start after reaching end
- LOOK: Like SCAN but reverses direction at the last request, not end
- C-LOOK: Like C-SCAN but jumps to the first request instead of disk start

Utility: Helps in understanding disk scheduling, seek time optimization, and efficient utilization of disk I/O in operating systems.

Strength of CO-PO-PSO Mapping														
CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	3	2	–	–	1	–	–	–	–	–	1	2	2	2
CO2	3	2	1	1	2	–	–	–	–	–	1	3	3	3
CO3	2	2	2	1	3	–	–	1	–	–	1	3	3	3
CO4	2	3	1	2	2	–	–	–	–	–	1	2	2	2
CO5	2	2	1	1	3	–	–	–	–	–	1	3	3	3
Avg.	3	2	1	1	2	–	–	1	–	–	1	3	3	3

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

This course serves as a prerequisite for / maps with the following future courses:

This course serves as a foundation for advanced courses related to operating systems, system programming, cloud computing, cybersecurity, distributed systems, and system-level software development. It supports future learning in areas such as Advanced Operating Systems, Linux System Administration, Cloud Computing, Virtualization, Network Security, Distributed Computing, Embedded Systems, Parallel Computing, and DevOps.

Job Mapping:

- System Administrator
- Technical Support Engineer
- Linux Support Engineer
- Backend Developer
- Cloud Support Associate



Program:		S. Y. B. Tech (Computer Engineering)		Semester:		IV	
Course:		Database Management Systems Lab		Course Code:		R25-CO-PCC-220	
Teaching Scheme (Hrs./week)			Credits (PR)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
-	2	-	1	-	-	-	25
Course Description: The Database Management Laboratory provides hands-on experience in database design and implementation using relational and NoSQL databases. Students practice ER modeling, SQL programming, transaction control, database objects, and MongoDB operations, and apply these concepts in a mini-project following SDLC principles.							
Course Relevance: This course equips students with essential database skills required for modern software applications. It strengthens backend development capabilities, supports data-driven decision-making, and prepares students for industry, internships, and advanced studies.							
Prerequisite: Basic knowledge of DBMS concepts, SQL fundamentals, data models, and elementary programming concepts is required.							
Bridge Content: Revision of DBMS fundamentals, ER diagrams, relational mapping, basic SQL operations, integrity constraints, transactions, and an introduction to NoSQL databases and database tools.							
Course Objectives: 1. To UNDERSTAND and apply the concepts of database design by formulating case studies, creating E-R diagrams, and mapping them to the relational model. 2. To DEVELOP and execute SQL queries for creating, modifying, and managing database structures using DDL, DML, DCL, and TCL commands. 3. To IMPLEMENT advanced SQL operations, including aggregate functions, joins, subqueries, views, stored procedures, and triggers, for efficient database management. 4. To EXPLORE NoSQL databases by designing and implementing CRUD operations in MongoDB, understanding document-based storage and retrieval.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome						Bloom's Level
CO1	DESIGN database structures by creating E-R diagrams and converting them into relational models.						3
CO2	EXECUTE SQL queries for data manipulation, transaction control, and access management using DML, DCL, and TCL commands.						3
CO3	ANALYZE SQL operations, including joins, views, subqueries, stored procedures, and triggers, to optimize data retrieval and integrity.						4
CO4	IMPLEMENT CRUD operations in MongoDB to demonstrate NoSQL database concepts and their practical applications.						3
CO5	CREATE a database-driven application by integrating relational and NoSQL databases following software development life cycle principles.						6



List of Practical:			
Experiment No.1	Case Study and ER Diagram Develop a case study and design its Entity-Relationship (ER) Diagram. Convert the ER model into a relational model.	2 Hrs.	CO1
Objective: To apply database design concepts by developing a case study, creating an ER diagram, and converting it into a relational model. Associated Task: - Identify entities, attributes, and relationships - Draw ER diagram using standard notations - Convert ER model into relational schema with keys Exemplars: Student Management System, Library System, Hospital Database Utility: Helps in understanding systematic database design for real-world applications.			
Experiment No.2	Write and execute SQL Data Definition Language (DDL) commands such as CREATE, ALTER, DROP, RENAME, and TRUNCATE to define and modify tables. Insert data into the tables and apply appropriate integrity constraints such as NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, and CHECK. (The application may vary as per the subject teacher's requirement.)	2 Hrs.	CO2
Objective: To define and modify database structures using SQL DDL commands and apply integrity constraints. Associated Task: - Execute CREATE, ALTER, DROP, RENAME, and TRUNCATE commands - Insert records into tables - Apply NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, and CHECK constraints Exemplars: Employee Database, Course Registration System Utility: Ensures strong foundation in database schema creation and data integrity enforcement.			
Experiment No.3	SQL Queries for Data Manipulation, Access Control, and Transactions Design and run SQL queries to demonstrate the following: a) Data Manipulation (DML): Use SQL statements to INSERT, UPDATE, and DELETE records. Apply arithmetic, logical, set operators, pattern matching, and string functions. b) Access Control (DCL): Use GRANT, REVOKE, and ROLE commands to manage user access. c) Transaction Control (TCL): Apply START TRANSACTION, COMMIT, ROLLBACK, and SAVEPOINT commands to manage transactions.	2 Hrs.	CO2



Objective: To develop and execute SQL queries for data manipulation, access control, and transaction management. Associated Task: • Perform INSERT, UPDATE, DELETE operations • Use arithmetic, logical, set operators, pattern matching, and string functions • Implement GRANT, REVOKE, ROLE commands • Demonstrate COMMIT, ROLLBACK, SAVEPOINT Exemplars: Banking System, Student Fee Database Utility: Builds practical skills in secure and reliable database operations.			
Experiment No.4	Aggregate Functions and Grouping Use aggregate functions along with GROUP BY and HAVING clauses to retrieve summarized data from the database.	2 Hrs.	CO3
Objective: To retrieve summarized data using aggregate functions and grouping techniques. Associated Task: • Use COUNT, SUM, AVG, MIN, MAX • Apply GROUP BY and HAVING clauses Exemplars: Sales Database, Examination Result Analysis Utility: Supports analytical reporting and decision-making using databases.			
Experiment No.5	JOIN Operations and Views Perform various types of JOIN operations to extract meaningful relationships between tables. Create and manage different database views.	2 Hrs.	CO3
Objective: To perform JOIN operations and create database views for efficient data retrieval. Associated Task: • Implement INNER, LEFT, RIGHT, and FULL JOINS • Create, update, and drop views Exemplars: Inventory System, Order Management System Utility: Enhances understanding of relational data connectivity and abstraction.			
Experiment No.6	Subqueries Write and execute subqueries to retrieve data from one table based on results from another.	2 Hrs.	CO3
Objective: To retrieve data using subqueries for complex query execution. Associated Task: • Write single-row and multi-row subqueries • Use nested queries with conditional expressions Exemplars: Employee Salary Database, Academic Performance System Utility: Improves query-writing efficiency for complex data retrieval scenarios.			
Experiment No.7	Stored Procedures or Function with Cursors Create and execute stored procedures / function using cursors.	2 Hrs.	CO3



Objective: To create and execute stored procedures or functions using cursors. Associated Task: • Develop stored procedures/functions • Implement cursor-based operations Exemplars: Payroll Processing System Utility: Encourages modular, reusable, and efficient database programming.			
Experiment No.8	Database Triggers : Implement and test triggers to maintain data integrity in database.	2 Hrs.	CO3
Objective: To implement database triggers for maintaining data integrity. Associated Task: • Create BEFORE and AFTER triggers • Test triggers on INSERT, UPDATE, DELETE operations Exemplars: Audit Log System, Inventory Control Database Utility: Automates integrity enforcement and business rules in databases.			
Experiment No.9	CRUD Operations using MongoDB Design and implement basic Create, Read, Update, and Delete (CRUD) operations using MongoDB. Use the save method and logical operators where necessary.	2 Hrs.	CO4
Objective: To design and implement CRUD operations using MongoDB. Associated Task: • Perform Create, Read, Update, Delete operations • Use logical operators and save method Exemplars: User Profile Database, Product Catalog Utility: Introduces NoSQL concepts used in modern web and cloud applications.			
Experiment No.10	Aggregation and Indexing in MongoDB Design and execute MongoDB queries using aggregation and indexing techniques with suitable examples.	2 Hrs.	CO4
Objective: To implement aggregation and indexing techniques in MongoDB. Associated Task: • Use aggregation pipeline stages • Create and apply indexes for performance optimization Exemplars: E-commerce Analytics Database Utility: Enhances performance optimization and data analysis skills in NoSQL databases.			
Experiment No.11	Using the database concepts covered in above assignments, develop an application with following details: Engineering Course throughout the implementation. 2. Develop application considering: • Front End: Java/Perl/PHP/Python/Ruby/.net/any other language • Backend : MongoDB/ MySQL/Oracle 3. Test and validate application using Manual/Automation testing.	2 Hrs.	CO5



	4. Student should develop application in group of 2-3 students and submit the Project Report which will consist of documentation related to different phases of Software Development Life Cycle		
Objective: To design and develop a database-driven application by applying relational and NoSQL database concepts and following the Software Development Life Cycle (SDLC). Associated Task: • Develop an application using any suitable front-end technology • Implement backend using MongoDB / MySQL / Oracle • Apply database concepts such as ER modelling, SQL/CRUD operations • Perform manual or automated testing • Prepare and submit a project report covering SDLC phases Exemplars: Student Information System, Online Examination System, Inventory Management System Utility: Enhances practical understanding of end-to-end application development, teamwork, and real-world database implementation skills.			

Strength of CO-PO-PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	3	2	1	1	1	-	-	-	-	-	2	3	2	2
CO2	3	2	1	1	2	-	-	-	-	-	2	3	2	2
CO3	2	2	1	1	2	3	3	-	-	-	2	3	2	2
CO4	2	2	1	1	2	-	-	-	-	-	2	3	2	2
CO5	3	2	1	1	2	3	-	-	-	-	2	3	2	2
Avg.	3	2	1	1	2	3	3	-	-	-	2	3	2	2

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

This course serves as a prerequisite for / maps with the following future courses:

This practical-oriented Database Management Systems course provides hands-on experience in database design, SQL programming, transaction management, relational and NoSQL database operations, and database-driven application development skills required for advanced computing subjects and industry-oriented technologies. The course supports understanding and implementation of modern data management concepts using relational and MongoDB technologies.

Job Mapping:

- SQL Developer
- Database Administrator
- Backend Developer
- Data Analyst
- Software Developer



Program:		S. Y. B. Tech (Computer Engineering)		Semester:		IV	
Course:		Sustainable Technology & Green Computing		Course Code:		R25-CO-VEC-221	
Teaching Scheme (Hrs. / week)			Credits (TH)	Examination Scheme			
Lecture	Practical	Tutorial		CCE	ESE	TW	PR
2	-	-	2	50	-	-	-
Course Description: This course introduces the concepts of sustainable technology and green computing with a focus on energy-efficient, eco-friendly, and environmentally responsible computing practices. It covers green IT, e-waste management, sustainable software and hardware, and the role of emerging technologies in building a sustainable future.							
Course Relevance: The course is highly relevant in today’s technology-driven world where sustainability and environmental responsibility are major global concerns. It helps students understand green computing practices, energy-efficient technologies, and sustainable IT solutions required in industries, smart systems, and modern computing environments. The course also creates awareness about reducing carbon footprint, managing e-waste, and supporting sustainable development through technology.							
Prerequisite: Basic knowledge of computer systems, information technology, and environmental awareness. No prior specialized knowledge is required.							
Bridge Content: The bridge content covers the basic concepts of computer systems, information technology, and environmental sustainability required for understanding green computing practices. It includes fundamentals of energy consumption in computing, awareness about e-waste and environmental impact of technology, and introductory concepts of cloud computing, IoT, and smart technologies.							
Course Objectives: • To introduce the concepts of sustainable technology and green computing. • To understand energy-efficient and eco-friendly computing practices. • To study sustainable solutions and environmental impacts of IT systems. • To promote awareness of green technologies and responsible use of computing resources.							
Course Outcomes: After learning the course, students will be able to:							
CO	Course Outcome						Bloom’s Level
CO1	DESCRIBE concepts of sustainable technology and green computing						2
CO2	IDENTIFY energy-efficient and eco-friendly computing practices.						3
CO3	ANALYZE sustainable solutions using modern technologies.						4
CO4	APPLY green IT practices for sustainable development.						3



Course Contents			
Unit 1	Introduction to Sustainability and Green Computing	7 Hrs.	CO1
Concept of Sustainable Development, Need and Importance of Sustainable Technology, environmental Issues and Climate Change, Introduction to Green Computing, Goals and Principles of Green IT, Carbon Footprint and Energy Consumption in IT, Role of ICT in Sustainable Development. Exemplars: Carbon footprint analysis, Green IT awareness campaign Applications and Usage: Energy conservation in computer labs, Sustainable use of IT resources			
Unit 2	Energy Efficient Computing and Green Hardware	7 Hrs.	CO2
Energy Efficient Computing Concepts, Power Management Techniques, Green Data Centers and Virtualization, Energy Star Ratings and Green Standards, Eco-friendly Hardware Design, E-waste Management and Recycling, Lifecycle of Computing Devices Exemplars: E-waste recycling study, Green data center case study Applications and Usage: Power-efficient computing, Eco-friendly hardware management			
Unit 3	Sustainable Software and Smart Technologies	8 Hrs.	CO3
Green Software Engineering, Sustainable Software Design Principles, Cloud Computing and Sustainability, IoT for Smart Environment Monitoring, Smart Cities and Green Technologies, Renewable Energy Technologies in IT Infrastructure, Case Studies on Sustainable Technology Solutions. Exemplars: IoT-based smart monitoring system, Smart city case study Applications and Usage: Sustainable software solutions, Smart environmental monitoring			
Unit 4	Green Practices, Policies and Future Trends	8 Hrs.	CO4
Green Campus Initiatives, Sustainable IT Policies and Regulations, Environmental Ethics in Computing, Green Networking Concepts, Emerging Trends in Sustainable Technology, AI and Sustainability, Industry Best Practices and Future Scope of Green Computing Exemplars: Green campus audit, AI for sustainability presentation Applications and Usage: Implementation of green IT policies, Adoption of sustainable technologies in industries			
Textbooks:			
1. San Murugesan, G. R. Gangadharan, Harnessing Green IT: Principles and Practices, Wiley Publications. 2. Bud E. Smith, Green Computing Tools and Techniques for Saving Energy, Money and Resources, CRC Press. 3. Bhuvan Unhelkar, Green IT Strategies and Applications, CRC Press. 4. Carl Speshocky, Empowering Green Initiatives with IT, John Wiley & Sons.			
Reference Books:			
1. Bhuvan Unhelkar, Green IT Strategies and Applications, CRC Press. 2. Jason Harris, Green Computing and Green IT Best Practices, Emereo Publications. 3. Woody Leonhard, Green Home Computing for Dummies, Wiley Publishing. 4. Alin Gales, Michael Schaefer, Mike Ebberts, Green Data Center: Steps for the Journey, IBM Redbooks. 5. Arshdeep Bahga, Vijay Madiseti, Internet of Things – A Hands-on Approach,			



Universities Press.

MOOCs Links and additional reading material:

1. Green Computing – www.onlinecourses.nptel.ac.in/noc25_cs56/preview
2. Sustainable Development Goals – www.sdg.un.org/goals
3. Energy Efficient Computing – www.energystar.gov/products
4. Cloud Computing and Sustainability – www.sustainability.google/operating-sustainably
5. E-waste Management – www.cpcb.nic.in/e-waste
6. Internet of Things for Smart Systems – www.coursera.org/learn/iot

Strength of CO-PO-PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2	PSO3
CO1	2	1	-	-	1	2	3	2	-	-	-	2	1	1
CO2	2	2	1	-	2	2	3	2	-	-	-	2	2	1
CO3	2	3	2	1	2	2	3	2	1	-	-	2	2	2
CO4	2	2	2	1	2	3	3	3	1	1	1	3	2	2
Avg.	2.0	2.0	1.7	1.0	1.7	2.2	3.0	2.2	1.0	1.0	1.0	2.2	1.7	1.5

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

Assessment Pattern

BT Level	CCE			
	CO1	CO2	CO3	CO4
	12	13	12	13
L1	√	√	√	√
L2	√	√	√	√
L3	–	√	√	√
L4	–	–	√	–
L5	–	–	–	–
L6	–	–	–	–

This course serves as a prerequisite for / maps with the following future courses: Green IT and Sustainable Computing, Internet of Things (IoT), Smart City Technologies, Cloud Computing, Environmental Informatics, Energy Efficient Systems, Artificial Intelligence for Sustainability, Emerging Technologies and Innovation

Job Mapping:

- Green IT Consultant, Sustainability Analyst
- IoT Developer
- Cloud Engineer
- Software Developer
- System Analyst
- IT Support Engineer