GLOBAL INSTITUTE OF ENGINEERING AND TECHNOLOGY (AUTONOMOUS)
COURSE CATALOGUE
REGULATIONS B.TECH – GR - 25
COMPUTER SCIENCE AND ENGINEERING
II YEAR I SEMESTER

Course Code	Course Name	Subject Area	Category	Periods Per Week			Credits	Scheme of Examination Max Marks		
				L	T	P		CIA	SEE	Total
THEORY										
CS301PC	Discrete Mathematics	PCC	CORE	3	0	0	3	40	60	100
CS302PC	Computer Organization and Architecture	PCC	CORE	3	0	0	3	40	60	100
CS303PC	Object Oriented Programming through java	PCC	CORE	3	0	0	3	40	60	100
CS304PC	Software Engineering	PCC	CORE	3	0	0	3	40	60	100
CS305PC	Database Management Systems	PCC	CORE	3	0	0	3	40	60	100
MS306HS	Quantitative Aptitude and Logical Reasoning	HSMC	Foundation	2	0	0	2	40	60	100
PRACTICAL										
CS307PC	Object Oriented Programming through java Lab	PCC	CORE	0	0	2	1	40	60	100
CS308PC	Software Engineering Lab	PCC	CORE	0	0	2	1	40	60	100
CS309PC	Database Management Systems Lab	PCC	CORE	0	0	2	1	40	60	100
CS310SD	Node Js/React JS/ Django	SDC	Skill	0	0	2	1	40	60	100
			Total Credits	**17**	**0**	**8**	**21**			

COURSE CONTENT

DISCRETE MATHEMATICS								
II Year - I Semester: CSE								
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
CS301PC	**Core**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	**-**	**-**	**3**	**40**	**60**	**100**
Contact Classes: 48	Tutorial Classes: Nil	Practical Classes: Nil				**Total Classes: 48**		
Prerequisite: Mathematics courses of the first year of study.								

1. COURSE OVERVIEW

This course introduces students to the foundational concepts of **Discrete Mathematics**, which form the backbone of computer science and engineering. It equips students with essential tools to reason mathematically, model computational problems, and design efficient algorithms.

The course begins with **Mathematical Logic**, where students learn formal notation, propositional and predicate calculus, and methods of inference to construct precise proofs. It then covers **Set Theory, Relations, and Functions**, which are widely applied in database theory, automata, and programming languages.

Students also explore **Algebraic Structures** such as semigroups, monoids, lattices, and Boolean algebra, which play a vital role in digital logic design, cryptography, and theoretical computer science. The course further develops problem-solving skills through **Combinatorics and Counting Principles**, including permutations, combinations, binomial and multinomial theorems, and the principle of inclusion-exclusion—key tools for algorithm analysis and complexity.

Finally, students are introduced to **Graph Theory**, covering trees, spanning trees, binary trees, planar graphs, Euler and Hamiltonian circuits, chromatic numbers, and the Four-Colour problem, with applications in networking, data structures, and optimisation problems.

2. COURSE OBJECTIVE

The students will try to Learn:

1) Introduces elementary discrete mathematics for computer science and engineering.
2) Topics include formal logic notation, methods of proof, induction, sets, relations, algebraic structures, elementary graph theory, permutations and combinations, counting principles, recurrence relations and generating functions.

3. COURSE OUTCOMES

After successful completion of the course, students should be able to:

CO 1	Understand and construct precise mathematical proofs
CO 2	Apply logic and set theory to formulate precise statements
CO 3	Analyse and solve counting problems on finite and discrete structures
CO 4	Describe and manipulate sequences
CO 5	Apply graph theory in solving computing problems.

4. COURSE CONTENT

UNIT - I Mathematical logic: 10L

Introduction, Statements and Notation, Connectives, Normal Forms, Theory of Inference for the Statement Calculus, The Predicate Calculus, Inference Theory of the Predicate Calculus.

UNIT - II Set theory: 8L

Introduction, Basic Concepts of Set Theory, Representation of Discrete Structures, Relations and Ordering, Functions.

UNIT - III Algebraic Structures: 10L

Introduction, Algebraic Systems, Semigroups and Monoids, Lattices as Partially Ordered Sets, Boolean Algebra.

UNIT - IV Elementary Combinatorics: 10 L

Basics of Counting, Combinations and Permutations, Enumeration of Combinations and Permutations, Enumerating Combinations and Permutations with Repetitions, Enumerating Permutations with Constrained Repetitions, Binomial Coefficient, The Binomial and Multinomial Theorems, The Principle of Exclusion.

UNIT - V Graph Theory: 10L

Basic Concepts, Isomorphism and Subgraphs, Trees and their Properties, Spanning Trees, Directed Trees, Binary Trees, Planar Graphs, Euler's Formula, Multi-graphs and Euler Circuits, Hamiltonian Graphs, Chromatic Numbers, The Four-Colour Problem.

5. TEXT BOOKS

1) Discrete Mathematical Structures with Applications to Computer Science: J.P. Tremblay, R. Manohar, McGraw-Hill, 1st Edition.
2) Discrete Mathematics for Computer Scientists & Mathematicians: Joe L. Mott, Abraham Kandel, Theodore P. Baker, Prentis Hall of India, 2nd Edition.

6. REFERENCE BOOKS

1) Discrete and Combinatorial Mathematics - an applied introduction: Ralph P. Grimald, Pearson Education, 5th edition.
2) Discrete Mathematical Structures: Thomas Kosy, Tata McGraw-Hill Publishing Co.

CO-PO-PSO Mapping

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2
CO 1	**3**	**2**										**2**	**3**	**2**
CO 2	**2**	**3**										**2**	**2**	**3**
CO 3	**2**	**3**											**2**	**3**
CO 4	**3**	**-**	**1**		**2**								**3**	**-**
CO 5	**3**	**3**			**1**							**2**	**3**	**3**

COURSE CONTENT

COMPUTER ORGANIZATION AND ARCHITECTURE								
II Year - I Semester: CSE								
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
CS302PC	**Core**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	**-**	**-**	**3**	**40**	**60**	**100**
Contact Classes: 48	Tutorial Classes: Nil	Practical Classes: Nil				**Total Classes: 48**		
Prerequisites: No prerequisites. **Co-requisite**: A Course on "Digital Electronics".								

1. COURSE OVERVIEW

Computer Organization and Architecture (COA) explores the internal structure and functioning of computer systems. The course focuses on how hardware components are organized and how architectural design choices influence system performance. It covers data representation, processor design, instruction sets, memory organization, input/output mechanisms, and parallel processing concepts. Students gain an understanding of how software interacts with hardware, enabling them to analyze and design efficient computer systems and improve overall performance.

2. COURSE OBJECTIVE

The students will try to Learn:

1) The purpose of the course is to introduce principles of computer organization and the basic architectural concepts.
2) It begins with basic organization, design, and programming of a simple digital computer and introduces simple register transfer language to specify various computer operations.
3) Topics include computer arithmetic, instruction set design, microprogrammed control unit, pipelining and vector processing, memory organization and I/O systems, and multiprocessors.

3. COURSE OUTCOMES

After successful completion of the course, students should be able to:

CO 1	Explain Boolean algebra, logic gates, and data representation techniques, and describe the functional blocks and concepts of digital computers, computer organization, and architecture.
CO 2	Analyze and design combinational and sequential logic circuits using standard components, HDL descriptions, and state-based design techniques.
CO 3	Apply register transfer language and micro-operations to explain instruction execution, control signals, memory reference instructions, and I/O interrupt mechanisms.
CO 4	Analyze CPU organization, instruction formats, addressing modes, microprogrammed control, and arithmetic algorithms for efficient data manipulation and control.
CO 5	Explain input–output organization and memory hierarchy concepts, including DMA, cache memory, and auxiliary storage, to evaluate system performance.

4. COURSE CONTENT

UNIT - I **(10L)**

Boolean Algebra and Logic Gates: Binary codes, Binary Storage and Registers, Binary logic. **Digital logic gates. Data Representation**: Data types, Complements, Fixed Point Representation, Floating Point Representation.

Digital Computers: Introduction, Block diagram of Digital Computer, Definition of Computer Organization, Computer Design and Computer Architecture.

UNIT - II **(10L)**

Combinational Logic: Combinational Circuits, Analysis procedure Design procedure, Binary Adder-Subtractor Decimal Adder, Binary multiplier, magnitude comparator, Decoders, Encoders, Multiplexers, HDL for combinational circuits.

Sequential Logic: Sequential circuits, latches, Flip-Flops Analysis of clocked sequential circuits, state Reduction and Assignment, Design Procedure**.** Registers, shift Registers, Ripple counters, synchronous counters, other counters.

UNIT III **(10L)**

Register Transfer Language and Micro operations: Register Transfer language, Register Transfer, Bus and memory transfers, Arithmetic Micro operations, logic micro operations, shift micro operations, Arithmetic logic shift unit.

Basic Computer Organization and Design: Instruction codes, Computer Registers Computer instructions, Timing and Control, Instruction cycle, Memory Reference Instructions, Input – Output and Interrupt.

UNIT - IV **(9L)**

Microprogrammed Control: Control memory, Address sequencing, micro program example, design of control unit.

Central Processing Unit: General Register Organization, Instruction Formats, Addressing modes, Data Transfer and Manipulation, Program Control.

Computer Arithmetic: Addition and subtraction, multiplication Algorithms, Division Algorithms, Floating – point Arithmetic operations. Decimal Arithmetic unit, Decimal Arithmetic operations.

UNIT - V **(9L)**

Input-Output Organization: Input-Output Interface, Asynchronous data transfer, Modes of Transfer, Priority Interrupt Direct memory Access.

Memory Organization: Memory Hierarchy, Main Memory, Auxiliary memory, Associate Memory, Cache Memory.

5. TEXT BOOKS

1) Digital Design – M. Morris Mano, Third Edition, Pearson/PHI.
2) Computer System Architecture – M. Morris Mano, Third Edition, Pearson/PHI.

6. REFERENCE BOOKS

1) Switching and Finite Automata Theory, ZVI. Kohavi, Tata Mc Graw Hill.
2) Computer Organization – Carl Hamacher, Zvonks Vranesic, SafeaZaky, 5th Edition, McGraw Hill.
3) Computer Organization and Architecture – William Stallings Sixth Edition, Pearson/PHI.
4) Structured Computer Organization – Andrew S. Tanenbaum, 4th Edition, PHI/Pearson.

CO-PO-PSO Mapping

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2
CO 1	**3**	**2**	**1**	**1**	**1**	**–**	**–**	**–**	**–**	**–**	**–**	**2**	**3**	**–**
CO 2	**3**	**3**	**3**	**2**	**2**	**–**	**–**	**–**	**–**	**–**	**–**	**2**	**3**	**–**
CO 3	**3**	**3**	**2**	**2**	**2**	**–**	**–**	**–**	**–**	**–**	**–**	**2**	**3**	**–**
CO 4	**3**	**3**	**3**	**2**	**2**	**1**	**–**	**–**	**–**	**–**	**–**	**2**	**3**	**1**
CO 5	**3**	**2**	**2**	**2**	**2**	**1**	**–**	**–**	**–**	**–**	**–**	**2**	**3**	**1**

COURSE CONTENT

OBJECT ORIENTED PROGRAMMING THROUGH JAVA								
II Year - I Semester: CSE								
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
CS303PC	**Core**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	**-**	**-**	**3**	**40**	**60**	**100**
Contact Classes: 48	Tutorial Classes: Nil	Practical Classes: Nil				**Total Classes: 48**		
Prerequisite: Nil								

1. COURSE OVERVIEW

This course introduces the fundamentals of **Object-Oriented Programming (OOP)** using Java. It emphasizes key OOP principles such as abstraction, encapsulation, inheritance, and polymorphism while providing practical exposure to Java constructs. Students will learn exception handling, multithreading, event-driven programming, and GUI development using AWT and Swing. The course also covers database connectivity (JDBC), enabling students to develop robust and scalable Java applications.

2. COURSE OBJECTIVE

The students will try to Learn:

1) To understand the principles of OOP and apply them in solving real-world problems.
2) To illustrate and apply inheritance for program reusability.
3) To implement multitasking using multithreading and event handling.
4) To develop database-driven applications using JDBC.
5) To design console-based and GUI-based applications using Java.

3. COURSE OUTCOMES

After successful completion of the course, students should be able to:

CO 1	Apply object-oriented principles and Java programming fundamentals to develop simple Java applications using classes, objects, constructors, methods, control statements, and core language features.
CO 2	Analyze and implement inheritance, packages, interfaces, and polymorphism concepts to design reusable, modular, and extensible Java programs.
CO 3	Demonstrate the use of exception handling and multithreading mechanisms to develop robust, concurrent, and efficient Java applications.
CO 4	Utilize Java standard libraries, event handling mechanisms, graphics, and layout managers to build interactive and well-structured graphical user interface applications.
CO 5	Design and implement user-friendly GUI applications using Swing components, menus, and MVC architecture to create platform-independent desktop applications.

4. COURSE CONTENT

UNIT – I **(10L)**

Object oriented thinking and Java Basics- Need for oop paradigm, summary of oop concepts, coping with complexity, abstraction mechanisms. History of Java, Java buzzwords, data types, variables, scope and lifetime of variables, arrays, operators, expressions, control statements, type conversion and casting, simple java program, concepts of classes, objects, constructors, methods, access control, this keyword, garbage collection, overloading methods and constructors, parameter passing, recursion, nested and inner classes, exploring String class.

UNIT – II **(10L)**

Inheritance, Packages and Interfaces – Hierarchical abstractions, Base class object, subclass, subtype, substitutability, forms of inheritance specialization, specification, construction, extension, limitation, combination, benefits of inheritance, costs of inheritance. Member access rules, super keyword uses, using final keyword with inheritance, polymorphism- method overriding, abstract classes, the Object class. Defining, Creating and Accessing a Package, Understanding CLASSPATH, importing packages, differences between classes and interfaces, defining an interface, implementing interface, applying interfaces, variables in interface and extending interfaces.

UNIT – III **(10L)**

Exception handling and Multithreading-- Concepts of exception handling, benefits of exception handling, Termination or resumptive models, exception hierarchy, usage of try, catch, throw, throws and finally, built in exceptions, creating own exception subclasses. Differences between multithreading and multitasking, thread life cycle, creating threads, thread priorities, synchronizing threads, inter thread communication, thread groups, daemon threads.

UNIT – IV **(9L)**

Exploring String class, Object class, Exploring java.util package, Exploring java.io package Event Handling: Events, Event sources, Event classes, Event Listeners, Delegation event model, handling mouse and keyboard events, Adapter classes. graphics, layout manager – layout manager types – border, grid, flow, card and grid bag.

UNIT – V **(9L)**

Swing – Introduction, limitations of AWT, MVC architecture, components, containers, exploring swingJFrame and JComponent, JLabel, ImageIcon, JTextField, JButton, JCheckBox, JRadioButton, JList, JComboBox, Tabbed Panes, Scroll Panes, Trees, and Tables. Menu Basics, Menu related classes - JMenuBar, JMenu, JMenuItem, JCheckBoxMenuItem, JRadioButtonMenuItem, JSeperator. creating a popup menu.

5. TEXT BOOKS

1) Java the complete reference, 13th edition, Herbert schildt, Dr. Denny Coward, Mc Graw Hill. 2. Understanding OOP with Java, updated edition, T. Budd, Pearson education.

6. REFERENCE BOOKS

1) An Introduction to programming and OO design using Java, J.Nino and F.A. Hosch, John Wiley& sons.
2) An Introduction to OOP, third edition, T. Budd, Pearson education.
3) Introduction to Java programming, Y. Daniel Liang, Pearson education.
4) An introduction to Java programming and object-oriented application development, R.A.Johnson-Thomson.
5) Core Java 2, Vol 1, Fundamentals, Cay.S. Horstmann and Gary Cornell, eighth Edition,Pearson Education.
6) Core Java 2, Vol 2, Advanced Features, Cay.S. Horstmann and Gary Cornell, eighth Edition,Pearson Education.
7) Object Oriented Programming with Java, R.Buyya, S.T.Selvi, X.Chu, TMH.
8) Java and Object Orientation, an introduction, John Hunt, second edition, Springer.
9) Maurach's Beginning Java2 JDK 5, SPD.

CO-PO-PSO Mapping

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2
CO 1	**3**	**2**	**2**		**2**							**1**	**3**	**1**
CO 2	**3**	**3**	**3**		**2**							**1**	**2**	**1**
CO 3	**3**	**3**	**2**	**2**	**2**							**2**	**3**	**1**
CO 4	**2**	**2**	**3**	**2**	**3**							**2**	**2**	**1**
CO 5	**2**	**2**	**3**		**3**							**2**	**2**	**1**

COURSE CONTENT

SOFTWARE ENGINEERING								
II Year - I Semester: CSE								
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
CS304PC	**Core**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	**-**	**-**	**3**	**40**	**60**	**100**
Contact Classes: 48	Tutorial Classes: Nil	Practical Classes: Nil				**Total Classes: 48**		
Prerequisite: Nil								

1. COURSE OVERVIEW

This course introduces fundamental concepts of software engineering, including the evolving role of software, software processes, and development models. It covers requirements engineering techniques such as requirement analysis, validation, and documentation. The course explains software design principles, architectural design, and UML-based modelling techniques. Students learn various software testing strategies, debugging methods, and validation techniques. Software metrics and measurement methods are discussed to assess process and product quality. The course also addresses risk management and software quality assurance practices, including ISO standards.

2. COURSE OBJECTIVE

The students will try to Learn:

1) The aim of the course is to provide an understanding of the working knowledge of the techniques for estimation, design, testing and quality management of large software development projects.
2) Topics include process models, software requirements, software design, software testing, software process/product metrics, risk management, quality management and UML diagrams.

3. COURSE OUTCOMES

After successful completion of the course, students should be able to:

CO 1	Explain the evolving role of software, software process frameworks, CMMI, and compare different software process.
CO 2	Translate end-user requirements into structured system and software requirements using UML, and prepare a Software Requirement Document (SRD).
CO 3	Apply software design principles and concepts to develop architectural and detailed design models using UML diagrams.
CO 4	Apply appropriate software testing strategies, debugging techniques, and software metrics to evaluate and improve the quality of software processes and products.
CO 5	Analyze and apply risk management strategies and software quality assurance practices for reliable software development.

4. COURSE CONTENT

UNIT – I **(10L)**

Introduction to Software Engineering: The evolving role of software, changing nature of software, software myths.

A Generic View of Process: Software engineering as layered technology, process framework, capability maturity model integration (CMMI).

Process Models: The waterfall model, Spiral model, Incremental Process Models, Concurrent Models, Component based development and Agile Development.

UNIT – II **(10L)**

Software Requirements: Functional and non-functional requirements, user requirements, system requirements, interface specification, software requirements document.

Requirements Engineering Process: Feasibility studies, requirements elicitation and analysis, requirements validation, requirements management.

UNIT – III **(10L)**

Design Engineering: Design process and quality, design concepts, design model.

Architectural Design: Software architecture, data design, architectural styles and patterns.

UML Modeling: Conceptual model of UML, structural modeling, class diagrams, sequence diagrams, collaboration diagrams, use case diagrams, component diagrams.

UNIT – IV **(9L)**

Testing Strategies: A strategic approach to software testing, conventional software test strategies, black-box and white-box testing, validation testing, system testing, debugging techniques.
Metrics for Process and Products: Software measurement, metrics for software quality.

UNIT – V **(9L)**

Risk Management: Reactive vs. proactive risk strategies, software risks, risk identification, risk projection, risk refinement, RMMM.

Quality Management: Quality concepts, software quality assurance (SQA), reviews, formal technical reviews, statistical SQA, software reliability, ISO 9000 standards.

5. TEXT BOOKS

1) Software Engineering: A Practitioner's Approach – Roger S. Pressman, 6th Edition, McGraw Hill.
2) Software Engineering – Ian Sommerville, 7th Edition, Pearson Education.
3) The unified modeling language user guide, Grady Booch, James Rumbaugh, Ivar Jacobson, Pearson Education.

6. REFERENCE BOOKS

1) Software Engineering, an Engineering approach- James F. Peters, Witold Pedrycz, John Wiley.
2) Software Engineering principles and practice- Waman S Jawadekar, The McGraw-Hill Companies.
3) Fundamentals of object-oriented design using UML Meiler page-Jones: Pearson Education.
4) Fundamentals of Software Engineering-Rajib Mall, PHI.

CO-PO-PSO Mapping

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2
CO 1	**3**	**2**	**1**	**1**	**2**				**2**	**2**	**3**	**2**	**2**	**1**
CO 2	**3**	**2**	**2**	**1**	**2**				**2**	**2**	**3**	**2**	**2**	**1**
CO 3	**2**	**3**	**3**	**2**	**2**				**2**	**2**	**3**	**3**	**2**	**1**
CO 4	**3**	**3**	**3**	**2**	**2**				**2**	**2**	**3**	**3**	**2**	**1**
CO 5	**2**	**2**	**3**	**3**	**3**				**2**	**3**	**3**	**3**	**2**	**1**

COURSE CONTENT

DATABASE MANAGEMENT SYSTEMS								
II Year - I Semester: CSE								
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
CS305PC	**Core**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	**-**	**-**	**3**	**40**	**60**	**100**
Contact Classes: 48	Tutorial Classes: Nil	Practical Classes: Nil				**Total Classes: 48**		
Prerequisite: Data Structures								

1. COURSE OVERVIEW

The "Database Management Systems" course introduces students to the principles, design, and implementation of database systems. It covers data models, ER modeling, relational algebra, SQL, normalization, transaction management, concurrency control, recovery, and indexing structures. Students will develop both theoretical foundations and practical skills to design, query, and manage databases. By the end of the course, students will be equipped to apply DBMS concepts in real-world applications requiring efficient and reliable data management.

2. COURSE OBJECTIVE

The students will try to Learn:

1) To understand the basic concepts, architecture, and applications of database systems.
2) To master SQL for data definition, manipulation, and control.
3) To apply relational algebra and design normalized database schemas.
4) To understand transaction management, concurrency control, and recovery.
5) To explore indexing and hashing structures for optimized query performance.

3. COURSE OUTCOMES

After successful completion of the course, students should be able to:

CO 1	Understand database concepts, ER modeling, and convert ER diagrams into relational schemas.
CO 2	Apply relational algebra and SQL to define, manipulate, and query data.
CO 3	Normalize relational schemas using appropriate normal forms to reduce redundancy.
CO 4	Demonstrate understanding of transaction concepts, concurrency control, and recovery techniques.
CO 5	Implement indexing and hashing techniques to improve query processing and data retrieval.

4. COURSE CONTENT

UNIT - I: (10L)

Database System Applications: A Historical Perspective, File Systems versus a DBMS, the Data Model, Levels of Abstraction in a DBMS, Data Independence, Structure of a DBMS Introduction to Database Design: Database Design and ER Diagrams, Entities, Attributes, and Entity Sets, Relationships and Relationship Sets, Additional Features of the ER Model, Conceptual Design With the ER Model

UNIT - II: (8L)

Introduction to the Relational Model: Integrity constraint over relations, enforcing integrity constraints, querying relational data, logical database design, introduction to views, destroying/altering tables and views. Relational Algebra, Tuple relational Calculus, Domain relational calculus.

UNIT – III **(10L)**

SQL: QUERIES, CONSTRAINTS, TRIGGERS: form of basic SQL query, UNION, INTERSECT, and EXCEPT, Nested Queries, aggregation operators, NULL values, complex integrity constraints in SQL, triggers and active databases. Schema Refinement: Problems caused by redundancy, decompositions, problems related to decomposition, reasoning about functional dependencies, FIRST, SECOND, THIRD normal forms, BCNF, lossless join decomposition, multivalued dependencies, FOURTH normal form, FIFTH normal form.

UNIT - IV: **(10L)**

Transaction Concept, Transaction State, Implementation of Atomicity and Durability, Concurrent Executions, Serializability, Recoverability, Implementation of Isolation, Testing for serializability, Lock Based Protocols, Timestamp Based Protocols, Validation- Based Protocols, Multiple Granularity, Recovery and Atomicity, Log–Based Recovery, Recovery with Concurrent Transactions.

UNIT - V: **(10L)**

Data on External Storage, File Organization and Indexing, Cluster Indexes, Primary and Secondary Indexes, Index data Structures, Hash Based Indexing, Tree based Indexing, Comparison of File Organizations, Indexes- Intuitions for tree Indexes, Indexed Sequential Access Methods (ISAM), B+ Trees: A Dynamic Index Structure.

5. TEXT BOOKS

1) Database System Concepts, Silberschatz, Korth, McGraw hill, V edition.3rd Edition.
2) Database Management Systems, Raghurama Krishnan, Johannes Gehrke, Tata Mc Graw Hill.

6. REFERENCE BOOKS

1) Database Systems design, Implementation, and Management, Peter Rob & Carlos Coronel 7th Edition.
2) Fundamentals of Database Systems, Elmasri Navrate, Pearson Education
3) Introduction to Database Systems, C. J. Date, Pearson Education
4) Oracle for Professionals, The X Team, S.Shah and V. Shah, SPD.
5) Database Systems Using Oracle: A Simplified guide to SQL and PL/SQL, Shah, PHI.
6) Fundamentals of Database Management Systems, M. L. Gillenson, Wiley Student Edition.

CO-PO-PSO Mapping

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2
CO 1	**3**	**3**	**2**										**3**	**1**
CO 2	**3**	**3**	**3**	**2**									**2**	**1**
CO 3	**3**	**3**	**3**	**2**									**3**	**1**
CO 4	**3**	**3**	**3**	**2**	**2**								**2**	**1**
CO 5	**3**	**3**	**3**	**3**	**3**							**2**	**2**	**1**

COURSE CONTENT

QUANTITATIVE APTITUDE AND LOGICAL REASONING								
II Year - I Semester: CSE								
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
MS306HS	**Foundation**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		2	**-**	**-**	**2**	**40**	**60**	**100**
Contact Classes: 32	Tutorial Classes: Nil	Practical Classes: Nil				**Total Classes: 32**		
Prerequisite: Nil								

1. COURSE OVERVIEW

The **Quantitative Aptitude and Logical Reasoning** course is designed to strengthen numerical ability, analytical thinking, and problem-solving skills required for competitive examinations, campus placements, and professional aptitude tests. The course covers fundamental arithmetic, algebra, data interpretation, and logical reasoning techniques, enabling learners to analyze problems systematically, apply appropriate methods, and arrive at accurate solutions efficiently. Emphasis is placed on speed, accuracy, and logical thinking to enhance overall aptitude and decision-making skills.

2. COURSE OBJECTIVE

The students will try to Learn:

1) To answer general problems in his everyday life within in short time and to improves the certain skills of a student such as numerical and logical ability, mental capacity and also in sharpening minds.

3. COURSE OUTCOMES

After successful completion of the course, students should be able to:

CO 1	Apply concepts of number systems, HCF and LCM, averages, ages, and ratio and proportion to solve quantitative problems.
CO 2	Solve problems related to various important topics of quantitative aptitude using appropriate mathematical techniques.
CO 3	Analyze and solve problems involving mensuration and data interpretation.
CO 4	Apply logical reasoning techniques to solve analytical and reasoning-based problems.
CO 5	Solve problems based on Venn diagrams, cubes and dice, and clock and calendar concepts.

4. COURSE CONTENT

UNIT – I **(6L)**

Number System: Test for Divisibility, Test of prime number, Division and Remainders – HCF and LCM of Numbers–Fractions and Decimals -Average-Problems on Ages- Problems on Numbers- Ratio and Proportion.

UNIT -- II **(6L)**

Percentage – Profit, Loss and Discount – Partnership and Share-Simple Interest – Compound Interest. Time and Work- Pipes and Cisterns-Time and Distance- Problems on Trains- Boats and Streams.

UNIT -- III **(6L)**

Allegation or Mixtures, Clocks &Calendar, Mensuration : Area of Plane Figures, Volume and Surface Area of Solid Figures.

Data Interpretation: Tabulation, Bar Graphs, Pie Charts, Line Graphs.

UNIT -- IV **(7L)**

Series Completion: Number Series, Alphabet Series, Alpha – Numeric Series.

Classification: Word Classification, Number Classification and Letter Classification.

Mathematical Operations-Arithmetical Reasoning. Puzzle Test: Classification Type Questions, Seating Arrangements, Comparison Type Questions, Sequential Order of Things, Selection Based on Given Conditions, Jumbled Problems.

UNIT -- V **(7L)**

Logical Venn Diagrams –Cubes and Dice – Analytical Reasoning-Assertions and Reason– Logical Deductions-Syllogism -Statement and Arguments-Statement and Conclusions- -Data Sufficiency.

5. TEXT BOOKS:

1. R. S. Agarwal, *Quantitative Aptitude*, Revised Edition, S. Chand Publishing, New Delhi.
2. R. S. Agarwal, *Verbal and Non-Verbal Reasoning*, Revised Edition, S. Chand Publishing, New Delhi.

CO-PO-PSO Mapping

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2
CO 1	**3**	**2**										**2**	**3**	**1**
CO 2	**3**	**3**	**1**									**2**	**2**	**1**
CO 3	**2**	**3**	**2**	**1**								**1**	**3**	**1**
CO 4	**2**	**3**	**1**						**1**			**2**	**2**	**1**
CO 5	**2**	**2**	**1**									**1**	**2**	**1**

COURSE CONTENT

OBJECT ORIENTED PROGRAMMING THROUGH JAVA LAB								
II Year - I Semester: CSE								
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
CS307PC	**Core**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		-	**-**	**2**	**1**	**40**	**60**	**100**
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 32				**Total Classes: 32**		
Prerequisite: Basic knowledge of programming fundamentals and object-oriented concepts.								

1) COURSE OVERVIEW

This laboratory course provides **hands-on programming experience in Java**, focusing on **object-oriented programming concepts** such as abstraction, inheritance, multithreading, exception handling, event-driven programming, and GUI development using Swing. Students will learn to work with Java tools such as Eclipse IDE and practice problem-solving through collection framework, applets, and file handling.

2) COURSE OBJECTIVE

The students will try to Learn:

1) To write programs using abstract classes.
2) To write programs for solving real world problems using the java collection framework.
3) To write multithreaded programs.
4) To write GUI programs using swing controls in Java.
5) To introduce java compiler and eclipse platform.
6) To impart hands-on experience with java programming.

3) COURSE OUTCOMES

After successful completion of the course, students should be able to:

CO 1	Apply Java development tools and IDE features (Eclipse/NetBeans) such as project creation, code formatting, refactoring, debugging, and control structures to develop and test basic Java programs.
CO 2	Design and implement GUI-based Java applications using AWT/Swing components, layouts, event handling, and exception handling to solve real-world problems like calculators, traffic lights, and division operations.
CO 3	Develop Java programs using object-oriented principles such as abstraction, inheritance, polymorphism, and data structures to build modular and reusable applications.
CO 4	Implement multithreaded Java applications using thread creation, synchronization, inter-thread communication, and producer–consumer concepts to handle concurrent tasks efficiently.
CO 5	Utilize Java I/O, collections, file handling, and event-handling mechanisms to process data from files, handle mouse events, manage directories, and implement hash-based data retrieval systems.

NOTE

1) Use LINUX and MySQL for the Lab Experiments. Though not mandatory, encourage the use of the Eclipse platform.

2) The list suggests the minimum program set. Hence, the concerned staff is requested to add more problems to the list as needed.

4) COURSE CONTENT

LIST OF EXPERIMENTS

1) Use Eclipse or Net bean platform and acquaint yourself with the various menus. Create a test project, add a test class, and run it. See how you can use auto suggestions, auto fill. Try code formatter and code refactoring like renaming variables, methods, and classes. Try debug step by step with a small program of about 10 to 15 lines which contains at least one if else conditionand a for loop.
2) Write a Java program that works as a simple calculator. Use a grid layout to arrange buttons for the digits and for the +, -,*, % operations. Add a text field to display the result. Handle any possible exceptions like divided by zero.
3) A) Develop an applet in Java that displays a simple message.

   B) Develop an applet in Java that receives an integer in one text field, and computes its factorial
4) Value and returns it in another text field, when the button named "Compute" is clicked.
5) Write a Java program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num 2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a Number Format Exception. If Num2 were Zero, the program would throw an Arithmetic Exception. Display the exception in a message dialog box.
6) Write a Java program that implements a multi-thread application that has three threads. First thread generates a random integer every 1 second and if the value is even, the second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of the cube of the number.
7) Write a Java program for the following:

   Create a doubly linked list of elements.

   Delete a given element from the above list.

   Display the contents of the list after deletion.
8) Write a Java program that simulates a traffic light. The program lets the user select one of three lights: red, yellow, or green with radio buttons. On selecting a button, an appropriate message with "Stop" or "Ready" or "Go" should appear above the buttons in the selected color. Initially, there is no message shown.
9) Write a Java program to create an abstract class named Shape that contains two integers and an empty method named print Area (). Provide three classes named Rectangle, Triangle, and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method print Area () that prints the area of the given shape.
10) Suppose that a table named Table.txt is stored in a text file. The first line in the file is the header, and the remaining lines correspond to rows in the table. The elements are separated by commas.
11) Write a java program to display the table using Labels in Grid Layout.
12) Write a Java program that handles all mouse events and shows the event name at the center of the window when a mouse event is fired (Use Adapter classes).
13) Write a Java program that loads names and phone numbers from a text file where the data is organized as one line per record and each field in a record are separated by a tab (\t). It takes a name or phone number as input and prints the corresponding other value from the hash table (hint: use hash tables).
14) Write a Java program that correctly implements the producer – consumer problem using the concept of inter thread communication.
15) Write a Java program to list all the files in a directory including the files present in all its subdirectories.

5) TEXT BOOKS

1) Java for Programmers, P. J. Deitel and H. M. Deitel, 10th Edition Pearson education.
2) Thinking in Java, Bruce Eckel, Pearson Education.

6) REFERENCE BOOKS

1) Java Programming, D. S. Malik and P. S. Nair, Cengage Learning.
2) Core Java, Volume 1, 9th edition, Cay S. Horstmann and G Cornell, Pearson.

CO-PO-PSO Mapping

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2
CO 1	**3**	**2**	**2**		**3**							**1**	**3**	**1**
CO 2	**3**	**3**	**3**	**2**	**3**							**2**	**2**	**1**
CO 3	**3**	**3**	**3**		**2**							**2**	**3**	**1**
CO 4	**3**	**3**	**2**	**2**	**2**							**2**	**2**	**1**
CO 5	**3**	**2**	**2**	**2**	**3**							**2**	**2**	**1**

COURSE CONTENT

SOFTWARE ENGINEERING LAB								
II Year - I Semester: CSE								
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
CS308PC	**Core**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		-	**-**	**2**	**1**	**40**	**60**	**100**
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 32				**Total Classes: 32**		
Prerequisites: A course on "Programming for Problem Solving". **Co-requisite**: A Course on "Software Engineering".								

1. COURSE OVERVIEW

A Software Engineering Lab course offers hands-on experience in applying software engineering principles through practical projects, covering requirement analysis, design, implementation, testing, and project management.

2. COURSE OBJECTIVE

The students will try to Learn:

1) To have hands on experience in developing a software project by using various software engineering principles and methods in each of the phases of software development.

3. COURSE OUTCOMES

After successful completion of the course, students should be able to:

CO 1	Ability to translate end-user requirements into system and software requirements.
CO 2	Ability to generate a high-level design of the system from the software requirements
CO 3	Will have experience and/or awareness of testing problems and will be able to develop a simple testing report
CO 4	Develop architectural and detailed software designs using standard modeling languages.
CO 5	Manage projects by applying software engineering principles, including configuration and risk management.

4. COURSE CONTENT

Do the following seven exercises for any two projects given in the list of sample projects or any other Projects:

1) Development of problem statements.
2) Preparation of Software Requirement Specification Document, Design Documents and Testing Phase related documents.
3) Preparation of Software Configuration Management and Risk Management related documents.
4) Study and usage of any Design phase CASE tool
5) Performing the Design by using any Design phase CASE tools.
6) Develop test cases for unit testing and integration testing
7) Develop test cases for various white box and black box testing techniques.

Sample Projects:

1) Passport automation System
2) Book Bank
3) Online Exam Registration

4) Stock Maintenance System
5) Online course reservation system
6) E-ticketing
7) Software Personnel Management System
8) Credit Card Processing
9) E-book management System.
10) Recruitment system

5. TEXT BOOKS

1) Software Engineering, A practitioner's Approach- Roger S. Pressman, 6th edition, McGraw Hill International Edition.
2) Software Engineering- Sommerville, 7th edition, Pearson Education.
3) The unified modeling language user guide Grady Booch, James Rambaugh, Ivar Jacobson, Pearson Education.

6. REFERENCE BOOKS

1) Software Engineering, an Engineering approach- James F. Peters, Witold Pedrycz, John Wiley.
2) Software Engineering principles and practice- Waman S Jawadekar, The McGraw-Hill.

CO-PO-PSO Mapping

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2
CO 1	**3**	**3**	**2**	**2**	**2**				**1**	**1**	**3**	**1**	**2**	**1**
CO 2	**3**	**3**	**3**	**2**	**2**				**1**	**1**	**3**	**1**	**2**	**1**
CO 3	**3**	**3**	**3**	**2**	**2**				**1**	**1**	**3**	**1**	**2**	**1**
CO 4	**3**	**3**	**3**	**2**	**2**				**1**	**1**	**3**	**1**	**2**	**1**
CO 5	**3**	**3**	**3**	**3**	**3**				**1**	**1**	**3**	**1**	**2**	**1**

COURSE CONTENT

DATABASE MANAGEMENT SYSTEMS LAB								
II Year - I Semester: CSE								
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
CS309PC	**Core**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		-	**-**	**2**	**1**	**40**	**60**	**100**
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 32				**Total Classes: 32**		
Prerequisite: Nil								

1. COURSE OVERVIEW

The "Database Management Systems Lab" course provides hands-on experience in the design and implementation of databases. Students will practice ER modeling, relational schema design, normalization, and SQL programming to manage and query data. The course also focuses on database programming using procedures, triggers, and cursors, enabling students to develop complete database applications.

2. COURSE OBJECTIVE

The students will try to Learn:

1) To design conceptual models using ER diagrams and convert them into relational schemas.
2) To implement normalization techniques to reduce redundancy and improve database design.
3) To gain practical skills in SQL for data definition, manipulation, and querying.
4) To apply advanced SQL features like triggers, procedures, and cursors for application development.
5) To strengthen database application development skills through practical problem-solving.

3. COURSE OUTCOMES

After successful completion of the course, students should be able to:

CO 1	Design database schemas using ER modeling and normalization.
CO 2	Apply SQL commands for data definition, manipulation, and constraints.
CO 3	Use advanced SQL queries including joins, nested queries, and aggregate functions.
CO 4	Implement database triggers, procedures, and cursors for application development.
CO 5	Develop complete database applications integrating SQL features to solve real-world problems.

4. COURSE CONTENT

Practice Sessions / Experiments

1. Concept design with E-R Model.
2. Relational Model implementation.
3. Normalization up to 3NF / BCNF.
4. Practicing DDL commands.
5. Practicing DML commands.
6. **Queries:**
   (i) A. Using ANY, ALL, UNION, INTERSECT, JOIN, Constraints etc.
   (ii) B. Nested & Correlated subqueries.
7. Queries using Aggregate functions, GROUP BY, HAVING; Creating and Dropping Views.
8. Triggers: Insert, Delete, Update Triggers.
9. Procedures: Writing and executing stored procedures.
10. Cursors: Usage of cursors in database applications.

5. TEXT BOOKS

1) Database Management Systems, Raghurama Krishnan, Johannes Gehrke, Tata Mc Graw Hill,3rd Edition.
2) Database System Concepts, Silberschatz, Korth, McGraw Hill, V edition.

6. REFERENCE BOOKS

1) Database Systems design, Implementation, and Management, Peter Rob & Carlos Coronel 7th Edition.
2) Fundamentals of Database Systems, Elmasri Navrate, Pearson Education.
3) Introduction to Database Systems, C.J. Date, Pearson Education.
4) Oracle for Professionals, The X Team, S. Shah and V. Shah, SPD.
5) Database Systems Using Oracle: A Simplified guide to SQL and PL/SQL, Shah, PHI.
6) Fundamentals of Database Management Systems, M. L. Gillenson, Wiley Student Edition.

CO-PO-PSO Mapping

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2
CO 1	**2**	**3**		**2**									**3**	**1**
CO 2	**2**	**3**	**2**										**2**	**1**
CO 3	**2**	**3**	**3**	**2**									**3**	**1**
CO 4	**2**	**3**	**3**	**2**	**2**								**2**	**1**
CO 5	**2**	**3**	**3**	**3**	**3**							**2**	**2**	**1**

COURSE CONTENT

NODE JS / REACT JS / DJANGO LAB								
II Year - I Semester: CSE								
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
CS310SD	**Skill**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		-	**-**	**2**	**1**	**40**	**60**	**100**
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 32				**Total Classes: 32**		
Prerequisite: Nil								

1. COURSE OVERVIEW

This course introduces students to **full stack web development** with emphasis on client-side design, server-side programming, and modern frameworks. Students begin by designing static and responsive websites, move on to database connectivity using Java, and then implement server-side applications with **Node.js**. The course concludes with building **single-page applications (SPAs)** using **React**.

2. COURSE OBJECTIVE

The students will try to Learn:

1) To implement static and responsive web pages using **HTML, CSS, Bootstrap, and JavaScript**.
2) To perform **client-side validation** with JavaScript and use advanced ES6 features.
3) To design and interact with **databases** using JDBC.
4) To implement **server-side applications** using Java and Node.js.
5) To design and develop **single-page applications** with React.

3. COURSE OUTCOMES

After successful completion of the course, students should be able to:

CO 1	Build responsive web applications using HTML, CSS, Bootstrap, and JavaScript.
CO 2	Demonstrate advanced JavaScript features (ES6, promises, async/await) with real-time APIs.
CO 3	Implement database-driven applications using JDBC and servlets.
CO 4	Develop server-side implementations using Node.js and Express framework.
CO 5	Design and deploy **Single Page Applications (SPA)** using React.

4. COURSE CONTENT

LIST OF EXPERIMENTS

1. Build a **responsive shopping cart web application** with registration, login, catalog, and cart pages using CSS3 (flex & grid).
2. Enhance the above web application using the **Bootstrap framework**.
3. Perform **client-side validation** using JavaScript for the pages developed in Experiments 1 and 2.
4. Explore **ES6 features** (arrow functions, callbacks, promises, async/await). Implement an application that fetches weather data from *openweathermap.org* and displays it graphically.
5. Develop a **Java standalone application** that connects to a database (Oracle/MySQL) and performs CRUD operations.
6. Create an **XML document** for a bookstore and validate it using both DTD and XSD.
7. Design a **Servlet-based controller** that integrates the shopping cart application with the database created in Experiment 5.

8. Explore and implement **session tracking mechanisms** (Cookies, HTTP Session) for maintaining user transactional history.
9. Create a **custom server in Node.js** using the HTTP module and explore additional Node.js modules (OS, path, event).
10. Develop an **Express.js web application** that interacts with a REST API to perform CRUD operations on student data. (Test using Postman).
11. Secure the above application by creating **authorized endpoints using JWT (JSON Web Token)**.
12. Create a **React application** for a student management system with registration, login, contact, and about pages. Implement **routing** for navigation.
13. Build a **React service** that fetches weather information from *openweathermap.org* and displays current and historical data graphically using **Chart.js**.
14. Create a **React TODO application** with reusable components and deploy it on **GitHub Pages**.

5. REFERENCE BOOKS

1) Jon Duckett, *Beginning HTML, XHTML, CSS, and JavaScript*, Wrox Publications, 2010.
2) Bryan Basham, Kathy Sierra, Bert Bates, *Head First Servlets and JSP*, O'Reilly Media, 2nd Edition, 2008.
3) Vasan Subramanian, *Pro MERN Stack: Full Stack Web App Development with Mongo, Express, React, and Node*, 2nd Edition, A press.

CO-PO-PSO Mapping

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2
CO 1	**3**	**3**	**3**	**2**	**3**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**3**	**1**
CO 2	**3**	**3**	**3**	**2**	**3**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**2**	**1**
CO 3	**3**	**3**	**3**	**3**	**3**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**3**	**1**
CO 4	**3**	**3**	**3**	**3**	**3**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**2**	**1**
CO 5	**3**	**3**	**3**	**3**	**3**	**0**	**0**	**0**	**0**	**0**	**0**	**2**	**2**	**1**