Srinivasa Ramanujan Institute of Technology

Ananthapuramu (Autonomous), Andhra Pradesh

Department of Computer Science and Engineering

Scheme and Syllabus

(SRIT R23 Regulations)

for

B. Tech. (Regular) Four-Year Degree Program

(Applied for the Batches admitted from AY: 2023-2024)

&

(B. Tech. (LES) for the Batches admitted from AY: 2024-2025)

Vision of the Institute

To become a premier Educational Institution in India offering the best teaching and learning environment for our students that will enable them to become complete individuals with professional competency, human touch, ethical values, service motto, and a strong sense of responsibility towards environment and society at large.

Mission of the Institute

- Continually enhance the quality of physical infrastructure and human resources to evolve in to a centre of excellence in engineering education.
- Provide comprehensive learning experiences that are conducive for the students to acquire professional competences, ethical values, life-long learning abilities and understanding of the technology, environment and society.
- Strengthen industry institute interactions to enable the students work on realistic problems and acquire the ability to face the ever-changing requirements of the industry.
- Continually enhance the quality of the relationship between students and faculty which is a key to the development of an exciting and rewarding learning environment in the college.

Vision of the Department

To evolve as a leading department by offering best comprehensive teaching and learning practices for students to be self-competent technocrats with professional ethics and social responsibilities.

Mission of the Department

- DM 1: Continuous enhancement of the teaching-learning practices to gain profound knowledge in theoretical & practical aspects of computer science applications.
- DM 2: Administer training on emerging technologies and motivate the students to inculcate self-learning abilities, ethical values and social consciousness to become competent professionals.
- DM 3: Perpetual elevation of Industry-Institute interactions to facilitate the students to work on real-time problems to serve the needs of the society.

Program Educational Objectives (PEOs)

An SRIT graduate in Computer Science and Engineering, after three to four years of graduation will:	
PEO1	Lead a successful professional career in IT / ITES industry / Government organizations with ethical values
PEO2	Become competent and responsible computer science professional with good communication skills and leadership qualities to respond and contribute significantly for the benefit of society at large.
PEO3	Engage in life-long learning, acquiring new and relevant professional competencies / higher academic qualifications.

Program Outcomes (POs)

PO1	**Engineering Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2	**Problem Analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	**Design / Development Of Solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	**Conduct Investigations Of Complex Problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	**Modern Tool Usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.
PO6	**The Engineer And Society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO7	**Environment And Sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO8	**Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO9	**Individual And Team Work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	**Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO11	**Project Management and Finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	**Life-Long Learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes

At the end of the B. Tech program in Computer Science and Engineering (Artificial Intelligence and Machine Learning), the graduate will be able to:	
PSO1	Apply concepts to design and develop multi-disciplinary computing systems and applications.
PSO2	Develop models in machine learning, deep learning, and big data technologies using AI knowledge and modern tools.
PSO3	Provide a solid foundation and enhance their abilities to qualify for higher education, research, and employment in Artificial Intelligence and Machine Learning with ethical values.

B. Tech Course Structure

Semester 0

(Common for all branches of Engineering)

S. No.	Course Name	L–T–P -C
1.	Physical Activities -- Sports, Yoga and Meditation, Plantation	0–0–6-0
2.	Career Counseling	2–0–2-0
3.	Orientation to all branches -- career options, tools, etc.	3–0–0-0
4.	Orientation on admitted Branch --corresponding labs, tools and platforms	2–0–3-0
5.	Proficiency Modules & Productivity Tools	2–1–2-0
6.	Assessment on basic aptitude and mathematical skills	2–0–3-0
7.	Remedial Training in Foundation Courses	2–1–2-0
8.	Human Values & Professional Ethics	3–0–0-0
9.	Communication Skills -- focus on Listening, Speaking, Reading, Writing skills	2–1–2-0
10.	Concepts of Programming	2–0–2-0

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY (AUTONOMOUS)

Department of Computer Science and Engineering

B. Tech in Computer Science and Engineering

II B. Tech – I Semester										
S.No	Course Code	Course Title	Subject Area	Contact Hours Per Week			Credits	Scheme of Examination Max. Marks		
				L	T	P		CIA	SEE	Total
1		Discrete Mathematics & Graph Theory	BSC	3	0	0	3	30	70	100
2		Universal Human Values–Understanding Harmony & Human Ethical Conduct	HSMC	2	1	0	3	30	70	100
3		Digital Logic & Computer Organization	ESC	3	0	0	3	30	70	100
4		Advanced Data Structures & Algorithm Analysis	PCC	3	0	0	3	30	70	100
5		Database Management Systems	PCC	3	0	0	3	30	70	100
6		Advanced Data Structures and Algorithm Analysis Lab	PCC	0	0	3	1.5	30	70	100
7		Database Management Systems Lab	PCC	0	0	3	1.5	30	70	100
8		Python Programming	SEC	0	1	2	2	30	70	100
9		Environmental Science	AC	2	0	0	-	30	-	-
Total							**20**	**270**	**560**	**830**

II B. Tech – II Semester										
S.No	Course Code	Course Title	Subject Area	Contact Hours Per Week			Credits	Scheme of Examination Max. Marks		
				L	T	P		CIA	SEE	Total
1		Management Elective- I Managerial Economics and Financial Analysis / Business Environment / Organizational Behavior	Management Elective- I	2	0	0	2	30	70	100
2		Probability & Statistics	BSC	3	0	0	3	30	70	100
3		Object Oriented Programming Through Java	PCC	3	0	0	3	30	70	100
4		Operating Systems	PCC	3	0	0	3	30	70	100
5		Software Engineering	PCC	3	0	0	3	30	70	100
6		Object Oriented Programming Through Java Lab	PCC	0	0	3	1.5	30	70	100
7		Operating Systems Lab	PCC	0	0	3	1.5	30	70	100
8		Full Stack Development –I	SEC	0	1	2	2	30	70	100
9		Design Thinking & Innovation	BS&H	1	0	2	2	30	70	100
Total							**21**	**270**	**630**	**900**
Mandatory Community Service Project Internship of 08 weeks duration during summer vacation										

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

Advanced Data Structures & Algorithm Analysis

(Common to CSE and CSM))

II B. Tech – I Semester								SRIT R23
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
	PCC	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	0	0	3	30	70	100

Course Objectives: The main objective of the course is to

- Understand the Fundamentals of Algorithm Analysis
- Provide knowledge on advance data structures frequently used in Computer Science
- Develop skills in algorithm design techniques
- Understand the use of various data structures in the algorithm design
- Understand NP Hard and NP Complete Problems

Unit I: Algorithm Analysis, AVL and B-Trees

Introduction to Algorithm Analysis, Space and Time Complexity analysis, Asymptotic Notations.

AVL Trees – Creation, Insertion, Deletion operations and Applications
B-Trees – Creation, Insertion, Deletion operations and Applications

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
  1. https://youtu.be/mRGQyIRWAsI

Unit II: Heap Trees, Graphs, Divide and Conquer

Heap Trees (Priority Queues) – Min and Max Heaps, Operations and Applications.

Graphs – Terminology, Representations, Basic Search and Traversals, Connected Components and Biconnected Components, applications.

Divide and Conquer: The General Method, Quick Sort, Merge Sort, Strassen's matrix multiplication, Convex Hull.

Pedagogy/Course Delivery tools: Chalk & Talk & ICT
Links for online material:

1. NPTEL :: Computer Science and Engineering - Data Structures And Algorithms

Unit III: Greedy Method and Dynamic Programming

Greedy Method: General Method, Job Sequencing with deadlines, Knapsack Problem, Minimum cost spanning trees, Single Source Shortest Paths.

Dynamic Programming: General Method, All pairs shortest paths, Single Source Shortest Paths– General Weights (Bellman Ford Algorithm), Optimal Binary Search Trees, 0/1 Knapsack, String Editing, Travelling Salesperson problem.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
  1. https://www.tutorialspoint.com/advanced_data_structures/index.asp
  2. Lecture -18 Dynamic Programming (youtube.com)

Unit IV: Backtracking and Branch and Bound

Backtracking: General Method, 8-Queens Problem, Sum of Subsets problem, Graph Coloring, 0/1 Knapsack Problem.

Branch and Bound: The General Method, 0/1 Knapsack Problem, Travelling Salesperson problem.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT

- Links for online material:
    1. INTRODUCTION TO BACKTRACKING (youtube.com)

Unit V: NP Hard and NP Complete Problems

NP Hard and NP Complete Problems: Basic Concepts, Cook's theorem.

NP Hard Graph Problems: Clique Decision Problem (CDP), Chromatic Number Decision Problem (CNDP), Traveling Salesperson Decision Problem (TSP).

NP Hard Scheduling Problems: Scheduling Identical Processors, Job Shop Scheduling.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
    1. P and NP (youtube.com)

Text books	
1	Fundamentals of Data Structures in C++, Horowitz, Ellis; Sahni, Sartaj; Mehta, Dinesh, 2nd Edition Universities Press.
2	Computer Algorithms in C++, Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran, and 2nd Edition University Press.

References:	
1	The Art of Computer Programming, Vol.1: Fundamental Algorithms, Donald E Knuth, Addison-Wesley, 1997.
2	Data Structures using C & C++: Langsam, Augenstein & Tanenbaum, Pearson, 1996.
3	Data Structures and program design in C, Robert Kruse, Pearson Education Asia.

Course Outcomes: At the end of the course, the student should have acquired the ability to	
CO1	Use various data structures in the design of algorithms. **Cognitive Level: L2**
CO2	Analyze algorithms with respect to space and time complexities and implement AVL trees, B-Trees in various applications. **Cognitive Level: L3**
CO3	Apply graph components and use the divide and conquer approach to solve problems. **Cognitive Level: L3**
CO4	Solve optimization problems using greedy and dynamic programming methods. **Cognitive Level: L3**
CO5	Use backtracking and branch and bound to solve combinatorial problems. **Cognitive Level: L3**
CO6	Analyze and solve NP-hard graph problems and examine NP-hard scheduling problems. **Cognitive Level: L4**

Course Assessment & Evaluation		
Continuous Internal Evaluation (CIE): 20 Marks		
Assessment Tool	Marks	Course Outcomes addressed
Continuous Internal Examination-I	20	CO1, CO2, CO3, CO4
Continuous Internal Examination-II	20	CO1, CO4, CO5, CO6
Final CIE marks shall be evaluated by considering 80% weightage to the best CIE and 20% weightage to the other CIE.		
Continuous Alternative Assessment (CAA): 10 marks The CAA may include assignments, seminars, term paper, open ended experiments, METE (Modeling and Experimental Tools in Engineering), five minutes video, MOOC's etc.		
Continuous Alternative Assessment-I	10	CO1, CO2, CO3, CO4
Continuous Alternative Assessment-II	10	CO1, CO4, CO5, CO6
Final CAA marks shall be evaluated by averaging CAA-I & CAA-II for 10 marks		
Semester End Examination (SEE): 70 marks		
Semester End Examination (SEE)	70	CO1, CO2, CO3, CO4, CO5, CO6
Total marks	**100**	

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

Database Management Systems

(Common to CSE and CSM)

II B. Tech – I Semester								SRIT R23
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
	PCC	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	0	0	3	30	70	100

Course Objectives: The main objective of the course is to

- Introduce database management systems and to give a good formal foundation on the relational model of data and usage of Relational Algebra.
- Introduce the concepts of basic SQL as a universal Database language.
- Demonstrate the principles behind systematic database design approaches by covering conceptual design, logical design through normalization.
- Understand Concurrent Executions and Serializability.
- Provide an overview of physical design of a database system by discussing Database indexing techniques and storage techniques.

Unit I: Introduction

Database system, Characteristics (Database Vs File System), Database Users, Advantages of Database systems, Database applications. Brief introduction of different Data Models, Concepts of Schema, Instance and data independence, Three tier schema architecture for data independence, Database system structure, environment, Centralized and Client Server architecture for the database.

Entity Relationship Model: Introduction, Representation of entities, attributes, entity set, relationship, relationship set, constraints, sub classes, super class, inheritance, specialization, generalization using ER Diagrams.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
  1. https://youtu.be/sEaYXwmsLw0?si=O_528MfKW2_aI3Rg
  2. https://youtu.be/sEaYXwmsLw0?si=O_528MfKW2_aI3Rg

Unit II: Relational Model

Introduction to relational model, concepts of domain, attribute, tuple, relation, importance of null values, constraints (Domain, Key constraints, integrity constraints) and their importance, Relational Algebra, Relational Calculus.

BASIC SQL: Simple Database schema, data types, table definitions (create, alter), different DML operations (insert, delete, update).

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
  1. https://youtu.be/Z2Zx2G02aI4?si=BoEtJiLc3oso6Hez
  2. https://youtu.be/9oAWVvUfa-E?si=ZGTcpL17RIwb3BeR

Unit III: SQL

SQL: Basic SQL querying (select and project) using where clause, arithmetic & logical operations, SQL functions (Date and Time, Numeric, String conversion).Creating tables with relationship, implementation of key and integrity constraints, nested queries, sub queries, grouping, aggregation, ordering, implementation of different types of joins, view(updatable and non-updatable), relational set operations.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
  1. https://youtu.be/PVE4e7_fU0U?si=Ywce1ioeoH80esz5

Unit IV: Schema Refinement (Normalization)

Purpose of Normalization or schema refinement, concept of functional dependency, normal forms based on functional dependency Lossless join and dependency, preserving decomposition, (1NF, 2NF and 3 NF), concept of surrogate key, Boyce-Codd normal form (BCNF), MVD, Fourth normal form(4NF), Fifth Normal Form (5NF).

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT

- Links for online material:
    1. https://youtu.be/NfHGvgMi6rw?si=Pz6jMnTMekGgP4px

Unit V: Transaction Concept

Transaction State, ACID properties, Concurrent Executions, Serializability, Recoverability, Implementation of Isolation, Testing for Serializability, lock based, time stamp based, optimistic, concurrency protocols, Deadlocks, Failure Classification, Storage, Recovery and Atomicity, Recovery algorithm.

Introduction to Indexing Techniques: B+ Trees, operations on B+Trees, Hash Based Indexing.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
    1. https://youtu.be/p4bA-gM8ulQ?si=ZoDQMfTbEFk7W5Fw

Text books	
1	Database Management System, 6th edition, Ramez Elmasri, Shamkant B. Navathe, Pearson.
2	Database System Concepts,5th edition, Silberschatz, Korth, Sudarsan, TMH (For Chapter 1 and Chapter 5)

References	
1	Database Management Systems, 3rd edition, Raghurama Krishnan, Johannes Gehrke, TMH (For Chapters 2, 3, 4)
2	Introduction to Database Systems, 8thedition, C J Date, Pearson.
3	Database Principles Fundamentals of Design Implementation and Management, 10th edition, Corlos Coronel, Steven Morris, Peter Robb, Cengage Learning, 2022.

Course Outcomes: At the end of the course, the student should have acquired the ability to	
CO1	Learn about databases, their benefits, data models, and basic operations. **Cognitive Level: L2**
CO2	Explore how databases are structured, including roles, applications, and different architectures. **Cognitive Level: L4**
CO3	Create ER diagrams, understand relational models, and perform SQL operations. **Cognitive Level: L3**
CO4	Design tables, write SQL queries, and use SQL functions for data manipulation. **Cognitive Level: L3**
CO5	Normalize databases; understand schema optimization techniques, and the role of surrogate keys. **Cognitive Level: L2**
CO6	Implement transaction management, handle concurrency, and use indexing techniques for data retrieval. **Cognitive Level: L3**

Course Assessment & Evaluation		
Continuous Internal Evaluation (CIE): 20 Marks		
Assessment Tool	Marks	Course Outcomes addressed
Continuous Internal Examination-I	20	CO1, CO2, CO3, CO4
Continuous Internal Examination-II	20	CO1, CO4, CO5, CO6
Final CIE marks shall be evaluated by considering 80% weightage to the best CIE and 20% weightage to the other CIE.		
Continuous Alternative Assessment (CAA): 10 marks The CAA may include assignments, seminars, term paper, open ended experiments, METE (Modeling and Experimental Tools in Engineering), five minutes video, MOOC's etc.		
Continuous Alternative Assessment-I	10	CO1, CO2, CO3, CO4
Continuous Alternative Assessment-II	10	CO1, CO4, CO5, CO6
Final CAA marks shall be evaluated by averaging CAA-I & CAA-II for 10 marks		
Semester End Examination (SEE): 70 marks		
Semester End Examination (SEE)	70	CO1, CO2, CO3, CO4, CO5, CO6
Total marks	**100**	

SRINIVASA RAMANUJAN INSTITUTE OFTECHNOLOGY

Advanced Data Structures and Algorithm Analysis Lab

(Common to CSE and CSM)

II B. Tech – I Semester								SRIT R23
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
		L	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		0	0	3	1.5	30	70	100

Course Objectives: The objective of the course is to

- Acquire practical skills in constructing and managing Data structures.
- Implement and analyze various data structures (AVL trees, B-Trees, Heaps).
- Compare the efficiency of various algorithms.
- Apply the popular algorithm design methods in problem-solving scenarios.
- Design, implementing, and analyzing these concepts to solve complex computational problems.

List of Experiments

S. No.	Title of the Experiment
1	Construct an AVL tree for a given set of elements which are stored in a file. And implement insert and delete operation on the constructed tree. Write contents of tree into a new file using in-order.
2	Construct B-Tree an order of 5 with a set of 100 random elements stored in array. Implement searching, insertion and deletion operations.
3	Construct Min and Max Heap using arrays, delete any element and display the content of the Heap.
4	Implement BFT and DFT for given graph, when graph is represented by a) Adjacency Matrix b) Adjacency Lists
5	Write a program for finding the biconnected components in a given graph.
6	Implement Quick sort and Merge sort and observe the execution time for various input sizes (Average, Worst and Best cases).
7	**Greedy Method:** a) Compare the performance of Single Source Shortest Paths using Greedy method when the graph is represented by adjacency matrix and adjacency lists. b) Implement Job Sequencing with deadlines using Greedy strategy.
8	Write a program to solve 0/1 Knapsack problem Using Dynamic Programming.
9	**Back Tracking:** a) Implement N-Queens Problem Using Backtracking. b) Use Backtracking strategy to solve 0/1 Knapsack problem.
10	Implement Travelling Sales Person Problem using Branch and Bound approach.
	Virtual Lab Experiments
1	https://www.w3resource.com/c-programming-exercises/tree/c-tree-exercises-10.php
2	https://www.geeksforgeeks.org/

Reference Books/Lab Manuals

1	"Introduction to Algorithms" by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Fourth Edition., The MIT Press Cambridge, Massachusetts London, England.
2	Fundamentals of Data Structures in C++, Horowitz Ellis, Sahni Sartaj, Mehta, Dinesh, 2nd Edition, Universities Press
3	An introduction to Data Structures with applications, Trembley & Sorenson, McGraw Hill

Course Outcomes: At the end of the course, the student should have acquired the ability to	
CO1	Learn and use AVL trees, B-Trees, Heaps, graph representations, and sorting algorithms. **Cognitive Level: L2**
CO2	Solve problems using dynamic programming, backtracking, greedy methods, and branch and bound techniques. **Cognitive Level: L3**
CO3	Measure and improve algorithm performance for different scenarios. **Cognitive Level: L3**
CO4	Implement and manage data structures and graphs for efficient data handling. **Cognitive Level: L3**
CO5	Practice solving various algorithmic problems in areas like sorting, graph algorithms, and optimization. **Cognitive Level: L3**
CO6	Address real-world computational issues and experiment to analyze and improve algorithm efficiency. **Cognitive Level: L3**

Course Internal Assessment & Evaluation		
Continuous Internal Evaluation (CIE): 30 Marks		
Assessment Tool	Marks	Course Outcomes addressed
Day-to-day Evaluation & Record work	20	CO1, CO2, CO3, CO4, CO5, CO6
Internal Practical Exam & Viva Voce	10	
Total CIE Marks	**30**	
Semester End Examination (SEE): 70 Marks		
Semester End Examination	**70**	CO1, CO2, CO3, CO4, CO5, CO6
Total Marks	**100**	

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

Database Management Systems Lab

(Common to CSE and CSM)

II B. Tech – I Semester								SRIT R23
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
	PCC	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		0	0	3	1.5	30	70	100

Course Objectives: This Course will enable students to

- Populate and query a database using SQL DDL/DML Commands.
- Declare and enforce integrity constraints on a database.
- Writing Queries using advanced concepts of SQL.
- Programming PL/SQL including procedures, functions, cursors and triggers.
- Understand the Basics of JDBC.

List of Experiments

S. No.	Title of the Experiment
1	Creation, altering and dropping of tables and inserting rows into a table (use constraints while creating tables) examples using SELECT command.
2	a) Queries (along with sub-Queries) using ANY, ALL, IN, EXISTS, NOTEXISTS, UNION, INTERSET, Constraints. Example - Select the roll number and name of the student who secured fourth rank in the class. b) Queries using Aggregate functions (COUNT, SUM, AVG, MAX and MIN), GROUP BY, HAVING and Creation and dropping of Views.
3	a) Queries using Aggregate functions (COUNT, SUM, AVG, MAX and MIN), GROUP BY, HAVING and Creation and dropping of Views. b) Queries using Conversion functions (to_char, to_number and to_date), string functions (Concatenation, lpad, rpad, ltrim, rtrim, lower, upper, initcap, length, substr and instr), date functions (Sysdate, next_day, add_months, last_day, months_between, least, greatest, trunc, round, to_char, to_date)
4	a) Create a simple PL/SQL program which includes declaration section, executable section and exception –Handling section (Ex. Student marks can be selected from the table and printed for those who secured first class and an exception can be raised if no records were found) b) Insert data into student table and use COMMIT, ROLLBACK and SAVEPOINT in PL/SQL block.
5	a) Develop a program that includes the features NESTED IF, CASE and CASE expression. The program can be extended using the NULLIF and COALESCE functions. b) Program development using WHILE LOOPS, numeric FOR LOOPS, nested loops using ERROR Handling, BUILT –IN Exceptions, USE defined Exceptions, RAISE-APPLICATION ERROR.
6	a) Programs development using creation of procedures, passing parameters IN and OUT of PROCEDURES. b) Program development using creation of stored functions, invoke functions in SQL Statements and write complex functions.
7	Develop programs using features parameters in a CURSOR, FOR UPDATE CURSOR, WHERE CURRENT of clause and CURSOR variables.
8	Develop Programs using BEFORE and AFTER Triggers, Row and Statement Triggers and INSTEAD OF Triggers.
9	Create a table and perform the search operation on table using indexing and non-indexing techniques.
10	a) Write a Java program to connect to a database using JDBC and insert values into it.

	b) Write a Java program to connect to a database using JDBC and delete values from it.
Virtual Lab Experiments:	
1	
2	

Reference Books/Lab Manuals:	
1	Oracle: The Complete Reference by Oracle Press
2	Rick F Vander Lans, "Introduction to SQL", Fourth Edition, Pearson Education, 2007
3	RamezElmasri, Shamkant, B. Navathe, "Database Systems", Pearson Education, 6th Edition, 2013.

Course Outcomes: At the end of the course, the student able to	
CO1	Create, modify, and manage tables with constraints, and efficiently insert data using SQL commands. **Cognitive Level: L3**
CO2	Write complex SQL queries with sub-queries, set operations, and aggregate functions. **Cognitive Level: L3**
CO3	Use SQL aggregate functions, GROUP BY, and HAVING clauses effectively. **Cognitive Level: L3**
CO4	Develop basic PL/SQL programs with exception handling, transaction control commands (COMMIT, ROLLBACK, SAVEPOINT) and handle specific query scenarios. **Cognitive Level: L3**
CO5	Implement conditional statements (NESTED IF, CASE) and loops (WHILE, FOR) in PL/SQL. **Cognitive Level: L3**
CO6	Apply stored procedures with parameters for enhanced data processing. **Cognitive Level: L3**

Course Internal Assessment & Evaluation		
Continuous Internal Evaluation (CIE) : 30 Marks		
Assessment Tool	Marks	Course Outcomes addressed
Day-to-day Evaluation & Record work	20	CO1, CO2, CO3, CO4, CO5, CO6
Internal Practical Exam & Viva Voce	10	
Total CIE Marks	**30**	
Semester End Examination (SEE): 70 Marks		
Semester End Examination	**70**	CO1, CO2, CO3, CO4, CO5, CO6
Total Marks	**100**	

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

Python Programming

(Skill Enhancement Course)

(Common to CSE and CSM)

II B. Tech – I Semester								SRIT R23
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
	SEC	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		0	1	2	2	30	70	100

Course Objectives: The main objectives of the course are to

- Introduce core programming concepts of Python programming language.
- Demonstrate about Python data structures like Lists, Tuples, Sets and dictionaries.
- Implement Functions, Modules and Regular Expressions in Python Programming.
- Implement Object Oriented Principles through Python.
- Create practical and contemporary applications using Python.

List of Experiments

S. No.	Title of the Experiment
1	**Introduction:** History of Python Programming, Thrust areas of Python, Installing Anaconda Python Distribution, Installing and using Jupyter Notebook. **Basics of Python Programming:** Identifiers, Keywords, Statements and Expressions, Variables, Operators, Precedence and Associativity, Data Types, Indentation, Comments, Reading Input, Printing Output, Type Conversions, the type () Function and Is Operator, Dynamic and Strongly Typed Language. **Sample Experiments:** a) Write a python script to swap two numbers without using a temporary variable. b) Demonstrate the following Operators in Python with suitable examples. a. Arithmetic Operators ii) Relational Operators iii) Assignment Operators iv) Logical Operators v) Bitwise Operators vi) Ternary Operator vii) Membership Operators viii) Identity Operators c) Write a python script to add and multiply complex numbers. d) Write a python script to a display a user-defined message. e) Write a python script to illustrate the use of type() function. f) Write a Python script to convert temperatures to and from Celsius, Fahrenheit.
2	**Control Flow Statements:** if statement, if-else statement, if...elif…else, Nested if statement, while Loop, for Loop, continue, break and pass statements, Catching Exceptions using try and except statement. **Sample Experiments:** a) Write a python script to find the largest among three numbers. Write a python script to display all prime numbers within an interval. b) Write a python script to print multiplication table of a given number. c) Write a python script to check a given number is Armstrong or not. d) Write a python script to check whether a given number is perfect or not. e) Write a python script to calculate the factorial of a given number. f) Write a python script to calculate sum of the digits of a given number.
3	**Functions:** Built-in Functions, commonly used Modules, Function Definition and Calling the function, return statement and void function, Scope and Lifetime of Variables, Default Arguments, Keyword Arguments, *args and **kwargs, Anonymous Functions, Command Line Arguments. **Strings**: Creating and Storing Strings, Basic String Operations, Accessing Characters in String by Index Number, String Slicing and Joining, String Methods, Formatting

	Strings. **Sample Experiments:** a) Write a python script to define a function with multiple return values. b) Write a python script to define a function using default arguments. c) Write a python script to check a given year is leap or not by passing year as an argument to a function isLeap(). d) Write a python script to check a given number is palindrome or not by passing a number as an argument to a function isPalindrome().
4	**Strings**: Creating and Storing Strings, Basic String Operations, Accessing Characters in String by Index Number, String Slicing and Joining, String Methods, Formatting Strings. **Lists:** Creating Lists, Basic List Operations, Indexing and Slicing in Lists, Built-In Functions used on Lists, List Methods, del Statement, List Comprehension. **Sample Experiments:** a) Write a python script to find the length of the string without using any library functions. b) Write a python script to check if the substring is present in a given string or not. c) Write a python script to perform the given operations on a list: d) Addition ii. Insertion iii. slicing e) Write a python script to perform any 5 built-in functions by taking any list. f) Write a python script to print characters from a string that are present at an even index. g) Write a python script to create a list by picking odd-index items from the first list and even index items from the second list.
5	**Dictionaries:** Creating Dictionary, Accessing and Modifying key:value pairs in Dictionaries, Built-In Functions used on Dictionaries, Dictionary Methods, del Statement. **Sample Experiments:** a) Write a python script to count the number of vowels in a string (No control flow allowed). b) Write a python script to check if a given key exists in a dictionary or not. c) Write a python script to add a new key-value pair to an existing dictionary. d) Write a python script to sum all the items in a given dictionary. e) Write a python script to count the occurrence of each element from a list. f) Write a python script to iterate a given list and check if a given element exists as a key's value in a dictionary. If not, delete it from the list.
6	**Tuples and Sets:** Creating Tuples, Basic Tuple Operations, tuple() Function, Indexing and Slicing in Tuples, Built-In Functions used on Tuples, Relation between Tuples and Lists, Relation between Tuples and Dictionaries, Using zip() Function, Sets, Set Methods, Frozenset. **Sample Experiments:** a) Write a python script to create tuples (name, age, address, college) for at least two members and concatenate the tuples and print the concatenated tuples. b) Write a python script to copy the specific elements from one tuple to another tuple. c) Write a python script to unpack a tuple into several variables. d) Write a Python script to find the repeated items in a tuple. e) Write a Python script to unzip a list of tuples into individual lists. f) Write a Python script to count the elements in a list until an element is a tuple.

	g) Write a Python script to calculate the product, by multiplying all the numbers in a given tuple.
7	**Files:** Types of Files, Creating and Reading Text Data, File Methods to Read and Write Data, Reading and Writing Binary Files, Pickle Module, Reading and Writing CSV Files, Python os and os.path Modules. **Sample Experiments:** a) Write a python script to sort words in a file and put them in another file. The output file should have only lower-case words, so any upper-case words from source must be lowered. b) Write a python script to print each line of a file in reverse order. c) Write a python script to compute the number of characters, words and lines in a file. d) Write a python script to create a simple text file, write the contents into the created file and display the same on to the console screen. e) Write a python script to remove all the occurrences of a given character from a text file; copy the resultant text into another text file. Find the total occurrences of the eliminated character and display the count along with the contents of the text file on to the console.
8	**Object-Oriented Programming:** Classes and Objects, Creating Classes in Python, Creating Objects in Python, Constructor Method, Classes with Multiple Objects, Class Attributes Vs Data Attributes, Encapsulation, Inheritance, Polymorphism. **Sample Experiments:** a) Write a python script to create, display, append, insert and reverse the order of the items in the array. b) Write a python script to add, transpose and multiply two matrices. c) Write a python script to illustrate the constructors. d) Write a python script to create a class that represents a shape. Include methods to calculate its area and perimeter. Implement subclasses for different shapes like circle, triangle, and square. e) Write a python script to create a class representing a shopping cart. Include methods for adding and removing items, and calculating the total price. f) Write a python script to create a calculator class. Include methods to perform basic arithmetic operations.
9	**Introduction to Data Science:** Functional Programming, JSON and XML in Python, NumPy with Python, Pandas. **Sample Experiments:** a) Write a Python program to check whether a JSON string contains complex object or not. b) Write a Python Program to demonstrate NumPy arrays creation using array () function. c) Write a Python program to demonstrate use of ndim, shape, size, dtype. d) Write a Python program to demonstrate basic slicing, integer and Boolean indexing. e) Write a Python program to find min, max, sum, cumulative sum of array. f) Create a dictionary with at least five keys and each key represent value as a list where this list contains at least ten values and convert this dictionary as a pandas data frame and explore the data through the data frame as follows: g) Apply head () function to the pandas data frame h) Perform various data selection operations on Data Frame i) Select any two columns from the above data frame, and observe the change in

	one attribute with respect to other attribute with scatter and plot operations in matplotlib.
10	a) Write a python script to create a Countdown Timer. b) Write a python script to design a number guessing game.

Reference Books/Lab Manuals	
1	Gowri shankar S, Veena A., Introduction to Python Programming, CRC Press.
2	Python Programming, S Sridhar, J Indumathi, V M Hariharan, 2nd Edition, Pearson, 2024.
3	Introduction to Programming Using Python, Y. Daniel Liang, Pearson.

Course Outcomes: At the end of the course, the student able to	
CO1	Learn basic Python concepts and solve simple problems. **Cognitive Level: L2**
CO2	Use Python features like functions, data structures, and file handling for data manipulation. **Cognitive Level: L3**
CO3	Design software with classes, objects, and OOP principles. **Cognitive Level: L3**
CO4	Analyze and visualize data using Python tools and libraries. **Cognitive Level: L3**
CO5	Create practical Python scripts for tasks like file manipulation and problem-solving. **Cognitive Level: L3**
CO6	Enhance problem-solving abilities with algorithm design and data manipulation techniques. **Cognitive Level: L3**

Course Internal Assessment & Evaluation		
Continuous Internal Evaluation (CIE) : 30 Marks		
Assessment Tool	Marks	Course Outcomes addressed
Day-to-day Evaluation & Record work	20	CO1, CO2, CO3, CO4, CO5, CO6
Internal Practical Exam & Viva Voce	10	
Total CIE Marks	**30**	
Semester End Examination (SEE): 70 Marks		
Semester End Examination	**70**	CO1, CO2, CO3, CO4, CO5, CO6
Total Marks	**100**	

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

Object Oriented Programming Through Java

(Common to CSE and CSM)

II B.Tech–II Semester								SRITR23
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
	PCC	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	0	0	3	30	70	100

Course Objectives: The learning objectives of this course are to

- Identify Java language components and how they work together in applications.
- Learn the fundamentals of object-oriented programming in Java, including defining classes, invoking methods, using class libraries.
- Learn how to extend Java classes with inheritance and dynamic binding and how to use exception handling in Java applications.
- Understand how to design applications with threads in Java.
- Understand how to use Java apisfor program development.

Unit I: Introduction to Object Oriented Programming

Basic concepts: Principles, Program Structure in Java, Writing Simple Java Programs, Java Tokens, Java Statements, Command Line Arguments, User Input to Programs, Escape Sequences, Comments, Programming Style.

Data Types and Variables: Introduction, Data Types in Java, Declaration of Variables, Type Casting, Scope of Variable Identifier, Literal Constants, Symbolic Constants, Formatted Output with printf() Method, Static Variables and Methods, Attribute Final.

Operators: Assignment Operator (=), Basic Arithmetic Operators, Increment (++) and Decrement (- -) Operators, Ternary Operator, Relational Operators, Boolean Logical Operators, Bitwise Logical Operators. Precedence and Associativity of Operators.

Control Statements: Introduction, if Expression, Nested if Expressions, if–else Expressions, Ternary Operator ?:, Switch Statement, Iteration Statements, while Expression, do–while Loop, for Loop, Nested for Loop, For–Each for Loop, Break Statement, Continue Statement.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
  1. https://infyspringboard.onwingspan.com/web/en/app/toc/lex_auth_013847939122315264565 22_shared/overview
  2. https://cse.iitkgp.ac.in/~dsamanta/joywithjava/index.php

Unit II: Classes, Objects and Methods

Classes and Objects: Class Declaration and Modifiers, Class Members, Declaration of Class Objects, Assigning One Object to Another, Access Control for Class Members, Accessing Private Members of Class, Constructor Methods for Class, Overloaded Constructor Methods, Overriding Methods Nested Classes, Final Class and Methods, Passing Arguments by Value and by Reference, Keyword this.

Methods: Introduction, Defining Methods, Overloaded Methods, Class Objects as Parameters in Methods, Access Control, Recursive Methods, Nesting of Methods, Attributes Final and Static.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
  1. https://infyspringboard.onwingspan.com/web/en/app/toc/lex_auth_013501581834002432108 20/overview
  2. https://infyspringboard.onwingspan.com/web/en/app/toc/lex_auth_013501557419261952819 4/overview

Unit III: Arrays, Inheritance and Interfaces

Arrays: Introduction, Declaration and Initialization of Arrays, Storage of Array in Computer Memory, Accessing Elements of Arrays, Operations on Array Elements, Assigning Array to Another Array, Dynamic Change of Array Size, Sorting of Arrays, Search for Values in Arrays, Class Arrays, Two-dimensional Arrays, Arrays of Varying Lengths, Three-dimensional Arrays, Arrays as Vectors.

Inheritance: Introduction, Process of Inheritance, Types of Inheritances, Universal Super Class-Object Class, Inhibiting Inheritance of Class Using Final, Access Control and Inheritance,

Multilevel Inheritance, Application of Keyword Super, Constructor Method and Inheritance, Method Overriding, Dynamic Method Dispatch, Abstract Classes.

Interfaces: Introduction, Declaration of Interface, Implementation of Interface, Multiple Interfaces, Nested Interfaces, Inheritance of Interfaces, Default Methods in Interfaces, Static Methods in Interface, Functional Interfaces, Annotations.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
    1. https://infyspringboard.onwingspan.com/web/en/app/toc/lex_auth_013501561248161792947/overview
    2. https://infyspringboard.onwingspan.com/web/en/app/search/learning?lang=en&q=java%20inheritance

Unit IV: Packages, Exception Handling and Files

Introduction: Defining Package, Importing Packages and Classes into Programs, Path and Class Path, Access Control, Packages in Java SE, Java. lang Package and its Classes, Class Object, Enumeration, class Math, Wrapper Classes, Auto-boxing and Auto-unboxing, Java util Classes and Interfaces, Formatter Class, Random Class, Time Package, Class Instant (java.time.Instant), Formatting for Date/Time in Java, Temporal Adjusters Class, Temporal Adjusters Class.

Exception Handling: Introduction, Hierarchy of Standard Exception Classes, Keywords throws and throw, try, catch, and finally Blocks, Multiple Catch Clauses, Class Throwable, Unchecked Exceptions, Checked Exceptions.

Java I/O and File: Java I/O API, standard I/O streams, types, Byte streams, Character streams, Scanner class, Files in Java.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
    1. https://www.geeksforgeeks.org/packages-in-java/?ref=lbp
    2. https://www.javatpoint.com/exception-handling-in-java

Unit V: Strings, Multithreading, JDBC and Java FX

String Handling in Java: Interface Char Sequence, Class String, Methods for Extracting Characters from Strings, Comparison, Modifying, Searching; Class String Buffer.

Multithreaded Programming: Need for Multiple Threads Multithreaded Programming for Multi-core Processor, Thread Class, Main Thread-Creation of New Threads, Thread States, Thread Priority-Synchronization, Deadlock and Race Situations, Inter-thread Communication - Suspending, Resuming, and Stopping of Threads.

Java Database Connectivity: Introduction, JDBC Architecture, Installing MySQL and MySQL Connector/J, JDBC Environment Setup, Establishing JDBC Database Connections, ResultSet Interface.

Java FX GUI: Java FX Scene Builder, Java FX App Window Structure, displaying text and image, event handling, laying out nodes in scene graph, mouse events.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
    1. https://www.javatpoint.com/java-string
    2. https://infyspringboard.onwingspan.com/web/en/app/toc/lex_auth_013842053032140800010965_shared/overview

Text books

1	JAVA one step ahead, Anitha Seth, B.L.Juneja, Oxford.
2	The complete Reference Java, 11thedition, Herbert Schildt, TMH.

References	
1	JAVA 9 for Programmers, Paul Deitel, Harvey Deitel, 4th Edition, Pearson.
2	Introduction to Java programming, 7th Edition, Y Daniel Liang, Pearson.
3	Programming with Java: A Primer, 4e, Dr. E. Balagurusamy.

Course Outcomes: At the end of the course, the student should have acquired the ability to	
CO1	Learn Java fundamentals, including basic syntax, data types, variables, and program structure. **Cognitive Level: L2**
CO2	Write well-organized Java programs using data types, operators, and control flow statements. **Cognitive Level: L3**
CO3	Work with objects, constructors, methods, and manage access control in Java classes. **Cognitive Level: L3**
CO4	Use arrays, multi-dimensional arrays, inheritance, and abstract classes. **Cognitive Level: L3**
CO5	Define interfaces, handle exceptions, and perform file I/O operations. **Cognitive Level: L3**
CO6	Learn multithreading, JDBC for databases, and JavaFX for GUI development and design. **Cognitive Level: L4**

Course Assessment & Evaluation		
Continuous Internal Evaluation (CIE): 20 Marks		
Assessment Tool	Marks	Course Outcomes addressed
Continuous Internal Examination-I	20	CO1, CO2, CO3, CO4
Continuous Internal Examination-II	20	CO1, CO4, CO5, CO6
Final CIE marks shall be evaluated by considering 80% weightage to the best CIE and 20% weightage to the other CIE.		
Continuous Alternative Assessment (CAA): 10 marks The CAA may include assignments, seminars, term paper, open ended experiments, METE (Modeling and Experimental Tools in Engineering), five minutes video, MOOC's etc.		
Continuous Alternative Assessment-I	10	CO1, CO2, CO3, CO4
Continuous Alternative Assessment-II	10	CO1, CO4, CO5, CO6
Final CAA marks shall be evaluated by averaging CAA-I & CAA-II for 10 marks		
Semester End Examination (SEE): 70 marks		
Semester End Examination (SEE)	70	CO1, CO2, CO3, CO4, CO5, CO6
Total marks	**100**	

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

Operating Systems

(Computer Science and Engineering)

II B. Tech – II Semester								SRIT R23
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
	PCC	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	0	0	3	30	70	100

Course Objectives: The main objectives of the course are to

- Understand the basic concepts and principles of operating systems.
- Make use of process scheduling algorithms, threads.
- Make use of synchronization techniques to achieve better performance of a computer system.
- Explore the memory management, file systems, and Protection.
- Illustrate different conditions for deadlock and their possible solutions.

Unit I: Operating Systems Overview

Overview: Introduction, Operating system functions, Operating systems operations, Computing environments, Free and Open-Source Operating Systems.

System Structures: Operating System Services, User and Operating-System Interface, system calls, Types of System Calls, system programs (system services), Operating system Design and Implementation, Operating system structure, Building and Booting an Operating System, Operating system debugging.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
    1. https://archive.nptel.ac.in/courses/106/105/106105214/

Unit II: Process Management

Processes: Process Concept, Process scheduling, Operations on processes, Inter-process communication.
Threads and Concurrency: Overview, Multithreading models, Thread libraries, Threading issues.
CPU Scheduling: Basic concepts, Scheduling criteria, Scheduling algorithms, Multiple processor scheduling.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
    1. https://archive.nptel.ac.in/courses/106/105/106105214/

Unit III: Process Coordination

Synchronization Tools: The Critical Section Problem, Peterson's Solution, Mutex Locks, Semaphores, Monitors, Classic problems of Synchronization.
Deadlocks: system Model, Deadlock characterization, Methods for handling Deadlocks, Deadlock prevention, Deadlock avoidance, Deadlock detection, Recovery from Deadlock.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
    1. https://archive.nptel.ac.in/courses/106/105/106105214/

Unit IV: Memory Management

Main Memory: Introduction, Contiguous memory allocation, Paging, Structure of the Page Table, Swapping.
Virtual Memory Management: Introduction, Demand paging, Copy-on-write, Page replacement, Allocation of frames, Thrashing
Storage Management: Overview of Mass Storage Structure, HDD Scheduling.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
    1. https://archive.nptel.ac.in/courses/106/105/106105214/

Unit V: File System & Protection

File System: File System Interface, File concept, Access methods, Directory Structure.
File system Implementation: File-system structure, File-system Operations, Directory implementation, Allocation method, Free space management.
File-System Internals: File-System Mounting, Partitions and Mounting, File Sharing.
Protection: Goals of protection, Principles of protection, Protection Rings, Domain of protection, Access matrix.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:
    1. https://archive.nptel.ac.in/courses/106/105/106105214/

Text books:	
1	Operating System Concepts, Silberschatz A, Galvin P B, Gagne G, 10th Edition, Wiley, 2018.
2	Modern Operating Systems, Tanenbaum A S, 4th Edition, Pearson, 2016.

References:	
1	Operating Systems -Internals and Design Principles, Stallings W, 9 th edition, Pearson, 2018.
2	Operating Systems: A Concept Based Approach, D.M Dhamdhere, 3 rd Edition,McGraw- Hill, 2013.
3	Operating Systems: A Concept Based Approach, D.M Dhamdhere, 3 rd Edition.

Course Outcomes: At the end of the course, the student should have acquired the ability to	
CO1	Learn OS functions like process management, memory techniques, and file system structures. **Cognitive Level: L2**
CO2	Explore computing environments, free/open-source systems, and analyze system services and debugging. **Cognitive Level: L2**
CO3	Manage processes, threads, CPU scheduling, and solve multi-processor challenges. **Cognitive Level: L3**
CO4	Apply mutex locks, semaphores, and monitors for critical sections, and manage deadlocks. **Cognitive Level: L3**
CO5	Understand memory management methods like paging, page tables, and virtual memory concepts. **Cognitive Level: L2**
CO6	Manage file systems, including operations, directory management, and protection principles. **Cognitive Level: L3**

Course Assessment & Evaluation		
Continuous Internal Evaluation (CIE): 20 Marks		
Assessment Tool	Marks	Course Outcomes addressed
Continuous Internal Examination-I	20	CO1, CO2, CO3, CO4
Continuous Internal Examination-II	20	CO1, CO4, CO5, CO6
Final CIE marks shall be evaluated by considering 80% weightage to the best CIE and 20% weightage to the other CIE.		
Continuous Alternative Assessment (CAA): 10 marks The CAA may include assignments, seminars, term paper, open ended experiments, METE (Modeling and Experimental Tools in Engineering), five minutes video, MOOC's etc.		
Continuous Alternative Assessment-I	10	CO1, CO2, CO3, CO4
Continuous Alternative Assessment-II	10	CO1, CO4, CO5, CO6
Final CAA marks shall be evaluated by averaging CAA-I & CAA-II for 10 marks		
Semester End Examination (SEE): 70 marks		
Semester End Examination (SEE)	70	CO1, CO2, CO3, CO4, CO5, CO6
Total marks	**100**	

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

Software Engineering

(Computer Science and Engineering)

II B. Tech – II Semester								SRIT R23
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
		L	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
23A05403	**PCC**	3	0	0	3	30	70	100

Course Objectives: The objectives of this course are

- Understand the evolution and development of software engineering.
- Software life cycle models, Software requirements and SRS document.
- Understand Requirements Analysis and Specification.
- Project Planning, quality control and ensuring good quality software.
- Software Testing strategies, use of CASE tools, Implementation issues, validation & verification procedures.

Unit I: Basics Of Software Engineering

Introduction: Evolution, Software development projects, Exploratory style of software developments, Emergence of software engineering, Notable changes in software development practices, Computer system engineering.

Software Life Cycle Models: Basic concepts, Waterfall model and its extensions, Rapid application development, Agile development model, Spiral model.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:

Unit II: Project Management And Specification

Software Project Management: Software project management complexities, Responsibilities of a software project manager, Metrics for project size estimation, Project estimation techniques, Empirical Estimation techniques, COCOMO, Halstead's software science, risk management.

Requirements Analysis And Specification: Requirements gathering and analysis, Software Requirements Specification (SRS), Formal system specification, Axiomatic specification, Algebraic specification, Executable specification and 4GL.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:

Unit III: Design Concepts

Software Design: Overview of the design process, How to characterize a good software design? Layered arrangement of modules, Cohesion and Coupling. approaches to software design.

Agility: Agility and the Cost of Change, Agile Process, Extreme Programming (XP), Other Agile Process Models, Tool Set for the Agile Process .

Function-Oriented Software Design: Overview of SA/SD methodology, Structured analysis, Developing the DFD model of a system, Structured design, Detailed design, and Design Review

User Interface Design: Characteristics of a good user interface, Basic concepts, Types of user interfaces, Fundamentals of component-based GUI development, and user interface design methodology.

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:

Unit IV: Testing Strategies and Management

Coding And Testing: Coding, Code review, Software documentation, Testing, Black-box testing, White-Box testing, Debugging, Program analysis tools, Integration testing**,** Testing object-oriented programs, Smoke testing, and Some general issues associated with testing.

Software Reliability and Quality Management: Software reliability. Statistical testing, Software quality, Software quality management system, ISO 9000. SEI Capability maturity model. Few other important quality standards, and Six Sigma

- Pedagogy/Course Delivery tools: Chalk & Talk & ICT
- Links for online material:

Unit V: Software Maintenance and Reuse
Computer-Aided Software Engineering (Case): CASE and its scope, CASE environment, CASE support in the software life cycle, other characteristics of CASE tools, Towards second generation CASE Tool, and Architecture of a CASE Environment. **Software Maintenance:** Characteristics of software maintenance, Software reverse engineering, Software maintenance process models and Estimation of maintenance cost. **Software Reuse:** reuse- definition, introduction, reason behind no reuse so far, Basic issues in any reuse program, A reuse approach, and Reuse at organization level • Pedagogy/Course Delivery tools: Chalk & Talk & ICT • Links for online material:

Text books	
1	Fundamentals of Software Engineering, Rajib Mall, 5th Edition, PHI.
2	Software Engineering A practitioner's Approach, Roger S. Pressman, 9th Edition, Mc-Graw Hill International Edition.

References	
1	Software Engineering, Ian Sommerville,10th Edition, Pearson.
2	Software Engineering, Principles and Practices, Deepak Jain, Oxford University Press.
3	Software Engineering, an Engineering approach- James F. Peters, Witold Pedrycz, John Wiley.

Course Outcomes: At the end of the course, the student should have acquired the ability to	
CO1	Learn about software engineering principles like methodologies, project management, and design. **Cognitive Level: L2**
CO2	Explore the history of software engineering and understand different life cycle models for projects. **Cognitive Level: L3**
CO3	Manage projects, estimate sizes, and specify requirements using advanced techniques. **Cognitive Level: L3**
CO4	Use design principles, agile methodologies, and function-oriented design techniques. **Cognitive Level: L3**
CO5	Practice coding, testing, and using tools for debugging and quality management. **Cognitive Level: L3**
CO6	Use CASE tools, manage software maintenance, and apply software reuse strategies. **Cognitive Level: L3**

Course Assessment & Evaluation		
Continuous Internal Evaluation (CIE): 20 Marks		
Assessment Tool	Marks	Course Outcomes addressed
Continuous Internal Examination-I	20	CO1, CO2, CO3, CO4
Continuous Internal Examination-II	20	CO1, CO4, CO5, CO6
Final CIE marks shall be evaluated by considering 80% weightage to the best CIE and 20% weightage to the other CIE.		
Continuous Alternative Assessment (CAA): 10 marks The CAA may include assignments, seminars, term paper, open ended experiments, METE (Modeling and Experimental Tools in Engineering), five minutes video, MOOC's etc.		
Continuous Alternative Assessment-I	10	CO1, CO2, CO3, CO4
Continuous Alternative Assessment-II	10	CO1, CO4, CO5, CO6
Final CAA marks shall be evaluated by averaging CAA-I & CAA-II for 10 marks		
Semester End Examination (SEE): 70 marks		
Semester End Examination (SEE)	70	CO1, CO2, CO3, CO4, CO5, CO6
Total marks	**100**	

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

Object-Oriented Programming through JAVA Lab

(Common to CSE and CSM)

II B. Tech – II Semester							SRIT R23
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**	
	PCC	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		0	0	3	1.5	30	70	100

Course Objectives: The main objectives of the course are to

- Learn core OOP principles.
- Create and use classes and objects.
- Explore inheritance and polymorphism.
- Handle exceptions and file I/O.
- Build graphical user interfaces with JavaFX.

List of Experiments

S. No.	Title of the Experiment
1	a) Write a JAVA program to display default value of all primitive data type of JAVA. b) Write a java program that display the roots of a quadratic equation $ax^2+bx=0$. Calculate the discriminate D and basing on value of D, describe the nature of root.
2	a) Implement class mechanism in Java. Create a class, methods and invoke them inside main method. b) Write a JAVA program implement method overloading. c) Write a JAVA program to implement constructor.
3	a) Implement constructor overloading in JAVA. b) Write a JAVA program to search for an element in a given list of elements using binary search mechanism. c) Develop a JAVA program to sort for an element in a given list of elements using bubble sort.
4	a) Write a JAVA program to implement Single Inheritance. b) Write a JAVA program to implement multi-level Inheritance. c) Construct abstract class in JAVA to find areas of different shapes.
5	b) Write a JAVA program to implement Interface. What kind of Inheritance can be achieved? c) Write a JAVA program that implements Runtime polymorphism. d) Write a JAVA program using StringBuffer to delete, remove character.
6	a) Write a JAVA program that describes exception handling mechanism. b) Write a JAVA program Illustrating Multiple catch clauses. c) Write a JAVA program for creation of Java Built-in Exceptions.
7	a) Write a JAVA program for creation of User Defined Exception. b) Write a JAVA program that creates threads by extending Thread class. First thread display "Good Morning "every 1 sec, the second thread displays "Hello "every 2 seconds and the third display "Welcome" every 3 seconds, (Repeat the same by implementing Runnable) . c) Write a program illustrating **is Alive** and **join ().**
8	a) Write a Program illustrating Daemon Threads. b) Write a JAVA program Producer Consumer Problem. c) Write a JAVA program that import and use the user defined packages.
9	a) Write a java program that connects to a database using JDBC. b) Write a java program to connect to a database using JDBC and insert values into it. c) Write a java program to connect to a database using JDBC and delete values from it.
10	a) Without writing any code, build a GUI that display text in label and image in an ImageView (use JavaFX). b) Build a Tip Calculator app using several JavaFX components and learn how to respond to user interactions with the GUI.
Virtual Lab Experiments	
1	Introduction to Java Programming Language. https://virtual-labs.github.io/exp-introduction-java-iitd/index.html
2	Learning Abstraction through Interface. https://virtual-labs.github.io/exp-abstraction-iitd/

Reference Books/Lab Manuals	
1	JAVA one step ahead, Anitha Seth, B.L.Juneja, Oxford.
2	Joy with JAVA, Fundamentals of Object-Oriented Programming, Debasis Samanta, Monalisa Sarma, Cambridge, 2023.
3	The complete Reference Java, 11th edition, Herbert Schildt,TMH

Course Outcomes: At the end of the course, the student able to	
CO1	Learn Java basics like primitive data types and solve quadratic equations. **Cognitive Level: L2**
CO2	Create classes, use method overloading, and work with constructors in Java. **Cognitive Level: L3**
CO3	Write Java programs for algorithms such as binary search and bubble sort. **Cognitive Level: L3**
CO4	Use inheritance and abstract classes to model shapes and calculate areas. **Cognitive Level: L3**
CO5	Implement interfaces for multiple inheritances, use method overriding for polymorphism, and manipulate text with `'StringBuffer'`. **Cognitive Level: L3**
CO6	Handle exceptions, work with threads and Daemon threads, and understand the Producer-Consumer problem. **Cognitive Level: L3**

Course Internal Assessment & Evaluation		
Continuous Internal Evaluation (CIE) : 30 Marks		
Assessment Tool	Marks	Course Outcomes addressed
Day-to-day Evaluation & Record work	20	CO1, CO2, CO3, CO4, CO5, CO6
Internal Practical Exam & Viva Voce	10	
Total CIE Marks	**30**	
Semester End Examination (SEE): 70 Marks		
Semester End Examination	**70**	CO1, CO2, CO3, CO4, CO5, CO6
Total Marks	**100**	

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

Operating Systems Lab

(Computer Science and Engineering)

II B. Tech – II Semester								SRIT R23
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
	PCC	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		0	0	3	1.5	30	70	100

Course Objectives: The main objectives of the course are to

- Practicing basic Unix commands and Provide insights into system calls, file systems, semaphores.
- Develop and debug CPU Scheduling algorithms, page replacement algorithms,
- thread implementation.
- Implement Bankers Algorithms to Avoid the Dead Lock.
- Simulate the file allocation strategies.

List of Experiments

S. No.	Title of the Experiment
1	a) Practicing of Basic UNIX Commands. b) Simulate UNIX commands like cp, ls, grep, etc.,
2	Write programs using the following UNIX operating system calls fork, exec, getpid, exit, wait, close, stat, opendir and readdir.
3	Simulate the following CPU scheduling algorithms. a) FCFS b) SJF c) Priority d) Round Robin
4	Control the number of ports opened by the operating system with a) Semaphore b) Monitors.
5	Write a program to illustrate concurrent execution of threads using pthreads library.
6	a) Write a program to solve producer-consumer problem using Semaphores. b) Implement Bankers Algorithm for Dead Lock avoidance and prevention.
7	Implement the following memory allocation methods for fixed partition. a) First fit b) Worst fit c) Best fit
8	Simulate the following page replacement algorithms. a) FIFO b) LRU c) LFU
9	Simulate Paging Technique of memory management.
10	Simulate the following file allocation strategies. a) Sequential b) Indexed c) Linked
Virtual Lab Experiments	
1	Simulate the following CPU scheduling algorithms a) L J F b) SRJF 1. https://naim30.github.io/OS-virtual-lab/shedulingAlgo/shedulingAlgo.html
2	Implement Peterson algorithm for critical section problem. 1. https://mail.google.com/mail/u/0/?tab=rm&ogbl#inbox/FMfcgzQVxRKgPNfQfjmjmQzpjvnXRTDq?projector=1&messagePartId=0.3

Reference Books/Lab Manuals	
1	Operating System Concepts, Silberschatz A, Galvin P B, Gagne G, 10 th Edition, Wiley, 2018.
2	Modern Operating Systems, Tanenbaum A S, 4 th Edition, Pearson, 2016.
3	Operating Systems -Internals and Design Principles, Stallings W, 9 th edition, Pearson, 2018.

Course Outcomes: At the end of the course, the student able to	
CO1	Demonstrate Proficiency in UNIX Commands and System Calls. **Cognitive Level: L2**
CO2	Implement and Evaluate CPU Scheduling Algorithms. **Cognitive Level: L3**
CO3	Utilize Semaphore and Monitor Concepts for Process Synchronization. **Cognitive Level: L3**
CO4	Implement Memory Allocation and Paging Techniques. **Cognitive Level: L3**
CO5	Design and Implement File Allocation Strategies. **Cognitive Level: L3**
CO6	Use concepts like deadlock avoidance, page replacement, and memory management to solve problems. **Cognitive Level: L3**

Course Internal Assessment & Evaluation		
Continuous Internal Evaluation (CIE) : 30 Marks		
Assessment Tool	Marks	Course Outcomes addressed
Day-to-day Evaluation & Record work	20	CO1, CO2, CO3, CO4, CO5, CO6
Internal Practical Exam & Viva Voce	10	
Total CIE Marks	**30**	
Semester End Examination (SEE): 70 Marks		
Semester End Examination	**70**	CO1, CO2, CO3, CO4, CO5, CO6
Total Marks	**100**	

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

Full Stack Development - I

(Common to CSE and CSM)

II B. Tech – II Semester								SRIT R23
Course Code	**Category**	**Hours/Week**			**Credits**	**Maximum Marks**		
	SEC	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		0	1	2	2	30	70	100

Course Objectives: The main objectives of the course are to

- Make use of HTML elements and their attributes for designing static web pages
- Apply different types of CSS styles
- Build a web page by applying appropriate CSS styles to HTML elements
- Experiment with JavaScript to develop dynamic web pages and validate forms
- Understand the basics of Node.js

List of Experiments

S. No.	Title of the Experiment
1	**Links, Lists and Images** a) Write a HTML program, to explain the working of lists. Note: It should have an ordered list, unordered list, nested lists and ordered list in an unordered list and definition lists. b) Write a HTML program, to explain the working of hyperlinks using <a> tag and href, target Attributes. c) Create a HTML document that has your image and your friend's image with a specific height and width. Also, when clicked on the images it should navigate to their respective profiles. d) Write a HTML program, in such a way that, rather than placing large images on a page, the preferred technique is to use thumbnails by setting the height and width parameters to something like to 100*100 pixels. Each thumbnail image is also a link to a full-sized version of the image. Create an image gallery using this technique
2	**HTML Forms, Table and Frames** a) Write a HTML program, to explain the working of tables. (use tags: <table>, <tr>, <th>, <td> and attributes: border, rowspan, colspan) b) Write a HTML program, to explain the working of tables by preparing a timetable. (Note: Use <caption> tag to set the caption to the table & also use cell spacing, cell padding, border, rowspan, colspan etc.). c) Write a HTML program, to explain the working of forms by designing Registration form. (Note: Include text field, password field, number field, date of birth field, checkboxes, radio buttons, list boxes using <select>&<option> tags, <text area> and two buttons ie: submit and reset. Use tables to provide a better view). d) Write a HTML program, to explain the working of frames, such that page is to be divided into 3 parts on either direction. (Note: first frame → image, second frame → paragraph, third frame → hyperlink. And also make sure of using "no frame" attribute such that frames to be fixed).
3	**CSS** a) Write a HTML program, that makes use of <article>, <aside>, <figure>, <figcaption>, <footer>, <header>, <main>, <nav>, <section>, , tags. b) Write a HTML program, to embed audio and video into HTML web page. c) Write a program to apply different types (or levels of styles or style specification formats) - inline, internal, external styles to HTML elements. (identify selector, property and value).
4	**CSS Selectors** Write a program to apply different types of selector forms • Simple selector (element, id, class, group, universal)

	• Combinator selector (descendant, child, adjacent sibling, general sibling) • Pseudo-class selector • Pseudo-element selector • Attribute selector
5	**CSS Properties** a) Write a program to demonstrate the various ways you can reference a color in CSS. b) Write a CSS rule that places a background image halfway down the page, tilting it horizontally. The image should remain in place when the user scrolls up or down. c) Write a program using the following terms related to CSS font and text: i. font-size ii. font-weight iii. font-style iv. text-decoration v. text-transformation vi. text-alignment d) Write a program, to explain the importance of CSS Box model using Content ii. Border iii. Margin iv. padding
6	**JavaScript – Basics** a) Write a program to embed internal and external JavaScript in a web page. b) Write a program to explain the different ways for displaying output. c) Write a program to explain the different ways for taking input. d) Create a webpage which uses prompt dialogue box to ask a voter for his name and age. Display the information in table format along with either the voter can vote or not
7	**JavaScript – Objects** a) Write a program using document object properties and methods. b) Write a program using window object properties and methods. c) Write a program using array object properties and methods. d) Write a program using math object properties and methods. e) Write a program using string object properties and methods. f) Write a program using regex object properties and methods. g) Write a program using date object properties and methods. h) Write a program to explain user-defined object by using properties, methods, accessors, constructors and display.
8	**JavaScript – Loops** a) Write a program which asks the user to enter three integers, obtains the numbers from the user and outputs HTML text that displays the larger number followed by the words "LARGER NUMBER" in an information message dialog. If the numbers are equal, output HTML text as "EQUAL NUMBERS". b) Write a program to display week days using switch case. c) Write a program to print 1 to 10 numbers using for, while and do-while loops. d) Write a program to print data in object using for-in, for-each and for-of loops e) Develop a program to determine whether a given number is an 'ARMSTRONG NUMBER' or not. [Eg: 153 is an Armstrong number, since sum of the cube of the digits is equal to the number i.e.,13 + 53+ 33 = 153] f) Write a program to display the denomination of the amount deposited in the bank in terms of 100's, 50's, 20's, 10's, 5's, 2's & 1's. (Eg: If deposited amount is Rs.163, the output should be 1-100's, 1-50's, 1- 10's, 1-2's & 1-1's).
9	**JavaScript – Functions and Events** a) Design a appropriate function should be called to display i. Factorial of that number ii. Fibonacci series up to that number iii. Prime numbers up to that number iv. Is it palindrome or not b) Design a HTML having a text box and four buttons named Factorial, Fibonacci,

	Prime, and Palindrome. When a button is pressed an appropriate function should be called to display i. Factorial of that number ii. Fibonacci series up to that number iii. Prime numbers up to that number iv. Is it palindrome or not c) Write a program to validate the following fields in a registration page i. Name (start with alphabet and followed by alphanumeric and the length should not be less than 6 characters) ii. Mobile (only numbers and length 10 digits) iii. E-mail (should contain format like xxxxxxx@xxxxxx.xxx)
10	**NodeJS** a) Install and Configure the Node Js. b) Write a program to display the "Hello World" using node.js.

Reference Books/Lab Manuals	
1	John Dean, Web Programming with HTML5, CSS and JavaScript, Jones & Bartlett Learning, 2019.
2	Web Programming: Building Internet Applications, 3ed, Chris Bates, Wiley, 2007.
3	Programming the World Wide Web, 7th Edition, Robet W Sebesta, Pearson, 2013.

Course Outcomes: At the end of the course, the student able to	
CO1	Create visually appealing and user-friendly websites. **Cognitive Level: L6**
CO2	Implement effective styling techniques to enhance web page aesthetics. **Cognitive Level: L3**
CO3	Develop interactive elements to improve user engagement on web pages. **Cognitive Level: L3**
CO4	Design and integrate functional forms for application processes. **Cognitive Level: L6**
CO5	Select appropriate control structures to optimize code logic and performance. **Cognitive Level: L4**
CO6	Utilize advanced web design principles to build professional websites. **Cognitive Level: L3**

Course Internal Assessment & Evaluation		
Continuous Internal Evaluation (CIE) : 30 Marks		
Assessment Tool	Marks	Course Outcomes addressed
Day-to-day Evaluation & Record work	20	CO1, CO2, CO3, CO4, CO5, CO6
Internal Practical Exam & Viva Voce	10	
Total CIE Marks	**30**	
Semester End Examination (SEE): 70 Marks		
Semester End Examination	**70**	CO1, CO2, CO3, CO4, CO5, CO6
Total Marks	**100**	