DATA STRUCTURES AND APPLICATION

(Effective from the academic year 2018 -2019)

SEMESTER – III

Subject Code	18CS32	**CIE Marks**	40
Number of Contact Hours/Week	3:2:0	**SEE Marks**	60
Total Number of Contact Hours	50	**Exam Hours**	3 Hrs

CREDITS –4

Course Learning Objectives: This course (18CS32) will enable students to:

- Explain fundamentals of data structures and their applications essential for programming/problem solving
- Illustrate linear representation of data structures: Stack, Queues, Lists, Trees and Graphs
- Demonstrate sorting and searching algorithms
- Find suitable data structure during application development/Problem Solving

Module 1	**Contact Hours**
Introduction: Data Structures, Classifications (Primitive & Non Primitive), Data structure Operations, Review of Arrays, Structures, Self-Referential Structures, and Unions. Pointers and Dynamic Memory Allocation Functions. Representation of Linear Arrays in Memory, Dynamically allocated arrays. **Array Operations**: Traversing, inserting, deleting, searching, and sorting. Multidimensional Arrays, Polynomials and Sparse Matrices. **Strings:** Basic Terminology, Storing, Operations and Pattern Matching algorithms. Programming Examples. **Textbook 1: Chapter 1: 1.2, Chapter 2: 2.2 - 2.7** **Text Textbook 2: Chapter 1: 1.1 - 1.4,** **Chapter 3: 3.1 - 3.3, 3.5, 3.7,** **Ch apter 4: 4.1 - 4.9, 4.14** **Reference 3: Chapter 1: 1.4** **RBT: L1, L2, L3**	10
Module 2	
Stacks: Definition, Stack Operations, Array Representation of Stacks, Stacks using Dynamic Arrays, Stack Applications: Polish notation, Infix to postfix conversion, evaluation of postfix expression. **Recursion -** Factorial, GCD, Fibonacci Sequence, Tower of Hanoi, Ackerman's function. **Queues:** Definition, Array Representation, Queue Operations, Circular Queues, Circular queues using Dynamic arrays, Dequeues, Priority Queues, A Mazing Problem. Multiple Stacks and Queues. Programming Examples. **Textbook 1: Chapter 3: 3.1 -3.7** **Textbook 2: Chapter 6: 6.1 -6.3, 6.5, 6.7-6.10, 6.12, 6.13** **RBT: L1, L2, L3**	10
Module 3	
Linked Lists: Definition, Representation of linked lists in Memory, Memory allocation; Garbage Collection. Linked list operations: Traversing, Searching, Insertion, and Deletion. Doubly Linked lists, Circular linked lists, and header linked lists. Linked Stacks and Queues.	10

Applications of Linked lists – Polynomials, Sparse matrix representation. Programming Examples **Textbook 1: Ch apter 4: 4.1 – 4.6, 4.8** **Textbook 2: Ch apter 5: 5.1 – 5.10** **RBT: L1, L2, L3**	
Module 4	
Trees: Terminology, Binary Trees, Properties of Binary trees, Array and linked Representation of Binary Trees, Binary Tree Traversals - Inorder, postorder, preorder; Additional Binary tree operations. Threaded binary trees, Binary Search Trees – Definition, Insertion, Deletion, Traversal, Searching, Application of Trees-Evaluation of Expression, Programming Examples **Textbook 1: Chapter 5: 5.1 –5.5, 5.7** **Textbook 2: Chapter 7: 7.1 – 7.9** **RBT: L1, L2, L3**	10
Module 5	
Graphs: Definitions, Terminologies, Matrix and Adjacency List Representation Of Graphs, Elementary Graph operations, Traversal methods: Breadth First Search and Depth First Search. **Sorting and Searching**: Insertion Sort, Radix sort, Address Calculation Sort. **Hashing:** Hash Table organizations, Hashing Functions, Static and Dynamic Hashing. **Files and Their Organization:** Data Hierarchy, File Attributes, Text Files and Binary Files, Basic File Operations, File Organizations and Indexing **Textbook 1: Chapter 6 : 6.1 –6.2, Chapter 7:7.2, Chapter 8 : 8.1-8.3** **Textbook 2: Chapter 8 : 8.1 – 8.7, Chapter 9 : 9.1-9.3, 9.7, 9.9** **Reference 2: Chapter 16 : 16.1 - 16.7** **RBT: L1, L2, L3**	10
Course Outcomes: The student will be able to :	
• Use different types of data structures, operations and algorithms • Apply searching and sorting operations on files • Use stack, Queue, Lists, Trees and Graphs in problem solving • Implement all data structures in a high-level language for problem solving.	
Question Paper Pattern:	
• The question paper will have ten questions. • Each full Question consisting of 20 marks • There will be 2 full questions (with a maximum of four sub questions) from each module. • Each full question will have sub questions covering all the topics under a module. • The students will have to answer 5 full questions, selecting one full question from each module.	
Textbooks:	
1. Ellis Horowitz and Sartaj Sahni, Fundamentals of Data Structures in C, 2nd Ed, Universities Press, 2014. 2. Seymour Lipschutz, Data Structures Schaum's Outlines, Revised 1st Ed, McGraw Hill, 2014.	
Reference Books:	
1. Gilberg & Forouzan, Data Structures: A Pseudo-code approach with C, 2nd Ed, Cengage Learning,2014	

2. Reema Thareja, Data Structures using C, 3rd Ed, Oxford press, 2012.
3. Jean-Paul Tremblay & Paul G. Sorenson, An Introduction to Data Structures with Applications, 2nd Ed, McGraw Hill, 2013
4. A M Tenenbaum, Data Structures using C, PHI, 1989
5. Robert Kruse, Data Structures and Program Design in C, 2nd Ed, PHI, 1996.

ANALOG AND DIGITAL ELECTRONICS
(Effective from the academic year 2018 -2019)
SEMESTER – III

Subject Code	18CS33	**CIE Marks**	40
Number of Contact Hours/Week	3:0:0	**SEE Marks**	60
Total Number of Contact Hours	40	**Exam Hours**	3 Hrs

CREDITS –3

Course Learning Objectives: This course (18CS33) will enable students to:

- Explain the use of photoelectronics devices, 555 timer IC, Regulator ICs and uA741 opamap IC
- Make use of simplifying techniques in the design of combinational circuits.
- Illustrate combinational and sequential digital circuits
- Demonstrate the use of flipflops and apply for registers
- Design and test counters, Analog-to-Digital and Digital-to-Analog conversion techqniues.

Module 1	**Contact Hours**
Optoelectronic Devices: Photodiodes, Phototransistors, Light Emitting Diodes, Liquid Crystal Displays, and Optocouplers. **Wave Shaping Circuits:** Integrated Circuit Multivibrators **Linear Power Supplies:** Linear IC Voltage, Regulated Power Suppy Parameters **Operational Amplifier Application Circuits**: Inverting Amplifier, Non-inverting amplifier, Voltage Follower, Summing Amplifier, Difference Amplifier, Averagor, Integrator, Differentiator, Peak Detector, Absolute Value Circuit, Comparotor, Instrumentation Amplifier, Relaxation Oscillator, Current-to-Voltage and Voltage-to-Current Converter **Textbook 1: Chapter7 – 7.4, 7.5, 7.10, 7.11, 7.14; Chapter13 – 13.10; Chapter14 – 14.6, 14.7;** **Chapter17 – 17.1, 17.2, 17.3, 17.4, 17.5, 17.6, 17.7, 17.8, 17.12, 17.13, 17.14, 17.17, 17.19, 17.20, 17.21** **RBT: L1, L2**	08
Module 2	
Combinational Logic Circuits: Sum-of-Products Method, Truth Table to Karnaugh Map, Pairs Quads, and Octets, Karnaugh Simplifications, Don't-care Conditions, Product-of-sums Method, Product-of-sums simplifications, Simplification by Quine-McClusky Method Introduction to HDL, HDL Implementation Models. **Text book 2: Chapter2 – 2.5; Chapter3 – 3.2 to 3.9, 3.11.** **RBT: L1, L2**	08
Module 3	
Data-Processing Circuits: Multiplexers, Demultiplexers, 1-of-16 Decoder, BCD to Decimal Decoders, Seven Segment Decoders, Encoders, Exclusive-OR Gates, Parity Generators and Checkers, Magnitude Comparator, Programmable Array Logic, Programmable Logic Arrays, HDL Implementation of Data Processing Circuits. **Text book 2: Chapter4 – 4.1 to 4.9, 4.11, 4.12, 4.14.**	08

RBT: L1, L2, L3	
Module 4	
Flip- Flops: RS Flip-Flops, Gated Flip-Flops, Edge-triggered RS FLIP-FLOP, Edge-triggered D FLIP-FLOPs, Edge-triggered JK FLIP-FLOPs, FLIP-FLOP Timing, JK Master-slave FLIP-FLOP, HDL Implementation of FLIP-FLOP. **Registers:** Types of Registers, Serial In - Serial Out, Serial In - Parallel out, Parallel In - Serial Out, Parallel In - Parallel Out, Universal Shift Register, Applications of Shift Registers. **Text book 2: Chapter8 – 8.1 to 8.7, 8.12; Chapter9: 9.1 to 9.6** **RBT: L1, L2, L3**	08
Module 5	
Counters: Asynchronous Counters, Decoding Gates, Synchronous Counters, Changing the Counter Modulus, Decade Counters, Counter Design using HDL. **D/A Conversion and A/D Conversion:** Variable, Resistor Networks, Binary Ladders, D/A Converters, D/A Accuracy and Resolution, A/D Converter-Simultaneous Conversion, A/D Converter-Counter Method, Continuous A/D Conversion **Text book 2:- Chapter10 – 10.1 to 10.5, 10.9; Ch 12: 12.1 to 12.7** **RBT: L1, L2, L3**	08
Course Outcomes: The student will be able to :	
• Design and analyze application analog circuits using photodevices, timer IC, power supply and regulator IC and opamp. • Simplify digital circuits using Karnaugh Map , POS and Quine-McClusky Methods • Explain Gates and flipflops and make us in designing different data processing circuits, registers and counters and compare the types. • Develop simple HDL programs • Explain the basic principles of A/D and D/A conversion circuits and develop the same.	
Question Paper Pattern:	
• The question paper will have ten questions. • Each full Question consisting of 20 marks • There will be 2 full questions (with a maximum of four sub questions) from each module. • Each full question will have sub questions covering all the topics under a module. • The students will have to answer 5 full questions, selecting one full question from each module.	
Textbooks:	
1. Anil K Maini, Varsha Agarwal, Electronic Devices and Circuits, Wiley, 2012. 2. Donald P Leach, Albert Paul Malvino & Goutam Saha, Digital Principles and Applications, 8th Edition, Tata McGraw Hill, 2015.	
Reference Books:	
1. M. Morris Mani, Digital Design, 4th Edition, Pearson Prentice Hall, 2008. 2. David A. Bell, Electronic Devices and Circuits, 5th Edition, Oxford University Press, 2008	

COMPUTER ORGANIZATION
(Effective from the academic year 2018 -2019)
SEMESTER – III

Subject Code	18CS34	**CIE Marks**	40
Number of Contact Hours/Week	3:0:0	**SEE Marks**	60
Total Number of Contact Hours	40	**Exam Hours**	3 Hrs

CREDITS –3

Course Learning Objectives: This course (18CS34) will enable students to:

- Explain the basic sub systems of a computer, their organization, structure and operation.
- Illustrate the concept of programs as sequences of machine instructions.
- Demonstrate different ways of communicating with I/O devices and standard I/O interfaces.
- Describe memory hierarchy and concept of virtual memory.
- Describe arithmetic and logical operations with integer and floating-point operands.
- Illustrate organization of a simple processor, pipelined processor and other computing systems.

Module 1	**Contact Hours**
Basic Structure of Computers: Basic Operational Concepts, Bus Structures, Performance – Processor Clock, Basic Performance Equation, Clock Rate, Performance Measurement. Machine Instructions and Programs: Memory Location and Addresses, Memory Operations, Instructions and Instruction Sequencing, Addressing Modes, Assembly Language, Basic Input and Output Operations, Stacks and Queues, Subroutines, Additional Instructions, Encoding of Machine Instructions **Text book 1: Chapter1 – 1.3, 1.4, 1.6 (1.6.1-1.6.4, 1.6.7), Chapter2 – 2.2 to 2.10** **RBT: L1, L2, L3**	08
Module 2	
Input/Output Organization: Accessing I/O Devices, Interrupts – Interrupt Hardware, Enabling and Disabling Interrupts, Handling Multiple Devices, Controlling Device Requests, Exceptions, Direct Memory Access, Buses, Interface Circuits, Standard I/O Interfaces – PCI Bus, SCSI Bus, USB. **Text book 1: Chapter4 – 4.1, 4.2 (4.2.1 to 4.2.5), 4.4, 4.5, 4.6, 4.7** **RBT: L1, L2, L3**	08
Module 3	
Memory System: Basic Concepts, Semiconductor RAM Memories, Read Only Memories, Speed, Size, and Cost, Cache Memories – Mapping Functions, Replacement Algorithms, Performance Considerations, Virtual Memories, Secondary Storage. **Text book 1: Chapter5 – 5.1 to 5.7, 5.9** **RBT: L1, L2, L3**	08
Module 4	
Arithmetic: Numbers, Arithmetic Operations and Characters, Addition and Subtraction of Signed Numbers, Design of Fast Adders, Multiplication of Positive Numbers, Signed Operand Multiplication, Fast Multiplication, Integer Division, Floating-point Numbers and Operations.	08

Text book 1: Chapter6 – 6.1 to 6.7 **RBT: L1, L2, L3**	
Module 5	
Basic Processing Unit: Some Fundamental Concepts, Execution of a Complete Instruction, Multiple Bus Organization, Hard-wired Control, Micro programmed Control. Pipelining, Embedded Systems and Large Computer Systems: Basic Concepts of pipelining, Examples of Embedded Systems, Processor chips for embedded applications, Simple Microcontroller. **Text book 1: Chapter7, Chapter8 – 8.1, Chapter9 – 9.1, 9.2, 9.3** **RBT: L1, L2, L3**	08
Course Outcomes: The student will be able to :	
• Explain the basic organization of a computer system. • Demonstrate functioning of different sub systems, such as processor, Input/output,and memory. • Illustrate hardwired control and micro programmed control, pipelining, embedded and other computing systems. • Design and analyse simple arithmetic and logical units.	
Question Paper Pattern:	
• The question paper will have ten questions. • Each full Question consisting of 20 marks • There will be 2 full questions (with a maximum of four sub questions) from each module. • Each full question will have sub questions covering all the topics under a module. • The students will have to answer 5 full questions, selecting one full question from each module.	
Textbooks:	
1. Carl Hamacher, Zvonko Vranesic, Safwat Zaky, Computer Organization, 5th Edition, Tata McGraw Hill, 2002. (Listed topics only from Chapters 1, 2, 4, 5, 6, 7, 8, 9 and12)	
Reference Books:	
1. William Stallings: Computer Organization & Architecture, 9th Edition, Pearson, 2015.	

SOFTWARE ENGINEERING
(Effective from the academic year 2018 -2019)
SEMESTER – III

Subject Code	18CS35	**CIE Marks**	40
Number of Contact Hours/Week	3:0:0	**SEE Marks**	60
Total Number of Contact Hours	40	**Exam Hours**	3 Hrs

CREDITS –3

Course Learning Objectives: This course (18CS35) will enable students to:

- Outline software engineering principles and activities involved in building large software programs.
- Identify ethical and professional issues and explain why they are of concern to software engineers.
- Describe the process of requirements gathering, requirements classification, requirements specification and requirements validation.
- Differentiate system models, use UML diagrams and apply design patterns.
- Discuss the distinctions between validation testing and defect testing.
- Recognize the importance of software maintenance and describe the intricacies involved in software evolution.
- Apply estimation techniques, schedule project activities and compute pricing.
- Identify software quality parameters and quantify software using measurements and metrics.
- List software quality standards and outline the practices involved.
- Recognize the need for agile software development, describe agile methods, apply agile practices and plan for agility.

Module 1	**Contact Hours**
Introduction: Software Crisis, Need for Software Engineering. Professional Software Development, Software Engineering Ethics. Case Studies. **Software Processes**: Models: Waterfall Model **(Sec 2.1.1)**, Incremental Model **(Sec 2.1.2)** and Spiral Model **(Sec 2.1.3)**. Process activities. **Requirements Engineering**: Requirements Engineering Processes **(Chap 4)**. Requirements Elicitation and Analysis **(Sec 4.5).** Functional and non-functional requirements **(Sec 4.1)**. The software Requirements Document **(Sec 4.2)**. Requirements Specification **(Sec 4.3)**. Requirements validation **(Sec 4.6)**. Requirements Management **(Sec 4.7)**. **RBT: L1, L2, L3**	08
Module 2	
System Models: Context models **(Sec 5.1)**. Interaction models **(Sec 5.2)**. Structural models **(Sec 5.3)**. Behavioral models **(Sec 5.4)**. Model-driven engineering **(Sec 5.5)**. **Design and Implementation**: Introduction to RUP **(Sec 2.4)**, Design Principles **(Chap 17)**. Object-oriented design using the UML **(Sec 7.1)**. Design patterns **(Sec 7.2)**. Implementation issues **(Sec 7.3)**. Open source development **(Sec 7.4)**. **RBT: L1, L2, L3**	08
Module 3	
Software Testing: Development testing **(Sec 8.1)**, Test-driven development **(Sec 8.2)**, Release testing **(Sec 8.3)**, User testing **(Sec 8.4)**. Test Automation **(Page no 42, 70,212, 231,444,695)**. **Software Evolution**: Evolution processes **(Sec 9.1)**. Program evolution dynamics **(Sec 9.2)**.	08

Software maintenance **(Sec 9.3)**. Legacy system management **(Sec 9.4)**. **RBT: L1, L2, L3**	
Module 4	
Project Planning: Software pricing **(Sec 23.1)**. Plan-driven development **(Sec 23.2)**. Project scheduling **(Sec 23.3)**: Estimation techniques **(Sec 23.5)**. **Quality management**: Software quality **(Sec 24.1)**. Reviews and inspections **(Sec 24.3)**. Software measurement and metrics **(Sec 24.4)**. Software standards **(Sec 24.2)** **RBT: L1, L2, L3**	08
Module 5	
Agile Software Development: Coping with Change **(Sec 2.3)**, The Agile Manifesto: Values and Principles. Agile methods: SCRUM (Ref "**The SCRUM Primer, Ver 2.0**") and Extreme Programming **(Sec 3.3)**. Plan-driven and agile development **(Sec 3.2)**. Agile project management **(Sec 3.4),** Scaling agile methods **(Sec 3.5)**. **RBT: L1, L2, L3**	08

Course Outcomes: The student will be able to :

- Design a software system, component, or process to meet desired needs within realistic constraints.
- Assess professional and ethical responsibility
- Function on multi-disciplinary teams
- Use the techniques, skills, and modern engineering tools necessary for engineering practice
- Analyze, design, implement, verify, validate, implement, apply, and maintain software systems or parts of software systems

Question Paper Pattern:

- The question paper will have ten questions.
- Each full Question consisting of 20 marks
- There will be 2 full questions (with a maximum of four sub questions) from each module.
- Each full question will have sub questions covering all the topics under a module.
- The students will have to answer 5 full questions, selecting one full question from each module.

Textbooks:

1. Ian Sommerville: Software Engineering, 9th Edition, Pearson Education, 2012. (Listed topics only from Chapters 1,2,3,4, 5, 7, 8, 9, 23, and 24)
2. The SCRUM Primer, Ver 2.0, http://www.goodagile.com/scrumprimer/scrumprimer20.pdf

Reference Books:

1. Roger S. Pressman: Software Engineering-A Practitioners approach, 7th Edition, Tata McGraw Hill.
2. Pankaj Jalote: An Integrated Approach to Software Engineering, Wiley India

Web Reference for eBooks on Agile:

1. http://agilemanifesto.org/
2. http://www.jamesshore.com/Agile-Book/

DISCRETE MATHEMATICAL STRUCTURES
(Effective from the academic year 2018 -2019)
SEMESTER – III

Subject Code	18CS36	**CIE Marks**	40
Number of Contact Hours/Week	3:0:0	**SEE Marks**	60
Total Number of Contact Hours	40	**Exam Hours**	3 Hrs

CREDITS –3

Course Learning Objectives: This course (18CS36) will enable students to:

- Provide theoretical foundations of computer science to perceive other courses in the programme.
- Illustrate applications of discrete structures: logic, relations, functions, set theory and counting.
- Describe different mathematical proof techniques,
- Illustrate the use of graph theory in computer science

Module 1	**Contact Hours**
Fundamentals of Logic: Basic Connectives and Truth Tables, Logic Equivalence – The Laws of Logic, Logical Implication – Rules of Inference. Fundamentals of Logic contd.: The Use of Quantifiers, Quantifiers, Definitions and the Proofs of Theorems. **Text book 1: Chapter2** **RBT: L1, L2, L3**	08
Module 2	
Properties of the Integers: The Well Ordering Principle – Mathematical Induction, Recursive Definitions, The division algorithm, The Greatest common divisor. **Fundamental Principles of Counting:** The Rules of Sum and Product, Permutations, Combinations – The Binomial Theorem, Combinations with Repetition. **Text book 1: Chapter4 – 4.1, 4.2, 4.3, 4.4, Chapter1** **RBT: L1, L2, L3**	08
Module 3	
Relations and Functions: Cartesian Products and Relations, Functions – Plain and One-to-One, Onto Functions. The Pigeon-hole Principle, Function Composition and Inverse Functions. **Relations:** Properties of Relations, Computer Recognition – Zero-One Matrices and Directed Graphs, Partial Orders – Hasse Diagrams, Equivalence Relations and Partitions. **Text book 1: Chapter5, Chapter7 – 7.1 to 7.4** **RBT: L1, L2, L3**	08
Module 4	
The Principle of Inclusion and Exclusion: The Principle of Inclusion and Exclusion, Generalizations of the Principle, Derangements – Nothing is in its Right Place, Rook Polynomials. **Recurrence Relations:** First Order Linear Recurrence Relation, The Second Order Linear Homogeneous Recurrence Relation with Constant Coefficients. **Text book 1: Chapter8 – 8.1 to 8.4, Chapter10 – 10.1, 10.2**	08

RBT: L1, L2, L3	
Module 5	
Introduction to Graph Theory: Definitions and Examples, Sub graphs, Complements, and Graph Isomorphism, Vertex Degree, Euler Trails and Circuits. **Trees**: Definitions, Properties, and Examples, Routed Trees, Trees and Sorting, Weighted Trees and Prefix Codes **Text book 1: Chapter11 – 11.1 to 11.3 Chapter12 – 12.1 to 12.4** **RBT: L1, L2, L3**	08
Course Outcomes: The student will be able to :	
• Use propositional and predicate logic in knowledge representation and truth verification. • Demonstrate the application of discrete structures in different fields of computer science. • Solve problems using recurrence relations and generating functions. • Application of different mathematical proofs techniques in proving theorems in the courses. • Compare graphs, trees and their applications.	
Question Paper Pattern:	
• The question paper will have ten questions. • Each full Question consisting of 20 marks • There will be 2 full questions (with a maximum of four sub questions) from each module. • Each full question will have sub questions covering all the topics under a module. • The students will have to answer 5 full questions, selecting one full question from each module.	
Textbooks:	
1. Ralph P. Grimaldi: Discrete and Combinatorial Mathematics, 5th Edition, Pearson Education. 2004.	
Reference Books:	
1. Basavaraj S Anami and Venakanna S Madalli: Discrete Mathematics – A Concept based approach, Universities Press, 2016 2. Kenneth H. Rosen: Discrete Mathematics and its Applications, 6th Edition, McGraw Hill, 2007. 3. Jayant Ganguly: A Treatise on Discrete Mathematical Structures, Sanguine-Pearson, 2010. 4. D.S. Malik and M.K. Sen: Discrete Mathematical Structures: Theory and Applications, Thomson, 2004. 5. Thomas Koshy: Discrete Mathematics with Applications, Elsevier, 2005, Reprint 2008.	

ANALOG AND DIGITAL ELECTRONICS LABORATORY
(Effective from the academic year 2018 -2019)
SEMESTER – III

Subject Code	18CPL37	**CIE Marks**	40
Number of Contact Hours/Week	0:2:2	**SEE Marks**	60
Total Number of Lab Contact Hours	36	**Exam Hours**	3 Hrs

Credits – 2

Course Learning Objectives: This course (18CSL37) will enable students to:

This laboratory course enable students to get practical experience in design, assembly and evaluation/testing of

- Analog components and circuits including Operational Amplifier, Timer, etc.
- Combinational logic circuits.
- Flip - Flops and their operations
- Counters and registers using flip-flops.
- Synchronous and Asynchronous sequential circuits.
- A/D and D/A converters

Descriptions (if any):

- Simulation packages preferred: Multisim, Modelsim, PSpice or any other relevant.
- For Part A (Analog Electronic Circuits) students must trace the wave form on Tracing sheet / Graph sheet and label trace.
- Continuous evaluation by the faculty must be carried by including performance of a student in both hardware implementation and simulation (if any) for the given circuit.
- A batch not exceeding 4 must be formed for conducting the experiment. For simulation individual student must execute the program.

Laboratory Programs:

PART A (Analog Electronic Circuits)

1.	Design an astable multivibrator ciruit for three cases of duty cycle (50%, <50% and >50%) using NE 555 timer IC. Simulate the same for any one duty cycle.
2.	Using appropriate linear IC regulators, design fixed +5V and -12V regulator circuits. For the rectification design a full wave bridge rectifier ciruit. And simulate the same.
3.	Using ua 741 Opamp, design a 1 kHz Relaxation Oscillator with 50% duty cycle. And simulate the same.
4.	Using ua 741 opamap, design a window comparate for any given UTP and LTP. And simulate the same.
5.	Demonstrate the use of LED and photodiode for an alarm system.

PART B (Digital Electronic Circuits)

6.	Design and implement Half adder, Full Adder, Half Subtractor, Full Subtractor using basic gates. And implement the same in HDL.
7.	Given a 4-variable logic expression, simplify it using appropriate technique and realize the simplified logic expression using 8:1 multiplexer IC. And implement the same in HDL.
8.	Realize a J-K Master / Slave Flip-Flop using NAND gates and verify its truth table. And implement the same in HDL.
9.	Design and implement a mod-n (n<8) synchronous up counter using J-K Flip-Flop ICs and demonstrate its working.
10.	Design and implement an asynchronous counter using decade counter IC to count up from 0 to n (n<=9) and demonstrate on 7-segment display (using IC-7447)

Laboratory Outcomes: The student should be able to:

- Use appropriate design equations / methods to design the given circuit.
- Examine and verify the design of both analog and digital circuits using simulators.
- Make us of electronic components, ICs, instruments and tools for design and testing of circuits for the given the appropriate inputs.
- Compile a laboratory journal which includes; aim, tool/instruments/software/components used, design equations used and designs, schematics, program listing, procedure followed, relevant theory, results as graphs and tables, interpreting and concluding the findings.

Conduct of Practical Examination:

- All laboratory experiments, excluding the first, are to be included for practical examination.
- Experiment distribution
  - For questions having only one part: Students are allowed to pick one experiment from the lot and are given equal opportunity.
  - For questions having part A and B: Students are allowed to pick one experiment from part A and one experiment from part B and are given equal opportunity.
- Change of experiment is allowed only once and marks allotted for procedure part to be made zero.
- Marks Distribution *(Subjected to change in accoradance with university regulations)*
  - a) For questions having only one part – Procedure + Execution + Viva-Voce: 15+70+15 = 100 Marks
  - b) For questions having part A and B
    - i. Part A – Procedure + Execution + Viva = 4 + 21 + 5 = 30 Marks
    - ii. Part B – Procedure + Execution + Viva = 10 + 49+ 11 = 70 Marks

DATA STRUCTURES LABORATORY
(Effective from the academic year 2018 -2019)
SEMESTER – III

Subject Code	18CPL38	**CIE Marks**	40
Number of Contact Hours/Week	0:2:2	**SEE Marks**	60
Total Number of Lab Contact Hours	36	**Exam Hours**	3 Hrs

Credits – 2

Course Learning Objectives: This course (18CSL38) will enable students to:

This laboratory course enable students to get practical experience in design, develop, implement, analyze and evaluation/testing of

- Asymptotic performance of algorithms.
- Linear data structures and their applications such as stacks, queues and lists
- Non-Linear data structures and their applications such as trees and graphs
- Sorting and searching algorithms

Descriptions (if any):

- Implement all the programs in ‘C / C++’ Programming Language and Linux / Windows as OS.

Programs List:

1.	Design, Develop and Implement a menu driven Program in C for the following array operations. a. Creating an array of N Integer Elements b. Display of array Elements with Suitable Headings c. Inserting an Element (ELEM) at a given valid Position (POS) d. Deleting an Element at a given valid Position (POS) e. Exit. Support the program with functions for each of the above operations.
2.	Design, Develop and Implement a Program in C for the following operations on Strings. a. Read a main String (STR), a Pattern String (PAT) and a Replace String (REP) b. Perform Pattern Matching Operation: Find and Replace all occurrences of PAT in STR with REP if PAT exists in STR. Report suitable messages in case PAT does not exist in STR Support the program with functions for each of the above operations. Don't use Built-in functions.
3.	Design, Develop and Implement a menu driven Program in C for the following operations on STACK of Integers (Array Implementation of Stack with maximum size MAX) a. Push an Element on to Stack b. Pop an Element from Stack c. Demonstrate how Stack can be used to check Palindrome d. Demonstrate Overflow and Underflow situations on Stack e. Display the status of Stack f. Exit Support the program with appropriate functions for each of the above operations
4.	Design, Develop and Implement a Program in C for converting an Infix Expression to Postfix Expression. Program should support for both parenthesized and free parenthesized expressions with the operators: +, -, *, /, % (Remainder), ^ (Power) and alphanumeric operands.

5.	Design, Develop and Implement a Program in C for the following Stack Applications a. Evaluation of Suffix expression with single digit operands and operators: +, -, *, /, %, ^ b. Solving Tower of Hanoi problem with n disks
6.	Design, Develop and Implement a menu driven Program in C for the following operations on Circular QUEUE of Characters (Array Implementation of Queue with maximum size MAX) a. Insert an Element on to Circular QUEUE b. Delete an Element from Circular QUEUE c. Demonstrate Overflow and Underflow situations on Circular QUEUE d. Display the status of Circular QUEUE e. Exit Support the program with appropriate functions for each of the above operations
7.	Design, Develop and Implement a menu driven Program in C for the following operations on Singly Linked List (SLL) of Student Data with the fields: *USN, Name, Branch, Sem, PhNo* a. Create a SLL of N Students Data by using *front insertion*. b. Display the status of SLL and count the number of nodes in it c. Perform Insertion / Deletion at End of SLL d. Perform Insertion / Deletion at Front of SLL(Demonstration of stack) e. Exit
8.	Design, Develop and Implement a menu driven Program in C for the following operations on Doubly Linked List (DLL) of Employee Data with the fields: *SSN, Name, Dept, Designation, Sal, PhNo* a. Create a DLL of N Employees Data by using *end insertion*. b. Display the status of DLL and count the number of nodes in it c. Perform Insertion and Deletion at End of DLL d. Perform Insertion and Deletion at Front of DLL e. Demonstrate how this DLL can be used as Double Ended Queue. f. Exit
9.	Design, Develop and Implement a Program in C for the following operationson Singly Circular Linked List (SCLL) with header nodes a. Represent and Evaluate a Polynomial $P(x,y,z) = 6x^2y^2z-4yz^5+3x^3yz+2xy^5z-2xyz^3$ b. Find the sum of two polynomials POLY1(x,y,z) and POLY2(x,y,z) and store the result in POLYSUM(x,y,z) Support the program with appropriate functions for each of the above operations
10.	Design, Develop and Implement a menu driven Program in C for the following operations on Binary Search Tree (BST) of Integers . a. Create a BST of N Integers: 6, 9, 5, 2, 8, 15, 24, 14, 7, 8, 5, 2 b. Traverse the BST in Inorder, Preorder and Post Order c. Search the BST for a given element (KEY) and report the appropriate message d. Exit
11.	Design, Develop and Implement a Program in C for the following operations on Graph(G) of Cities a. Create a Graph of N cities using Adjacency Matrix. b. Print all the nodes reachable from a given starting node in a digraph using DFS/BFS method

12.	Given a File of N employee records with a set K of Keys (4-digit) which uniquely determine the records in file F. Assume that file F is maintained in memory by a Hash Table (HT) of m memory locations with L as the set of memory addresses (2-digit) of locations in HT. Let the keys in K and addresses in L are Integers. Design and develop a Program in C that uses Hash function H: K →L as H(K)=K mod m (remainder method), and implement hashing technique to map a given key K to the address space L. Resolve the collision (if any) using linear probing.

Laboratory Outcomes: The student should be able to:

- Analyze and Compare various linear and non-linear data structures
- Code, debug and demonstrate the working nature of different types of data structures and their applications
- Implement, analyze and evaluate the searching and sorting algorithms
- Choose the appropriate data structure for solving real world problems

Conduct of Practical Examination:

- All laboratory experiments, excluding the first, are to be included for practical examination.
- Experiment distribution
  - o For questions having only one part: Students are allowed to pick one experiment from the lot and are given equal opportunity.
  - o For questions having part A and B: Students are allowed to pick one experiment from part A and one experiment from part B and are given equal opportunity.
- Change of experiment is allowed only once and marks allotted for procedure part to be made zero.
- Marks Distribution *(Subjected to change in accoradance with university regulations)*
  - c) For questions having only one part – Procedure + Execution + Viva-Voce: 15+70+15 = 100 Marks
  - d) For questions having part A and B
    - i. Part A – Procedure + Execution + Viva = 4 + 21 + 5 = 30 Marks
    - ii. Part B – Procedure + Execution + Viva = 10 + 49+ 11 = 70 Marks

DESIGN AND ANALYSIS OF ALGORITHMS
(Effective from the academic year 2018 -2019)
SEMESTER – IV

Subject Code	18CS42	**CIE Marks**	40
Number of Contact Hours/Week	3:2:0	**SEE Marks**	60
Total Number of Contact Hours	50	**Exam Hours**	3 Hrs

CREDITS –4

Course Learning Objectives: This course (18CS42) will enable students to:

- Explain various computational problem solving techniques.
- Apply appropriate method to solve a given problem.
- Describe various methods of algorithm analysis.

Module 1	**Contact Hours**
Introduction: What is an Algorithm? **(T2:1.1),** Algorithm Specification **(T2:1.2),** Analysis Framework **(T1:2.1), Performance Analysis**: Space complexity, Time complexity **(T2:1.3)**. **Asymptotic Notations:** Big-Oh notation (*O*), Omega notation (Ω), Theta notation (*Θ*), and Little-oh notation (*o*), Mathematical analysis of Non-Recursive and recursive Algorithms with Examples **(T1:2.2, 2.3, 2.4). Important Problem Types:** Sorting, Searching, String processing, Graph Problems, Combinatorial Problems. **Fundamental Data Structures:** Stacks, Queues, Graphs, Trees, Sets and Dictionaries. **(T1:1.3,1.4).** **RBT: L1, L2, L3**	10
Module 2	
Divide and Conquer: General method, Binary search, Recurrence equation for divide and conquer, Finding the maximum and minimum **(T2:3.1, 3.3, 3.4)**, Merge sort, Quick sort **(T1:4.1, 4.2)**, Strassen's matrix multiplication **(T2:3.8)**, Advantages and Disadvantages of divide and conquer. **Decrease and Conquer Approach:** Topological Sort. **(T1:5.3).** **RBT: L1, L2, L3**	10
Module 3	
Greedy Method: General method, Coin Change Problem, Knapsack Problem, Job sequencing with deadlines **(T2:4.1, 4.3, 4.5). Minimum cost spanning trees:** Prim's Algorithm, Kruskal's Algorithm **(T1:9.1, 9.2)**. **Single source shortest paths:** Dijkstra's Algorithm **(T1:9.3)**. **Optimal Tree problem:** Huffman Trees and Codes **(T1:9.4)**. **Transform and Conquer Approach:** Heaps and Heap Sort **(T1:6.4)**. **RBT: L1, L2, L3**	10
Module 4	
Dynamic Programming: General method with Examples, Multistage Graphs **(T2:5.1, 5.2)**. **Transitive Closure:** Warshall's Algorithm, **All Pairs Shortest Paths:** Floyd's Algorithm, Optimal Binary Search Trees, Knapsack problem **((T1:8.2, 8.3, 8.4)**, Bellman-Ford Algorithm **(T2:5.4)**, Travelling Sales Person problem **(T2:5.9)**, Reliability design **(T2:5.8)**. **RBT: L1, L2, L3**	10
Module 5	
Backtracking: General method **(T2:7.1)**, N-Queens problem **(T1:12.1)**, Sum of subsets problem **(T1:12.1)**, Graph coloring **(T2:7.4)**, Hamiltonian cycles **(T2:7.5)**. **Branch and Bound:** Assignment Problem, Travelling Sales Person problem **(T1:12.2)**, **0/1 Knapsack problem (T2:8.2, T1:12.2):** LC Branch and Bound solution **(T2:8.2)**, FIFO Branch and	10