## SEMESTER II B. TECH. COMPUTER ENGINEERING (COE)

| Course Code | Type | Subject | L | T | P | Credits | CA | MS | ES | CA | ES | Pre-requisites |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CECSC01 | CC | Discrete Structures | 3 | 1 | 0 | 4 | 25 | 25 | 50 | - | - | None |

**COURSE OUTCOMES**

1. To be able to analyze and compute time and space complexity of various computing problems.
2. To be able to design algorithms for solving various problems using the concepts of discrete mathematics.
3. To apply the concepts and algorithms learnt in developing large scale applications and modify them.
4. Get a grasp of the practical problems and their relation with discrete structures.
5. Implement practical problems using the discrete structures approach.

**COURSE CONTENT**

**UNIT-I**

**Logic**: Mathematical Logic, Propositions, Truth Tables, and Logical inferences, Methods of Proof, Propositional Logic, Logical Inference, First order logic, applications, Predicates and quantifiers.

**Set Theory, Relations and Functions:** Elements of Set Theory, Primitives of set theory, binary Relation and its Representation, type of Binary Relations, Equivalence relations and partitions. Functions, Types of functions, Inverses and composition of Functions.

**UNIT-II**

**Counting:** Counting and analysis of algorithms, Principles of inclusion-exclusion, Pigeon hole principle, Permutations, Combinations.

**Mathematical induction:** proof by induction, Recursion, Characteristic Polynomial, Recurrence relation, generating functions, Asymptotic behavior of algorithms.

**UNIT-III**

**Posets, Lattices and Group Theory**: Posets, Hasse Diagram, Lattices: Definition, Properties of lattices – Bounded, Complemented, Modular and Complete lattice, Boolean Algebra, Groups & rings.

**Number Theory:** Infinity and Natural numbers, Integers, Divisibility and Euclidean algorithm, Prime numbers, Congruence, Modular arithmetic, Euler $\phi$function.

**UNIT-IV**

**Graphs**: Graph isomorphism, Paths and Cycles, Graph coloring, Critical Path, Eulerian paths and circuits, Hamiltonian paths and circuits, Bipartite Graphs, Digraphs, Multigraphs.

**UNIT-V**

**Probability:** Overview of probability theory, Discrete distributions.

**SUGGESTED READINGS**


1. Keneth H. Rosen, “Discrete Mathematics and Its Applications”, TMH.
2. C.L. Liu, “Elements of Discrete Mathematics”, TMH.

3. Kolman, Busby & Ross, “Discrete Mathematical Structures”, PHI.
4. NarsinghDeo, “Graph Theory With Application to Engineering and Computer Science”,PHI.
5. Charles S. Grimmstead, J. Laurie Snell “Introduction to Probability”.
Kai Lai Chung, “A Course in probability theory”.
6. J.P.Tremblay & R. Manohar, “Discrete Mathematical Structure with Applications to Computer Science” Mc.Graw Hill.

| Course Code | Type | Subject | L | T | P | Credits | CA | MS | ES | CA | ES | Pre-requisites |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CECSC02 | CC | Data Structures | 3 | 0 | 2 | 4 | 15 | 15 | 40 | 15 | 15 | None |

**COURSE OUTCOMES**

1. Candidate will be able to choose the appropriate data structure for a specified problem and determine the same in different scenarios of real world problems.
2. Become familiar with writing recursive methods and reducing larger problems recursively in smaller problems with applications to practical problems.
3. Be able to understand the abstract properties of various data structures such as stacks, queues, lists, trees and graphs and apply the same to real life problems of sorting, searching, and traversals for skill enhancement in problem solving.
4. Be able to implement various data structures in more than one manner
5. Understand the advantages and disadvantages of the different implementations by using efficient representation of problems.

**COURSE CONTENT**

**UNIT-I**

**Introduction:** Basic Terminology: Elementary Data Organization, Data Structure Operations, Algorithms Complexity and Time-Space Trade off.

**Arrays:** Array Definition and Analysis, Representation of Linear Arrays in Memory, Traversing, Insertion And Deletion in Array, Single Dimensional Arrays, Two Dimensional Arrays, Bubble Sorting, Selection Sorting, Linear Search, Binary Search, Multidimensional Arrays, Function Associated with Arrays, Character String in C, Character String Operations, Arrays as parameters, Implementing One Dimensional Array.

**UNIT-II**

**Stacks and Queues:** Introduction to Operations Associated with Stacks Push & Pop, Array representation of stacks, Operation associated with stacks: Create, Add, Delete, Application of stacks recursion polish expression and their compilation conversion of infix expression to prefix and postfix expression, Tower of Hanoi problem, Representation of Queues, Operations of queues: Create, Add, Delete, Front, Empty, Priority Queues and Heaps, Dequeue.

**UNIT-III**

**Recursion:** Recursive thinking, Recursive Definition of Mathematical Formulae, Recursive Array Search, Recursive Data Structure, Problem Solving With Recursion, Back Tracking

**Linked Lists:**More operations on linked list, polynomial addition, Header nodes, doubly linked list, generalized list, circular linked lists.

**UNIT-IV**

**Trees:**Trees – mathematical properties, Binary Search Trees and their representation, expression evaluation, Complete Binary trees, Extended binary trees, Traversing binary trees, Searching, Insertion and Deletion in binary search trees, Complexity of searching algorithm, Path length, Huffman's algorithm, General trees, AVL trees, Threaded trees, B trees, Trie data structure

**UNIT-V**

**Sorting:** Insertion Sort, Quick sort, two-way Merge sort, Heap sort, sorting on different keys, External sorting.

**Graphs:** Sequential representation of graphs, Adjacency matrices, Search and Traversal of graphs: Depth first, breadth first, topological sort.

**Outline of Practical Work:**

- Programs based on sorting and searching, implementing stacks, queues , simple calculator using postfix expression, command line calculator changing infix to postfix, implementation of linked lists - a simple editor program, traversal of binary trees , binary search tree creation, insertion, deletion, traversal sorting. AVL tree creation and rotations, Traversal of graphs using BFS and DFS , implementation of topological sorting. Templates and Containers Survey of new data structures.

**Suggestive List of Experiments**

1. Write a program to find the mean and the median of the numbers stored in an array.
2. Write a program to insert one element in an array and delete an element from an array.
3. Write a program to search for a number in an array.
4. Write a program to sort an array.
5. Write a program to merge two sorted arrays.
6. Write a program to store the marks obtained by 10 students in 5 courses in a two-dimensional array.
7. Write a program to implement a linked list.
8. Write a program to insert a node in a linked list and delete a node from a linked list.
9. Write a program to print the elements of a linked list in reverse order without disturbing the linked list.
10. Write a program to reverse a linked list.
11. Write a program to add two polynomials using linked lists.
12. Write a program to implement a doubly-linked list.
13. Write a program to implement a stack using an array.
14. Write a program to implement a stack using a linked list.
15. Write a program to implement a queue using an array.
16. Write a program to implement a queue using a linked list.
17. Write a program to implement a circular queue using an array.
18. Write a program to implement a priority queue using a linked list.
19. Write a program to implement a double-ended queue using a linked list.
20. Write a program to construct a binary tree and display its preorder, inorder and

postorder traversals.
21. Write a program to construct a binary search tree.
22. Write a program to construct a graph.
23. Write a program to calculate the distance between two vertices in a graph.
24. Write a program to calculate the distances between every pairs of vertices in a graph.
25. Write a program to construct a minimal spanning tree of a graph.

**References and Text Books:**


1. Nell B Dale,"C++ data structures",ISBN-10: 1449646751, 5-th edition.
2. Freetextbooks.com. Algorithms and data structures. Available :http://www.freetechbooks.com/algorithms-and-data-structures-f11.html
3. Robert Lafore,"Data structures in Java".
4. Data Structures – Horowitz Sahani PHI
5. Data Structures – Lipshutz TMH


| Course Code | Type | Subject | L | T | P | Credits | CA | MS | ES | CA | ES | Pre-requisites |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CEECC03 | CC | **Digital Logic Design** | 3 | 0 | 2 | 15 | 15 | 40 | 15 | 15 | 15 | None |

**COURSE OUTCOMES**

1. To get familiarized with number systems, codes, logic gates and Boolean algebra
2. To understand fundamental concepts of VHDL modelling for basic digital circuits
3. To understand the basic characteristics of various logic families
4. To analyze and understand the design process associated with sequential circuits
5. To develop basic understanding of programmable logic devices

**COURSE CONTENT**

**UNIT-I**

**Introduction to Digital Systems, Number Systems and Codes**: Binary, octal and hexadecimal number systems, Number-Base Conversions, Complements of Numbers, Signed numbers, Fixed and floating point numbers, Binary Arithmetic, Binary Codes: BCD, Gray, Excess-3, ASCII, Error detection and correction codes - parity check codes and Hamming code.
**Logic gates, Boolean Algebra and logic minimization:** Basic logic operation, Logic gates and Truth tables, Positive and Negative Logic, Boolean Algebra: Basic postulates and fundamental theorems, SOP and POS forms, Min terms, Max terms, Canonical Form, Gate level Minimization: K-map and Quine-McCluskey tabular methods, NAND/NOR implementations

**UNIT-II**

**Design Concepts using Hardware Description Language:** VHDL Programming Structure, Model, Test Bench, Simulation Tool.
**Combinational Logic Modules, their applications and VHDL Modeling:** Decoders, encoders, multiplexers, demultiplexers, Parity circuits, Comparators, Code Converters, Arithmetic modules- adders, subtractors, BCD Adder, ALU and multipliers, Implementing boolean function with multiplexers / decoders

**UNIT-III**

**Introduction to different logic families:** Operational characteristics of BJT and MOSFET as switch, Structure and operations of TTL and CMOS gates, Electrical characteristics of logic gates – logic levels and noise margins, fan-out, propagation delay, transition time, power consumption and power-delay product etc.

**UNIT-IV**

**Sequential Logic systems and VHDL Modeling:** Basic sequential circuits- latches and flip-flops: RS-latch, SR-flip flop, D-latch, D flip-flop, JK flip-flop, T flip-flop, Setup-time, HOLD Time, Propagation delay, Timing hazards and races, Characteristic Equations.
**Sequential logic modules, their applications and VHDL Modeling:** shift register: Bidirectional, Universal and Ring Counter; counters: Ripple, Up/Down, Mod N, BCD Counters etc.

**UNIT-V**

**State machines:** Definition, Classification: Mealy, Moore; Analysis and design of state machines using D flip-flops and JK flip-flops etc.
**Memory:** Read-only memory, Read/Write memory - SRAM and DRAM, EPROM, EEPROM, USB Flash drive, Testing and testability of logic circuits, Programmable Logic Devices: PROM, PLA, PAL, Basics of CPLD, FPGA etc.

**Text Books:**

1. M. Morris Mano and Michael D. Ciletti,"Digital Design", 5th Edition, Pearson
2. Charles Roth and Larry Kinney, “Fundamentals of Logic Design,” Cengage Learning, 7th Edition.

**References:**

1. Stephen Brown and Zvonko Vranesic, "Fundamentals of Digital Logic with VHDL Design", 3rd Edition, McGraw-Hill
2. R.J. Tocci., N.S.Widmer, G.L. Moss, “Digital Systems, Principles and Applications”,11th Edition, Pearson Education
3. Mohammed Ferdjallah,"Introduction to Digital Systems: Modeling, Synthesis, and Simulation Using VHDL", Wiley.

## DIGITAL LOGIC DESIGN LAB

**LIST OF EXPERIMENTS**

(1) Verify the truth table of AND, OR, NOT, NAND, NOR, X-OR, X-NOR gates
(2) Implement all the above mentioned gates by using NAND gates and NOR gates only.
(3) Design and Implement Half-adder, Full-adder, Half-subtractor, Full-subtractor using logic gates.
(4) Design a 4 bit parallel adder and subtractor using IC. Further using the same IC implement BCD to excess-3 code convertor.
(5) Design a 4 bit magnitude comparator using IC. Also implement 2 bit magnitude comparator using gates only.
(6) Design and implement a full adder circuit using DECODER and gates. Also implement the same by using complimentary output decoder.
(7) Design the following Flip-flop using NAND/NOR gates
    (i) S-R FF
    (ii) D FF
    (iii) J-K FF
    (iv) T FF
(8) Design and implement a MOD 6 synchronous UP counter using T FF.
(9) Design a 2 bit UP/DOWN counter using J-K FF
(10) Implementation of full adder
(11) Implementation of 4X1 MUX
(12) Conversion of BCD to Excess-3 code
(13) Implement 3X8 decoder
(14) Implement 2 bit by 2 bit magnitude comparator