SEC–1 [A] | **SciLab – 1** | **BS301**

Theory 2 Hours/Week 2 credits

Unit – I

Introduction to Scilab – what is scilab, downloading & installing scilab, a quick taste of scilab.
The Scilab Environment – manipulating the command line, working directory, comments, variables in memory, recording sessions, the scilab menu bar, demos.
Scalars & Vectors – introduction, initializing vectors in scilab, mathematical operations on vectors, relational operations on vectors, logical operations on vectors, built-in logical functions.

Unit – II

Scalars & Vectors – elementary mathematical functions, mathematical functions on scalars, complex numbers, trigonometric functions, inverse trigonometric functions, hyperbolic functions.
Matrices – introduction, arithmetic operators for matrices, basic matrix processing.
Polynomials – introduction, creating polynomials, basic polynomial commands, finding roots of polynomial, polynomial arithmetic, miscellaneous polynomial handling.

Text

Er. Hema Ramachandran, Dr. Achuthsankar S. Nair, *Computer SCILAB–A Free Software to MATLAB*

References

Digite, *Introduction to Scilab*
Digite, *Optimization in Scilab*
Scilab Enterprises, *Scilab for Very Beginners*
Digite, *Introduction to Discrete Probabilities with Scilab*

Note: Student friendly video lecturers pertaining to this course are available at http://spoken-tutorial.org/
Teachers are advised to teach this courses in the computer lab itself, so that the interested students may derive some time to perform few programs their own.

SEC–1 [B] | **Boolean Algebra** | **BS301**

Theory 2 Hours/Week 2 credits

Unit – I

Introduction Number Systems and Conversion: Digital Systems and Switching Circuits, Number Systems and Conversion, Binary Arithmetic, Representation of Negative Numbers, Binary Codes.
Boolean Algebra: Basic Operations, Boolean Expressions and Truth Tables, Basic Theorems, Commutative, Associative, Distributive, and DeMorgan's Laws, Simplification Theorems, Multiplying Out and Factoring, Complementing Boolean Expressions.

Unit – II

Boolean Algebra: Multiplying Out and Factoring Expressions, Exclusive-OR and Equivalence Operations, The Consensus Theorem, Algebraic Simplification of Switching Expressions, Proving Validity of an Equation, Programmed Exercises.
Applications of Boolean Algebra Minterm and Maxterm Expansions: Conversion of English Sentences to Boolean Equations: Combinational Logic Design Using a Truth Table, Minterm and Maxterm Expansions, General Minterm and Maxterm Expansions, Examples of Truth Table Construction, Design of Binary Adders and Subtracters.

Text Charles H. Roth, Jr. and Larry L. Kinney, *Fundamentals of Logic Design (7e)*

References
M. Morris Mano, Michael D. Ciletti, *Digital Design (4e)*
A. Saha and N. Manna, *Digital Principles and Logic Design*
M. Rafiquzzaman, *Fundamentals of Digital Logic and Microcontrollers (6e)*
Elliott Mendelson, *Theory and Problems of Boolean Algebra and Switching Circuit*
M. Morris Mano, Charles R. Kime, Tom Martin, *Logic and Computer Design Fundamentals*

DSC–3C Data Structures BS306

Theory	4 Hours/Week	4 credits
Practical	2 Hours/Week	1 credit

Unit – I

Fundamental Concepts: Introduction to Data Structures, Types of Data Structures, Introduction to Algorithm, Pseudo-code, Flow Chart, Analysis of Algorithms.
Linear Data Structure Using Arrays: 1-D Arrays, 2-D Arrays, N-D Arrays, Memory Representation and Address Calculation of 1-D, 2-D, N-D Arrays, Concept of Ordered List, String Manipulation, Pros and Cons of Arrays.
Stacks: Concept, Primitive Operations, Abstract Data Type, Representation Stacks Using Arrays, Prefix, Infix, Postfix Notations for Arithmetic Expression, Applications of Stacks– Converting Infix Expression to Postfix Expression, Evaluating the Postfix Expression, Checking Well-formed (Nested) Parenthesis, Processing of Function Calls, Reversing a String.

Unit – II

Recursion: Introduction, Recurrence, Use of Stack in Recursion, Variants of Recursion, Execution of Recursive Calls, Recursive Functions, Iteration versus Recursion.
Queues: Concept, Primitive Operations, Abstract Data Type, Representation Queues Using Arrays, Circular Queue, Double-Ended Queue, Applications of Queues.
Linked Lists: Introduction, Concept, Terminology, Primitive Operations-creating, inserting, deleting, traversing, Representation of Linked Lists, Linked List Abstract Data Type, Linked List Variants - Singly Linked List, Doubly Linked List, Linear and Circular Linked List, Representation Stacks and Queues Using Linked Singly Lists, Application of Linked List–Garbage Collection.

Unit – III

Trees: Introduction, Representation of a General Tree, Binary Tree Introduction, Binary Tree Abstract Data Type, Implementation of Binary Trees, Binary Tree Traversals – Preorder, Inorder, Postorder Traversals, Applications of Binary Trees Briefly.
Graphs: Introduction, Graph Abstract Data Type, Representation of Graphs, Graph Traversal – Depth-First Search, Breadth-First Search, Spanning Tree – Prim's Algorithm, Kruskal's Algorithm.
Hashing: Introduction, Hash Functions, Collision Resolution Strategies.

Unit – IV

Searching and Sorting: Sequential (Linear) Search, Binary Search, Bubble Sort, Insertion Sort, Selection Sort, Quick Sort, Merge Sort, and Comparison of Sorting Techniques.
Heaps: Concept, Implementation, Abstract Data Type, Heap Sort.

Text

Varsha H. Patil, *Data Structures Using C++*

References

Nell Dale, *C++ Plus Data Structures*
Seymor Lipschutz, *Data Structures (Revised 1e)*
Adam Drozdek, *Data Structures and Algorithms in C++*
Mark Allen Weiss, *Data structures and Algorithm Analysis in C++ (4e)*
D.S. Malik, C*++ Programming: Program Design Including Data Structures (6e)*
Michael Main, Walter Savitch, *Data Structures and Other Objects Using C++ (4e)*
Michael T. Goodrich, R. Tamassia, David M. Mount, *Data Structures and Algorithms in C++*
Yonghui Wu, Jiande Wang, *Data Structure Practice for Collegiate Programming Contests and Education*

Data Structures Lab BS306

Practical 2 Hours/Week 1 credit

1. Write programs to implement the following using an array: a) Stack ADT b) Queue ADT.
2. Write a program to convert the given infix expression to postfix expression using stack.
3. Write a program to evaluate a postfix expression using stack.
4. Write a program to ensure the parentheses are nested correctly in an arithmetic expression.
5. Write a program to find following using Recursion
   a) Factorial of +ve Integer b) n^{th} term of the Fibonacci Sequence c) GCD of two +ve integers
6. Write a program to create a single linked list and write functions to implement the following operations.
   a) Insert an element at a specified position
   b) Delete a specified element in the list
   c) Search for an element and find its position in the list
   d) Sort the elements in the list ascending order
7. Write a program to create a double linked list and write functions to implement the following operations.
   a) Insert an element at a specified position
   b) Delete a specified element in the list
   c) Search for an element and find its position in the list
   d) Sort the elements in the list ascending order
8. Write a program to create singular circular linked lists and function to implement the following operations.
   a) Insert an element at a specified position
   b) Delete a specified element in the list
   c) Search for an element and find its position in the list
9. Write programs to implement the following using a single linked list:
   a) Stack ADT b) Queue ADT.
10. Write a program to implement Binary search technique using Iterative method and Recursive methods.
11. Write a program for sorting the given list numbers in ascending order using the following technique: Bubble sort and Selection sort
12. Write a program for sorting the given list numbers in ascending order using the following technique: Insertion sort and Quick sort
13. Write a program for sorting the given list numbers in ascending order using the following technique: Merge sort and Heap sort
14. Write a program to traverse a binary tree in following way.
    a) Pre-order b) In-order c) Post-order
15. Write a program to the implementation graph traversals – BFS and DFS.
16. Write a program to find the minimum spanning tree for a weighted graph using
    a) Prim's Algorithm b) Kruskal's Algorithm.

Note : Write the Pseudo Code for the above programs.

Recommended to use Open Source Software: GCC on Linux; DevC++ (or) CodeBlocks on Windows.