**This document is prepared from the following University Notifications**

- https://www.du.ac.in/uploads/new-web/15092023_Indis_sem1.pdf
- https://www.du.ac.in/uploads/new-web/notifications-2021/28032023_nep-Faculty%20of%20Interdisciplinary%20&%20Applied%20Sciences.pdf
- https://www.du.ac.in/uploads/new-web/15092023_Indis_sem3.pdf
- https://www.du.ac.in/uploads/new-web/18092023_Inter_4.pdf

## DISCIPLINE SPECIFIC ELECTIVES (DSE-2)

### CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

| Course title & Code | Credits | Credit distribution of the course | | | Eligibility criteria | Pre-requisite of the course (if any) |
|---|---|---|---|---|---|---|
| | | Lecture | Tutorial | Practical/ Practice | | |
| Algorithm Design and Analysis ELDSE-1B | 4 | 3 | 0 | 1 | Course Admission Eligibility | Basic Knowledge of Python language |

### Learning Objectives

The Learning Objectives of this course are as follows:

- To develop the understanding of usage of basic data structures like stack, queue, linked list, trees
- To introduce the students to design and analyse algorithms
- To highlight the differences between various problem-solving techniques for an efficient algorithm design
- To provide an understanding of algorithm design through a survey of the common algorithm design paradigms of Iterative techniques, Divide and Conquer, Dynamic Programming, Greedy Optimization
- To develop proficiency in Problem Solving and Programming
- To provide an understanding of time and space complexities of algorithms designed to solve computational problems
- To familiarize with various Searching and Sorting techniques

### Learning outcomes

The Learning Outcomes of this course are as follows:

- Implement data structures like Stacks, Queues. Linked List, trees
- Use an appropriate algorithm using the algorithm design techniques, namely,
- Iterative, Divide and Conquer, Greedy, Dynamic Programming for a series of
- computational problems
- Apply various Searching and Sorting techniques
- Solve computational problems with an understanding of time and space complexities of algorithms

### SYLLABUS OF ELDSE-1B Total Hours- Theory: 45 Hours, Practicals: 30 Hours

**UNIT – I ( 11 Hours)**

**Data Structures:** Stacks, array implementation of stack, operation on stacks, application of stacks-conversion of infix expression to prefix and postfix, evaluation of expression; Queues, array implementation of queues, operation on queues, Linked List and its implementation of stack and queue.

**UNIT – II (11 Hours)**

**Trees:** Introduction to trees, Binary search tree, preorder, postorder and inorder traversal (recursive)

**Searching Techniques:** Linear and Binary Search, Hashing techniques

**UNIT – III (12 Hours)**

**Algorithm Design Techniques:** Iterative techniques-Insertion Sort, Divide and Conquer-Merge Sort, Dynamic Programming-Weighted Interval Scheduling, 0-1 Knapsack Problem

**UNIT – IV (11 Hours)**

**Greedy Algorithm-** Interval Scheduling, Fractional Knapsack problem, Dijkstra's shortest path problem. Comparison between Dynamic programming and Greedy algorithm

**Sorting Techniques:** Quick Sort, Heap sort, Sorting in Linear Time - Bucket Sort, Radix Sort and Count Sort, Time and Space complexity

**Practical component (if any) – Algorithm Design and Analysis**
***(Python/MATLAB software)***

### Learning outcomes

The Learning Outcomes of this course are as follows:

- Implement Data Structures
- Develop algorithms and write programs in Python language
- Write programs based on Algorithm design techniques
- Implement various Sorting techniques
- Prepare a Technical Report on the experiments carried

### LIST OF PRACTICALS ( Total Practical Hours – 30 Hours)

1. Program to create a stack and perform Pop, Push, traverse operations on the stack using Linear Linked List
2. Program to create a linear queue using Linked List and implement insertion, deletion and display of the queue elements
3. Program to create a Binary Tree to perform traversals (Preorder, Postorder, Inorder) using the concept of recursion.
4. Program to solve the Interval Scheduling problem
5. Program to solve the Weighted Interval Scheduling problem
6. Program to solve the 0-1 Knapsack problem
7. Program to implement Insertion Sort
8. Program to implement Merge Sort
9. Program to implement Heap Sort
10. Program to implement Quick Sort

11. Program to implement Bucket Sort
12. Program to implement Radix Sort
13. Program to implement Binary Search

Note: Students shall sincerely work towards completing all the above listed practicals for this course. In any circumstance, the completed number of practicals shall not be less than eleven.

**Essential/recommended readings**

1. M.T.Goodrich, R.Tamassia, M.H.Goldwasser, Data Structures & Algorithms, Wiley
2. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein, Introduction to Algorithms, Prentice Hall India. Third edition (2015).
3. J. Kleinberg and E. Tardos, Algorithm Design, Pearson Education India, First Edition (2013).
4. S. Lipschutz, Data Structures with C, Schaum's Outlines Series, Tata McGraw Hill
5. A.M.Tenenbaum, Y.Langsam, M.J. Augenstein, Data Structures using C, Pearson/PHI

**Suggestive readings**

1. Sarabasse and A.V. Gleder, Computer Algorithm-Introduction to Design and Analysis, Pearson Education, Third Edition (1999).

**Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.**