COURSEPACK (Winter 2023-24)

SCHEME

The scheme is an overview of work-integrated learning opportunities and gets students out into the real world. This will show what a course entails.

Course Title	**Advanced Algorithmic Problem Solving**			**Course Type**				**Blended**			
Course Code	**R1UC601B**			**Class**				**B. Tech. (CSE) VI Sem.**			
Instruction delivery	**Activity**	**Credits**	**Weekly Hours**	**Total Number of Classes per Semester**				**Assessment in Weightage**			
	Lecture	**4**	**4**	**Theory**	**Tutorial**	**Practical**	**Practical**	**Self-study**	**CIE**	**SEE**	
	Tutorial	**0**	**0**								
	Practical	**2**	**4**								
	Self-study	**0**	**0**								
	Total	**4**	**8**	**44**	**0**	**44**			**50%**	**50%**	
Course Lead	**Dr. Anupam Kumar Sharma**		**Course Coordinator**	**Dr. Subhash Gupta**							

S.No.	Names Course Instructors Theory	Names Course Instructors Practical
1	Adjunct SCSE (Four faculty members from GFG)	Four faculty members from GFG
2	Vipul Narayan	P Selvaraj
3	Pooja Singh	Manish Verma
4	Neeraj Kumar Bharti	Saurabh Raj Sangwan
5	Gopal Chandra Jana	Manish Verma
6	Shweta	Uppiliraja P
7	Subhash Chandra Gupta	Saurabh Raj Sangwan
8	Vinod Kumar (GUSCSE202332309)	Pragya
9	N. Partheeban	Uppiliraja P
10	Vikas Kumar (GUSCSE202332341)	Rahul Kumar
11	Ashwin Perti	Kalaikovan M
12	Vinod Kumar (GUSCSE202332309)	Pragya
13	Manoj Kumar Tyagi	Pragya

14	Manoj Kumar Tyagi	Rahul Kumar
15	Akhilesh Kumar Tripathi	Pragya
16	Vikas Kumar (GUSCSE202332341)	Ragini Kumari
17	Ashwin Perti	Kalaikovan M
18	Mohammad Faiz	Ragini Kumari
19	Gopal Chandra Jana	P Selvaraj
20	Raghvendra Ajay Mishra	Rahul Kumar
21	Akhilesh Kumar Tripathi	Ragini Kumari
22	Subhash Chandra Gupta	Rahul Kumar
23	Raghvendra Ajay Mishra	Ravi Sharma (231893)
24	Shweta	Uppiliraja P
25	Anupam Kumar Sharma	Kalaikovan M
26	Pooja Singh	Manish Verma
27	Mohammad Faiz	Saurabh Raj Sangwan
28	Vipul Narayan	P Selvaraj
29	Neeraj Kumar Bharti	Raj Kumar Parida

COURSE OVERVIEW

An advanced course in data structures and algorithms builds upon the foundational knowledge of basic data structures and algorithms. It delves deeper into more complex data structures and advanced algorithmic techniques. This course explores advanced data structures and algorithmic techniques used in computer science and software development. It focuses on the design, analysis, and implementation of data structures and algorithms for solving complex computational problems efficiently.

PREREQUISITE COURSE

PREREQUISITE COURSE REQUIRED	YES	
If, yes please fill in the Details.	**Prerequisite course code**	**Prerequisite course name**
	E2UC503C	**Advanced Data Structure and algorithms.**

COURSE OBJECTIVE

The comprehensive course will be able to Understand Fundamental Concepts, master algorithm analysis, develop problem-solving skills, explore advanced data structures, implement, and analyse sorting and searching algorithms, understand hashing, learn about graphs and trees, apply algorithmic techniques.

COURSE OUTCOMES(COs)

After the completion of the course, the student will be able to:

CO No.	Course Outcomes
R1UC601B.1	understanding and practical ability in the use and implementation of a wide range of data structures (arrays, linked lists, stacks, queues, trees, graphs) and algorithms (sorting, searching, recursion, backtracking), enabling them to analyze and solve complex computational problems efficiently.
R1UC601B.2	analyze the time and space complexity of algorithms using asymptotic notations, and apply algorithm design techniques such as dynamic programming, greedy algorithms, divide and conquer, and backtracking to create optimized solutions for a variety of problems.
R1UC601B.3	acquire proficiency in advanced computing concepts, including bit manipulation, hashing, advanced string algorithms, and mathematical foundations for algorithms, enhancing their ability to tackle optimization problems and implement effective solutions.
R1UC601B.4	develop the skills to analyze the time and space complexity of algorithms using asymptotic notations, and apply algorithm design techniques such as dynamic programming, greedy algorithms, divide and conquer, and backtracking to create optimized solutions for a variety of problems.

BLOOM'S LEVEL OF THE COURSE OUTCOMES

Bloom's taxonomy is a set of hierarchical models used for the classification of educational learning objectives into levels of complexity and specificity. The learning domains are cognitive, affective, and psychomotor.

COMPREHENSIVE

CO No.	Remember **KL1**	Understand **KL 2**	Apply **KL 3**	Analyse **KL 4**	Evaluate **KL 2**	Create **KL 6**
R1UC601B.1	√	√				
R1UC601B.2			√	√		
R1UC601B.3			√	√	√	
R1UC601B.4			√	√	√	

PROGRAM OUTCOMES (POs): AS DEFINED BY CONCERNED THE APEX BODIES

PO1 Computing Science knowledge: Apply the knowledge of mathematics, statistics, computing science, and information science fundamentals to the solution of complex computer application problems.

PO2 Problem analysis: Identify, formulate, review research literature, and analyze complex computing science problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and computer sciences.

PO3 Design/development of solutions: Design solutions for complex computing 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 computing science and IT tools including prediction and modeling to complex computing activities with an understanding of the limitations.

PO6 IT specialist 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 computing science and information science practice.

PO7 Environment and sustainability: Understand the impact of professional computing science 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 computing science practice.

PO9 Individual and teamwork: 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 IT analyst 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 computing science 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.

PROGRAMME SPECIFIC OUTCOME (PSO):

The students of Computer Science and Engineering shall:

PSO1: Have the ability to work with emerging technologies in computing requisite to Industry 4.0.

PSO2: Demonstrate Engineering Practice learned through industry internship and research project tosolve live problems in various domains.

COURSE ARTICULATIONMATRIX

The Course articulation matrix indicates the correlation between Course Outcomes and Program Outcomesand theirexpected strengthofmappinginthreelevels(low,medium,and high).

COs#/ POs	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
R1UC601B.1	2				2									
R1UC601B.2	2	2			2									
R1UC601B.3	2	2	1		2									
R1UC601B.4	2	2	2		2						1	1	1	

Note: 1-Low, 2-Medium, 3-High

COURSE ASSESSMENT

The course assessment patterns are the assessment tools used both in formative and summative examinations.

Type of course	CIE		Total Marks		Final Marks CIE*0.5+SEE*0.5
	Lab @ (Work + Record)	MTE^	CIE	SEE	
Blended	50	50	100	100	100

@Lab Work-25 marks from Quizzes, and Lab performance and Lab Record-25 marks

^ MTE 25 marks from weekly contest and 25 marks from MTE exams.

COURSE CONTENT

Theory
Introduction and Time & Space Complexity-Asymptotic Notations-Analysis of Algorithm Efficiency-Arrays: Basic operations on arrays, Problems on Prefix and Suffix, sliding window, 2 pointers, Kadane's algorithm, Arrays and Digit Manipulation-String Algorithms-Advanced string searching algorithms- Bit Manipulation. LINKED LIST, STACK, RECURSION AND BACKTRACKING: Linked List: Singly Liked List, Doubly Linked List, Circular Linked List, Applications- Stack: Representation, Basic operations, Implementation, Applications. Recursion and backtracking: Analysis of Backtracking Algorithm, Types, Problems on n-Queens, subset sum problem, Hamiltonian Cycle-State-Space Tree- Applications. SORTING AND SEARCHING: Searching: Problems related to Linear Search, Binary Search, Problems using Binary Search Sorting: Types, Selection Sort, Bubble Sort, Insertion Sort, Heap Sort, Quick Sort, Merge Sort, Time Complexity. Hashing Universal hashing, optimization problems based on searching and sorting. Queues: Representation- Basic Operations, Applications, Deque, priority queue. TREE AND GRAPHS: Binary Tree: Implementation, Properties, Types, Operations- Binary Search Tree- Segment Trees: Lazy Propagation-Binary Indexed Tree-Binomial Heap-Operations on binomial heaps-Fibonacci Heap-Fibonacci heap operations-Disjoint set union data structures and its operations Graphs: Basic terminologies and representation, DFS, BFS, Topological sort, problems based on Shortest path algorithms, Minimum spanning tree, Articulation points and bridges, Applications of graph. ALGORITHM TECHNIQUES: Dynamic programming: Basics, Hill climbing- Problems based on Dynamic Programming- Strasson's Matrix multiplication, shortest uncommon subsequence, Greedy Technique: Activity selection problem, task scheduling problem Divide and Conquer: Counting Inversions, Closest pair of points, maximum subarray problem-Hiring problem, randomised algorithms-Maths Algorithms-Game Theory-Trie-Geometric Algorithms.

Lab (To be implemented in C/C++/Java/Python)

S. No.	**Write a program to implement:**
1	Missing number in array, Rotate Bits, Power of 2, Bit-Difference
2	Set kth bit, Toggle bits given range, Find first set bit, Check whether K-th bit is set, Rightmost different bit
3.	Detecting Duplicates in Array, Finding Unique element in Array, Finding Maximum XOR Subarray, Longest Consecutive 1's (Hamming Weight)
4	Swap all odd and even bits/ Detecting Parity Bits, Generate all subsets of a given set using bitwise operations.
5	Find the Kth largest elements in an array, Move all zeroes to end of array, Rearrange array such that even positioned are greater than odd.
6	Rearrange an array in maximum minimum form using Two Pointer Technique, Segregate even and odd numbers, Print All Distinct Elements of a given integer array.
7	Find sub-array with given sum, Reorder an array according to given indexes, Search an element in a sorted and rotated array.
8	Find the Rotation Count in Rotated Sorted array, Find the smallest missing number, Merge two sorted arrays with O(1) extra space.
9	Mean of Array using Recursion, Sum of natural numbers using recursion, Decimal to binary number using recursion.
10	Program for length of a string using recursion, Sum of digit of a number using recursion, Tail recursion to calculate sum of array elements. Program to print first n Fibonacci Numbers
11	Sort the Queue using Recursion, Reversing a queue using recursion, Delete a linked list using recursion
12	Length of longest palindromic sub-string: Recursion, Program for Tower of Hanoi Algorithm Find geometric sum of the series using recursion
13	Problems related to Linear Search, Binary Search, Problems using Binary Search
14	Convert a String to an Integer using Recursion
15	Reverse a Doubly linked list using recursion, Check if a string is a scrambled form of another string, N Queen Problem
16	How to Sort a Stack using Recursion
17	Sort elements by frequency, Sort an array of 0s, 1s and 2s, Sort numbers stored on different machines
18	Sorting Strings using Bubble Sort, Sort even-placed elements in increasing and odd-placed in decreasing order,
19	Print array of strings in sorted order without copying one string into another. K-Way Merge Sort
20	Nuts and bolts Problems.
21	Order Statistics over array elements. Space optimization using bit manipulations
22	Sort an array when two halves are sorted Sort a nearly sorted (or K sorted) array
23	Implementation of Prims algorithm.
24	Implementation of Kruskal algorithms

25	Construct tree from preorder traversal, Minimum distance between two given nodes , Maximum sum leaf to root path
26	Odd Even Level Difference, Lowest Common Ancestor of a Binary Tree , Ancestors in Binary Tree
27	Remove BST keys outside the given range Pair with given target in BST. Clone binary tree with random pointer, Maximum sum of non-adjacent nodes
28	Largest BST in a Binary Tree, Extreme nodes in alternate order Connect nodes at same level
29	Nodes at given distance in a Binary Tree, Sorted Linked List to BST , Binary Tree to Doubly Linked List
30	Find length of the largest region in Boolean Matrix, Count number of trees in a forest
31	Clone an Undirected Graph, Graph Coloring (Introduction and Applications)
32	Vertex Cover Problem
33	Activity Selection Problem, Job Sequencing Problem
34	Huffman Coding, Huffman Decoding
35	Minimum product subset of an array, Maximize array sum after K negations using Sorting
36	Fibonacci numbers, Coin change problem, Subset Sum Problem
37	Program for Bridge and Torch problem
38	Matrix Chain Multiplication
39	Maximum profit by buying and selling a share at most k times
40	Minimum cost to sort strings using reversal operations of different costs
41	Cutting a Rod, Longest Common Subsequence
42	Minimum number of jumps to reach end, Floyd Warshall Algorithm

LESSON PLAN FOR COMPREHENSIVE COURSES

FOR THEORY 15 weeks * 4 Hours = 60 Classes) (1credit = 1Lecture Hour)
FOR PRACTICAL 15 weeks * 4Hours = 60 Hours lab sessions (1 credit = 2 lab hours)

SL-No	Topic for Delivery	Lecture/ Practical Plan	Skill	Competency
1	Introduction and Time & Space Complexity-Asymptotic Nota-tions-Analysis of Algorithm Ef-ficiency	Lecture	write program implement bitwise operation, sliding window, and two pointers	CO1 and CO2
2	Arrays: Basic operations on arrays,	Lecture		
3	Missing number in array, Rotate Bits, Power of 2, Bit-Difference	Practical		
4	Set kth bit, Toggle bits given range, Find first set bit Check whether K-th bit is set Rightmost different bit	Practical		

5	Problems on Prefix and Suffix, sliding window, 2 pointers,	Lecture		
6	Kadane's algorithm, Arrays and Digit Manipulation-String Algo-rithms-	Lecture	Implement Kadanes algorithm, able to implement various operations on arrays. Implement Advanced string algorithms	
7	Detecting Duplicates in Array Finding Unique element in Ar-ray Finding Maximum XOR Subarray Longest Consecutive 1's (Hamming Weight)	Practical		
8	Swap all odd and even bits/ Detecting Parity Bits Generate all subsets of a given set using bitwise operations	Practical		
9	Advanced string searching al-gorithms- Rabin Karp and KMP.	Lecture		
10	Bit Manipulation.	Lecture		
11	Find the Kth largest elements in an array Move all zeroes to end of array Rearrange array such that even positioned are greater than odd	Practical		
12	Rearrange an array in maximum minimum form using Two Pointer Technique Segregate even and odd numbers Print All Distinct Elements of a given integer array	Practical		
13	Linked List: Singly Liked List, Doubly Linked List, Circular Linked List	Lecture	Implement linked lists, and merge in O(1) space.	CO1, CO2, and CO3.
14	Linked List Applications	Lecture		
15	Find sub-array with given sum Reorder an array according to given indexes Search an element in a sorted	Practical		

	and rotated array		Implement stack, use concept of recursion. Implement backtracking.	
16	Find the Rotation Count in Rotated Sorted array Find the smallest missing number Merge two sorted arrays with O(1) extra space	Practical		
17	Stack: Representation, Basic operations, Implementation	Lecture		
18	Stack: Applications	Lecture		
19	Mean of Array using Recursion Sum of natural numbers using recursion Decimal to binary number using recursion	Practical		
20	Program for length of a string using recursion Sum of digit of a number using recursion Tail recursion to calculate sum of array elements. Program to print first n Fibonacci Numbers	Practical		
21	Recursion and backtracking: Analysis of Backtracking Algorithm	Lecture		
22	Recursion Types, Problems on n-Queens,	Lecture		
23	Sort the Queue using Recursion Reversing a queue using recursion Delete a linked list using recursion	Practical		
24	Length of longest palindromic sub-string : Recursion Program for Tower of Hanoi Algorithm Find geometric sum of the series using recursion	Practical		
25	subset sum problem, Hamiltonian Cycle	Lecture		
26	State-Space Tree- Applications.	Lecture		

27	Problems related to Linear Search, Binary Search, Problems using Binary Search	Practical	Implement problems based on sorting and searching	
28	Convert a String to an Integer using Recursion	Practical		
29	Problems related to Linear Search, Binary Search, Problems using Binary Search	Lecture		
30	Selection Sort, Bubble Sort, Insertion Sort	Lecture		
31	Reverse a Doubly linked list using recursion Check if a string is a scrambled form of another string N Queen Problem	Practical		
32	How to Sort a Stack using Recursion	Practical		CO1, CO2, and CO3.
33	Heap Sort, Quick Sort, Merge Sort, Time Complexity	Lecture		
34	Hashing Universal hashing, optimization	Lecture		
35	Sort elements by frequency Sort an array of 0s, 1s and 2s Sort numbers stored on different machines	Practical		
36	Sorting Strings using Bubble Sort Sort even-placed elements in increasing and odd-placed in decreasing order	Practical		
37	Queues: Representation- Basic Operations, Applications, Deque, priority queue	Lecture	Implement Queue data structure, and solve problems based on queue data structure.	
38	Binary Tree: Implementation, Properties, Types, Operations- Binary Search Tree-	Lecture		
39	Print array of strings in sorted order without copying one string into another. K-Way Merge Sort	Practical		
40	Nuts and bolts Problems	Practical	Implement disjoint data structure. Graphs traversals and	
41	Disjoint set union data structures and its operations	Lecture		
42	DFS, BFS, Topological sort,	Lecture		

	problems based on shortest path algorithms,		problems based on graph traversal.	
43	Order Statistics over array elements. Space optimization using bit manipulations	Practical		
44	Sort an array when two halves are sorted Sort a nearly sorted (or K sorted) array	Practical		
45	binomial heaps-Fibonacci Heap-Fibonacci heap operations	Lecture	Implement binomial and Fibonacci heap, and problems based on heaps.	
46	Minimum spanning tree, Articulation points and bridges, Applications of graph.	Lecture		
47	Implementation of Prims algorithms	Practical	Implement MST, and problems based on MST.	
48	Implementation of Kruskal algorithms	Practical		
49	Minimum spanning tree, Articulation points and bridges, Applications of graph.	Lecture		
50	Basics, Hill climbing	Lecture		
51	Construct tree from preorder traversal. Minimum distance between two given nodes Maximum sum leaf to root path	Practical		
52	Odd Even Level Difference Lowest Common Ancestor of a Binary Tree Ancestors in Binary Tree	Practical		
53	Problems based on Dynamic Programming- Strasson's Matrix multiplication	Lecture	Implement problems based on dynamic programming.	
54	Problems based on Dynamic Programming	Lecture		
55	Remove BST keys outside the given range Pair with given target in BST. Clone binary tree with random pointer	Practical		

	Maximum sum of non-adjacent nodes			
56	Largest BST in a Binary Tree Extreme nodes in alternate order Connect nodes at same level	Practical		
57	shortest uncommon subsequence	Lecture	Implement problems based on dynamic programming, greedy approach.	CO2, CO3, and CO4
58	Activity selection problem	Lecture		
59	Nodes at given distance in a Binary Tree , Sorted Linked List to BST Binary Tree to Doubly Linked List	Practical		
60	Find length of the largest region in Boolean Matrix Count number of trees in a forest	Practical		
61	task scheduling problem	Lecture		
62	Counting Inversions	Lecture		
63	Clone an Undirected Graph Graph Coloring (Introduction and Applications)	Practical		
64	Vertex Cover Problem	Practical		
65	Closest pair of points	Lecture		
66	maximum subarray problem	Lecture		
67	Activity Selection Problem Job Sequencing Problem	Practical		
68	Huffman Coding Huffman Decoding	Practical		
69	Hiring problem	Lecture		
70	randomised algorithms	Lecture		
71	Minimum product subset of an array Maximize array sum after K negations using Sorting	Practical		
72	Fibonacci numbers Coin change problem Subset Sum Problem	Practical		
73	Maths Algorithms	Lecture		
74	Game theory	Lecture		

75	Program for Bridge and Torch problem	Practical		
76	Matrix Chain Multiplication	Practical		
77	Trie data structure	Lecture		
78	Segment Trees: Lazy Propagation	Lecture		
79	Maximum profit by buying and selling a share at most k times	Practical		
80	Minimum cost to sort strings using reversal operations of different costs	Practical		
81	Binary Indexed Tree-Binomial Heap-Operations on binomial heaps	Lecture		
82	Fibonacci Heap-Fibonacci heap operations	Lecture		
83	Cutting a Rod Longest Common Subsequence	Practical		
84	Minimum number of jumps to reach end Floyd Warshall Algorithm	Practical		
85	Revision	Lecture		
86	Revision	Lecture		
87	Revision	Practical		
88	Revision	Practical		

BIB107LIOGRAPHY

- **TextBook** Corman, Introduction to Algorithms
- **ReferenceBooks**
  - Data Structures and Algorithms in Java, Roberto Tamassia and Michael T Goodrich, John Wiley
  - R. Kruse etal, "Data Structures and Program Design in C", Pearson Education
- **Webliography**
  - https://www.geeksforgeeks.org/advanced-data-structures/
  - https://github.com/topics/advanced-data-structures
- **SWAYAM/NPTEL/MOOCs Certification**
  - https://www.coursera.org/specializations/data-structures-algorithms.

- https://www.codespaces.com/best-data-structures-and-algorithms-courses-classes.html#3-data-structures-and-algorithms-nanodegree-certification-udacity
-

PRACTICE PROBLEMS

SNo	Problem	KL
1	Write a program in Java or Python and find time complexity for 2–Sum Problem: Finding two numbers in an array that add up to a given target value.	K3
2	Write a program in Java or Python and find time complexity for Longest Common Subsequence Problem: Finding longest subsequence which is common in all given input sequences.	K3
3	Write a program in Java or Python and find time complexity for Maximum Subarray Problem: Finding a contiguous subarray with the largest sum, within a given one-dimensional array A[1...n] of numbers.	K3
4	Write a program in Java or Python and find time complexity for Coin Change Problem: Finding the number of ways to make sum by using different denominations from an integer array of coins[ ] of size N representing different types of denominations and an integer sum.	K3
5	Write a program in Java or Python and find time complexity for 0–1 Knapsack Problem: Restrict the number of copies of each kind of item to zero or one.	K3
6	Write a program in Java or Python and find time complexity for Subset Sum Problem: Checking if there is a subset of the given set whose sum is equal to the given sum.	K3
7	Write a program in Java or Python and find time complexity for Longest Palindromic Subsequence Problem: Finding a maximum-length subsequence of a given string that is also a Palindrome.	K3
8	Write a program in Java or Python and find time complexity for Matrix Chain Multiplication Problem: Finding the most efficient way to multiply these matrices together such that the total number of element multiplications is minimum.	K3
9	Write a program in Java or Python and find time complexity for Longest Common Substring Problem: A set of strings can be found by building a generalized suffix tree for the strings, and then finding the deepest internal nodes which have leaf nodes from all the strings in the subtree below it.	K3
10	Write a program in Java or Python and find time complexity for Rod Cutting Problem: A rod is given of length n. Another table is also provided, which contains different size and price for each size. Determine the maximum price by cutting the rod and selling them in the market.	K3
11	Write a program in Java or Python and find time complexity for Word Break Problem: You will be given a string, say "s", and a dictionary of strings say "wordDict". You have to return true if s can be segmented into a space-separated sequence of one or more dictionary words.	K3
12	Write a program in Java or Python and find time complexity for Edit Distance	K3

	Problem: Quantifying how dissimilar two strings (e.g., words) are to one another, that is measured by counting the minimum number of operations required to transform one string into the other.	
13	Write a program in Java or Python and find time complexity for Chess Knight Problem: A knight starting at any square of the board and moving to the remaining 63 squares without ever jumping to the same square more than once.	K3
14	Write a program in Java or Python and find time complexity for Partition Problem: A given set can be partitioned in such a way, that sum of each subset is equal.	K3
15	Write a program in Java or Python and find time complexity for 3–Partition Problem: Deciding whether a given multiset of integers can be partitioned into triplets that all have the same sum.	K3
16	Write a program in Java or Python and find time complexity for Snake and Ladder Problem: Write a function that returns the minimum number of jumps to take top or destination position. You can assume the dice you throw results in always favor of you means you can control the dice.	K3
17	Write a program in Java or Python and find time complexity for Largest Consecutive Subarray Problem: Finding out the largest sum of the consecutive numbers of the array.	K3
18	Write a program in Java or Python and find time complexity for Dutch National Flag Problem: The flag of the Netherlands consists of three colors: white, red, and blue. The task is to randomly arrange balls of white, red, and blue such that balls of the same color are placed together.	K3
19	Write a program in Java or Python and find time complexity for Knight's Tour Problem: a puzzle where a chess knight is placed on an empty chess board and the goal is to move the knight to every square on the board exactly once without re-visiting any squares.	K3
20	Write a program in Java or Python and find time complexity for Maximum Sum Submatrix Problem: A 2D array arr[][] of dimension N*M is given, the task is to find the maximum sum sub-matrix from the matrix arr[][].	K3
21	Write a program in Java or Python and find time complexity for Longest Palindromic Substring Problem: Finding a maximum-length contiguous substring of a given string that is also a palindrome.	K3
22	Write a program in Java or Python and find time complexity for Job Sequencing Problem: You have a single processor operating system and a set of jobs that have to be completed with given deadline constraints.	K3
23	Write a program in Java or Python and find time complexity for N–Queens Problem: Placing N chess queens on an N×N chessboard so that no two queens attack each other.	K3
24	Write a program in Java or Python and find time complexity for Maximum Product Subarray Problem: Find the contiguous subarray within the array which has the largest product of its elements. You have to report this maximum product.	K3
25	Write a program in Java or Python and find time complexity for Longest Repeated Subsequence Problem: Find the length of the longest repeating subsequence in a given string such that the two subsequences don't have the same original string character at the same position.	K3

26	Write a program in Java or Python and find time complexity for 3–Sum Problem: Given an array and a value, find if there is a triplet in array whose sum is equal to the given value. If there is such a triplet present in array, then print the triplet and return true. Else return false.	K3
27	Write a program in Java or Python and find time complexity for Shortest Common Super Sequence Problem: Given two strings X and Y of lengths m and n respectively, find the length of the smallest string which has both, X and Y as its sub-sequences.	K3
28	Write a program in Java or Python and find time complexity for Longest Alternating Subarray Problem: Given an array containing positive and negative elements, find a subarray with alternating positive and negative elements, and in which the subarray is as long as possible.	K3
29	Write a program in Java or Python and find time complexity for 4–Sum Problem: Given an array nums of n integers and an integer target, are there elements a, b, c, and d in nums such that a + b + c + d = target we need to find all unique quadruplets in the array which gives the sum of target.	K3
30	Write a program in Java or Python and find time complexity for K–Partition Problem: Partitioning an array of positive integers into k disjoint subsets that all have an equal sum, and they completely cover the set.	K3
31	Write a program in Java or Python and find time complexity for Minimum Sum Partition Problem: Given a set of positive integers S, partition set S into two subsets, S1 and S2, such that the difference between the sum of elements in S1 and S2 is minimized. The solution should return the minimum absolute difference between the sum of elements of two partitions.	K3
32	Write a program in Java or Python and find time complexity for Wildcard Pattern Matching Problem: We have a string and a pattern then we have to compare the string with a pattern that whether the pattern matches with a string or not	K3
33	Write a program in Java or Python and find time complexity for Maximum Overlapping Intervals Problem: Print the maximum number of overlap among these intervals at any time.	K3
34	Write a program in Java or Python and find time complexity for Graph Coloring Problem: Assigning colors to the vertices such that no two adjacent vertexes have the same color.	K3
35	Write a program in Java or Python and find time complexity for Longest Increasing Subsequence Problem: Given an array **arr[]** of size **N**, the task is to find the length of the Longest Increasing Subsequence (LIS).	K3
36	Write a program in Java or Python and find time complexity for Pots of Gold Game Problem: Two players X and Y are playing a game in which there are pots of gold arranged in a line, each containing some gold coins. They get alternating turns in which the player can pick a pot from one of the ends of the line. The winner is the player who has a higher number of coins at the end. The objective is to maximize the number of coins collected by X, assuming Y also plays optimally. Return the maximum coins X could get while playing the game. Initially, X starts the game.	K3
37	Write a program in Java or Python and find time complexity for Activity	K3

	Selection Problem: Selection of non-conflicting activities that needs to be executed by a single person or machine in a given time frame.	
38	Write a program in Java or Python and find time complexity for Longest Alternating Subsequence Problem:One wants to find a subsequence of a given sequence in which the elements are in alternating order, and in which the sequence is as long as possible.	K3
39	Write a program in Java or Python and find time complexity for Longest Consecutive Subsequence Problem: First sort the array and find the longest subarray with consecutive elements. After sorting the array and removing the multiple occurrences of elements, run a loop and keep a count and max (both initially zero).	K3
40	Write a program in Java or Python and find time complexity for Weighted Interval Scheduling Problem: A value is assigned to each executed task and the goal is to maximize the total value. The solution need not be unique.	K3
41	Write a program in Java or Python and find time complexity for Longest Bitonic Subarray Problem: Find a subarray of a given sequence in which the subarray's elements are first sorted in increasing order, then in decreasing order, and the subarray is as long as possible.	K3
42	Write a program in Java or Python and find time complexity for Water Jugs Problem: You are given two jugs, a 4-gallon one and a 3-gallon one, a pump which has unlimited water which you can use to fill the jug, and the ground on which water may be poured. Neither jug has any measuring markings on it. How can you get exactly 2 gallons of water in the 4-gallon jug?	K3
43	Write a program in Java or Python and find time complexity for Hat Check Problem: Given a positive number n, find the total number of ways in which n hats can be returned to n people such that no hat makes it back to its owner.	K3
44	Write a program in Java or Python and find time complexity for Merging Overlapping Intervals: Start from the first interval and compare it with all other intervals for overlapping.	K3
45	Write a program in Java or Python and find time complexity for Longest Common Prefix (LCP) Problem: An array of strings is the common prefix between 2 most dissimilar strings.	K3

SELF-LEARNING THROUGH MOOCs (Cognitive Skills): Certifications

1. **"Algorithms Specialization" by Stanford University on Coursera:** This specialization covers multiple courses on algorithms and data structures, including "Divide and Conquer, Sorting, and Searching" and "Graph Search, Shortest Paths, and Data Structures."
2. **"Data Structures and Algorithms" by University of California San Diego & National Research University Higher School of Economics on Coursera:** This course covers essential data structures and algorithms, and the programming assignments are often in C++.
3. **"Data Structures and Algorithms - The Complete Masterclass" on Udemy:** This course covers a wide range of data structures and algorithms topics using C++ and includes coding exercises.

4. **"Mastering Data Structures & Algorithms using C and C++" on Udemy:** Another comprehensive course that covers data structures and algorithms with a focus on C and C++ programming languages.
5. **"Data Structures and Algorithms in C++" by Coding Blocks on YouTube:** This is a free YouTube series that covers data structures and algorithms in C++.

Course Lead **Programme Chair** **Dean (SCSE)**