COURSEPACK (Fall 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 Data Structures and Algorithms**			**Course Type**			**Integrated**		
Course Code	**E2UC503C**			**Class**					
Instruction delivery	**Activity**	**Credits**	**Credit Hours**	**Total Number of Classes per Semester**				**Assessment in Weightage**	
	Lecture	**3**	**3**						
	Tutorial	**0**	**0**	**Theory**	**Tutorial**	**Practical**	**Self-study**	**CIE**	**SEE**
	Practical	**1**	**2**						
	Self-study	**1**	**10**						
	Total	**5**	**15**	**45**	**0**	**30**	**150**	**50%**	**50%**
Course Lead	**Dr. Anupam Kumar Sharma**		**Course Coordinator**	**Dr. Nabanita Mahata**					

Names Course Instructors	**Theory**	**Practical**
	Nabanita Mahata	Nabanita Mahata
	Vipul Narayan	Vipul Narayan
	Vijayant Pawar	Vijayant Pawar
	Biswa Mohan Sahoo	Biswa Mohan Sahoo
	Gopal Chandra Jana	Gopal Chandra Jana
	Alok Kumar	Alok Kumar
	Mohd. Arquam	Mohd. Arquam
	Abdul Mazid	Abdul Mazid
	Anupam Kumar Sharma	Anupam Kumar Sharma
	Aditya Kishore Saxena	Aditya Kishore Saxena
	Krishna Kant Agrawal	Krishna Kant Agrawal
	Gyanendra Kumar	Gyanendra Kumar
	Pragya	Pragya
	Shivani Goswami	Shivani Goswami
	Abdul Aleem	Abdul Aleem
	Ravi Kumar	Ravi Kumar
	Vimal Kumar	Vimal Kumar
	G. Sakthi	G. Sakthi
	Abhist Kumar	Abhist Kumar
	Vimal Kumar	Vimal Kumar
	Himanshu Verma	Himanshu Verma
	Pragya	Pragya
	Gyanendra Kumar	Gyanendra Kumar
	Biswa Mohan Sahoo	Biswa Mohan Sahoo
	Aditya Kishore Saxena	Aditya Kishore Saxena
	Abdul Mazid	Abdul Mazid

	Vijayant Pawar Vipul Narayan Raghvendra Ajay Mishra	Vijayant Pawar Vipul Narayan Raghvendra Ajay Mishra

COURSE OVERVIEW

An advanced course in data structures and algorithms typically 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**
		Data Structure and algorithms.

COURSE OBJECTIVE

1. To study and implement complex data structures.
2. To learn advanced algorithm design paradigms and techniques.
3. To study and analyze algorithms for solving complex problems.
4. To apply the knowledge gained to implement and analyze algorithms and data structures in programming assignments and projects.

COURSE OUTCOMES(COs)

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

CO No.	Course Outcomes
E2UC503.1	Explain the concept of Dynamic memory management, priority queue, hashing, backtracking, graphs, dynamic programming, and greedy algorithms.
E2UC503.2	Apply dynamic programming, greedy algorithms, and backtracking to solve real world problems.
E2UC503.3	Analyze the problem for suitable data structures and algorithms.
E2UC503.4	Develop a project based on dynamic programming, greedy algorithms, backtracking, and hashing evaluating the scenario as per requirements.

BLOOM'S LEVEL OF THE COURSE OUTCOMES

Bloom'staxonomyisasetofhierarchicalmodelsusedfortheclassificationofeducationallearning objectivesintolevelsofcomplexityandspecificity.Thelearningdomainsarecognitive,affective,and psychomotor.

COMPREHENSIVE

CO No.	Remember **KL1**	Understand **KL 2**	Apply **KL 3**	Analyse **KL 4**	Evaluate **KL 2**	Create **KL 6**
E2UC 503.1		√				
E2UC 503.2			√			
E2UC 503.3				√		
E2UC 503.4					√	√

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 the 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 the computing science practice.

PO9 Individual and team work: 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 Outcomes and their expected strength of mapping in three levels (low, medium, and high).

COs#/ POs	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
E2UC 503.1	1													1
E2UC 503.2	2	2												1
E2UC 503.3		2	1											1
E2UC 503.4	1	1	1	1	2								1	2

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	Course based project^	CIE	SEE	
Comprehensive	25	50	25	100	100	100

@Lab Work-15 marks + Lab Record-10 marks

^ Typical Rubric for the Course-based project

Type of Assessment Tools	Preliminary Project Plan	Technical Seminar 1	Technical Seminar 2	Viva-voce
Course-based Project Work	05	05	05	10

COURSE CONTENT

THEORY+ PRACTICAL

Contents

Theory

Arrays: Definition, Single and Multidimensional Arrays, Representation of Arrays: Row Major Order, and Column Major Order, Derivation of Index Formulae for 1-D, 2-D, 3-D, Sparse Matrices and their representations.

Linked lists: Implementation of Singly Linked Lists. Operations on Linked List. Insertion, Deletion, Traversal.

Stacks: List Implementation of Stack, Application of stack: infix, Prefix and Postfix Expressions, infix to postfix expression, Evaluation of postfix expression,

Recursion- Principles of recursion, Tail recursion, Removal of recursion Problem solving using iteration and recursion with examples such as binary search, Tower of Hanoi.

Queues: Operations on Queue: Create, Add, Delete, Full & Empty,

Searching, Index Sequential Search, Concept of Hashing & Collision resolution Techniques used in Hashing.

Tree: Linked List Representation, Binary Search Tree, Tree Traversal algorithms: In-order, Pre-order, and Post-order, Constructing Binary Tree from given Tree Traversal, Operation of Insertion, Deletion, Searching Modification of data in Binary Search. Concept & Basic Operations for AVL Tree, B+ tree. Implementation of Quad tree and Oct Tree

Graph: Representations: Adjacency Matrices, Adjacency List, Graph Traversal: Depth First Search & Breadth First Search.

Minimum Cost Spanning Trees: Prims and Kruskal algorithm. Shortest Path algorithm: Warshal Algorithm.

Dynamic Programming: Elements of dynamic programming, Longest common subsequence, Optimal binary search trees, 0/1 Knapsack.

Greedy Algorithms: Elements of the greedy strategy, Offline caching, Fractional knapsack

Back Tracking: Sum of subset, N queens’ problem

Bipartite Graphs: Maximum bipartite matching, The stable-marriage problem **Trie** Data structure, Trie insert and search Bitonic Sort and Radix sort

Lab (To be implemented in Java and Python)

Write a program to implement Stack operations using linked list implementation of Stack.
Write a program to implement Tower of Hanoi.
Write a program for to convert infix expression to postfix and evaluate it.
Write a program to implement Queue operations using linked list implementation.
Write a program to implement Binary Search Tree.
Write a program to traverse a binary tree using PRE-ORDER, IN-ORDER, POST-ORDER traversal techniques.
Write a program to construct a binary tree using given tree traversal.
Write a program for B+ Tree
Write a program to implement AVL Tree.
Write a program to implement Quad Tree.
Write a program to implement Oct Tree.
Write a program to traverse a graph using breadth-first search (BFS).
Write a program to traverse a graph using depth-first search (DFS).
Write a program for Kruskal's algorithm for a given graph
Write a program for Prim's algorithm for a given graph
Write a program for all pairs shortest path.
Write a program for Longest Common Subsequence.
Write a program for 0/1 knapsack.
Write a program for fractional knapsack.
Write a program to implement N queens' problem.
Write a program to insert and delete in Trie data structure.
Write a program to implement Bitonic sort.
Write a program to implement Radix sort.

LESSON PLAN FOR COMPREHENSIVE COURSES

FOR THEORY 15 weeks * 3 Hours = 45 Classes) (1credit = 1Lecture Hour)
FOR PRACTICAL 15 weeks * 2Hours = 30 Hours lab sessions (1 credit = 2 lab hours)

SL-No	Topic for Delivery	Tutorial/ Practical Plan	Skill	Competency
1	Arrays: Definition, Single and Multidimensional Arrays, Representation of Arrays: Row Major Order, and Column Major Order	Theory	understand concepts of array, stack, and linked list.	Student will be able to apply data structure concepts using static and dynamic memory allocation.
2	Derivation of Index Formulae for 1-D, 2-D array	Theory		
3	Derivation of Index Formulae for 3D array, Sparse Matrices, and their representations	Theory		
4	Write a program to implement Stack operations using linked list implementation of Stack.	Practical		
5		Practical		
6	Implementation of Singly Linked Lists	Theory		
7	Operations on Linked List: Insertion, Deletion, Traversal	Theory		
8	Linked list Implementation of Stack	Theory		
9	Write a program to implement Tower of Hanoi.	Practical		
10		Practical		
11	Infix to postfix expression, Evaluation of postfix expression	Theory		

12	Recursion: Principles of recursion, Tail recursion	Theory	understand recursion, and queue, hashing	Student will be able to apply recursion, queue, hashing concepts in complex problems.
13	Removal of recursion Problem solving using iteration and recursion with examples such as binary search	Theory		
14	Write a program for to convert infix expression to postfix and evaluate it.	Practical		
15		Practical		
16	recursion examples: Tower of Hanoi.	Theory		
17	Queues: linked implementation of queues Operations on Queue: Create, Add, Delete, Full & Empty	Theory		
18	Index Sequential Search	Theory		
19	Write a program to implement Queue operations using linked list implementation.	Practical		
20		Practical		
21	Concept of Hashing & Collision resolution Techniques used in Hashing.	Theory		
22	Linked List Representation, Binary Search Tree,	Theory	understand tree data structure, balanced tree (AVL), Quad and Oct tree.	Student will be able to solve problems using tree data structure.
23	Operation of Insertion, Deletion, Searching Modification of data in Binary Search.	Theory		
24	Write a program to implement Binary Search Tree.	Practical		
25		Practical		
26	Tree Traversal algorithms: In-order, Pre-order, and Post-order,	Theory		
27	Constructing Binary Tree from given Tree Traversal,	Theory		
28	Concept & Basic Operations for AVL Tree	Theory		

29	Write a program to traverse a binary tree using PRE-ORDER, IN-ORDER, POST-ORDER traversal techniques.	Practical		
30		Practical		
31	Insertion in AVL trees	Theory		
32	Deletion in AVL trees	Theory		
33	Implementation of Quad tree and Oct Tree	Theory		
34	Write a program to construct a binary tree using given tree traversal.	Practical		
35		Practical		
36	Introduction B+ tree, insertion and Deletion in B+ tree	Theory		
37	Data Structure for Graph Representations: Adjacency Matrices, Adjacency List,	Theory	understand graph data structure and its traversal, spanning tree and conversion of graph to MST.	Student will be able to apply graph algorithms in problems.
38	Graph Traversal: Depth First Search, Breadth First Search	Theory		
39	Write a program for B+ Tree	Practical		
40		Practical		
41	Graph Traversal: Breadth First Search	Theory		
42	Spanning Trees, Minimum Cost Spanning Trees: Prim's algorithm	Theory		
43	Kruskal algorithm	Theory		
44	Write a program to implement AVL Tree.	Practical		
45		Practical		
46	Shortest Path algorithm: Warshal's Algorithm	Theory		
47	Elements of dynamic programming	Theory	understand dynamic programming, longest common subsequence, 0/1 knapsack, greedy method,	Student will be able to apply a concepts of dynamic programming greedy method, back tracking, trie
48	Longest common subsequence	Theory		
49	Write a program to implement Quad Tree.	Practical		
50	Write a program to implement Oct Tree.	Practical		
51	Optimal binary search trees (1)	Theory		

52	Optimal binary search trees (2)	Theory	backtracking, Bipartite graph, trie datastructure	data structure in programming problems.
53	0/1 knapsack problem	Theory		
54	Write a program to traverse a graph using breadth-first search (BFS).	Practical		
55	Write a program to traverse a graph using depth-first search (DFS).	Practical		
56	Elements of the greedy strategy	Theory		
57	Offline caching	Theory		
58	Fractional knapsack	Theory		
59	Write a program for Kruskal's algorithm for a given graph	Practical		
60	Write a program for Prim's algorithm for a given graph	Practical		
61	Back Tracking, Sum of subset	Theory		
62	N queens' problem	Theory		
63	Bipartite graph, maximum bipartite matching	Theory		
64	Write a program for all pairs shortest path.	Practical		
65	Write a program for Longest Common Subsequence.	Practical		
66	The stable-marriage problem	Theory		
67	Trie Data structure	Theory		
68	Trie insert and search	Theory		
69	Write a program for 0/1 knapsack.	Practical		
70	Write a program to implement N queens' problem.	Practical		
71	Trie delete	Theory		
72	Radix Sort	Theory		
73	Bitonic Sort	Theory		
74	Write a program to insert and delete in Trie data structure.	Practical		
75	Write a program to implement Bitonic sort, radix sort	Practical		

BIBLIOGRAPHY

- □ **Text Book** Corman, Introduction to Algorithms
- □ **Reference Books**
  - R. Kruse etal, “Data Structures and Program Design in C”, Pearson Education
  - G A V Pai, “Data Structures and Algorithms”, TMH

Journals/Magazines/Govt. Reports/Gazatte/Industry Trends (Two Numbers)

- □ **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
  -

PROBLEM-BASED LEARNING

Exercises in Problem-based Learning (Assignments) (Min 45 Problems*)

SNo	Problem	KL
1	2–Sum Problem: Finding two numbers in an array that add up to a given target value.	K3
2	Longest Common Subsequence Problem: Finding longest subsequence which is common in all given input sequences.	K4
3	Maximum Subarray Problem: Finding a contiguous subarray with the largest sum, within a given one-dimensional array A[1...n] of numbers.	K3
4	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	0–1 Knapsack Problem: Restrict the number of copies of each kind of item to zero or one.	K4
6	Subset Sum Problem: Checking if there is a subset of the given set whose sum is equal to the given sum.	K3
7	Longest Palindromic Subsequence Problem: Finding a maximum-length subsequence of a given string that is also a Palindrome.	K3
8	Matrix Chain Multiplication Problem: Finding the most efficient way to multiply these matrices together such that the total number of element multiplications is minimum.	K4
9	Longest Common Substring Problem: A set of strings can be found by building	K3

	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.	
10	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	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	Edit Distance 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.	K4
13	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	Partition Problem: A given set can be partitioned in such a way, that sum of each subset is equal.	K3
15	3–Partition Problem: Deciding whether a given multiset of integers can be partitioned into triplets that all have the same sum.	K3
16	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.	K4
17	Largest Consecutive Subarray Problem: Finding out the largest sum of the consecutive numbers of the array.	K3
18	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	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	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	Longest Palindromic Substring Problem: Finding a maximum-length contiguous substring of a given string that is also a palindrome.	K4
22	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	N–Queens Problem: Placing N chess queens on an N×N chessboard so that no two queens attack each other.	K3
24	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	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	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	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.	K4
28	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	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	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	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	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	Maximum Overlapping Intervals Problem: Print the maximum number of overlap among these intervals at any time.	K3
34	Graph Coloring Problem: Assigning colors to the vertices such that no two adjacent vertexes have the same color.	K3
35	Longest Increasing Subsequence Problem: Given an array **arr[]** of size **N**, the task is to find the length of the Longest Increasing Subsequence (LIS).	K4
36	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	Activity Selection Problem: Selection of non-conflicting activities that needs to be executed by a single person or machine in a given time frame.	K3
38	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	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).	K4
40	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	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	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?	K4
43	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	Merging Overlapping Intervals: Start from the first interval and compare it with all other intervals for overlapping.	K4
45	Longest Common Prefix (LCP) Problem: An array of strings is the common prefix between 2 most dissimilar strings.	K4
46	Maximum Product Rod Cutting Problem: Get the maximum product by making a cut at different positions and comparing the values obtained after a cut.	K4
47	Box Stacking Problem: Pile up boxes one on top of the other so that you form the maximum height of box pile-up.	K4
48	Maximum Product Subset Problem: Given an integer array, find the maximum product of its elements among all its subsets.	K4
49	Maximum Independent Set Problem: Finding the largest independent set in a graph, where an independent set is a set of vertices such that no two vertices are adjacent.	K4

STUDENT-CENTEREDLEARNING (SELF-LEARNING TOWARDS LIFE-LONG-LEARNING)

Self-Learning (it’s a typical course-based project to be carried out by a whole class in groups of four students each; they should exhibit higher level KLs)

The students, in a group, are expected to conceive an idea based on the content (objectives/outcomes) and apply suitable knowledge to demonstrate their learning.

A) COURSE-BASED PROJECT (Psychomotor skills) (Min 45 Projects*)

To enhance their skill set in the integrated course, the students are advised to execute course-based **design projects**. Some sample projects are given below:

SNo	Suggested Projects	KL
1	Snake Game	K5
2	Sudoku	K5
3	To-Do list	K5
4	Social Media Network	K6
5	Phonebook	K5
6	Library Management System	K6
7	Maze	K5

8	Music Playlist	K5
9	Calendar	K5
10	Student Grade Checker	K5
11	Flight Route Planner	K5
12	Spell Checker	K5
13	Web Crawler	K5
14	File Compression Tool	K6
15	Real-Time Trafic Analysis	K5
16	Shopping Cart App	K5
17	Word Frequency Counter	K5
18	Online Bookstore	K6
19	Decision Support System	K5
20	Banking System	K5
21	Tic-Tac-Toe	K5
22	Memory Matching Game	K5
23	Tower of Hanoi Puzzle	K5
24	Crossword Puzzle	K6
25	Hangman	K5
26	Real Estate Property Search	K5
27	Email Spam Filter	K6
28	Contacts directory System	K5
29	LIBRARY MANAGEMENT SYSTEM	K5
30	Travelling Agency Project In C	K5
31	Traffic Light Implementation	K6
32	Tic Tac Toe Game	K5
33	Telephone Directory Project	K5
34	Restaurant billing Project	K5
35	Travelling Agency Project	K5
36	Hospital Management System	K5
37	Restaurant billing Project	K5
38	Travelling Agency Project	K5

39	Banking Management System Project	K5
40	Train Reservation System	K5
41	Supermarket Billing System Project	K5
42	Cruise Management Project	K6
43	Diabetes Analysis Project	K6
44	Calendar Application	K6
45	3D Bounce game	K6

B) SELF-LEARNING THROUGH MOOCs (Cognitive Skills): Certification

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++.