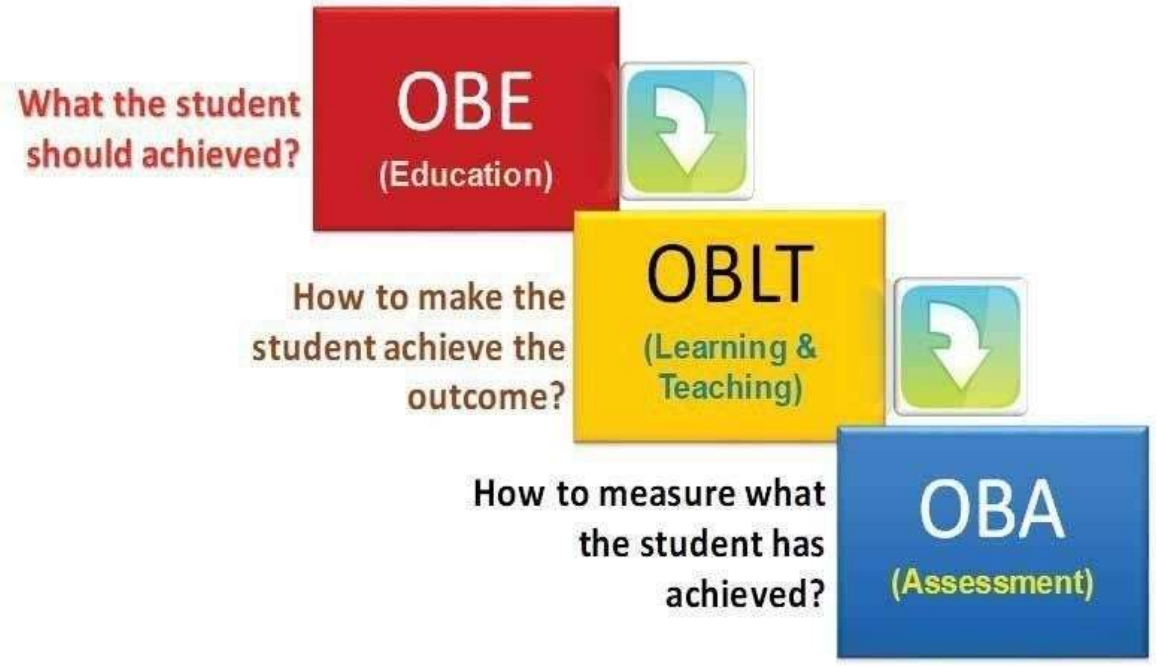
GALGOTIAS UNIVERSITY

GUSOP-01

Course-Pack Framework

A comprehensive instructional delivery document



Prepared by:

VCO

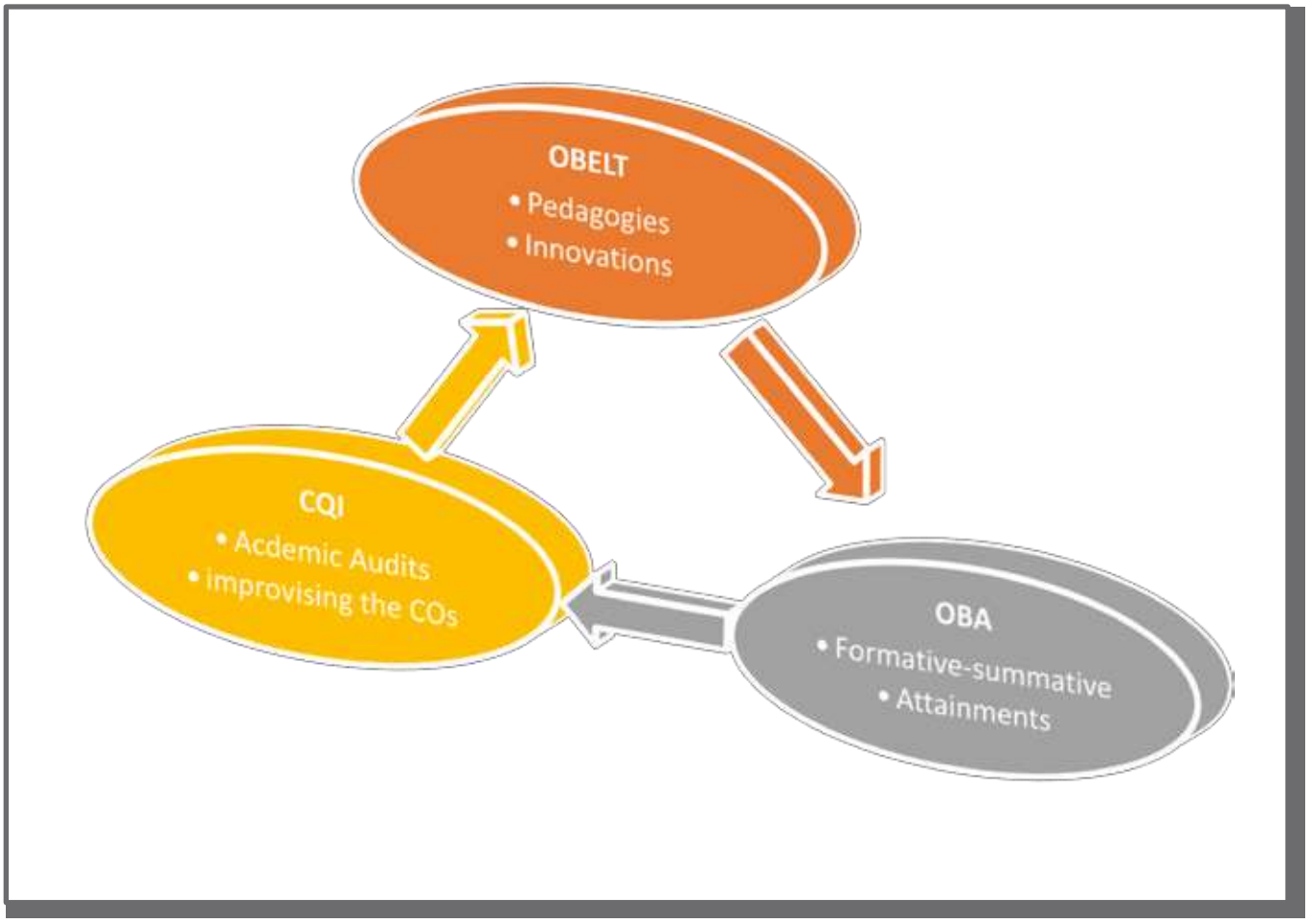
(Revised on Aug-2023)

COURSEPACK
FRAMEWORK

The Course Pack is a comprehensive and complete pedagogical guideline document that describes the components of instruction delivery by a faculty member. It consists of the scheme of the course, Course Overview, Course Objectives, Prerequisite course, Program-specific Outcomes (PSOs), Course outcomes (COs), Bloom's taxonomy (Knowledge Levels), Types of Courses, Course articulation matrix, Course assessment patterns, Course content, Lesson Plan, Bibliography, Problem-based learning/case-studies/clinical, and Student-Centered learning (self- learning towards life-long-learning). It not only provides a uniform design of Course delivery across the University but also ensures freedom and flexibility to introduce innovations in learning and teaching and create vivid kinds of assessment tools (alternate assessment tools) by a faculty member.

The course pack is developed by the faculty member teaching a course. If more than one faculty teaches the same course, all the faculty members teaching the course shall be formed as a cluster, and a senior faculty member (Course-lead) lead the Course delivery design in a team effort. The Course Pack provides ample scope and opportunity to bring innovations in teaching pedagogies in a school/department.

Hence, the Course pack is a comprehensive learning-teaching strategy framework to be followed by all the faculty members in schools/departments in the university. It is not only a tool for measuring the learning of a class but also analyses the achievement levels (learning outcomes of the course) of all the students in a class in a continuous manner.



COURSEPACK

SCHEME

Course Title	**Data Structures**			**Course Type**				**Integrated**		
Course Code	**R1UC308B**			**Class**				B.Tech CSE and Specialization III Sem		
Instruction delivery	**Activity**	**Credits**	**Credit Hours**	**Total Number of Classes per Semester**				**Assessment in Weightage**		
	Lecture	3	3	**Theory**	**Tutorial**	**Practical**	**Self-study**	**CIE**	**SEE**	
	Tutorial	0	0							
	Practical	1	2							
	Self-study	0	7							
	Total	4	12	45	0	30	105	**50%**	**50%**	
Course Lead	Dr. Subhash Chandra Gupta		**Course Coordinator**	Dr. Riman Mandal						

Names Course Instructors	**Theory**		**Practical**	
	Dr.	Subhash Chandra Gupta, **Course Lead**	Dr.	Subhash Chandra Gupta, **Course Lead**
	Dr.	Riman Mandal, **Coure Coordinator**	Dr.	Riman Mandal, **Coure Coordinator**
	Dr.	Gaurav Agarwal	Dr.	Gaurav Agarwal
	Dr.	SHACHI MALL	Dr.	SHACHI MALL
	Dr.	Radha Rani	Dr.	Radha Rani
	Mr.	Abhishek Kumar Pandey	Mr.	Abhishek Kumar Pandey
	Mr.	Akhilesh Kumar Tripathi	Mr.	Akhilesh Kumar Tripathi
	Mr.	Jawed Akhtar	Mr.	Jawed Akhtar
	Ms.	Jyoti Yaduwanshi	Ms.	Jyoti Yaduwanshi
	Mr.	MANDEEP	Mr.	MANDEEP
	Mr.	Kushal Gupta	Mr.	Kushal Gupta
	Mr.	Sachin Kumar tyagi	Mr.	Sachin Kumar tyagi
	Mr.	Sarvesh Kumar Swarnakar	Mr.	Sarvesh Kumar Swarnakar
	Dr.	Gaurav Agarwal	Dr.	Gaurav Agarwal
	Mr.	Pradeep Chauhan	Mr.	Pradeep Chauhan
			Dr.	Tarun Maini

COURSE OVERVIEW

Data structures are one of the foundation subjects for the computer science program. This course enables the students to understand and apply the concepts of Linear Data Structure and Non-linear Data Structure approaches in different scenarios of information processing and storing and organizing data in a computer's memory so that these data can be used efficiently later. It includes identifying the issue in that particular scenario required to deal with the data structures approaches, considering several techniques to address it, and selecting the best data structure to store, organize, and efficiently use the stored data. In this course, Data structures include- Array, lists, stacks, queues, trees, and graphs. The Java programming language will be used to demonstrate the concepts discussed in the lecture, and programming problems must be completed in Java.

PREREQUISITE COURSE

PREREQUISITE COURSE REQUIRED	Yes	
If, yes please fill in the Details	**Prerequisite course code**	**Prerequisite course name**
	C/ C++ Programming	**C/ C++ Programming**

COURSE OBJECTIVE

This course will introduce students to searching and sorting as well as data structures like stack, queue, and binary tree. Students will write many programs around these topics during this course

COURSE OUTCOMES (COs)

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

CO No.	Course Outcomes
CO1	Understand and analyze the fundamentals of data structures, including arrays, linked lists, stacks, queues, trees, and graphs, and apply appropriate representations in problem-solving. *(Blooms Level: Understand & Apply)*
CO2	Implement searching and sorting algorithms efficiently and evaluate their performance using time and space complexity analysis. *(Blooms Level: Apply & Analyze)*
CO3	Develop recursive and iterative solutions for real-world problems, comparing trade-offs in terms of efficiency and readability. *(Blooms Level: Apply & Analyze)*
CO4	Apply various data structures and algorithmic techniques (e.g., trees, graphs, heaps) to solve problems in structured and modular programming environments. *(Blooms Level: Apply & Create)*

PROGRAM OUTCOMES (POs):

PO1	**Engineering Knowledge:** Apply knowledge of mathematics, natural science, computing, engineering fundamentals and an engineering specialization as specified in WK1 to WK4 respectively to develop to the solution of complex engineering problems.
PO2	**Problem analysis:** Identify, formulate, review research literature and analyze complex engineering problems reaching substantiated conclusions with consideration for sustainable development. (WK1 to WK4)
PO3	**Design/development of solutions:** Design creative solutions for complex engineering problems and design/develop systems/components/processes to meet identified needs with consideration for the public health and safety, whole-life cost, net zero carbon, culture, society and environment as required. (WK5).
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.(WK8)
PO5	**Engineering Tool Usage:** Create, select and apply appropriate techniques, resources and modern engineering & IT tools, including prediction and modelling recognizing their limitations to solve complex engineering problems. (WK2 and WK6)
PO6	**The Engineer and The World:** Analyze and evaluate societal and environmental aspects while solving complex engineering problems for its impact on sustainability with reference to economy, health, safety, legal framework, culture and environment. (WK1, WK5, and WK7).
PO7	**Ethics:** Apply ethical principles and commit to professional ethics, human values, diversity and inclusion; adhere to national & international laws. (WK9)
PO8	**Individual and Collaborative Team work:** Function effectively as an individual, and as a member or leader in diverse/multi-disciplinary teams.
PO9	**Communication:** Communicate effectively and inclusively within the engineering community and society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations considering cultural, language, and learning differences
PO10	**Project Management and Finance:** Apply knowledge and understanding of engineering management principles and economic decision-making and apply these to one's own work, as a member and leader in a team, and to manage projects and in multidisciplinary environments.
PO11	**Life-long learning:** Recognize the need for, and have the preparation and ability for i) independent and life-long learning ii) adaptability to new and emerging technologies and iii) critical thinking in the broadest context of technological change. (WK8)

Knowledge and Attitude Profile (WK)

WK1: A systematic, theory-based understanding of the natural sciences applicable to the discipline and awareness of relevant social sciences.

WK2: Conceptually-based mathematics, numerical analysis, data analysis, statistics and formal aspects of computer and information science to support detailed analysis and modelling applicable to the discipline.

WK3: A systematic, theory-based formulation of engineering fundamentals required in the engineering discipline.

WK4: Engineering specialist knowledge that provides theoretical frameworks and bodies of knowledge for the accepted practice areas in the engineering discipline; much is at the forefront of the discipline.

WK5: Knowledge, including efficient resource use, environmental impacts, whole-life cost,re- use of resources, net zero carbon, and similar concepts, that supports engineering design and operations in a practice area.

WK6: Knowledge of engineering practice (technology) in the practice areas in theengineering discipline.

WK7: Knowledge of the role of engineering insociety and identified issues in engineering practice in the discipline, such as the professional responsibility of an engineer topublic safety and sustainable development.

WK8: Engagement with selected knowledge in the current research literature of the discipline, awareness of the power of critical thinking and creative approaches to evaluate emerging issues.

WK9: Ethics, inclusive behavior and conduct. Knowledge of professional ethics, responsibilities, and norms of engineering practice. Awareness of the need for diversity by reason of ethnicity, gender, age, physical ability etc. with mutual understanding.

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 to solve live problems in various domains

BLOOM'S LEVEL OF THE COURSE OUTCOMES

INTEGRATED

CO No.	Remember **BTL1**	Understand **BTL2**	Apply **BTL3**	Analyse **BTL4**	Evaluate **BTL5**	Create **BTL6**
CO1			√			
CO2				√		
CO3				√		
CO4						√

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

COURSE ARTICULATIONMATRIX

COs#/POs	**PO1**	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**		**PSO1**	**PSO2**
CO1	3	2	1	-	-	-	-	-	-	-	-	-	-	-
CO2	3	3	2	-	2	-	-	-	-	-	-	-	-	-
CO3	3	3	1	2	1	-	-	-	-	-	1	-	-	-
CO4	3	3	3	2	3	-	-	-	-	-	2	-	-	-

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

COURSE CONTENT

Content
THEORY: Data Structures (R1UC308B) **Introduction:** Basic Terminology, Types and application of Data Structures, Algorithm, Efficiency of an algorithm, Time-space trade-off and complexity, asymptotic notation. **Array:** Single and Multidimensional Arrays, Representation of Arrays: Row Major Order, and Column Major Order, Derivation of Index Formulae for 1-D, 2-D, and multi-D Array, Application of arrays, Sparse Matrices, and their representations, arithmetic operations on matrices. **Recursion:** Tail recursion, Head Recursion, Nested recursion, Removal of recursion. Problem solving using iteration and recursion with examples such as Fibonacci numbers, and Hanoi towers. Trade-offs between iteration and recursion. **Searching & Sorting:** Linear search, and Binary Search, Insertion Sort, Bubble sort, Selection sort, Quick Sort, Merge Sort. **Linked lists:** Introduction of Structure and Union, Singly Linked Lists, Doubly Linked List, Circularly Linked List, Operations on a Linked List. Insertion, Deletion, Traversal, Reversing, Application of Linked List: Polynomial Representation. **Stack:** Introduction, Abstract Data Type, Primitive Stack operations: Push & Pop, Array and Linked List Implementation of Stack, Application of Stack: Prefix and Postfix Expressions, Evaluation of postfix expression. **Queue:** Introduction, Operations on Queue: Create, Add, Delete, Full and Empty, Circular queues, Array and linked implementation of queues, Double Ended queue, and Priority Queue. **Trees:** Binary Tree and Its array and linked list representation, Strict Binary Tree, Complete Binary Tree, Tree Traversal algorithms: In-order, Pre-order, and Post-order, Constructing Binary Tree from given Tree Traversal, BST Operation: Searching, Insertion, Deletion, Threaded Binary Trees, Heaps, Heap Sort. **Graph**-Introduction to graph, Concepts, and representation, Graph Traversal: BFS and DFS.

COURSE ASSESSMENT

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

Assessment Tools	CIE			Total Marks		Final Marks
	LAB @ (Work + Record)	MTE	LAB EXAM*	CIE	SEE	CIE * 0.5 + SEE * 0.5
Integrated	**25**	**50**	**25**	**100**	**100**	**100**

@ Lab Work – 15 marks + Lab record – 10 marks

* Passing criteria – 30% of marks to be secured in the lab exam conducted by two examiners (one internal + one external).

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)

L.No.	Topics for Delivery	Theory / Tutorial / Practical Plan	Skills	Competency
1	**Introduction:** Basic Terminology, Types and application of Data Structures,	Theory	Understand and analyze the fundamentals of data structures, including arrays, linked lists, stacks, queues, trees, and graphs, and apply appropriate representations in problem-solving.	**CO1**
2	Algorithm, Efficiency of an algorithm, asymptotic notation.	Theory		
3	Time-space trade-off and complexity,	Theory		
4	**Array:** Single and Multidimensional Arrays, Representation of Arrays: Row Major Order, and Column Major Order,	Theory		
5	Derivation of Index Formulae for 1-D,2-D, and multi-D Array,	Theory		
6	Application of arrays, Sparse Matrices, and their representations,	Theory		
7	arithmetic operations on matrices.	Theory		
8	**Recursion:** Tail recursion, Head Recursion,	Theory		
9	Nested recursion, Removal of recursion.	Theory		
10	Problem solving using iteration and recursion with examples such as Fibonacci numbers, and	Theory		
11	Hanoi towers, Trade-offs between iteration and recursion	Theory		
12	**Searching & Sorting:** Linear search, Binary Search,	Theory	Implement searching and sorting algorithms efficiently and evaluate their performance using time and space complexity analysis.	**CO2**
13	Bubble Sort	Theory		
14	Insertion Sort, Selection Sort	Theory		
15	Quick Sort	Theory		
16	Merge Sort	Theory		
17	**Linked lists:** Introduction to Structure and Union, Singly Linked Lists	Theory		

18	Single Linked list : Insert & Traversal	Theory		
19	Single Linked list : Deletion, Searching,	Theory		
20	Double Linked List	Theory		
21	Circular Linked list	Theory		
22	Polynomial Representation	Theory		
23	**Stack:** Introduction, Abstract Data Type, Primitive Stack operations: Push & Pop,	Theory	Develop recursive and iterative solutions for real-world problems, comparing trade-offs in terms of efficiency and readability.	**CO3**
24	Array and Linked List Implementation of Stack,	Theory		
25	Application of Stack: Prefix, Postfix Expression	Theory		
26	Conversion of infix to postfix with algorithm	Theory		
27	Evaluation of postfix expression.	Theory		
28	**Queue:** Introduction of Linear Queue, Queue: Create, Add, Delete, Full and Empty,	Theory		
29	Circular queues Queue: Create, Add, Delete, Full and Empty,	Theory		
30	Array and linked implementation of queues,	Theory		
31	Double Ended queue, and Priority Queue	Theory		
32	**Trees:** Introduction of Tree and their types,	Theory	Apply various data structures and algorithmic techniques (e.g., trees, graphs, heaps) to solve problems in structured and modular programming environments.	**CO4**
33	Binary Tree and Its array and linked list representation, Strict Binary Tree, Complete Binary Tree,	Theory		
34	Tree Traversal algorithms: In-order, Pre-order, and	Theory		
35	Constructing Binary Tree from given Tree Traversal,	Theory		
36	BST Operation: Searching, Insertion	Theory		
37	BST Operation: Deletion, Threaded Binary Trees	Theory		
38	Heaps, Heap Sort.	Theory		
39	**Graph**-Introduction to graph, Concepts, and representation.	Theory		
40	Graph Traversal BFS and DFS	Theory		
41	Revision and Problem Discussion	Theory		

42	Revision and Problem Discussion	Theory		
43	Revision and Problem Discussion	Theory		
44	Revision and Problem Discussion	Theory		
45	Revision and Problem Discussion	Theory		

List of Experiments:

S.No.	Experiments	Schedule date
1	Write a Program to insert and delete an element in 1-D arrays	Week-1
2	Write a Program to implement reverse an arrays	
3	Write a Program to implement addition and multiplication of two 2D arrays	Week-2
4	Write a Program to find max and min of arrays	
5	Write a program to find factorial of any number using iteration and recursion.	Week-3
6	Write a program to find fibonacci series till a number using iteration and recursion.	
7	Write a program to implement Tower of Hanoi problem.	Week-4
8	You are given an integer n. You have to print all numbers from 1 to n using recursion only.	
9	Write a program to implement linear search in 2D-matrix	Week-5
10	Write a program to implement Binary search in an array.	
11	Given sorted array **arr** with possibly some duplicates, the task is to find the first and last occurrences of an element **x** in the given array.	
12	Write a program to implement Bubble Sort	Week-6
13	Write a program to implement Selection Sort	
14	Write a program to implement Insertion Sort	
15	Write a program to implement Quick Sort	Week-7
16	Write a program to implement Merge Sort	
17	Write a menu driven program to implement Single Linked List for operations : Insert (start, end and at position), Delete (start, end and at position), Search, and Display	Week-8
18	Write a program to implement doubly linked list and display data of all nodes.	Week-9
19	Write a program to implement circular linked list and display data of all nodes.	
20	Write a program to implement Stack operations push, pop, isEmpty and isFull using array.	Week-10
21	Write a program to implement Circular Queue operations push, pop, isEmpty and isFull using array.	

22	Write a program to implement in-order, pre-order and post-order traversal of binary tree.	**Week-11**
23	Write a program to implement BST. Write a function to insert, delete and search from BST.	
24	Write a Program to implement BFS of graph.	**Week-12**
25	Write a Program to implement DFS of graph.	

BIBLIOGRAPHY

Text Book

1. Aaron M. Tenenbaum, Yedidyah Langsam and Moshe J. Augenstein“Data Structures”.
2. Sahni Sartaj, “Data structures, algorithms, and applications in Java”, McGraw-Hill

Reference Books

1. Michael T. Goodrich; Roberto Tamassia; Michael H. Goldwasser; Subhasish Banerjee “Data Structures and Algorithms in Java”, Wiley.
2. Jean Paul Trembley and Paul G. Sorenson, “An Introduction to Data Structures with applications”, Mc Graw Hill
3. James Cutajar “Beginning Java Data Structures and Algorithms”, O’Reilly.
4. John Hubbard "Data Structures with Java" Schaum's Outline Series.
5. Narasimha Karumanchi "Data Structures and Algorithms Made Easy in Java" Career Monk

Webliography

1. https://cse.iitkgp.ac.in/~dsamanta/javads/index.htm
2. https://www.geeksforgeeks.org/data-structures/
3. https://www.javatpoint.com/data-structures-in-java
4. https://www.programiz.com/dsa
5. https://www.w3resource.com/
6. https://www.javaguides.net/p/data-structures-and-algorithms-in-java.html

SWAYAM/NPTEL/MOOCs Certification

1. **Data Structure and Algorithms using Java**

   URL: https://archive.nptel.ac.in/courses/106/105/106105225/

2. **Data Structures and Algorithms In Java (DSA)**

   URL: https://www.udemy.com/course/data-structures-and-algorithms-in- java/

3. **Object Oriented Java Programming: Data Structures and Beyond Specialization**

   URL: https://www.coursera.org/specializations/java-object-oriented

4. **Introduction to Data Structures & Algorithms in Java (LinkedIn)**

   URL: https://www.linkedin.com/learning/introduction-to-data-structures- algorithms-in-java

5. **Data Structures and Algorithms – Self Paced (GeeksforGeeks)** URL: https://practice.geeksforgeeks.org/courses/dsa-self-paced

PRACTICE PROBLEMS

1. Write a program to find the number of elements in the largest increasing sequence in an array.
2. Write a program to remove duplicates from an array.
3. Write a Program to find the transpose of a matrix.
4. Write a program to find the second smallest element in a linked list.
5. Write a program to check whether doubly linked list elements make palindrome or not.
6. Given an array, arr[2……11][5……20] with base value 200, and the size of each element is 2 Byte in memory. Find the address of arr[7][11] with the help of row- major order.
7. Given an array, arr[2:11, -3:4, 7:18] with a base value of 300 and the size of each element is 3 Bytes in memory find the address of element arr[4][-2][12] with the help of row-major order?
8. Write a program to calculate the average value of array elements.
9. Write a program to find the Middle Element of a Linked List in a single traversal.
10. A circular queue has a size of 5 and has 3 elements 10,20 and 40 where F=2 and R=4. After inserting 50 and 60, what is the value of F and R. Trying to insert 30 at this stage what happens? Delete 2 elements from the queue and insert 70, 80 & 90. Show the sequence of steps with necessary diagrams with the value of F & R.
11. Construct an expression tree for the expression (a+b*c) + ((d*e+f)*g). Give the outputs when you apply in-order, preorder, and post-order traversals.
12. Given input {4371, 1323, 6173, 4199, 4344, 9679, 1989} and a hash function h(x) =x mod 10. Prepare the results for the following: i) Open addressing hash table using linear probing. (ii)Open addressing hah table using quadratic probing.
13. Write a bubble sort program with a condition that its best-case complexity will be O(n).
14. Write a program to merge two sorted arrays in a single array.
15. Write a program to reverse elements of the queue using stack.
16. Write a recursive program to find the greatest common divisor of two numbers.
17. Develop a program to merge two sorted linked lists (P & Q). Assume that they are available to get a single sorted list S. Eg. P:1->2->45->56 Q:3->24->56->63->66
18. Evaluate the postfix expression 10 5 + 60 6/ * 8 -

19. Drive a formula to calculate the address of a 3-dimensional array in column-major order.
20. Implement Queue using two stacks.
21. Data are pushed to (PUSH operation) and popped from (POP operation) a stack in the following order: PUSH 3; TOP; PUSH 7; TOP; PUSH 6; PUSH 9; TOP; POP; POP;
22. TOP; where the PUSH, POP, and TOP are the standard operations of the stack. Write the values returned by TOP for the sequence of operations above.
23. Write a program to reverse the number using stack.
24. Write a program to reverse a stack using recursion, without using any loop.
25. Write a program to reverse a string using stack.
26. Convert the following Infix expression into a Postfix expression using the Tabular method. a – b / c * d + e * f / g
27. Write a program to find the length of the longest consecutive elements sequence from an unsorted array of integers.
28. Sample array: [49, 1, 3, 200, 2, 4, 70, 5]
29. Let, the longest consecutive elements sequence is [1, 2, 3, 4, 5], therefore the
30. program will return its length 5.
31. What is the output of quick sort after the 3rd iteration given the following
32. sequence? 24 56 47 35 10 90 82 31
33. Implement two stacks in a single array.
34. The following postfix expression is evaluated using a Stack: 8 2 3 ^ / 2 3 * + 5 1 * – ^ is an exponential operator, find out the top two elements after * on the top of the stack.
35. Write a program to get the nth element from the bottom of the stack.
36. Write a program to convert Decimal to binary using recursion.
37. Solve Tower of Hanoi problem with 5 disks and three pegs. (Show all steps)
38. Write a recursive program to find the sum of digits of given numbers.
39. Write a program to find all pairs of elements in an array whose sum is equal to a specified number.
40. Write a program to convert binary to decimal using recursion.
41. Write a recursive method to calculate the product of all numbers in an array.

SELF-LEARNING THROUGH MOOCs (Cognitive Skills): Certification

1. https://www.hackerrank.com
2. https://www.codechef.com
3. https://exercism.org/
4. https://www.codewars.com/
5. https://www.topcoder.com/
6. https://www.coderbyte.com/

(Course Lead) **(Program Chair)** **(Dean)**

GALGOTIAS UNIVERSITY

GALGOTIAS UNIVERSITY

Plot No. 2, Sector 17A, Yamuna Expressway, Opposite Buddha International Circuit,
Greater Noida, Gautam Buddha Nagar, Uttar Pradesh 203201 (INDIA)

0120 - 4370000 9717300418, 9582847072 9971026125

admissions@galgotiasuniversity.edu.in | www.galgotiasuniversity.edu.in