Department of Computer Science & Engineering

Academic Year	:	2025-26	Course Name	:	Analysis & Design of Algorithms
Semester	:	4th -A	Course Code	:	BCS401
Scheme	:	2022	L: T: P: S	:	3:0:0:0 Credits-3
Total Contact hours	:	40	CIE Marks	:	50
Course Plan Author	:	Mrs. Shweta S Bagali	SEE Marks	:	50
Date	:	23/02/2026	Total Marks	:	100

Course Prerequisites:

- Basics of Data Structure
- Discrete Mathematics - sets, functions, relations; Recurrence Relation, Substitution Method Boolean logic

Learning Objectives:

- To learn the methods for analyzing algorithms and evaluating their performance.
- To demonstrate the efficiency of algorithms using asymptotic notations.
- To solve problems using various algorithm design methods, including brute force, greedy, divide and conquer, decrease and conquer, transform and conquer, dynamic programming, backtracking, and branch and bound.
- To learn the concepts of P and NP complexity classes.

Course Outcomes:

CO		*At the end of the course, student should be able to . . .*	Blooms' Level
CO1	:	Apply asymptotic notations and mathematical techniques to analyze the time and space complexity of algorithms.	L3
CO2	:	Apply brute force, decrease-and-conquer, and divide-and-conquer strategies to design and analyze algorithms.	L3
CO3	:	Analyze and implement transform-and-conquer techniques and space-time tradeoff strategies for efficient problem solving.	L3
CO4	:	Design and evaluate solutions using Dynamic Programming and Greedy strategies for optimization problems.	L3
CO5	:	Analyze computational complexity classes and design solutions using backtracking, branch-and-bound, and approximation algorithms for NP-Hard problems.	L4

Blooms' Taxonomy:

L1	L2	L3	L4	L5	L6
Remembering	**Understanding**	**Applying**	**Analyzing**	**Evaluating**	**Creating**

Program Outcomes:

PO1	:	Engineering knowledge	**PO7**	:	Environment and sustainability
PO2	:	Problem analysis	**PO8**	:	Ethics
PO3	:	Design/development of solutions	**PO9**	:	Individual and team work
PO4	:	Conduct investigations of complex problems	**PO10**	:	Communication
PO5	:	Modern tool usage	**PO11**	:	Project management and finance
PO6	:	The engineer and society	**PO12**	:	Life-long learning

Program Specific Outcomes:

PSO1:	Model computational problems by applying mathematical concepts and design solutions using suitable data structures & algorithmic techniques.
PSO2:	Demonstrate basic knowledge of computer science in efficient design of problem solutions of varying complexity.
PSO3:	Create career path to become a successful computer science professional, entrepreneur and relish for higher studies.

CO-PO-PSO Mapping:

CO	Program Outcomes														
	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO 3
CO1	3	3	1	2	1	-	-	-	-	-	-	2	3	2	1
CO2	3	3	2	2	2	-	-	-	-	-	-	2	3	3	2
CO3	3	3	2	2	2	-	-	-	-	-	-	2	3	3	2
CO4	3	3	3	2	2	-	-	-	-	-	-	2	3	3	2
CO5	3	3	2	2	1	-	-	-	-	-	-	3	3	2	2

CO-PO Mapping justification

CO	PO/PSO	Correlation Weightage	Justification
CO1	**PO1**	**3**	Students apply engineering mathematics (asymptotic analysis, recurrence relations, growth functions) to analyze algorithm efficiency, directly supporting engineering knowledge.
	PO2	**3**	Complexity analysis and comparison of algorithms require systematic problem analysis and logical reasoning.

	PO3	**1**	Limited involvement in design; primarily focuses on analysis rather than solution development.
	PO4	**2**	Students investigate algorithm behavior using mathematical proofs and empirical complexity verification.
	PO5	**1**	Basic implementation and use of programming tools for validating time complexity.
	PO12	**2**	Builds strong theoretical foundation necessary for continuous learning in advanced computing domains.
	PSO1	**3**	Strongly related as asymptotic analysis and mathematical modeling directly support computational problem modeling.
	PSO2	**2**	Enabling students to apply fundamental complexity analysis techniques to evaluate and design efficient solutions for problems of varying complexity.
	PSO3	**1**	Builds analytical thinking skills essential for higher studies and professional growth.
CO2	**PO1**	**3**	Application of core computing principles to classical algorithm design strategies.
	PO2	**3**	Students analyze problems and identify suitable design paradigms.
	PO3	**2**	Students develop algorithmic solutions using structured paradigms.
	PO4	**2**	Comparative investigation of sorting/searching algorithms and performance evaluation.
	PO5	**2**	Implementation of algorithms using programming platforms and analysis tools.
	PO12	**2**	Encourages learning new paradigms for evolving computational problems.
	PSO1	**3**	Applies algorithm design strategies to model and develop structured problem solutions.
	PSO2	**3**	Strengthens the application of core CS concepts to design efficient solutions for varying complexities.
	PSO3	**2**	Enhances problem-solving skills required for professional careers and entrepreneurship.
CO3	**PO1**	**3**	Students apply data structures (heaps, trees) and theoretical concepts for efficient solutions.
	PO2	**3**	Selection of appropriate tradeoff strategy requires analytical decision-making.
	PO3	**2**	Moderate involvement in structured solution design using transformation techniques.
	PO4	**2**	Students experimentally compare time-space efficiency of algorithms.
	PO5	**2**	Use of programming tools to implement heaps, balanced trees, and string matching algorithms.

	PO12	**2**	Develops adaptability to choose efficient strategies in evolving computing problems.
	PSO1	**3**	Enables efficient problem modeling using transform-and-conquer and space–time tradeoff techniques.
	PSO2	**3**	Applies fundamental CS strategies to optimize solutions for complex problems.
	PSO3	**2**	Develops analytical and optimization skills beneficial for advanced studies and industry roles.
CO4	**PO1**	**3**	Strong application of algorithmic principles in optimization problems.
	PO2	**3**	Students analyze complex optimization problems before selecting appropriate strategy.
	PO3	**3**	High involvement in designing optimal solutions using DP and Greedy techniques.
	PO4	**2**	Investigation of optimality conditions and performance comparison of approaches.
	PO5	**2**	Implementation of graph and optimization algorithms using programming tools.
	PO12	**2**	Encourages independent exploration of advanced optimization strategies.
	PSO1	**3**	Supports structured design of optimized solutions using Dynamic Programming and Greedy methods.
	PSO2	**3**	Strengthens the ability to implement efficient solutions for optimization problems.
	PSO3	**2**	Enhances decision-making and optimization skills valuable for research and professional growth.
CO5	**PO1**	**3**	Application of theoretical foundations of computational complexity and advanced algorithmic techniques.
	PO2	**3**	Requires deep analysis of computational limits and problem hardness.
	PO3	**2**	Moderate design of solutions using backtracking and approximation methods
	PO4	**2**	Investigation of solution feasibility and performance bounds.
	PO5	**1**	Limited use of tools for simulation and validation of NP-based problems.
	PO12	**3**	Strong encouragement toward lifelong learning in theoretical computer science and research.
	PSO1	**3**	Advanced algorithmic strategies for NP-Hard problems require strong modeling and structured design approaches.
	PSO2	**2**	Applying fundamental computer science concepts to analyze computational complexity and design efficient solutions for complex and NP-Hard problems.

	PSO3	**2**	Understanding complexity classes and advanced techniques supports higher studies and research orientation.

Course Content (Syllabus)

Module 1	**CH**
INTRODUCTION: What is an Algorithm? Fundamentals of Algorithmic Problem Solving. FUNDAMENTALS OF THE ANALYSIS OF ALGORITHM EFFICIENCY: Analysis Framework, Asymptotic Notations and Basic Efficiency Classes, Mathematical Analysis of Non recursive Algorithms, Mathematical Analysis of Recursive Algorithms. **BRUTE FORCE APPROACHES**: Selection Sort and Bubble Sort, Sequential Search and Brute Force String Matching. **Text Book**: Chapter 1 (Sections 1.1,1.2), Chapter 2(Sections 2.1,2.2,2.3,2.4), Chapter 3(Section 3.1,3.2) RBT:L3	**08**
Module 2	
BRUTE FORCE APPROACHES (Contd...): Exhaustive Search (Travelling Salesman problem and Knapsack Problem). **DECREASE-AND-CONQUER**: Insertion Sort, Topological Sorting. **DIVIDE AND CONQUER**: Merge Sort, Quick Sort, Binary Tree Traversals, Multiplication of Large Integers and Strassen's Matrix Multiplication**.** **Text Book:** Chapter 3(Section 3.4), Chapter 4 (Sections 4.1,4.2), Chapter 5 (Section 5.1,5.2,5.3, 5.4) RBT:L3	**08**
Module 3	
TRANSFORM-AND-CONQUER: Balanced Search Trees, Heaps and Heapsort. **SPACE-TIME TRADEOFFS**: Sorting by Counting: Comparison counting sort, Input Enhancement in String Matching: Horspool's Algorithm. **Text Book:** Chapter 6 (Sections 6.3,6.4), Chapter 7 (Sections 7.1,7.2) RBT:L3	**08**
Module 4	
DYNAMIC PROGRAMMING: Three basic examples, The Knapsack Problem and Memory Functions, Warshall's and Floyd's Algorithms. **THE GREEDY METHOD**: Prim's Algorithm, Kruskal's Algorithm, Dijkstra's Algorithm, Huffman Trees and Codes**.** **Text Book:** Chapter 8 (Sections 8.1,8.2,8.4), Chapter 9 (Sections 9.1,9.2,9.3,9.4) RBT:L3	**08**
Module 5	
LIMITATIONS OF ALGORITHMIC POWER: Decision Trees, P, NP, and NP-Complete Problems.	**08**

COPING WITH LIMITATIONS OF ALGORITHMIC POWER: Backtracking (n-Queens problem, Subset-sum problem), Branch-and-Bound (Knapsack problem), Approximation algorithms for NP-Hard problems (Knapsack problem).

Text Book: Chapter 11 (Section 11.2, 11.3), Chapter 12 (Sections 12.1,12.2,12.3) RBT:L4

Schedule of Instruction:

Class No	Topic	Date	RBT	CO	Mode
Module-1		**A**			
Introduction, Fundamentals of the Analysis of Algorithm Efficiency & Brute Force Approaches					
1.	Introduction to Course Handout, Course Website, Subject	23/02/26	L1	1	Chalk & board
2.	What is an Algorithm? Fundamentals of Algorithmic Problem Solving	25/02/26	L1	1	Chalk & board
3.	Analysis Framework: Space complexity, Time complexity, Asymptotic Notations and Basic Efficiency Classes	26/02/26	L1	1	Chalk & board
4.	Examples on Asymptotic notations	27/02/226	L1	1	Chalk & board
5.	Mathematical Analysis of Non-Recursive Algorithms	2/3/26	L2	1	Chalk & board
6.	Mathematical Analysis of Recursive Algorithms, Selection Sort and Bubble Sort,	4/3/26	L2	1	Chalk & board
7.	More Examples solved on Selection Sort and Bubble Sort,	5/3/26	L2	1	Chalk & board
8.	Sequential Search and Brute Force String Matching.	6/3/26	L2	1	Chalk & board
9.	Sequential Search and Brute Force String Matching.	9/3/26	L2	1	Chalk & board
Module-2					
Brute Force Approaches, Decrease-And-Conquer & Divide And Conquer					
10.	Exhaustive Search (TSP and Knapsack Problem). Exhaustive Search: Knapsack Problem).	11/3/26	L3	2	Chalk & board
11.	**DECREASE-AND-CONQUER**: Insertion Sort, Topological Sorting.	12/3/26	L3	2	Chalk & board
12.	Cont: Topological Sorting.	13/3/26	L3	2	Chalk & board
13.	**DIVIDE AND CONQUER**: Merge Sort, examples, Quick Sort, example solution, Binary Tree Traversals	16/3/26	L3	2	Chalk & board And Roleplay
14.	Binary Tree Traversals, Multiplication of Large Integers	18/3/26	L3	2	Chalk & board
15.	Examples on Multiplication of Large Integers Strassen's Matrix Multiplication.	23/3/26	L3	2	Chalk & board
16.	Strassen's Matrix Multiplication	25/3/26	L3	2	Chalk & board
17.	Exhaustive Search (TSP and Knapsack Problem). Exhaustive Search: Knapsack Problem).	26/3/26	L3	2	Chalk & board

Class No	Topic	Date	RBT	CO	Mode
18	TSP and Knapsack Problem	27/3/26	L3	2	Chalk & board
19	Exhaustive Search: Knapsack Problem).	30/3/26	L3	2	Chalk & board
Module-3					
Transform-And-Conquer & Space-Time Tradeoffs					
20	**TRANSFORM-AND-CONQUER**: Balanced Search Trees -AVL tress	6/4/26	L3	3	Chalk & board
21	2-3 trees	8/4/26	L3	3	Chalk & board
22	Heaps and Heapsort.-Notion of the Heap	9/4/26	L3	3	Chalk & board and Visualize
23	**Heapsort, SPACE-TIME TRADEOFFS**: Sorting by Counting	10/4/26	L3	3	Chalk & board
24	Comparison counting sort,	13/4/26	L3	3	Chalk & board
25	Distribution counting	15/4/26	L3	3	Chalk & board
26	Input Enhancement in String Matching: Horspool's Algorithm	16/4/26	L3	3	Chalk & board
27	Cont: Examples on Horspool's Algorithm	17/4/26	L3	3	Chalk & board and Visualize
28	**Boyer-Moore Algorithm**	20/4/26	L3	3	Chalk & board
29	**Revision Module 3**	27/4/26	L3	3	
Module-4					
Dynamic Programming & The Greedy Method					
30	**DYNAMIC PROGRAMMING**: Three basic examples	29/4/26	L3	4	
31	The Knapsack Problem	30/4/26	L3	4	Chalk & board
32	Memory Functions	4/5/26	L3	4	Chalk & board
33	Warshall's Algorithms	6/5/26	L3	4	Chalk & board

Class No	Topic	Date	RBT	CO	Mode
34	Example Soutions	7/5/26	L3	4	Chalk & board
35	Floyd's Algorithms for All pair shortest-Paths problem	8/5/26	L3	4	Chalk & board
36	**THE GREEDY METHOD**: Prim's Algorithm,	11/5/26	L3	4	Chalk & board
37	Kruskal's Algorithm	13/5/26	L3	4	Chalk & board
38	Dijkstra's Algorithm,	14/5/26	L3	4	Chalk & board & Case Study
39	Revision on Module 4	15/5/26	L3	4	
Module-5					
Limitations Of Algorithmic Power & Coping With Limitations Of Algorithmic Power					
40	**LIMITATIONS OF ALGORITHMIC POWER**: Decision Trees for sorting algorithms ,Decision trees for Searching a sorted array	18/5/26	L4	5	Chalk & board
41	P and NP problems, NP-Complete Problems	20/5/26	L4	5	Chalk & board
42	**COPING WITH LIMITATIONS OF ALGORITHMIC POWER**: Backtracking (n-Queens problem,	21/5/26	L4	5	Chalk & board
43	Subset-sum problem	22/5/26	L4	5	Chalk & board
44	Branch-and-Bound (Knapsack problem),	25/5/26	L4	5	Chalk & board
45	Examples on (Knapsack problem),	27/5/26	L4	5	Chalk & board
46	Approximation algorithms for NP-Hard problems (Knapsack problem).	29/5/26	L4	5	Chalk & board
47	Problems solution on (Knapsack problem).	1/6/26	L4	5	Chalk & board
48	Revision module-5	3/6/26			
49	Revision QP-solution	4/6/26			
50	Revision QP-solution	5/6/26			

Textbooks:

T1	Introduction to the Design and Analysis of Algorithms, By Anany Levitin, 3rd Edition (Indian), 2017, Pearson.

Reference books:

R1	Computer Algorithms/C++, Ellis Horowitz, SatrajSahni and Rajasekaran, 2nd Edition, 2014, Universities Press.
R2	Introduction to Algorithms, Thomas H. Cormen, Charles E. Leiserson, Ronal L. Rivest, Clifford Stein, 3rd Edition, PHI.
R3	Design and Analysis of Algorithms, S. Sridhar, Oxford (Higher Education)

Web links and Video Lectures (e-Resources):

1.	https://sites.google.com/skit.org.in/testdmdw/home (Website for Study Material)
2.	Design and Analysis of Algorithms: https://nptel.ac.in/courses/106/101/106101060/
3.	http://elearning.vtu.ac.in/econtent/courses/video/CSE/06CS43.html
4.	http://elearning.vtu.ac.in/econtent/courses/video/FEP/ADA.html
5.	https://www.cs.usfca.edu/~galles/visualization/Algorithms.html - Simulation
6.	https://ds1-iiith.vlabs.ac.in/

Assessment Schedule:

S.N.	Assessment Type	Content	CO	Duration	Marks	Date
1.	CIE Test 1	Mod-1,2,3(Half)	CO1,CO2,CO3	1.30hr	50(Scale down to 25	21/4/26
2.	CIE Test 2	Mod -3(Half),4,5	CO3,CO4,CO5	1.30hr	50(Scale down to 25	8/6/26
3.	Unit test 1,2,3	Module 1 to 5	CO 1-5	1 Hr	15	
4.	Online Quiz	Module 1 to 5	CO 1-5	1 Hr	05	
5.	Problem Solving Assignments	Module 1to 5	CO 1-5	1Hr	05	
6.	Semester End Examination	All 5 modules	CO1 to 5	3 Hrs	50	

***RB** – Text Book/**R**eference **B**ook, ***L** – **L**ecture, **V**- Videos or any other mode, ***RBT** – **R**evised **B**looms' **T**axonomy, **L: T: P: C** – Theory/**L**ecture: **T**utorial: **P**ractical/Drawing: **C**redits, **SEE**: **S**emester **E**nd **E**xamination, **CIE**: **C**ontinuous **I**nternal **E**valuation,*

The sum of total marks of two tests, two assignments, and Quiz will be out of 100 marks and will be scaled down to 50 marks. (As per the scheme), **CIE + SEE = 50 + 50 = 100 marks

Faculty In charge **Course Coordinator** **HoD**

Prof. Shweta S Bagali