✆ STD - 0431 - 2670142
FAX - 0431 - 2670143

DHANALAKSHMI SRINIVASAN
INSTITUTE OF TECHNOLOGY
(Approved by AICTE, New Delhi & Affiliated to Anna University)
NH - 45, Trichy - Chennai Trunk Road,
SAMAYAPURAM,TRICHY - 621 112.
E.mail:dsit2011@gmail.com Website:www.dsit.ac.in

COURSE PLAN

Subject code: CS8451	**Branch/Year/Sem/Section: B.E CSE/II/IV**
Subject Name: DESIGN AND ANAYSIS OF ALGORITHM	**Batch:2018-2022**
Staff Name: F.ANGEL IGNISHYAA	**Academic year:2019-2020**

COURSE OBJECTIVE

1. To learn the algorithm analysis techniques.
2. To critically analyze the efficiency of alternative algorithmic solutions for the same problem
3. To understand the limitation of algorithm power.
4. To understand different algorithm design techniques.

TEXT BOOK:

T1. Anany Levitin, ―Introduction to the Design and Analysis of Algorithms‖, Third Edition, Pearson Education, 2012.

T2. Ellis Horowitz, Sartaj Sahni and Sanguthevar Rajasekaran, Computer Algorithms/ C++, Second Edition, Universities Press, 2007.

REFERENCES:

R1. Thomas H.Cormen, Charles E.Leiserson, Ronald L. Rivest and Clifford Stein, ―Introduction to Algorithms‖, Third Edition, PHI Learning Private Limited, 2012.

R2. Alfred V. Aho, John E. Hopcroft and Jeffrey D. Ullman, ―Data Structures and Algorithms‖, Pearson Education, Reprint 2006.

R3. Donald E. Knuth, "The Art of Computer Programming", Volumes 1& 3 Pearson Education, 2009. Steven S. Skiena, "The Algorithm Design Manual", Second Edition, Springer, 2008.

R4. Harsh Bhasin, ―Algorithms Design and Analysis‖, Oxford university press, 2016.

R5. S. Sridhar, ―Design and Analysis of Algorithms‖, Oxford university press, 2014.

WEB RESOURCES

W1: https://nptel.ac.in

TEACHING METHODOLOGIES:

- BB - BLACK BOARD
- VIDEO - VIDEO TUTORIAL
- PPT - POWER POINT PRESENTATION

✆ STD - 0431 - 2670142
FAX - 0431 - 2670143

DHANALAKSHMI SRINIVASAN
INSTITUTE OF TECHNOLOGY
(Approved by AICTE, New Delhi & Affiliated to Anna University)
NH - 45, Trichy - Chennai Trunk Road,
SAMAYAPURAM,TRICHY - 621 112.
E.mail:dsit2011@gmail.com Website:www.dsit.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CS8451 **DESIGN AND ANALYSIS OF ALGORITHMS** **L T P C**
3 0 0 3

UNIT I INTRODUCTION 9

Notion of an Algorithm – Fundamentals of Algorithmic Problem Solving – Important Problem Types – Fundamentals of the Analysis of Algorithmic Efficiency –Asymptotic Notations and their properties. Analysis Framework – Empirical analysis - Mathematical analysis for Recursive and Non-recursive algorithms – Visualization

UNIT II BRUTE FORCE AND DIVIDE-AND-CONQUER 9

Brute Force – Computing an – String Matching - Closest-Pair and Convex-Hull Problems - Exhaustive Search - Travelling Salesman Problem - Knapsack Problem - Assignment problem. Divide and Conquer Methodology – Binary Search – Merge sort – Quick sort – Heap Sort - Multiplication of Large Integers – Closest-Pair and Convex - Hull Problems.

UNIT III DYNAMIC PROGRAMMING AND GREEDY TECHNIQUE 9

Dynamic programming – Principle of optimality - Coin changing problem, Computing a Binomial Coefficient – Floyd‘s algorithm – Multi stage graph - Optimal Binary Search Trees – Knapsack Problem and Memory functions. Greedy Technique – Container loading problem - Prim‘s algorithm and Kruskal's Algorithm – 0/1 Knapsack problem, Optimal Merge pattern - Huffman Trees.

UNIT IV ITERATIVE IMPROVEMENT 9

The Simplex Method - The Maximum-Flow Problem – Maximum Matching in Bipartite Graphs, Stable marriage Problem.

UNIT V COPING WITH THE LIMITATIONS OF ALGORITHM POWER 9

Lower - Bound Arguments - P, NP NP- Complete and NP Hard Problems. Backtracking – n-Queen problem - Hamiltonian Circuit Problem – Subset Sum Problem. Branch and Bound – LIFO Search and FIFO search - Assignment problem – Knapsack Problem – Travelling Salesman Problem - Approximation Algorithms for NP-Hard Problems – Travelling Salesman problem – Knapsack problem.

TOTAL: 45 PERIODS

Topic No	Topic Name	Books For reference	Page No	Teaching Methodology	No of periods required	Cumulative periods
UNIT I	**INTRODUCTION**					**(9)**
1.	Introduction, Algorithm, Notion of algorithm	T1 T2	(1-7) (5-11)	BB	1	1
2.	Fundamentals of Algorithmic Problem Solving-steps in designing and analyzing an algorithm	T1 T2	(9-16) (30-39)	BB	1	2
3.	Important Problem Types-Sorting. Searching, string processing, graph, geometric, numeric and combinatorial problems	T1	(18-23)	BB	1	3
4.	Fundamentals of the Analysis of Algorithm Efficiency, Analysis	T1	(41-45)	BB	1	4
5.	Analysis Framework, Orders of growth, worst, best and average case analysis, recapitulation of Analysis Framework.	T1 T2	(45-50) (23-29)	BB	1	5
6.	Asymptotic Notations and its properties- Informal, Big-oh, Big-omega, Big-theta notation	T1 T2 T4	(52-58) (43-53) (31-57)	BB	1	6
7.	Mathematical analysis for Non-recursive algorithms –General plan for analyzing time efficiency	T1	(61-67)	BB	1	7
8.	Mathematical analysis for Recursive algorithms –General plan for analyzing time efficiency	T1 T2	(70-76) (65-75)	BB	1	8
9.	Sample problems	T1	(8,16-18,)	BB	1	9
LEARNING OUTCOME: **At the end of unit , the students will be able to** • To understand the basics about algorithm • Students to learn how to analyze and design algorithms.						
UNIT II	**BRUTE FORCE AND DIVIDE-AND-CONQUER**					**(9)**
10.	Brute Force-selection.bubble sort,	T1	(97-101)	BB	1	10
11.	Brute Force -sequential search,brute force string manipulation	T1	(104-106)	BB	1	11
12.	Closest-Pair and Convex-Hull Problems	T1	(108-113)	BB	1	12

13.	Traveling Salesman Problem - Knapsack Problem	T1 R3	(115-119) (401-405)	BB	1	13
14.	Assignment problem	T1 R4	(119-121) (498-501)	BB	1	14
15.	Divide and conquer methodology master theorem	T1 T2	(169-172) (93-97)	BB	1	15
16.	Merge sort, Quick sort	T1 T2	(172-181) (170-182)	BB & PPT	1	16
17.	Binary search-Binary tree traversal and properties.	T1 R3	(181185) (132139)	BB	1	17
18.	Multiplication of Large Integers – Strassen's Matrix Multiplication, Closest-Pair and Convex-Hull Problems.	T1 T2 R3	(186197), (75-82), (135-137)	BB	2	18
LEARNING OUTCOME: **At the end of unit , the students will be able to** • To make the students solve computing problems. • To using brute force and divide and conquer methods.						
UNIT – III	**DYNAMIC PROGRAMMING AND GREEDY TECHNIQUE**					**(9)**
19.	Dynamic programming-three basic examples –coin row problem	T1	(283-287)	BB	1	19
20.	Change-making problem, Coin-collecting problem	T1	(287-290)	BB & PPT	1	20
21.	Warshall's algorithm	T1	(217-225)	BB	1	21
22.	Floyd' algorithm	T1 R3	(226-237) (210-212)	BB	1	22
23.	Optimal Binary Search Trees	T1 R1	(241-255) (397-403)	BB	1	23
24.	Knapsack Problem and Memory functions.	T1 T2	(249-257) (425-427) (427-431)	BB	1	24
25.	Greedy Technique– Prim's algorithm	T1 R1	(315-322) (634-636)	BB	1	25
26.	Kruskal's Algorithm	T1 R1	(325-331) (631-633)	BB	1	26
27.	Dijkstra's Algorithm, Huffman Trees.	T1 R1 T1	(333-337) (658-662) (338-343)	BB	1	27
LEARNING OUTCOME:						

At the end of unit , the students will be able to

- To given students the exposure about solving problems
- And using dynamic programming and greedy techniques.

UNIT IV	**ITERATIVE IMPROVEMENT**					**(9)**
28.	The Simplex Method	T1	(345-351)	BB	1	28
29.	Geometric interpretation of Linear programming	R1	(846-850)	BB	1	29
30.	Outline of simplex method	T1	(351-359)	BB & PPT	1	30
31.	The Maximum-Flow Problem	T1	1(361-369)	BB	1	31
32.	Max flow-min cut Theorem	T1 R3	(369-371) (258-262)	BB	1	32
33.	Maximum Matching in Bipartite Graphs	T1 R1	(372-375) (732-735)	BB	1	33
34.	Maximum Matching in Bipartite Graphs- theorem and proof	T1 R1	(375-378) (732-735) (217-222)	BB	1	34
35.	The Stable marriage Problem.	T1	(380-381)	BB	1	35
36.	Stable marriage algorithm- theorem and proof	T1	(381-383)	BB	1	36

LEARNING OUTCOME:

At the end of unit , the students will be able to

- To make the students understand solve problems using iterative method.
- Problems solved using iterative methods are discussed for better understanding

UNIT V	**COPING WITH THE LIMITATIONS OF ALGORITHM POWER**					**(9)**
37.	Limitations of Algorithm Power-Lower-Bound Arguments-Methods for establishing lower bounds	T1	(387-392)	BB	1	37
38.	Decision Trees - Decision Tree for sorting and searching in sorted arrays.	T1	(394-397)	BB	1	38
39.	P, NP and NP-Complete Problems	T1	(401-409)	BB	1	39
40.	Coping with the Limitations of algorithm power-backtracking	T1 R3	(423-425) (231-238)	BB	1	40
41.	n-Queens problem – Hamiltonian Circuit Problem- Subset sum problem	T1	(426-430)	BB	1	41
42.	Branch and Bound – Assignment problem	T1	(432-436)	BB	1	42
43.	Knapsack Problem – Traveling Salesman Problem	T1	(436-440) (427-430)	BB & PPT	1	43

44.	Approximation Algorithms for NP – Hard Problems – Traveling Salesman problem – Knapsack problem.	T2 R1	(778-788) (1048-1053)	BB & PPT	1	44
45.	Limitations of Algorithm Power-Lower-Bound Arguments-Methods for establishing lower bounds	T1	(387-392)	BB & PPT	1	45

LEARNING OUTCOME:

At the end of unit , the students will be able to

To make students understand and solve complex problems using backtracking .branch and bound techniques.

COURSE OUTCOME

At the end of the course, the student should be able to:

Design algorithms for various computing problems.

- Analyze the time and space complexity of algorithms.
- Critically analyze the different algorithm design techniques for a given problem.
- Modify existing algorithms to improve efficiency Interpret the fundamental needs of algorithms in problem solving

CONTENT BEYOND THE SYLLABUS

Algorithm design techniques.
Various computing problems.
Time and space complexity algorithms.

CONTINUES INTERNAL ASSESSMENT DETAILS

ASSESMENT NUMBER	I	II	MODEL
TOPIC NO.(UNIT)	1-18 (1^{st} & 2^{nd} units)	19-36 (3^{rd} & 4^{th} units)	1-45 (units 1-5)

ASSIGNMENT DETAILS

ASSIGNMENT NUMBER	I	II	III
TOPIC NUMBER FOR REFERENCE	1-18 (1^{st} & 2^{nd} units)	19-36 (3^{rd} & 4^{th} units)	1-45 (units 1-5)
DEAD LINE			

ASSIGNMENT NUMBER	DESCRIPTIVE QUESTIONS/TOPIC (Minimum of 8 Pages)
I	1. Notion of an Algorithm Describe evaluation of operating System 2. Fundamentals of Algorithmic Problem Solving 3. Fundamentals of the Analysis of Algorithmic Efficiency
II	1. Brute Force 2. String Matching 3. Closest-Pair and Convex-Hull Problems
III	1. Dynamic programming 2. Binomial Coefficient 3. Floyd‘s algorithm

PREPARED BY **VERIFIED BY**

F.ANGEL IGNISHYAA, AP/CSE **HOD/CSE**

APPROVED BY

PRINCIPAL