SSN College of Engineering (Autonomous)

(Affiliated to Anna University, Chennai)

COURSE FOLDER – SYLLABUS

Subject:	**DESIGN AND ANALYSIS OF ALGORITHMS**
Code:	UCS1403
Semester:	4
Class:	BE CSE
Year:	2020-2021
Batch:	2019-2023
Faculty:	R S Milton, C Aravindan

SYLLABUS

L	T	P	C
3	0	2	4

COURSE OBJECTIVES

- Learn the algorithm analysis techniques
- Become familiar with different algorithm design techniques
- Understand the limitations of algorithm power.

Unit I Introduction and Analysis 9

Introduction: Fundamentals of algorithmic problem solving – Important problem types; Fundamentals of the analysis of algorithm efficiency: Analysis framework – Asymptotic notations and basic efficiency classes – Mathematical analysis for recursive and non-recursive algorithms.

Unit II Brute Force and Divide and Conquer 9

Brute Force: String matching – Closest-pair problem; Exhaustive search: Traveling salesman problem – Knapsack problem. Divide and Conquer: Mergesort – Quicksort – Multiplication of large integers – Strassen's matrix multiplication.

Unit III Dynamic Programming and Greedy Technique 9

Dynamic Programming: Computing a binomial coefficient – Knapsack problem and memory functions – Warshall's and Floyd's algorithm – Greedy Technique: Prim's algorithm – Kruskal's algorithm – Dijkstra's algorithm.

Unit IV Iterative Improvement and Backtracking 9

Iterative Improvement: The simplex method – Maximum matching in bipartite graphs; Backtracking: N-queens problem – Hamiltonian circuit problem.

Unit V Limitations of Algorithm Power 9

Branch and Bound: Knapsack problem – Traveling salesman problem; Limitations of algorithm power: Lower-bound arguments – P, NP and NP-complete problems; Coping with the Limitations of algorithm power: Approximation algorithms for NP-Hard problems – Traveling salesman problem – Knapsack problem.

Theory Periods: 45

SUGGESTIVE EXPERIMENTS

1. Implementation of non-recursive and recursive algothms for the given problem
2. Implementation of string matching using Brute Force technique
3. Implementation of Knapsack problem using Exhaustive Search technique
4. Implementation of merge sort and quick sort using Divide and Conquer technique
5. Implementation of Knapsack Problem using Dynamic Programming
6. Implementation of Prim's and Dijkstra's algorithms.
7. Implementation of n-Queens problem using Backtracking technique
8. Implementation of Knapsack using Branch and Bound technique
9. Mini project

Practical Periods: 30
Total Periods: 75

COURSE OUTCOMES

After the completion of this course, students will be able to:

- Design algorithms for various computing problems (K3)
- Analyze the time and space complexity of algorithms (K4)
- Compare different algorithm design techniques for a given problem (K4)
- Modify existing algorithms to improve efficiency (K4)
- Understand the limitations of algorithmic power (K2)

TEXT BOOKS

1. Anany Levitin, “Introduction to the Design and Analysis of Algorithms”, 3rd Edition, Pearson Education, 2012.

2. S Dasgupta, C H Papadimitriou, U V Vazirani, “Algorithms”, 1st Edition, McGraw Hill Education, 2017.

REFERENCES

1. Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, Clifford Stein, “Introduction to Algorithms”, 3rd Edition, PHI Learning Private Limited, 2012.

2. Steven S Skiena, “The Algorithm Design Manual”, 2nd Edition, Springer, 2008.

Prepared	Verified	Approved
R S Milton C Aravindan	PAC	HoD, CSE