Course No.:	Algorithm Design & Implementation	Credits: 3-0-0-6	Prerequisites : NIL

Course Objectives:

1. Develop a deeper understanding of the theoretical foundations of algorithm design and analysis, including key concepts such as asymptotic analysis and NP-hardness.
2. Learn to apply various algorithmic design paradigms, including backtracking, dynamic programming, and greedy algorithms, to solve a range of real-world problems.
3. Understand the basic principles of approximation algorithms and learn to apply them to a variety of optimization problems.
4. Gain practical experience in implementing and analyzing algorithms through programming assignments and projects.

Course Outcomes:

1. Students will be able to analyze the complexity of an algorithm and make decisions about which algorithms are best suited for a given problem.
2. Students will be able to design algorithms using a variety of techniques, including backtracking, dynamic programming, and greedy algorithms, and apply them to solve problems in various domains.
3. Students will be able to apply approximation algorithms to optimization problems, and understand the tradeoffs between approximation quality and runtime.
4. Students will be able to implement and analyze algorithms using programming languages such as Python and Java, and use this knowledge to develop and evaluate algorithms for real-world applications.

Module 1: Introduction to Algorithms (4 hours)

- Basic concepts review
- Asymptotic analysis
- Solving recurrence relations: recursion tree and Master Theorem

Module 2: Backtracking (4 hours)

- N queens
- Game Trees
- Longest increasing subsequence
- Subset sum
- Optimal binary search trees: analysis

Module 3: Dynamic Programming (6 hours)

- Principle of dynamic programming: memorization or iteration over subproblems
- Longest increasing subsequence
- Optimal binary search trees
- Shortest paths in a graph
- Negative cycles in a graph

Module 4: Greedy Algorithms (3 hours)

- Scheduling classes
- Huffman codes
- Stable matching
- The minimum spanning tree problem
-

Module 5: NP-Hardness (6 hours)

- P vs NP, NP-hard, and NP-complete
- Reduction and SAT
- 3 SAT
- Clique and Vertex Cover
- Graph coloring
-

Module 6: Network Flow (3 hours)

- The maximum flow problem
- Ford-Fulkerson algorithm
- Max flow and Min cut in a Network
- Airline scheduling
-

Module 7: Approximation Algorithm (6 hours)

- Introduction to approximation technique
- Deterministic rounding algorithm
- Rounding a dual solution
- Constructing a dual solution-primal dual method
- Randomized rounding algorithm
-

Module 8: Application of Approximation Algorithm (6 hours)

- Scheduling jobs with deadlines on a single machine
- The k-center problem
- The traveling salesman problem
- Scheduling jobs on identical parallel machines
- Minimizing the sum of completion times on a single machine
-

Module 9: Randomized Algorithm (3 hours)

- Randomization as Algorithm Design Technique
- Randomized min cut algorithm
- Randomized find
- Birthday paradox

TEXTBOOKS :

1. Thomas H Cormen, Charles E Lieserson, Ronald L Rivest and Clifford Stein,Introduction to Algorithms, MIT Press, 2009.
2. Jon Kleinberg and Éva Tardos, Algorithm Design, Pearson, 2005.
3. David P. Williamson and David B. Shmoys, The Design of Approximation Algorithms, Cambridge University Press, 2010.
4. Jeff Erickson, Algorithms, 2019.