| **Course Code** | 18CSC204J | **Course Name** | DESIGN AND ANALYSIS OF ALGORITHMS | **Course Category** | *C* | *Professional Core* | L | T | P | C |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 0 | 2 | 4 |

| **Pre-requisite Courses** | *18CSC201J, 18CSC202J* | | **Co-requisite Courses** | *18CSC207J* | | **Progressive Courses** | *Nil* |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Course Offering Department** | | *Computer Science and Engineering* | | | **Data Book / Codes/Standards** | *Nil* | |

| **Course Learning Rationale (CLR):** | | | *The purpose of learning this course is to:* | |  | **Learning** | | |  | **Program Learning Outcomes (PLO)** | | | | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |
| **CLR-1 :** | *Design efficient algorithms in solving complex real time problems* | | | |  | 1 | 2 | 3 |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| **CLR-2 :** | *Analyze various algorithm design techniques to solve real time problems in polynomial time* | | | |  | Level of Thinking (Bloom) | Expected Proficiency (%) | Expected Attainment (%) |  | Engineering Knowledge | Problem Analysis | Design & Development | Analysis, Design, Research | Modern Tool Usage | Society & Culture | Environment & Sustainability | Ethics | Individual & Team Work | Communication | Project Mgt. & Finance | Life Long Learning | PSO - 1 | PSO - 2 | PSO – 3 |
| **CLR-3 :** | *Utilize various approaches to solve greedy and dynamic algorithms* | | | |  |  |
| **CLR-4 :** | *Utilize back tracking and branch and bound paradigms to solve exponential time problems* | | | |  |  |
| **CLR-5 :** | *Analyze the need of approximation and randomization algorithms, utilize the importance Non polynomial algorithms* | | | |  |  |
| **CLR-6 :** | *Construct algorithms that are efficient in space and time complexities* | | | |  |  |
|  | |  | | |  |  |
| **Course Learning Outcomes (CLO):** | | | | *At the end of this course, learners will be able to:* | |  |
| **CLO-1 :** | *Apply efficient algorithms to reduce space and time complexity of both recurrent and non-recurrent relations* | | | | | *3* | *80* | *70* |  | *L* | *H* | *-* | *H* | *L* | *-* | *-* | *-* | *L* | *L* | *-* | *H* | *-* | *-* | *-* |
| **CLO-2 :** | *Solve problems using divide and conquer approaches* | | | | | *3* | *85* | *75* |  | *M* | *H* | *L* | *M* | *L* | *-* | *-* | *-* | *M* | *L* | *-* | *H* | *-* | *-* | *-* |
| **CLO-3 :** | *Apply greedy and dynamic programming types techniques to solve polynomial time problems.* | | | | | *3* | *75* | *70* |  | *M* | *H* | *M* | *H* | *L* | *-* | *-* | *-* | *M* | *L* | *-* | *H* | *-* | *-* | *-* |
| **CLO-4 :** | *Create exponential problems using backtracking and branch and bound approaches.* | | | | | *3* | *85* | *80* |  | *M* | *H* | *M* | *H* | *L* | *-* | *-* | *-* | *M* | *L* | *-* | *H* | *-* | *-* | *-* |
| **CLO-5 :** | *Interpret various approximation algorithms and interpret solutions to evaluate P type, NP Type, NPC, NP Hard problems* | | | | | *3* | *85* | *75* |  | *H* | *H* | *M* | *H* | *L* | *-* | *-* | *-* | *M* | *L* | *-* | *H* | *-* | *-* | *-* |
| **CLO-6 :** | *Create algorithms that are efficient in space and time complexities by using divide conquer, greedy, backtracking technique* | | | | | *3* | *80* | *70* |  | *L* | *H* | *M* | *H* | *L* | *-* | *-* | *-* | *L* | *L* | *-* | *H* | *-* | *-* | *-* |

| Duration (hour) | | **15** | **15** | **15** | **15** | **15** |
| --- | --- | --- | --- | --- | --- | --- |
| **S-1** | SLO-1 | *Introduction-Algorithm Design* | *Introduction-Divide and Conquer* | *Introduction-Greedy and Dynamic Programming* | *Introduction to backtracking - branch and bound* | *Introduction to randomization and approximation algorithm* |
| SLO-2 | *Fundamentals of Algorithms* | *Maximum Subarray Problem* | *Examples of problems that can be solved by using greedy and dynamic approach* | *N queen’s problem - backtracking* | *Randomized hiring problem* |
| **S-2** | SLO-1 | *Correctness of algorithm* | *Binary Search* | *Huffman coding using greedy approach* | *Sum of subsets using backtracking* | *Randomized quick sort* |
| SLO-2 | *Time complexity analysis* | *Complexity of binary search* | *Comparison of brute force and Huffman method of encoding* | *Complexity calculation of sum of subsets* | *Complexity analysis* |
| **S-3** | SLO-1 | *Insertion sort-Line count, Operation count* | *Merge sort* | *Knapsack problem using greedy approach* | *Graph introduction* | *String matching algorithm* |
| SLO-2 | *Algorithm Design paradigms* | *Time complexity analysis* | *Complexity derivation of knapsack using greedy* | *Hamiltonian circuit - backtracking* | *Examples* |
| **S**  **4-5** | SLO-1 | *Lab 1: Simple Algorithm-Insertion sort* | *Lab 4: Quicksort, Binary search* | *Lab 7: Huffman coding, knapsack and using greedy* | *Lab 10: N queen’s problem* | *Lab 13: Randomized quick sort* |
| SLO-2 |
| **S-6** | SLO-1 | *Designing an algorithm* | *Quick sort and its Time complexity analysis* | *Tree traversals* | *Branch and bound - Knapsack problem* | *Rabin Karp algorithm for string matching* |
| SLO-2 | *And its analysis-Best, Worst and Average case* | *Best case, Worst case, Average case analysis* | *Minimum spanning tree - greedy*  *Kruskal's algorithm - greedy* | *Example and complexity calculation. Differentiate with dynamic and greedy* | *Example discussion* |
| **S-7** | SLO-1 | *Asymptotic notations Based on growth functions.* | *Strassen's Matrix multiplication and its recurrence relation* | *Minimum spanning tree - Prims algorithm* | *Travelling salesman problem using branch and bound* | *Approximation algorithm* |
| SLO-2 | *O,O,Ө, ω, Ω* | *Time complexity analysis of Merge sort* | *Introduction to dynamic programming* | *Travelling salesman problem using branch and bound example* | *Vertex covering* |
| **S-8** | SLO-1 | *Mathematical analysis* | *Largest sub-array sum* | *0/1 knapsack problem* | *Travelling salesman problem using branch and bound example* | *Introduction Complexity classes* |
| SLO-2 | *Induction, Recurrence relations* | *Time complexity analysis of Largest sub-array sum* | *Complexity calculation of knapsack problem* | *Time complexity calculation with an example* | *P type problems* |
| **S**  **9-10** | SLO-1 | *Lab 2: Bubble Sort* | *Lab 5: Strassen Matrix multiplication* | *Lab 8: Various tree traversals, Krukshall’s*  *MST* | *Lab 11: Travelling salesman problem* | *Lab 14: String matching algorithms* |
| SLO-2 |
| **S-11** | SLO-1 | *Solution of recurrence relations* | *Master Theorem Proof* | *Matrix chain multiplication using dynamic programming* | *Graph algorithms* | *Introduction to NP type problems* |
| SLO-2 | *Substitution method* | *Master theorem examples* | *Complexity of matrix chain multiplication* | *Depth first search and Breadth first search* | *Hamiltonian cycle problem* |
| **S-12** | SLO-1 | *Solution of recurrence relations* | *Finding Maximum and Minimum in an array* | *Longest common subsequence using dynamic programming* | *Shortest path introduction* | *NP complete problem introduction* |
| SLO-2 | *Recursion tree* | *Time complexity analysis-Examples* | *Explanation of LCS with an example* | *Floyd-Warshall Introduction* | *Satisfiability problem* |
| **S-13** | SLO-1 | *Solution of recurrence relations* | *Algorithm for finding closest pair problem* | *Optimal binary search tree (OBST)using dynamic programming* | *Floyd-Warshall with sample graph* | *NP hard problems* |
| SLO-2 | *Examples* | *Convex Hull problem* | *Explanation of OBST with an example.* | *Floyd-Warshall complexity* | *Examples* |
| **S**  **14-15** | SLO-1 | *Lab 3: Recurrence Type-Merge sort, Linear search* | *Lab 6: Finding Maximum and Minimum in an array, Convex Hull problem* | *Lab 9: Longest common subsequence* | *Lab 12: BFS and DFS implementation with array* | *Lab 15: Discussion over analyzing a real time problem* |
| SLO-2 |

| **Learning**  **Resources** | 1. *Thomas H Cormen, Charles E Leiserson, Ronald L Revest, Clifford Stein, Introduction to Algorithms, 3rd ed., The MIT Press Cambridge, 2014* 2. *Mark Allen Weiss, Data Structures and Algorithm Analysis in C, 2nd ed., Pearson Education, 2006* | 1. *Ellis Horowitz, Sartajsahni, Sanguthevar, Rajesekaran, Fundamentals of Computer Algorithms, Galgotia Publication, 2010* 2. *S. Sridhar, Design and Analysis of Algorithms, Oxford University Press, 2015* |
| --- | --- | --- |

| **Learning Assessment** | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Bloom’s  Level of Thinking | Continuous Learning Assessment (50% weightage) | | | | | | | | Final Examination (50% weightage) | |
| CLA – 1 (10%) | | CLA – 2 (15%) | | CLA – 3 (15%) | | CLA – 4 (10%)# | |
| Theory | Practice | Theory | Practice | Theory | Practice | Theory | Practice | Theory | Practice |
| Level 1 | Remember | *20%* | *20%* | *15%* | *15%* | *15%* | *15%* | *15%* | *15%* | *15%* | *15%* |
| Understand |
| Level 2 | Apply | *20%* | *20%* | *20%* | *20%* | *20%* | *20%* | *20%* | *20%* | *20%* | *20%* |
| Analyze |
| Level 3 | Evaluate | *10%* | *10%* | *\15%* | *15%* | *15%* | *15%* | *15%* | *15%* | *15%* | *15%* |
| Create |
|  | Total | 100 % | | 100 % | | 100 % | | 100 % | | - | |

# CLA – 4 can be from any combination of these: Assignments, Seminars, Tech Talks, Mini-Projects, Case-Studies, Self-Study, MOOCs, Certifications, Conf. Paper etc.,

| **Course Designers** |  |  |  |  |  |  | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Experts from Industry | | Experts from Higher Technical Institutions | | | | | Internal Experts |
| *1. G. Venkiteswaran, Wipro Technologies, gvenki@pilani.bits-pilani.ac.in* | | *1. Mitesh Khapra, IITM Chennai, miteshk@cse.iitm.ac.in* | | | | | *1. Mr.K.Senthil Kumar, SRMIST* |
| *2. Dr.Sainarayanan Gopalakrishnan, HCL Technologies, sai.jgk@gmail.com* | | *2. V. Masilamani. IIITDM, masila@iiitdm.ac.in* | | | | | *2. Dr.A.Razia Sulthana, SRMIST* |
|  | |  | | | | | *3. Mr. V. Sivakumar, SRMIST* |
|  | |  | | | | | *4. Ms. R. Vidhya, SRMIST* |