# CSE330:COMPETITIVE CODING APPROACHES-TECHNIQUES

L:2 T:0 P:1 Credits:3

**Course Outcomes:** Through this course students should be able to

CO1 :: analyze the time and space complexity of algorithms

CO2 :: apply efficient algorithms for primality testing and number theory problems

CO3 :: apply recursion and backtracking techniques to solve computational problems

CO4 :: analyze and implement dynamic programming techniques for optimization problems

CO5 :: develop optimized solutions for classical dynamic programming problems

CO6 :: evaluate and implement efficient sorting algorithms for problem solving.

**Unit I**

**Behaviour Analysis** : Introduction to limits and behaviour of logic, understanding taxonomy in worst case, analysing the effectiveness and efficiency of algorithms, measuring time and space complexity of algorithm, trade-off concept

**Unit II**

**Primality Testing** : Introduction to Primality Testing, O(sqrt(n)) Algorithm for Primality Testing, Factorization of a number, Finding prime factors by taking the square root, Binary Exponentiation, Fermat method, Sieve of Eratosthenes, Segmented Sieve, Mansi and her series, Collections of Pens, next prime palindrome

**Unit III**

**Recursion and Advanced Techniques** : Introduction to recursion, base condition, Solving problems using recursion, Classic and Modern Approaches, Direct vs. Indirect Recursion, Tailed vs. Non-Tailed Recursion, Memory Allocation in Recursion, Advantages & disadvantages of recursive programming, Backtracking, Permutations, Combination Sum, N-Queens, recursive problems- next happy number, sum string, water overflow

**Unit IV**

**Basic Dynamic Programming** : Introduction to Dynamic Programming, Fibonacci Sequence, Tiling problem, Climbing Stairs, Tabulation vs Memoizatation, State Definition and State Transition, Optimal Substructure Property, Overlapping Subproblems Property, Dynamic Programming Process and Techniques, Formulating Dynamic Programming Problems

**Unit V**

**Dynamic Programming Problems** : Longest increasing subsequence, Longest common subsequence, Binomial coefficient, Box Stacking, Integer Knapsack Problem (Duplicate Items Forbidden), Edit Distance, Matrix Chain Multiplication, Balanced Partition, Longest Increasing Subsequence(LIS), Longest Common Subsequence (LCS), Balanced Partition Problem

**Unit VI**

**Efficient Sorting Algorithms & Analysis** : Introduction to O(n logn) Sorting Algorithms, Iterative & Recursive Merge Sort, Quick Sort, Sorting Elements by Frequency, Finding Minimum Length Sorted Sub-array to Sort an Array, Sorting Strings, case-specific sorting of strings, Count Distinct Pairs with Difference of K

**List of Practicals / Experiments:**

**List of Practical's**

- Finding prime factors by taking the square root
- Fermat method
- Sieve of Eratosthenes
- Segmented Sieve
- Tiling problem
- Longest increasing subsequence
- Longest common subsequence

- Binomial coefficient
- Box Stacking
- Integer Knapsack Problem (Duplicate Items Forbidden)
- Edit Distance
- Matrix Chain Multiplication
- Balanced Partition
- Merge Sort
- Quick Sort
- Counting Sort
- String sorting

**Text Books:**

1. PROGRAMMING PEARLS by JOE BENTLEY, PEARSON

**References:**

1. CRACKING THE CODING INTERVIEW by GAYLE LAAKMANN MCDOWELL, CAREERCUP