|  |  |  |
| --- | --- | --- |
| **Unit** | **Content** | **No. of Lectures** |
| **1** | **1.1 Foundations:** Role of Algorithms in Computing, Analyzing Algorithms, The divide-and-conquer approach, Analyzing divide-and-conquer algorithms, Growth of Functions- Asymptotic notations, Standard notations and common functions, Recurrences- The substitution method, The recursion-tree method, The master method.  **1.2 Sorting and Order Statistics:** Heapsort- Heaps, Maintaining the heap property, Building a heap, The heapsort algorithm, Priority queues, Quicksort- Description of quicksort, Performance of quicksort, A randomized version of quicksort, Analysis of quicksort, Sorting in Linear Time- Lower bounds for sorting, Counting sort, Radix sort, Bucket sort, Medians And Order Statistics- Minimum and maximum, Selection in expected linear time, Selection in worst-case linear time. | **15** |
| **2** | **Advanced Design and Analysis Techniques:**  **2.1 Dynamic Programming:** The General Method, Elements of Dynamic Programming, Longest Common Subsequence, 0/1 Knapsack, Matrix Chain Multiplication, The Subset Sum Problem, Multistage graphs, All pairs shortest paths, The Travelling Salesman Problem.  **2.2 Greedy Algorithms:** The General Method, Elements of greedy strategy, Knapsack Problem, Huffman codes, Job sequencing with deadlines, Optimal merge patterns, Minimum cost spanning tree algorithms, Single Source Shortest Paths.  **2.3 Amortized Analysis:** Aggregate analysis, The accounting method, The potential method, Dynamic tables. | **15** |
| **3** | **3.1 Elementary and Advanced Data Structures:** Stacks and Queues, Linked lists, Maps and Dictionaries, Hash Tables, Binary Trees, Tree Traversal Algorithms, Search Trees- Binary Search Trees, Balanced Search Trees, AVL Trees, Splay Trees, (2, 4) Trees, Red-Black Trees, B-Trees.  **3.2 Maximum Flow:** Flow Networks, The Ford Fulkerson method, Maximum bipartite matching.  **3.3 String Matching:** Naive Pattern Matching Algorithm, KMP Algorithm, Rabin Karp Algorithm. | **15** |
| **4** | **4.1 Number-Theoretic Algorithms:** Elementary number-theoretic notions, Greatest common divisor, Modular arithmetic, Solving modular linear equations, The Chinese remainder theorem, Powers of an element, The RSA public-key cryptosystem.  **4.2 Approximation Algorithms:** The vertex cover problem, The travelling salesman problem, The set covering problem, The subset sum problem.  **4.3 NP Completeness:** Polynomial Time, Polynomial time verification, NP – completeness and reducibility, NP – Complete problems.  **4.4 Linear Programming:** Standard and Slack forms, Formulating problems as linear programs, The simplex algorithm. | **15** |

**References:**

1. T. H. Cormen, C. L. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, MIT Press.
2. Horowitz, Ellis, and Sartaj Sahni, Fundamentals of Computer Algorithms, Rockville, Md: Computer Science Press.
3. Michael Goodrich, Roberto Tamassia, Michael H. Goldwasser, Data Structures and Algorithms in Python, New York: John Wiley & Sons, Inc., [2013].
4. Dr. Basant Agarwal, Benjamin Baka, Hands-On Data Structures and Algorithms with Python: Write complex and powerful code using the latest features of Python 3.7, 2nd Edition (2018)
5. Grokking Algorithms: An illustrated guide for programmers and other curious people, MEAP, Aditya Bhargava