Paper Name: Algorithms and Complexity Theory

Session: March 2022 to June 2022 Semester: 2

Paper Code: CSC-2026 LTP-420

1 I Concepts in algorithm analysis, time and space complexity. 2 I Asymptotic notations (Ω, 0) used for time complexity - I 3 I Asymptotic notations (ω, Ω) used for time complexity - II 4 I Asymptotic notations (θ) used for time complexity - III 5 I Common Mathematical functions monotonicity, floors and ceilings, polynomials. Asymptotic behaviour of polynomials factorials and their relations using asymptotic behaviour. 7 I Iterated logarithmic functions and relations with other functions. 8 I Relational properties of asymptotic notations 9 I Relative asymptotic growth, ordering functions by asymptotic growth rates. 10 I Recurrences - substitution method 11 I Recurrences - using Master theorem 13 I Introduction to the concept of amortized analysis. Use of aggregate method to some simple problems like stack operations. 14 I Use of aggregate method to some simple problems like incrementing binary counter 15 II Algorithm design technique – Divide and Conquer with illustrations - I 16 Algorithm design technique – Divide and Conquer with illustrations - II 17 II Algorithm design technique – Dynamic programming with illustrations - II 18 Algorithm design technique – Dynamic programming with illustrations - II 19 II Algorithm design technique – Greedy Algorithm with illustrations - II 20 Algorithm design technique – Greedy Algorithm with illustrations - II 21 II Algorithm design technique – Back-tracking with illustrations - II 22 II Algorithm design technique – Branch and Bound with illustrations	Day	Unit	Topics
I Asymptotic notations (O, o) used for time complexity - I Asymptotic notations (ω, Ω) used for time complexity - II I Asymptotic notations (θ) used for time complexity - III Common Mathematical functions monotonicity, floors and ceilings, polynomials. Asymptotic behaviour of polynomials Common Mathematical functions exponentials, logarithms, factorials and their relations using asymptotic behaviour. I Iterated logarithmic functions and relations with other functions. Relational properties of asymptotic notations I Relational properties of asymptotic notations Relative asymptotic growth, ordering functions by asymptotic growth, ordering functions by asymptotic growth rates. Recurrences - substitution method Recurrences - iteration method (using recursion tree). Recurrences - using Master theorem I Introduction to the concept of amortized analysis. Use of aggregate method to some simple problems like stack operations. I Use of aggregate method to some simple problems like incrementing binary counter Algorithm design technique – Divide and Conquer with illustrations - II Algorithm design technique – Dynamic programming with illustrations - II Algorithm design technique – Dynamic programming with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II	1	I	
1	2	T	
4 I Asymptotic notations (0) used for time complexity - III 5 I Common Mathematical functions monotonicity, floors and ceilings, polynomials. Asymptotic behaviour of polynomials 6 I Common Mathematical functions exponentials, logarithms, factorials and their relations using asymptotic behaviour. 7 I Iterated logarithmic functions and relations with other functions. 8 I Relational properties of asymptotic notations 9 I Relative asymptotic growth, ordering functions by asymptotic growth rates. 10 I Recurrences - substitution method 11 I Recurrences - using Master theorem 12 I Recurrences - using Master theorem 13 I Introduction to the concept of amortized analysis. Use of aggregate method to some simple problems like stack operations. 14 I Use of aggregate method to some simple problems like incrementing binary counter 15 II Algorithm design technique – Divide and Conquer with illustrations - I 16 Algorithm design technique – Divide and Conquer with illustrations - II 17 II Algorithm design technique – Dynamic programming with illustrations - II 18 Algorithm design technique – Dynamic programming with illustrations - II 19 II Algorithm design technique – Greedy Algorithm with illustrations - II 20 Algorithm design technique – Greedy Algorithm with illustrations - II 21 II Algorithm design technique – Greedy Algorithm with illustrations - II 22 II Algorithm design technique – Back-tracking with illustrations			
Solution Solution	3	1	Asymptotic notations (ω, Ω) used for time complexity - Π
floors and ceilings, polynomials. Asymptotic behaviour of polynomials Common Mathematical functions exponentials, logarithms, factorials and their relations using asymptotic behaviour. Ilerated logarithmic functions and relations with other functions. Relational properties of asymptotic notations Relative asymptotic growth, ordering functions by asymptotic growth rates. Recurrences - substitution method Recurrences - iteration method (using recursion tree). Recurrences - using Master theorem Introduction to the concept of amortized analysis. Use of aggregate method to some simple problems like stack operations. Introduction to the concept of amortized analysis. Use of aggregate method to some simple problems like incrementing binary counter International Algorithm design technique – Divide and Conquer with illustrations - I Algorithm design technique – Divide and Conquer with illustrations - II Algorithm design technique – Dynamic programming with illustrations - II Algorithm design technique – Dynamic programming with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Back-tracking with illustrations	4	I	
Common Mathematical functions exponentials, logarithms, factorials and their relations using asymptotic behaviour. I	5	I	• • • • • • • • • • • • • • • • • • • •
Relational properties of asymptotic notations I Relative asymptotic growth, ordering functions by asymptotic growth rates. Recurrences - substitution method Recurrences - iteration method (using recursion tree). Recurrences - using Master theorem I Introduction to the concept of amortized analysis. Use of aggregate method to some simple problems like stack operations. I Use of aggregate method to some simple problems like incrementing binary counter I Algorithm design technique – Divide and Conquer with illustrations - I Algorithm design technique – Divide and Conquer with illustrations - II Algorithm design technique – Dynamic programming with illustrations - II Algorithm design technique – Dynamic programming with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Back-tracking with illustrations Algorithm design technique – Back-tracking with illustrations	6	I	Common Mathematical functions exponentials, logarithms,
notations Relative asymptotic growth, ordering functions by asymptotic growth rates.	7	I	Iterated logarithmic functions and relations with other functions.
9 I Relative asymptotic growth, ordering functions by asymptotic growth rates. 10 I Recurrences - substitution method 11 I Recurrences - iteration method (using recursion tree). 12 I Recurrences - using Master theorem 13 I Introduction to the concept of amortized analysis. Use of aggregate method to some simple problems like stack operations. 14 I Use of aggregate method to some simple problems like incrementing binary counter 15 II Algorithm design technique – Divide and Conquer with illustrations - I 16 Algorithm design technique – Divide and Conquer with illustrations - II 17 II Algorithm design technique – Dynamic programming with illustrations - I 18 Algorithm design technique – Dynamic programming with illustrations - II 19 II Algorithm design technique – Greedy Algorithm with illustrations - I 20 Algorithm design technique – Greedy Algorithm with illustrations - II 21 II Algorithm design technique – Back-tracking with illustrations	8	I	
11 I Recurrences – iteration method (using recursion tree). 12 I Recurrences – using Master theorem 13 I Introduction to the concept of amortized analysis. Use of aggregate method to some simple problems like stack operations. 14 I Use of aggregate method to some simple problems like incrementing binary counter 15 II Algorithm design technique – Divide and Conquer with illustrations - I 16 Algorithm design technique – Divide and Conquer with illustrations - II 17 II Algorithm design technique – Dynamic programming with illustrations - II 18 Algorithm design technique – Dynamic programming with illustrations - II 19 II Algorithm design technique – Greedy Algorithm with illustrations - II 20 Algorithm design technique – Greedy Algorithm with illustrations - II 21 II Algorithm design technique – Back-tracking with illustrations 22 II Algorithm design technique - Branch and Bound with illustrations	9	Ι	Relative asymptotic growth, ordering
I Recurrences - using Master theorem I Introduction to the concept of amortized analysis. Use of aggregate method to some simple problems like stack operations. I Use of aggregate method to some simple problems like incrementing binary counter I Algorithm design technique – Divide and Conquer with illustrations - I Algorithm design technique – Divide and Conquer with illustrations - II Algorithm design technique – Dynamic programming with illustrations - II Algorithm design technique – Dynamic programming with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Greedy Algorithm with illustrations - II Algorithm design technique – Back-tracking with illustrations Algorithm design technique - Back-tracking with illustrations	10	I	Recurrences - substitution method
Introduction to the concept of amortized analysis. Use of aggregate method to some simple problems like stack operations. 14 I Use of aggregate method to some simple problems like incrementing binary counter 15 II Algorithm design technique – Divide and Conquer with illustrations - I 16 Algorithm design technique – Divide and Conquer with illustrations - II 17 II Algorithm design technique – Dynamic programming with illustrations - I 18 Algorithm design technique – Dynamic programming with illustrations - II 19 II Algorithm design technique – Greedy Algorithm with illustrations - II 20 Algorithm design technique – Greedy Algorithm with illustrations - II 21 II Algorithm design technique – Back-tracking with illustrations 22 II Algorithm design technique - Branch and Bound with illustrations	11	I	Recurrences – iteration method (using recursion tree).
analysis. Use of aggregate method to some simple problems like stack operations. 14 I Use of aggregate method to some simple problems like incrementing binary counter 15 II Algorithm design technique – Divide and Conquer with illustrations - I 16 Algorithm design technique – Divide and Conquer with illustrations - II 17 II Algorithm design technique – Dynamic programming with illustrations - I 18 Algorithm design technique – Dynamic programming with illustrations - II 19 II Algorithm design technique – Greedy Algorithm with illustrations - I 20 Algorithm design technique – Greedy Algorithm with illustrations - II 21 II Algorithm design technique - Back-tracking with illustrations 22 II Algorithm design technique - Branch and Bound with illustrations	12	I	Recurrences - using Master theorem
14 I Use of aggregate method to some simple problems like incrementing binary counter 15 II Algorithm design technique – Divide and Conquer with illustrations - I 16 Algorithm design technique – Divide and Conquer with illustrations - II 17 II Algorithm design technique – Dynamic programming with illustrations - I 18 Algorithm design technique – Dynamic programming with illustrations - II 19 II Algorithm design technique – Greedy Algorithm with illustrations - I 20 Algorithm design technique – Greedy Algorithm with illustrations - II 21 II Algorithm design technique - Back-tracking with illustrations 22 II Algorithm design technique - Branch and Bound with illustrations	13	I	analysis. Use of aggregate method to some
15 II Algorithm design technique – Divide and Conquer with illustrations - I 16 Algorithm design technique – Divide and Conquer with illustrations - II 17 II Algorithm design technique – Dynamic programming with illustrations - I 18 Algorithm design technique – Dynamic programming with illustrations - II 19 II Algorithm design technique – Greedy Algorithm with illustrations - I 20 Algorithm design technique – Greedy Algorithm with illustrations - II 21 II Algorithm design technique - Back-tracking with illustrations 22 II Algorithm design technique - Branch and Bound with illustrations	14	I	Use of aggregate method to some
17 II Algorithm design technique – Dynamic programming with illustrations - I 18 Algorithm design technique – Dynamic programming with illustrations - II 19 II Algorithm design technique – Greedy Algorithm with illustrations - I 20 Algorithm design technique – Greedy Algorithm with illustrations - II 21 Algorithm design technique - Back-tracking with illustrations 22 Algorithm design technique - Branch and Bound with illustrations	15	II	
Algorithm design technique – Dynamic programming with illustrations - II 19 II Algorithm design technique – Greedy Algorithm with illustrations - I 20 Algorithm design technique – Greedy Algorithm with illustrations - II 21 Algorithm design technique - Back-tracking with illustrations 22 Algorithm design technique - Branch and Bound with illustrations	16		Algorithm design technique – Divide and Conquer with illustrations - II
19 II Algorithm design technique – Greedy Algorithm with illustrations - I 20 Algorithm design technique – Greedy Algorithm with illustrations - II 21 Algorithm design technique - Back-tracking with illustrations 22 Algorithm design technique - Branch and Bound with illustrations	17	II	Algorithm design technique – Dynamic programming with illustrations - I
20 Algorithm design technique – Greedy Algorithm with illustrations - II 21 Algorithm design technique - Back-tracking with illustrations 22 Algorithm design technique - Branch and Bound with illustrations	18		Algorithm design technique – Dynamic programming with illustrations - II
21 Algorithm design technique - Back-tracking with illustrations 22 II Algorithm design technique - Branch and Bound with illustrations	19	II	
22 II Algorithm design technique - Branch and Bound with illustrations	20		
	21	II	Algorithm design technique - Back-tracking with illustrations
23 II Dynamic storage allocation, garbage collection.	22	II	Algorithm design technique - Branch and Bound with illustrations
	23	П	Dynamic storage allocation, garbage collection.

24 III 25 III 26 III 27 III 28 III 30 III 31 III 32 III 33 III 35 III 36 IV 37 IV 38 39 IV 40 40 40 40 40 40 40 4	Breadth-first search with illustrations, properties, algorithm and complexity-II Topological sort with illustrations, algorithm and complexity. Minimum spanning tree – Definitions and Theorems. Minimum spanning tree – Kruskal's algorithm Minimum spanning tree – Prim's algorithm Minimum spanning tree – Analysis and Comparison of Kruskal's and Prim's algorithm Single source shortest path problem and algorithm due to Dijkstra
26 27 28 30 31 32 33 34 35 36 V 37 V 38 39 V 40	Depth-first search with illustrations, properties, algorithm and complexity - I Depth-first search with illustrations, properties, algorithm and complexity - II Breadth-first search with illustrations, properties, algorithm and complexity-I Breadth-first search with illustrations, properties, algorithm and complexity-II Topological sort with illustrations, algorithm and complexity. Minimum spanning tree – Definitions and Theorems. Minimum spanning tree – Kruskal's algorithm Minimum spanning tree – Prim's algorithm Minimum spanning tree – Analysis and Comparison of Kruskal's and Prim's algorithm . Single source shortest path problem and algorithm due to Dijkstra
27 28 29 30 31 32 33 34 35 36 V 37 V 38 39 V 40	Breadth-first search with illustrations, properties, algorithm and complexity-I Breadth-first search with illustrations, properties, algorithm and complexity-II Topological sort with illustrations, algorithm and complexity. Minimum spanning tree – Definitions and Theorems. Minimum spanning tree – Kruskal's algorithm Minimum spanning tree – Prim's algorithm Minimum spanning tree – Analysis and Comparison of Kruskal's and Prim's algorithm Single source shortest path problem and algorithm due to Dijkstra
28 29 30 31 32 33 34 35 36 V 37 V 38 39 V 40	Breadth-first search with illustrations, properties, algorithm and complexity-II Topological sort with illustrations, algorithm and complexity. Minimum spanning tree – Definitions and Theorems. Minimum spanning tree – Kruskal's algorithm Minimum spanning tree – Prim's algorithm Minimum spanning tree – Analysis and Comparison of Kruskal's and Prim's algorithm Single source shortest path problem and algorithm due to Dijkstra
29 III 30 III 31 III 32 III 33 III 34 III 35 III 36 IV 37 IV 38 39 IV 40	Topological sort with illustrations, algorithm and complexity. Minimum spanning tree – Definitions and Theorems. Minimum spanning tree – Kruskal's algorithm Minimum spanning tree – Prim's algorithm Minimum spanning tree – Analysis and Comparison of Kruskal's and Prim's algorithm Single source shortest path problem and algorithm due to Dijkstra
30 III 31 III 32 III 33 III 34 III 35 III 36 IV 37 IV 38 39 IV 40	Minimum spanning tree – Definitions and Theorems. Minimum spanning tree – Kruskal's algorithm Minimum spanning tree – Prim's algorithm Minimum spanning tree – Analysis and Comparison of Kruskal's and Prim's algorithm Single source shortest path problem and algorithm due to Dijkstra
31 III 32 III 33 III 34 III 35 III 36 IV 37 IV 38 39 IV 40	Minimum spanning tree – Kruskal's algorithm Minimum spanning tree – Prim's algorithm Minimum spanning tree – Analysis and Comparison of Kruskal's and Prim's algorithm . Single source shortest path problem and algorithm due to Dijkstra
32 III 33 III 34 III 35 III 36 IV 37 IV 38 39 IV 40	Minimum spanning tree – Prim's algorithm Minimum spanning tree – Analysis and Comparison of Kruskal's and Prim's algorithm Single source shortest path problem and algorithm due to Dijkstra
33 III 34 III 35 III 36 IV 37 IV 38 39 IV 40	Minimum spanning tree –Analysis and Comparison of Kruskal's and Prim's algorithm . Single source shortest path problem and algorithm due to Dijkstra
34 III 35 III 36 IV 37 IV 38 39 IV 40	Prim's algorithm . Single source shortest path problem and algorithm due to Dijkstra
35 III 36 IV 37 IV 38 39 IV 40	
36 IV 37 IV 38 39 IV 40	
37 IV 38 39 IV 40	Algorithm due to Dijkstra (continued) and analysis
38 39 IV 40	Decision Problem and Formal language framework
39 IV 40	Complexity classes – P, NP. co-NP - I
40	Complexity classes – P, NP. co-NP - II
	Reducibility and NP-Completeness, NP-Hard
	Theorems related to NP-Completeness - I
41 IV	Theorems related to NP-Completeness - II
42 V	Computing lower bounds for sorting
43 V	Computing lower bounds for merging.
44 V	Computing lower bounds for finding maximum and second maximum
45	Basic idea about neural network.
46	Neural network
47	Basic idea of genetic algorithm
48	Genetic algorithm