JEPPIAAR
ENGINEERING COLLEGE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CS8451 DESIGN AND ANALYSIS OF ALGORITHMS

QUESTION BANK

II YEAR A & B / BATCH: 2017 -2021

Vision of Institution: To build Jeppiaar Engineering College as an Institution of Academic Excellence in Technical education and Management education and to become a World Class University.

Mission of Institution

M1	To excel in teaching and **learning, research and innovation** by promoting the principles of scientific analysis and creative thinking
M2	To participate in the production, **development and dissemination of knowledge** and interact with **national and international communities**
M3	To equip students with values, ethics and life skills needed to enrich their lives and enable them to meaningfully contribute to the progress of society
M4	To prepare students for higher studies and lifelong learning, enrich them with the practical and entrepreneurial skills necessary to excel as future professionals and contribute to Nation's economy

PROGRAM OUTCOMES (POs)

PO1	**Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of computer science engineering problems.
PO2	**Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	**Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	**Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	**Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6	**The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO7	**Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO8	**Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO9	**Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	**Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO11	**Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	**Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Vision of Department: To emerge as a globally prominent department, developing ethical computer professionals, innovators and entrepreneurs with academic excellence through quality education and research**.**

Mission of Department

M1	To create **computer professionals** with an ability to identify and **formulate the engineering problems** and also to provide **innovative solutions** through **effective teaching learning process.**
M2	To **strengthen the core-competence** in computer science and engineering and to create an ability to **interact** effectively with industries.
M3	To produce engineers with good professional skills, **ethical values** and life skills for the **betterment of the society.**
M4	To encourage students towards **continuous and higher level learning** on technological advancements and provide a platform for **employment and self-employment.**

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

PEO 01: To address the real time complex engineering problems using innovative approach with strong core computing skills.

PEO 02: To apply core-analytical knowledge and appropriate techniques and provide solutions to real time challenges of national and global society.

PEO 03: Apply ethical knowledge for professional excellence and leadership for the betterment of the society**.**

PEO 04: Develop life-long learning skills needed for better employment and entrepreneurship.

PROGRAMME SPECIFIC OUTCOME (PSOs)

PSO1 – An ability to understand the core concepts of computer science and engineering and to enrich problem solving skills to analyze, design and implement software and hardware based systems of varying complexity.

PSO2 - To interpret real-time problems with analytical skills and to arrive at cost effective and optimal solution using advanced tools and techniques.

PSO3 - An understanding of social awareness and professional ethics with practical proficiency in the broad area of programming concepts by lifelong learning to inculcate employment and entrepreneurship skills.

Course Outcomes (COs)

C213.1	**Design** algorithms for various computing problems and time and space complexity analysis.
C213.2	**Analyze** the Brute Force and Divide and Conquer algorithm design techniques for a given problem.
C213.3	**Examine** the Dynamic and Greedy algorithm design techniques for a given problem
C213.4	**Discuss** the Iterative improvement algorithm design techniques for a given problem
C213.5	**Solve** problems using Backtracking and Branch and Bound Techniques and **Modify** the existing algorithms to improve efficiency.

CS8451 DESIGN AND ANALYSIS OF ALGORITHMS T P C 3 0 0 3

OBJECTIVES:

- To understand and apply the algorithm analysis techniques.
- To critically analyze the efficiency of alternative algorithmic solutions for the same problem
- To understand different algorithm design techniques.
- To understand the limitations of Algorithmic power.

UNIT I INTRODUCTION 9

Notion of an Algorithm – Fundamentals of Algorithmic Problem Solving – Important Problem Types-Fundamentals of the Analysis of Algorithmic Efficiency –Asymptotic Notations and their properties. Analysis Framework – Empirical analysis - Mathematical analysis for Recursive and Non-recursive algorithms - Visualization

UNIT II BRUTE FORCE AND DIVIDE-AND-CONQUER 9

Brute Force – Computing a^n – String Matching - Closest-Pair and Convex-Hull Problems - Exhaustive Search - Travelling Salesman Problem - knapsack Problem - Assignment problem.Divide and Conquer Methodology – Binary Search – Merge sort – Quick sort – Heap Sort - Multiplication of Large Integers – Closest-Pair and Convex - Hull Problems.

UNIT III DYNAMIC PROGRAMMING AND GREEDY TECHNIQUE 9

Dynamic programming – Principle of optimality - Coin changing problem, Computing a Binomial Coefficient – Floyd‘s algorithm – Multi stage graph - Optimal Binary Search Trees – knapsack Problem and Memory functions. Greedy Technique – Container loading problem - Prim‘s algorithm and kruskal's Algorithm – 0/1 knapsack problem, Optimal Merge pattern - Huffman Trees.

UNIT IV ITERATIVE IMPROVEMENT 9

The Simplex Method - The Maximum-Flow Problem – Maximum Matching in Bipartite Graphs, Stable marriage Problem.

UNIT V COPING WITH THE LIMITATIONS OF ALGORITHM POWER 9

Lower - Bound Arguments - P, NP NP- Complete and NP Hard Problems. Backtracking – n-Queen problem - Hamiltonian Circuit Problem – Subset Sum Problem. Branch and Bound – LIFO Search and FIFO search - Assignment problem – Knapsack Problem – Travelling

www.padeepz.net

Salesman Problem - Approximation Algorithms for NP-Hard Problems – Travelling Salesman problem – Knapsack problem.

TOTAL: 45 PERIODS

TEXT BOOBTLS:

1. Anany Levitin, -Introduction to the Design and Analysis of Algorithms‖, Third Edition, Pearson Education, 2012.
2. Ellis Horowitz, Sartaj Sahni and Sanguthevar Rajasekaran, Computer Algorithms/ C++, Second Edition, Universities Press, 2007.

REFERENCES:

1. Thomas H.Cormen, Charles E.Leiserson, Ronald L. Rivest and Clifford Stein, -Introduction to Algorithms‖, Third Edition, PHI Learning Private Limited, 2012.
2. Alfred V. Aho, John E. Hopcroft and Jeffrey D. Ullman, -Data Structures and Algorithms‖, Pearson Education, Reprint 2006.
3. Harsh Bhasin, -Algorithms Design and Analysis‖, Oxford university press, 2015.
4. http://nptel.ac.in/

UNIT I

Q. No.	Questions	CO	Bloom's Level
1	**Define Algorithm. [MAY/JUNE 2013] APRIL/MAY 2017,NOV/DEC 2018** An algorithm is a sequence of unambiguous instructions for solving a problem in a finite amount of time	C213.1	BTL1
2	**Compare Time Efficiency and Space Efficiency? [APRIL/MAY 2010]** Time Efficiency measured by counting the number of times the algorithms basic operation is executed. Space Efficiency is measured by counting the number of extra memory units consumed by the algorithm	C213.1	BTL 2
3	**What is Big 'Oh' Notation? [MAY/JUNE 2013, MAY/JUNE 2012]** The Big 'Oh' notation provides an upper bound for the function t.A function t(n) is said to be in O(g(n)), if there exist some positive constant C and some non negative number No, such that , t(n)<=Cg(n), for all n>=no	C213.1	BTL 1
4	**Analyze the time complexity using the step count method when 2 m X n matrices are added. [APRIL / MAY 2011]** Time complexity = Θ(mn)	C213.1	BTL 4
5	**What is a recurrence equation? [APRIL/MAY 2010]** An equality or inequality describing the function in terms of its behavior on smaller inputs T(n) = T(n –1) + c; T(1) = 1.	C213.1	BTL 1

6	**An array has exactly n nodes. They are filled from the set {0,1,2,....,n-1,n}. There are no duplicates in the list. Design an O(n) worst case time algorithm to find which one of the elements from the above set is missing in the array. [APRIL / MAY 2011]**
	int linearsearch(int a[], int size, int ch)
	{
	for(int i=0;i<size;i++)
	{
	if(a[i]==ch)
	return(i);
	}
	return(-1);
	}
		C213.1	BTL 6

7	**If** $f(n)=a_m n^m + \dots + a_0$, **then prove that** $f(n)=O(n^m)$ **[NOV/ DEC 2010]** $f(n) \le \sum_{i=0}^{m} \lvert a_i \rvert n^i$ $\le n^m \sum_{i=0}^{m} \lvert a_i \rvert n^{i-m}$ $\le n^m \sum_{i=0}^{m} \lvert a_i \rvert$ for $n \ge 1$ $f(n)=O(n^m)$ assuming that m is constant	C213.1	BTL BTL 5
8	**What is the properties of Asymptotic notation[NOV / DEC 2011,MAY/JUNE 2015]** a. [illegible] b. Any function can be said as an order of itself. c. Any constant value is equivalent to O(1).	C213.1	BTL 1
9	**What is meant by linear search? [NOV/DEC 2011, MAY/JUNE 2012]** Linear search or sequential search is a method for finding a particular value in a list that consists of checking every one of its elements, one at a time and in sequence, until the desired one is found. Best case – O(1) Worst case –O(n) Average case – O(n)	C213.1	BTL 1

10	**Develop an algorithm to find the number of binary digits in the binary representation of a positive decimal number. [MAY/JUNE 2015]**
	ALGORITHM *Binary(n)*
	//Input: A positive decimal integer *n*
	//Output: The number of binary digits in *n*'s binary representation
	count ← 1
	while *n* > 1 **do**
	count ← *count* + 1
	n ← ⌊*n*/2⌋
	return *count*
		C213.1	BTL
			BTL6

	What is meant by Notion of Algorithm. Problem ↓ Algorithm ↓	C213.1	BTL 1

11	Input ⟶ Computer ⟶ Output		
12	**What are the 2 Kinds of Algorithm Efficiency** Time Efficiency-How fast your algorithm runs? Space Efficiency-How much extra memory your algorithm needs?	C213.1	BTL 1
13	**What is Pseudo Code?** Pseudo Code is a mixture of Natural Language and Programming Language Constructs such as functions, loops, decision making statements..etc	C213.1	BTL 1
14	**Show the Euclid Algorithm MAY/JUNE 2016,APR/MAY 2018** Algorithm Euclid(m,n) Step 1: While n not equal do Step 2: r = m mod r Step 3: m=n Step 4: n=r Step 5: return n	C213.1	BTL 1
15	**What are the different types of time complexity?** The time complexity can be classified into 3 types, they are • Worst case analysis • Average case analysis • Best case analysis	C213.1	BTL 1

16	**What is recursive algorithm? Write an algorithm using Recursive function to fine sum of n numbers,**
	An algorithm is said to be recursive if the same algorithm is invoked in the body. An algorithm that calls itself is Direct recursive. Algorithm A is said to be indeed recursive if it calls another algorithm, which in turn calls A.
	algorithm using Recursive function to fine sum of n numbers,
	Algorithm Rsum (a,n)
	{
	If(n≤ 0) then
	Return 0.0;
	Else Return Rsum(a, n- 1) + a(n);}
		C213.1	BTL 1

17	**Classify Algorithm Design and Analysis of Process.** • Understand the problem ▪ Decide on computational means ▪ Exact Vs Approximate Algorithms Data Structures ▪ Algorithm Design techniques ▪ Design an algorithms ▪ Prove Correctness ▪ Analyze the Algorithm	C213.1	BTL 4

	▪ Code the algorithm		
18	**How can you specify Algorithms?** Algorithms can be specified in a natural language or pseudo code.	C213.1	BTL1
19	**List the important Problem Types?** ▪ Sorting ▪ Searching ▪ String Processing ▪ Graph Problem ▪ Combinatorial Problem ▪ Geometric Problem ▪ Numerical Problem	C213.1	BTL4
20	**What is amortized efficiency?** In some situations a single operation can be expensive, but the total time for the entire sequence of n such operations is always significantly better that the worst case efficiency of that single operation multiplied by n. this is called amortized efficiency.	C213.1	BTL1
21	**What is Sorting Problem?** Sorting algorithm is rearranging the items of a given list in descending/ascending order. Sorting algorithms classified into Stable Sorting Algorithm Non-Stable Algorithm	C213.1	BTL1
22	**What is Searching Problem?** Finding a given value, called search key in a given set. Searching Algorithms needs more memory space and sorted array.	C213.1	BTL1
23	**What is Graph Problem?** Graph is a collection of edges and vertices. G=(V,E). For eg. Traversal Algorithms, Shortest Path Algorithm, Graph Coloring Problem	C213.1	BTL1
24	**What is Combinatorial Problem?** This problem that ask to find a combinatorial object such as permutations, combinations or a subset. Combinatorial problems are most difficult to solve. For eg Traveling sales man problem	C213.1	BTL1
25	**List the features of efficient algorithm?** ▪ Free of ambiguity ▪ Efficient in execution time ▪ Concise and compact Completeness ▪ Definiteness Finiteness	C213.1	BTL4

26	**Define Order of Algorithm.** The order of algorithm is a standard notation of an algorithm that has been developed to represent function that bound the computing time for algorithms. The order of an algorithm is a way of defining its efficiency. It is usually referred as O-notation	C213.1	BTL1
27	**Illustrate the different criteria used to improve the effectiveness of algorithm?** Input- Zero or more quantities are externally supplied Output-At least one quantity is produced Definiteness-Each instruction is clear and unambiguous Finiteness-If we trace out the instructions of an algorithm, then for all case the algorithm terminates after a finite number of steps Effectiveness-Every instruction must be very clear	C213.1	BTL2
29	**What is the substitution method?** One of the methods for solving any such recurrence relation is called the substitution method. Types of Substitution : 1. Forward Substitution **2.** Backward Substitution	C213.1	BTL1
30	**Define the asymptotic t\notation "theta" (θ)** The function $f(n) = \theta(g(n))$ if there exist positive constant C_1, C_2, and no such that $C_1\, g(n) \leq f(n) \leq C_2\, g(n)$ for all n, $n \geq n_0$.	C213.1	BTL1
31	**What is a basic operation? APR/MAY 2018** A basic operation is one that best characterizes the efficiency of the particular algorithm of interest **For time analysis it is the operation that we expect to have the most influence on the algorithm's total running time:** - key comparisons in a searching algorithm - Numeric multiplications in a matrix multiplication algorithm - Visits to nodes (or arcs) in a graph traversal algorithm **For space analysis it is an operation that increases memory usage** - A procedure call that adds a new frame to the run-time stack - Creation of a new object or data structure in the run-time heap The basic operation may occur in more than one place in the algorithm	C213.1	BTL1
32	**What is performance measurement?** Performance measurement is concerned with obtaining the space and the time requirements of a particular algorithm.	C213.1	BTL1

<table>
<tr><td>33</td><td>List the desirable properties of algorithm.NOV/DEC 2018
The characteristics of a good algorithm are:
Precision – the steps are precisely stated(defined).
Uniqueness – results of each step are uniquely defined and only depend on the input and the result of the preceding steps.
Finiteness – the algorithm stops after a finite number of instructions are executed.
Input – the algorithm receives input.
Output – the algorithm produces output.
Generality – the algorithm applies to a set of inputs.</td><td>C213.1</td><td>BTL1</td></tr>
<tr><td>34</td><td>Give the two major phases of performance evaluation
Performance evaluation can be loosely divided into two major phases: (i) a prior estimates (performance analysis) (ii) a Posterior testing(performance measurement)</td><td>C213.1</td><td>BTL1</td></tr>
<tr><td>35</td><td>Define input size.
The input size of any instance of a problem is defined to be the number of words(or the number of elements) needed to describe that instance.</td><td>C213.1</td><td>BTL1</td></tr>
<tr><td>36</td><td>Define best-case step count.
The best-case step count is the minimum number of steps that can be executed for the given parameters.</td><td>C213.1</td><td>BTL1</td></tr>
<tr><td>37</td><td>Define worst-case step count.
The worst-case step count is the maximum number of steps that can be executed for the given parameters.</td><td>C213.1</td><td>BTL1</td></tr>
<tr><td>38</td><td>Define average step count.
The average step count is the average number of steps executed an instances with the given parameters.</td><td>C213.1</td><td>BTL1</td></tr>
<tr><td>39</td><td>Define best ,worst, average case time complexity.NOV/DEC2018
Best, worst, and average cases of a given algorithm express what the resource usage is at least, at most and on average, respectively. Usually the resource being considered is running time, i.e. time complexity, but it could also be memory or other resource. Best case is the function which performs the minimum number of steps on input data of n elements. Worst case is the function which performs the maximum number of steps on input data of size n. Average case is the function which performs an average number of steps on input data of n elements.</td><td>C213.1</td><td>BTL1</td></tr>
<tr><td>40</td><td>Differentiate: Mathematical and Empirical analysis.
<table><tr><th>Mathematical analysis.</th><th>Empirical analysis</th></tr><tr><td>The algorithm is analyzed with the help of mathematical derivations and there is no need of</td><td>The algorithm is analyzed by taking some sample of input and no mathematical derivation is involved</td></tr></table></td><td>C213.1</td><td>BTL1</td></tr>
</table>

	specific input		
	The principal weakness is limited applicability	The principal strength is it is applicable for any algorithm	
	The principal strength is it is independent of any input	The principal weakness is it depends upon the sample input	
41	**What is algorithm visualization?** Algorithm visualization can be defines as the use of images to convey some useful information about algorithms. Two principal variations are Static algorithm visualization Dynamic Algorithm visualization(also called algorithm animation)	C213.1	BTL1
42	**What is correctness of algorithm?** The algorithm's correctness is ascertained, if the algorithm yields the required results for every legitimate input in a finite amount of time.	C213.1	BTL1
43	**How can you Classify Algorithms** Among several ways to classify algorithms, the 2 principal alternatives are • To group algorithms according to types of problem they solve com • To group algorithms according to underlying design techniques they are based upon	C213.1	BTL1
44	**What are fundamental data structures?** Linear data structures – Linked lists,stacks,queues Graphs Trees Sets and dictionaries	C213.1	BTL1
45	**What is a Abstract Data type?** It is a set of abstract objects with a collection of operations that can be performed on them	C213.1	BTL1
46	**List 5 of basic efficiency classes**. log n logarithmic n linear nlogn n-log-n n2 quadratic 2n exponential	C213.1	BTL1
47	**What is the formula used to calculate the algorithm's running time?** The running time T(n) of a program implementing the algorithm on a computer is given by the formula : T(n) = Cop x C(n) where Cop is the time of execution of an algorithm's basic operations C(n) is the the number of times the basic operation is executed	C213.1	BTL1
48	**What is the order of growth?** The Order of growth is the scheme for analyzing an algorithm's efficiency for different input sizes which ignores the multiplicative constant used in calculating the algorithm's running time. Measuring the performance of an algorithm in relation with the input size n is called the order of growth.	C213.1	BTL1

49	**Write general plan for analyzing non-recursive algorithms**. i. Decide on parameter indicating an input's size. **ii.** Identify the algorithm's basic operation **iii.** . Cheking the no.of times basic operation executed depends on size of input.if it depends on some additional property,then best,worst,avg.cases need to be investigated **iv.** . Set up sum expressing the no.of times the basic operation is executed. (establishing order of growth)	C213.1	BTL1
50	**Write general plan for analyzing recursive algorithms.** i. Decide on parameter indicating an input's size. **ii.** . Identify the algorithm's basic operation **iii.** Cheking the no.of times basic operation executed depends on size of input.if it depends on some additional property,then best,worst,avg.cases need to be investigated **iv.** Set up the recurrence relation,with an appropriate initial condition,for the number of times the basic operation is executed **v.** Solve recurrence (establishing order of growth)	C213.1	BTL1
51	**What is a scatter plot?** Graphical representation of empirical data obtained as the result of an experiment is called a scatter plot.	C213.1	BTL1
52	**What is the principal alternative to the mathematical analysis of algorithm's efficiency?** Empirical analysis It is done by running the algorithm on the sample inputs and recording the data observed. Then the data is analyzed and a scatter plot is prepared.	C213.1	BTL1
53	**What is the possible application of empirical analysis?** One of the possibilities of the empirical analysis is to attempt predicting the algorithm's performance on the sample size not included in the experiment's sample.	C213.1	BTL1

PART-B

Q. No.	Questions	CO	Bloom's Level
1.	**Explain fundamentals of Algorithmic problem solving?** Refer page no 33 in Anany Levitin	C213.1	BTL5
2.	**Explain important problem types.** Refer page no 43 in Anany Levitin.	C213.1	BTL5
3.	**Elaborate on Asymptotic Notations .MAY/JUNE 2016, APRIL / MAY 2017, NOV/DEC 2017,APR/MAY 2018**	C213.1	BTL6

	Refer page no 76 in Anany Levitin.		
4.	**Explain mathematical Analysis of Non recursive Algorithm with examples .<u>APRIL/MAY 2017</u>** Refer page no 85 in Anany Levitin	C213.1	BTL5
5.	**Explain mathematical Analysis of Recursive Algorithm with examples.<u>APRIL / MAY 2017</u> ,NOV/DEC 2017, APR/MAY 2018 ,NOV/dec 2018** Refer page no 93 in Anany Levitin	C213.1	BTL5
6.	**Write the Asymptotic notations used for worst-case,best-case and the average case analysis of algorithms.Write an algorithm for finding maximum element in an array.Give worst-case,best-case and the average case complexities.NOV/DEC 2018**	C213.1	BTL1
7.	**Explain Basic Efficiency Classes.**	C213.1	BTL1
8	**What is empirical analysis of an algorithm? Discuss its strength &weakness?**	C213.1	BTL1
9.	**Write short notes on algorithm visualization.**	C213.1	BTL1
10	**If you have to solve the searching problem for a list of n numbers, how can you take advantage of the fact that the list is known to be sorted? Give separate answers for** **(i) lists represented as arrays** **(ii) lists represented as linked lists. Compare the time complexities involved in the analysis of both the algorithms. <u>[APR / MAY 2014]</u>** Refer page no 93 in Anany Levitin.	C213.1	BTL6
11	**Write Euclid's algorithm and explain the steps.**	C213.1	BTL1
12	**Explain the general plan for analyzing efficiency of recursive algorithms.**	C213.1	BTL1
13	**Explain in detail the general framework for analyzing an algorithm's efficiency.**	C213.1	BTL1
14.	**Write the general plan for analyzing efficiency of non recursive algorithms.**	C213.1	BTL1
15	**Write sieve of Eratosthenes algorithm which generates consecutive primes and explain**	C213.1	BTL1

UNIT II

Q. No.	Questions	CO	Bloom's Level
1	**Define Convex-Hull Problem.** A set of points (finite or infinite) on the plane is called convex if for any two points P and Q in the set, the entire line segment with the end points at P and Q belongs to the set.	C213.2	BTL1
2	**What is Divide and Conquer Algorithm?[MAY/JUNE 2016] NOV/DEC 2017** It is a general algorithm design techniques that solved a problem's instance by dividing it into several smaller instance, solving each of them recursively, and then combining their solutions to the original instance of the problem	C213.2	BTL1
3	**What is Fibonacci Numbers?** The Fibonacci numbers are an important sequence of integers in which every element is equal to the sum of its two immediate predecessors. There are several algorithms for computing the Fibonacci numbers with drastically different efficiency.	C213.2	BTL1
4	**What is Brute Force method?** Brute Force is a straightforward approach to solving problem, usually directly based on the problem's statement and definitions of the concepts involved.	C213.2	BTL1
5	**List out the Advantages in Quick Sort** It is in-place since it uses only a small auxiliary stack .• It requires only n log(n) time to sort n items .• It has an extremely short inner loop • This algorithm has been subjected to a thorough mathematical analysis, a very precise statement• can be made about performance issues	C213.2	BTL4
6	**Discuss binary search algorithm and Give computing time for Binary search? [MAY/ JUNE 2015]** The binary search algorithm is one of the most efficient searching techniques which requires the list to be sorted in ascending order. To search for an element in the list, the binary search algorithms split the list and locate the middle element of the list. First compare middle Key k1, with given Key If k1=k then the element is found. Successful searches θ(1) θ(logn) θ(Logn) best average	C213.2	BTL6

	worst unsuccessful searches θ(logn) best, average, worst		
7	**List the advantages of insertion sort. NOV/DEC 2017** Simple implementation. Efficient for (quite) small data sets. Adaptive, i.e. efficient for data sets that are already substantially sorted.	C213.2	BTL4
8	**Show the recurrence relation of divide-and-conquer?** The recurrence relation is $T(n) = \begin{cases} g(n) \\ T(n_1) + T(n_2) + \text{----} + T(n_{BTL}) + f(n) \end{cases}$	C213.2	BTL2
9	**What is exhaustive search? APR/MAY 2018** A brute force solution to a problem involving search for an element with a special property, usually among combinatorial objects such as permutations, combinations, or subsets of a set.	C213.2	BTL2
10	**Elaborate the recurrence relation of merge sort?** If the time for the merging operation is proportional to n, then the computing time of merge sort is described by the recurrence relation $T(n) = \begin{cases} a & n = 1, a \text{ a constant} \\ 2T(n/2) + n & n \geq 1, c \text{ a constant} \end{cases}$	C213.2	BTL6
11	**What is Knapsack problem?** A bag or sack is given capacity n and n objects are given. Each object has weight wi and profit pi .Fraction of object is considered as xi (i.e) 0<=xi<=1 .If fraction is 1 then entire object is put into sack. When we place this fraction into the sack we get $w_i x_i$ and $p_i x_i$.	C213.2	BTL1
12	**What is the use of TSP?** The traveling salesman problem (TSP) is a popular mathematics problem that asks for the most efficient trajectory possible given a set of points and distances that must all be visited. In computer science, the problem can be applied to the most efficient route for data to travel between various nodes.	C213.2	BTL1

13	**Design a brute-force algorithm for computing the value of a polynomial $p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x \cdots a_0$ at a given point x_0 and determine its worst-case efficiency class. [MAY/JUNE 2015]**
	Algorithm BetterBruteForcePolynomialEvaluation(P[0..n], x)
	//The algorithm computes the value of polynomial P at a given point x
	//by the "lowest-to-highest term" algorithm //Input: Array P[0..n] of the coefficients of a polynomial of degree n,
	// from the lowest to the highest, and a number x
	//Output: The value of the polynomial at the point x
	p ← P[0];
	power ← 1 for i ← 1 to n do
	power ← power ∗ x
	p ← p + P[i] ∗ power
		C213.2	BTL6

	return p Worst case efficiency € n^2		
14	**What is the Quick sort and Write the Analysis for the Quick sort?** In quick sort, the division into sub arrays is made so that the sorted sub arrays do not need to be merged later. In analyzing QUICKSORT, we can only make the number of element comparisons c(n). It is easy to see that the frequency count of other operations is of the same order as C(n).	C213.2	BTL1
15	**Define Sum of Subsets problem.** Given n distinct positive numbers usually called as weights , the problem calls for finding all the combinations of these numbers whose sums are m.	C213.2	BTL1
16	**List out the 4 steps in Strassen's Method?** 1. Divide the input matrices A and B into n/2 * n/2 sub matrices, as in equation (1). 2. Using Θ(n2) scalar additions and subtractions, compute 14 n/2 * n/2 matrices A1, B1, A2, B2, …, A7, B7. 3. Recursively compute the seven matrix products Pi =AiBi for i =1, 2, 7. 4. Compute the desired sub matrices r, s, t, u of the result matrix C by adding and/or subtracting various combinations of the Pi matrices, using only Θ(n2) scalar additions and subtractions..	C213.2	BTL4
17	**What is approximate solution?** A feasible solution with value close to the value of an optimal solution is called approximate solution.	C213.2	BTL1
18	**Give the time efficiency and drawback of merge sort algorithm.** Time efficiency : The best, worst and average case time complexity of merge sort is O(nlogn) The drawbacks : (I) This algorithm requires extra storage to execute this method (ii) This method is slower than the quick sort method (iii) This method is complicated to code.	C213.2	BTL1
19	**What is the maximum and minimum problem?** The problem is to find the maximum and minimum items in a set of „n" elements. Though this problem may look so simple as to be contrived, it allows us to demonstrate divide-and-conquer in simple setting.	C213.2	BTL1
20	**List the strength and weakness of brute force algorithm**. Strengths a. wide applicability, b. simplicity c. yields reasonable algorithms for some important problems (e.g., matrix multiplication, sorting, searching, string matching) Weaknesses a. rarely yields efficient algorithms b. some brute-force algorithms are unacceptably slow not as constructive as some other design techniques	C213.2	BTL1
21	**Summarize the general plan of exhaustive search**. • generate a list of all potential solutions to the problem in a systematic manner • evaluate potential solutions one by one, disqualifying infeasible ones and, for an optimization problem, keeping track of the best one found so far • when search ends, announce the solution(s) found.	C213.2	BTL2

22	**Define recursive call?** An algorithm is said to be recursive if the same algorithm invoked in the body. There are 2 types of algorithm. They are 1) Direct Recursive 2) Indirect Recursive	C213.2	BTL1
23	**What is meant by Direct recursive call?** An algorithm that calls itself is direct recursive call. Eg. int fun(int x) { if(x<=0) return x; return (fun(x-1));	C213.2	BTL1
24	**Define indirect recursive call?** Algorithm A is said to be indirect recursive if it calls another algorithm which in turn call A Eg: int fun(int x) { if(x<=0) return x; return (f1(x-1)); } Int fun1(int y){ return fun(y-1) }	C213.2	BTL1
25	**Define Extrapolation?** Extrapolation is approach, which deals with values of n, that are outside of the range of the samples values.	C213.2	BTL1
26	**Define profiling?** Profiling is an important resource the empirical analysis of an algorithm running time. Measuring n different segments of program can pinpoint a bottleneck in the program's performance that can be missed by an abstract deliberation about the algorithm's basic operations. The process of getting such data is called profiling.	C213.2	BTL1
27	**What is closest pair problem? MAY/JUNE 2016, APRIL/MAY 2017** The closest pair problem is to find the two closest points in a set of n points. The distance between two Cartesian coordinates is calculated by Euclidean distance formula. d(pi,pj)=(xi-xj)2+(yi-yj)2	C213.2	BTL1
28	**Illustrate the Assignment problem? MAY/JUNE 2016** There are n people who need to be assigned to execute n jobs as one person per job. Each person is assigned to exactly one job and each job is assigned to exactly one person.	C213.2	BTL2
29	**Define of feasibility** A feasible set (of candidates) is promising if it can be extended to produce not merely a solution, but an optimal solution to the problem.	C213.2	BTL1
30	**What is the general divide-and-conquer recurrence relation?** An instance of size _n' can be divided into several instances of size n/b, with _a' of them needing to be solved. Assuming that size _n' is a power of _b', to simplify the analysis, the following recurrence for the running time is obtained: T(n) = aT(n/b)+f(n) Where f(n) is a function that accounts for the time spent on dividing the problem into smaller ones and on combining their solutions	C213.2	BTL1
31	**State Master's Theorem. APR/MAY 2018** Master Method is a direct way to get the solution. The master method works only for following type of recurrences or for recurrences that can be	C213.2	BTL1

	transformed to following type. $T(n) = aT(n/b) + f(n)$ where $a >= 1$ and $b > 1$ There are following three cases: **1.** If $f(n) = \Theta(n^c)$ where $c < Log_b a$ then $T(n) = \Theta(n^{Log_b a})$ **2.** If $f(n) = \Theta(n^c)$ where $c = Log_b a$ then $T(n) = \Theta(n^c Log\ n)$ **3.**If $f(n) = \Theta(n^c)$ where $c > Log_b a$ then $T(n) = \Theta(f(n))$		
32	**Is insertion sort better than the merge sort?** Insertion sort works exceedingly fast on arrays of less then 16 elements, though for large „n" its computing time is O(n2).	C213.2	BTL1

33	**Write a algorithm for straightforward maximum and minimum algorithm .**
	straight MaxMin(a,n,max,min) //set max to the maximum and min to the minimum of a[1:n]
	{
	max := min: = a[i];
	for i = 2 to n do
	{
	if(a[i] >max) then
	max: = a[i];
	if(a[i] >min) then
	min: = a[i];
	}
		C213.2	BTL1

34	**Write the algorithm for Iterative binary search?** Algorithm BinSearch(a,n,x) //Given an array a[1:n] of elements in nondecreasing // order, n>0, determine whether x is present { low : = 1; high : = n; while (low < high) do { mid : = [(low+high)/2]; a[mid]) then high:= mid-1;<if(x a[mid]) then low:=mid<else if (x + 1; else return mid; } return 0;	C213.2	BTL1
35	**What is the method of backward substitution?** Among several techniques available for solving recurrence relations, one of the method used is called the method of backward substitution. The method's idea will be clear by referring to the particular case as shown below : M(n) = M(n-1) + 1 for n > 0. M(0) = 0 = [M(n-2) + 1] + 1 = M(n-2) + 2 = [M(n-3) +1] + 2 = M(n-3) + 3 =M(n-n) + n = n.	C213.2	BTL1
36	**Give some examples of Brute force approach?** a) Selection sort b) bubble sort c) string matching	C213.2	BTL1
37	**What is the principal strength of brute force approach?** Wide applicability and simplicity	C213.2	BTL1
38	**What is a pivot?** In quick sort ,we partition the given array into two sub arrays based on the value stored in the element called pivot	C213.2	BTL1
39	**What is decrease-and-conquer technique?** The decrease-and-conquer technique is based on exploiting the relationship	C213.2	BTL1

	between a solution to a given instance of a problem and solution to a smaller instance of the same problem.		
40	**What are the three major variations of the decrease-and-conquer technique?** The three major variations of the decrease-and-conquer technique are Decrease by a constant decrease by a constant factor variable size decrease	C213.2	BTL1
41	**What is heap sort ?** The heapsort algorithm involves preparing the list by first turning it into a max heap. The algorithm then repeatedly swaps the first value of the list with the last value, decreasing the range of values considered in the heap operation by one, and sifting the new first value into its position in the heap. This repeats until the range of considered values is one value in length.	C213.2	BTL1
42	**Write Closest Pair of Points algorithm using Divide and Conquer .** 1) Find the middle point in the sorted array, we cantakeP[n/2] as middle point. 2) Divide the given array in two halves. ... 3) Recursively find the smallest distances in both subarrays. ... 4) From above 3 steps, we have an upper bound d of minimum distance. ... 5) Sort the array strip[] according to y coordinates.	C213.2	BTL1
43	**What is meant by string-searching algorithms?** String-searching algorithms sometimes called string-matching algorithms, are an important class of string algorithms that try to find a place where one or several strings (also called patterns) are found within a larger string or text.	C213.2	BTL1
44	**Write the efficiency of heap sort algorithm.** Heap sort is an in-place algorithm. Time Complexity: Time complexity of heapify is O(Logn). Time complexity of createAndBuildHeap() is O(n) and overall time complexity of Heap Sort is O(nLogn)	C213.2	BTL1
45	**Write heap sort algorithm** The steps are: Call the buildMaxHeap() function on the list. Also referred to as heapify(), this builds a heap from a list in O(n) operations. Swap the first element of the list with the final element. Decrease the considered range of the list by one. Call the siftDown() function on the list to sift the new first element to its appropriate index in the heap. Go to step (2) unless the considered range of the list is one element.	C213.2	BTL1
46	**Write the selection sort algorithm** The selection sort algorithm sorts an array by repeatedly finding the minimum element (considering ascending order) from unsorted part and putting it at the beginning. The algorithm maintains two sub arrays in a given array.	C213.2	BTL1

	1) The subarray which is already sorted. 2) Remaining subarray which is unsorted. In every iteration of selection sort, the minimum element (considering ascending order) from the unsorted subarray is picked and moved to the sorted subarray.		
47	**What is the efficiency of selection sort** O(n2) time complexity	C213.2	BTL1
48	**Write the bubble sort algorithm** Bubble sort, sometimes referred to as sinking sort, is a simple sorting algorithm that repeatedly steps through the list, compares adjacent pairs and swaps them if they are in the wrong order. The pass through the list is repeated until the list is sorted. The algorithm, which is a comparison sort, is named for the way smaller or larger elements "bubble" to the top of the list.	C213.2	BTL1
49	**What is the efficiency of bubble sort?** Bubble sort has a worst-case and average complexity of O(n2), where n is the number of items being sorted.	C213.2	BTL1
50	**Define control abstraction.** A control abstraction we mean a procedure whose flow of control is clear but whose primary operations are by other procedures whose precise meanings are left undefined.	C213.2	BTL1

PART-B

Q. NO.	QUESTIONS	CO	BLOOM'S LEVEL
1	**Explain closest pair Problems by Brute Force method. NOV/DEC 2017** Refer page no 127 in Anany Levitin	C213.2	BTL5
2	**Explain Traveling Salesman Problem by Brute Force method.[MAY/JUNE 2016],NOV?DEC 2018** Refer page no 137 in Anany Levitin	C213.2	BTL5
3	**Explain Knapsack problem by Brute Force method.** Refer page no 139 in Anany Levitin	C213.2	BTL5
4	**Explain Merge sort, and arrange the following numbers in increasing order using merge sort**. (18, 29, 68, 32, 43,37, 87, 24, 47, 50)**,APR/MAY 2015,MAY/JUNE 2016 NOV/DEC 2017, APR/MAY 2018** Refer page no 148 in Anany Levitin	C213.2	BTL5

	Write an algorithm for quick sort and write its time complexity with example list are 5, 3, 1, 9, 8, 2, 4, 7. (15) **APR/MAY 2017**		
5	**Explain Multiplication of Large integers And Strassen's Matrix multiplication**. [**NOV/DEC 2015,MAY/JUNE 2016] APR/MAY 2018** Refer page no 166in Anany Levitin	C213.2	BTL5
6	**Explain Closest pair and Convex-Hull Problems by divide and Conquer method. APR/MAY 2015]** Refer page no 171 in Anany Levitin	C213.2	BTL1
7	**A pair contains 2 numbers, and its second number is on the right side of the first one in an array. The difference of a pair is the minus result while subtracting the second number from the first one. Construct a function which gets the maximal difference of all pairs in an array (using Divide and Conquer Method). [APR/MAY 2015]**	C213.2	BTL6
8	**Explain the binary search algorithm with suitable example. APRIL / MAY 2017**	C213.2	BTL5
9	**Consider the problem of finding the smallest and largest elements in an array of N numbers.** **i)Design a presorting –based algorithm for solving this problem and determine its efficiency class.** **ii)Compare the efficiency of the three algorithms:** **(A)the brute –force algorithm.(B)this presorting based algorithm ,and (C)the Divide-and conquer algorithm.**	C213.2	BTL6
10	**Explain Assignment problem by Brute Force method.**	C213.2	BTL1
11	**Explain Quick sort and arrange the following numbers in increasing order using QUICK sort. (18, 29, 68, 32, 43,37, 87, 24, 47, 50), NOV/DEC 2017, APR/MAY 2018,NOV/DEC 2018**	C213.2	BTL5
12	**Write the algorithm for Computing a^n**	C213.2	BTL1
13	**Explain String Matching algorithm in detail**	C213.2	BTL1
14	**Explain Convex-Hull Problems by Brute Force method. NOV/DEC 2017**	C213.2	BTL1
15	**Explain heap sort and arrange the following numbers in increasing order using heap sort. (18, 29, 68, 32, 43,37, 87, 24, 47, 50),**	C213.2	BTL5

UNIT III

Q. No.	Questions	CO	Bloom's Level
1	**Explain principle of Optimality? NOV/DEC 2017** The principle of optimality says that an optimal solution to any instance of an optimization problem is composed of optimal solution to its sub-instances.	C213.3	BTL2
2	**What is need for finding minimum spanning tree?** Spanning tree has many applications. Any connected graphs with n vertices mush have at least n-1 edges and all connected graphs with n-1 edges are trees. If the nodes of G represent cities and edges represent possible communication links connecting 2 cities, then the minimum number of links needed to connect the n cities is n-1. Therefore, it is necessary for finding minimum spanning tree.	C213.3	BTL1
3	**What is critical path?** A path of longest length is called critical path. For example in tree	C213.3	BTL1
4	**Define minimum Spanning Tree problem?APR/MAY 2018** A **minimum spanning tree** (**MST**) or **minimum weight spanning tree** is a subset of the edges of a connected, edge-weighted (un)directed graph that connects all the vertices together, without any cycles and with the minimum possible total edge weight.	C213.3	BTL1
5	**What is Dynamic programming?** Dynamic programming is an algorithm design technique for solving problem with overlapping subprograms. The smaller subprograms are solved only once and recording the results in a table from which the solution to the original problem is obtained	C213.3	BTL1
6	**What is greedy method? APRIL/MAY 2017** The greedy method is the most straight forward design, which is applied for change making problem. The greedy technique suggests constructing a solution to an optimization problem through a sequence of steps, each expanding a partially constructed solution obtained so far, until a complete solution to the problem is reached. On each step, the choice made must be feasible, locally optimal and irrevocable.	C213.3	BTL1
7	**List the advantage of greedy algorithm** a.Greedy algorithm produces a feasible solution b.Greedy method is very simple to solve a problem c.Greedy method provides an optimal solution directly.	C213.3	BTL4

<table>
<tr><td>8</td><td>List the applications of minimum spanning tree?
Spanning tree are used to obtain an independent set of circuit equations for an electric network. Another application of spanning tree arises from the property that a spanning tree is a minimal sub graph G’ of G such that V(G’)=V(G) and G’ is connected</td><td>C213.3</td><td>BTL4</td></tr>
<tr><td>9</td><td>What do you mean by row major and column major?
In a given matrix, the maximum elements in a particular row is called row major.
In a given matrix, the maximum elements in a particular column is called column major.</td><td>C213.3</td><td>BTL1</td></tr>
<tr><td>10</td><td>What is meant by feasible solution?
Given n inputs and we are required to form a subset such that it satisfies some given constraints then such a subset is called feasible solution.</td><td>C213.3</td><td>BTL1</td></tr>
<tr><td>11</td><td>Illustrate any two characteristics of Greedy Algorithm?
a) To solve a problem in an optimal way construct the solution from given set of candidates.
b) As the algorithm proceeds, two other sets get accumulated among this one set contains the candidates that have been already considered and chosen while the other set contains the candidates that have been considered but rejected.</td><td>C213.3</td><td>BTL2</td></tr>
<tr><td>12</td><td>Define optimal solution?
A feasible solution either maximizes or minimizes the given objective function is called as optimal solution.</td><td>C213.3</td><td>BTL1</td></tr>
<tr><td>13</td><td>What are the differences between dynamic programming and divide and conquer approaches?NOV?DEC 2018
Divide and Conquer
Divide and Conquer works by dividing the problem into sub-problems, conquer each sub-problem recursively and combine these solutions.
Dynamic Programming
Dynamic Programming is a technique for solving problems with overlapping sub problems. Each sub-problem is solved only once and the result of each sub-problem is stored in a table (generally implemented as an array or a hash table) for future references. These sub-solutions may be used to obtain the original solution and the technique of storing the sub-problem solutions is known as memorization.</td><td>C213.3</td><td>BTL1</td></tr>
<tr><td>14</td><td>Compare Greedy method and Dynamic programming.
<table><tr><th>Greedy method</th><th>Dynamic programming</th></tr><tr><td>1.Only one sequence of decision is generated.</td><td>1.Many number of decisions are generated.</td></tr><tr><td>2.It does not guarantee to give an optimal solution always.</td><td>2.It definitely gives an optimal solution always.</td></tr></table></td><td>C213.3</td><td>BTL5</td></tr>
</table>

15	**List the features of dynamic programming?** Optimal solutions to sub problems are retained so as to avoid recomputing their values. Decision sequences containing subsequences that are sub optimal are not considered. It definitely gives the optimal solution always.	C213.3	BTL4
16	**What are the drawbacks of dynamic programming?** Time and space requirements are high, since storage is needed for all level. Optimality should be checked at all levels.	C213.3	BTL1
17	**Show the general procedure of dynamic programming. APRIL/MAY 2017** The development of dynamic programming algorithm can be broken into a sequence of 4 steps. **1.** Characterize the structure of an optimal solution. **2.** Recursively define the value of the optimal solution. **3.** Compute the value of an optimal solution in the bottom-up fashion. Construct an optimal solution from the computed information	C213.3	BTL2
18	**How dynamic programming is used to solve Knapsack problem.NOV?DEC 2018** An example of dynamic programming is Knapsack problem. The solution to the Knapsack problem can be viewed as a result of sequence of decisions. We have to decide the value of xi for $1<i\leq n$. First we make a decision on x1 and then on x2 and so on. An optimal sequence of decisions maximizes the object function $\Sigma p_i x_i$.	C213.3	BTL2
19	**Define warshall's algorithm?** Warshall's algorithm is an application of dynamic programming technique, which is used to find the transitive closure of a directed graph.	C213.3	BTL1
20	**What does Floyd's algorithm do? NOV/DEC 2017** Floyd's algorithm is an application, which is used to find the entire pairs shortest paths problem. Floyd's algorithm is applicable to both directed and undirected weighted graph, but they do not contain a cycle of a negative length.	C213.3	BTL1
21	**Define prim's algorithm.** Prim's algorithm is a greedy and efficient algorithm, which is used to find the minimum spanning tree of a weighted connected graph.	C213.3	BTL1
22	**How efficient is prim's algorithm?** The efficiency of the prim's algorithm depends on data structure chosen for the graph.	C213.3	BTL1
23	**Define kruskal's algorithm?** kruskal's algorithm is another greedy algorithm for the minimum spanning tree problem. kruskal's algorithm constructs a minimum spanning tree by selecting edges in increasing order of their weights provided that the inclusion does not create a cycle. kruskals algorithm provides a optimal solution.	C213.3	BTL1

24	**What is path compression?** The better efficiency can be obtained by combining either variation of quick union with path compression. Path compression makes every node encountered during the execution of a find operation point to the tree's node.	C213.3	BTL1
25	**Define Dijkstra's Algorithm?** Dijkstra's algorithm solves the single source shortest path problem of finding shortest paths from a given vertex(the source), to all the other vertices of a weighted graph or digraph. Dijkstra's algorithm provides a correct solution for a graph with non negative weights.	C213.3	BTL1
26	**What is Huffman trees?** A Huffman tree is binary tree that minimizes the weighted path length from the root to the leaves containing a set of predefined weights. The most important application of Huffman trees are Huffman code.	C213.3	BTL1
27	**What do you mean by Huffman code?** A Huffman code is a optimal prefix tree variable length encoding scheme that assigns bit strings to characters based on their frequencies in a given text.	C213.3	BTL1
28	**What is meant by compression ratio?** Huffman's code achieves the compression ratio, which is a standard measure of compression algorithm's effectiveness of (3-2.25)/3*100 = 0.75/3*100 = 0.25 *100 = 25%.	C213.3	BTL1
29	**List the advantage of Huffman's encoding?** a. Huffman's encoding is one of the most important file compression methods. b. It is simple c. It is versatility **d.** It provides optimal and minimum length encoding	C213.3	BTL4
30	**Define the single source shortest path problem.<u>MAY/JUNE 2016</u>]** Single source shortest path problem can be used to find the shortest path from single source to all other vertices. Example:Dijikstras algorithm	C213.3	BTL1
31	**List out the memory functions used under dynamic programming. <u>[MAY/JUNE 2015</u>]** Refer notes	C213.3	BTL4
32	**Define transitive closure of a directed graph.APR/MAY 2018** Given a **directed graph**, find out if a vertex j is reachable from another vertex i for all vertex pairs (i, j) in the given **graph**. Here reachable mean that there is a path from vertex i to j. The reach-ability matrix is called **transitive closure** of a **graph**.	C213.3	BTL1
33	**What is meant by coin changing problem** Given a set of coins and amount, Write an algorithm to find out how many ways we can make the change of the amount using the coins given.	C213.3	BTL1

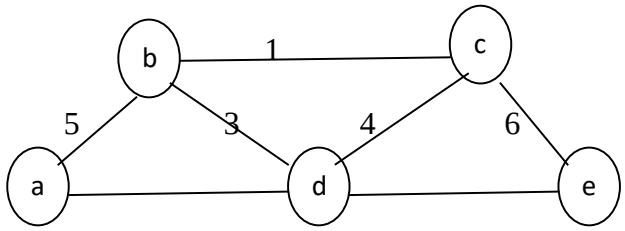
34	**Write the method for Computing a Binomial Coefficient** Computing binomial coefficients is non optimization problem but can be solved using dynamic programming. Binomial coefficients are represented by C(n, k) or (nk) and can be used to represent the coefficients of a binomial: (a + b)n = C(n, 0)an + ... + C(n, k)an-kbk + ... + C(n, n)bn The recursive relation is defined by the prior power C(n, k) = $C(n-1, k-1) + C(n-1, k)$ for $n > k > 0$ IC $C(n, 0) = C(n, n) = 1$	C213.3	BTL1
35	**Define multistage graph.NOV/DEC 2018** A Multistage graph is a directed graph in which the nodes can be divided into a set of stages such that all edges are from a stage to next stage only (In other words there is no edge between vertices of same stage and from a vertex of current stage to previous stage).	C213.3	BTL1
36	**Define Container Loading Problem** The basic Container Loading Problem can be defined as the problem of placing a set of boxes into the container respecting the geometric constraints: the boxes cannot overlap and cannot exceed the dimensions of the container.	C213.3	BTL1
37	**What is meant by Optimal merge pattern** Optimal merge pattern is a pattern that relates to the merging of two or more sorted files in a single sorted file. This type of merging can be done by the two-way merging method.	C213.3	BTL1

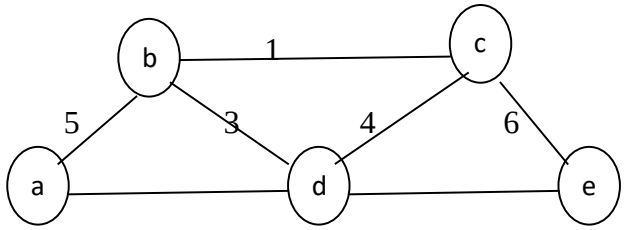
38	**Write Optimal merge pattern algorithm**
	Least (L): find a tree in L whose root has the smallest weight.
	Function : Tree (L,n).
	Integer i;
	Begin
	For i=1 to n -1 do
	Get node (T) /* create a node pointed by T */
	Left child (T)= Least (L) /* first smallest */
	Right child (T)= Least (L) /* second smallest */
	Weight (T) = weight (left child (T)) + weight (right child (T))
	Insert (L,T); /* insert new tree with root T in L */
	End for
	Return (Least (L)) /* tree left in L */ End.
		C213.3	BTL1

39	**Write the time complexity of optimal merge pattern algorithm** If we have two sorted files containing n and m records respectively then they could be merged together, to obtain one sorted file in time O (n+m).	C213.3	BTL1
40	**Write the Algorithm for building a Huffman coding tree.** make a list of all symbols with their frequencies	C213.3	BTL1

	sort the list so the symbols with the least frequency are in front if the list only has one element, the element is the root of the tree and we are done remove the first two elements from the list and put them into a binary tree add the frequencies of the two sub trees to give the frequency of this binary tree insert this tree in the right place in the sorted list return to step 3		
41	**Define 0/1 Knapsack problem.** The solution to the Knapsack problem can be viewed as a result of sequence of decisions. We have to decide the value of xi. xi is restricted to have the value 0 or 1 and by using the function knap(l, j, y) we can represent the problem as maximum Σpi xi subject to Σwi xi < y where l - iteration, j - number of objects, y – capacity.	C213.3	BTL1
42	**What is the formula to calculate optimal solution in 0/1 Knapsack problem?** The formula to calculate optimal solution is $g_0(m)=\max\{g_1, g_1(m-w_1)+p_1\}$.	C213.3	BTL1
43	**Write some applications of traveling salesperson problem.** Routing a postal van to pickup mail from boxes located at n different sites. Using a robot arm to tighten the nuts on some piece of machinery on an assembly line. Production environment in which several commodities are manufactured on the same set of machines	C213.3	BTL1
44	**Give the time complexity and space complexity of traveling salesperson problem.** Time complexity is O ($n^2 2^n$). Space complexity is O ($n 2^n$).	C213.3	BTL1
45	**Define Distance matrix** Recording the lengths of shortest path in n x n matrix is called Distance matrix(D)	C213.3	BTL1
46	**Define All pair shortest path problem** Given a weighted connected graph, all pair shortest path problem asks to find the lengths of shortest paths from each vertex to all other vertices.	C213.3	BTL1
47	**State the time efficiency of floyd's algorithm** O(n3) It is cubic	C213.3	BTL1
48	Define OBST • Dynamic programming • If probabilities of searching for elements of a set are known then finding optimal BST for which the average number of comparisons in a search is smallest possible	C213.3	BTL1
49	**Define catalan number** The total number of binary search trees with n Keys is equal to nth catalan	C213.3	BTL1
50	**State efficiency of prim's algorithm** O(|v| 2) (Weight Matrix And Priority Queue As Unordered Array) O(|E| Log|V|) (Adjacency List And Priority Queue As Minheap)	C213.3	BTL1

PART-B

Q. NO.	QUESTIONS	CO	BLOOM'S LEVEL
1	**Explain the algorithm to solve all pairs shortest paths problem APRIL/MAY 2010, NOV/DEC 2010, MAY/JUNE 2012** Refer page no 304 in Anany Levitin	C213.3	BTL5
2	**Apply function OBST to compute w(i, j), r(i, j), and c(i, j), 0<=i<j<=4, for the identifier set ( ) 4 3 2 1 , , , a a a a = (cout, float, if, while) with p(1) = 1/20, p(2) =1/5, p(3) = 1/10, p(4) = 1/20, q(0) = 1/5, q(1) = 1/10, q(2) = 1/5, q(3) = 1/20,and q(4) 1/20. Using the r(i, j)'s, construct the optimal binary search tree.[APRIL/MAY 2011, APR/MAY 2015]**	C213.3	BTL3
3	**Apply Prim's algorithm to find a minimum spanning tree for the following graph: [APRIL/MAY 2011, APR /MAY 2015]** 	C213.3	BTL3
4	Given the mobile numeric keypad. You can only press buttons that are up, left, right or down to the first number pressed to obtain the subsequent numbers. You are not allowed to press bottom row corner buttons (i.e. * and #). Given a number N, how many key strokes will be involved to press the given number. What is the length of it? Which dynamic programming technique could be used to find solution for this? Explain each step with the help of a pseudo code and derive its time complexity. (12) **[APR/MAY 2015]**	C213.3	BTL5
5	**Discuss Dijkstra's Algorithm with example. NOV/DEC 2017,NOV/DEC 2108** Refer page no 343 in Anany Levitin	C213.3	BTL6
6	**Explain Huffman Trees with the following example. [APR/MAY 2015]** Let $A = \{l/119, m/96, c/247, g/283, h/72, f/77, k/92, j/19\}$ be the letters and its frequency of distribution in a text file. Compute a suitable Huffman coding to compress the data effectively. (8) Refer page no 348 in Anany Levitin	C213.3	BTL5
7	**Discuss about the algorithm and pseudo code to find the minimum spanning tree using prim's algorithm.[MAY/JUNE 2016], NOV/DEC**	C213.3	BTL6

	2017, APR/MAY 2018 **Refer page no 343 in Anany Levitin**		
8	**Construct the Huffman's tree for following data and obtain its Huffman's code. Write the Huffman's Algorithm.APR/MAY 2017** Character A B C D E – Probability 0.5 0.35 0.5 0.1 0.4 0.2	C213.3	BTL6
9	**Explain Multi stage graph in detail**	C213.3	BTL1
10	**Explain the memory function method for the Knapsack problem and give the algorithm? APR/MAY 2018**	C213.3	BTL6
11	**Explain Coin changing problem in detail**	C213.3	BTL6
12	**Apply Kruskal's algorithm to find a minimum spanning tree for the following graph: [APRIL/MAY 2011, APR /MAY 2015]**  	C213.3	BTL6
13	**Apply Warshall's algorithm to find the transitive closure of the digraph.Prove that the time efficiency of Warshall's algorithm is cubic. Explain why the time efficiency of Warshall's algorithm is inferior to that of the traversal-based algorithm for sparse graphs represented by their adjacency lists.APR/MAY 2018,NOV/DEC 2018**	C213.3	BTL6
14	**Explain Computing a Binomial Coefficient**	C213.3	BTL1
15	**Solve the following instance of the 0/1 Knapsack problem given the Knapsack capacity W=5 using dynamic programming and explain it, APR/MAY 2017** Items Weight Value 1 4 10 2 3 20 3 2 15 4 5 25	C213.3	BTL6

UNIT IV

PART – A

Q. No.	Questions	CO	Bloom's Level
1	**What is iterative improvement method?** This is a computational technique in which with the help of initial feasible solution the optimal solution is obtained iteratively until no improvement is found.	C213.4	BTL1
2	**List various applications of iterative improvement method.** 1.Simplex method 2.Matching graph vertices 3.Stable marriage problem 4.Finding maximum network flow.	C213.4	BTL4
3	**What is Simplex Method?** The Simplex Method is "a systematic procedure for generating and testing candidate vertex solutions to a linear program." It begins at an arbitrary corner of the solution set. At each iteration, the Simplex Method selects the variable that will produce the largest change towards the minimum (or maximum) solution. That variable replaces one of its compatriots that is most severely restricting it, thus moving the Simplex Method to a different corner of the solution set and closer to the final solution. In addition, the Simplex Method can determine if no solution actually exists.	C213.4	BTL1
4	**How Iterative improvement solves problems.NOV/DEC 18** Iterative improvement solves problems where: ❖ The problem is an optimization problem, to find the solution that minimizes or ❖ maximizes some value (cost/profit). ❖ An initial solution can be easily found. ❖ It can be improved by a sequence of small changes. ❖ It is returned when no more improvements can be made.	C213.4	BTL1
5	**What is linear programming problem?** The standard form of linear programming is P=ax+by+cz LP problem is a problem in which we have to find the maximum or minimum value of a linear objective function.	C213.4	BTL1
6	**What is meant by Bipartite Graph? NOV/DEC 2017** A Bipartite Graph G = (V;E) is a graph in which the vertex set V can be divided into two disjoint subsets X and Y such that every edge e € E has one end point in X and the other end point in Y .A matching M is a subset of edges such that each node in V appears in at most one edge in M.	C213.4	BTL1

7	**What is two colorable graph?** It is a graph that can be colored with only two colors in such a way that no edge connects the same color. The bipartite graph is two colorable graph.	C213.4	BTL1
8	**What is maximum cardinality matching?APR/MAY 2018** A **maximum matching** (also known as **maximum-cardinality matching**) is a **matching** that contains the largest possible number of edges. There may be many **maximum matchings**. The **matching** number of a graph is the size of a **maximum matching**.	C213.4	BTL1
9	**What is meant by Maximum Matching?** A maximum matching is a matching with the largest possible number of edges; it is globally optimal.	C213.4	BTL1
10	**What is network?** **A flow network** G=(V,E) is a directed graph in which each edge (u,v) € E has a nonnegative capacity c(u,v) ≥ 0.	C213.4	BTL1
11	**Define Maximum Flow Theorem.** A flow has maximum value if and only if it has no augmenting path.	C213.4	BTL1
12	**What is Augmenting path in bipartite graph.?** The Augmenting path P is a path in Graph G,such that it is an alternating path with special property that-Its start and end vertices are free or unmatched.	C213.4	BTL1
13	**When can we say that the optimal solution is obtained in simplex method?** When objective function (ie value of z) is largest then the optimal solution is said to be obtained in simplex method.	C213.4	BTL1
14	**What is entering variable?** The entering variable is the smallest negative entry in the bottommost row of simplex table.	C213.4	BTL1
15	**What is pivot element in simplex method?** The intersection of entering variable's column and departing variable's row is called pivot.	C213.4	BTL1
16	**Illustrate the Stable Marriage Problem.** The **stable marriage problem** (**SMP**) is the problem of finding a **stable matching** between two sets of elements given a set of preferences for each element. A matching is a mapping from the elements of one set to the elements of the other set.	C213.4	BTL2
17	**Explain Stable marriage problem algorithm.** **Input**: A set of *n* men and a set of *n* women along with rankings of the women by each man and rankings of the men by each woman with no ties allowed in the rankings **Output:** A stable marriage matching. **Step 0 :**Start with all the men and women being free. **Step 1 :**While there are free men, arbitrarily select one of them and do the following: ***Proposal:*** The selected free man *m* proposes to *w*, the next woman	C213.4	BTL2

	on his preference list (who is the highest-ranked woman who has not rejected him before). ***Response:*** If *w* is free, she accepts the proposal to be matched with *m*. If she is not free, she compares *m* with her current mate. If she prefers *m* to him, she accepts *m*'s proposal, making her former mate free; otherwise, she simply rejects *m*'s proposal, leaving *m* free. **Step 2** Return the set of *n* matched pairs		
18	**When we can tell that the matching is stable in SMP?** A matching is stable whenever it is *not* the case that both: *a.* some given element *A* of the first matched set prefers some given element *B* of the second matched set over the element to which *A* is already matched, and **b.** *B* also prefers *A* over the element to which *B* is already matched.	C213.4	BTL1
19	**Show the requirements of the standard form in simplex method.** • It must be a maximization problem. • All the constraints (except the nonnegativity constraints) must be in the form • of linear equations with nonnegative right-hand sides. All the variables must be required to be nonnegative	C213.4	BTL2
20	**How to find the entering variable in simplex method.** Select a negative entry from among the first *n* elements of the objective row. (A commonly used rule is to select the negative entry with the largest absolute value, with ties broken arbitrarily.) Mark its column to indicate the entering variable and the pivot column.	C213.4	BTL1
21	**How to find the departing variable in simplex method.** For each positive entry in the pivot column, calculate the θ-ratio by dividing that row's entry in the rightmost column by its entry in the pivot column. (If all the entries in the pivot column are negative or zero, the problem is unbounded—stop.) .Find the row with the smallest θ-ratio (ties may be broken arbitrarily),and mark this row to indicate the departing variable and the pivot row.	C213.4	BTL1
22	**What is flow network.** It contains exactly one vertex with no entering edges; this vertex is called the ***source*** and assumed to be numbered 1. It contains exactly one vertex with no leaving edges; this vertex is called the ***sink*** and assumed to be numbered *n*. The weight *uij* of each directed edge *(i, j)* is a positive integer, called the edge ***capacity***. (This number represents the upper bound on the amount of the material that can be sent from *i* to *j* through a link represented by this edge.) .A digraph satisfying these properties is called a ***flow network*** or simply a ***network***.	C213.4	BTL1
23	**What is a cuts in flow networks. [APR/MAY 2015]** Cut is a collection of arcs such that if they are removed there is no path from source to sink	C213.4	BTL1
24	**What is meant by flow-conservation requirement** It is assumed that the source and the sink are the only source and destination	C213.4	BTL1

	of the material, respectively; all the other vertices can serve only as points where a flow can be redirected without consuming or adding any amount of the material. In other words, the total amount of the material entering an intermediate vertex must be equal to the total amount of the material leaving the vertex. This condition is called the *flow-conservation requirement*.		
25	**Define *Max-Flow Min-Cut Theorem*.** The value of a maximum flow in a network is equal to the capacity of its minimum cut.	C213.4	BTL1
26	**What is a state space graph?[MAY/JUNE 2016]** Graph organization of the solution space is state space tree.	C213.4	BTL1
27	**Define slack variable.** Variables transforming inequality constraints into equality constraints are called slack variables.	C213.4	BTL1
28	**Define extreme point theorem. NOV/DEC 2017** Any LP problem with a non empty bounded feasible region has an optimal solution; moreover, an optimal solution can always be found at an extreme point of the problems feasible region. This theorem implies that to solve a linear programming problem, at least in the case of a bounded feasible region, we can ignore all but a finite number of points in its feasible region.	C213.4	BTL1
29	**What do you mean by perfect matching in Bipartite graph? APRIL/MAY 2017** A perfect matching of a graph is a matching (i.e., an independent edge set) in which every vertex of the graph is incident to exactly one edge of the matching. A perfect matching is therefore a matching containing $n/2$ edges (the largest possible), meaning perfect matching are only possible on graphs with an even number of vertices.	C213.4	BTL1
30	**Explain Planar colouring graph problem. APRIL/MAY 2017** A graph is planar if it can be drawn in a plane without edge-crossings. The four color theorem states that any planar map can be colored with at most four colors. In graph terminology, this means that using at most four colors, any planar graph can have its nodes colored such that no two adjacent nodes have the same color.	C213.4	BTL2
31	**What is an articulation point in a graph? APR/MAY 2017** A vertex in an undirected connected **graph** is an **articulation point**(or cut vertex) iff removing it (and edges through it) disconnects the **graph**. It can be thought of as a single **point** of failure.	C213.4	BTL1
32	**How is a transportation network represented? APR/MAY 2018** **Transportation networks** generally refer to a set of links, nodes, and lines that **represent** the infrastructure or supply side of the **transportation**. The links have characteristics such as speed and capacity for roadways; frequency and travel time data are defined on **transit** links or lines for the **transit system**	C213.4	BTL1
33	**What is Solution Space ?Give An Example NOV/DEC 18** In mathematical optimization, a feasible region, feasible set, search space, or solution space is the set of all possible points (sets of values of the choice	C213.4	BTL1

	variables) of an optimization problem that satisfy the problem's constraints, potentially including inequalities, equalities, and integer constraints. In linear programming problems, the feasible set is a convex polytope		
34	**Write the requirements of linear programming problem standard form** 1.It must be a maximization problem. 2.All the constraints (except the nonnegative constraints) must be in the form of linear equations with nonnegative right hand sides. 3.All the variables must be required to be nonnegative.	C213.4	BTL1
35	**Standard form of linear programming problem** **Maximize c1x1+…+cnxn// Objective function** **Subject to $a_{i1}x_1+….+a_{in}x_n=b_i$** **$_{xi}$>=0**	C213.4	BTL1
36	**Write the optimality test in simplex method.** If all the entries in the objective row except the one in the rightmost column, which represents the value of the objective function are non negative then stop.	C213.4	BTL1
37	**How to form the next table in the simple method.** Divide all the entries in the pivot row by its entry in the pivot column. Subtract from each of the other rows including the objective row ,the new pivot row multiplied by the entry in the pivot column of the row in question	C213.4	BTL1
38	**What is matching in bipartite graph.** A matching in a graph is a subset of its edges with the property that no two edges share a vertex.	C213.4	BTL1
39	**What is free vertex in bipartite graph?** A vertex is set to be a free vertex if no edge from matching M is incident to V.ie if v is not matched	C213.4	BTL1
40	**Define source node.** Vertex with no entering edges is called the ***source*** and assumed to be numbered 1.	C213.4	BTL1
41	**Define sink node**. Vertex with no leaving edges is called the ***sink*** and assumed to be numbered *n*.	C213.4	BTL1
42	**What is edge capacity?** The weight *uij* of each directed edge *(i, j)* is a positive integer, called the edge ***capacity.***	C213.4	BTL1
43	**What is meant by feasible flow in maximum flow problem.** It is an assignment of real numbers Xij to edges i.j of a network that satisfy flow conservation constraints and the capacity constraints.	C213.4	BTL1
44	**Write the three important things in Ford-Fulkereson method.** 1.Residual network 2.Augmenting path 3.Cuts	C213.4	BTL1

45	**What is Augmenting path in maximum flow problem.?** The path which never violates the capacity constraints is called Augmenting path in maximum flow problem	C213.4	BTL1
46	**What is residual network?** A representation of a network with flow and capacity value for every node is called residual network**.**	C213.4	BTL1
47	**What is forward edge in maximum flow problem.** It is connected by a directed edge with some positive unused capacity so that we can increase the flow through that edge .	C213.4	BTL1
48	**Define st-cut** An st-cut is a cut that places s in one of its sets (Cs) and t in the other (Ct).	C213.4	BTL1
49	**Write the Maxflow / mincut applications** Data mining. • Open-pit mining. • Project selection. • Image processing. • Airline scheduling. • Bipartite matching.	C213.4	BTL1
50	**Write Ford-Fulkerson algorithm** Generic method for solving maxflow problem. • Start with 0 flow everywhere. • Find an augmenting path. • Increase the flow on that path, by as much as possible. • Repeat until no augmenting paths are left.	C213.4	BTL1

PART – B

Q. NO.	QUESTIONS	CO	BLOOM'S LEVEL
1	**Explain the maximum flow problem in detail. [APR/MAY 2015,MAY/JUNE 2016]** Refer page no 643 in Thomas H.Cormen	C213.4	BTL5
2	**Explain Maximum Matching in Bipartite Graphs. [APR/MAY 2015]** Refer page no 664 in Thomas H.Cormen	C213.4	BTL5
3	**Summarize the simplex method.APR/MAY 2015, MAY/JUNE 2016, APR/MAY 2017, NOV/DEC 2017, APR/MAY 2018,NOV/DEC 2018**	C213.4	BTL2
4	**Explain the Stable marriage problem APR/MAY 2015, NOV/DEC 2017, APR/MAY 2018,NOV/DEC 2018** Refer notes	C213.4	BTL5
5	**Apply the shortest augmenting path algorithm to the network.[MAY/JUNE 2016]**	C213.4	BTL3

	Refer notes		
6	**Explain briefly on bipartite perfect matching prototype.** Refer notes	C213.4	BTL5
7	**Explain the string matching algorithm for finding the pattern on a text and analyze the algorithm. APR?MAY 2017**	C213.4	BTL5
8	**Solve using Simplex method** (i) Maximize $p = 2x + 3y + z$ (8) subject to $x + y + z \le 40$ $2x + y - z \ge 10$ $-y + z \ge 10$ $x \ge 0, y \ge 0, z \ge 0.$	C213.4	BTL5

UNIT V

Q. No.	Questions	CO	Bloom's Level
1	**Define 0/1 Knapsack problem.** The solution to the Knapsack problem can be viewed as a result of sequence of decisions. We have to decide the value of x_i. x_i is restricted to have the value 0 or 1 and by using the function knap(l, j, y) we can represent the problem as maximum $\Sigma p_i x_i$ subject to $\Sigma w_i x_i \le y$ where l - iteration, j - number of objects, y – capacity	C213.5	BTL1
2	**What is the formula to calculate optimal solution in 0/1 Knapsack problem?** The formula to calculate optimal solution is $g_0(m)=\max\{g_1, g_1(m-w_1)+p_1\}$.	C213.5	BTL1
3	**Illustrate traveling salesperson problem.** Let g = (V, E) be a directed. The tour of G is a directed simple cycle that includes every vertex in V. The cost of a tour is the sum of the cost of the edges on the tour. The traveling salesperson problem to find a tour of minimum cost.	C213.5	BTL2
4	**List some applications of traveling salesperson problem.** Routing a postal van to pick up mail from boxes located at n different sites. Using a robot arm to tighten the nuts on some piece of machinery on an assembly line. Production environment in which several commodities are manufactured on the same set of machines	C213.5	BTL4

5	**Show the time complexity and space complexity of traveling salesperson problem.** Time complexity is $O(n^2 2^n)$. Space complexity is $O(n 2^n)$.	C213.5	BTL2
6	**Summarize the requirements that are needed for performing Backtracking?** i. To solve any problem using backtracking, it requires that all the solutions satisfy a complex set of constraints. ii. They are: Explicit constraints. Implicit constraints	C213.5	BTL2
7	**Define explicit constraint.** They are rules that restrict each x_i to take on values only from a give set. They depend on the particular instance I of the problem being solved. All tuples that satisfy the explicit constraints define a possible solution space.	C213.5	BTL1
8	**Define implicit constraint.** They are rules that determine which of the tuples in the solution space of I satisfy the criteria function. It describes the way in which the x_i must relate to each other.	C213.5	BTL1
9	**Define state space of the problem.** All the paths from the root of the organization tree to all the nodes is called as state space of the problem	C213.5	BTL1
10	**What are static trees?** The tree organizations that are independent of the problem instance being solved are called as static tree.	C213.5	BTL1
11	**What are dynamic trees?** The tree organizations those are independent of the problem instance being solved are called as static tree.	C213.5	BTL1
12	**Define a live node.** A node which has been generated and all of whose children have not yet been generated is called as a live node	C213.5	BTL1
13	**Define a E – node.** E – node (or) node being expanded. Any live node whose children are currently being generated is called as a E – node.	C213.5	BTL1
14	**Define a dead node.** Dead node is defined as a generated node, which is to be expanded further all of whose children have been generated	C213.5	BTL1
15	**List the factors that influence the efficiency of the backtracking algorithm?** The efficiency of the backtracking algorithm depends on the following four factors. They are: o The time needed to generate the next x_{BTL} o The number of x_k satisfying the explicit constraints. o The time for the bounding functions B_k o The number of x_k satisfying the B_k	C213.5	BTL4

16	**Define Branch-and-Bound method.** The term Branch-and-Bound refers to all the state space methods in which all children of the E-node are generated before any other live node can become the E- node.	C213.5	BTL1
17	**Define backtracking?** Depth first node generation with bounding function is called backtracking. The backtracking algorithm has its virtue the ability to yield the answer with far fewer than m trials.	C213.5	BTL1
18	**What is Hamiltonian cycle in an undirected graph? [APR/MAY 2015]** A Hamiltonian cycle is round trip along n edges of G that visits every vertex once and returns to its starting position.	C213.5	BTL1
19	**What is Feasible solution?** It is obtained from given n inputs Subsets that satisfies some constraints are called feasible solution. It is obtained based on some constraints	C213.5	BTL1
20	**What is optimal solution?** It is obtained from feasible solution. Feasible solution that maximizes or minimizes a given objective function It is obtained based on objective function	C213.5	BTL1
21	**List the application of backtracking technique?** The application of backtracking technique is 8-Queens problem	C213.5	BTL4
22	**Show the application for Knapsack problem?** The Knapsack problem is problem in combinatorial optimization. It derives its name from the maximum problem of choosing possible essential that can fit into one bag to be carried on a trip. A similar problem very often appears in business, combinatory, complexity theory, cryptography and applied mathematics.	C213.5	BTL2
23	**Define subset sum problem?** Subset sum problem is a problem, which is used to find a subset of a given set S={S1,S2,S3,…….Sn} of n positive integers whose sum is equal to given positive integer d.	C213.5	BTL1
24	**What is heuristic?** A heuristic is a common sense rule drawn from experience rather than from a mathematically proved assertion. For example, going to the nearest un visited city in the travelling salesman problem is good example for heuristic	C213.5	BTL1
25	**What is promising and non promising node? NOV/DEC 2017** A node in a state space tree is said to be promising, if it corresponds to a partially constructed solution that may still lead to a complete solution. Otherwise, a node is called non- promising.	C213.5	BTL1
26	**What are the additional items are required for branch and bound compare to backtracking technique?** Compared to backtracking, branch and bound requires 2 additional items. 1) A way to provide, for every node of a node of a state space tree, a bound on the best value of the objective function on any solution that can	C213.5	BTL1

	be obtained by adding further components to the partial solution represented by the node. **2)** The value of the best solution seen so far.		
27	**Compare backtracking and branch bound techniques.** Backtracking is applicable only to non optimization problems. Backtracking generates state space tree in depth first manner. Branch and bound is applicable only to optimization problem. Branch and bound generated a node of state space tree using best first rule.	C213.5	BTL2
28	**What are the searching techniques that are commonly used in Branch-and-Bound method.** The searching techniques that are commonly used in Branch-and-Bound method are: **i.** FIFO ii. LIFO iii. LC iv. Heuristic search	C213.5	BTL1
29	**Illustrate 8 – Queens problem.** The problem is to place eight queens on a 8 x 8 chessboard so that no two queen "attack" that is, so that no two of them are on the same row, column or on the diagonal.	C213.5	BTL2
30	**Show the purpose of lower bound.[MAY/JUNE 2016]** Lower bound of a problem is an estimate on a minimum amount of work needed to solve a given problem.	C213.5	BTL2
31	**Compare NP- hard and Np-complete problems? [APR/MAY 2015]** The problems whose solutions have computing times are bounded by polynomials of small degree.	C213.5	BTL2
32	**Define P and NP problem. APR/MAY 2017. NOV/DEC 18** In computational complexity theory, **P**, also known as PTIME or DTIME(n), is a fundamental complexity class. It contains all decision **problems** that can be solved by a deterministic Turing machine using a polynomial amount of computation time, or polynomial time. ***NP***: the class of decision problems that are solvable in polynomial time on a *nondeterministic* machine (or with a nondeterministic algorithm).(A *deterministic* computer is what we know).A *nondeterministic* computer is one that can "guess" the right answer or solution think of a nondeterministic computer as a parallel machine that can freely spawn ***an infinite number*** of processes	C213.5	BTL1
33	**Compare feasible solution and optimal solution.NOV/DEC 2017** Feasible solution means set which contains all the possible solution which follow all the constraints. An **optimal solution** is a feasible **solution** where the objective function reaches its maximum (or minimum) value – for example, the most profit or the least cost. A globally **optimal solution** is one where there are no other feasible **solutions** with better objective function values	C213.5	BTL1
34	**How is lower bound found by problem reduction? APR/MAY2018** If problem P is at least as hard as problem Q ,then a lower bound for Q is also a lower bound for P. Hence ,find problem Q with a known lower bound that can be reduced to problem P in question. then any algorithm that solves P will also solve Q	C213.5	BTL1
35	**What are tractable and non tractable problems ?APR?MAY 2018** **Problems** that are solvable by polynomial time algorithms as	C213.5	BTL1

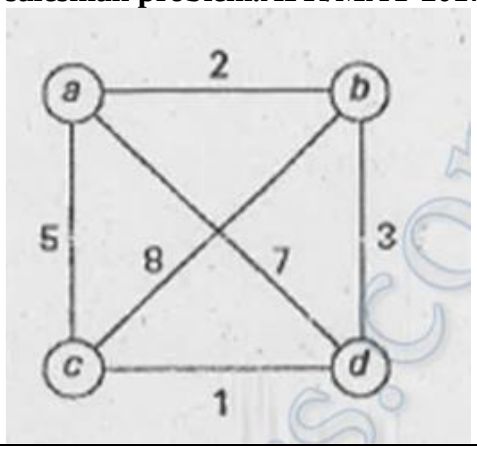
	being **tractable**, and **problems** that require super polynomial time as being **intractable**. • Sometimes the line between what is an 'easy' **problem** and what is a 'hard' **problem** is a fine one		
36	**Define state space tree.** The tree organization of the solution space is referred to as state space tree	C213.5	BTL1
37	**What is a decision problem?** Any problem for which the answer is either zero or one is called decision problem.	C213.5	BTL1
38	**What is maxclique problem?** A maxclique problem is the optimization problem that has to determine the size of a largest clique in Grapg G where clique is the maximal sub graph of a graph	C213.5	BTL1
39	**State m – colorability decision problem.** Let G be a graph and m be a given positive integer. We want to discover whether the nodes of G can be colored in such a way that no two adjacent nodes have the same color yet only m colors are used	C213.5	BTL1
40	**Define chromatic number of the graph.** The m – colorability optimization problem asks for the smallest integer m for which the graph G can be colored. This integer is referred to as the chromatic number of the graph.	C213.5	BTL1
41	**Define optimal finish time** Optimal finish time scheduling for a given set of tasks is a non preemptive schedule S for which F (S) is minimum over all non preemptive schedules S.	C213.5	BTL1
42	**Give an example of sub set sum problem.NOV/DEC 18** Input: set[] = {3, 34, 4, 12, 5, 2}, sum = 9 Output: True //There is a subset (4, 5) with sum 9.	C213.5	BTL1
43	**Define LIFO search** LIFO is the acronym for last-in, first-out. Under LIFO the latest or more recent costs of products purchased (or produced) are the first costs expensed as the cost of goods sold.	C213.5	BTL1
44	**What FIFO means?** FIFO" stands for first-in, first-out, meaning that the oldest inventory items are recorded as sold first but do not necessarily mean that the exact oldest physical object has been tracked and sold	C213.5	BTL1
45	**What is meant by approximation algorithm** Given an optimization problem P, an algorithm A is said to be an approximation algorithm for P, if for any given instance I, it returns an approximate solution, that is a feasible solution.	C213.5	BTL1
46	**What is randomized algorithm** A randomized algorithm is an algorithm that employs a degree of randomness as part of its logic. The algorithm typically uses uniformly random bits as an auxiliary input to guide its behavior, in the hope of achieving good performance in the "average case" over all possible choices of random bits.	C213.5	BTL1
47	**Define Optimization Problem** An optimization problem is the problem of finding the best solution from all feasible solutions. The objective may be either min. or max. depending on the problem considered	C213.5	BTL1
48	**Features of Heuristics algorithm** 1. Develop intuitive algorithms. 2. Guaranteed to run in polynomial time. 3. No guarantees on quality of solution.	C213.5	BTL1

49	**Features of Approximation algorithms** 1. Guaranteed to run in polynomial time. 2. Guaranteed to get a solution which is close to the optimal solution (near optimal).	C213.5	BTL1
50	**What is Absolute approximation?** A is an absolute approximation algorithm if there exists a constant k such that, for every instance I of P, $\|A^*(I) - A(I)\| \leq k$. I For example, Planar graph coloring.	C213.5	BTL1
51	**What is Relative approximation?** A is an relative approximation algorithm if there exists a constant k such that, for every instance I of P, $\max\{ A^*(I)\, A(I) , A(I)\, A^*(I) \} \leq k$. I Vertex cover.	C213.5	BTL1

.

PART – B

Q. NO.	QUESTIONS	CO	BLOOM'S LEVEL
1	**Explain how the branch and bound technique is used to solve Knapsack problem (OR) Implement an algorithm for Knapsack problem using NP-Hard Approach. [APRIL/MAY 2010, APR/MAY 2015]** Refer page no 404 in Anany Levitin	C213.5	BTL5
2	**Explain NP-hard and NP-completeness**. [**APRIL/MAY 2011]** Refer page no 369 in Anany Levitin	C213.5	BTL5
3	**Discuss the backtracking solution to solve 8-Queens problem, APR/MAY 2017** Refer page no 393 in Anany Levitin	C213.5	BTL6
4	**What is Hamiltonian problem? Explain with an example using backtracking? NOV/DEC 2017** Refer page no 395 in Anany Levitin	C213.5	BTL5
5	**Apply Branch and Bound Technique to solve travelling salesperson problem.** Refer page no 406 in Anany Levitin	C213.5	BTL3
6	**Apply Branch and Bound Technique to solve Assignment problem. Explain how job assignment problem could be solved, given n tasBTLs and n agents where each agent has a cost to complete each tasBTL, using Branch and Bound technique. [APR/MAY 2015], NOV/DEC 2017**	C213.5	BTL3

	Refer page no 402 in Anany Levitin		
7	**Apply approximation algorithm (nearest neighbour algorithm,multifrgment-heuristic algorithm)for travelling salesperson problem. Assume that the cost function satisfies the triangle inequality. [APR/MAY 2015], APR/MAY 2018** Refer Notes	C213.5	BTL3
8	**State the subset-sum problem and complete state space tree of the backtracking algorithm applied to the instance A={3,5,6,7} and d=15 of the subset-sum problem. [MAY/JUNE 2016]**	C213.5	BTL3
9	**Apply branch and bound algorithm to solve the following travelling salesman problem.APR/MAY 2017,** 	C213.5	BTL6
10	**Explain the methods for establishing lower bounds.NOV/DEC 2017**	C213.5	BTL1
11	**What is class NP? Discuss about any five problems for which no polynomial time algorithm has been found. APR/MAY 2018**	C213.5	BTL
12	**Discuss the approximation algorithm for NP hard problems. APR/ MAY 2017,NOV/DEC 2018**	C213.5	BTL1
13	**Write short notes on FIFO search**	C213.5	BTL1
14	**Explain in detail about LIFO Search**	C213.5	BTL1
15	**Consider the travelling salesperson instance defined by the following cost matrix** **∞ 20 30 10 11** **3 5 16 4 2** **19 6 18 ∞ 3** **16 4 7 16 ∞** **Draw the state space tree and show the reduced matrices corresponding to each of the node. Nov/dec 2018**	C213.5	BTL5