CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE - New Delhi. Affiliated to JNTUH and Accredited by NAAC & NBA)

COURSE INSTRUCTOR NAME: Mr.S Kiran Kumar **ACADEMIC YEAR**: 2023-24

SUBJECT NAME: Design and Analysis of Algorithms

EMAIL-ID: siliverikirankumar1990@cmrec.ac.in **CLASS ROOM NO**: B-218

CONTACT NO: 8919850117

SEM START DATE AND END DATE: 21-8-23 T0 23-12-23

CONTENTS OF COURSE FILE

15. Previous year question papers

16. Power point presentations (PPTs)

17. References (Textbook/Websites/Journals)

HOD

1. DEPARTMENT VISION & MISSION

Vision:

To produce globally competent and industry-ready graduates in Computer Science & Engineering by imparting quality education with the know-how of cutting-edge technology and holistic personality.

Mission:

1. To offer high-quality education in Computer Science & Engineering in order to build core competence for the graduates by laying a solid foundation in Applied Mathematics and program framework with a focus on concept building.

2. The department promotes excellence in teaching, research, and collaborative activities to prepare graduates for a professional career or higher studies.

3. Creating an intellectual environment for developing logical skills and problem-solving strategies, thus developing, an able and proficient computer engineer to compete in the current global scenario.

2. LIST OF PEOs, POs AND PSOs

2.1 Program Educational Objectives (PEO):

PEO 1: Excel in professional career and higher education by acquiring knowledge of mathematical computing and engineering principles.

PEO 2: To provide an intellectual environment for analyzing and designing computing systems for technical needs.

PEO 3: Exhibit professionalism to adapt current trends using lifelong learning with legal and ethical responsibilities.

PEO 4: To produce responsible graduates with effective communication skills and multidisciplinary practices to serve society and preserve the environment.

2.2. Program Outcomes (POs):

Engineering Graduates will be able to satisfy these NBA graduate attributes:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems
2. **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
3. **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
4. **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
5. **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
6. **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
7. **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
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings
10. **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
11. **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
12. **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

2.3 Program Specific Outcomes (PSOs):

PSO1: Professional Skills and Foundations of Software development: Ability to analyze, design and develop applications by adopting the dynamic nature of Software developments.

PSO2: Applications of Computing and Research Ability: Ability to use knowledge in cutting edge technologies in identifying research gaps and to render solutions with innovative ideas.

3. COURSE OUTCOMES

CO1 Understand the notion of an algorithm, asymptotic notations and divide and conquer, Analyze the recursive and non-recursive algorithms and divide and conquer.
CO2. Ability to choose appropriate data structures and algorithm design methods for a specified application
CO3. Understand the algorithm design techniques using dynamic programming.
CO4. Understand the algorithm design techniques using greedy method.
CO5. Ability to understand algorithm design techniques using backtracking, branch and bound and NP-complete and NP-hard problems

4. SYLLABUS COPY

UNIT - I

Introduction: Algorithm, Performance Analysis-Space complexity, Time complexity, Asymptotic Notations- Big oh notation, Omega notation, Theta notation and Little oh notation.

Divide and conquer: General method, applications-Binary search, Quick sort, Merge sort, Strassen’s matrix multiplication.

UNIT - II

Disjoint Sets: Disjoint set operations, union and find algorithms

Backtracking**:** General method, applications, n-queen’s problem, sum of subsets problem, graph coloring

UNIT - III

Dynamic Programming: General method, applications- Optimal binary search trees, 0/1 knapsack problem, All pairs shortest path problem, Traveling sales person problem, Reliability design.

UNIT - IV

Greedy method: General method, applications-Job sequencing with deadlines, knapsack problem, Minimum cost spanning trees, Single source shortest path problem.

UNIT - V

Branch and Bound: General method, applications - Travelling sales person problem, 0/1 knapsack problem - LC Branch and Bound solution, FIFO Branch and Bound solution.

NP-Hard and NP-Complete problems: Basic concepts, non deterministic algorithms, NP - Hard and NP-Complete classes, Cook's theorem.

TEXT BOOK:

1. Fundamentals of Computer Algorithms, Ellis Horowitz, Satraj Sahni and Rajasekharan, University Press.

REFERENCE BOOKS:

1. Design and Analysis of algorithms, Aho, Ullman and Hopcroft, Pearson education.

2. Introduction to Algorithms, second edition, T. H. Cormen, C.E. Leiserson, R. L. Rivest, and C.Stein, PHI Pvt. Ltd./ Pearson Education.

3. Algorithm Design: Foundations, Analysis and Internet Examples, M.T. Goodrich and R. Tamassia, John Wiley and sons.

5. SESSION PLAN/LESSON PLAN

S.NO	Topic (JNTU syllabus)	Sub-Topic	NO. OF LECTURES REQUIRED	Suggested Books	Teaching Methods
UNIT - I					
1	Introduction to Algorithms	Introduction To Algorithms	**L1**	**T1, T2**	**M1**
2		Performance Analysis-Space Complexity, Time Complexity	**L2,L3,L4**	**T1, T2**	**M1**
3		Asymptotic Notations- Big oh notation, Omega notation, Theta notation and Little oh notation.	**L5, L6, L7**	**T1**	**M1**
4		Divide and Conquer Algorithm	**L8**	**T1**	**M1**
5		Applications – Binary search, Merge sort, Quick sort, Strassen’s Matrix Multiplication.	**L9-L14**	**T1, R3**	**M2 (PPT)**
UNIT - II					
6	Disjoint Set	Disjoint set operations	**L15**	**T1**	**M1**
7		Union and find algorithms, AND/OR graphs	**L16-L17**	**T1**	**M1**
8		Backtracking-General method, applications	**L18**	**T1**	**M1**
9		The N-queen problem, sum of subsets problem, graph coloring	**L19-L24**	**T1, T2**	**M1**
UNIT –III					
10		Dynamic programming- general method	**L25**	**T1**	**M1**
11		Applications- Optimal binary search trees, 0/1 knapsack problem, All pairs shortest path problem,	**L26-L32**	**T1**	**M1**

12		Traveling sales person problem, Reliability design.	**L33, L37**	**T1**	**M2(PPT)**
UNIT –IV					
13	Greedy Method	Greedy method: General method	**L38**	**T1**	**M1**
14		Applications-Job sequencing with deadlines, knapsack problem,	**L39-L44**	**T1**	**M1**
15		Minimum cost spanning trees, Single source shortest path problem.	**L45-L49**	**T1**	**M1**
UNIT –V					
16	Branch and Bound	Branch and Bound	**L50**	**T1**	**M1**
17		Applications-Travelling sales person problem, 0/1 knapsack problem - LC Branch and Bound solution, FIFO Branch and Bound solution	**L51-L58**	**T1**	**M1**
18		NP-Hard and NP-Complete problems	**L59-L60**	**T1**	**M1**

METHODS OF TEACHING:

M1 : Lecture Method	**M4 : Presentation /PPT**	**M7 : Assignment**
M2 : Demo Method	**M5 : Lab/Practical**	**M8 : Industry Visit**
M3 : Guest Lecture	**M6 : Tutorial**	**M9 : Project Based**

NOTE:

1. Any Subject in a Semester is suppose to be completed in 55 to 65 periods.
2. Each Period is of 50 minutes.
3. Each unit duration & completion should be mentioned in the Remarks Column.
4. List of Suggested books can be marked with Codes like T1, T2, R1, R2 etc.

6. INDIVIDUAL TIME TABLE (SILIVERI KIRAN KUMAR)

Mr.S.Kiran kumar

	I(9:10-10:10)	II(10:10-11:00)	III(11:00-11:50)	IV(11:50-12:40)		V(1:20-2:20)	VI(2:20-3:10)	VII(3:10-4:00)
MON			DAA-III-A				DAA-III-C	
TUE								
WED								
THU			DAA-III-C					DAA-III-A
FRI	DAA-III-C		DAA-III-C					
SAT		DAA-III-A	→ DAA LAB-III-C ←					DAA-III-A

7. Session Execution Log:

S no	Units	Scheduled started date	Completed date	Remarks
1	I	21/8/23	21/9/23	completed
2	II	23/9/23	4/10/23	completed
3	III	5/10/23	28/11/23	completed
4	IV	29/11/23	12/12/23	completed
5	V	12/12/23	20/12/23	completed

8. Lecture Notes – (hand written)

FIVE UNIT'S HANDWRITTEN and SOFTCOPY PRINT Attached

9. ASSIGNMENT QUESTIONS ALONG SAMPLE ASSIGNMENT SCRIPTS

CMR ENGINEERING COLLEGE
UGC Autonomous

(Accredited by NBA, Approved by AICTE, Affiliated to JNTU, Hyderabad)
KANDLAKOYA (V), MEDCHAL ROAD, HYDERABAD-501401.
Ph: 08418 200037, 92470 22662, Fax: 08418 200240, www.cmrec.org.

Department of Computer Science and Technology

III.B.TECH I SEM ASSIGNMENT-I QUESTIONS
Subject: Design and Analysis of Algorithms BRANCH: CSE

Answer all the five questions

1. Explain the designing of Algorithms? **[C01]**
2. Explain about Merge Sort?**[CO1]**
3. Define the disjoint sets and explain the disjoint set, find the union operation**.[C02]**
4. a)What is dynamic programming list out the applications of dynamic programming**.[CO2]**

   b)Explain the 0/1 knapsack problem.[**CO3]**
5. Define the OBST. Explain the algorithm of OBST and analysis. **[C03]**

CMR ENGINEERING COLLEGE
UGC Autonomous

(Accredited by NBA, Approved by AICTE, Affiliated to JNTU, Hyderabad)
KANDLAKOYA (V), MEDCHAL ROAD, HYDERABAD-501401.
Ph: 08418 200037, 92470 22662, Fax: 08418 200240, www.cmrec.org.

Department of Computer Science and Technology

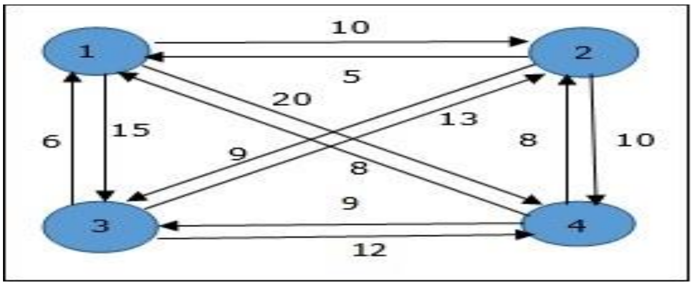
III.B.TECH I SEM ASSIGNMENT-II QUESTIONS

Subject: Design and Analysis of Algorithms BRANCH: CSE

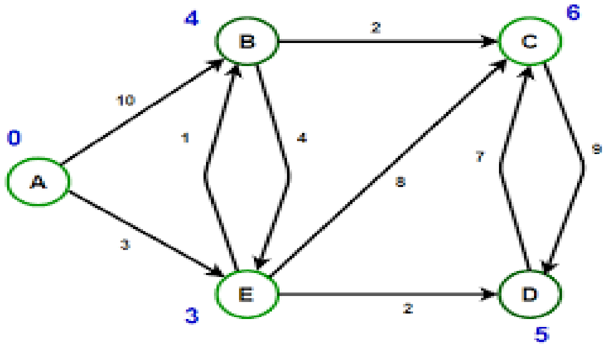
Answer all the five questions

1. Apply dynamic programming to obtain optimal binary search tree solution for the identifier set (a1, a2, a3, a4)=(switch, for, int, while) with (p1, p2, p3, p4)=(3, 4, 2, 1), (q0, q1, q2,q3, q4)=(5, 2, 5, 1, 2) and also write algorithm for its construction. **[CO3]**

2. Describe travelling sales person problem. Find the minimum cost tour for the following graph using dynamic programming, the cost in adjacency is available of the problem (graph). **[CO3]**



3. Generate the solution for Single Source Shortest Path Problem by considering the given graph. Treat A is a starting vertex. **[CO4]**



4. Describe about reliability design with an example. **[CO3]**

5. a) Find the optimal solution for following 0/1 knapsack instance and construct state space tree using FIFO BB approach for the given n=5, Knapsack capacity m=12,

(W1, W2,W3, W4,W5)=(4,8,2,6,2)

(P1, P2, P3, P4, P5)=(22, 42, 30, 26, 61)

[CO5]

b) State and explain cook's theorem? **[CO5]**

10. MID EXAM QUESTION PAPER ALONG SAMPLE ANSWER SCRIPTS

MID I EXAMINATION

CMR
ENGINEERING COLLEGE
EXPLORE TO INVENT

CMR ENGINEERING COLLEGE
UGC AUTONOMOUS
(Approved by AICTE - New Delhi. Affiliated to JNTUH and Accredited by NAAC & NBA)
Kandlakoya (V), Medchal (M), Medchal - Malkajgiri (D)-501401

NBA
NATIONAL BOARD OF ACCREDITATION

NAAC

III-B.TECH I-SEM MID EXAMINATIONS

SET - 3

Date: 30 -10-2023 **Time: 10 to 11.30 am**

Subject: DAA **Branch: CSE/IT/CS/DS**

Note: Question paper contains two parts, Part-A and Part- B.

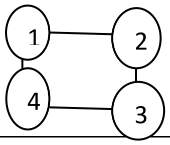
Part-A is compulsory which carries 10 marks. Attempt all questions in part-A.

Part-B- Attempt one questions from each units. Each question carries5 marks.

PART-A **5 x 2 M = 10 M**

		CO
1.	Write definitions of Time and Space Complexity.	**01**
2.	What do you understand by Algorithm and define its characteristics?	**01**
3.	Define the Backtracking, what are the problems over which this can be apply?	**02**
4.	What is cyclic graph? Give an example.	**02**
5.	What is Dynamic programming?	**02**

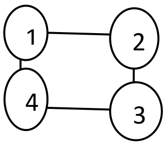
PART-B 3 x5 M = 15 M

		CO
6.	Write Binary Search algorithm. Further, find number 64, from the list 6, 10, 45, 64, 70, 89, 92, 97, 99 using Binary Search algorithm?	**01**
	or	
7.	What is Divide and Conquer technique? Explain with an example.	**01**
8.	Draw the state-space tree along with answer nodes for 4-queens problem.	**02**
	or	
9.	What is sum-of-subsets problem? Find all possible solution for given elements whose weights are (w1,w2,w3,w4,w5)=(2,3,5,6,8,10) and m=10. Draw its State-Space Tree.	**02**
10.	Write Graph coloring algorithm. Draw the state space tree for the following graph. 	**03**
	or	
11.	Solve the instance of 0/1 knapsack problem using Dynamic Programming given weights = {1,3,4,5} and profit = {1, 4, 5, 7} having Knapsack weight = 7 and n=4.	**03**

SCHEME OF EVALUATION

PART-A

SNO	THEORY	MARKS	TOTAL
1	Write definitions of Time and Space Complexity.	2	2
2	What do you understand by Algorithm and define its characteristics?	2	2
3	Define the Backtracking, what are the problems over which this can be apply?	2	2
4	What is cyclic graph? Give an example.	2	2
5	What is Dynamic programming?	2	2

SNO	THEORY	MARKS	TOTAL
6	Write Binary Search algorithm. Further, find number 64, from the list 6, 10, 45, 64, 70, 89, 92, 97, 99 using Binary Search algorithm?	**5**	**5**
7	What is Divide and Conquer technque? Explain with an example.	**5**	**5**
8	Draw the state-space tree along with answer nodes for 4-queens problem.	**5**	**5**
9	What is sum-of-subsets problem? Find all possible solution for given elements whose weights are (w1,w2,w3,w4,w5)=(2,3,5,6,8,10) and m=10. Draw its State-Space Tree.	**5**	**5**
10	Write Graph coloring algorithm. Draw the state space tree for the following graph. 	**5**	**5**
11	Solve the instance of 0/1 knapsack problem using Dynamic Programming given weights = {1,3,4,5} and profit = {1, 4, 5, 7} having Knapsack weight = 7 and n=4.	**5**	**5**

MID II EXAMINATION

CMR ENGINEERING COLLEGE

EXPLORE TO INVENT

UGC AUTONOMOUS

(Approved by AICTE - New Delhi. Affiliated to JNTUH and Accredited by NAAC & NBA)

Kandlakoya (V), Medchal (M), Medchal - Malkajgiri (D)-501401

III-B.TECH I-SEM MID II EXAMINATIONS

SET - 3

Date: 26 -12-2023 **Time: 10 to 11.30 am**

Subject: DAA **Branch: CSE/IT/CS/DS**

Note: Question paper contains two parts, Part-A and Part- B.

Part-A is compulsory which carries 10 marks. Attempt all questions in part-A.

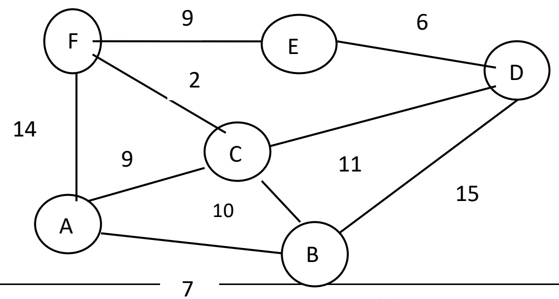
Part-B Attempt one questions from each units. Each question carries5 marks.

PART-A 5 x 2 M = 10 M

		CO
1.	List the difference between back tracking and branch & bound.	**01**
2.	Describe Floyd-Warshall Algorithm.	**01**
3.	Define optimal binary search tree with example.	**02**
4.	Define Greedy method.	**02**
5.	Define DijkstraAlgorithm.	**02**

PART-B 3 x5 M = 15 M

		CO
6.	Write the algorithm to construct a spanning tree using Kruskal's algorithm with an example.	**01**
	or	

<table>
<tr><td><b>7.</b></td><td>Find Least Cost (LC) branch and bound of the following with given total weight of knapsack is 15 and number of items i.e. n=4.
<table><tr><td><b>Profit</b></td><td>10</td><td>10</td><td>12</td><td>18</td></tr><tr><td><b>Weight</b></td><td>2</td><td>4</td><td>6</td><td>9</td></tr></table></td><td><b>01</b></td></tr>
<tr><td><b>8.</b></td><td>Explain Traveling sales person problem using Dynamic programming with an example.</td><td><b>02</b></td></tr>
<tr><td></td><td><b>or</b></td><td></td></tr>
<tr><td><b>9.</b></td><td>1) Find the Reliability design for the following:
<table><tr><th>Di</th><th>Ci</th><th>Ri</th></tr><tr><td>D1</td><td>30</td><td>0.9</td></tr><tr><td>D2</td><td>15</td><td>0.8</td></tr><tr><td>D3</td><td>20</td><td>0.5</td></tr></table>Given budget is 105.</td><td><b>02</b></td></tr>
<tr><td><b>10.</b></td><td>Single source shortest path for the following graph.


F
E
D
C
A
B
9
6
2
14
9
11
10
15
7

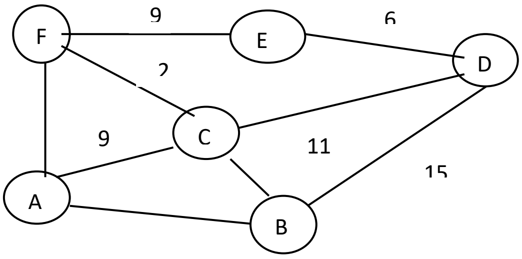
</td><td><b>03</b></td></tr>
<tr><td></td><td><b>or</b></td><td></td></tr>
<tr><td><b>11.</b></td><td>a) State and prove Cook’s theorem.

b) What are NP-Hard and NP-complete?</td><td><b>03</b></td></tr>
</table>

PART-A

SNO	THEORY	MARKS	TOTAL
1	List the difference between back tracking and branch & bound.	2	2
2	Describe Floyd-Warshall Algorithm.	2	2
3	Define optimal binary search tree with example.	2	2
4	Define Greedy method.	2	2
5	Define DijkstraAlgorithm.	2	2

<table>
<tr><th>SNO</th><th>THEORY</th><th>MARKS</th><th>TOTAL</th></tr>
<tr><td>6</td><td>Write the algorithm to construct a spanning tree using Kruskal’s algorithm with an example.</td><td>5</td><td>5</td></tr>
<tr><td></td><td>or</td><td></td><td></td></tr>
<tr><td>7</td><td>Find Least Cost (LC) branch and bound of the following with given total weight of knapsack is 15 and number of items i.e. n=4.
<table>
<tr><td><b>Profit</b></td><td>10</td><td>10</td><td>12</td><td>18</td></tr>
<tr><td><b>Weight</b></td><td>2</td><td>4</td><td>6</td><td>9</td></tr>
</table></td><td>5</td><td>5</td></tr>
<tr><td>8</td><td>Explain Traveling sales person problem using Dynamic programming with an example.
.</td><td>5</td><td>5</td></tr>
<tr><td></td><td>or</td><td></td><td></td></tr>
</table>

<table>
<tr><td>9</td><td>2) Find the Reliability design for the following:
<table><tr><th>Di</th><th>Ci</th><th>Ri</th></tr><tr><td>D1</td><td>30</td><td>0.9</td></tr><tr><td>D2</td><td>15</td><td>0.8</td></tr><tr><td>D3</td><td>20</td><td>0.5</td></tr></table>Given budget is 105.</td><td>5</td><td>5</td></tr>
<tr><td>10</td><td>Single source shortest path for the following graph.
 </td><td>5</td><td>5</td></tr>
<tr><td></td><td><b>or</b></td><td></td><td></td></tr>
<tr><td>11</td><td>a) State and prove Cook’s theorem.
b) What are NP-Hard and NP-complete?</td><td>5</td><td>5</td></tr>
</table>

11. Scheme of evaluation

MID -I

PART-A

SNO	THEORY	MARKS	TOTAL
1	What do you understand by Algorithm and define its characteristics?	2	2
2	Write definitions of Time and Space Complexity.	2	2
3	Define the Backtracking, what are the problems over which this can be apply?	2	2
4	Explain set representation using trees with an example.	2	2
5	Find an element from the tree using Collapsing rule with an example?	2	2

PART-B

SNO	THEORY	MARKS	TOTAL
6	Write Binary Search algorithm. Further, to apply the same to find items present 64 and 85, in number of comparisons, from the list 6, 10, 45, 64, 70, 89, 92, 97, 99?	5	5
7	What is an asymptotic notation? Explain different types of asymptotic notations with examples?	5	5
8	Draw the state-space tree along with answer nodes for 4-queens problem. Briefly explain n-queen problem using backtracking.	5	5
9	What is sum-of-subsets problem? Find all possible solution for given elements whose weights are (w1,w2,w3,w4,w5,w6,w7)=(4,5,7,8,11,12,15) and m=20. Draw its State-Space Tree.	5	5

10	Explain the methodology of Dynamic Programming. List the applications of Dynamic Programming.	**5**	**5**
11	What is the principle difference between the divide and conquer technique and dynamic programming technique.	**5**	**5**

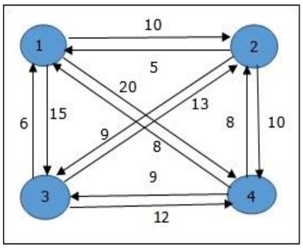
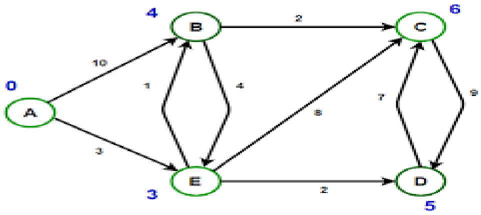
MID -II

PART-A

SNO	THEORY	MARKS	TOTAL
1	What do you understand by Dynamic Programming (DP)? Where are applications?	**2**	**2**
2	What is the significance of Reliability Design?	**2**	**2**
3	Write the functional difference of Back tracking and Branch & Bound.	**2**	**2**
4	Compare the Greedy and Dynamic Approach, write at least 5 differences.	**2**	**2**
5	Define the following terminologies: P, NP, NP-Hard and NP-Complete.	**2**	**2**

PART-B

SNO	THEORY	MARKS	TOTAL
6	Apply dynamic programming to obtain optimal binary search tree for the identifier set (a1, a2, a3, a4)=(cin, for, int, while) with (p1, p2, p3, p4)=(1, 4, 2, 1), (q0, q1, q2, q3, q4)=(4, 2, 4, 1, 1) and also write algorithm for its construction.	**5**	**5**

7	Describe travelling sales person problem. Find the minimum cost tour for the following graph using dynamic programming, the cost in adjacency is available of the problem (graph).  1 2 3 4 10 5 20 13 6 15 9 8 8 10 9 12	**5**	**5**
8	Suppose we have a Knapsack M=10 and n=4 number of objects we need to fill where each objects profits P (P1,P2,P3,P4)=(1,2,5,6) and weights W (W1,W2,W3,W4)=(2,4,3,6). Then find an optimal solution using Greedy Approach.	**5**	**5**
9	Generate the solution for Single Source Shortest Path Problem by considering the given graph. Treat A is a starting vertex.  A B C D E 0 4 6 3 5 10 3 2 1 4 8 7 9 2	**5**	**5**
10	How branch and bound method works for finding the solution of FIFO, LIFO and LC.	**5**	**5**
11	State and explain cook's theorem?	**5**	**5**

12. Mapping of Course Objectives, Course Outcomes with PEOs and Pos

CO1 Understand the notion of an algorithm, asymptotic notations and divide and conquer, Analyze the recursive and non-recursive algorithms and divide and conquer.

CO2. Ability to choose appropriate data structures and algorithm design methods for a specified application

CO3. Understand the algorithm design techniques using dynamic programming.

CO4. Understand the algorithm design techniques using greedy method.

CO5. Ability to understand algorithm design techniques using backtracking, branch and bound and NP-complete and NP-hard problems

Course Outcomes	**Relationship of Course outcomes to Program Outcomes (PO AVG)**													
CO/PO	**PO1**	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**
CO1	3	3	1	-	-	-	-	-	-	-	-	1	3	-
CO2	2	3	3	-	-	-	-	-	-	-	-	1	3	2
CO3	2	2	3	-	-	-	-	-	-	-	-	1	3	-
CO4	2	2	3	-	-	-	-	-	-	-	-	1	3	3
CO5	2	2	3	-	-	-	-	-	-	-	-	1	3	3

13. Cos,POs,PSOs JUSTIFICATION

CO No.	PO/PSO	CL	Justification
CO1	PO1	3	Strongly mapped as students will be able to gain the knowledge of Asymptotic notations, divide and conquer, recursive and non-recursive algorithms and theory of backward substitution in divide and conquer technique.
	PO2	3	Strongly mapped as students will be able to analyze the algorithms of divide and conquer.
	PO3	1	Slightly mapped as students will be able to design new algorithms using divide and conquer technique.
	PO4	1	Slightly mapped as students will be able to choose the appropriate algorithms such as merge and quick sort to design the applications to solve complex problems.
	PO5	1	Slightly mapped as students will be able use tools such as eclipse, netbeans to design and develop applications.
	PO12	1	Slightly mapped as students will be able to apply the concept of divide and conquer technique in searching and sorting etc. problem types.
	PSO1	3	Strongly mapped as students will be able to apply the searching and sorting techniques in real world problems.
	PSO2	1	Slightly mapped as students will be able to apply the concept of algorithms in system software such as compilers and debuggers.
CO2	PO1	2	Moderately mapped as students will be able to gain the knowledge of general plan of Disjoint sets
	PO2	3	Strongly mapped as students will be able to analyze the sets and pointer representation.
	PO3	1	Slightly mapped as students will be able to design new algorithms with appropriate data structures.
	PO4	1	Slightly mapped as students will be able to find the elements from sets.
	PO5	1	Slightly mapped as students will be able to perform find and union operations
	PO12	1	Slightly mapped as students will be able to do set representations
	PSO1	3	Strongly mapped as students will be able to find the elements from the sets
	PSO2	1	Slightly mapped as students will be able to apply the concept of Disjoint sets

CO3	PO1	2	Moderately mapped as students will be able to gain the knowledge of dynamic method concepts.
	PO2	2	Moderately mapped as students will be able to analyze the time and space complexity of dynamic programming algorithms.
	PO3	3	Strongly mapped as students will be able to design new algorithms using dynamic programming technique.
	PO4	1	Slightly mapped as students will be able to choose the dynamic algorithms in design of applications to solve complex problems.
	PO5	1	Slightly mapped as students will be able use tools such as eclipse, netbeans to design and develop applications.
	PO12	1	Slightly mapped as students will be able to analyze the efficiency of any new algorithm designed by dynamic programming techniques.
	PSO1	3	Strongly mapped as students will be able to apply the greedy techniques in real world problems such as TSP, reliability design.
	PSO2	1	Slightly mapped as students will be able to apply the concept of algorithms in system software such as compilers and debuggers.
CO4	PO1	2	Moderately mapped as students will be able to gain the knowledge of greedy method concepts.
	PO2	2	Moderately mapped as students will be able to analyze the time and space complexity of greedy algorithms.
	PO3	3	Strongly mapped as students will be able to design new algorithms using greedy technique.
	PO4	1	Slightly mapped as students will be able to choose the greedy algorithms in design of applications to solve complex problems.
	PO5	1	Slightly mapped as students will be able use tools such as eclipse, netbeans to design and develop applications.
	PO12	1	Slightly mapped as students will be able to analyze the efficiency of any new algorithm designed by greedy techniques.
	PSO1	3	Strongly mapped as students will be able to apply the greedy techniques in real world problems such as Resource management in software.
	PSO2	1	Slightly mapped as students will be able to apply the concept of algorithms in system software such as compilers and debuggers.

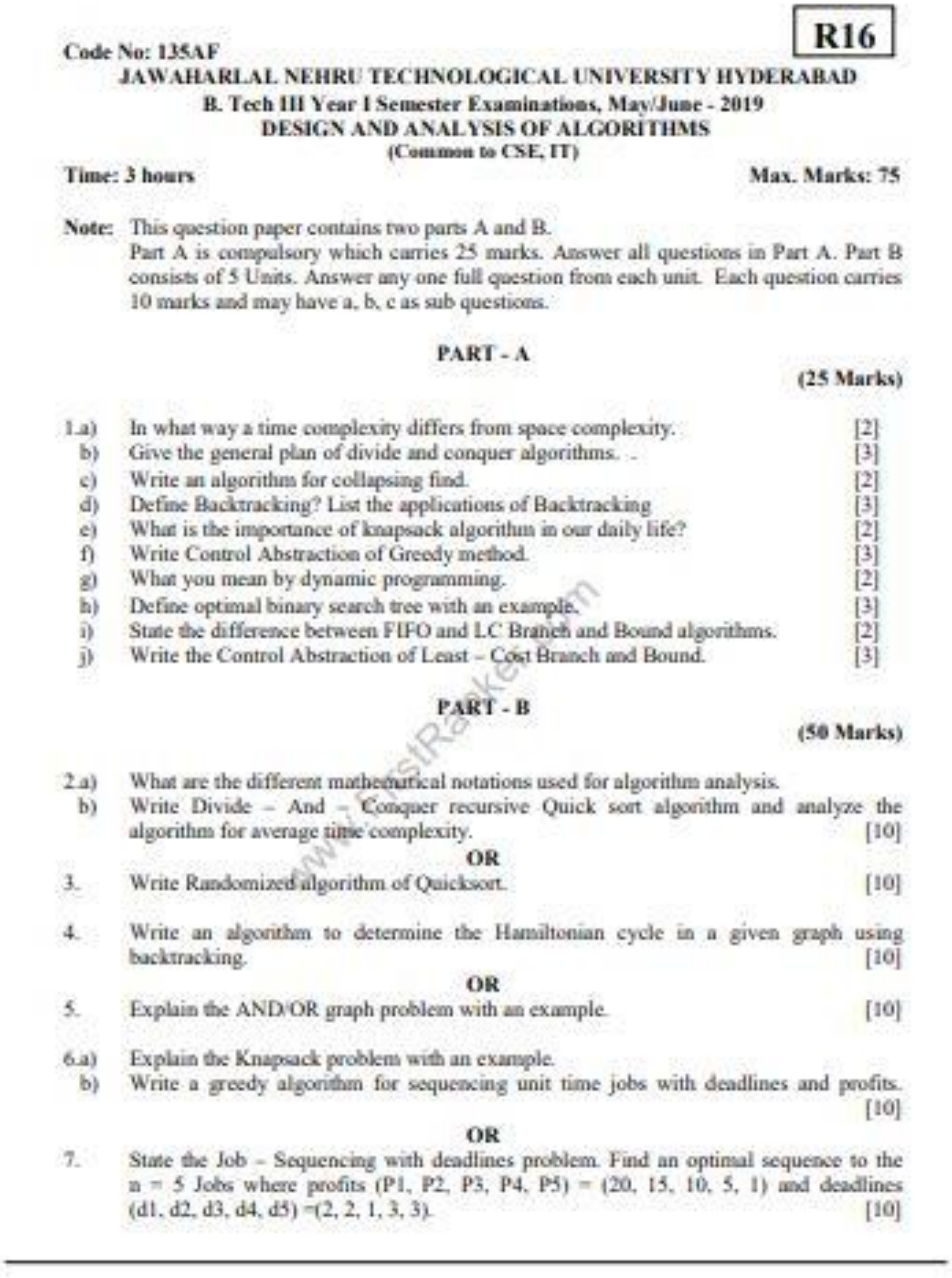
CO5	PO1	2	Moderately mapped as students will be able to gain the knowledge of backtracking, branch and bound concepts.
	PO2	2	Moderately mapped as students will be able to analyze the time and space complexity of backtracking, branch and bound algorithms.
	PO3	3	Strongly mapped as students will be able to design new algorithms using backtracking, branch and bound technique.
	PO4	1	Slightly mapped as students will be able to choose the backtracking andbranch & bound algorithms in design of applications to solve complex problems.
	PO5	1	Slightly mapped as students will be able use tools such as eclipse, net beans to design and develop applications.
	PO12	1	Slightly mapped as students will be able to analyze the efficiency of any new algorithm designed by backtracking, branch and bound techniques.
	PSO1	3	Strongly mapped as students will be able to apply the backtracking, branch and bound in real world problems such as event scheduling, stratification of triangle meshes in computer graphics.
	PSO2	1	Slightly mapped as students will be able to apply the concept of algorithms in system software such as compilers and debuggers.

14. ATTAINMENT OF CO’s, PO’s AND PSO’s (EXCEL SHEET): [Attached After Result]

Relationship of Course outcomes to Program Outcomes (PO AVG)													
PO1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO1 0	PO1 1	PO1 2	PSO 1	PSO 2

ASSESSMENT OF POs THROUGH THE COURSE			
PO	**CO**	**Value**	**AVG PO (Univ)**
PO1	CO1		
	CO3		
	CO4		
	CO5		
	CO6		
PO2	CO1		
	CO2		
	CO3		
	CO4		
	CO5		
PO3	CO1		
	CO2		
	CO3		
	CO4		
	CO5		
	CO6		
PO4	CO2		
	CO3		
	CO6		
PO5	CO5		
PO12	CO2		
PSO1	CO2		
	CO3		
PSO2	CO1		
	CO2		
	CO3		
	CO4		
	CO6		

15. University Question Papers/Model Question paper



Code No: 135AF

R16

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

B. Tech III Year I Semester Examinations, May/June - 2019

DESIGN AND ANALYSIS OF ALGORITHMS

(Common to CSE, IT)

Time: 3 hours **Max. Marks: 75**

Note: This question paper contains two parts A and B.
Part A is compulsory which carries 25 marks. Answer all questions in Part A. Part B consists of 5 Units. Answer any one full question from each unit. Each question carries 10 marks and may have a, b, c as sub questions.

PART - A

(25 Marks)

1.a) In what way a time complexity differs from space complexity. [2]
b) Give the general plan of divide and conquer algorithms. . [3]
c) Write an algorithm for collapsing find. [2]
d) Define Backtracking? List the applications of Backtracking [3]
e) What is the importance of knapsack algorithm in our daily life? [2]
f) Write Control Abstraction of Greedy method. [3]
g) What you mean by dynamic programming. [2]
h) Define optimal binary search tree with an example. [3]
i) State the difference between FIFO and LC Branch and Bound algorithms. [2]
j) Write the Control Abstraction of Least – Cost Branch and Bound. [3]

PART - B

(50 Marks)

2.a) What are the different mathematical notations used for algorithm analysis.
b) Write Divide – And – Conquer recursive Quick sort algorithm and analyze the algorithm for average time complexity. [10]

OR

3. Write Randomized algorithm of Quicksort. [10]

4. Write an algorithm to determine the Hamiltonian cycle in a given graph using backtracking. [10]

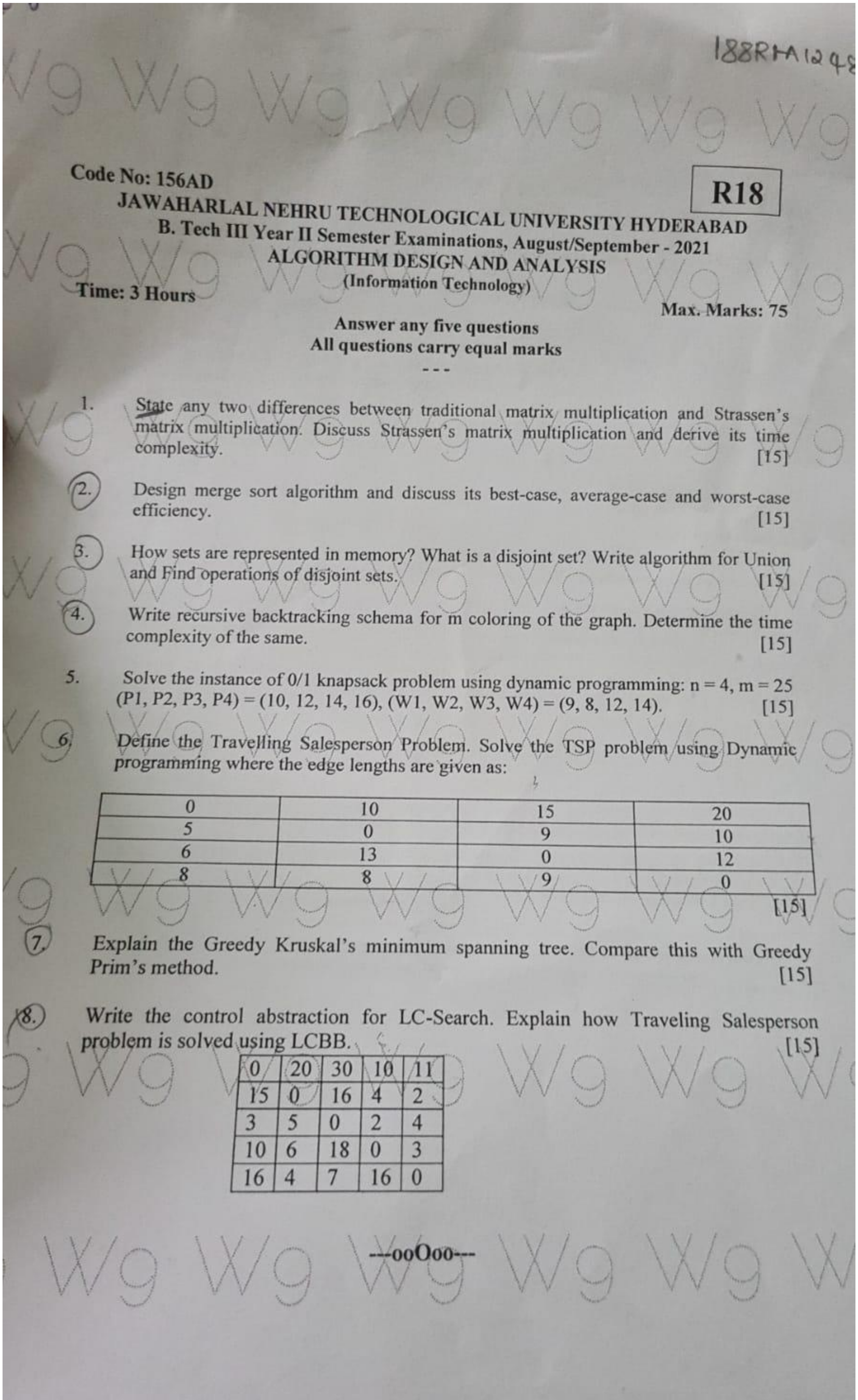
OR

5. Explain the AND/OR graph problem with an example. [10]

6.a) Explain the Knapsack problem with an example.
b) Write a greedy algorithm for sequencing unit time jobs with deadlines and profits. [10]

OR

7. State the Job - Sequencing with deadlines problem. Find an optimal sequence to the n = 5 Jobs where profits (P1, P2, P3, P4, P5) = (20, 15, 10, 5, 1) and deadlines (d1, d2, d3, d4, d5) =(2, 2, 1, 3, 3). [10]



188R1A124

Code No: 156AD

R18

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

B. Tech III Year II Semester Examinations, August/September - 2021

ALGORITHM DESIGN AND ANALYSIS

(Information Technology)

Time: 3 Hours **Max. Marks: 75**

Answer any five questions

All questions carry equal marks

- - -

1. State any two differences between traditional matrix multiplication and Strassen's matrix multiplication. Discuss Strassen's matrix multiplication and derive its time complexity. [15]

2. Design merge sort algorithm and discuss its best-case, average-case and worst-case efficiency. [15]

3. How sets are represented in memory? What is a disjoint set? Write algorithm for Union and Find operations of disjoint sets. [15]

4. Write recursive backtracking schema for m coloring of the graph. Determine the time complexity of the same. [15]

5. Solve the instance of 0/1 knapsack problem using dynamic programming: n = 4, m = 25 (P1, P2, P3, P4) = (10, 12, 14, 16), (W1, W2, W3, W4) = (9, 8, 12, 14). [15]

6. Define the Travelling Salesperson Problem. Solve the TSP problem using Dynamic programming where the edge lengths are given as:

0	10	15	20
5	0	9	10
6	13	0	12
8	8	9	0

[15]

7. Explain the Greedy Kruskal's minimum spanning tree. Compare this with Greedy Prim's method. [15]

8. Write the control abstraction for LC-Search. Explain how Traveling Salesperson problem is solved using LCBB. [15]

0	20	30	10	11
15	0	16	4	2
3	5	0	2	4
10	6	18	0	3
16	4	7	16	0

---ooOoo---

Code No.: CS501PC

R20 H.T.No.

		8	R						

CMR ENGINEERING COLLEGE: : HYDERABAD
UGC AUTONOMOUS
III–B.TECH–I–Semester End Examinations (Regular) - December- 2022
DESIGN AND ANALYSIS OF ALGORITHMS
(Common for CSE, IT, CSC, CSD)

[Time: 3 Hours] **Max. Marks: 70]**

Note: This question paper contains two parts A and B.
Part A is compulsory which carries 20 marks. Answer all questions in Part A.
Part B consists of 5 Units. Answer any one full question from each unit. Each question carries 10 marks.

PART-A (20 Marks)

1. a) Estimate the time and space complexity for the following code. [2M]

```
for(i=1; i<=n; i++)
{
 for(j=1;j<=i; j++)
 {
   for(k=1;k<=100;k++)
   {
       Printf ("India");  } } }
```

b) What is the best, worst and average case time complexity of quick sort and merge sort? [2M]

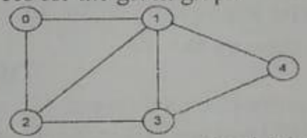
c) Write an algorithm to perform Union operation on disjoint sets? [2M]

d) List applications of backtracking? [2M]

e) State Principle of Optimality? [2M]

f) What does dynamic programming have in common with divide-and-conquer, and what is the principal difference between the two techniques. [2M]

g) Find all possible spanning trees for the given graph [2M]



h) List out the various constraints to prepare an optimal Job schedule? [2M]

i) List the differences between back tracking and branch and bound. [2M]

j) Define P and NP problems? [2M]

PART-B (50 Marks)

2. a) List and explain various asymptotic notations with suitable examples? [5M]

b) Compute the time complexity for Strassen's Matrix Multiplication. [5M]

OR

3. a) Write non-recursive algorithm to search an element using binary search. [5M]

b) Consider the following array elements [5M]
77,32,45,12,88,66,10,8 arrange them in sorted order using Quick sort

4. a) Discuss briefly about collapsing find algorithm with suitable example. [5M]

b) Construct space tree for the following sum of subsets problem using backtracking for the data (w1,w2,w3,w4,w4)=(1,2,5,6,8) and M=9 [5M]

OR

5. a) Construct state space tree for placing 4 queens in the 4X4 chess board. [5M]

b) Design an algorithm to assign the colors to vertices using backtracking. [5M]

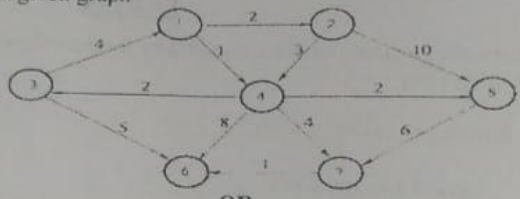
6. Design a 3-stage system with device types D1, D2,D3 whose costs are Rs.35,Rs.20,Rs.10 respectively, the cost of the entire system is Rs.100,and the reliabilities are 0.9,0.8,0.75 [10M]

OR

7. a) Consider the 0/1 Knapsack instance with 5 objects and a capacity M=11, Profits P=(5,4,7,2,3) and weights W=(4,3,6,2,2). Solve it using Dynamic Programming. [5M]

b) Write an algorithm to implement all pairs shortest path problem. [5M]

8. List the shortest paths from source 1 to remaining all the nodes using Dijkstra's algorithm in a given graph [10M]



OR

9. a) Construct optimal Job schedule for the following n=6,
(p1,p2,p3,p4,p5,p6) = (10,20,30,15,25,35)
and (d1,d2,d3,d4,d5,d6)=(2,3,4,1,2,1) [5M]

b) Write the algorithm to construct a spanning tree using Kruskal's approach with example? [5M]

10. Find the optimal solution for following 0/1 knapsack instance and construct state space tree using FIFO BB approach for the given n =5,
(W1 W2 W3 W4 W5) =(4,8,2,6,1)
(P1 P2 P3 P4 P5) = (12,32,40,30,50)
Knapsack capacity m = 10 [10M]

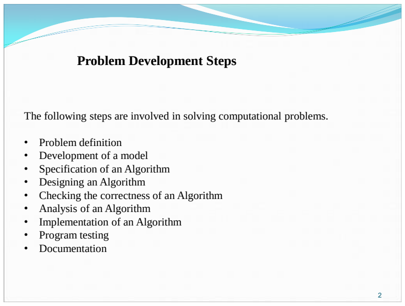
OR

11. a) State and prove Cook's theorem. [5M]

b) Illustrate NP-Hard and NP-complete problems. [5M]

****

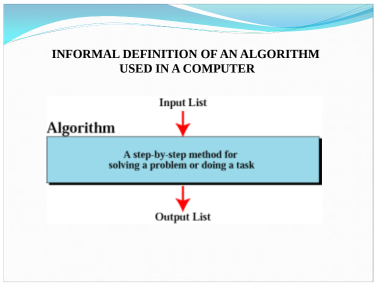
16.PPTS AND PRESENTATIONS



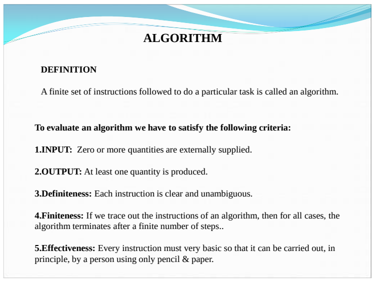
Problem Development Steps

The following steps are involved in solving computational problems.

- Problem definition
- Development of a model
- Specification of an Algorithm
- Designing an Algorithm
- Checking the correctness of an Algorithm
- Analysis of an Algorithm
- Implementation of an Algorithm
- Program testing
- Documentation



INFORMAL DEFINITION OF AN ALGORITHM USED IN A COMPUTER



ALGORITHM

DEFINITION

A finite set of instructions followed to do a particular task is called an algorithm.

To evaluate an algorithm we have to satisfy the following criteria:

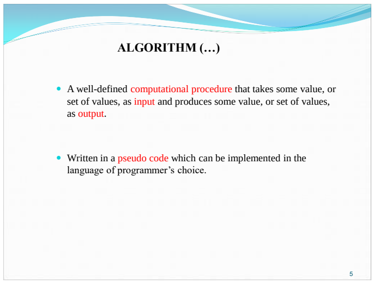
1.INPUT: Zero or more quantities are externally supplied.

2.OUTPUT: At least one quantity is produced.

3.Definiteness: Each instruction is clear and unambiguous.

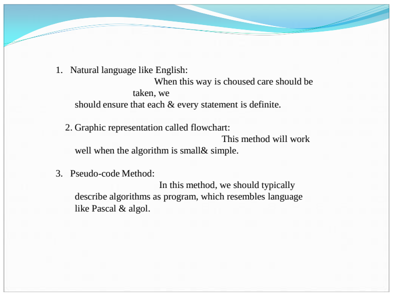
4.Finiteness: If we trace out the instructions of an algorithm, then for all cases, the algorithm terminates after a finite number of steps..

5.Effectiveness: Every instruction must very basic so that it can be carried out, in principle, by a person using only pencil & paper.



ALGORITHM (…)

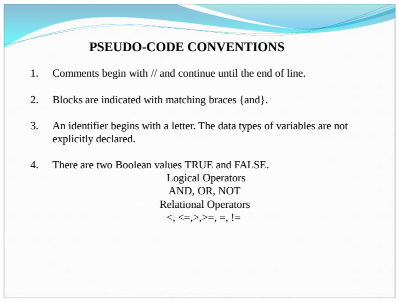
- A well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values, as output.
- Written in a pseudo code which can be implemented in the language of programmer's choice.



1. Natural language like English:
When this way is choused care should be taken, we should ensure that each & every statement is definite.

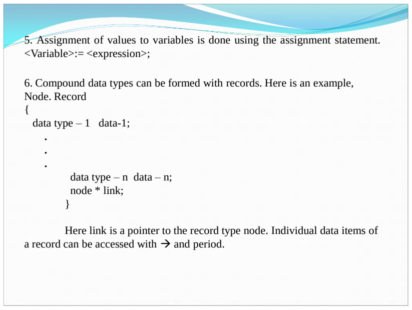
2. Graphic representation called flowchart:
This method will work well when the algorithm is small& simple.

3. Pseudo-code Method:
In this method, we should typically describe algorithms as program, which resembles language like Pascal & algol.



PSEUDO-CODE CONVENTIONS

1. Comments begin with // and continue until the end of line.
2. Blocks are indicated with matching braces {and}.
3. An identifier begins with a letter. The data types of variables are not explicitly declared.
4. There are two Boolean values TRUE and FALSE.
Logical Operators
AND, OR, NOT
Relational Operators
<, <=,>,>=, =, !=

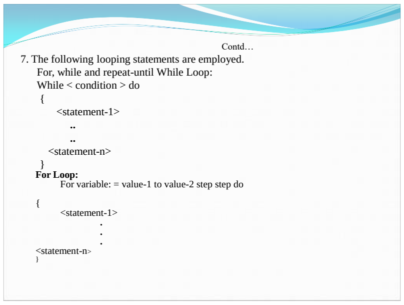


5. Assignment of values to variables is done using the assignment statement.
<Variable>:= <expression>;

6. Compound data types can be formed with records. Here is an example,
Node. Record

```
{
  data type – 1   data-1;
     .
     .
     .
          data type – n  data – n;
          node * link;
        }
```

Here link is a pointer to the record type node. Individual data items of a record can be accessed with → and period.



Contd…

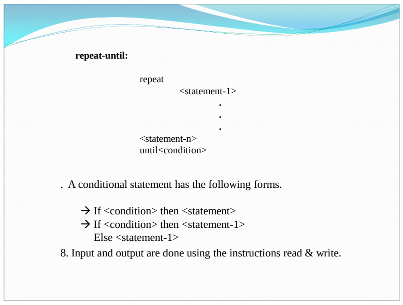
7. The following looping statements are employed.
For, while and repeat-until While Loop:

```
While < condition > do
 {
     <statement-1>
          ..
          ..
   <statement-n>
 }
```

For Loop:

```
      For variable: = value-1 to value-2 step step do

{
      <statement-1>
                 .
                 .
                 .
<statement-n>
}
```



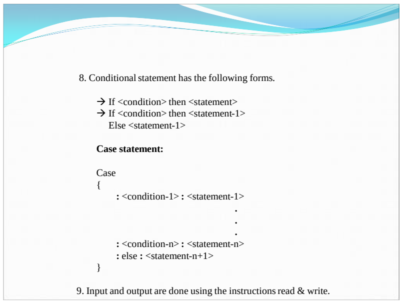
repeat-until:

```
repeat
        <statement-1>
               .
               .
               .
<statement-n>
until<condition>
```

. A conditional statement has the following forms.

```
→ If <condition> then <statement>
→ If <condition> then <statement-1>
  Else <statement-1>
```

8. Input and output are done using the instructions read & write.



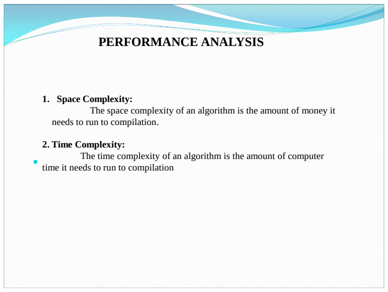
8. Conditional statement has the following forms.

```
→ If <condition> then <statement>
→ If <condition> then <statement-1>
  Else <statement-1>
```

Case statement:

```
Case
{
     : <condition-1> : <statement-1>
                          .
                          .
                          .
     : <condition-n> : <statement-n>
     : else : <statement-n+1>
}
```

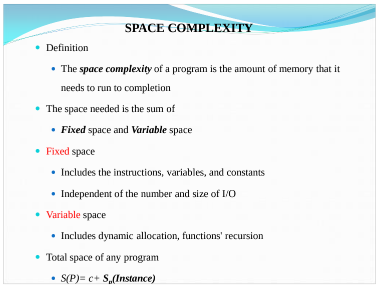
9. Input and output are done using the instructions read & write.



PERFORMANCE ANALYSIS

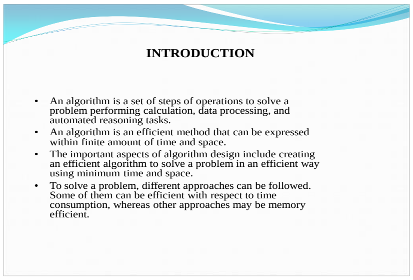
1. **Space Complexity:**
The space complexity of an algorithm is the amount of money it needs to run to compilation.

2. **Time Complexity:**
The time complexity of an algorithm is the amount of computer time it needs to run to compilation



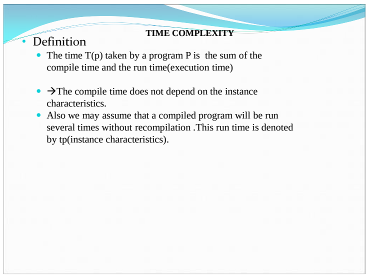
SPACE COMPLEXITY

- Definition
  - The ***space complexity*** of a program is the amount of memory that it needs to run to completion
- The space needed is the sum of
  - ***Fixed*** space and ***Variable*** space
- Fixed space
  - Includes the instructions, variables, and constants
  - Independent of the number and size of I/O
- Variable space
  - Includes dynamic allocation, functions' recursion
- Total space of any program
  - $S(P)= c+ \mathbf{S_p(Instance)}$



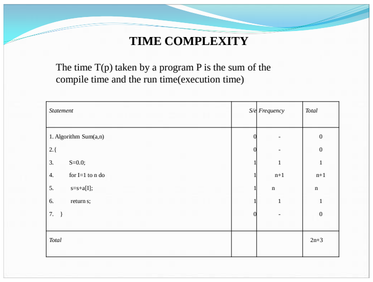
INTRODUCTION

- An algorithm is a set of steps of operations to solve a problem performing calculation, data processing, and automated reasoning tasks.
- An algorithm is an efficient method that can be expressed within finite amount of time and space.
- The important aspects of algorithm design include creating an efficient algorithm to solve a problem in an efficient way using minimum time and space.
- To solve a problem, different approaches can be followed. Some of them can be efficient with respect to time consumption, whereas other approaches may be memory efficient.



TIME COMPLEXITY

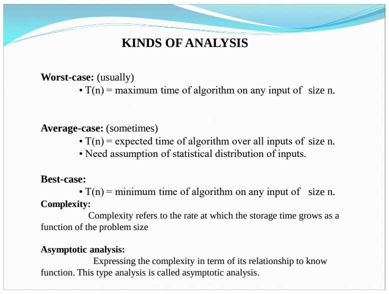
- Definition
  - The time T(p) taken by a program P is the sum of the compile time and the run time(execution time)
  - →The compile time does not depend on the instance characteristics.
  - Also we may assume that a compiled program will be run several times without recompilation .This run time is denoted by tp(instance characteristics).



TIME COMPLEXITY

The time T(p) taken by a program P is the sum of the compile time and the run time(execution time)

Statement	*S/e*	*Frequency*	*Total*
1. Algorithm Sum(a,n)	0	-	0
2.{	0	-	0
3. S=0.0;	1	1	1
4. for I=1 to n do	1	n+1	n+1
5. s=s+a[I];	1	n	n
6. return s;	1	1	1
7. }	0	-	0
Total			2n+3



KINDS OF ANALYSIS

Worst-case: (usually)

• T(n) = maximum time of algorithm on any input of size n.

Average-case: (sometimes)

• T(n) = expected time of algorithm over all inputs of size n.

• Need assumption of statistical distribution of inputs.

Best-case:

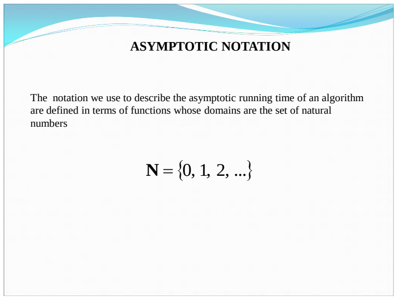
• T(n) = minimum time of algorithm on any input of size n.

Complexity:

Complexity refers to the rate at which the storage time grows as a function of the problem size

Asymptotic analysis:

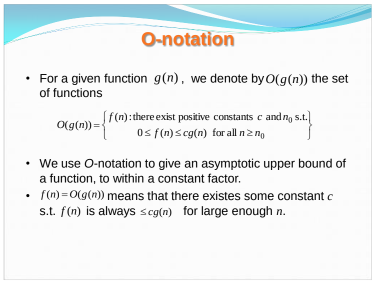
Expressing the complexity in term of its relationship to know function. This type analysis is called asymptotic analysis.



ASYMPTOTIC NOTATION

The notation we use to describe the asymptotic running time of an algorithm are defined in terms of functions whose domains are the set of natural numbers

$$\mathbf{N} = \{0,\ 1,\ 2,\ \ldots\}$$

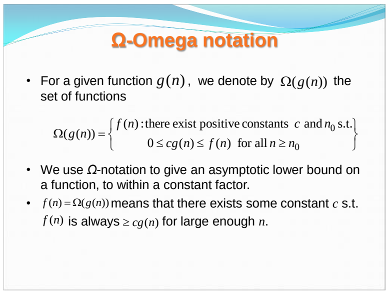


O-notation

- For a given function $g(n)$, we denote by $O(g(n))$ the set of functions

$$O(g(n)) = \left\{ \begin{array}{l} f(n): \text{there exist positive constants } c \text{ and } n_0 \text{ s.t.} \\ 0 \le f(n) \le cg(n) \text{ for all } n \ge n_0 \end{array} \right\}$$

- We use *O*-notation to give an asymptotic upper bound of a function, to within a constant factor.
- $f(n) = O(g(n))$ means that there existes some constant c s.t. $f(n)$ is always $\le cg(n)$ for large enough n.

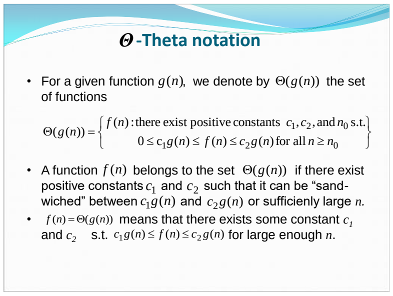


Ω-Omega notation

- For a given function $g(n)$, we denote by $\Omega(g(n))$ the set of functions

$$\Omega(g(n)) = \left\{ \begin{array}{l} f(n): \text{there exist positive constants } c \text{ and } n_0 \text{ s.t.} \\ 0 \le cg(n) \le f(n) \text{ for all } n \ge n_0 \end{array} \right\}$$

- We use *Ω*-notation to give an asymptotic lower bound on a function, to within a constant factor.
- $f(n) = \Omega(g(n))$ means that there exists some constant c s.t. $f(n)$ is always $\ge cg(n)$ for large enough n.

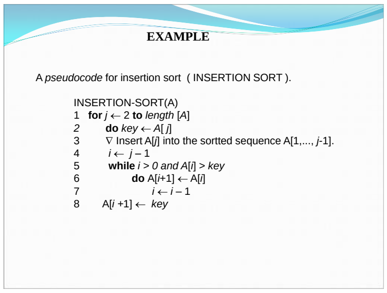


Θ -Theta notation

- For a given function $g(n)$, we denote by $\Theta(g(n))$ the set of functions

$$\Theta(g(n)) = \left\{ \begin{array}{l} f(n): \text{there exist positive constants } c_1, c_2, \text{and } n_0 \text{ s.t.} \\ 0 \le c_1 g(n) \le f(n) \le c_2 g(n) \text{ for all } n \ge n_0 \end{array} \right\}$$

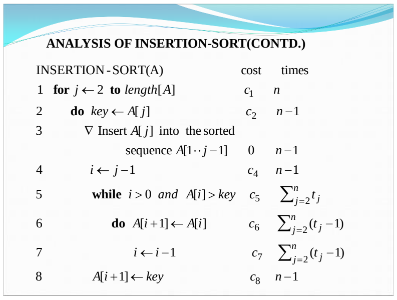
- A function $f(n)$ belongs to the set $\Theta(g(n))$ if there exist positive constants c_1 and c_2 such that it can be "sand-wiched" between $c_1 g(n)$ and $c_2 g(n)$ or sufficienly large *n.*
- $f(n) = \Theta(g(n))$ means that there exists some constant c_1 and c_2 s.t. $c_1 g(n) \le f(n) \le c_2 g(n)$ for large enough n.



EXAMPLE

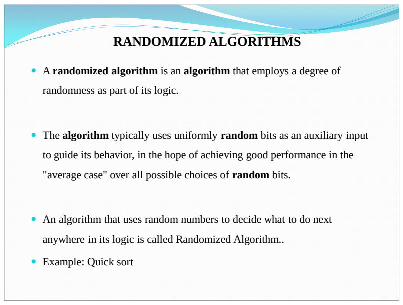
A *pseudocode* for insertion sort (INSERTION SORT).

```
INSERTION-SORT(A)
for $j \leftarrow 2$ to $length\ [A]$
    do $key \leftarrow A[j]$
    ∇ Insert A[$j$] into the sortted sequence A[1,..., $j$-1].
     $i \leftarrow j - 1$
     while $i > 0$ and $A[i] > key$
          do A[$i$+1] ← A[$i$]
               $i \leftarrow i - 1$
    A[$i$ +1] ← $key$
```



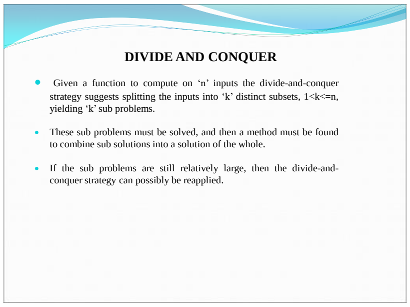
ANALYSIS OF INSERTION-SORT(CONTD.)

INSERTION - SORT(A)	cost	times
1 **for** $j \leftarrow 2$ **to** $length[A]$	c_1	n
2 **do** $key \leftarrow A[j]$	c_2	$n-1$
3 ∇ Insert $A[j]$ into the sorted sequence $A[1 \cdot\cdot j-1]$	0	$n-1$
4 $i \leftarrow j-1$	c_4	$n-1$
5 **while** $i > 0$ *and* $A[i] > key$	c_5	$\sum_{j=2}^{n} t_j$
6 **do** $A[i+1] \leftarrow A[i]$	c_6	$\sum_{j=2}^{n} (t_j - 1)$
7 $i \leftarrow i-1$	c_7	$\sum_{j=2}^{n} (t_j - 1)$
8 $A[i+1] \leftarrow key$	c_8	$n-1$



RANDOMIZED ALGORITHMS

- 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.
- An algorithm that uses random numbers to decide what to do next anywhere in its logic is called Randomized Algorithm..
- Example: Quick sort



DIVIDE AND CONQUER

- Given a function to compute on 'n' inputs the divide-and-conquer strategy suggests splitting the inputs into 'k' distinct subsets, 1<k<=n, yielding 'k' sub problems.
- These sub problems must be solved, and then a method must be found to combine sub solutions into a solution of the whole.
- If the sub problems are still relatively large, then the divide-and-conquer strategy can possibly be reapplied.

17. Websites or URLs e- Resources

- Introduction to Algorithms - GeeksforGeeks
- Design and analysis of algorithms - Course (nptel.ac.in)
- Divide and Conquer, Sorting and Searching, and Randomized Algorithms | Coursera
- https://www.tutorialspoint.com/design_and_analysis_of_algorithms/index.htm
- https://www.javatpoint.com/daa-tutorial