School of Computer Engineering
KIIT deemed to be University
Laboratory Lesson Plan – Autumn'2023 (5th Semester)

Course Name and Code: **Algorithms Laboratory-CS2098** (L-T-P-Cr: 0-0-2-1)

Discipline: B.Tech (CSE, IT, CSCE, CSSE, ECS)

Instructor Name : Prof. Anil Kumar Swain, **Mobile/WhatsApp:** +91-9938853866

Email :: anil.swainfcs@kiit.ac.in

Instructor Chamber : Faculty Chamber-1, Ground Floor, Block-B, Campus-15

Technical Assistant Contact Details:

TA Name	Mr. Krushna Chandra Tripathy	Ms. Anshita Samal
TA Mail ID	krushnachandra.tripathy@kiit.ac.in	anshita.samal@kiit.ac.in
TA Phone No	9861416812	7750011733/7377649028

Course Contents:

Random numbers or numbers from a file as input to an array.

Fundamentals of Algorithmic Problem Solving: Analysis of time complexity of small algorithms through step/frequency count method, asymptotic notations. (Insertion Sort Algorithms)

Divide and Conquer Method : Binary Search, Merge Sort, Quick Sort, Randomized Quick Sort

Heap & Priority Queues: Building a heap, Heap sort algorithm, Min-Priority queue, Max-Priority queue

Greedy Technique: Fractional knapsack problem, Activity selection problem, Huffman's code

Dynamic Programming: Matrix Chain Multiplication, Longest Common Subsequence

Graph Algorithms:

Dis-joint Set Data Structure, Representation Of Graph, Graph Traversals (BFS DFS), Single Source Shortest Path (Dijkstra's Algorithm, Bellman-Ford), All Pair Shortest Path (Floyd-Warshall Algorithm), Minimum Cost Spanning Tree (Kruskal's Algorithm, Prim's Algorithm),

List of Experiments (Day wise):

Lab - 1

❖ Lab-1 Assignments

1.1 Write a program to store random numbers into an array of n integers and then find out the smallest and largest number stored in it. n is the user input.

Sample Input	Sample Output
Enter number of random numbers to be stored in an array: 10	The content of the array with random input are as follows: 10 40 35 47 68 22 40 46 98 10 The smallest number is 10 The largest number is 98

1.2 Write a program to store random numbers into an array of n integers, where the array must contains some duplicates. Do the following:

a) Find out the total number of duplicate elements.

b) Find out the most repeating element in the array.

Hints: Write the random generation function in such a way that it must generate some duplicate numbers. As for example if you generate 10 random numbers within a range say 11 to 15, then atleast 5 numbers will be duplicate.

Sample Input	Sample Output
Enter number of random numbers to be stored in an array: 15	The content of the array with random input are as follows: 10 40 35 47 68 22 40 10 98 10 50 35 68 40 10 Total number of duplicates = 4 The most repeating element in the array = 10

1.3 Write a program to rearrange the elements of an array of n integers such that all even numbers are followed by all odd numbers. How many different ways you can solve this problem. Write your approaches & strategy for solving this problem.

Sample Input	Sample Output
Enter number of random numbers to be stored in an array: 10	The content of the array with random input are as follows: 0, 1, 3, 6, 4, 9, 8, 7, 5, 2 The content of the array with all even numbers followed by all odd numbers are as follows: 1, 3, 5, 7, 9, 0, 2, 4, 6, 8
	Or 5, 1, 3, 7, 9, 4 , 8, 6, 0, 2

Note: Other outputs may also posible depending on the logic used, but it must satisfy the requirements, First all odd numbers will be stored, then all even numbers, but the sequence within a group may vary.

❖ Lab-1 Practice Problem Set

1.4 Write a program to display an array of n integers (n>1), where at every index of the array should contain the product of all elements in the array except the element at the given index. Solve this problem by taking single loop and without an additional array.

Sample Input	Sample Output
Enter number of random numbers to be stored in an array: 5	Input Array : 3 4 5 1 2 Output Array :40 30 24 120 60

1.5 Write a program to test whether a number the number is

a) Ugly number
b) Kaprekar number
c) Happy Number

Sample Input	Sample Output
Input an integer number: 15	15 is an ugly number. 15 is not a kaprekar number. 15 is not a happy number.
Input an integer number: 45	45 is an ugly number. 45 is a kaprekar number. 45 is not a happy number.
Input an integer number: 19	19 is not an ugly number. 19 is not a kaprekar number. 19 is a happy number.

Lab - 2

❖ Lab-2 Assignments

2. 1 Write a menu driven program as given below, to sort an array of n integers in ascending order by **insertion sort algorithm** and determine the **time required (in terms of step/frequency count)** to sort the elements. Repeat the experiment for different values of n and different nature of data (i.e.apply insertion sort algorithm on the data of array that are already sorted, reversely sorted and random data). Finally plot a graph of the time taken versus n for each type of data. The elements can be read from a file or can be generated using the random number generator. Assume the cost of each statement is 1.

INSERTION SORT MENU

---

0. Quit
1. n Random numbers=>Array
2. Display the Array
3. Sort the Array in Ascending Order by using Insertion Sort Algorithm
4. Sort the Array in Descending Order by using any sorting algorithm
5. Time Complexity (step count) to sort ascending of data for all Cases (Data Ascending, Data in Descending & Random Data) in tabular form for values n=5 to 9, step=1.
6. Time Complexity (step count) to sort ascending of data for all Cases (Data Ascending, Data in Descending & Random Data) in tabular form for values n=5000 to 50000, step=5000

Enter your choice:

Sample Input & Output

In the output the above menu will be displayed.

Enter your choice: 1
Enter how many random numbers to store into an array: 10

Enter your choice: 2
The content of array is as follows: 10 30 34 56 70 36 90 88 72 38
(**Note:** Based on user choice as 1, each time this output may vary)

Enter your choice: 3
The content of array is as follows: 10 30 34 36 38 56 70 72 88 90

If option 4 is entered, then the output will be displayed as follows:

Sl.	Data Size	#Steps (Ascending data)	#Steps (Descending data)	#Step (Random data)
1	5	19	39	29
2	6	23	53	29
3	7	27	69	49
4	8	31	87	55
5	9	35	107	81

Note: For a insertion sort function with specific data size, the number of steps required for that insertion sort function for ascending data and descending data will remain same always. For random data it will vary for each execution, but the value must come in between #Steps (Ascending data) and #Steps (Descending data).
For option-4, solve using pen and paper and match with the output data after execution. If it does not match, then recheck your program, do necessary correction with step count variables and get the exact output.
Applying insertion sort on ascending data and descending data, the number of steps can be calculated using a formulae wrf to your write-up insertion sort function. If the nature of data is random, then number of steps can be calculated manually by looking at the data.

Once option 4's output is found correct, Copy option 4's data to option 5 and change the code accordingly.

❖ Lab-2 Practice Problem Set

2. 2 Write a menu driven program wrf to program 2.1, to sort an array of n integers in ascending order by **Selection sort algorithm** and determine the **time required (in terms of step/frequency count)** to sort the elements. Repeat the experiment for different values of n and different nature of data (i.e., Apply selection sort algorithm on the data of array that are already sorted, reversely sorted and random data). Finally plot a graph of the time taken versus n for each type of data. The elements can be read from a file or can be generated using the random number generator.

Lab - 3

❖ Lab-3 Assignments

3. 1 Rewrite the program no-2.1 (Insertion Sort) with the following details.

I. Compare the best case, worst case and average case time complexity with the same data except time complexity will count the **CPU clock cycle time.**

II. Plot a graph showing the above comparison (n, the input data Vs. CPU times for best, worst & average case)

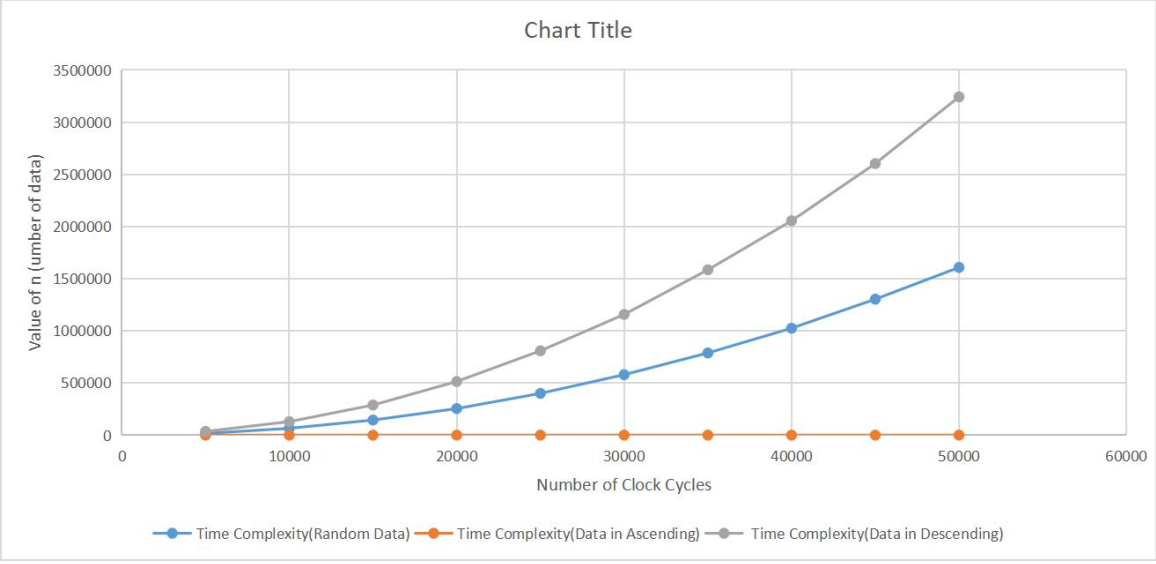
III. Compare manually program no-2.1 graph vs program no-3.1 graph and draw your conclusion.

Sample Input & Output

If option 4 is entered, then the output may be displayed as follows:

Sl.	Data Size	#CPU Cycles (Ascending data)	#CPU Cycles (Descending data)	#CPU Cycles (Random data)
1	5000	18	32292	16351
2	10000	35	129217	64351
3	15000	53	293226	145881
4	20000	71	517780	257292

Note: The above data may vary each time program is executed. However the graph with the output data must be similar to the following graph. If data size increses, so also #CPU Cycles (Ascending data) and also #CPU Cycles (Descending data).



❖ Lab-3 Practice Problem Set

3. 2 Consider an n × n matrix A = (a_{ij}), each of whose elements a_{ij} is a nonnegative real number, and suppose that each row and column of A sums to an integer value. We wish to replace each element a_{ij} with either ⌊ a_{ij} ⌋ or ⌈ a_{ij} ⌉ without disturbing the row and column sums. Write a program by defining an user defined function that is used to produce the rounded matrix as described in the above example. Find out the time complexity of your algorithm/function.

Sample Input

Input Matrix

10.9	2.5	1.3	9.3
3.8	9.2	2.2	11.8
7.9	5.2	7.3	0.6
3.4	13.1	1.2	6.3

Sample Output

Output Matrix

11	3	1	9
4	9	2	12
8	5	7	1
3	13	2	6

Lab - 4

❖ Lab-4 Assignments

4.1 Write a program to search an element x in an array of n integers using binary search algorithm that uses divide and conquer technique. Find out the best case, worst case and average case time complexities for different values of n and plot a graph of the time taken versus n. The n integers can be generated randomly and x can be choosen randomly, or any element of the array or middle or last element of the array depending on type of time complexity analysis. Compare this time complexities with the time complexity of linear search.

Sample Input	Sample Output
Enter numner of random numbers: 10	The content of the array with random input are as follows: 10 40 35 47 68 22 40 46 98 10 Enter a number to be searched: 46 Search is successful. The number 46 is found in 7th index of the array position.

4.2 Write a program to sort a list of n elements using the **merge sort** method in ascending order and analyse the time complexity. Refer the style of analysis of program 2.1 and 3.1.

Sample Input	Sample Output
Enter how many random numbers: 10	The content of the array with random input are as follows: 10 40 35 47 68 22 40 46 98 10 (**Note:** This may be one of the possible output) The sorted array (ascending order) is as follows: 10 10 22 35 40 40 46 47 68 98

❖ Lab-4 Practice Problem Set

4.3 The quick sort algorithm is an efficient and popular sorting technique that sorts a list of keys recursively by choosing a pivot key. A pivot may be chosen as the first or last or mean or median or any random element of the list. Write a program to implement this sorting algorithm and execute the sorting programs for the following sets of data.

i. Ordered List
ii. Reverse order List
iii. A list containing the same value through out

iv. Random List
v. 50% of the List sorted

Also measure CPU time for data sizes 5K to 50K with step size=5k. Present your results using tables and graphs and write a 1-page report that summarize the behavior of sorting algorithms tested and their suitability in each case as mentioned above.

Sample Input & Output

Data size	Same_element	50% sorted	Fully_random	Fully_sorted	Reverse_sorted
5000	30340	763	748	96758	96758
10000	122580	1640	1644	393798	388534
15000	277871	2531	2309	836642	818704
20000	482881	3506	3697	1221151	1025017
25000	735951	3422	3579	1574616	1866181
30000	1083254	5162	5324	2382314	2278576
35000	1440601	4953	5139	3089523	3085750
40000	1882988	5718	5837	5402945	4067477
45000	2400503	8009	7967	5516962	5100755
50000	2942720	7840	7642	6362107	7983041

4.4 A and B are playing a guessing game where B first thinks up an integer X (positive, negative or zero, and could be of arbitrarily large magnitude) and A tries to guess it. In response to A's guess, B gives exactly one of the following three replies:

a) Try a bigger number
b) Try a smaller number or
c) You got it.

Write a program by designing an efficient algorithm to minimize the number of guesses A has to make.

Sample Input & Output

Let B thinks up the number: 35

A's guess	B's response
10	Try a bigger number
20	Try a bigger number
30	Try a bigger number
40	Try a smaller number
35	You got it

Lab - 5

❖ Lab-5 Assignments

5.1 Write a menu (given as follows) driven program to sort an array of n integers in ascending order by heap sort algorithm and perform the operations on max heap.

MAX-HEAP & PRIORITY QUEUE MENU

0. Quit
1. n Random numbers=>Array
2. Display the Array
3. Build a max-heap
4. Extract largest element
5. Replace value at a node with new value
6. Insert a new element
7. Delete an element
8. Sort the Array in Ascending Order by using Max-Heap Sort technique

Enter your choice:

Sample Input & Output

In the output the above menu will be displayed. If option 1 is entered, then the output will be displayed as follows:
Enter how many random numbers to store into an array: 5

If option 2 is entered, then the output will be displayed as follows:
The content of array: 2 3 5 4 7

If option 3 is entered, then the output will be displayed as follows:
Max-Heap array: 7 3 5 4 2

If option 4 is entered, then the output will be displayed as follows:
7

If option 5 is entered, then the output will be displayed as follows:
Enter index and value : 1 8
The content of array: 8 7 5 4 2

If option 6 is entered, then the output will be displayed as follows:
Enter an element to be inserted :9
The content of array: 9 7 8 4 2 5

If option 7 is entered, then the output will be displayed as follows:
Enter an element to be deleted: 5
The content of array: 9 7 8 4 2

If option 8 is entered, then the output will be displayed as follows:
2 4 7 8 9

Note: the above outputs are fixed wrf to choosing option 1 first, then option 2 onwards are choosen sequentially. Otherwise output may vary as satisfying max-heap property.

❖ Lab-5 Practice Problem Set

5.2 Similar to above program no.1.1, write a menu driven program to sort an array of n integers in descending order by heap sort algorithm. Hints: Use min heap and accordingly change the menu options.

Lab - 6

❖ Lab-6 Assignments

6.1 Write a program to implementation of Fractional Knapsack Problem, stated as follows:
A thief with a Knapsack having capacity W kg, rubbing a store finds n items. Each item is tagged with item number, weights and profits in the form of {item number, value or profit, weight}. How the thief will, put these items in the knapsack, so that after selling them in the market, he will get the maximum total profit. If the theif wants, the he may take a part of any item.

Sample Input	Sample Output
Enter Knpsack data Number of items : 8 Knapsack Capacity: 17 Enter each items name, value, weight Item1: a 10 5 Item2: b 15 4 Item3: c 8 3 Item4: d 7 7 Item5: e 3 2 Item6: f 15 3 Item7: g 8 2 Item8: h 27 6 (**Note:** Use array of structure to store the Knpsack data)	Item To be taken (Weight) a Not b Full c 2/3 taken d Not e Not f Full g Full h Full

Other variant

Sample Input	Sample Output
Enter Knpsack dataNumber of items : 8 Knapsack Capacity: 17 Enter details of each item such as	Item To be taken (Weight) 1 Not 2 Full 3 2/3 taken

item-no, value, weight in matrix form (3x8 matrix) 0 row for item number 1 row for market value of each item 2 row for weight of each item 1 2 3 4 5 6 7 8 10 15 8 7 3 15 8 27 5 4 3 7 2 3 2 6 (**Note:** Define an input 5xn matrix, all initialized with zero. In Row 3, store value per wight and ro4 4, te solution vector.	4 Not 5 Not 6 Full 7 Full 8 Full

6.2 Write a program to implement the activity-selection problem stated as follows: You are given n activities with their start and finish times. Select the maximum number of activities that can be performed by a single person, assuming that a person can only work on a single activity at a time.

Sample Input	Sample Output
Enter number of activity: 6 Enter each activity's name, start and finish time Activity1: a 1 2 Activity2: b 3 4 Activity3: c 1 6 Activity4: d 5 7 Activity5: e 8 9 Activity6: f 5 9	The maximum number of activity that can be executed by a person is {a, b, d, e}

Other variant

Similar to program 6.1, input may be given through matrix form.

❖ Lab-6 Practice Problem Set

6.3 Huffman coding assigns variable length codewords to fixed length input characters based on their frequencies/probabilities of occurrence. Given an array of characters along with their frequency of occurrences.

Sample Input	Sample Output
Enter the number of distinct characters- 6 Enter the characters into an array : a b c d e Enter its corresponding frequencies: 40 20 15 30 75	Character — Huffman Code a — 10 b — 1111 c — 1110 d — 110 e — 0

Lab - 7

❖ Lab-7 Assignments

7.1 Write a C program to implement the Longest Common Subsequence. Also findout all possible LCSs.

Sample Input	Sample Output
Enter string1: abbababa Enter string2: babaabaab	The LCS is abbaab with length 6.

7.2 Write a program to implement the matrix chain multiplication problem using M-table & S-table to find optimal ordering of matrix multiplication.

Sample Input	Sample Output
Enter number of matrices: 4 Enter the sequence of dimensions of the matrix matrix chain multiplication: 2 4 5 6 3 The input may also be given as follows: Enter number of matrices: 4 Enter the row and column of each matrix that obeys the rule of matrix chain multiplication: 2 4 4 5 5 6 6 3 (**Note:** Number of column of each matrix must be same as row of next matrix if it exists.)	The required m Table is- 0 1 2 3 4 1 0 40 100 136 2 0 0 120 150 3 0 0 0 90 4 0 0 0 0 The required s Table is- 0 1 2 3 4 1 0 1 2 3 2 0 0 2 2 3 0 0 0 3 4 0 0 0 0 The optimal ordering is – (((A1 A2) A3) A4) The optimal ordering of the given matrices requires 136 multiplications.

❖ Lab-7 Practice Problem Set

7.3 Given n elements input through keyboard, write a program that prints the length of the longest increasing subsequence whose adjacent element difference is one.

Sample Input	Sample Output
Enter maximum size of array: 10 Enter 10 integers: 3 10 3 11 4 5 6 7 8 12	The longest increasing subsequence whose adjacent element differs by one: {3, 4, 5, 6, 7, 8} The length of increasing subsequence: 6

7.4 Given an array of n positive integers, input through keyboard. Write a program to find the sum of maximum sum subsequence of the given array such that the integers in the subsequence are sorted in increasing order.

Sample Input	Sample Output
Enter maximum size of array: 7 Enter 10 integers (separated by space): 1 101 2 3 100 4 5	The maximum sum subsequence : {1, 2, 3,100} with sum 106

Lab - 8

❖ Lab-8 Assignments

8.1 Write a program to input a graph using adjacency matrix form and perform depth-first and breadth-first traversal.

Sample Input	Sample Output
Enter number of vertices of the graph: 3 Enter the adjacency matrix of the graph 1 1 0 1 0 1 0 1 1 Enter the source vertex: 2	The BFS sequence is 2 1 3 The DFS sequence is 2 3 1

8.2 Write a program to input a weighted graph with adjacency matrix form and findout the minimum cost spanning tree by using Kruskal's algorithm.

Sample Input	Sample Output
Enter number of vertices of the graph: 7 Enter the adjacency matrix of the graph 0 4 4 0 0 0 0 4 0 2 0 0 0 0 4 2 0 3 4 0 0 0 0 3 0 3 0 0 0 0 4 3 0 0 0 0 0 2 0 3 0 0	Edge Cost (2, 1) 2 (5, 2) 2 (3, 2) 3 (4, 3) 3 (1, 0) 4 The cost of Minimum Spanning tree: 14

❖ Lab-8 Practice Problem Set

8.3 Write a program to input a weighted graph with adjacency matrix form and findout the minimum cost spanning tree by using Prim's algorithm.

Sample Input	Sample Output
Enter number of vertices of the graph: 5	Edge Cost (3, 1) 4

Enter the adjacency matrix of the graph 0 0 3 0 0 0 0 10 4 0 3 10 0 2 6 0 4 2 0 1 0 0 6 1 0	(0, 2) 3 (2, 3) 2 (3, 4) 1 The cost of Minimum Spanning tree: 10

Lab - 9

❖ Lab-9 Assignments

9.1 Write a program to input a directed wighted graph with adjacency matrix form and findout the shortest path from a given source vertex to all other vertices by using Dijkstra's algorithm.

Sample Input	Sample Output
Enter number of vertices of the graph: 7 Enter the adjacency matrix of the graph 0 4 0 2 3 0 0 0 0 1 0 0 0 0 0 2 0 0 0 3 2 0 2 0 0 1 0 0 0 1 2 0 0 5 0 0 5 1 0 0 0 1 0 0 0 0 0 0 0 Enter the source vertex: 0	The cost and path from source vertex 0 to all other destination vertex are as follows: Vertex Cost Path 1 4 0-1 2 5 0-4-2 3 2 0-3 4 3 0-4 5 8 0-4-5 6 7 0-4-2-6

❖ Lab-9 Practice Problem Set

9.2 Write a program to input a directed wighted graph with adjacency matrix form and findout the shortest path from a given source vertex to all other vertices by using Bellman-Ford algorithm.

Sample Input	Sample Output
Enter number of vertices of the graph: 5 Enter grapg in adjacency matrix form 0 6 0 7 0 0 0 5 8 -4 0 -2 0 0 0 0 0 -3 0 9 2 0 7 0 0 Enter the source vertex: 0	Vertex Cost Parent 0 0 0 1 2 3 2 4 4 3 7 1 4 -2 2

Lab - 10

❖ Lab-10 Assignments

10.1 Write a program to input a directed wighted graph and findout the shortest path from every vertex to every other vertex by using Floyd-Warshalls algorithm.

Sample Input	Sample Output
Enter the number of vertices: 4 Enter the edges: [0][0]: 0 [0][1]: 2 [0][2]: 4 [0][3]: 1 [1][0]: 2 [1][1]: 3 [1][2]: 4 [1][3]: 2 [2][0]: 5 [2][1]: 2 [2][2]: 0 [2][3]: 1 [3][0]: 6 [3][1]: 4 [3][2]: 2 [3][3]: 0	The original graph in adjancy matrix form is: 0 2 4 1 2 3 4 2 5 2 0 1 6 4 2 0 The shortest path matrix is: 0 2 3 1 2 3 4 2 4 2 0 1 6 4 2 0

❖ Lab-10 Practice Problem Set

10.2 Modify program no.10.1 so that it will display all intermediate path matrices and its corresponding predecessor matrices.

Grading Policies:

- The entire lab course consists of 100 marks. The marking scheme is as follows:

Components	Marks
Continuous Evaluation Marks	60
End Sem. Lab. Evaluation	40
Total	**100**

Continuous Evaluation Marks (60)		
Sl. No.	Components	FM
1	Program Execution (Evaluated by TAs)	20
2	Lab Record Evaluation (Evaluated by TAs)	10
3	Class Participation (Evaluated by Faculty)	10
4	Quiz (Evaluated by Faculty)	10
5	Viva (Evaluated by Faculty)	10
	Total =	**60**

End Sem. Lab. Evaluation (40)		
Sl. No.	Components	FM
1	Programing Test (Evaluated by TAs)	20
2	Viva & Quiz (Evaluated by Faculty)	20
	Total =	**40**

Reference Materials:

1. T. H. Coreman, C. E. Leiserson, R. L. Rivest, “Introduction to Algorithms, PHI
2. Sara Basse, A. V. Gelder, “Computer Algorithms”, Eddison Wesley.
3. Michael Goodrich, Roberto Tamassia, “Algorithm Design: Foundations, Analysis & Internet Examples”, John Wiley & Sons.
4. Fundamentals of comp Alg E.Harwitz,S. sahani,S.Rajsekharan,Galgotia