Nirma University

Institute of Technology

Semester End Examination (IR), December - 2015

B. Tech. in Information Technology, Semester-VII

2CE339 Analysis and Design of Algorithm

Roll/ Exam No		Supervisor's initial with date	

Time: 3 Hours | Max Marks: 100

Instructions:

i) Attempt all questions.
ii) Figures to the right indicate full marks.
iii) Use section-wise separate answer books.
iv) Draw neat sketches wherever necessary.

SECTION-I

Q-1 Do as directed. **[18]**

A. Using ε (epsilon) and δ (delta) definition of limit, show that given arbitrary functions f and $g : N\text{->}R^{\geq 0}$, if $\lim_{n\to\infty} \frac{f(n)}{g(n)} \in +\infty$, then g(n) $\in O(f(n))$ but f(n) does not belongs to $\Theta(g(n))$. **6**

B. Prove the following statements: **4**

i. $n^2 = O(n^4)$
ii. $2^{n+1} = O(2^n)$ but $2^{2n} \neq O(2^n)$

OR

B. Show that for any real constants a and b, where b > 0, $(n + a)^b = \Theta(n^b)$ **4**

C. Solve the following recurrence relation using characteristic polynomial and find the general as well as particular solution (find the values of all necessary constants) : **4**

$$T_n = n \text{ if } n = 0, 1 \text{ or } 2$$
$$T_n = 5T_{n-1} - 8T_{n-2} + 4T_{n-3} \text{ otherwise}$$

D. Solve the following recurrence relation using master theorem. $T(n) = 2T(n/4) + \sqrt{n}$. **4**

Q-2 Do as directed. **[16]**

A. Design and analyze an algorithm for building binary heap in the following two different ways. **10**

- One starts at the last internal node and works backwards along the array till the root running heapify at each intermediate node, assuming that the subtrees rooted at its children are already heaps.

- The other grows the size of the heap one at a time, initially assuming only one element is present, at the first array position. When it increases to include one extra element, it checks if it is correctly placed with respect to its parent. At this stage it is assumed that the array of elements proceeding it constitute a heap. The algorithm then moves the new node up the tree using exchanges until a correct position has been found.

Derive exact asymptotic bounds on the running times of the two different build-heap procedures, in terms of n.

B. Elements 53, 77, 108, 44, 86, 47 and 114 are inserted into hash table of size 7 in the same order. The hash function is Key % (size of table), where Key is the value of an element. Show the content of table after inserting every element into the hash table. Use linear probing technique to resolve hash collision. Apply bubble sort on the final content of table and give the final sequence of elements after third pass of bubble sort. Note that, % indicates the modulo operation and for sorting, the element at index 0 (in the hash table) is the first element. **6**

OR

B. Give the characteristics of a good hash function. Discuss the advantages of quadratic probing over linear probing using necessary hash functions. **6**

Q-3 Do as directed. **[16]**

A. Using divide and conquer approach one can find the rank of any element (position of an element in a sorted sequence) from an array in linear time by dividing array into groups of 5. Discuss the design steps to find median (rank (n/2)) from an array. Will the algorithm work in linear time if they are divided into groups of 7? Argue that the algorithm will not work in linear time if groups of size 3 are used. **8**

B. Let X[1 . . . n]and Y [1 . . . n] be two arrays, each containing n elements already in sorted order. Give an O(log n) time algorithm to find the median of all 2n elements in arrays X and Y . **5**

C. "Quick sort in its original form is not stable". Justify this statement with an example. **3**

Section II

Q-4 Do as directed. **[18]**

A. Give the necessary modifications in quick sort algorithm so that the maximum size of call stack in all three cases (worse case, average case and best case) remains O(log n). Also discuss the average case running time analysis of quick sort using necessary recurrence relation. **8**

OR

A. Suppose T(n) is defined on nonnegative integers by recurrence as below: **8**
$T(n) = aT(n/b) + f(n)$. Where $a \geq 1$, $b > 1$ and f(n) is a positive function. Show that if $f(n) = \Theta\ (n^{\log_b a})$ for some constant $\varepsilon > 0$, then $T(n)= \Theta(n^{\log_b a} \log n)$.

B. Using divide and conquer approach, discuss the design steps of O(n log n) algorithm to find closest pair among n distinct points. Discuss it for One-dimensional as well as for two-dimensional space. Show necessary sketch and recurrence relations for the same. **10**

Q-5 Do as directed. **[16]**

A. Write the recursive solution using dynamic programming to find Longest Common Subsequence (LCS) of two sequences. Also find the LCS X=< 'a', 'm', 'p', 'u', 't', 'a', 't', 'i', 'o', 'n'> and Y=< 's', 'p', 'a', 'n', 'k', 'i', 'n', 'g'>. **8**

B. Find the optimal order of multiplying following matrices using dynamic programming. $A_{total} = A_1A_2A_3A_4$ where A_1:7×6, A_2:6×5 A_3:5×4 and A_4:4×3. Show all intermediate steps. **8**

Q-6 Do as directed. **[16]**

A. Generate an undirected graph form the matrix representation as shown in Fig (1). Apply Breadth First Search algorithm considering {A} as a start node. Show all intermediate steps with queuing operations (Enqueue and Dequeue). Also give a set of edges visited after each step. **8**

	A	B	C	D	E	F
A	0	1	1	1	0	0
B	1	0	1	1	0	1
C	1	1	0	0	1	1
D	1	1	0	0	0	1
E	0	0	1	0	0	0
F	0	1	1	1	0	0

Fig (1)

OR

A. Generate an undirected graph form the matrix representation as shown in Fig (1). Apply Depth First Search algorithm considering {E} as a start node. Show all intermediate steps with start time and finish time of each visited vertices. Also give a set of edges visited after each step. **8**

B. Matrix representation of complete undirected graph is shown in Fig (2). Find the minimum spanning tree for this graph using prim's algorithm. Show all necessary steps. What is the minimum possible weight of a spanning tree T in this graph such that vertex A is the leaf node in the tree T? **8**

	A	B	C	D	E
A	0	1	8	1	4
B	1	0	12	4	9
C	8	12	0	7	3
D	1	4	7	0	2
E	4	9	3	2	0

Fig (2)