# Nirma University

Institute of Technology

Semester End Examination (IR), May - 2015

B. Tech. in Computer Engineering, Semester-VI

2CE339 Analysis and Design of Algorithms

| 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** To define the relationship between two arbitrary functions using asymptotic notation, the corresponding domain and codomain cannot be arbitrary. Why? 4

**OR**

**A** Show that $n(\log n)^{10} \in O(n^2 \log n)$. 4

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

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

$T_n = 0$ if $n = 0$
$T_n = 5$ if $n = 1$
$T_n = 3T_{n-1} + 4T_{n-2}$ otherwise

**D** Solve the following recurrence relation using master theorem. 4
$T(n) = T(n/3) + n$.

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

**A** 4

```
MAX_HEAPIFY (A, i)
    l= Left(i)
    r= Right(i)
    if l ≤ A.heap_size and A[l] > A[i]
          largest = l
    else  largest = i
    if r ≤ A.heap-size and A[r] > A[largest]
          largest = r
    if largest ≠ i
          exchange A[i] with A[largest]
          MAX_HEAPIFY (A, largest)
```

The above routine maintains the binary max-heap property at index i in the array A. Apply the above routine to create a binary max-heap from array A = {32, 15, 46, 24, 11, 19, 28}. Show all intermediate steps.

**B** Is it possible to build max-heap from an unordered array in linear time? If yes, then prove it. If no then justify. **6**

**C** Elements 46, 70, 101, 37, 79, 40 and 107 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 selection sort on the final content of table and give the final sequence of elements after third pass of selection 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**

**C.** Discuss the disadvantage of linear probing strategy and provide the necessary solutions for it. **6**

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

**A** Compare heap sort and quick sort based on following three criteria. **8**

1- Stability
2- Space complexity in worse case (excluding call stack)
3- Worse case time complexity

Give proper justifications with examples (if necessary) for your solutions.

**B** One can find the rank of any element from an array (position of an element in a sorted sequence) in linear time using divide and conquer technique. Briefly discuss the design steps for it. Does it work in all cases? Provide a proper justification of your answer with necessary proofs. **8**

## Section II

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

**A** In mathematics, the notion of multiset is a generalization of the notion of a set in which members are allowed to appear more than once. An absolute majority element in a multiset of n elements is any element which occurs at least $\lceil 2n/3 \rceil$ times. Obviously, any set can have at most one absolute majority element. Give a worst case O(n) time algorithm to find an absolute majority element of an n element multiset given as input, if one exists, and reporting no otherwise. **10**

**B** Discuss divide and conquer approach to multiply large integers. Also provide the necessary recurrence relation for it and solve it using recursion tree method. **8**

**OR**

**B** **8**

```
Insertion_sort (A)
  For j = 2  to A.length
     key = A[j]
     i= j-1
     While i > 0 and A[i] > key
        A[i+1] = A[i]
        i= i-1
     A[i+1] = key
```

Briefly discuss the average case and worse case analysis for the above insertion sort. Also provide the necessary upper bounds for the same.

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

**A** Using dynamic programming find Longest Common Subsequence (LCS) of two sequences X=<A, B, C, B, D, A, B> and Y=<B, D, C, A, B, A>. Mention all necessary steps. **8**

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

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

| | M | N | O | P | Q | R |
|---|---|---|---|---|---|---|
| M | 0 | 1 | 0 | 0 | 1 | 1 |
| N | 1 | 0 | 1 | 0 | 1 | 0 |
| O | 0 | 1 | 0 | 1 | 0 | 0 |
| P | 0 | 0 | 1 | 0 | 1 | 0 |
| Q | 1 | 1 | 0 | 1 | 0 | 0 |
| R | 1 | 0 | 0 | 0 | 0 | 0 |

**Fig (1)**

**A.** Generate an undirected graph form the matrix representation as shown in Fig (1). Apply depth first search algorithm considering {O} as the 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**

**OR**

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

**B.** An Undirected graph G(V,E) contains n( n > 2) nodes named $v_1$ , $v_2$, $v_3$, $v_4$ .... $v_n$. Two nodes $v_i$, $v_j$ are connected if and only if $0 < |i-j| \le 4$. Each edge ($v_i$, $v_j$) is assigned a weight 2(i+j). Generate a sample weighted graph with n=5 and find the minimum spanning tree using Kruskal's algorithm. **8**