

**DESIGN & ANALYSIS OF ALGORITHMS**  
(CSEN 2201)

Time Allotted : 2½ hrs

Full Marks : 60

*Figures out of the right margin indicate full marks.*

*Candidates are required to answer Group A and  
any 4 (four) from Group B to E, taking one from each group.*

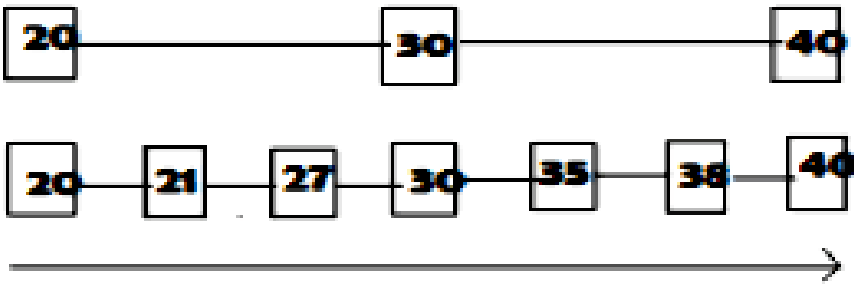
*Candidates are required to give answer in their own words as far as practicable.*

**Group - A**

1. Answer any twelve:

12 × 1 = 12

*Choose the correct alternative for the following*

- (i) The partition function of Quicksort helps to  
(I) Divide the original problem into similar subproblems  
(II) Solve the sub problems recursively  
(III) Combine the solutions of the sub problems into solution of the original problem  
Which of the following statements is true?  
(a) (I) and (II)                      (b) (I) and (III)                      (c) (II) and (III)                      (d) Only (I)
- (ii) If the rank of the pivot element is 5 and there are n elements in the array, what would be the size of the partitions in Quicksort?  
(a) 1 and n-1                      (b) 4 and n-5                      (c) 5 and n - 6                      (d) n - 4 and 4
- (iii) Suppose that there are 3 programs X1, X2 and X3 having time complexities  $f_1(n)$ ,  $f_2(n)$  but  $f_3(n)$  respectively such that  $f_1(n)$  is  $O(f_2(n))$ ,  $f_2(n)$  is  $O(f_1(n))$ ,  $f_1(n)$  is  $O(f_3(n))$  but  $f_3(n)$  is not  $O(f_1(n))$ .  
Then which one of the statements is true from the following statements?  
(a) X3 is always faster than X1 and X2 for every large size inputs  
(b) X1 is faster than X2 and X3 for every large inputs  
(c) X3 is slower than X1 and X2 for very large inputs  
(b) X2 is faster than X1 and X3 for very large size inputs.
- (iv) Say, we need the product of three 2D arrays A<sub>1</sub>, A<sub>2</sub> and A<sub>3</sub> with dimensions [20, 4] [4, 10] and [10, 5] respectively. What will be the minimum number of scalar multiplication needed?  
(a) 600                      (b) 120                      (c) 800                      (d) 200.
- (v) Fractional knapsack problem can be effectively solved by  
(a) Back tracking                      (b) Greedy algorithm                      (c) Dynamic programming                      (d) Divide and conquer
- (vi) Consider the 2-level Skip List given below
- 
- How is 38 to be accessed?  
(a) travel 20-30-35-38                      (b) travel 20-30-40-38  
(c) travel 20-21-27-30-35-38                      (d) travel 20-40-38.
- (vii) In the KMP algorithm for pattern matching, the prefix function  $(x)$  for a pattern P of length m, can be calculated in time –  
(a)  $O(m \log m)$                       (b)  $O(m)$                       (c)  $O(m \log n)$                       (d)  $O(\log m)$
- (viii) Which of the following methods takes overcharge for certain operations while performing amortized analysis?  
(a) Aggregate method                      (b) Accounting method  
(c) Potential method                      (d) Both (a) and (c).
- (ix) Let S be an NP-complete problem and Q and R be two other problems not known to be in NP. Q is polynomial time reducible to S and S is polynomial time reducible to R. Which one the following statements is true?  
(a) R is NP-complete                      (b) R is NP-hard  
(c) Q is NP-complete                      (d) Q is NP-hard.
- (x) Choose the correct answer for the following statements:  
I. The theory of NP-completeness provides a method of obtaining a polynomial time bound for NP algorithms.  
II. All NP-complete problems are NP-Hard.  
(a) I is FALSE and II is TRUE                      (b) I is TRUE and II is FALSE  
(c) Both are TRUE                      (d) Both are FALSE.

- (xi) Mergesort uses \_\_\_\_\_ and conquer strategy.
- (xii) The worst case time complexity of Bellman-Ford algorithm for a directed, weighted graph of  $|V|$  vertices and  $|E|$  edges is\_\_\_\_\_.
- (xiii) The Single-source shortest path problem in a graph  $G$  becomes undefined if there exists a \_\_\_\_\_reachable from the source.
- (xiv) After an execution of the partition subroutine of quicksort, at least one element of the array will occupy its final position as per the sorted ordering. This element is known as \_\_\_\_\_.
- (xv) The height of an almost complete binary tree which is representing a max-heap of 1000 elements is \_\_\_\_\_.

**Group - B**

2. Consider the recurrence  $T(n)=3T(n/4) + 2n^2$
- (i) Draw the recursion tree for the above recurrence and calculate the cost of the tree. The Cost of the tree is the solution of the recurrence.
  - (ii) Prove using substitution method that your solution is correct.
  - (iii) Solve the above recurrence using master theorem.

[[CO3](Apply/IOCQ)]  
**(4 + 4 + 4) = 12**

3. (a) Consider the following recurrence:

$$T(n) = 1, \text{ if } n = 1$$

$$T(n) = 2T(n/4) + \sqrt{n}, \text{ if } n > 1$$

What is the asymptotic upper bound of  $T(n)$ ?

[[CSEN2201.3](Analyse/IOCQ)]

- (b) Show that the asymptotic lower bound for comparison-based sort is  $\Omega(n \log_2 n)$ . [[CSEN2201.3](Understand/LOCQ)]
- (c) Suggest an algorithm that will find the maximum and minimum from a given set of  $n$  elements in at most  $(3 \lfloor \ln n/2 \rfloor)$  comparisons. [[CSEN2201.1, .2, .3, .4](Apply/LOCQ)]

**4 + 4 + 4 = 12**

**Group - C**

4. (a) Consider the directed graph shown in the figure (Fig. 1) below. There can be multiple shortest paths from  $S$  to  $E$ . Which one will be reported by your shortest path algorithm? Do not forget to justify your choice of the algorithm and step-wise evaluation.

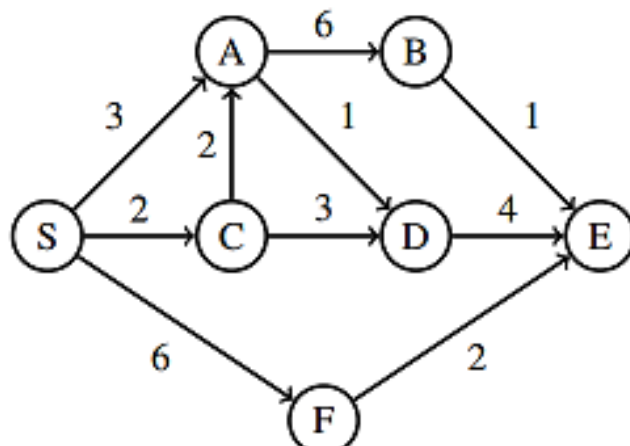


Fig.1

Which shortest path algorithm cannot be used to find the shortest path from  $S$  to  $T$  on the directed graph (Fig. 2) given below and why?

Name a shortest path algorithm that can be used to find the shortest path from  $S$  to  $T$  on the directed graph (Fig. 2) given below and why? What will be the time complexity for that algorithm?

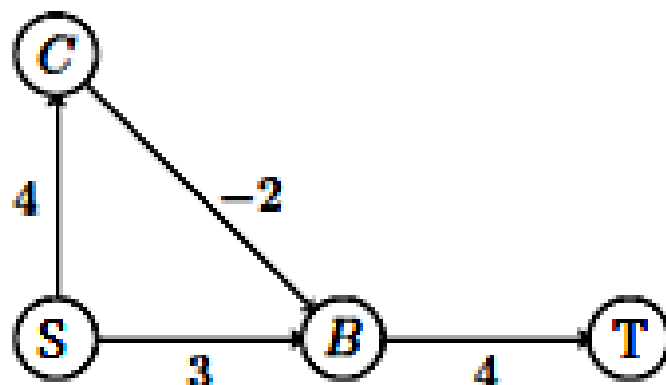


Fig. 2

- (b) Consider two strings  $X = "qpqrr"$ ,  $Y = "pqprrqp"$ . Let 'a' be the length of the longest common subsequence (not necessarily contiguous) between  $X$  and  $Y$  and let 'b' be the number of such longest common subsequences between  $X$  and  $Y$ . Find a and b.

[[CSEN2201.2, .3, .4, .6](Apply, Analyse/IOCQ)]

[[CSEN2201.2, .3](Understand/LOCQ)]

**(5 + 3 + 1) + 3 = 12**

5. (a) The characters a to h have the set of frequencies based on the first eight Fibonacci numbers as follows: a:1, b:1, c:3, d:7, e:8, f:15, g:25, h:40. The Huffman code is used to represent the characters. What is the sequence of characters corresponding to the following code?

**110111100111010**

[[CSEN2201.2, .3](Apply/IOCQ)]

- (b) Let A1, A2, A3, A4, and A5 be four matrices of dimension 30 X 35, 35 X 15, 15 X 5, 5 X 10, and 10 X 20 respectively. What will be the minimum number of scalar multiplications required to find the product of A1A2A3A4A5? How will the matrices be parenthesized? (Do not forget to show the intermediate steps).

[[CSEN2201.2, .3](Apply/IOCQ)]

**5 + 7 = 12**

### Group - D

6. (a) A sequence of n operations is performed on a data structure. The cost of i-th operation is  $C(i) = i^2$ , if i is an exact power of 3  
 $= 3$ , otherwise.

Calculate the exact expression for finding the cost for n successive operations using Aggregate Analysis.

Determine the asymptotic amortized cost per operation. To make things simple, let us assume that  $n = 3k$ , i.e., n is an exact power of 3.

Hint:  $\sum_{i=1}^n C(i) = 32 + 92 + 272 + \dots + (n)^2 + \text{some quantity}$ .

[[CO4](Analyze/IOCQ)]

- (b) We know that in a stack operation if we have multipop in addition to push and pop Operation, then the amortized cost per operation is  $O(1)$ . Will this amortized cost change if we add a 4th operation called multi-push, which may push as many items as needed in a single operation. If yes, what will be the new amortized cost? If not, why not? Justify your answer.

[[CO4](Analyze/IOCQ)]

**8 + 4 = 12**

7. (a) Show that the expected number of nodes used by a Skip List containing n items formed by tossing an unbiased coin is bounded by  $2n$ .

[[CO2](Understand/LOCQ)]

- (b) Suppose you are searching a pattern P in a text T such that  $|P| = m$  and  $|T| = n$ .

Write the pseudo-code for a Naïve String Matcher and state its complexity.

[[CO1](Remember/LOCQ)]

- (c) Given below is the pseudo-code for the function that the KMP matcher uses to match patterns. Some portions are intentionally removed and are replaced by bold-faced capital letters like **A, B, C, D, E, F, G, H, I, and J** that you have to replace with proper text to make it a complete working code.

#### COMPUTE -PREFIX -FUNCTION (P)

1.  $m \leftarrow \text{length}[P]$
2.  $\pi[A] \leftarrow 0$
3.  $k \leftarrow 0$
4. For  $q \leftarrow \mathbf{B}$  to  $\mathbf{C}$
5.     do while  $k > \mathbf{D}$  and  $p[\mathbf{E}] \neq p[\mathbf{F}]$
6.         do  $\mathbf{G} \leftarrow \pi[\mathbf{H}]$
7.     if  $p[\mathbf{I}] = p[\mathbf{J}]$
8.         then  $k \leftarrow k + 1$
9.      $\pi[q] \leftarrow k$
10. return  $\pi$

[[CO4](Analyze/IOCQ)]

**4 + (2 + 1) + 5 = 12**

### Group - E

8. (a) Show that the VERTEX-COVER problem is NP-Complete.

[[CSEN2201.5](Remember,Understand/IOCQ)]

- (b) Can you propose an approximation algorithm to solve the above-mentioned NP-Complete problem? How accurate / optimal your solution is?

[[CSEN2201.3, .5, .6](Understand, Apply/IOCQ)]

**5 + (4 + 3) = 12**

9. (a) Suppose you are doing a linked list implementation of Weighted-Union Find algorithm in C. Please fill up only the portion with comments in bold fonts.

```

struct set_header
{
    int weight;
    // implement this portion
};
struct set_object
{
    int item;
    // implement this portion
};

void union(struct set_object *a, struct set_object *b)
{
    if(a->weight _____) // write the blank portion of the if
condition
    {
        // implementation NOT required
    }
    else
    {
        // implement this portion
    }
}

```

- (b) Name a special version of TSP problem that has a constant approximation algorithm and also mention the constant factor. [[CO2](Understand/LOCQ)]
- (c) State and prove the Max-Flow Min-Cut theorem. [[CO1](Remember/LOCQ)]

[[CO2](Understand/LOCQ)]

**6 + 1 + 5 = 12**

Cognition Level	LOCQ	IOCQ	HOCQ
Percentage distribution	31.25	68.75	0

#### Course Outcome (CO):

After the completion of the course students will be able to

CSEN2201.1. Remember time complexities of various existing algorithms in different situations.

CSEN2201.2. Understand the basic principles of different paradigms of designing algorithms.

CSEN2201.3. Apply mathematical principles to solve various problems.

CSEN2201.4. Analyze the complexities of various algorithms in worst case, best case and average case.

CSEN2201.5. Assess the computational hardness of a problem and learn how some of the well-known problems are proved to be NP-hard and also design approximation algorithms for some of them.

CSEN2201.6. Create / Design a good algorithm for a new problem given to him/ her.

\*LOCQ: Lower Order Cognitive Question; IOCQ: Intermediate Order Cognitive Question; HOCQ: Higher Order Cognitive Question.