H Roll No. .....

TCS–409

B. TECH. (CS) (FOURTH SEMESTER) MID SEMESTER EXAMINATION, April/May, 2022

DESIGN AND ANALYSIS OF ALGORITHMS

Time : 1½ Hours

Maximum Marks : 50

Note : (i) Answer all the questions by choosing any *one* of the sub-questions.

(ii) Each question carries 10 marks.

1. (a) Explain the following three asymptotic notations : 0, θ, Ω. Also prove or disprove the following : 5 + 5 = 10 Marks (CO1)

(i) Is $2^{2n} = O\left(2^n\right)$

(ii) Is $\left(\log n\right)^{10} = \Omega\left(n\right)$

OR

(b) Solve the following questions as asked :

2 × 5 = 10 Marks (CO1)

(i) Algorithms A and B spend exactly $T_A(n) = c_A\, n \log_2 n$ and $T_B(n) = c_B n^2$

P. T. O.

microseconds, respectively, for a problem of size n. Find the best algorithm out of A and B (with proper calculations) for processing $n = 2^{20}$ data items if algorithm A spends 10 microseconds to process 1024 items and algorithm B spends only 1 microsecond to process 1024 items.

(ii) Order the following expressions by their asymptotic growth (fastest to slowest) :

$2^{\log_{10} n}, n^{\log_2 10}, n^3, n^{7/2}, 2^{2^n}, 4^n, 5.1^n$

2. (a) Write the recursive psedudocode for Binary search along with the proper explanation of its time and space complexities. Mention what can be the place of the overflow situation in binary search algorithm/pseudocode and how to resolve it. 10 Marks (CO1)

OR

(b) Write the recursive pseudocode for finding the sum of the Fibonacci series up to n terms along with the proper explanation of its time and space complexities.

Mention what is the exact time complexity of printing the Fibonacci series in terms of θ (theta notation). 10 Marks (CO1)

3. (a) Solve each of the below recurrence relations : $2 \times 5 = 10$ Marks (CO1)

(i) $T(n) = T(n/3) + T(n/6) + T(n/9) + n$

(ii) $T(n) = 2T(\sqrt{n}) + \log_2 n$

OR

(b) Consider the following Pseudocode :

```
foo (a [0, ....n -1]
{
    while (n > 1)
    {
        i = 0, j = n -1;
        while (i < j)
        {
            a [i] = a [i] + a [j];
            i++, j--;
        }
```

```
        n = (n + 1)/2;
    }
}
```

Answer the following questions :

$5 \times 2 = 10$ Marks (CO1)

(i) Calculate the time complexity of foo() ?

(ii) Run the above algorithm for the following instance of input and show the final contents in the array, a[] :

a[] = {2, 1, 9, 6, 3, 11, 9, 5, 4};

4. (a) What is sorting ? Explain stable and unstable sorting with the help of a suitable example. 10 Marks (CO2)

Write a pseudo-code for 2 way merge-sort and run it on the following list of elements :

{29, 12, 55, 17, 28, 35, 27, 48, 33, 52, 45, 14}

Explain the space and time complexities in the best case and worst case for 2 way merge sort.

OR

(b) Given two sorted lists of values **list 1** and **list 2**, we would like to find the list of common elements, and this final **list** should also be sorted.

For example **list 1** = [2, 2, 3, 7] and **list 2** = [2, 3, 3, 3, 5, 7], the list of common elements is [2, 3, 7].

Write an algorithm/pseudocode for the above problem. Also find the time and space complexity of your proposed solution in terms of m and n, where m = size of **list 1**, n = size of **list 2**.

10 Marks (CO2)

5. (a) Answer the following questions :

(i) What can be the possible input type/s on which the running time complexity of Quicksort becomes $0(n^2)$?

(ii) Can we reduce the number of comparisons in insertion sort, which are normally done using linear search to find the proper position of an

element ? If Yes/No, give a proper explanation for you answer.

(iii) Give an example of a situation where bubble sort is faster than Quicksort.

(iv) Explain why the counting sort algorithm is (or isn't) in-place.

(v) What are Online Sorting and Offline sorting ? Give one example of each.

$5 \times 2 = 10$ Marks (CO2)

OR

(b) Answer the following questions as asked :

$5 \times 2 = 10$ Marks (CO2)

(i) Prove that the creation of a binary max-heap takes $\theta(n)$ time with an array on n elements.

(ii) Show with the help of pseudocode/algorithm that extracting a maximum element from a max heap takes $\theta(\log n)$ time.