$\sqrt{\log n} = 2^{w}$

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY (INDIAN SCHOOL OF MINES), DHANBAD
Monsoon Mid Semester Examination, Session 2022-23

Examination: III BTech (CSE) & BTech (Common) — Time: 2 Hours

Subject: Data Structures (CSC201) — Weightage: 28%, Max. Marks: 40

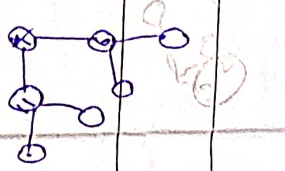
Instructions:

(i) Answer **ALL** questions.

(ii) Attempt all sub-parts of a same question at same place and in order.

(iii)If you use any algorithm (covered in the class) to solve any problem, just mention it, do not elaborate.

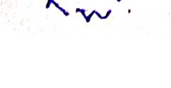
Q. No.	Question	Mark
1. (a)	Take the following list of functions and arrange them in ascending order of growth rate. That is, if function $g(n)$ immediately follows function $f(n)$ in your list, then it should be the case that $f(n)$ is $O(g(n))$. Write your answer with proper explanation. $f_1(n) = 2^{\sqrt{\log n}}$, $f_2(n) = 2^n$, $f_3(n) = n^{4/3}$, $f_4(n) = n^{\log n}$, $f_5(n) = 2^{2^n}$	5

(b)	Measure the step counts for all 4 statements in the following program segment to determine the overall asymptotic upper bound on the running time of the given program segment.
	`void example(int n)`
	`{`
	`int i = 1, s = 1 ;          /* Statement 1 */`
	`while (s <= n)               /* Statement 2 */`
	`{    i++ ;                   /* Statement 3 */`
	`     s += i ;  }             /* Statement 4 */`
	`}`
	
		5

2. (a)	Consider a doubly linked list in which only the first node (left most) is pointed by a header pointer *db_head*. Write two separate algorithms to perform the tasks as mentioned below: (i) Insert a new node in the given doubly linked list which is already sorted in ascending order of the key values stored in its nodes such a manner that after the insertion the sorted order must be retained as it is. (ii) Append a new doubly linked list in which only the last node (right most) is pointed by a header pointer *db_tail* at the end of the given doubly linked list.	6+3
(b)	Suppose that each row of an $n \times n$ array A consists of 1's and 0's such that, in any row of A, all the 1's come before any 0's in that row. Assuming A is already in memory, write an algorithm running in $O(n)$ time for finding the row of A that contains the most 1's.	3
3. (a)	You are given an array $Arr[1:n]$ representing a min-heap and an integer t. You have to design an algorithm to output all the keys in the array that are less than t. For example, if $t = 9$ and the array Arr is as shown below, then your algorithm should output the keys 4, 6, and 7. Note that the time complexity of the designed algorithm should be $O(k)$, where k be the number of keys in Arr that are smaller than t.	5
(b)	Consider the sequence of letters and asterisks given as: MID*S**EME**X*A***MD*S***. Now, consider the queue data structure with two supporting operations: enqueuer and dequeue. Suppose that for the above sequence, each letter (such as M) corresponds to an enqueue operation of that letter on	3

Table in 3. (a):

$Arr[1:n]$	1	2	3	4	5	6	7	8	9	10	11	12	13	14
key	4	11	6	14	18	7	12	16	17	22	19	10	9	24



temp→data < key.

temp→next→data > key.

the queue and each asterisk (*) corresponds a dequeue operation on the queue. Show the sequence of values returned by the dequeue operations.

4. (a) Consider the arithmetic infix expression as: $a + (b/c - (d \times e \uparrow f)/g) \times h$. Now, apply a stack-based algorithm to transform this infix expression into its equivalent postfix expression with a reference of the *ISP-ICP table* as given below. Note that the symbol "↑" represents the exponential arithmetic operation such a manner that $x \uparrow y = x^y$. No credit will be given without showing all intermediate steps clearly. 6

Symbol	In-Stack Priority (ISP)	In-Coming Priority (ICP)
)	—	—
↑	3	4
×, /	2	2
Binary +, −	1	1
(	0	4

(b) A full node, half node, and leaf node in a binary tree is a particular node with two children, one child, and no child, respectively. Prove or disprove that the number of full nodes plus one is equal to the number of leaves in a non-empty binary tree. 4