NATIONAL INSTITUTE OF TECHNOLGY PATNA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
MID-SEMESTER EXAMINATION - OCTOBER, 2022
B. Tech (Computer Science & Engineering) IIIrd Semester
(SECTION – A and SECTION - B)
CS3401/ CS34104– Data Structures
Max.Marks:30

Answer All questions

Q. No	Question	Marks	CO	BL
1	**Note: Answer all parts of this question at the same place:**			
	a. Consider an empty stack that is used to store integers. Let the numbers 1, 2, 3, 4, 5, and 6 be pushed into this stack in the order they appear from left to right. Let **P** indicate a push operation and **Z** indicate a pop operation. i. What is the output (the collection of pop operations) generated for the following permutation with six push operations (**P**) and six pop operations (**Z**)? **P P P Z Z P Z P P Z Z Z** ii. Can any permutation with six push operations (**P**) and six pop operations (**Z**) generates the output 154623? If yes, what is the permutation, otherwise write the reason why it is not possible.	2M	CO-4	Evaluate
	b. Apply the master method for finding the tight asymptotic bounds for the following two recurrences: i. $T(n) = 2T(n/2) + O(1)$ ii. $T(n) = 2T(n/4) + O(n^2)$	2M	CO-6	Apply
	c. Provide the increasing order of asymptotic complexity of the following four functions: $\log(n!)$, $\sqrt{\log n}$, nlogn, $2^{\log n}$.	2M	CO-6	Analyze
2	a. You have given two doubly linked lists, ***L1*** and ***L2***; In both lists every node stores one integer value. Write a Function/ Pseudocode/ Algorithm named ***SetDifference (List L1, List L2)*** that deletes all the nodes from ***L1*** whose node element (i.e., integer value) matches with an element of some node in ***L2*** (i.e., After running your algorithm, list ***L1*** should not contain any node whose value is present in some node in L2). Mention the time complexity of your algorithm by assuming L1 contains ***m*** nodes and L2 contains ***n*** nodes.	4M	CO-1	Understand &Apply
	b. Mention any two limitations of arrays. Write the time complexities for the following operations in Big-Oh notation. i. Searching an element in an unsorted array (Array contains ***n*** elements). ii. Removing all the duplicate elements in the array (Array contains ***n*** elements). iii. Merging two sorted lists and storing them in another array (The first array contains ***m*** elements; the second array contains ***n*** elements).	2M	CO-6	Remember

PTO

3	Write a function/pseudocode/algorithm which takes a singly linked list ***S*** as input and move all even-positioned nodes of ***S*** to the end of ***S***. While moving nodes, keep the relative order of all even-positioned and odd-positioned nodes. For example, if the list contains the nodes: **a→1→b→2→c→3→d→4**, then your program should convert it into **a→b→c→d→1→2→3→4.** Mention the time complexity of your algorithm by assuming the given list contains *n* nodes. **Note: Move the nodes in the list not the data in the nodes, and the list has only a header pointer no tail pointer. Assume a, b, c, …., 1, 2, 3, … are all addresses of the nodes, and initially, we don't know how many nodes are in the list.**	6M	CO-4	Understand & Apply
4	a. Sort the following elements using merge sort and show the output after every step: **23, 12, 34, 65, 21, 76, 35, 19**	3M	CO-5	Remember
	b. Mention the best- and worst-case time complexities of insertion sort and bubble sort.	1M	CO-6	Remember
	c. Assume that the operators +, -, ×, are left-associative and ^ is a right-associative operator (***^ is an exponential operator i.e., 2^3^2 is 512***). The order of precedence (from highest to lowest) is: ^ × +, -. (+ and – has equal precedence). Translate the following infix expression into a postfix expression based on the above precedence and associative rules using stack. You must show the contents of the stack and partial output after reading every character from the infix expression. **Infix Expression:** ***a + b × c - d ^ e ^ f + g*** ***Caution: Think about what to do when the precedence of the two operators (one is on top of the stack and another one is the currently reading from input) is same and their associativity is right to left?***	2M	CO-3	Understand
5	The **PUSH** and **POP** operations of a stack ***S*** pushes and pops one item at a time. Consider a **specialized stack *S'***, where the **PUSH** and **POP** of ***S'*** pushes and pops an *arbitrary number* of items instead of one item, but it follows the principle of the stack (if it pushes two elements '*a*' followed by 'b' into ***S'*** in its first **PUSH** operation then it has to pop those two elements in the order '*b*' followed by *'a'* during the last **POP** operation). Design a structure for ***S'*** and write the algorithms for the **PUSH** and **POP** operations of ***S'***. Mention the complexity of your **PUSH** and **POP** operations also defend your structure so that it is supporting the principle of the stack. *Hint: Design your structure in such a way that it should remember the number of elements pushed in each push operation.*	6M	CO-2	Create & Evaluate

********** ALL THE BEST **********