Reg. No.: 22BA[illegible]

Name : Anup Chopa[illegible]

VIT®

BHOPAL

www.vitbhopal.ac.in

MID TERM EXAMINATIONS – September 2023

Programme	: **B.Tech.**	Semester	: **Fall 2023-24**
Course Title/ Course Code	: **Data Structures and Analysis of Algorithms / CSD3009**	Slot	: **A11+A12+A13+A14**
Time	: **1 ½ hours**	Max. Marks	: **50**

Answer all the Questions

Q.No.	Sub. Sec.	Question Description	Marks
1		Solve the following relation to find time complexity of a computational problem. Illustrate any known computational problem that requires the following T(n) computational power to solve. $T(n) = 2T(n-1) + 1$, where $n >= 1$, and $T(0) = 0$	10
2		(i) Differentiate between simple singly linked list and simple doubly linked list. (ii) Write an algorithm to reverse the elements of a simple singly linked list with space complexity O(1). (iii) Mention the time complexity of your algorithm and the node structure considered for that algorithm.	10
3		Consider a deque ADT has the following operations/methods. (table below)	10

Operations/ Methods	Functionality
addFirst(i)	Insert a new element i at the front of the deque.
addLast(i)	Insert a new element i at the back of the deque.
removeFirst()	Remove and return the first element of the deque (or null if the deque is empty).
removeLast()	Remove and return the last element of the deque (or null if the deque is empty).
first()	Returns the first element of the deque without removing (or null if the deque is empty).
last()	Returns the last element of the deque without removing (or null if the deque is empty).

size()	Returns the number of elements in the deque.
isEmpty()	Returns a boolean indicating whether the deque is empty.

Show the return value of methods and state of deque for each operation/method in details if the following sequence of events is performed on the deque.

addLast(7), addFirst(3), addFirst(5), first(), removeLast(), size(). removeLast(), removeLast(), removeFirst(), addFirst(6), last(), addFirst(8), addFirst(9), isEmpty(), last().

i) Demonstrate the steps and status of stack in detail to convert the following infix expression into postfix expression using stack.

a + b * c ^ (d * e + f) * g

10

ii) Evaluate the following postfix expressing using stack.

6, 5, 3, 4, +, 8 *, +, 3, +, *

(i) Construct binary tree from given Preorder and Postorder tree traversal sequences. Show the steps and justification in details.

Preorder traversal sequence: {1,2,4,8,9,5,3,6,7}
Postorder traversal sequence: {8,9,4,5,2,6,7,3,1}

10

(ii) Differentiate between skewed binary search trees and height-balanced binary search trees with suitable examples.

⇔⇔⇔