Roll No. __________

Pandit Deendayal Energy University

Mid Semester Examination - September 2022

B. Tech. (Computer Science and Engineering)

Semester - III

Date: 23/09/2022

Course Name: Data Structure **Time: 2 hours**

Course Code: 20CP201T **Max. Marks: 50**

Instructions:

1. Do not write anything other than your roll number on question paper.
2. Assume suitable data wherever essential and mention it clearly.
3. Write appropriate units, nomenclature, and draw neat diagrams wherever required.

Question No.	**Description**	**Marks**	**CO Mapping**
Q1. (A)	Which one of the following is the disadvantage of arrays? i. Data structure like queue or stack cannot be implemented. ii. There are chances of wastage of memory space if elements inserted in an array are lesser than the allocated size. iii. Easier to store elements of same data type. iv. Elements are sequentially accessed.	2	CO-1
Q1. (B)	What does ‘stack underflow’ refers to? i. Accessing item from an undefined stack. ii. Adding items to a stack that is already full. iii. Removing items from an empty stack. iv. Index out of bounds exception.	2	CO-1
Q1. (C)	What is the time complexity of $pop()$ operation when the stack is implemented using an array? i. $O(1)$ ii. $O(n)$ iii. $O(\log n)$ iv. $O(n \log n)$	2	CO-2
Q1. (D)	What is the need for a circular queue? i. effective usage of memory ii. easier computations iii. to delete elements based on priority iv. implement FIFO principle in queues	2	CO-3
Q1. (E)	In a circular queue, how do you increment the rear end of the queue? Here, REAR points to the last element of circular queue and CAPACITY is the size of the queue. i. REAR = REAR + 1 ii. REAR = (REAR + 1) % CAPACITY iii. REAR = (REAR % CAPACITY) + 1 iv. REAR = REAR - 1	2	CO-3
Q2. (A)	Differentiate between the linear and non-linear data structures.	5	CO-1
Q2. (B)	Write an algorithm or pseudocode to evaluate the postfix expression using suitable data structure.	5	CO-4
Q3. (A)	Explain how Priority Queue can be applied, with an appropriate example.	5	CO-3
Q3. (B)	Apply infix to postfix algorithm that uses stack to convert the following infix expression into its corresponding postfix notation. $A + B * (C - D) / E + F - G$.	5	CO-5

Q4.	Consider existence of one stack ‘S’ and one linear queue ‘Q’ each of size **5**. The operations $push()$, $pop()$, $enqueue()$, and $dequeue()$ are defined as below: i. $push(A)$: insert element A onto stack S. ii. $pop()$: delete and print the top element of S. iii. $enqueue(A)$*:* insert element A into queue Q. iv. $dequeue()$: delete and print one element from Q. Find the output when each of the following operations are performed in the order given below. Also, print the content of both the data structures S and Q after each operation is performed. Note that the element is dropped if it doesn't find a place in the respective data structure. Assume that the data structures S and Q both are empty initially. $push(A)$, $push(B)$, $enqueue(C)$, $pop()$, $enqueue(D)$, $push(E)$, $dequeue()$, $push(F)$, $enqueue(G)$, $dequeue()$, $push(H)$, $dequeue()$, $push(I)$, $dequeue()$, $dequeue()$, $push(J)$, $pop()$, $enqueue(K)$, $push(L)$, $pop()$, $dequeue()$, $enqueue(M)$, $enqueue(N)$, $enqueue(O)$, $pop()$, $dequeue()$	10	CO-2
	Solve any two of the following questions.		

Q5. (A)	Suppose you are given an implementation of a linear queue of integers. The operations that can be performed on the queue Q are as follows:
	$isEmpty(Q)$: returns true if Q is empty, false otherwise.
	$dequeue(Q)$: deletes front element of Q and returns its value.
	$enqueue(Q, i)$: inserts the integer i at the rear of the Q.
	Consider the following function:
	$void\ function(Q)$
	{
	int i;
	if (! $isEmpty(Q)$)
	{
	$i = dequeue(Q)$;
	$function(Q)$;
	$enqueue(Q, i)$;
	}
	}
	What operation is performed by above $function(Q)$?
	Give proper justification for your answer.
		5	CO-2

Q5. (B)	Compute the time complexity (in Big-Oh notation) of the following program. Show the computation steps clearly.
	$main()$
	{
	$for\ (int\ i = 1;\ i <= n;\ i++)$
	$for\ (int\ j = i;\ j <= n;\ j++)$
	$printf$(“%d”, $i + j$);
	}
		5	CO-2

Q5. (C)	Compute the time complexity (in Big-Oh notation) of the following program. Show the computation steps clearly. Note that $sqrt(n)$ computes the square root of n.
	$main()$
	{
	$int\ n$;
	$while\ (n >= 2)$
	$n = sqrt(n)$;
	}
		5	CO-2