Roll Number:____________________

Thapar Institute of Engineering & Technology

Department of Computer Science and Engineering

END SEMESTER EXAMINATION

B. E. **(2nd Yr. COE/CSE)**
9th Dec., 2024
Monday, Time– 2:00 PM To 5:00 PM
Time: 3 Hours, Max Marks: 90 (Weightage: 40)

Course Code: UCS301
Course Name: Data Structures
Name of Faculty: Tarunpreet Bhatia, Rajesh Mehta, Charu Krishna, Nisha Chauhan, Ritesh Sharma, Kapil Tomar, Ashish Bajaj, Manisha Kaushal

Note: Attempt any 6 Questions in serial order. Answer all sub-parts of each question at one place. First 6 attempted questions will be evaluated only. Do mention Page No. of your attempt at front page of your answer sheet. Assume missing data (if any).

Q.No	Questions	M.M.	CO	BL
Q.1	(a) Consider a situation where you are designing a lightweight compiler for a new programming language. Your compiler converts infix expressions into postfix notation to evaluate mathematical expression efficiently. Your task is to convert the infix expression $Expr = A + B * C - D(E^F - G) + (H - I)$ into postfix notation using stack. Show intermediate steps.	(7)	CO1	L3
	(b) In a min-heap of distinct integers, where could the largest element be located? Insert integers 5, 3, 17, 10, 85, 2, 19, 4, 22, 1 one-by-one into an initially-empty min heap. Re-draw the heap each time an insertion causes one or more swaps.	(8)	CO1	L3
Q.2	(a) Use the Quick Sort algorithm to alphabetically sort the list: E, X, A, M, P, L, E. Assume the pivot is always the last element, and indexing starts from 0. Also, show the recursive calls during the sorting process. Determine the worst-case time complexity of the Quick Sort algorithm.	(7)	CO2	L3
	(b) A software development company is managing a large number of tasks with varying levels of priority (a smaller value indicates a higher priority). The goal is to sort the tasks based on their priority so that they can be executed in the ascending order of their priority. The company wants an efficient sorting algorithm that guarantees O(nlogn) performance and does not require additional memory for storage. Your task is to apply such algorithm to the given list of tasks with their respective priorities: Tasks= {T1, T2, T3, T4, T5, T6, T7, T8} and Priorities = {8, 5, 45, 24, 36, 11, 43, 21}. Show intermediate steps.	(8)	CO2	L3, L4
Q.3	(a) Given a queue of random elements, develop an algorithm/pseudocode to sort the queue elements in ascending order with O(1) space complexity. Apply your algorithm to the given queue Q = {5, 3, 1, 2} with 5 as front element: You are allowed to use queue STL functions.	(8)	CO1	L6
	(b) Design a data structure, *SpecialStack*, that implements the standard stack operations (push(), pop()) and an additional operation getMin(), which returns the minimum element in the stack, all in constant time O(1). This implementation should utilize an auxiliary stack to keep track of the minimum element dynamically during each operation. No additional data structures like array, lists etc. are allowed. Also, demonstrate the state of the stacks and the result of getMin() for the following sequence of operations: push(10), push(5), push(2), pop().	(7)	CO4	L6
Q.4	You are given the post order traversal of a Binary Search Tree (BST): 5, 15, 10, 35, 30, 25, 48, 50, 45, 40, 20. (a) Construct the BST from the given post order traversal (show intermediate steps) and determine its preorder traversal.	(7)	CO3	L3
	(b) Construct an AVL tree using the sequence of nodes as determined in preorder traversal (in part a) and show the rotations wherever applicable.	(8)		
Q.5	Consider a graph where the nodes (A-F) represent cities, and the edges indicate distances (weights) between them (as depicted in Fig. 1). Use Dijkstra's algorithm to perform the following tasks, showing the step-by-step execution: (a) Determine the shortest distance and path from the source node A to all other nodes. (b) Compute the shortest distance and paths from all other nodes to the destination node C.	(8) (7)	CO3	L3

	A	B	C	D	E	F
A	0	25	15	9	∞	8
B	∞	0	8	12	∞	∞
C	∞	∞	0	∞	6	∞
D	∞	∞	4	0	9	∞
E	∞	2	∞	∞	0	1
F	∞	∞	6	∞	∞	0

Fig. 1

Q.6 A telecommunications company manages a communication network modeled as a graph (as shown in Fig. 2). The nodes represent cities, and the edges denote direct communication lines between them. After a natural disaster, some communication lines are down (represented as dotted lines in Fig. 2). The company aims to identify the cities that are directly or indirectly connected to the central hub (city A) through the shortest path after the disaster. Answer the following: — CO3, CO4 — L3

(a) Develop an algorithm or pseudocode to solve the problem. (7)

(b) Apply your algorithm on Fig. 2 to determine the cities directly or indirectly connected to the central hub (city A) after the disaster and calculate the shortest paths. When choosing between vertices, follow lexicographic (alphabetical) order. Show all intermediate steps during the process. (8)

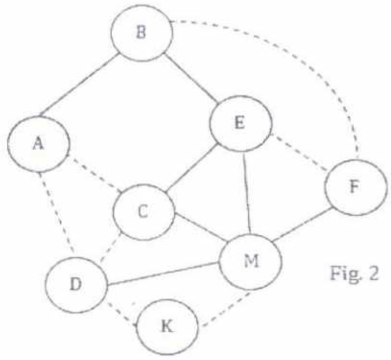

Fig. 2

Q.7 (a) Given a sorted doubly-linked list comprising positive distinct elements, your task is to complete function pairSum(struct Node *head, int x) which find pairs of elements whose sum is equal to the given value x with average time complexity of O(n) and space complexity of O(1). Each Node of doubly linked list has data field as data, pointer to the next node as next and pointer to the previous node as prev. You need not to write entire code just mention the statements corresponding to the fill in the blanks 1-7. (7) — CO4 — L3, 3

```
void pairSum(struct Node *head, int x)
{
  struct Node *first = head;
  struct Node *second = head;
  while (_____1_____!= NULL)
    second = second->next;
  bool found = false;
  while (____2____ && ____3_____ != first)
  {
    if (_____4_____== x ) {
      found = true;
      cout<<first->data<<" "<<second->data;
      first = first->next;
      second = _____5_____;  }
    else
    {
      if (_____6_____)
        first = first->next;
      else
        second = _____7_____;
    }
  }
  if (found == false)
    cout<<"No pair found";
}
```

(b) Consider a circular doubly linked list with the initial configuration as shown in Fig. 3. Each node of circular doubly linked list has data field as data, pointer to the next node as next and pointer to the previous node as prev. The following operations are performed one after the other on the given linked list. Update the structure of the circular doubly linked list after each of the following operations (i-iii) and provide the output of (iv). (8) — CO1 — 3 L3

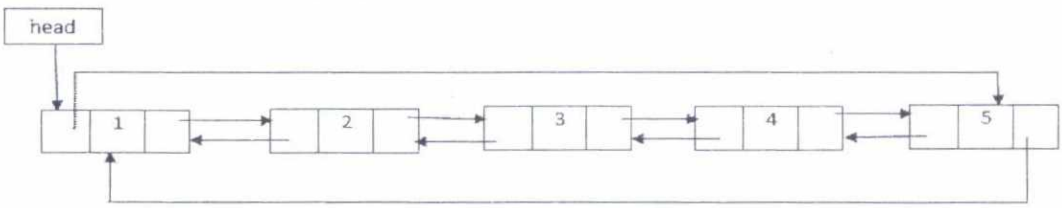

Fig. 3

(i) head ->next ->next ->next=head->prev

(ii) head -> prev ->prev -> prev = head -> next ->next ->next ->prev

(iii) head ->next ->next ->next ->prev = head ->prev ->prev ->prev

(iv) printf("%d", head -> prev ->prev -> prev->data)