ourdiu.com | Shared by Arpita Akter Shampa

Daffodil International University

Faculty of Science & Information Technology

Department of Software Engineering

Final Examination, Summer 2026

Course Code: SE131; Course Title: Data Structure

Level: 1 Term: 3 Sections: 46 (A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P)

Instructors: AB (A, B), DMA (C, D), MAK (F), TM (E, I, J), SHN (G), MAB (H, K, L), NSL (M, N, P), AAS (O)

Time: 2:00 Hrs **Marks: 40**

Answer ALL Questions

[The figures in the right margin indicate the full marks and corresponding course outcomes. All portions of each question must be answered sequentially.]

1.		You are developing a dynamic digital address book application for a large organization. The application stores contact names in a Binary Search Tree (BST) so that contacts can be efficiently organized and searched. Initially, the following contact names are added to the address book in the given order: **Harry, Daniel, Leo, Jack, Nord, Bob, Frank, Emily, Grace, Alan, Charlie, Mia, Oliver, Ian, Kevin**		**CLO-4** *Level-3*
	a)	Construct the Binary Search Tree (BST) by inserting the contact names in the exact order given above. **Sketch** the resulting BS tree clearly.	**[Marks-3]**	
	b)	Suppose the contact **Leo** is removed from the dress book. Delete the node **Leo** from the BST and **Sketch** the resulting BST.	**[Marks-2]**	
	c)	For the BST obtained after deleting **Leo, show** the following traversal sequences: *i.* Preorder traversal *ii.* Inorder traversal *iii.* Postorder traversal	**[Marks-3]**	
	d)	Define the C structure required to represent a node of the BST. Then, **construct** the main part of a C function that traverses the BST to print all contact names in alphabetical (ascending) order.	**[Marks-2]**	
2.	*a)*	During a football match, eight players—Player 1 (P1), Player 2 (P2), Player 3 (P3), Player 4 (P4), Player 5 (P5), Player 6 (P6), Player 7 (P7), and Player 8 (P8), take part in a passing sequence. The passing sequence begins when Player 1 (P1) passes the ball to Player 3 (P3) at a speed of 32 km/h. Player 3 then passes the ball to Player 2 (P2) at 28 km/h. As Player 2 is put under pressure by an opponent, Player 2 returns the ball to Player 3 at 25 km/h. Player 3 moves the attack forward by passing the ball to Player 5 (P5) at 34 km/h. Player 5 passes the ball to Player 4 (P4) at 30 km/h, and Player 4 returns it to Player 5 at 27 km/h. Player 5 then passes the ball to Player 6 (P6) at 39 km/h. Unable to move forward, Player 6 makes a back pass to Player 2 at 33 km/h. Player 2 passes the ball back into midfield to Player 5 at 31 km/h. Player 5 changes the direction of the attack by passing the ball to Player 7 (P7) at 40 km/h. Player 7 passes back to Player 4 at 29 km/h, and Player 4 passes the ball to Player 3 at 26 km/h. Player 3 then sends a long forward pass to Player 8 (P8) at 45 km/h. Player 8 lays the ball off to Player 6 at 24 km/h, and Player 6 returns the ball to Player 8 at 37 km/h. i. **Draw** the situation as a weighted directed graph, where each player is represented by a vertex, each pass is represented by a directed edge, and the passing speed is shown as the weight of the edge. ii. **Show** the adjacency List of the graph. iii. **Show** all the reachable players from Player 1 (P1) using the Depth-First Search (DFS) algorithm. Ignore the edge weights while performing DFS.	**[Marks-2+2+2]**	**CLO-4** *Level-3*

ourdiu.com

ourdiu.com | Shared by Arpita Akter Shampa

	b)	**Draw** the graph represented by the following adjacency matrix	[Marks-4]	

	A	B	C	D	E	F
A	0	12	0	7	0	0
B	5	0	9	0	0	0
C	0	4	0	4	11	0
D	0	6	0	0	8	0
E	0	0	3	0	0	10
F	14	0	0	5	0	0

3.	a)	**Construct** the main part of the code that performs insertion and deletion operations in a linear queue. *(No need to write the full program — only write the essential logic.)*	[Marks-2]	CLO-3 *Level-3*
	b)	On a Ferris wheel ride in an amusement park, visitors line up for a ride. The queue of people is managed as a Linear Deque (Double-Ended Queue) using an array, where each element represents a visitor's ticket number. The visitors can join from either end (front or rear), but can leave only from the front end (Output restricted). Initially, the queue is empty (front = -1 and rear = -1). Perform the following operations in the given sequence: 1. InsertRear (10) 2. InsertRear (20) 3. InsertFront (5) 4. DeleteFront 5. insertRear (30) 6. DeleteRear 7. InsertFront (15) 8. InsertFront (25) 9. DeleteFront 10. InsertRear (40) 11. DeleteRear 12. InsertFront (35) **Show** the sequence of elements in the deque after each operation (indicate front and rear positions).	[Marks 4]	CLO-3 *Level-3*
	c)	A hospital emergency room operates a priority queue for patients using a linked list. Patients arrive in the following order with severity scores: A1(4), A2(2), A3(6), A4(1), A5(5), A6(7), A7(3), A8(6), A9(9), A10(5), where a higher score indicates greater severity (and therefore higher priority for treatment). Throughout the shift, doctors treat patients strictly by severity, and among patients with the same severity score, whoever arrived earlier is treated first (FIFO). i. Insert patients A1 to A6 into the priority queue one at a time and **show** the priority queue using a linked list after each insertion. ii. Before the remaining patients arrive, three patients are treated and leave the queue. **Perform** these three dequeue operations and **show** the queue's linked list after each dequeue with updated front and rear. iii. Now insert the remaining patients A7 to A10 into the queue in their arrival order. **Show** the final linked list after insertion with front and rear pointers, ensuring priority order is maintained, and FIFO is preserved for patients with equal severity. iv. At the end of the shift, dequeue all remaining patients until the queue is empty. **Show** the exact order in which patients are treated one by one.	[Marks-4]	CLO-3 *Level-3*
4.		A Web Browser manages open tabs using a circular linked list structures to handle tab switching, backward/forward navigation, and memory cleanup. Initially, four tabs T_1, T_2, T_3, and T_4 are stored at memory addresses 1000, 1002, 1004, and 1006 respectively.		
	a)	**Construct** Circular Linked List using a header node with flag to store the tabs.	[Marks-2]	
	b)	I. Insert tab T_0 at the beginning with address 1008 II. Insert tab T_5 at the 4th position with address 1010 **Sketch** the updated Circular linked list from question number a	[Marks-2]	CLO-3 *Level-3*
	c)	**Customize** the updated circular linked list from question b into a Doubly linked list.	[Marks-2]	
	d)	I. Delete the tab at 2nd position. II. Delete the tab at 5th position. **Sketch** the updated Doubly linked list from question number c	[Marks-2]	
	e)	**Construct** the main part of the C code to construct the Singly linked list. (no need to write the full code)	[Marks-2]	