Roll Number:_____

Thapar Institute of Engineering & Technology

Department of Computer Science and Engineering

AUXILIARY EXAMINATION

B. E. **(2nd Yr. COE/CSE)**
21st Feb., 2025
Friday, Time– 5:30 PM To 8:30 PM
Time: 3 Hours, Max Marks: 100

Course Code: UCS301
Course Name: Data Structures
Name of Faculty: Tarunpreet Bhatia

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

Q.No	Questions	MM	CO	BL
Q.1	(a) Demonstrate the insertion sort results for each insertion for the following initial array of elements. A= [25, 6, 15, 12, 8, 34, 9, 18, 2]. Show the result after each pass.	(5)	CO2	L2
	(b) Given two sorted arrays A and B having numbers of elements m and n, respectively. Write a pseudo-code/algorithm of linear time complexity to merge A and B into a third array C, such that C is also sorted and having m + n number of elements. Show the working of your designed algorithm on the following array: A[] = {5, 8, 9}, B[] = {4, 7, 8}, C[]={4, 5, 7, 8, 8, 9}.	(10)	CO2	L2
Q.2	(a) Write an algorithm/pseudocode to print all the elements at the index of multiples of k with the first element assumed to have an index of 0. Do this for a single pass of the linked list. Input: k=3, 12 -> 15 -> 18 -> 17 -> 19 -> 20 -> 22 -> NULL Output: 12 -> 17 -> 22 -> NULL	(4)	CO1	L2
	(b) Extend the above solution algorithm assuming that the list is circular and the N^{th} index is the same as 0^{th} index. You may need multiple passes. However, every number should be printed only once during its first selection. Input: k=3, 12 -> 15 -> 18 -> 17 -> 19 -> 20 -> 22 -> NULL Output: 12 -> 17 -> 22 -> 18 -> 20 -> 15 -> 19 -> NULL	(6)	CO1	L2
Q.3	You are given a weighted directed graph G = (V, E), where each node v∈V represents a city, and each directed edge e∈E represents a road connection between two cities. Every directed edge e has a positive weight w(e)>0, which indicates the travel cost for that road (such as toll fees, fuel costs, etc.). Additionally, each node v has an associated visitation cost c(v)>0, representing expenses incurred while visiting that city (e.g., lodging, food, or miscellaneous costs). Your task is to plan a trip from a source city A∈V to a destination city F∈V while minimizing the total trip cost using Dijkstra's algorithm. The trip cost includes: a) The sum of edge weights along the path from s to t, representing travel costs. b) The sum of visitation costs for all intermediate cities in the path (i.e., all cities visited except the starting city s and the destination city t). Edge weights are given as below: A → B = 4, A → C = 2, B → C = 5, B → D = 10, C → E = 3, D → F = 11, E → D = 4, E → F = 5 Design the algorithm for the above problem and find the path from A to F that minimizes the combined cost of both travel and intermediate city expenses. Assume visitation cost of each vertex is 5. Ensure that the solution accounts for the weighted nature of both edges and nodes.	(7+8)	CO4	L6
Q.4	(a) Write the algorithm to clone the element of a source stack S to destination stack D maintaining the same order without using the extra space. Also show the step by step process for example S = {2, 5, 7, 9} where 9 corresponds to top of the stack.	(5)	CO4	L3
	(b) Evaluate the following postfix expression using stack: 6 2 3 + – 3 8 2 / + * 2 ↑ 3 +. Show intermediate steps.	(5)	CO1	L3

Q.5	(a) Given the code to implement double-ended queue (deque) using circular array that supports the following operations (i) isFull(): Deque is full (ii) insertFront(int x): Insert an element x at the front of deque (iii) isEmpty(): Deque is empty and (iv) deleteRear(): Remove an element from the rear. Assume array indexing starts from 0 and front and rear are initialized to -1. You need not to write entire code just mention the statements corresponding to the fill in the blanks (1-10). What will be the time complexity of these 4 functions?	(12)	CO1	L2

```
const int MAX_SIZE = 100;
int deque[MAX_SIZE];
int front = -1;
int rear = -1;
bool isFull() {
  if (_____1____== front) {
    return true;
  }
  return false;
}
void insertFront(int x) {
  if (!isFull()) {
    if (front == -1) {
      _____2_____;
      deque[front] = x;
    }
    else {
      if (_____3_____) {
        front = MAX_SIZE - 1;
      } else {
        _____4_____;
      }
      _____5_____; }}}
```

```
bool isEmpty() {
  if (_____6_____) {
    return true;
  }
  return false;
}

void deleteRear() {
  if (!isEmpty()) {
    if (_____7_____) {
      _____8_____;
    }
    else {
      if (_____9_____) {
        rear = MAX_SIZE - 1;
      } else {
        _____10_____;
      }
    }
  }
}
```

	(b) Consider the following sequence of integers: 1, 2, 3, 4, 5, 6, 7, 8, 9. You are required to construct an AVL tree from this sequence and show the necessary rotations at each step to maintain the balance property. After inserting all the elements, perform a pre-order order traversal of the final AVL tree. Further, delete the elements 6 and 8 one after the other and show the structure of AVL tree after each deletion. After deleting both the elements, perform a post-order traversal of the final AVL tree.	(12)	CO3	L3
Q.6	(a) A university has N computers connected through M direct wired connections. The IT department wants to check if all computers can communicate with each other, either directly or indirectly. Design an algorithm that uses Depth First Search (DFS) to determine whether all computers can communicate. If not, find how many isolated networks (connected components) exist. Also, apply your algorithm to the given adjacency matrix representation of the network (refer Fig 1) in which computers are labelled as A-G. Show all intermediate steps during the process. Follow lexicographic order whenever there is a choice of selecting a computer.	(8+8)	CO4	L3

	A	B	C	D	E	F	G
A	0	1	1	1	0	0	0
B	1	0	1	1	0	0	0
C	1	1	0	1	0	0	1
D	1	1	1	0	0	0	1
E	0	0	0	0	0	1	0
F	0	0	0	0	1	0	0
G	0	0	1	1	0	0	0

Fig. 1.

	(b) 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.	(10)	CO3	L3