Lab	Type	Practical
Hands on practice to get familiar with basic C programming concepts.		
1	A A A B B C	1. Write a program to find factorial of a number using loop and recursion. 2. Write a program to find factors of a given number. 3. Write a program to check whether a number is prime or not. 4. Write a program to find GCD using loop and recursion. 5. Write a program to calculate power using loop and recursion. 6. Write a program to display prime numbers between two intervals.
Regular operations on 1-D Array Data Structure.		
2	A A A B B C C	1. Read n numbers in an array and print their sum and average. 2. Write a program to find the largest element in an array. 3. Read n numbers in an array then read two different numbers, replace 1st number with 2nd number in an array and print the final array. 4. Write a program to copy all the elements of one array to another array. 5. Read n numbers in an array and print it in ascending order. 6. Write a program to find common elements between two arrays. 7. Write a program to remove duplicates from sorted array.
Regular operations on 2-D Array Data Structure.		
3	A B C	1. Read two 2x2 matrices and perform addition of matrices into third matrix and print it. 2. Read two matrices, first 3x2 and second 2x3, perform multiplication operation and store result in third matrix and print it. 3. Write a program to find transpose of a square matrix.
Implementation of the Pointer concept		
4	A A A B C	1. Read n numbers in an array and print it using pointer. 2. Write a C program to swap two numbers, calling an UDF by value. 3. Write a C program to swap two numbers, calling an UDF by reference. 4. Write a program to find largest element in the array using Pointer. 5. Write a program to check if the string is a palindrome or not using Pointer.
Implementation of the Structure concept		
5	A B	1. Create a structure Employee_Detail (Employee_id, Name, Designation, Salary). Write a program to read the detail from user and print it. 2. Create an array of structure Student_Detail (Enrollment_no, Name, Sem, CPI) for 5 students, scan their information and print it.

	B C	3. Create a structure Employee_Detail (Employee_id, Name, Designation, Salary). Write a program to read the detail from user and print it using Structure Pointer. 4. Write a program to add two complex numbers by passing structure to a Function.
Implementation of Stack		
6	A B	1. Write a menu driven program to perform following operations on Stack: PUSH, POP, PEEP, CHANGE and DISPLAY. 2. Write a program to reverse a string using Stack.
Implementation of Stack application: Infix to Postfix Expression Conversion		
7	A	1. Write a program to convert the given infix notation to postfix notation.
Implementation of Stack application: Infix to Prefix Expression Conversion		
8	A	1. Write a program to convert the given infix notation to prefix notation.
Implementation of Stack application: Evaluating Postfix & Prefix Expression		
9	A A	1. Write a program for evaluation of post-fix Expression. 2. Write a program for evaluation of pre-fix Expression.
Implementation of Data Structure Simple Queue		
10	A	1. Write a menu driven program to perform following operations on Simple Queue: ENQUEUE, DEQUEUE and DISPLAY.
Implementation of Data Structure Circular Queue		
11	A	1. Write a menu driven program to perform following operations on Circular Queue: ENQUEUE, DEQUEUE and DISPLAY.
Implementation of Data Structure Double Ended Queue		
12	A	1. Write a menu driven program to perform following operations on Double Ended Queue: • ENQUEUE Front • ENQUEUE Rear • DISPLAY • DEQUEUE Front • DEQUEUE Rear

Implementation of Data Structure Priority Queue		
13	A	1. Write a menu driven program to perform following operations on Priority Queue: ENQUEUE, DEQUEUE and DISPLAY.
Implementation of Dynamic Memory Allocation concept		
14	A A	1. Write a program to get n elements of an array from user and print those elements using pointer. 2. Write a program to display n elements and sum of those elements using dynamic memory allocation. Also release the memory occupied after displaying.
Implementation of Data Structure Singly Linked List – Insertion		
15	A	1. Write a menu driven program to implement following operations on the singly linked list: • Insert a node at the beginning of the linked list • Insert a node at the end of the linked list • Display the list • Count number of nodes
Implementation of Data Structure Singly Linked List – Deletion		
16	A	1. Write a menu driven program to implement following operations on the singly linked list: • Delete the first node of the linked list • Delete the last node of the linked list • Display the list • Delete a specific node
Implementation of Ordered Linked List		
17	A	1. Write a menu driven program to implement following operations on the Ordered linked list: • Insert a node • Delete a node • Display the list • Count number of nodes
Advanced Operations on Singly Linked List		

18	A A B B	1. Write a program to sort elements of a linked list. 2. Write a program to remove the duplicates nodes from given sorted Linked List. Input: 1 → 1 → 6 → 13 → 13 → 13 → 27 → 27 Output: 1 → 6 → 13 → 27 3. Write a program to copy a linked list. 4. Write a program to reverse a linked list.
Implementation of Stack and Queue using Linked List		
19	A A	1. Write a program to implement stack using linked list. 2. Write a program to implement queue using linked list.
Implementation of Data Structure Circular Linked List - Insertion		
20	A	1. Write a menu driven program to implement following operations on the Circular Linked List. • Insert a node at the beginning of the circular linked list • Insert a node at the end of the circular linked list • Display the list • Count the nodes
Implementation of Data Structure Circular Linked List - Deletion		
21	A	1. Write a menu driven program to implement following operations on the Circular Linked List. • Delete a node at the beginning of the circular linked list • Delete a node at the end of the circular linked list • Display the list • Delete a specific node
Implementation of Data Structure Doubly Linked List		
22	A	1. Write a menu driven program to implement following operations on the Doubly Linked List. • Insert a node in doubly linked list • Delete a node in doubly linked list • Display the list • Count the number of nodes
Implementation of Searching Techniques		
23	A	1. Write a program to implement Linear/Sequential Search. 2. Write a program to implement Binary Search using loop. 3. Write a program to implement Binary Search using recursion.

Implementation of Sorting Techniques : **Bubble Sort & Selection Sort**		
24	A	4. Read n numbers in an array from user and sort them in ascending order using Bubble Sort algorithm and print sorted array. 5. Read n numbers in an array from user and sort them in ascending order using Selection Sort algorithm and print sorted array.
Implementation of Sorting Techniques : **Quick Sort**		
25	A	1. Read n numbers in an array from user and sort them in ascending order using Quick Sort algorithm and print sorted array.
Implementation of Sorting Techniques : **Merge Sort**		
26	A	1. Read n numbers in an array from user and sort them in ascending order using Merge Sort algorithm and print sorted array.
Implementation of Sorting Techniques : **Insertion Sort & Time Complexity Calculation of all Sorting**		
27	A	1. Read n numbers in an array from user and sort them in ascending order using Insertion Sort algorithm and print sorted array. 2. Discuss and compare Time Complexity of following sorting techniques for all the cases (Best, Worst and Average case): • Bubble sort • Selection sort • Insertion sort • Quick sort • Merge sort
Implementation of Hash Table (HashSet)		
28	A	1. Write a program to implement a Hash Table using the hash function key % m. Use Linear Probing collision-resolution technique. The program must support the following operations: • Insert a key (unique) • Search for a key • Display the hash table
Implementation of HashMap		
29	A	1. Write a program to implement a HashMap using the hash function key % m. Use Linear Probing collision-resolution technique. The program must support the following operations: • Insert a key–value pair • Search for a value using a key • Display all key–value pairs in the HashMap

Solve application-based problems

30 A

1. **Valid Parenthesis Problem**

Chef has a string which contains only the characters '{', '}', '[', ']', '(' and ')'.
Now Chef wants to know if the given string is balanced or not.
If is balanced then print 1, otherwise print 0.
A balanced parenthesis string is defined as follows:

- The empty string is balanced
- If P is balanced then (P), {P}, [P] is also balanced
- if P and Q are balanced PQ is also balanced
- "([])", "({})[()]" are balanced parenthesis strings
- "([{]})", "())" are not balanced.

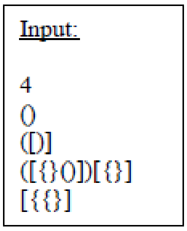
Input Format:

The first line of the input contains a single integer T denoting the number of test cases. The description of T test cases follows. The first and only line of each test case contains a single string

Output Format:

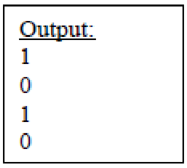
For each test case, print a single line containing the answer.

Sample Example:



```
Input:

4
()
([)]
([{}()])[{}]
[{{}]
```



```
Output:
1
0
1
0
```

B

2. **Merge Intervals Problem**

Given a set of time intervals in any order, our task is to merge all overlapping intervals into one and output the result which should have only mutually exclusive intervals.

Sample Example-1:

Input: Intervals = {{1,3},{2,4},{6,8},{9,10}}

Output: {{1, 4}, {6, 8}, {9, 10}}

Explanation: Given intervals: [1,3],[2,4],[6,8],[9,10], we have only two overlapping intervals here,[1,3] and [2,4]. Therefore, we will merge these two and return [1,4],[6,8], [9,10]

Sample Example-2:

Input: Intervals = {{6,8},{1,9},{2,4},{4,7}}

Output: {{1, 9}}