United International University (UIU)

Dept. of Computer Science & Engineering (CSE)

Set A

CSE 1112: Structured Programming Language Laboratory Fall 2024

Exam Name: Final Exam   Time: 50 Minutes   Total Marks: 25

Any examinee found engaging in unfair practices will be expelled from the trimester / program as per UIU disciplinary rules.

Name: **ID:**

Note: **Answer all the questions.** The numbers on the right side indicate marks for the respective question.

1. Write a C program which **takes a string as user input**, **removes all the duplicate characters** present within the string and finally, **checks whether the string is a palindrome or not**. The individual tasks should be done within specific functions, which are mentioned in the following. 13

- **void removeAllDuplicates(char str[])**
  The function should receive a string as a parameter, and remove all duplicate characters present within said string.
- **int getSize(char str[])**
  The function should receive a string as parameter, and return the size of the string.
- **void checkPalindrome(char *str)**
  The function should receive a string as parameter, and check whether the string is a palindrome or not.

Note that the use of any predefined library functions is not allowed for solving the problem

Sample Input	**Sample Output**
a	a palindrome
h	h palindrome
ssssss	s palindrome
ava	av not palindrome
madam	mad not palindrome
position	positn not palindrome

2.

A university wants to develop a **Student Performance Management System** to keep track of students' academic performance. You need to implement a C program using structures to store and process student records. Each **student** has the following attributes: **name, id, marks(array), average.**

Your task is to implement the following functionalities:

1. **Add a New Student**: Input student details, including name, ID, and **marks for 5 subjects**. Compute and store their **average marks**.

2. **Display All Students**: Show the list of students along with their details and average marks.

3. **Find Top Performer**: Identify and display the student with the highest average marks.

4. **Find Failing Students**: Display students who have at least one subject where marks are below **40**. Also show the count on how many subjects he or she has failed.

5. **Exit**: Terminate the program.

Use an **array of structures** to store up to 100 students. Implement a **menu-driven approach** using if-else/switch case.

Sample Input/Output:

```
Student Performance Management System
1. Add a new student
2. Display all students
3. Find top performer
4. Find failing students
5. Exit

Enter your choice: 1
Enter student name: Alice
Enter student ID: 101
Enter marks for 5 subjects: 85 90 78 88 92
Student added successfully.

Enter your choice: 1
Enter student name: Bob
Enter student ID: 102
Enter marks for 5 subjects: 50 60 30 45 80
Student added successfully.

Enter your choice: 2
Student Records:
ID: 101, Name: Alice, Marks: [85, 90, 78, 88, 92], Average: 86.6
ID: 102, Name: Bob, Marks: [50, 60, 30, 45, 80], Average: 53.0

Enter your choice: 3
Top Performer: Alice, Average Marks: 86.6

Enter your choice: 4
Failing Students:
ID: 102, Name: Bob, Failed Subjects: 1

Enter your choice: 5
Exiting the program.
```

12