SET A

# United International University

Department of Computer Science and Engineering

**Final Examination Fall 2024**

Course Code: **CSE 1112** Course Title: **Structured Programming Language Laboratory**
Date: Feb 11, 2025 (Tuesday) Time: 09:00 AM – 10:00 AM (1 hour) Full marks: 25

---

Name: Student ID:

---

Write down C programs for the following problems in Code Blocks (or any C compiler you prefer), and present the code to your instructor after the time is up. You can make rough calculations in this paper. **Any examinee found adopting unfair means will be expelled from the trimester/program as per UIU disciplinary rules.**

---

**Problem 1 (Marks: 13)**

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.

- **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<br>palindrome |
| h | h<br>palindrome |
| ssssss | s<br>palindrome |

| | |
|---|---|
| ava | av<br>not palindrome |
| madam | mad<br>not palindrome |
| position | positn<br>not palindrome |

## SET A

**Problem 2 (Marks: 12)**

**Student Performance Management System**

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, 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.
```