Student Management System

Specifications:

Variables: Name, age, roll number, and marks.

Static & Const: Static variable for counting students; const for maximum subjects.

Switch Case: Menu options to add, display, and search student details.

Looping Statements: Loop to input and display multiple students.

Pointers: Pointer to dynamically allocate memory for student names.

Functions: Separate functions for adding, displaying, and searching.

Arrays: Store marks for subjects.

Structures: Structure for student details.

Nested Structures: Nested structure for personal and academic details.

Unions: Union to store optional contact information.

Nested Unions: Nested unions for different modes of contact.

Output Expectations: Display student list with their details.

Menu Example:

1. Add Student
2. Display All Students
3. Search Student
4. Exit

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_SUBJECTS 5
#define MAX_STUDENTS 100

static int studentCount = 0;

struct AcademicDetails {
   float marks[MAX_SUBJECTS];
};
```

```
struct PersonalDetails {
    int age;
    char *name;
    union {
        char phone[15];
        char email[50];
    } contact;
    int contactType; // 1 for phone, 2 for email
};

struct Student {
    int rollNumber;
    struct PersonalDetails personal;
    struct AcademicDetails academic;
};

// Array to store student details
struct Student students[MAX_STUDENTS];

// Function Prototypes
void addStudent();
void displayStudents();
void searchStudent();
void displayMenu();

int main() {
    int choice;

    while (1) {
        displayMenu();
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                addStudent();
                break;
            case 2:
                displayStudents();
                break;
            case 3:
                searchStudent();
                break;
            case 4:
                printf("Exiting the program.\n");
                return 0;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    }

    return 0;
}


void addStudent() {
    if (studentCount >= MAX_STUDENTS) {
        printf("Cannot add more students. Maximum limit reached.\n");
        return;
    }
```

```
struct Student *student = &students[studentCount];
student->rollNumber = studentCount + 1;

char tempName[50];
printf("Enter student's name: ");
scanf("%s", tempName);
student->personal.name = (char *)malloc(strlen(tempName) + 1);
strcpy(student->personal.name, tempName);

printf("Enter age: ");
scanf("%d", &student->personal.age);

printf("Enter marks for %d subjects:\n", MAX_SUBJECTS);
for (int i = 0; i < MAX_SUBJECTS; i++) {
   printf("Subject %d: ", i + 1);
   scanf("%f", &student->academic.marks[i]);
}

printf("Enter contact type (1 for phone, 2 for email): ");
scanf("%d", &student->personal.contactType);

if (student->personal.contactType == 1) {
   printf("Enter phone number: ");
   scanf("%s", student->personal.contact.phone);
} else if (student->personal.contactType == 2) {
   printf("Enter email: ");
   scanf("%s", student->personal.contact.email);
} else {
   printf("Invalid contact type.\n");
}
```

```
    studentCount++;
    printf("Student added successfully! Roll Number: %d\n", student->rollNumber);
}

void displayStudents() {
    if (studentCount == 0) {
        printf("No students to display.\n");
        return;
    }

    printf("\n--- Student Details ---\n");
    for (int i = 0; i < studentCount; i++) {
        struct Student *student = &students[i];
        printf("Roll Number: %d\n", student->rollNumber);
        printf("Name: %s\n", student->personal.name);
        printf("Age: %d\n", student->personal.age);
        printf("Marks: ");
        for (int j = 0; j < MAX_SUBJECTS; j++) {
            printf("%.2f ", student->academic.marks[j]);
        }
        printf("\n");
        if (student->personal.contactType == 1) {
            printf("Phone: %s\n", student->personal.contact.phone);
        } else if (student->personal.contactType == 2) {
            printf("Email: %s\n", student->personal.contact.email);
        }
    }
}

void searchStudent() {
    int rollNumber;
    printf("Enter roll number to search: ");
    scanf("%d", &rollNumber);

    for (int i = 0; i < studentCount; i++) {
        if (students[i].rollNumber == rollNumber) {
            struct Student *student = &students[i];
            printf("\n--- Student Found ---\n");
            printf("Roll Number: %d\n", student->rollNumber);
            printf("Name: %s\n", student->personal.name);
            printf("Age: %d\n", student->personal.age);
            printf("Marks: ");
            for (int j = 0; j < MAX_SUBJECTS; j++) {
                printf("%.2f ", student->academic.marks[j]);
            }
            printf("\n");
            if (student->personal.contactType == 1) {
                printf("Phone: %s\n", student->personal.contact.phone);
            } else if (student->personal.contactType == 2) {
                printf("Email: %s\n", student->personal.contact.email);
            }
            return;
        }
    }
    printf("Student with roll number %d not found.\n", rollNumber);
}

void displayMenu() {
    printf("\n--- Student Management System ---\n");
    printf("1. Add Student\n");
    printf("2. Display All Students\n");
    printf("3. Search Student\n");
    printf("4. Exit\n");
}
```