Data Structure Lab Assignment (CS 2172)

Assignment 1: Array Concepts
Time: 1 week

Problem

Under this assignment you are required to create a suitable data-structure and associated functions for maintaining a simple student-register.

The Student Register

- **The Data Structure**: The maximum number of entries (students) that the register should support is a predefined constant. Each entry (student) in the register should consist of the following fields.
    - **name**: a null-terminated string containing the last and first names separated by a comma, for example "Tagore, Rabindranath". This string cannot be empty.
    - **roll**: a 6 digit integer number unique for each student. A roll number cannot start with 0.
    - **telephone**: a null-terminated non-empty string containing a telephone number.
    - **address**: consists of null-terminated strings containing address. These strings can be empty.

  Hint: Without using dynamic memory allocation, the structure definition should be as follows:

```
#define MAX_STUDENTS 10

struct Student {
    char name[20];
    int roll;
    char telephone[11];
    char address[20];
};

struct SReg {
    int count; /* represents valid student entries,
                  initially it should be set to 0 */

    struct Student studentArray[MAX_STUDENTS];
};
```

- **Associated Functions**: Let ***SReg*** and ***Student*** be the user-defined data-types for a student-register and a student, respectively. The Student Register should support the following operations.
    - ***int add(SRegsr, Student s)*** – adds a new student ***s*** to the student register ***sr***. A new student is a student for whom the roll does not already exist in the student register ***sr***. If ***s*** already exists in **sr** (that is, the roll field of ***s*** matches with roll field of some entry of ***sr***), the function returns ***0***, otherwise the function returns ***1***.
    - ***Student get(SReg sr, int r)*** – returns the student from ***sr*** whose roll field matches with ***r***. If there is no such student in ***sr***, the roll field of the returned ***Student*** is 0.
    - ***int delete*(*SRegsr, int r)*** – deletes the student from ***sr*** whose roll field matches with ***r***. If there is no such student in ***sr***, the function returns 0; it returns 1 otherwise.
    - ***int modify*(*SRegsr, Student s)*** – updates the fields of an existing student of ***sr*** whose roll field matches with that of ***s,*** taking values from the corresponding fields of ***s***. The function returns 0 if no such student exists in ***sr***; otherwise it returns 1.
    - ***sortStudents(SRegsr*)** – sorts the students of the student register ***sr*** in alphabetically ascending order of names.
    - ***intgetCount(SRegsr)*** – returns the number of students in the student register ***sr***..
    - ***export(SRegsr, String fname)*** – saves the student register ***sr*** to a file having name ***fname***.
    - ***load(SRegsr, String fname)*** – loads students in the student register ***sr*** from the file having name ***fname***. Please note that a file generated by the ***export(SRefsr, String fname)*** function can be used by this function.

C - like syntax has been used in the above specification of the student register. Please note that they are just suggestive. You apply your own judgement in choosing the data types of the functions and there parameters.

Implement the student register as a C program with all the above mentioned functionalities. Write a suitable main() function for demonstrating that your student register program supports all the features.