Q1.

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>


#define MAX_COURSES 4

#define MAX_STUDENTS 100


typedef struct {

    char name[50];

    int credits;

    int grade;

} Course;


typedef struct {

    int rollno;

    char name[50];

    char dept[10];

    Course courses[MAX_COURSES];

    int course_count;

    float gpa;

} Student;


Student students[MAX_STUDENTS];

int student_count = 0;

void readStudentsFromFile(const char *filename) {

    FILE *file = fopen(filename, "r");

    if (!file) {

        printf("Error opening file.\n");

        return;

    }


    student_count = 0;

    while (fscanf(file, "%d,%49[^,],%9[^,]", &students[student_count].rollno, students[student_count].name,
students[student_count].dept) == 3) {

        for (int i = 0; i < MAX_COURSES; i++) {

            if (fscanf(file, ",%49[^,],%d,%d", students[student_count].courses[i].name,
&students[student_count].courses[i].credits, &students[student_count].courses[i].grade) != 3) {

                break;
            }
            students[student_count].course_count++;
        }
        student_count++;
    }
    fclose(file);
}

void writeStudentsToFile(const char *filename) {
    FILE *file = fopen(filename, "w");
    if (!file) {
        printf("Error opening file.\n");
        return;
    }

    for (int i = 0; i < student_count; i++) {
        fprintf(file, "%d,%s,%s", students[i].rollno, students[i].name, students[i].dept);
        for (int j = 0; j < students[i].course_count; j++) {
            fprintf(file, ",%s,%d,%d", students[i].courses[j].name, students[i].courses[j].credits,
students[i].courses[j].grade);
        }
        fprintf(file, "\n");
    }
    fclose(file);
}
void insertStudent() {
    if (student_count >= MAX_STUDENTS) {
        printf("Maximum student limit reached.\n");
        return;
    }

    Student new_student;
    printf("Enter roll number: ");
    scanf("%d", &new_student.rollno);
    printf("Enter name: ");
    scanf("%s", new_student.name);
    printf("Enter department: ");
    scanf("%s", new_student.dept);
    printf("Enter number of courses (3 or 4): ");
    scanf("%d", &new_student.course_count);
    if (new_student.course_count < 3 || new_student.course_count > 4) {
        printf("Invalid number of courses.\n");
        return;
    }

    for (int i = 0; i < new_student.course_count; i++) {
        printf("Enter course %d name: ", i + 1);
        scanf("%s", new_student.courses[i].name);
        printf("Enter course %d credits: ", i + 1);
        scanf("%d", &new_student.courses[i].credits);
        printf("Enter course %d grade: ", i + 1);
        scanf("%d", &new_student.courses[i].grade);
    }

    students[student_count++] = new_student;
    writeStudentsToFile("students.txt");
}
void calculateGPA(Student *student) {
    int total_credits = 0;
    int total_points = 0;

    for (int i = 0; i < student->course_count; i++) {
        total_credits += student->courses[i].credits;
        total_points += student->courses[i].credits * student->courses[i].grade;
    }

    student->gpa = (float)total_points / total_credits;
}

void calculateAllGPAs() {
    for (int i = 0; i < student_count; i++) {
        calculateGPA(&students[i]);
    }
    writeStudentsToFile("students.txt");
}
```

```
void deregisterCourse(int rollno) {
    for (int i = 0; i < student_count; i++) {
        if (students[i].rollno == rollno && students[i].course_count == 4) {
            printf("Enter course name to deregister: ");
            char course_name[50];
            scanf("%s", course_name);

            int found = 0;
            for (int j = 0; j < students[i].course_count; j++) {
                if (strcmp(students[i].courses[j].name, course_name) == 0) {
                    found = 1;
                    for (int k = j; k < students[i].course_count - 1; k++) {
                        students[i].courses[k] = students[i].courses[k + 1];
                    }
                    students[i].course_count--;
                    break;
                }
            }

            if (!found) {
                printf("Course not found.\n");
            } else {
                writeStudentsToFile("students.txt");
                printf("Course deregistered successfully.\n");
            }
            return;
        }
    }
    printf("Student with roll number %d having four courses not found.\n", rollno);
}
void insertCourse(int rollno) {
    for (int i = 0; i < student_count; i++) {
        if (students[i].rollno == rollno && students[i].course_count == 3) {
            printf("Enter new course name: ");
            scanf("%s", students[i].courses[students[i].course_count].name);
            printf("Enter new course credits: ");
            scanf("%d", &students[i].courses[students[i].course_count].credits);
            printf("Enter new course grade: ");
            scanf("%d", &students[i].courses[students[i].course_count].grade);

            students[i].course_count++;
            writeStudentsToFile("students.txt");
            printf("Course inserted successfully.\n");
            return;
        }
    }
    printf("Student with roll number %d having three courses not found.\n", rollno);
}
void updateCourseName() {
    for (int i = 0; i < 2; i++) {
        printf("Enter roll number for student %d: ", i + 1);
        int rollno;
        scanf("%d", &rollno);

        int found = 0;
        for (int j = 0; j < student_count; j++) {
            if (students[j].rollno == rollno) {
                printf("Enter old course name to update: ");
                char old_name[50];
                scanf("%s", old_name);
                printf("Enter new course name: ");
                char new_name[50];
                scanf("%s", new_name);

                for (int k = 0; k < students[j].course_count; k++) {
                    if (strcmp(students[j].courses[k].name, old_name) == 0) {
                        strcpy(students[j].courses[k].name, new_name);
                        found = 1;
                        break;
                    }
                }

                if (!found) {
                    printf("Course not found for student %d.\n", rollno);
                } else {
                    printf("Course name updated successfully.\n");
                }
                break;
            }
        }

        if (!found) {
            printf("Student with roll number %d not found.\n", rollno);
        }
    }
    writeStudentsToFile("students.txt");
}
void upgradeGrade(int rollno) {
    for (int i = 0; i < student_count; i++) {
        if (students[i].rollno == rollno) {
            for (int j = 0; j < students[i].course_count; j++) {
                if (students[i].courses[j].grade == 7) {
                    students[i].courses[j].grade = 8;
                }
            }
            writeStudentsToFile("students.txt");
            printf("Grades upgraded successfully.\n");
            return;
        }
    }
    printf("Student with roll number %d not found.\n", rollno);
}
void generateGradeReport(int rollno) {
    for (int i = 0; i < student_count; i++) {
        if (students[i].rollno == rollno) {
            printf("Grade Report for Roll Number: %d\n", rollno);
            printf("Name: %s\n", students[i].name);
            printf("Department: %s\n", students[i].dept);
            printf("Courses:\n");
            for (int j = 0; j < students[i].course_count; j++) {
                printf("%s: Credits = %d, Grade = %d\n", students[i].courses[j].name, students[i].courses[j].credits,
students[i].courses[j].grade);
            }
            printf("GPA: %.2f\n", students[i].gpa);
            return;
        }
    }
    printf("Student with roll number %d not found.\n", rollno);
}
void menu() {
    int choice;
    do {
        printf("\n1. Insert Student Records\n");
        printf("2. Calculate GPAs\n");
        printf("3. Deregister a Course\n");
        printf("4. Insert a New Course\n");
        printf("5. Update Course Names\n");
        printf("6. Upgrade Grade\n");
        printf("7. Generate Grade Report\n");
        printf("8. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                insertStudent();
                break;
            case 2:
                calculateAllGPAs();
                break;
            case 3: {
                int rollno;
                printf("Enter roll number: ");
                scanf("%d", &rollno);
                deregisterCourse(rollno);
                break;
            }
            case 4: {
                int rollno;
                printf("Enter roll number: ");
                scanf("%d", &rollno);
                insertCourse(rollno);
                break;
            }
            case 5:
                updateCourseName();
                break;
            case 6: {
                int rollno;
                printf("Enter roll number: ");
                scanf("%d", &rollno);
                upgradeGrade(rollno);
                break;
            }
            case 7: {
                int rollno;
                printf("Enter roll number: ");
                scanf("%d", &rollno);
                generateGradeReport(rollno);
                break;
            }
            case 8:
                printf("Exiting...\n");
                break;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 8);
}

int main() {
    readStudentsFromFile("students.txt");menu();
return 0;}
```

Q2.

```
    CREATE TABLE student (
    rollnum INT PRIMARY KEY,
    name VARCHAR(50),
    dept VARCHAR(10),
    dob DATE NOT NULL,
    email VARCHAR(50) CHECK (email LIKE '%@nitt.edu'),
    course1 VARCHAR(50),
    course2 VARCHAR(50),
    course3 VARCHAR(50),
    course4 VARCHAR(50)
);
INSERT INTO student (rollnum, name, dept, dob, email, course1, course2, course3, course4)
VALUES
(106122122, 'sudhanshu', 'cse', '2022-08-22', '106122122@nitt.edu', 'DBMS', 'OS', 'CYK', 'FLAT'),
(106122120, 'sudhanshu', 'cse', '2022-08-22', '106122120@nitt.edu', 'DBMS', 'M1', 'M2', 'CHEM'),
(106122123, 'sudhanshu', 'cse', '2022-08-22', '106122123@nitt.edu', 'DBMS', 'PHYSICS', 'CHEM', 'MECH'),
(106122124, 'sudhanshu', 'cse', '2022-08-22', '106122124@nitt.edu', 'ROL', 'THKI', 'CHIK', 'M3');

ALTER TABLE student
DROP COLUMN course2,
DROP COLUMN course3;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| rollnum | int         | NO   | PRI | NULL    |       |
| name    | varchar(50) | YES  |     | NULL    |       |
| dept    | varchar(10) | YES  |     | NULL    |       |
| dob     | date        | NO   |     | NULL    |       |
| email   | varchar(50) | YES  |     | NULL    |       |
| course1 | varchar(50) | YES  |     | NULL    |       |
| course4 | varchar(50) | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+

ALTER TABLE student
MODIFY course1 VARCHAR2(50);

ALTER TABLE student
RENAME COLUMN rollnum TO std_rno;

UPDATE student
SET course1 = 'OS'
WHERE course1 = 'DBMS';

DELETE FROM student
WHERE name LIKE 'S%';

SELECT * FROM student
```

```
WHERE dob > '2005-01-01';

DROP TABLE student;

TRUNCATE TABLE student;
```