Assignment 1

The ATM Secretariat has decided to implement a new database to manage information about students, courses taught, and the relationships between them. The new database will be managed through a query and update system called ATMSQL.

For this purpose, your help has been sought to design and implement the essential functionalities of the system.

Data Structures

The proposed system will be modeled using the following structures:

1. **Secretariat**

This structure represents the main database and centralizes:

- The list of all students enrolled at the university.
- The list of all courses taught at the university.
- Details about students' enrollment in courses.

Structure Definition:

```
typedef struct secretariat {
        student* students;              // Student vector
        int nr_students;                // Total number of students
        course* courses;                // Course vector
        int nr_courses;                 // Total number of courses
        enrollment* enrollments;        // Enrollment vector (student-course relationships)
        int nr_enrollments;             // Total number of enrollments
} secretariat;
```

2. **Student**

The student structure describes each student enrolled at the university and has the following fields:

- id: A unique identifier for each student.
- name: The full name of the student.
- year_study: The student’s year of study (1, 2, 3, or 4).
- type : The type of study program:
    - 'b' for budget.

    - 't' for tuition.
- grade : The student's overall average, calculated based on grades from all courses.

Structure Definition:

```
typedef struct student {
        int id;                 // Unique student ID
        char name[40];          // Full name
        int year_study;         // Year of study
        char type;              // 'b' (budget) or 't' (tuition)
        float grade;            // Overall average
} student;
```

3. **Course**

The course structure describes details about each course taught at the university:

- id: A unique identifier for the course.
- name: The course name.
- professor_name: The name of the lead professor.

Structure Definition:

```
typedef struct course {
        int id;                         // Unique course ID
        char  name[30];                 // Course name
        char professor_name[30];        // Lead professor name
} course ;
```

4. **Enrollment**

The enrollment structure describes the many-to-many relationship between students and courses. Each enrollment contains:

- id_student: The ID of the student enrolled in the course.
- id_course: The ID of the course the student is enrolled in.
- grades: A vector with three grades:
    - The grade for laboratory and homework.
    - The grade for the partial exam.
    - The grade for the final exam.

The sum of these grades represents the student's grade for that course. For example, a student with grades 2.40, 2.00, and 3.70 for a course will have a total grade of 8.10/10 for that course.

Structure Definition:

```
typedef struct enrollment{
        int id_student;         // Student ID
        int id_course;          // Course ID
        float grades[3];        // Student grades (laboratory, partial, final)
} enrollment;
```

Task 1: Database Management

1.1. Loading the database from a file

Write a function that reads data from a file organized according to the described structure and loads it into a dynamically allocated secretariat structure.

secretariat *read_secretariat(const char *file_name);

1.2. Adding a student

Write a function that adds a student to the database passed as a parameter.

void add_student(secretariat *s, int id, char *name, int study_year, char status, float average_grade);

1.3. Freeing memory (2 points)

To prevent memory leaks, you need to implement a function that frees the dynamically allocated memory for the secretariat structure and its elements.

void free_secretariat(secretariat **s);

Task 2: Queries, Updates, Deletions

Complete file example:

```
[STUDENTS]
 0, Andrei Popescu, 2, b
 1, Ioana Ionescu, 1, t

[COURSES]
 0, MCU, Medvei Mirabela
 1, Programming_languages, Toma Stefan
```

[ENROLLMENTS]

```
 1, 1, 3.10 3.80 2.10
 2, 2, 2.65 1.20 3.00
```