class Student {

private String name;

private String ID;

private List<Course> enrolledCourses = new ArrayList<>();

private Map<Course, Integer> grades = new HashMap<>();

// getters and setters...

public void enrollCourse(Course course) {

this.enrolledCourses.add(course);

Course.incrementEnrolledStudents();

}

public void assignGrade(Course course, int grade) {

this.grades.put(course, grade);

}

}

class Course {

private String courseCode;

private String name;

private int maxCapacity;

private static int totalEnrolledStudents = 0;

// getters...

public static int getTotalEnrolledStudents() {

return totalEnrolledStudents;

}

public static void incrementEnrolledStudents() {

totalEnrolledStudents++;

}

}

class CourseManagement {

private static List<Course> courses = new ArrayList<>();

private static Map<Student, Double> overallGrades = new HashMap<>();

public static void addCourse(String courseCode, String name, int maxCapacity) {

Course course = new Course(courseCode, name, maxCapacity);

courses.add(course);

}

public static void enrollStudent(Student student, Course course) {

student.enrollCourse(course);

}

public static void assignGrade(Student student, Course course, int grade) {

student.assignGrade(course, grade);

}

public static void calculateOverallGrade(Student student) {

// calculate overall grade based on the grades of the student

}

}

Explanation

The provided code is a simplified representation of a Course Enrollment and Grade Management System for a university, written in Java. It consists of three classes: `Student`, `Course`, and `CourseManagement`. The `Student` class encapsulates the details of a student, including their name, ID, enrolled courses, and grades. It provides methods to enroll a student in a course and assign a grade to a student for a course. The `Course` class encapsulates the details of a course, including the course code, name, and maximum capacity. It also maintains a static variable to keep track of the total number of enrolled students across all instances of the `Course` class, and provides a static method to retrieve this information.

The `CourseManagement` class acts as the controller, managing the list of courses and the overall grades for each student. It provides static methods to add new courses, enroll students, assign grades, and calculate overall course grades for each student. The `addCourse` method creates a new `Course` object and adds it to the list of courses. The `enrollStudent` and `assignGrade` methods call the corresponding methods in the `Student` class to enroll a student in a course and assign a grade to a student for a course, respectively. The `calculateOverallGrade` method calculates the overall course grade for a student based on the grades assigned to them. This design demonstrates effective utilization of object-oriented principles, including encapsulation of data and behavior within classes, and use of static methods and variables to track information across multiple instances.