Programming Assignment Unit 5

Godfrey Ouma

University of the People

CS 1102: Programming 1

Ruth Alabi

December 21, 2023

```java
package textio;

import java.util.ArrayList;
import java.util.List;

class Student {
    private String name;
    private int id;
    private List<Course> enrolledCourses;

    // Constructor to initialize Student object with name and id
    public Student(String name, int id) {
        this.name = name;
        this.id = id;
        this.enrolledCourses = new ArrayList<>();
    }

    // Getter method for student's name
    public String getName() {
        return name;
    }

    // Setter method for student's name
    public void setName(String name) {
        this.name = name;
    }

    // Getter method for student's ID
    public int getId() {
        return id;
    }

    // Setter method for student's ID
    public void setId(int id) {
        this.id = id;
    }

    // Method to enroll students in courses
    public void enrollCourse(Course course) {
        enrolledCourses.add(course);
    }

    // Method to assign grades to students for a particular course
    public void assignGrade(Course course, int grade) {
        // Logic to assign grade to the student for a particular course
    }
}

class Course {
    private String courseCode;
    private String name;
    private int maxCapacity;
    private static int totalEnrolledStudents = 0;

    // Constructor to initialize Course object with courseCode, name, and maxCapacity
    public Course(String courseCode, String name, int maxCapacity) {
        this.courseCode = courseCode;
        this.name = name;
        this.setMaxCapacity(maxCapacity);
```

```
    }

    // Getter method for course code
    public String getCourseCode() {
        return courseCode;
    }

    // Getter method for course name
    public String getName() {
        return name;
    }

    // Static method to retrieve total enrolled students
    public static int getTotalEnrolledStudents() {
        return totalEnrolledStudents;
    }

    public int getMaxCapacity() {
        return maxCapacity;
    }

    public void setMaxCapacity(int maxCapacity) {
        this.maxCapacity = maxCapacity;
    }
}

class CourseManagement {
    private static List<Course> courses = new ArrayList<>();
    private static List<Double> overallGrades = new ArrayList<>();

    // Static method to add new courses
    public static void addCourse(String courseCode, String name, int maxCapacity) {
        Course newCourse = new Course(courseCode, name, maxCapacity);
        courses.add(newCourse);
    }

    // Static method to enroll students
    public static void enrollStudent(Student student, Course course) {
        student.enrollCourse(course);
    }

    // Static method to assign grades
    public static void assignGrade(Student student, Course course, int grade) {
        student.assignGrade(course, grade);
    }

    // Static method to calculate overall course grade for a student
    public static double calculateOverallGrade(Student student) {
        // Logic to calculate overall course grade for a student
        return 0.0; // Placeholder return
    }

    public static List<Double> getOverallGrades() {
        return overallGrades;
    }

    public static void setOverallGrades(List<Double> overallGrades) {
        CourseManagement.overallGrades = overallGrades;
    }
```

```
}

class AdministratorInterface {
    // Interactive command-line interface implementation
    // Implement necessary methods to interact with CourseManagement and Student classes
    // Handle user inputs and call appropriate methods
}

public class Students {
    public static void main(String[] args) {
        // Initialize and test the functionalities of the Course Enrollment and Grade
Management System
    }
}
```

Documentation

Classes

Student Class

- **Description:** Represents a student with name, ID, and enrolled courses.
- **Attributes:**
    - name (String): Student's name.
    - id (int): Student's identification number.
    - enrolledCourses (List of Course): List of courses in which the student is enrolled.
- **Methods:**
    - Student(String name, int id): Constructor to initialize a Student object with name and ID.
    - getName(): String: Retrieves the student's name.
    - setName(String name): void: Sets the student's name.
    - getId(): int: Retrieves the student's ID.
    - setId(int id): void: Sets the student's ID.
    - enrollCourse(Course course): void: Enrolls the student in a course.
    - assignGrade(Course course, int grade): void: Assigns a grade to the student for a particular course.

Course Class

- **Description:** Represents a course with a course code, name, and maximum capacity.
- **Attributes:**
    - courseCode (String): Course code.

- name (String): Course name.
- maxCapacity (int): Maximum capacity of the course.

- **Methods:**
  - **Course(String courseCode, String name, int maxCapacity):** Constructor to initialize a Course object with course code, name, and max capacity.
  - **getCourseCode():** String: Retrieves the course code.
  - **getName():** String: Retrieves the course name.
  - **getMaxCapacity():** int: Retrieves the maximum capacity of the course.
  - **getTotalEnrolledStudents():** int: Retrieves the total enrolled students across all courses.

CourseManagement Class

- **Description:** Handles course management operations.
- **Attributes:**
  - courses (List of Course): List of courses.
  - overallGrades (List of Double): List to store overall grades.
- **Methods:**
  - addCourse(String courseCode, String name, int maxCapacity): void: Adds a new course.
  - enrollStudent(Student student, Course course): void: Enrolls a student in a course.
  - assignGrade(Student student, Course course, int grade): void: Assigns a grade to a student for a course.
  - calculateOverallGrade(Student student): double: Calculates the overall course grade for a student.

- getOverallGrades(): List<Double>: Retrieves the list of overall grades.
- setOverallGrades(List<Double> overallGrades): void: Sets the list of overall grades.

AdministratorInterface Class

- **Description:** Implements an interactive command-line interface for administrators.

Usage Instructions

1. Initialize the Course Enrollment and Grade Management System by executing the Students class.
2. Utilize the following classes and methods to perform operations:
    - Student: Manage student information and enrollment.
    - Course: Manage course information and capacity.
    - CourseManagement: Manage course-related operations.
    - AdministratorInterface: Interact with the system through the command-line interface.