```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

class Student {
    private String studentId;
    private String name;
    private List<Course> registeredCourses;

    public Student(String studentId, String name) {
        this.studentId = studentId;
        this.name = name;
        this.registeredCourses = new ArrayList<>();
    }

    public String getStudentId() {
        return studentId;
    }

    public String getName() {
        return name;
    }

    public List<Course> getRegisteredCourses() {
        return registeredCourses;
    }

    public boolean registerCourse(Course course) {
        if (course.registerStudent()) {
            registeredCourses.add(course);
            return true;
```

```
        }
        return false;
    }

    public boolean dropCourse(Course course) {
        if (registeredCourses.remove(course) && course.dropStudent()) {
            return true;
        }
        return false;
    }

    public void listRegisteredCourses() {
        if (registeredCourses.isEmpty()) {
            System.out.println("No courses registered.");
        } else {
            for (Course course : registeredCourses) {
                System.out.println(course);
            }
        }
    }
}

class Course {
    private String courseCode;
    private String title;
    private String description;
    private int capacity;
    private int enrolledStudents;
    private String schedule;

    public Course(String courseCode, String title, String description, int capacity, String schedule) {
```

```
        this.courseCode = courseCode;
        this.title = title;
        this.description = description;
        this.capacity = capacity;
        this.enrolledStudents = 0;
        this.schedule = schedule;
    }

    public String getCourseCode() {
        return courseCode;
    }

    public String getTitle() {
        return title;
    }

    public String getDescription() {
        return description;
    }

    public int getCapacity() {
        return capacity;
    }

    public int getEnrolledStudents() {
        return enrolledStudents;
    }

    public String getSchedule() {
        return schedule;
    }
```

```
public boolean registerStudent() {
    if (enrolledStudents < capacity) {
        enrolledStudents++;
        return true;
    }
    return false;
}

public boolean dropStudent() {
    if (enrolledStudents > 0) {
        enrolledStudents--;
        return true;
    }
    return false;
}

public String getAvailability() {
    return (capacity - enrolledStudents) + " slots available";
}

@Override
public String toString() {
    return "Course Code: " + courseCode + "\n" +
           "Title: " + title + "\n" +
           "Description: " + description + "\n" +
           "Capacity: " + capacity + "\n" +
           "Enrolled: " + enrolledStudents + "\n" +
           "Schedule: " + schedule + "\n" +
           getAvailability() + "\n";
}
```

```
}

public class CourseRegistrationSystem {
    private List<Course> courses;
    private List<Student> students;

    public CourseRegistrationSystem() {
        courses = new ArrayList<>();
        students = new ArrayList<>();
        loadCourses();
    }

    private void loadCourses() {
        courses.add(new Course("CS101", "Introduction to Computer Science", "Learn the basics of programming.", 30, "MWF 10-11 AM"));
        courses.add(new Course("ENG202", "English Literature", "Explore classic and modern literature.", 25, "TTh 1-2:30 PM"));
        courses.add(new Course("MATH301", "Calculus I", "Introduction to derivatives and integrals.", 20, "MWF 9-10 AM"));
        // Add more courses as needed
    }

    public void addStudent(Student student) {
        students.add(student);
    }

    public void displayCourses() {
        System.out.println("Available Courses:");
        for (Course course : courses) {
            System.out.println(course);
        }
    }
```

```
public void registerStudentForCourse(Student student, String courseCode) {
    for (Course course : courses) {
        if (course.getCourseCode().equalsIgnoreCase(courseCode)) {
            if (student.registerCourse(course)) {
                System.out.println("Successfully registered for " + course.getTitle());
            } else {
                System.out.println("Failed to register. Course is full.");
            }
            return;
        }
    }
    System.out.println("Course not found.");
}

public void dropStudentFromCourse(Student student, String courseCode) {
    for (Course course : courses) {
        if (course.getCourseCode().equalsIgnoreCase(courseCode)) {
            if (student.dropCourse(course)) {
                System.out.println("Successfully dropped from " + course.getTitle());
            } else {
                System.out.println("Failed to drop course. Not registered.");
            }
            return;
        }
    }
    System.out.println("Course not found.");
}

public static void main(String[] args) {
    @SuppressWarnings("resource")
```

```
Scanner scanner = new Scanner(System.in);
CourseRegistrationSystem system = new CourseRegistrationSystem();

System.out.print("Enter Student ID: ");
String studentId = scanner.nextLine();
System.out.print("Enter Student Name: ");
String studentName = scanner.nextLine();
Student student = new Student(studentId, studentName);
system.addStudent(student);

while (true) {
    System.out.println("\nOptions:");
    System.out.println("1. Display Courses");
    System.out.println("2. Register for a Course");
    System.out.println("3. Drop a Course");
    System.out.println("4. List Registered Courses");
    System.out.println("5. Exit");
    System.out.print("Choose an option: ");
    int choice = scanner.nextInt();
    scanner.nextLine(); // Consume newline

    switch (choice) {
        case 1:
            system.displayCourses();
            break;
        case 2:
            System.out.print("Enter Course Code to register: ");
            String registerCode = scanner.nextLine();
            system.registerStudentForCourse(student, registerCode);
            break;
        case 3:
```

```
                System.out.print("Enter Course Code to drop: ");
                String dropCode = scanner.nextLine();
                system.dropStudentFromCourse(student, dropCode);
                break;
            case 4:
                System.out.println("Registered Courses:");
                student.listRegisteredCourses();
                break;
            case 5:
                System.out.println("Exiting...");
                return;
            default:
                System.out.println("Invalid option. Please try again.");
        }
    }
  }
}
```