import java.util.\*;

class Classroom {

private String name;

private List<Student> students = new ArrayList<>();

private List<Assignment> assignments = new ArrayList<>();

public Classroom(String name) {

this.name = name;

}

public String getName() {

return name;

}

public void addStudent(Student student) {

students.add(student);

}

public List<Student> getStudents() {

return students;

}

public void addAssignment(Assignment assignment) {

assignments.add(assignment);

}

public List<Assignment> getAssignments() {

return assignments;

}

@Override

public String toString() {

return "Classroom: " + name;

}

}

class Student {

private String id;

private String name;

public Student(String id, String name) {

this.id = id;

this.name = name;

}

public String getId() {

return id;

}

public String getName() {

return name;

}

@Override

public String toString() {

return "Student: " + name + " (ID: " + id + ")";

}

}

class Assignment {

private String title;

private String dueDate;

public Assignment(String title, String dueDate) {

this.title = title;

this.dueDate = dueDate;

}

public String getTitle() {

return title;

}

public String getDueDate() {

return dueDate;

}

@Override

public String toString() {

return "Assignment: " + title + " (Due Date: " + dueDate + ")";

}

}

public class VirtualClassroomManager {

private List<Classroom> classrooms = new ArrayList<>();

private Map<String, Student> students = new HashMap<>();

public void addClassroom(String name) {

classrooms.add(new Classroom(name));

System.out.println("Classroom " + name + " has been created.");

}

public void listClassrooms() {

System.out.println("List of Classrooms:");

for (Classroom classroom : classrooms) {

System.out.println(classroom);

}

}

public void removeClassroom(String name) {

Iterator<Classroom> iterator = classrooms.iterator();

while (iterator.hasNext()) {

Classroom classroom = iterator.next();

if (classroom.getName().equals(name)) {

iterator.remove();

System.out.println("Classroom " + name + " has been removed.");

return;

}

}

System.out.println("Classroom " + name + " not found.");

}

public void addStudent(String id, String name, String className) {

if (students.containsKey(id)) {

System.out.println("Student with ID " + id + " already exists.");

return;

}

Classroom classroom = findClassroom(className);

if (classroom != null) {

Student student = new Student(id, name);

students.put(id, student);

classroom.addStudent(student);

System.out.println("Student " + name + " (ID: " + id + ") has been enrolled in " + className + ".");

} else {

System.out.println("Classroom " + className + " not found.");

}

}

public void listStudents(String className) {

Classroom classroom = findClassroom(className);

if (classroom != null) {

System.out.println("List of Students in " + className + ":");

for (Student student : classroom.getStudents()) {

System.out.println(student);

}

} else {

System.out.println("Classroom " + className + " not found.");

}

}

public void scheduleAssignment(String className, String title, String dueDate) {

Classroom classroom = findClassroom(className);

if (classroom != null) {

Assignment assignment = new Assignment(title, dueDate);

classroom.addAssignment(assignment);

System.out.println("Assignment for " + className + " has been scheduled.");

} else {

System.out.println("Classroom " + className + " not found.");

}

}

public void listAssignments(String className) {

Classroom classroom = findClassroom(className);

if (classroom != null) {

System.out.println("List of Assignments for " + className + ":");

for (Assignment assignment : classroom.getAssignments()) {

System.out.println(assignment);

}

} else {

System.out.println("Classroom " + className + " not found.");

}

}

public void submitAssignment(String studentId, String className, String assignmentTitle) {

Student student = students.get(studentId);

Classroom classroom = findClassroom(className);

if (student != null && classroom != null) {

Assignment assignment = findAssignment(classroom, assignmentTitle);

if (assignment != null) {

System.out.println("Assignment submitted by Student " + studentId + " in " + className + ".");

} else {

System.out.println("Assignment with title " + assignmentTitle + " not found.");

}

} else {

System.out.println("Student or classroom not found.");

}

}

private Classroom findClassroom(String className) {

for (Classroom classroom : classrooms) {

if (classroom.getName().equals(className)) {

return classroom;

}

}

return null;

}

private Assignment findAssignment(Classroom classroom, String assignmentTitle) {

for (Assignment assignment : classroom.getAssignments()) {

if (assignment.getTitle().equals(assignmentTitle)) {

return assignment;

}

}

return null;

}

public static void main(String[] args) {

VirtualClassroomManager manager = new VirtualClassroomManager();

Scanner scanner = new Scanner(System.in);

while (true) {

System.out.println("\nVirtual Classroom Manager Menu:");

System.out.println("1. Add Classroom");

System.out.println("2. List Classrooms");

System.out.println("3. Remove Classroom");

System.out.println("4. Add Student");

System.out.println("5. List Students");

System.out.println("6. Schedule Assignment");

System.out.println("7. List Assignments");

System.out.println("8. Submit Assignment");

System.out.println("9. Exit");

System.out.print("Enter your choice: ");

int choice = scanner.nextInt();

scanner.nextLine(); // Consume newline

switch (choice) {

case 1:

System.out.print("Enter classroom name: ");

String className = scanner.nextLine();

manager.addClassroom(className);

break;

case 2:

manager.listClassrooms();

break;

case 3:

System.out.print("Enter classroom name to remove: ");

className = scanner.nextLine();

manager.removeClassroom(className);

break;

case 4:

System.out.print("Enter student ID: ");

String studentId = scanner.nextLine();

System.out.print("Enter student name: ");

String studentName = scanner.nextLine();

System.out.print("Enter classroom name to enroll in: ");

className = scanner.nextLine();

manager.addStudent(studentId, studentName, className);

break;

case 5:

System.out.print("Enter classroom name to list students: ");

className = scanner.nextLine();

manager.listStudents(className);

break;

case 6:

System.out.print("Enter classroom name to schedule assignment: ");

className = scanner.nextLine();

System.out.print("Enter assignment title: ");

String assignmentTitle = scanner.nextLine();

System.out.print("Enter assignment due date: ");

String dueDate = scanner.nextLine();

manager.scheduleAssignment(className, assignmentTitle, dueDate);

break;

case 7:

System.out.print("Enter classroom name to list assignments: ");

className = scanner.nextLine();

manager.listAssignments(className);

break;

case 8:

System.out.print("Enter student ID: ");

studentId = scanner.nextLine();

System.out.print("Enter classroom name: ");

className = scanner.nextLine();

System.out.print("Enter assignment title: ");

assignmentTitle = scanner.nextLine();

manager.submitAssignment(studentId, className, assignmentTitle);

break;

case 9:

System.out.println("Goodbye!");

scanner.close();

System.exit(0);

default:

System.out.println("Invalid choice. Please try again.");

}

}

}

}