#include <iostream>

#include <vector>

#include <string>

#include <map>

#include <algorithm>

using namespace std;

// Course structure

struct Course {

string name;

int students; // Number of students in the course

string instructor; // Instructor's name

};

// Room structure

struct Room {

string name;

int capacity; // Maximum number of students the room can hold

};

// Instructor structure

struct Instructor {

string name;

vector<int> availableSlots; // Available time slots for the instructor

};

// Time Slot structure

struct TimeSlot {

int slotId; // Unique time slot identifier

bool isAvailable; // Whether the slot is free or taken

};

// Global variables

vector<Course> courses;

vector<Room> rooms;

vector<Instructor> instructors;

vector<TimeSlot> timeSlots;

// Function to take course input from the user

void inputCourses() {

int n;

cout << "Enter the number of courses: ";

cin >> n;

courses.resize(n);

for (int i = 0; i < n; i++) {

cout << "\nEnter course name: ";

cin >> courses[i].name;

cout << "Enter number of students in " << courses[i].name << ": ";

cin >> courses[i].students;

cout << "Enter instructor's name for " << courses[i].name << ": ";

cin >> courses[i].instructor;

}

}

// Function to take room input from the user

void inputRooms() {

int n;

cout << "\nEnter the number of rooms: ";

cin >> n;

rooms.resize(n);

for (int i = 0; i < n; i++) {

cout << "\nEnter room name: ";

cin >> rooms[i].name;

cout << "Enter room capacity for " << rooms[i].name << ": ";

cin >> rooms[i].capacity;

}

}

// Function to take instructor input from the user

void inputInstructors() {

int n;

cout << "\nEnter the number of instructors: ";

cin >> n;

instructors.resize(n);

for (int i = 0; i < n; i++) {

cout << "\nEnter instructor's name: ";

cin >> instructors[i].name;

int m;

cout << "Enter number of available time slots for " << instructors[i].name << ": ";

cin >> m;

instructors[i].availableSlots.resize(m);

cout << "Enter the time slots (ID) for " << instructors[i].name << ": ";

for (int j = 0; j < m; j++) {

cin >> instructors[i].availableSlots[j];

}

}

}

// Function to take time slot input

void inputTimeSlots() {

int n;

cout << "\nEnter the number of time slots: ";

cin >> n;

timeSlots.resize(n);

for (int i = 0; i < n; i++) {

timeSlots[i].slotId = i + 1;

timeSlots[i].isAvailable = true; // Initially all time slots are available

}

}

// Function to assign courses to available time slots

void scheduleCourses() {

map<string, string> courseSchedule; // Course name -> "Room + Time Slot"

// Iterate over each course

for (auto& course : courses) {

bool assigned = false;

// Try to find a suitable time slot

for (auto& slot : timeSlots) {

if (!slot.isAvailable) continue;

// Check if instructor is available in this time slot

bool instructorAvailable = false;

for (auto& instructor : instructors) {

if (instructor.name == course.instructor) {

if (find(instructor.availableSlots.begin(), instructor.availableSlots.end(), slot.slotId) != instructor.availableSlots.end()) {

instructorAvailable = true;

break;

}

}

}

// Check if a room is available and large enough

for (auto& room : rooms) {

if (room.capacity >= course.students) {

// Assign the course

courseSchedule[course.name] = room.name + " at Slot " + to\_string(slot.slotId);

slot.isAvailable = false; // Mark the time slot as taken

assigned = true;

break;

}

}

if (assigned) break; // If the course is assigned, break out of the loops

}

// If not assigned, show a message

if (!assigned) {

cout << "\nUnable to schedule course: " << course.name << " (Instructor or Room not available)\n";

}

}

// Output the course schedule

cout << "\nCourse Schedule:\n";

for (auto& entry : courseSchedule) {

cout << entry.first << " is scheduled in " << entry.second << endl;

}

}

int main() {

cout << "University Course Scheduling System\n";

// Input data

inputCourses();

inputRooms();

inputInstructors();

inputTimeSlots();

// Schedule courses

scheduleCourses();

return 0;

}