**Problem Statement**

**Title:** Doctor-Patient Appointment System at PES University Hospital

In the reknowned PES University Hospital, known for its exceptional doctors and advanced systems, managing appointments between doctors and patients has always been a crucial task. However, as the hospital grew busier, managing these appointments became increasingly challenging, and the existing manual systems were no longer sufficient.

One day, a mysterious programmer from the year 2024 appeared at PES University Hospital. With advanced knowledge of data structures and algorithms, this programmer offered to help modernize the hospital's appointment management system using a structured, multilist approach.

**Task:**

Assume you are the programmer who has traveled back in time to help PES University Hospital. Your task is to develop an advanced appointment management system that can perform the following operations:

1. **Add Doctor:** Add a new doctor to the hospital's records. Each doctor is assigned a unique ID.
2. **Add Patient:** Add a new patient to the hospital's records. Each patient is assigned a unique ID.
3. **Add Appointment (D, P, T):** Schedule an appointment between a doctor (D) and a patient (P) at the specified time (T). The appointment must be stored in a way that allows it to be accessed quickly by both doctor and patient IDs.
4. **Delete Appointment (D, P):** Cancel an appointment between doctor (D) and patient (P). This operation should remove the appointment from both the doctor’s and the patient’s schedules.
5. **List Appointments for Doctor (D):** List all appointments for a specified doctor (D). Display the patient IDs and the corresponding appointment times.
6. **List Appointments for Patient (P):** List all appointments for a specified patient (P). Display the doctor IDs and the corresponding appointment times.

**Additional Information:**

* Each appointment is represented as a node in a multilist where rows represent patients and columns represent doctors.
* The system must be robust, allowing efficient traversal and modifications of the data structure.
* The initial system state is empty with no doctors, patients, or appointments.

**Example Usage:**

* **Add Doctor:** A doctor is added and assigned a unique ID (e.g., Doctor 1).
* **Add Patient:** A patient is added and assigned a unique ID (e.g., Patient 1).
* **Add Appointment (1, 1, "09:00"):** Schedules an appointment for Doctor 1 with Patient 1 at 09:00.
* **List Appointments for Doctor (1):** Shows all appointments for Doctor 1, including Patient IDs and times.

Develop this system to help PES University Hospital efficiently manage its growing patient load and ensure that it remains at the forefront of medical care.