Управление больницей

Описание задачи:
Создайте программу для управления больницей. В больнице есть пациенты, врачи и система назначения визитов.

Требования:

1. Создайте классы для пациентов, врачей и больницы.
2. Реализуйте возможность регистрации пациентов и врачей.
3. Реализуйте систему назначения визитов пациентов к врачам.
4. Храните информацию о назначенных визитах и обрабатывайте их.

Решение:

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

class Patient(Person):
    def __init__(self, name, age, patient_id):
        super().__init__(name, age)
        self.patient_id = patient_id
        self.appointments = []

class Doctor(Person):
    def __init__(self, name, age, specialty):
        super().__init__(name, age)
        self.specialty = specialty
        self.appointments = []

class Appointment:
    def __init__(self, patient, doctor, date):
        self.patient = patient
        self.doctor = doctor
        self.date = date

class Hospital:
    def __init__(self):
        self.patients = []
        self.doctors = []

    def register_patient(self, patient):
        self.patients.append(patient)

    def register_doctor(self, doctor):
        self.doctors.append(doctor)

    def schedule_appointment(self, patient_id, doctor_name, date):
        patient = next((p for p in self.patients if p.patient_id == patient_id), None)
        doctor = next((d for d in self.doctors if d.name == doctor_name), None)

        if patient and doctor:
            appointment = Appointment(patient, doctor, date)
            patient.appointments.append(appointment)
            doctor.appointments.append(appointment)
            print(f"Appointment scheduled for {patient.name} with Dr. {doctor.name} on {date}.")
        else:
            print("Patient or Doctor not found.")

    def show_appointments(self, name, is_doctor=True):
        person = next((d for d in self.doctors if d.name == name), None) if is_doctor else next((p for p
```

```
in self.patients if p.name == name), None)
        if person:
            for appointment in person.appointments:
                print(f"Appointment with {appointment.doctor.name if not is_doctor else
appointment.patient.name} on {appointment.date}")
        else:
            print(f"{name} not found.")

# Пример использования
hospital = Hospital()

patient1 = Patient("Alice", 30, 1)
patient2 = Patient("Bob", 45, 2)
doctor1 = Doctor("Dr. Smith", 50, "Cardiologist")
doctor2 = Doctor("Dr. Jones", 40, "Dermatologist")

hospital.register_patient(patient1)
hospital.register_patient(patient2)
hospital.register_doctor(doctor1)
hospital.register_doctor(doctor2)

hospital.schedule_appointment(1, "Dr. Smith", "2024-06-10")
hospital.schedule_appointment(2, "Dr. Jones", "2024-06-11")

hospital.show_appointments("Dr. Smith")
hospital.show_appointments("Alice", is_doctor=False)
```

Описание решения:

1. **Класс** `Person`: Базовый класс для пациентов и врачей, содержит общие свойства.
2. **Класс** `Patient`: Наследует от `Person`, добавляет идентификатор пациента и список назначенных визитов.
3. **Класс** `Doctor`: Наследует от `Person`, добавляет специальность и список назначенных визитов.
4. **Класс** `Appointment`: Представляет визит пациента к врачу, содержит ссылки на объекты пациента и врача, а также дату визита.
5. **Класс** `Hospital`: Управляет списками пациентов и врачей, реализует методы для регистрации, назначения визитов и отображения визитов.