🏥 Sistema de Gestión de Turnos para una Clínica

📝 Consigna

🎯 Objetivo

Desarrollar un sistema de gestión para una clínica médica que permita:

- Registrar y administrar pacientes y médicos.
- Agendar turnos entre pacientes y médicos.
- Emitir recetas médicas.
- Mantener una historia clínica para cada paciente, que incluya sus turnos y recetas.

📦 Clases y Responsabilidades

✅Paciente

- **Atributos privados**:
  - `_dni` (str)
  - `nombre` (str)
  - `fecha_nacimiento` (str, formato `dd/mm/aaaa`)
- **Métodos**:
  - `obtener_dni()` → `str`
  - `__str__()` → `str`

✅Medico

- **Atributos privados**:
  - `matricula` (str)
  - `nombre` (str)
  - `especialidad` (str)
- **Métodos**:
  - `obtener_matricula()` → `str`
  - `__str__()` → `str`

✅Turno

- **Atributos privados**:
  - `paciente` (Paciente)
  - `medico` (Medico)
  - `fecha_hora` (datetime)
- **Métodos**:
  - `obtener_fecha_hora()` → `datetime`
  - `__str__()` → `str`

✅Receta

- **Atributos privados**:
  - paciente (Paciente)
  - medico (Medico)
  - medicamentos(list[str])
  - _fecha (datetime)
- **Métodos**:
  - __str__() → str

✅HistoriaClinica

- **Atributos privados**:
  - paciente (Paciente)
  - turnos (list[Turno])
  - recetas (list[Receta])
- **Métodos**:
  - agregar_turno(turno: Turno)
  - agregar_receta(receta: Receta)
  - obtener_turnos() → list[Turno]
  - obtener_recetas() → list[Receta]
  - __str__() → str

✅Clinica

- **Atributos privados**:
  - pacientes (dict[DNI → Paciente])
  - medicos (dict[Matrícula → Medico])
  - turnos (list[Turno])
  - historias_clinicas (dict[DNI → HistoriaClinica])
- **Métodos**:
  - agregar_paciente(paciente: Paciente)
  - agregar_medico(medico: Medico)
  - agendar_turno(dni: str, matricula: str, fecha_hora: datetime)
  - emitir_receta(dni: str, matricula: str, medicamentos: list[str])
  - obtener_historia_clinica(dni: str) → HistoriaClinica
  - obtener_turnos() → list[Turno]

Validaciones en Clinica.agendar_turno

- Paciente y médico deben existir.
- No se puede agendar dos turnos con el mismo médico en el mismo fecha_hora.
- (Opcional) No agendar turnos en el pasado.

⚠ Excepciones Personalizadas

- PacienteNoExisteErro
- MedicoNoExisteError
- TurnoDuplicadoError

💻 CLI (Interfaz por Consola)

La clase CLI muestra un menú y llama a los métodos de `Clinica`.
No debe contener validaciones de negocio:

```
Menú Clínica:
1. Agregar paciente
2. Agregar médico
3. Agendar turno
4. Emitir receta
5. Ver historia clínica
6. Ver todos los turnos
7. Ver todos los pacientes
8. Ver todos los médicos
9. Salir
```

🧪 Unit Testing

Implementar pruebas con unittest para cubrir:

- Agregar pacientes y médicos correctamente y errores por duplicados o datos faltantes.
- Agendar turnos válidos.
- Evitar turnos duplicados (mismo médico y hora).
- Verificar excepciones: `PacienteNoExisteError`, `MedicoNoExisteError`, `TurnoDuplicadoError`.
- Emitir recetas válidas y errores si paciente/médico no existen.
- Comprobar que la HistoriaClinica recoge correctamente turnos y recetas.