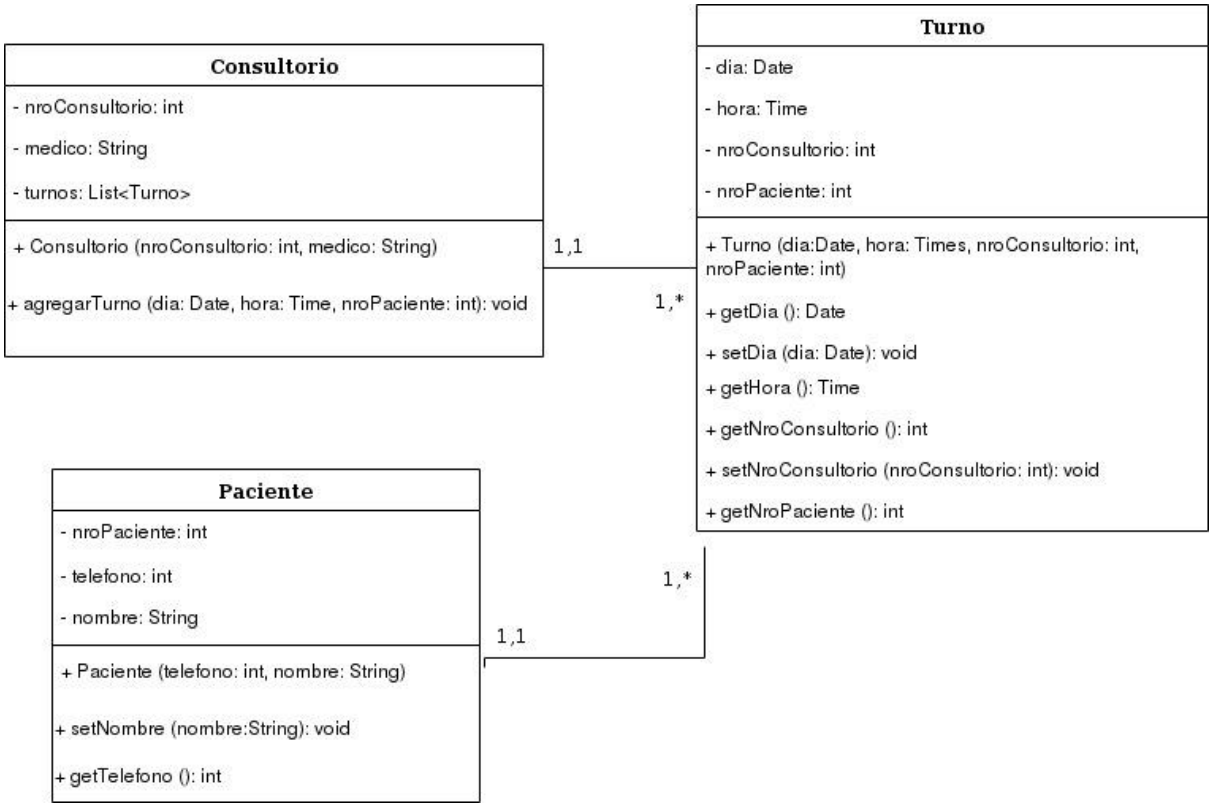
Examen 111Mil - Turnos de Consultorios

Se desea mantener un registro de los turnos de un consultorio médico. Hasta este momento la secretaria mantenía un registro de los mismos en su cuaderno. Como saben que hemos cursado el programa 111MIL nos pidieron que le ayudemos y colaboremos en la implementación

Ejercicio 1. Implementar desde el diagrama de clases

Se ha construido un diagrama UML de la aplicación, se han implementado las clases *Turno* y *Paciente,* pero nos han pedido que implementemos la clase *Consultorio.* Para ello tenga en cuenta que:

a) La lista de turnos debe ser inicializada vacía dentro del constructor.
b) El método *agregarTurno* debe crear y agregar un nuevo turno. En esta versión del sistema no es necesario chequear si ya hay turnos dados para el mismo dia y horario.



Ejercicio 2. Implementar un método a partir de un requerimiento

Eventualmente, alguno de los consultorios se pinta. Por esa razón, es necesario cancelar todos los turnos del consultorio para el dia en que se pinta. En esta versión del sistema no se reagendarán los turnos sino que se cancelan. Indique en qué clase iría dicha funcionalidad e implemente el/los método/s necesarios para satisfacer este nuevo requerimiento. Suponga que la clase cuenta con el método esMismoDia(Date dia1, Date dia2) que devuelve true en caso de que los días sean iguales y false en caso de no serlo.

Ejercicio 3.

Teniendo en cuenta solamente los métodos especificados en el diagrama de clases del ejercicio 1 y el siguiente código.

```
1  for (Iterator<Turno> iterator=turnos.iterator();iterator.hasNext();){
2                  Turno turno = iterator.next();
3
4                  System.out.print(turno.getDia());
5                  System.out.print(turno.getHora());
6                  System.out.print(turno.getNroConsultorio() );
7                  System.out.println(turno.getPaciente())
8
9  }
```

Indique qué sucedería si se realizan los siguientes cambios a la clase Consultorio? Considerar cada cambio de forma independiente de los otros.

Para cada cambio, considerar sólo tres posibilidades:

a. "Nada": Si no ocurrirá ningún cambio en el programa.
b. "Error". Si produciría que el programa no compile o que de un error durante su ejecución.
c. "Salida". Si cambiaría la salida de la ejecución.

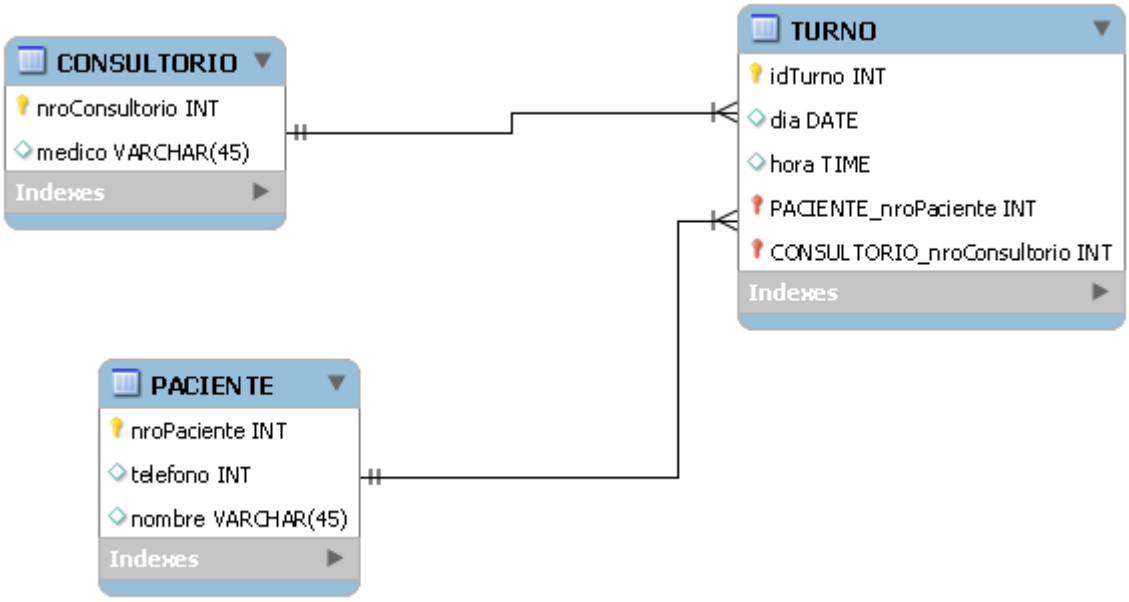
Eliminar la línea 8	
Cambiar la línea 2 por Turnos turnos = iterador.next()	
Reemplazar la línea 6 por int consultorio; consultorio = turno.getNroConsultorio(); System.out.print(consultorio);	
Agregar a continuación de la línea 5 System.out.print(turno.getDatos());	
Agregar a continuación de la línea 5 System.out.print(turno.getDia());	

Ejercicio 4. Javadoc

Elaborar la documentación técnica utilizando Javadoc del método *agregarTurno* implementado en el ejercicio 1. Incluya tanto la descripción del método como los tags que correspondan.

Ejercicio 5. Consulta SQL

Se ha modelado el siguiente diagrama de Entidad Relación.



Se nos pide que definamos una consulta SQL que liste los turnos del consultorio 5 ordenado por dia. Se definió la siguiente consulta pero no hace lo que realmente se pide. Modifique la consulta para que haga lo que se pide.

```
SELECT t.* FROM Turno t
ORDER BY t.dia
```

Ejercicio 6. Hibernate

Dado el diagrama de entidad-relación presentado en el ejercicio anterior y el diagrama UML presentado en el ejercicio 1, escriba la línea del archivo de mapeo de Hibernate correspondiente al mapeo del atributo “nombre” de la clase/tabla Paciente.

SOLUCION

Ejercicio 1

```
public class Consultorio{
        private int nroConsultorio;
        private String medico;
        private List<Turno> turnos;

        public Consultorio(int nroConsultorio, String medico){
                this.nroConsultorio=nroConsultorio;
                this.medico=medico;
                turnos=new ArrayList<Turno>();
        }
```

```
        public void agregarTurno(Date dia, Time hora, int nroPaciente){
                turnos.add(new Turno(dia, hora, nroConsultorio,nroPaciente));
        }
}
```

Ejercicio 2. Implementar un método a partir de un requerimiento

A implementar en la clase Consultorio:

```
public void cancelarTurnos(Date dia){
        for (Iterator<Turno> iterator=turnos.iterator();iterator.hasNext();){
                Turno turno = iterator.next();
                if(esMismoDia(dia, turno.getDia()))
                        turnos.remove(turno);
        }
}
```

Ejercicio 3.

Eliminar la linea 8	nada
Cambiar la linea 2 por Turnos turnos = iterador.next()	error
Reemplazar la linea 6 por int consultorio; consultorio = turno.getNroConsultorio(); System.out.print(consultorio);	nada
Agregar a continuación de la linea 5 System.out.print(turno.getDatos());	error
Agregar a continuación de la linea 5 System.out.print(turno.getDia());	salida

Ejercicio 4. Javadoc

```
/**
* Crea y agrega  un turno a la lista de turnos *
* @param dia dia del turno
* @param hora horario del turno
* @param nroPaciente nro del paciente que solicito el turno
*/
```

Ejercicio 5. Consulta SQL

```sql
SELECT t.* FROM Turno t
WHERE t.nroConsultorio=5
ORDER BY t.dia
```

Ejercicio 6. Hibernate

```
<property name = "nombre" column = "nombre" type = "java.lang.String"/>
```