

**Universidad Tecnológica Nacional  
Facultad Regional Avellaneda**



Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos

**Materia: Laboratorio de Programación II**

Apellido:		Fecha:	27/07/2017
Nombre:		Docente <sup>(2)</sup> :	
División:		Nota <sup>(2)</sup> :	
Legajo:		Firma <sup>(2)</sup> :	
Instancia <sup>(1)</sup> :	<input checked="" type="checkbox"/> PP <input type="checkbox"/> RPP <input type="checkbox"/> SP <input type="checkbox"/> RSP <input type="checkbox"/> FIN <input checked="" type="checkbox"/> X		


(1) Las instancias validas son: 1<sup>er</sup> Parcial (PP), Recuperatorio 1<sup>er</sup> Parcial (RPP), 2<sup>do</sup> Parcial (SP), Recuperatorio 2<sup>do</sup> Parcial (RSP), Final (FIN). Marque con una cruz.

(2) Campos a ser completados por el docente.

Colocar sus datos personales en el nombre del proyecto principal, colocando: Apellido.Nombre.AñoCursada.  
Ej: Pérez.Juan.2016. **No sé corregirán proyectos que no sea identificable su autor.**

TODAS las clases e interfaces deberán ir en una Biblioteca de Clases llamada *Entidades*.

**No se corregirán exámenes que no compilen.**

Al finalizar, colocar la carpeta de la Solución completa en un archivo ZIP que deberá tener como nombre Apellido.Nombre.AñoCursada.zip (los exámenes no identificables no serán corregidos) y dejar este último en el Escritorio de la máquina. Luego presionar el botón  de la barra superior, colocar un mensaje y apretar **Aceptar**. Luego retirarse del aula y aguardar por la corrección.

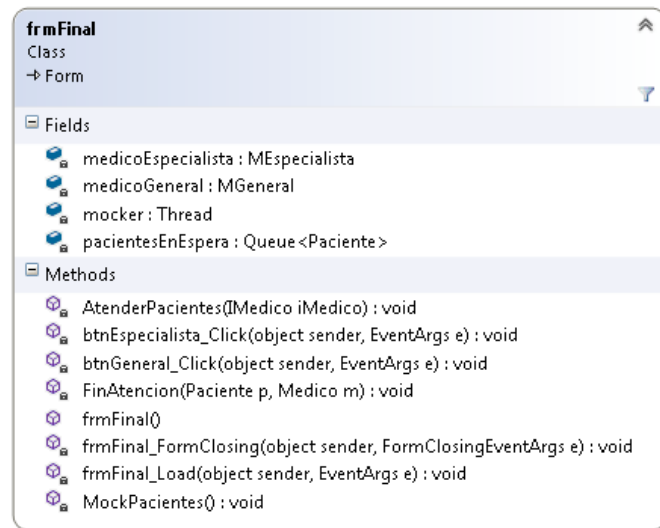
Modelar un sistema de atención para un Sanatorio.

Diseñar el siguiente formulario:

Final 27-07-2017

Atender Médico General

Atender Médico Especialista

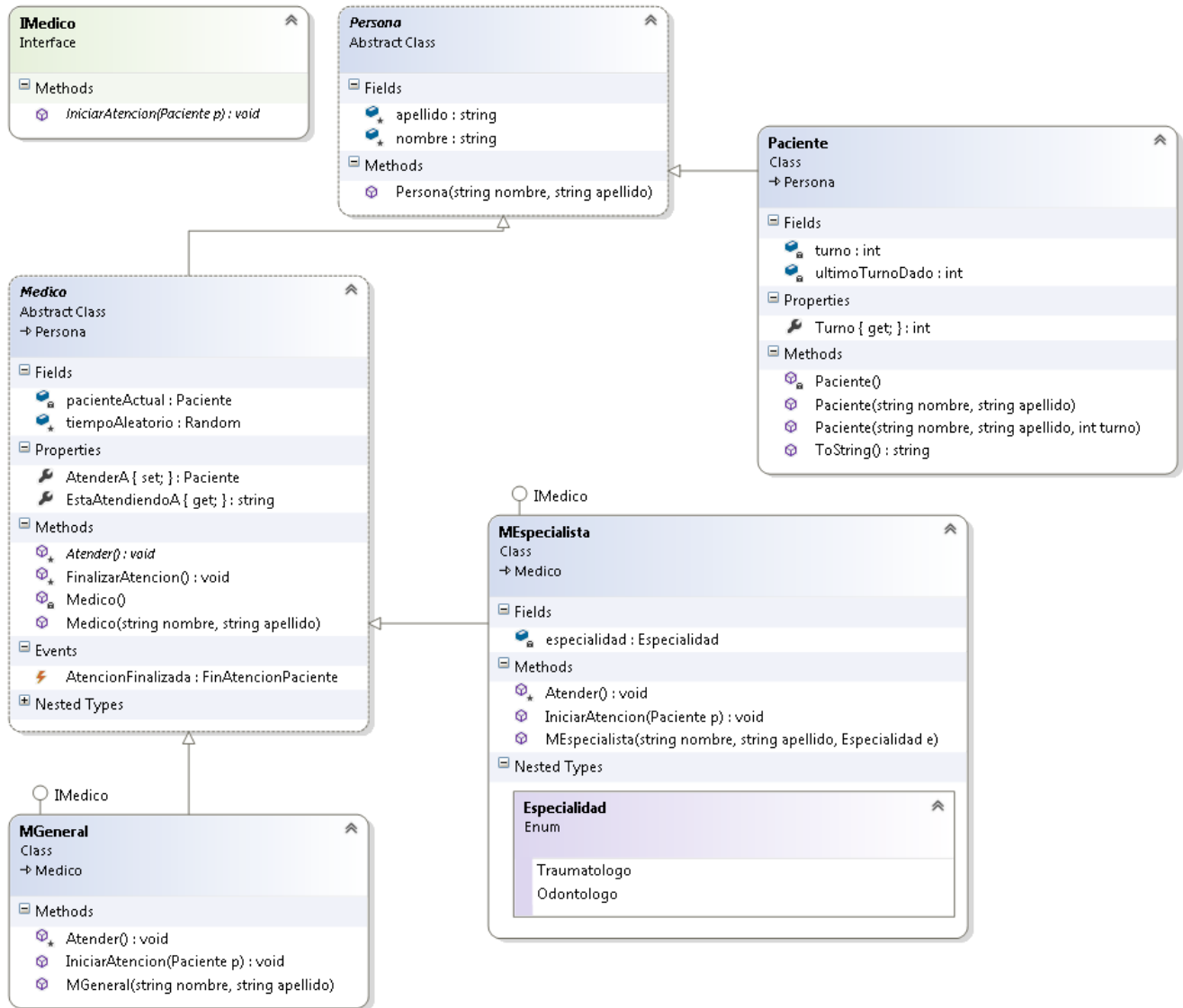


Siendo:

- `frmFinal()` el constructor del formulario, dónde se instanciarán los atributos del formulario, siendo `medicoEspecialista` y `medicoGeneral`:  

```
this.medicoGeneral = new MGeneral("Luis", "Salinas");
this.medicoEspecialista = new MEspecialista("Jorge", "Iglesias",
MEspecialista.Especialidad.Traumatologo);
```
- `frmFinal_Load()` el evento de carga del formulario, dónde inicializaremos el hilo `mocker`.
- `frmFinal_FormClosing()` el evento de cierre del formulario, dónde, si el hilo `mocker` aun está activo, se abortará.
- `MockPacientes()` dónde se agreguen pacientes a la cola `pacientesEnEspera`, haciendo un `Sleep` de 5000 (`Thread.Sleep(5000)`).
- `AtenderPacientes(IMedico)` será invocado por los eventos click de los botones (`btnEspecialista_Click` y `btnGeneral_Click`) pasandole el médico que corresponda (`medicoEspecialista` o `medicoGeneral`, respectivamente). En el caso de haber pacientes en espera, se deberá iniciar la atención del primer elemento de la cola.
- `FinAtencion(Paciente, Medico)` mostrará por medio de un `MessageBox` un mensaje con el formato `"Finalizó la atención de {0} por {1}!"`, dónde se indicará el nombre del paciente y el del médico que lo atendió, respectivamente.

Con el siguiente esquema de clases:



### Paciente:

- `ultimoTurnoDado` será de clase y privado. Se inicializará en el constructor, también de clase, con el valor 0.
- El constructor `Paciente(string, string, int)` asignará los valores a cada atributo, modificando también `ultimoTurnoDado` por el valor recibido.
- El constructor `Paciente(string, string)` incrementará el valor de `ultimoTurnoDado` en 1 y se lo asignará al turno.
- `ToString()` retornará los datos del paciente con el siguiente formato **"Turno N°{0}: {2}, {1}"**, siendo los valores número de turno, apellido y nombre respectivamente.

### Médico:

- El constructor de clase instanciará a `tiempoAleatorio`. El atributo tendrá visibilidad de protegido.
- La propiedad `EstaAtendiendoA` será de sólo lectura y virtual, retornando los datos del `pacienteActual`.
- La propiedad `AtenderA` será de sólo escritura, asignando el valor al atributo `pacienteActual`.
- `Atender` será protegido y abstracto.
- El método `FinalizarAtencion` lanzará el evento `AtencionFinalizada` y luego asignará **null** al paciente actual.

### MGeneral y MEspecialista:

- El método `IniciarAtencion` será el encargado de crear y lanzar un hilo dónde se ejecutará el método `Atender`.
- El método `Atender` hará un `Sleep` de un tiempo aleatorio (de entre 5000 y 10000 para `MEspecialista` y de entre 1500 y 2200 para `MGeneral`). Luego avisará que finalizó la atención.

### Test Unitario:

Realizar un Test Unitario de nombre ConstructoresPaciente que pruebe los constructores de paciente. Al ejecutar los siguientes constructores en el orden propuesto, deberán cumplirse los siguientes requerimientos:

- Si hacemos `new Paciente("Nombre", "Apellido");` el valor del Turno deberá ser 1.
- Si hacemos `new Paciente("Nombre", "Apellido", 5);` el valor del Turno deberá ser 5.
- Si hacemos `new Paciente("Nombre", "Apellido");` el valor del Turno deberá ser 6.