Universidad Tecnológica Nacional Facultad Regional Avellaneda										
Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos										
Materia: Laboratorio de Programación II										
Apellido:			Fecha:		06/07/2021					
Nombre:			Docente(2):							
División:	2ºD		Nota(2):							
Legajo:			Firma(2):							
Instancia(1):	**PP**		**RPP**		**SP**	X	**RSP**		**FIN**	

(1) Las instancias validas son: 1er Parcial (**PP**), Recuperatorio 1er Parcial (**RPP**), 2do Parcial (**SP**), Recuperatorio 2do Parcial (**RSP**), Final (**FIN**). Marque con una cruz.
(2) Campos a ser completados por el docente.

IMPORTANTE:

- ***2 (dos) errores en el mismo tema anulan su puntaje.***
- **La correcta documentación y reglas de estilo de la cátedra serán evaluadas.**
- Colocar sus datos personales en el nombre del proyecto principal, colocando: Apellido.Nombre.Division.
- Ej: Pérez.Juan.2D. No sé corregirán proyectos que no sea identificable su autor.
- **TODAS** las clases deberán ir en una Biblioteca de Clases llamada Entidades, a no ser que se indique explícitamente otra cosa.
- No se corregirán exámenes que no compilen.
- **Reutilizar** tanto código como crean necesario.
- Colocar nombre de la clase (en estáticos), **this** o **base** en todos los casos que corresponda.

---

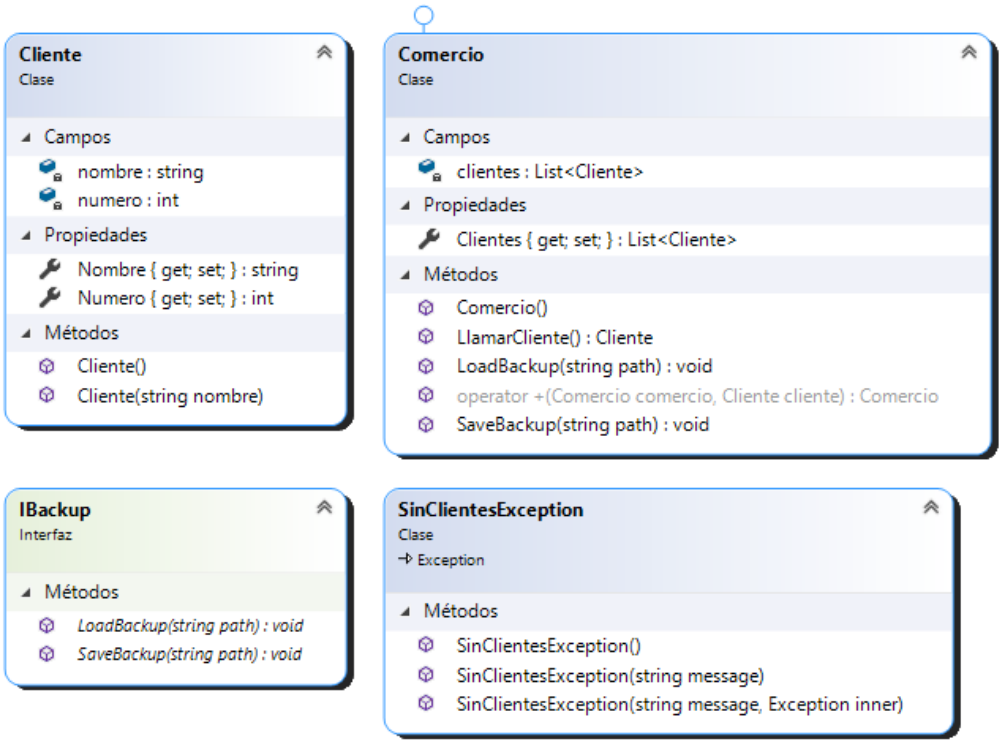
*TIEMPO MÁXIMO PARA RESOLVER EL EXAMEN **90** MINUTOS.*

---

1. Partir de la solución entregada. Modificar su nombre como se indicó anteriormente en este documento.
2. Base de datos

```
USE [master]
GO
CREATE DATABASE [20210706-SP]
GO
USE [20210706-SP]
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[atenciones](
	[atendidos] [int] NOT NULL,
	[noAtendidos] [int] NOT NULL,
	[alumno] [varchar](50) NOT NULL
) ON [PRIMARY]
GO
```

Esquema de Clases

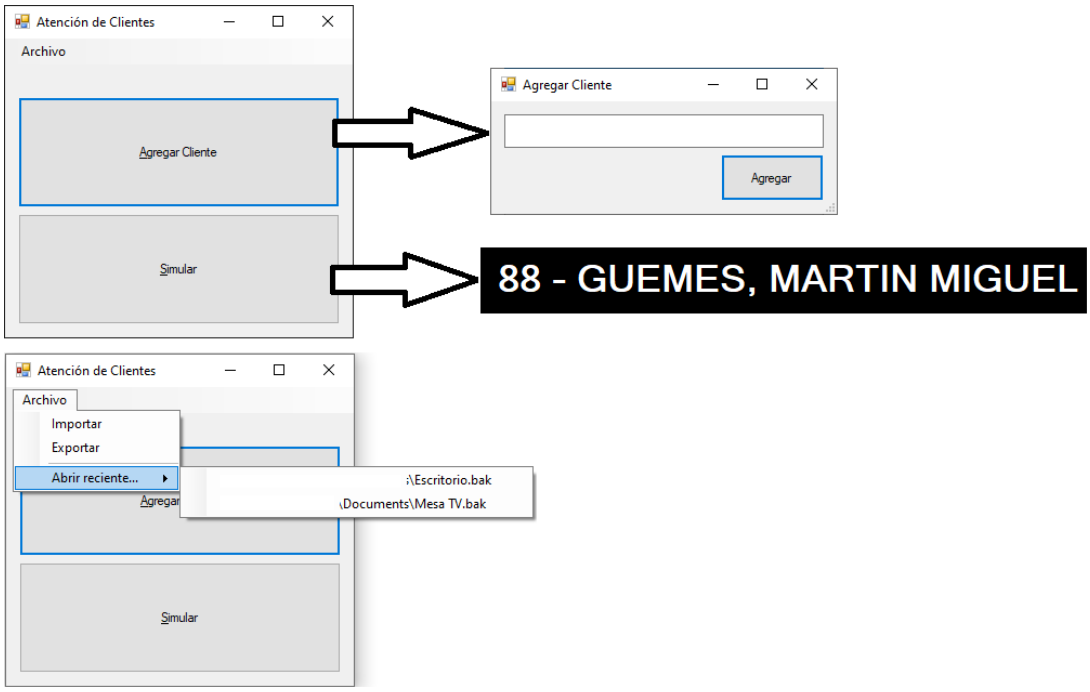


3. No agregar ninguna clase que no sea explícitamente requerida.

Comercio

4. Dentro de la clase Comercio, la lista de clientes será instanciada en el constructor.
5. El operator + le asignará el próximo número al cliente (el número que tenga el último + 1) y luego lo agregará al comercio. En caso de ser el primer cliente, su número se asignará en este mismo operador como 1.
6. LlamarCliente lanzará la excepción SinClientesExcepcion en caso de no tener ningún cliente en la lista.
7. Si tiene clientes, LlamarCliente deberá:
    a. Tomar el PRIMER cliente de la lista.
    b. Ese cliente será su retorno.
    c. Eliminará dicho cliente de la lista.
8. LoadBackup deberá ser capaz de recuperar un archivo XML con una lista de clientes.
9. SaveBackup deberá ser capaz de guardar un archivo XML con una lista de clientes.

Esquema de Formularios



10. El botón Agregar Cliente abrirá el formulario correspondiente:
    a. Si este formulario responde OK (DialogResult.OK), se agregará el cliente a la lista del comercio.
    b. Caso contrario, no se hará nada.
11. Simular disparará el formulario correspondiente:
    a. Se agregará una clase llamada Simulador al proyecto Entidades.
    b. Dicha clase deberá implementar Eventos, Delegados e Hilos debiendo:
        i. Mostrar cada número – nombre de la lista de clientes del comercio.
        ii. Cada nombre, incluso el último, deberá verse por al menos 2 segundos en pantalla. Utilizar Sleep.
    c. El formulario se puede cerrar presionando la tecla ESC. Controlar que el hilo sea cerrado con él.
    d. Al volver al formulario principal se deberá guardar en la base de datos cuantas atenciones fueron correctas y cuantas no respondieron al llamado.
        i. Para dicha estadística se utilizarán acumuladores y un Random que diga si fue o no atendido el llamado.
12. El menú Archivo / Importar utilizará el OpenFileDialog ya creado para ayudar a buscar y seleccionar que lista de clientes se quiere leer. Si se seleccionase un archivo válido, se deberá incorporar dicha lista al Comercio existente.
13. El menú Archivo / Exportar utilizará el SaveFileDialog ya creado para ayudar a buscar un lugar para guardar un backup de la lista actual de clientes del comercio.

PUNTO EXTRA (15 al 17):

14. Cada vez que importemos un archivo válido, se deberá:
    a. Guardar la ruta en un archivo txt ubicado en la ruta de nuestro ejecutable.
    b. Dichas rutas deben estar ordenadas del más reciente al más viejo.
    c. No se deben almacenar más de 10 rutas válidas.
15. Al elegir una de las rutas de Abrir Recientes se deberá preguntar si está seguro, y en caso afirmativo importar dicho archivo, siempre y cuando exista.
16. Este submenú se deberá cargar en el inicio. AYUDA:

```
ToolStripItem aux = new ToolStripMenuItem();
aux.Text = streamReader.ReadLine();
abrirRecienteToolStripMenuItem.DropDownItems.Add(aux);
```

Test Unitarios

17. Realizar un test unitario que pruebe la excepción SinClientesException.

18. Realizar un test unitario que pruebe que funcione el Save y Load Backup comprobando que lo guardado sea lo mismo que lo leído.
19. Realizar un test unitario libre, comprobando alguna funcionalidad útil del sistema. Se juzgará lo útil del método.

Al finalizar, colocar la carpeta de la carpeta de la Solución completa en un archivo ZIP que deberá tener como nombre Apellido.Nombre.division.zip y compartir este por Slack sólo con el docente titular de la cursada.