Universidad Tecnológica Nacional Facultad Regional Avellaneda											
Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos											
Materia: Laboratorio de Programación II											
Apellido:				Fecha:	02/12/2021						
Nombre:				Docente(2):							
División:	2ºC			Nota(2):							
Legajo:				Firma(2):							
Instancia(1):	**PP**		**RPP**		**SP**		**RSP**	X	**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.
- El proyecto debe ser creado en .Net 5.
- Colocar sus datos personales en el nombre de la carpeta principal y la solución: Apellido.Nombre.Div. Ej: Pérez.Juan.2D. No sé corregirán proyectos que no sea identificable su autor.
- 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.
- Aplicar los principios de los 4 pilares de la POO.

---

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

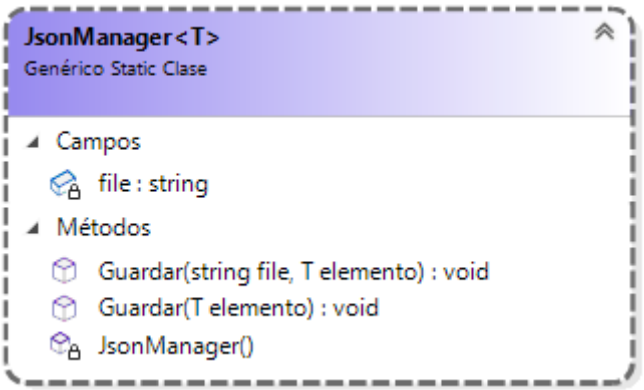
---

1. Partir de la solución entregada. Modificar su nombre con el siguiente formato: [APELLIDO].[NOMBRE].
2. Implementar la BD desde el script enviado.


QUERY_PARCIAL.sql

Files

3. Dentro del proyecto se deberá respetar el siguiente esquema:



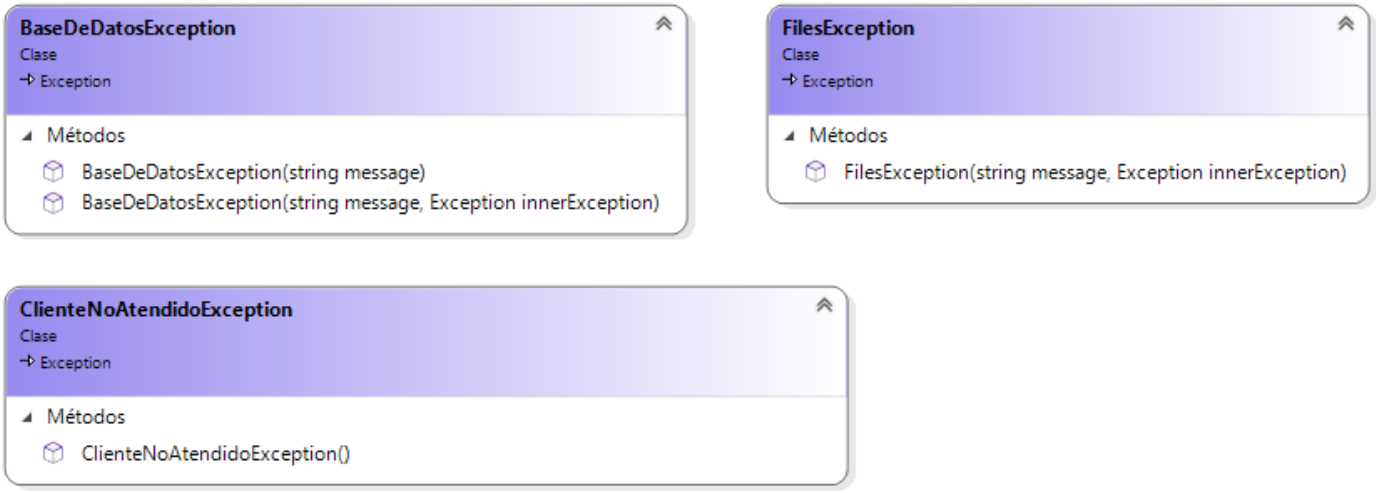
4. El constructor de la clase JsonManager estática asignará al atributo *file* el valor CentroDeAtencion.json.
5. Guardar(T elemento) guardará el elemento recibido como parámetro en un archivo de texto:
    a. Su contenido será el paramentro "elemento" un objeto y deberá ser convertido a JSON.

b. Dicho archivo será guardado en donde indique el atributo file.
c. De haber un error, se lanzará FilesException.

6. Guardar(string file, T elemento) modificará el atributo *file* y luego hará lo mismo que Guardar(T).

Excepciones

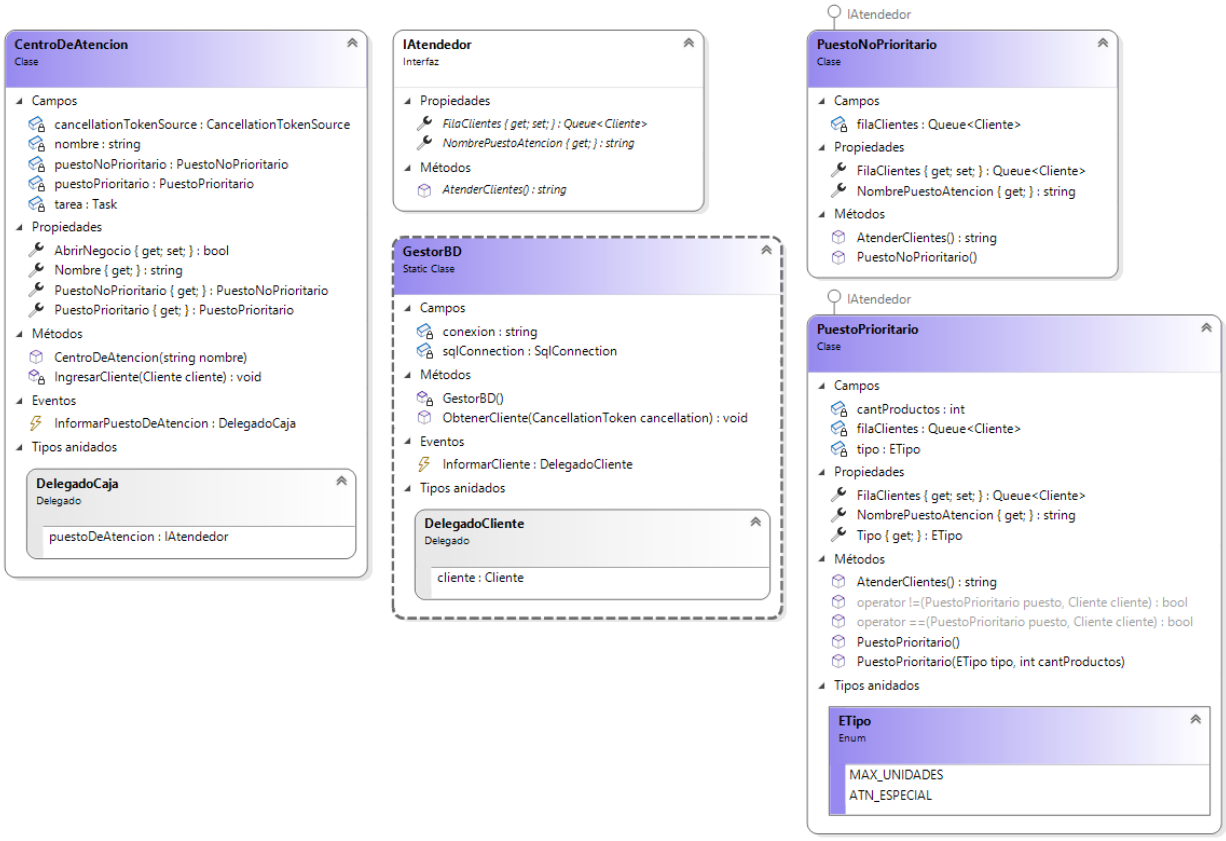
7. Dentro del proyecto respetar el siguiente esquema:



8. ClienteNoAtendidoException: su mensaje por defecto será Cliente no prioritario.

Entidades

9. Dentro del proyecto se deberá respetar el siguiente esquema:



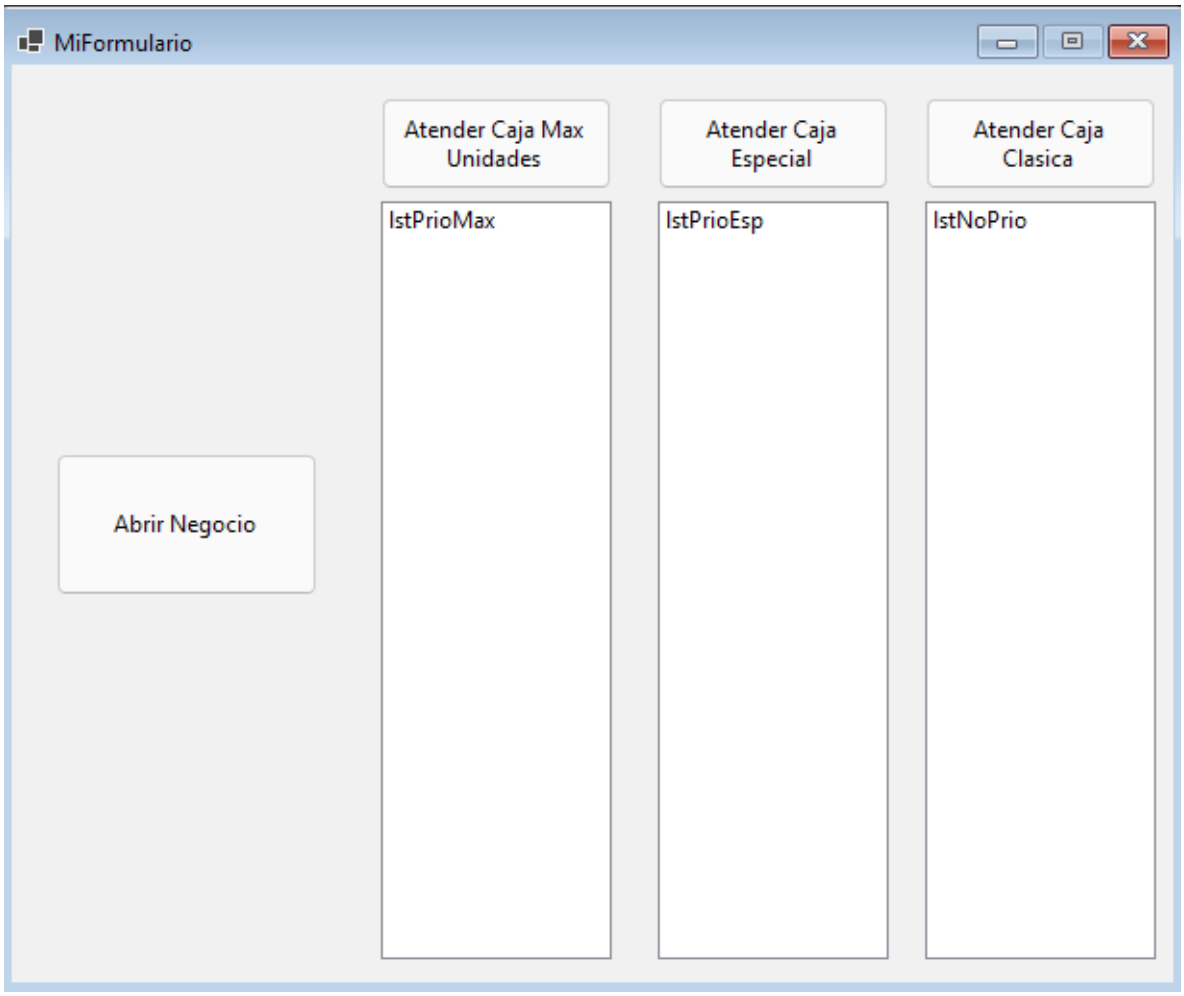
10. PuestoPrioritario:

a. Implementa IAtendedor.
b. Por defecto un puesto prioritario será de MAX_UNIDADES con 15 propductos.
c. La propiedad NombrePuestoAtencion retornara "Puesto de atencion Prioritario de tipo [tipo]"
d. La sobrecarga del operador == comparara un PuestoPrioritario con un cliente para determinar si puede ser atendido por dicho puesto. Se retornará True si puesto es de tipo ANT_ESPECIAL y el cliente requiere prioridad, o si el tipo es MAX_UNIDADES y la cantidad de productos del cliente es menor a los permitidos en el puesto de atención.
e. AtenderClientes verificara si hay clientes en espera en la fila, en caso de no haber clientes retornar Sin Clientes.De lo contrario retira el primer cliente de la Queue y verifica si este puede ser atendido de forma prioritaria. En caso afirmativo retornar Cliente Atendido en… Concatenando el nombre del puesto de atención. En caso negativo se lanzará una excepción ClienteNoAtendidoException.

11. PuestoNoPrioritario:
    a. Implementa IAtendedor.
    b. La propiedad NombrePuestoAtencion retornara "Puesto de atención No Prioritario"
    c. AtenderClientes verificara si hay clientes en espera en la fila, en caso de no haber clientes retornar Sin Clientes. De lo contrario retira el primer cliente de la Queue y retornar Cliente Atendido en… Concatenando el nombre del puesto de atención.
12. GestorBD:
    a. Sera estática.
    b. ObtenerCliente correrá en un hilo secundario, este leerá de la base de datos todos los clientes. Mientras no sea el fin de los registros o no se haya cancelado el hilo se informará el cliente leído a través del evento.
    c. Realizar un Sleep de 2 segundos.
    d. En caso de producirse un error encapsular la excepción y lanzar BaseDeDatosException.
13. CentroDeAtencion:
    a. Su único constructor instanciará una colección de puestos de atencion y será el encargado de agregar el manejador IngresarCliente al evento InformarCliente de GestorBD.
    b. AbrirNegocio será la propiedad encargada de poner a leer los registros de la BD en un hilo secundario. Implementar lo necesario de modo que se puede cerrar y abrir el negocio.
    c. IngresarCliente verificara si el cliente requiere prioridad, y lo encolara en el puesto de atención que corresponda. Luego de encolarlo informar el puesto de atención a través del evento.

Formularios

14. El resultado deberá ser:



a. Botón Abrir Negocio: permitirá el ingreso de clientes al centro de atención. Una vez abierto deberá de brindar la posibilidad de cerrarlo. Al cerrarlo se deberá guardar en Json el centro de atención.
b. Botón Atender Caja: atenderá los clientes de las distintas cajas. En caso que no se pueda atender al cliente informar mediante un mensaje.
c. Al cerrar el formulario se deberá cerrar el negocio.
d. En el constructor del formulario, asociar el evento a su manejador.

Test Unitarios

15. Darle un nombre claro al proyecto, sus clases y sus métodos
16. Agregar 2 test unitarios:
    a. Forzar, mediante el código la ejecución de ClienteNoAtendidoException, validar que suceda de forma correcta.
    b. Para mayor satisfacción de nuestros clientes, agregar al proyecto Files la posibilidad de leer el centro de atención serializado. Validar que suceda de forma correcta.

Punto Extra

17. *Agregar un boton adicional al FRM que al presionarlo, en un hilo secundario retire los clientes de la Queue de manera asincrónica*

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.