Universidad Tecnológica Nacional Facultad Regional Avellaneda										
Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos										
Materia: Laboratorio de Programación II										
Apellido:				Fecha:	11/08/2022					
Nombre:				Docente(2):						
División:	2ºE			Nota(2):						
Legajo:				Firma(2):						
Instancia(1):	**PP**		**RPP**		**SP**		**RSP**		**FIN**	X

(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:

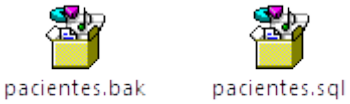
- **2 (dos) errores en el mismo tema anulan su puntaje.**
- La correcta documentación y reglas de estilo de la cátedra serán evaluadas.
- El proyecto debe ser creado en .Net 5.
- Colocar sus datos personales en el nombre de la carpeta principal y la solución: Apellido.Nombre.Div. Ej: Pérez.Juan.2D. No 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.*

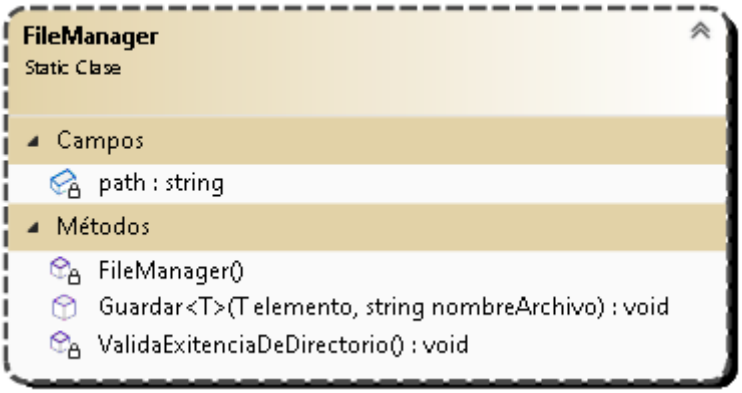
---

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



Files

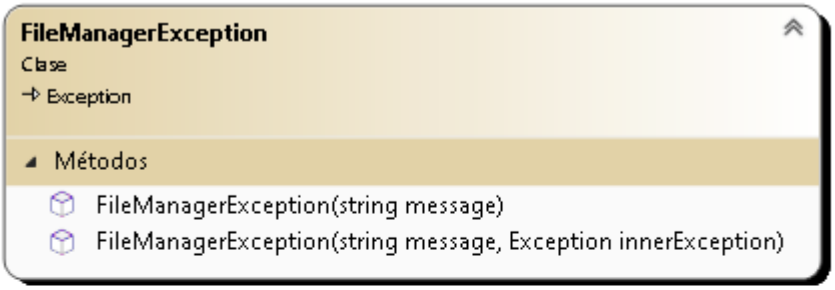
3. Dentro del proyecto se deberá respetar el siguiente esquema:

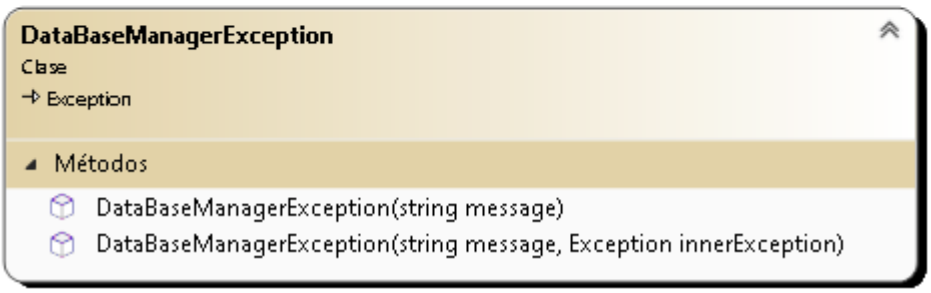


4. FileManager será estática.
    a. En el constructor de clase realizar:
        i. En el atributo path se almacenará la referencia al escritorio de la pc. Y se le concatenara un el nombre de la carpeta del parcial: ej {path escritorio}+`\\20220804_Alumno\\`
        ii. Llamar al método ValidaExistenciaDeDirectorio.
    b. ValidaExistenciaDeDirectorio:
        i. Si no existe el directorio almacenado en path, se creará.
        ii. En caso de producirse una excepción al momento de la creación, esta deberá ser capturada y relanzada en una nueva excepción denominada FileManagerException, la cual contendrá el mensaje: “Error al crear el directorio”.
    c. Guardar:
        i. Será genérico y solo permitirá que los elementos a almacenar sean tipos por referencia.
        ii. Validar la extensión del nombre del archivo. En caso de que sea:
            1. JSON, se serializará el elemento recibido.
            2. TXT, se almacena en texto plano.
            3. Cualquier otra extensión se lanzará una excepción denominada FileManagerException, la cual contendrá el mensaje “Extensión no permitida”.

Excepciones

5. Dentro del proyecto respetar el siguiente esquema:

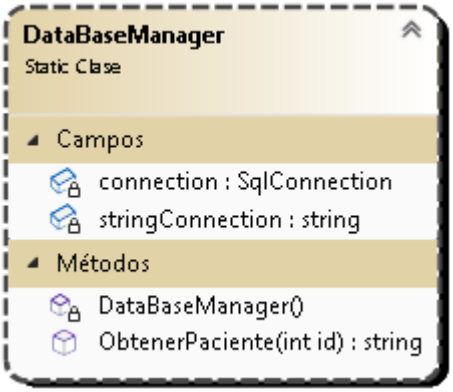




6. Controlar las posibles excepciones producidas e informar al usuario del error.

Bases de datos

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



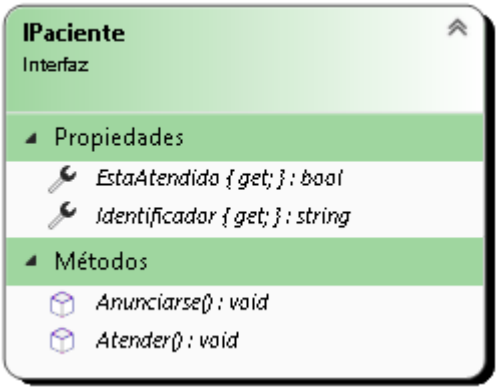
8. DataBaseManager será estática:
    a. En el constructor de clase inicializar el string connection.
    b. ObtenerPaciente, recibirá el id del paciente a obtener. Retornara un string que contendrá el nombre y apellido concatenado con un guion medio, Ej: `$"{reader.GetString(2)}-{reader.GetString(1)}"`.
    c. Si el id no existe se lanzará una excepción DataBaseManagerException, indicando Id inexistente.
    d. Por cualquier otro error se capturara y se re lanzara en una excepción DataBaseManagerException indicando error al leer de la base de datos.

Métodos de extensión

9. Extenderá la clase String la cual adicionará un método denominado `ObtenerNombreYApellido` en el cual su función principal será devolver un array de strings, mediante la separación de la cadena en subcadenas. El carácter para determinar la separación será el guion medio.

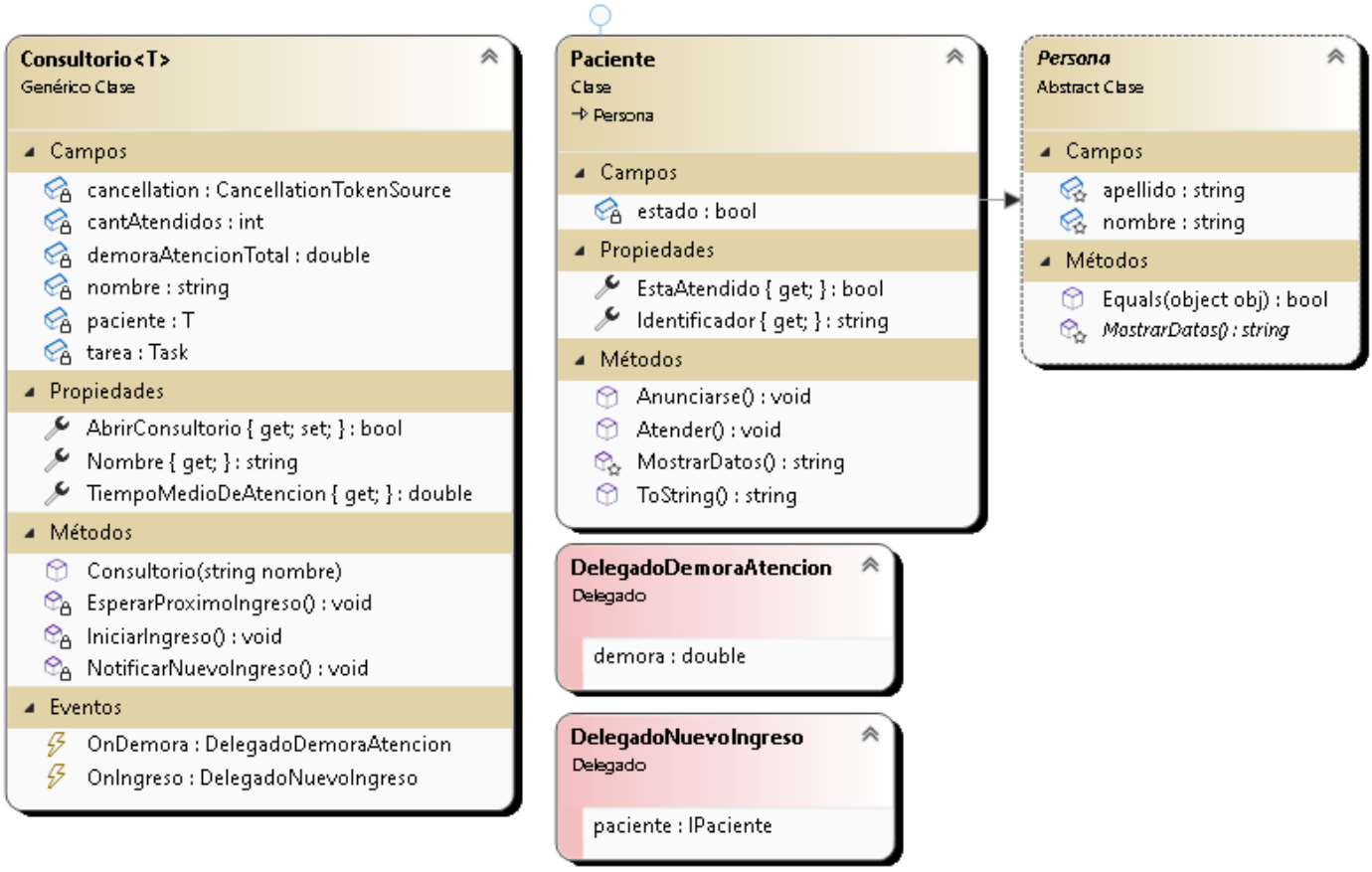
Interfaces

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



Entidades

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



12. **Persona**, será abstracta.
    a. Sus dos atributos serán privados.
    b. MostrarDatos protegido y abstracto.
    c. Sobre escribir Equals y comparar las personas por apellido.
13. **Paciente**, heredará de persona e implementará la interfaz IPaciente
    a. Atender, cambiara el estado del paciente a True.
    b. La propiedad Identificarse retornara un string con el texto “Paciente” + el HashCode de la instancia + el nombre y el apellido separados por coma.
    c. Anunciarse, leerá desde la BD una paciente de manera aleatoria (entre 1 y 100), separará el nombre y apellido de la cadena y los asignará a los atributos de la instancia.
    d. MostrarDatos retorna el nombre y el apellido separado por coma.
    e. Sobrescribir el toString y exponer MostrarDatos.
14. **Consultorio**, será genérica, solo podrá recibir tipos que implementen la interfaz **IPaciente** y posean un constructor publico sin parámetros:
    a. En su constructor publico inicializara un nombre para el consultorio y ademas:
        i. Inicializar:
            1. demoraAtencionTotal en 0 (cero).
            2. cantAtendidos en 0 (cero).
15. La propiedad AbrirConsultorio:
    a. El GET retornara True, si la tares no es nula y estado de la tarea es Running o WaitingToRun o WaitingForActivation.
    b. En el SET, si el valor recibido es TRUE y la tarea es nula o su estado no es Running o no es WaitingToRun o no es WaitingForActivation, se instanciará un nuevo CancelationTokenSource y se llamará a **IniciarIngreso**. De lo contrario se llamará al método Cancel de cancellation.
16. El método IniciarIngreso será privado y:
    a. Ejecutará en un hilo secundario la acción de que:
        i. Mientras no se requiera cancelación de la tarea invocara a los métodos NotificarNuevoIngreso, EsperarProximoIngreso y luego incrementar en 1 la cantidad de atendidos.

17. El método NotificarNuevoIngreso, verificara si el evento OnIngreso posee suscriptores y en caso exitoso realizara:
    a. Instanciar el paciente.
    b. Anunciará el ingreso del paciente.
    c. Notificar el paciente instanciado
18. El método EsperarProximoIngreso si posee un suscriptor notificara los segundos transcurridos mientras que el paciente no sea atendido o no se requiera cancelacion (Utilizar Thread.Sleep para dormir el hilo 1 segundo antes de ir incrementando)

Formulario

19. Desarrollar todo lo indicado con comentario //Alumno:

Test Unitarios

20. Darle un nombre claro al proyecto, sus clases y sus métodos
21. Agregar 2 test unitarios:
    a. Forzar, mediante el código la ejecución de FileManagerException, validar que suceda de forma correcta.
    b. Al instanciar un nuevo consultorio, se espera que el tiempo medio de atención sea igual a 0 (cero).

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.