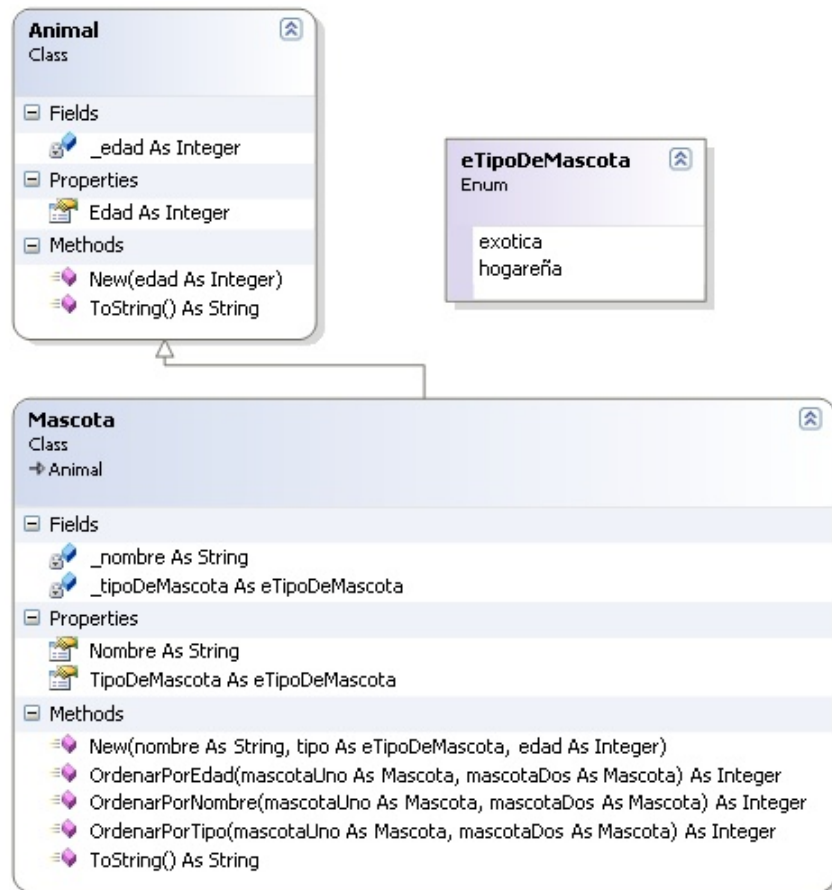


La aplicación deberá poder hacer alta, baja y modificación de la entidad Mascota.

- 1- En un proyecto de tipo **biblioteca de clase** se debe:
 - a. Desarrollar las clases **Animal** y **Mascota** de acuerdo al siguiente diagrama de clases:



Respetar los constructores y métodos mostrados en el diagrama de clases, no se pueden modificar, agregar u obviar a ninguno de los miembros de las clases.

- 2- En un proyecto de tipo **WindowsForm** (iniciará por un módulo que contenga el método **Main**, con su correspondiente control de errores) se crearán tres formularios:

FrmAnimal

Tiene un cuadro de texto para poder ingresar la edad del animal.

El manejador asignado al evento click del botón **Aceptar** debe ser **virtual** para poder ser sobrescrito en los formularios que hereden de este. En dicho manejador se asignará el valor **DialogResult.Ok** a la propiedad **DialogResult** del formulario.

El manejador asignado al evento click del botón **Cancelar** debe asignar el valor **DialogResult.Cancel** a la propiedad **DialogResult** del formulario.

El TextBox de este formulario debe ser **protegido**.

FrmMascota

Hereda de FrmAnimal y posee: un cuadro de texto (**protegido**) para poder ingresar el nombre y un ComboBox (**protegido**) cargado con los tipos de mascotas definidos en el enumerado **eTipoDeMascota**, este control tendrá seleccionado por defecto el valor “exótica” y no se permitirá la escritura dentro del mismo.

Además este formulario tiene un atributo **privado** de tipo **Mascota** que tendrá una propiedad de **sólo lectura**, en el manejador del evento click del botón *Aceptar* se instanciará el atributo Mascota del formulario.

FrmPrincipal

- 3- Este formulario tendrá como atributo **privado** una lista genérica de tipo **Mascota**, un menú con 4 botones y un ComboBox y un control ListBox.

El ComboBox será cargado con los valores del enumerado **eTipoDeOrdenamiento**. Al seleccionar un nuevo criterio, los datos de la lista se ordenarán con un delegado de tipo **Comparison** y se refrescarán los datos del ListBox.

- 4- **Botón de alta:**
Mostrará una instancia del formulario **FrmMascota** y agregará el objeto de tipo Mascota de dicho formulario a la lista de **FrmPrincipal**. Los objetos que se agreguen serán mostrados en el ListBox.
- 5- **Manejadores:**
Los manejadores de los eventos click de los botones de modificación y baja, sólo serán agregados al momento de seleccionar un elemento del **ListBox** y removidos al hacer click en cualquiera de los dos botones. Así mismo el manejador del evento **selectedIndexChanged** será quitado al seleccionar un elemento del ListBox y será agregado al clickear alguno de los dos botones.

Nota: No se deben producir excepciones al manipular dichos manejadores.

6- **Botón de baja:**

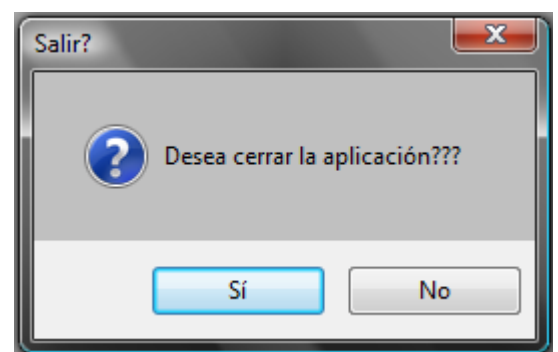
El manejador asociado al evento click se llamará “**ManejadorCentral**” e instanciará un objeto de tipo **FrmMascota** pasándole en el constructor del formulario el objeto de tipo **Mascota** seleccionado del ListBox. Los datos de este objeto serán mostrados en los controles del formulario instanciado y si el usuario pulsa **Aceptar** el objeto será removido de la lista genérica de Mascotas de **FrmPrincipal**. Luego se refrescará el ListBox.

7- **Botón de modificación:**

El evento click del botón también tendrá asociado al manejador “**ManejadorCentral**”. Este manejador deberá saber que control lo invocó. Si el botón es el de **Modificación**, tomará al objeto del formulario **FrmMascota** y lo asignará a la lista genérica de tipo **Mascota**, luego refrescará el ListBox.

8- **Botón salir:**

Al hacer click en el botón **Salir** o en la “**X**” del formulario principal, se mostrará un mensaje **IDENTICO** al de la imagen (**NO REPETIR LINEAS DE CODIGO**), y si el usuario selecciona “**Sí**”, se cerrará la aplicación y se guardarán los datos de la lista (**en un archivo de texto**) en el subdirectorío dónde se ejecute la aplicación en ese momento, utilizando las propiedades de la clase **Application**.



9- **Control de Usuario:**

Realizar un control propio en una **biblioteca de clases**, que herede de TextBox y que a través de una propiedad de tipo enumerado nos permita elegir si ese control va a permitir:

- SoloNumeros:** los números del “0” al “9” y la tecla de borrado (BACKSPACE).
- SoloLetras:** las letras de la “a” a la “z” tanto en mayúscula como en minúscula y la tecla de borrado (BACKSPACE).

Cambiar los cuadros de texto de nombre y de edad (en **FrmMascota**), por el control de usuario y configurarlos adecuadamente.

- 10- Se creará un nuevo formulario llamado **FrmMostrar** que posea solamente un ListBox. La instancia de FrmMostar deberá tener la propiedad **FormBorderStyle=“NONE”**. Este formulario tendrá un sólo método público llamado “**ActualizarListados**” que recibirá una lista genérica de tipo “Mascota” como parámetro. Al invocar a este método se actualizará el ListBox (**LstMostrar**) con los nombres de las mascotas de la lista genérica de tipo **Mascota**).

- Crear un tipo de **delegado** para poder invocar al método “**ActualizarListados**” en un módulo público.
- La variable de instancia de este tipo de delegado será creada en el mismo módulo.
- La variable se instanciará en FrmPrincipal, en la opción de menú ‘Mostrar Listado’, apuntando al método de la instancia de FrmMostrar.
- Invocar al delegado al agregar, modificar o eliminar un elemento de la lista de Mascotas.

TIEMPO MÁXIMO PARA RESOLVER EL EXAMEN: 120 MINUTOS.
TRANSCURRIDO ESE TIEMPO LAS MÁQUINAS SERÁN BLOQUEADAS.