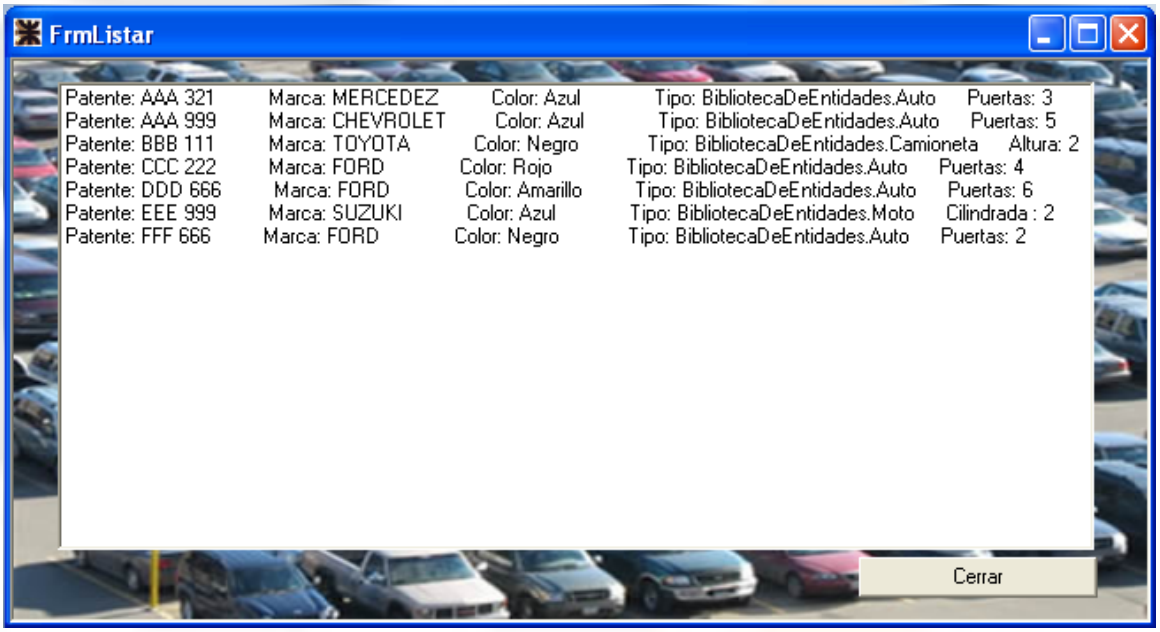
Profesores:

Neiner Maximiliano
Baus Christian

Ejercicio 12 (listado de estacionamiento)

El formulario FrmListar Tiene una lista de vehículos privada con sus propiedades de lectura escritura.
En el evento load de este formulario se invocara a un método protegido llamado "CargarListBox", que se encargara de mostrar los vehículos que están contenidos en la lista de vehículo que tienen este formulario como atributo.



El botón cerrar cerrará el formulario, a este formulario se accederá a través de un botón del menú de "**FrmGestionAutomovil"**

Ejercicio 13 (Baja y modificación con manejadores)

El formulario FrmListarManejadores Tiene una lista de vehículos privada con sus propiedades de lectura escritura.
Partiendo del formulario del punto anterior realizar el siguiente diseño colocándole los botones para Modificar Y Borrar.

El diseño del formulario es el siguiente:

Manejadores:

Al evento "**SelectedIndexChanged"** del ListBox se le agregará en forma dinámica el manejador solo si la lista tiene vehículos cargados.
Los botones de borrar y modificar no tendrán manejador asociado hasta que no seleccionemos un elemento de la lista. Esta operación se hará en el evento "**SelectedIndexChanged"** de la lista .
Al momento de seleccinar un item de la lista se deberá remover el manejador del listbox en forma dinamica.
Al momento de hacer click en alguno de los botones (**Modificar o Borrar**) se le quitaran los manejadores a los botones y se le colocaran nuevamente el manajador al "**SelectedIndexChanged"** de la lista .

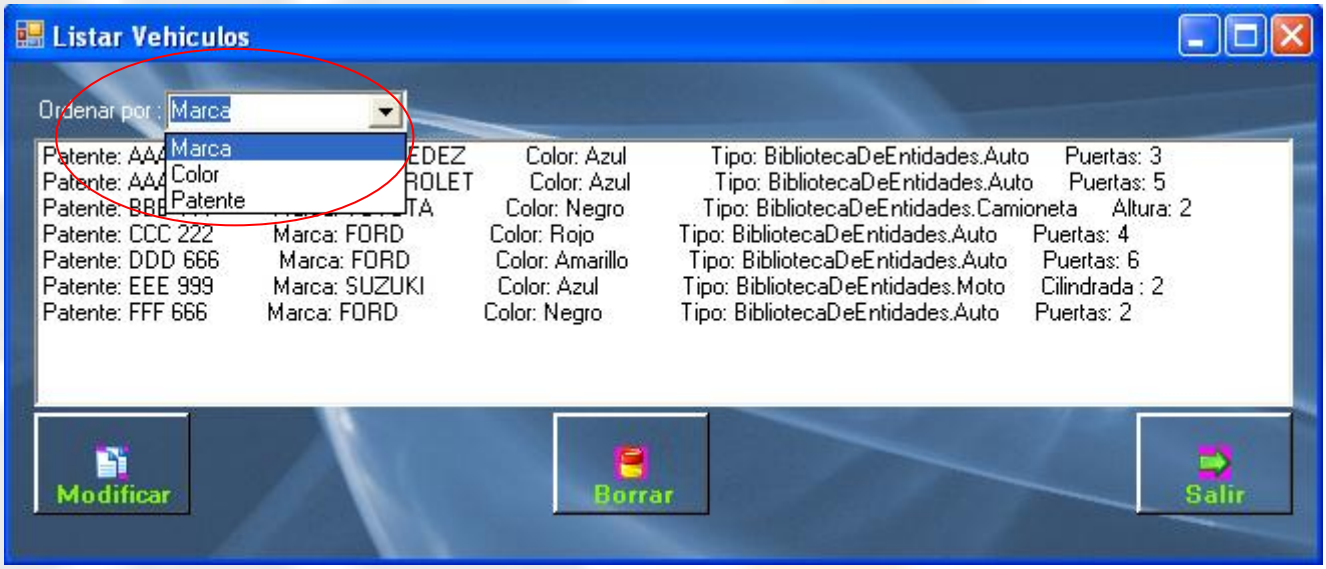
Esto evitara que se genere una excepcion al realizar correctamente la asignación y remoción de manejadores

Botones:

Al hacer click en los botones de borrar o modificar se creará una nueva instancia del formulario que corresponda según el tipo de vehículo seleccionado en el ítem de la lista (FrmAuto, FrmMoto, FrmCamioneta), cargando los datos

del objeto seleccionado en cada uno de los controles del formulario (esta operación se debe realizar en cada formulario).
Estos formularios se mostrarán en forma de dialogo y si su respuesta es un **"DialogResult.Ok"**, se eliminará o modificará los objetos de la lista según corresponda.
Automáticamente la lista se actualizará con los cambios realizados y se refrescará el ListBox.

Ejercicio 14 (Delegado para Ordenamiento)



Implementar el método de **Clase** "OrdenarPorPatente" en la clase vehículo de la siguiente manera:

```
public static int OrdenarPorPatente(Vehiculo vehiculo1,Vehiculo vehiculo2){

        return vehiculo1.Patente.CompareTo(vehiculo2.Patente);

}
```

Este método nos permitirá ordenar la lista de vehículos por patente, realizar los métodos "OrdenarPorColor" y "OrdenarPorMarca" en el modulo de inicio de la aplicación.
Al seleccionar en el ComboBox un nuevo criterio de ordenamiento, se creará un delegado de tipo "Comparison (Of Vehiculo)" con la dirección del método que corresponda, luego se pasará este delegado al método "Sort" de la lista, permitiendo el ordenamiento de la misma por el criterio seleccionado.
Refrescar el listBox.