Universidad Tecnológica Nacional Facultad Regional Avellaneda										
Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos										
Materia: Laboratorio de Programación II										
Apellido:			Fecha:	05-10-2023						
Nombre:			Docente(2):							
División:			Nota(2):							
Legajo:			Firma(2):							
Instancia(1):	**PP**	X	**RPP**		**SP**		**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.

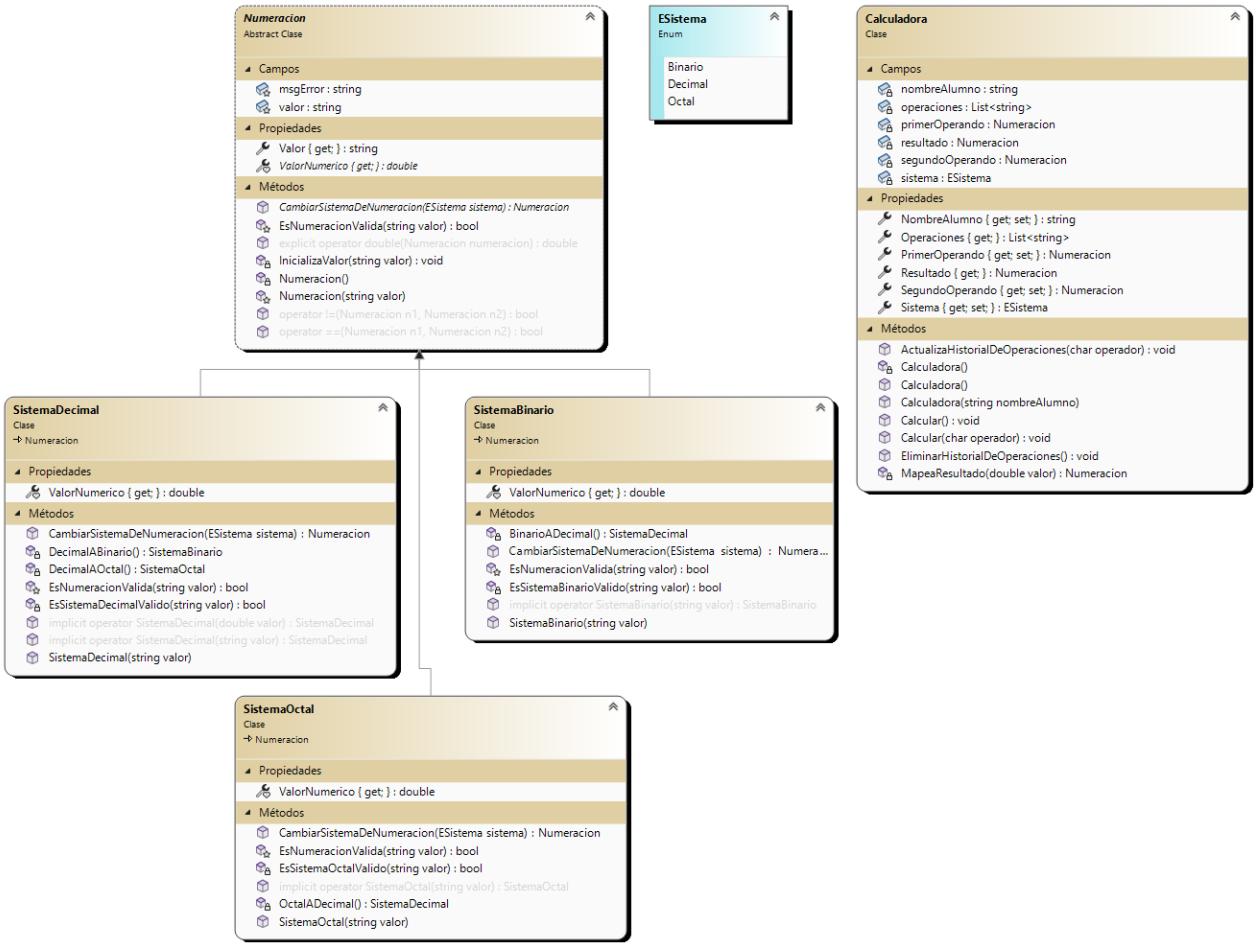
IMPORTANTE:

- **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 de la carpeta principal y la solución: Apellido.Nombre.Div. Ej: Pérez.Juan.2C. 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.

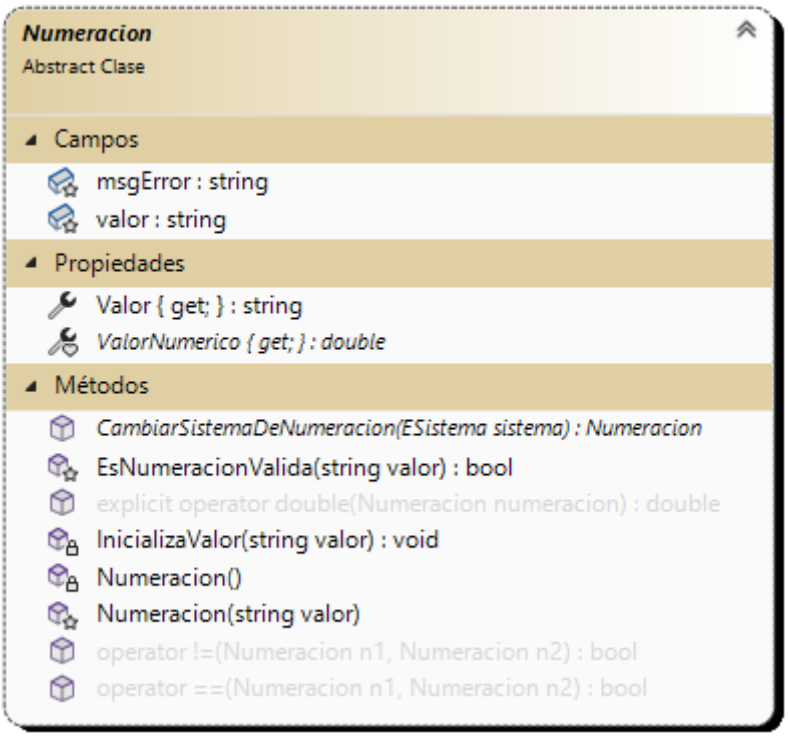
Se desea desarrollar una calculadora que permita operar entre 2 sistemas de numeración.

Para ello se debe:

1. Crear un proyecto de tipo biblioteca de clases y con el siguiente esquema:

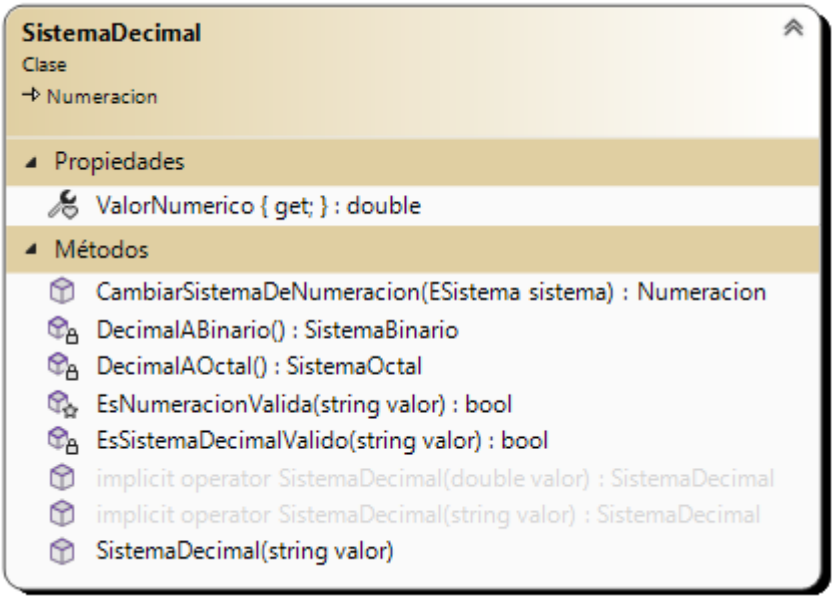


2. Clase Numeracion:



a. Sera abstracta.
b. Sus atributos serán protegidos. *msgError* será un atributo de clase y su valor deberá inicializarse en su constructor también de clase, siendo este `"Numero Invalido"`.
c. El constructor de **Numeración**, que recibe un valor, será protegido, este llamará al método **inicializaValor**, quien será el encargo de darle un estado inicial a dicho atributo.
d. La propiedad **Valor** será de solo lectura.
e. La propiedad **ValorNumerico** será internal y deberá de ser implementada de forma obligatoria en sus clases derivadas.
f. El método **CambiarSistemaDeNumeracion** será público y deberá de ser implementado de forma obligatoria en sus clases derivadas.
g. El método **EsNumeracionValida**, será protegido. En la clase base solo verificara que la cadena recibida no sea nula o con espacios en blanco.
h. El método **InicializaValor**, será privado y su función será darle un estado inicial al atributo valor de la instancia, para ello validará que el valor recibido sea una numeración valida, de lo contrario el atributo almacenará un mensaje de error.
i. Dos numeraciones serán iguales si no son nulas y son del mismo tipo.
j. Implementar una conversión explicita de **Numeración** a **double**. Esta devolverá el *valor* de la **Numeración**.

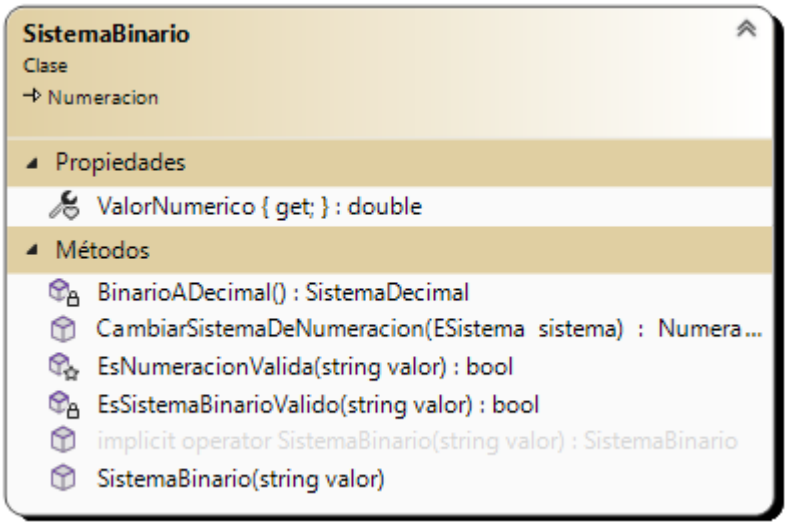
3. **Clase SistemaDecimal:**



a. Heredara de **Numeracion**.
b. La propiedad **ValorNumerico** devolverá el valor de la instancia en sistema decimal.

c. El método **CambiarSistemaDeNumeracion** devolverá una **Numeración** en el sistema recibido**.**
d. El método **EsNumeracionValida**, verificara que la cadena recibida no sea nula o con espacios vacíos y adicionalmente sea un sistema decimal valido.
e. El método **EsSistemaDecimalValido** verificara que la cadena recibida pueda convertirse a tipo **double.**
f. El método **DecimalABinario** verificará que el valor a convertir sea mayor que **0** (*cero*) y realizará la conversión de la parte **entera** de una **Numeracion** de tipo **SistemaDecimal** a **SistemaBinario**, caso contrario devolverá un *msgError*.
g. El método **DecimalAOctal**, solo deberá ser implementado por quien no entrego el trabajo integrador. Verificará que el valor a convertir sea mayor que **0** (*cero*) y realizará la conversión de la parte **entera** de una **Numeracion** de tipo **SistemaDecimal** a **SistemaOctal**, caso contrario devolverá un *msgError*.
h. Generar 2 conversiones implícitas de **double** y **string** a un **SistemaDecimal.**

4. **Clase SistemaBinario:**

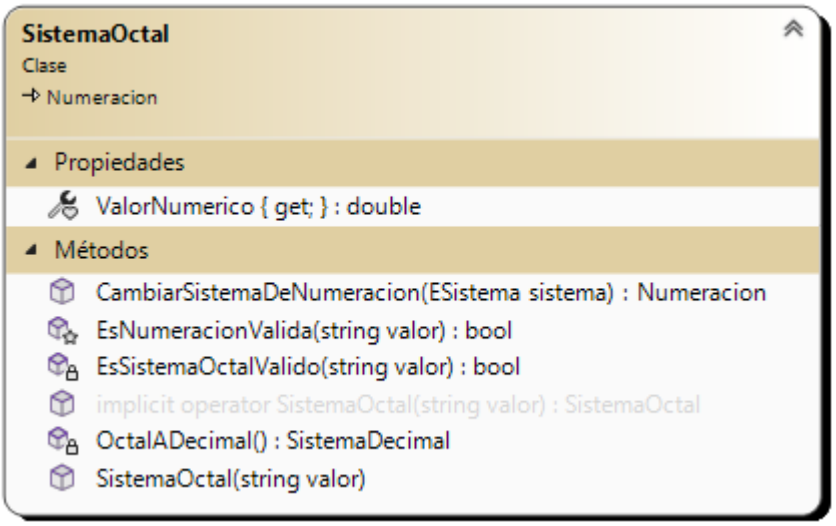


a. Heredara de **Numeracion**.
b. La propiedad **ValorNumerico** devolverá el valor de la instancia en sistema decimal.
c. El método **CambiarSistemaDeNumeracion** devolverá una **Numeración** en el sistema recibido**.**
d. El método **EsNumeracionValida** verificara que la cadena recibida no sea nula o con espacios vacíos y adicionalmente que sea un sistema binario valido.
e. El método **EsSistemaBinarioValido** verificara que la cadena recibida solo posea 1 (*unos*) o 0 (*ceros*).

f. El método **BinarioADecimal** verificará que el *valor* de la instancia no posea un *msgError* y convertirá este a **SistemaDecimal.** Caso contrario devolverá el mínimo de un *double.*

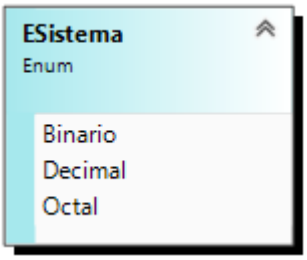
g. Generar una conversión implícita de **string** a **SistemaBinario**.

5. Clase SistemaOctal (Solo si no entregaste el trabajo integrador):

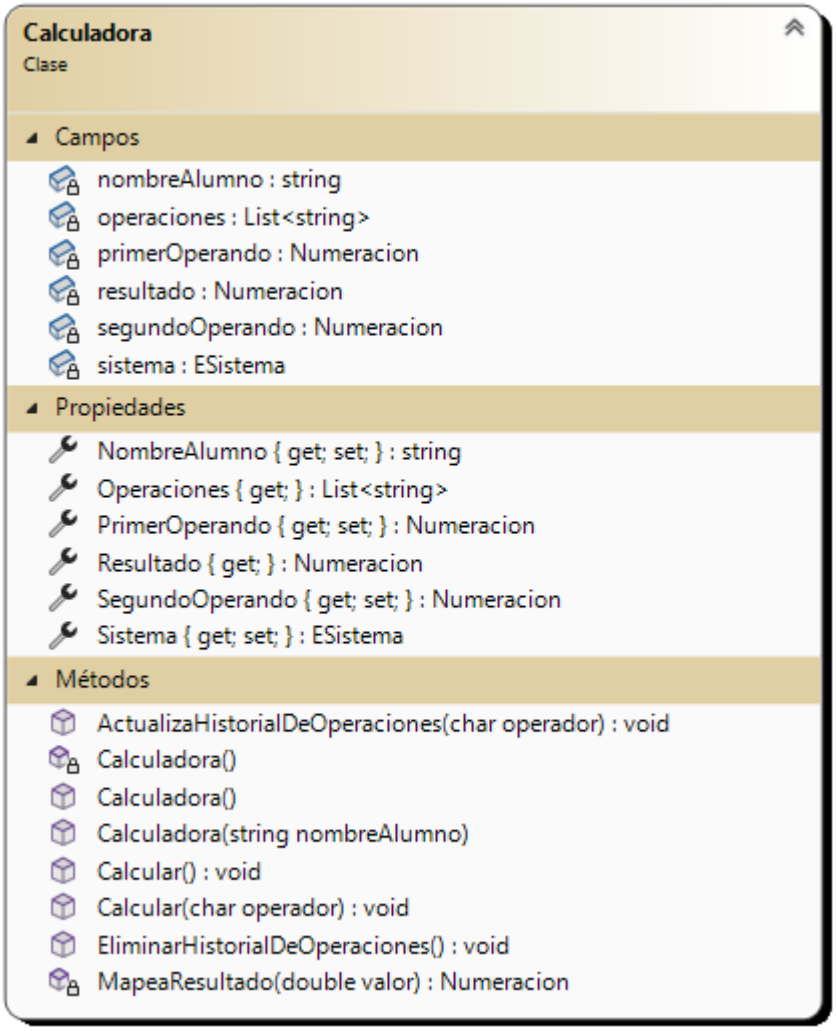


6. Enumerado:

a. Pertenecerá al espacio de nombres.



7. **Clase Calculadora:**



a. Todos sus atributos serán privados.

b. El atributo de tipo ESistema será de clase y u valor se inicializará en el constructor de clase. Su valor por defecto será *Decimal*.

c. El constructor publico sin parámetros será el encargado de instanciar la colección.

d. La propiedad **Operaciones** y **Resultado** serán de solo lectura.

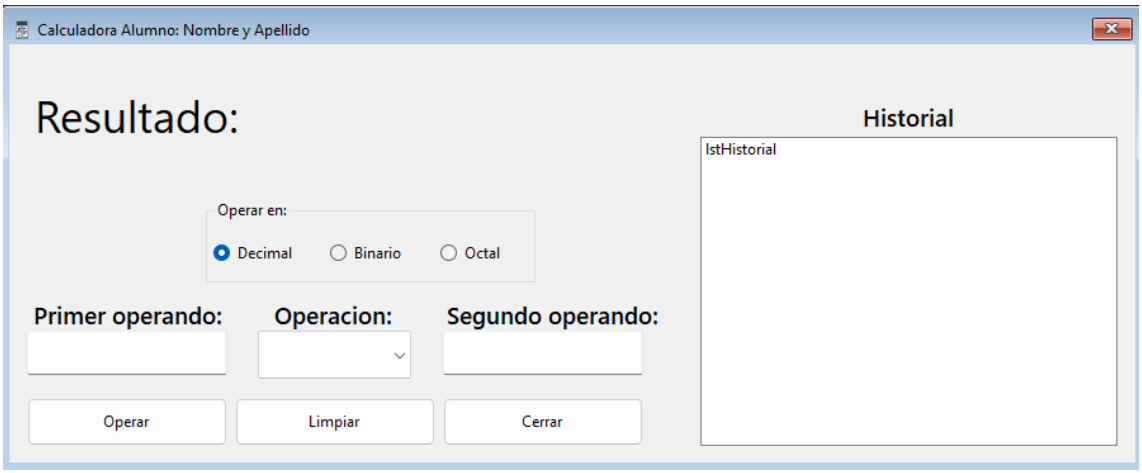
e. La propiedad **Sistema** será de clase.

f. El método Calcular será el encargado de realizar las operaciones matemáticas entre 2 numeraciones. Aspectos a tener en cuenta:

- **i.** Si no se recibe un operador, por defecto se realizará una suma.
- **ii.** Las operaciones se realizarán entre los valores numéricos de cada **Numeración**.
- **iii.** Estas solo se podrán llevar a cabo si ambas son del mismo tipo, de lo contrario su resultado será el mínimo de un **double**.
- **iv.** Le dará un valor al atributo resultado. Este siempre será un resultado mapeado, para ello deberá llamar al método correspondiente.

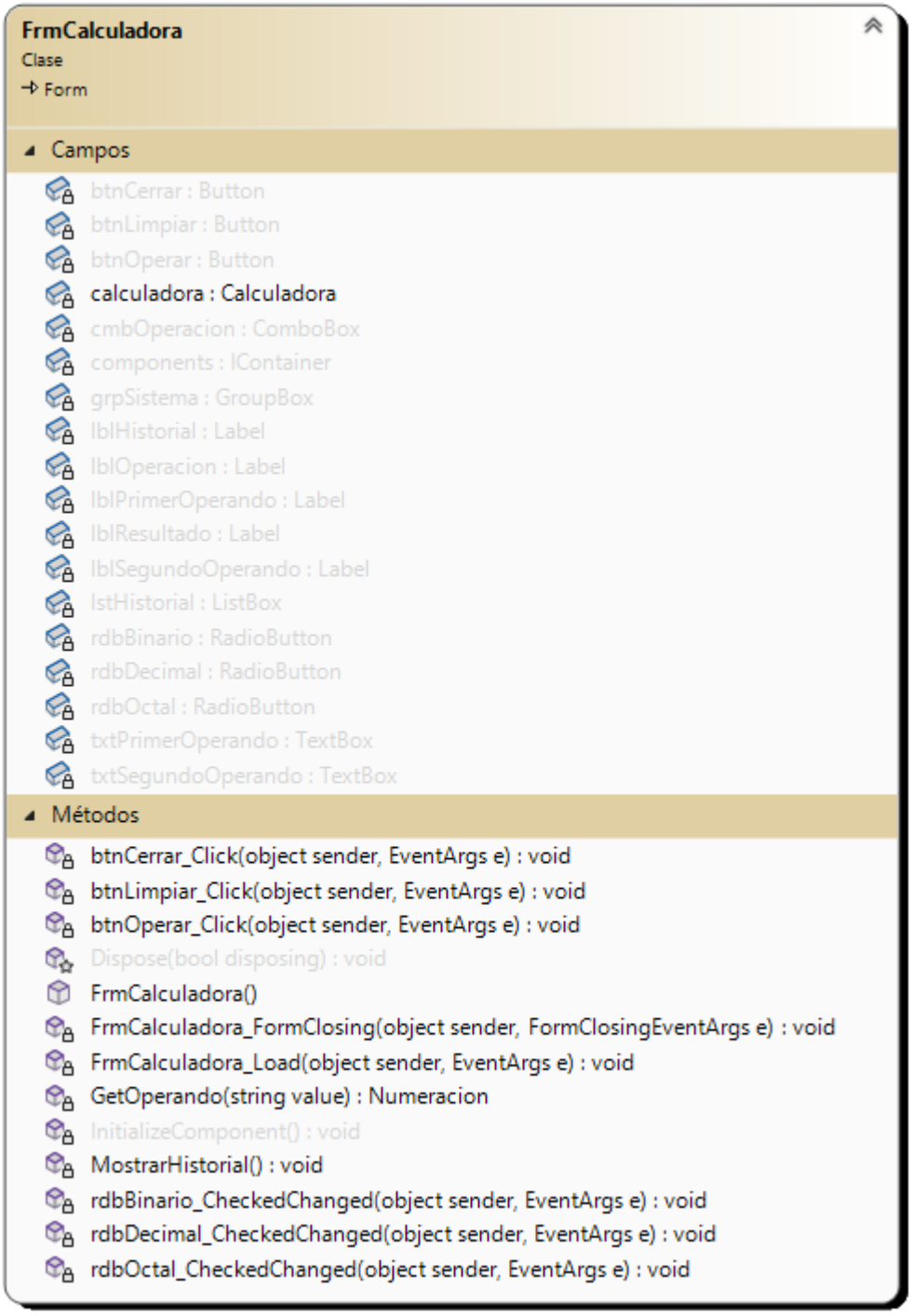
g. El método **MapeaResultado** devolverá una **Numeracion** en **SistemaDecimal**, **SistemaBinario o SistemaOctal,** esto dependerá del sistema de la **Calculadora**.
h. El método **ActualizaHistorialDeOperaciones** generara un string concatenado con:
    i. El *sistema* de la calculadora.
    ii. Los valores de *primerOperando* y *segundoOperando.*
    iii. El operador.
    iv. Utilizar **StringBuilder** para lograr esto.
i. El método **EliminarHistorialDeOperaciones**, limpiara la lista de operaciones.

8. Generar un proyecto de tipo Form denominado View con un diseño similar al siguiente:



a. El mismo deberá aparecer centrado en pantalla, no se podrá maximizar ni tampoco cambiar el tamaño de la ventana durante su ejecución.

b. Se deberá respetar las reglas de estilo para los nombres de los controles.



c. Declarar los siguientes atributos como parte de la clase del form:

```
private Calculadora calculadora;
```

d. En su constructor colocar:

```
this.calculadora = new Calculadora("Nombre y Apellido");
```

e. En el evento load del form colocar:

```
this.cmbOperacion.DataSource = new char[] { '+', '-', '*',
'/' };
```

f. En el evento click del botón operar colocar:

```
char operador;
calculadora.PrimerOperando =
this.GetOperador(this.txtPrimerOperando.Text);
calculadora.SegundoOperando =
this.GetOperador(this.txtSegundoOperando.Text);
operador = (char)this.cmbOperacion.SelectedItem;
this.calculadora.Calcular(operador);
this.calculadora.ActualizaHistorialDeOperaciones(operador)
;
this.lblResultado.Text = $"Resultado:
{calculadora.Resultado.Valor}";
this.MostrarHistorial();
```

g. En el evento click del botón limpiar colocar:

```
this.calculadora.EliminarHistorialDeOperaciones();
this.txtPrimerOperando.Text = string.Empty;
this.txtSegundoOperando.Text = string.Empty;
this.lblResultado.Text = $"Resultado:";
this.MostrarHistorial();
```

h. Generar un método privado que no devuelva nada denominado MostrarHistorial y colocar:

```
this.lstHistorial.DataSource = null;
this.lstHistorial.DataSource =
this.calculadora.Operaciones;
```

i. En el evento Checked Change del radio button de binario colocar:

```
Calculadora.Sistema = ESistema.Binario;
```

j. En el evento Checked Change del radio button de decimal colocar:

```
Calculadora.Sistema = ESistema.Decimal;
```

k. En el evento Checked Change del radio button de octal colocar (Solo si se desarrolla la clase octal)

```
Calculadora.Sistema = ESistema.Octal;
```

l. Generar un método privado que retorne una Numeracion denominado GetOperando y colocar (Si se desarrolla la clase octal, agregar lógica para que funcione):

```
if (Calculadora.Sistema == ESistema.Binario)
{
    return new SistemaBinario(value);
}
return new SistemaDecimal(value);
```

m. En el evento click del botón cerrar colocar:

```
this.Close();
```

n. En el evento closing del form colocar:

```
DialogResult result = MessageBox.Show("Desea cerrar la
calculadora?", "Cierre", MessageBoxButtons.YesNo,
MessageBoxIcon.Question);

if (result == DialogResult.No)
{
      e.Cancel = true;
}
```