



UNIVERSIDAD NACIONAL DE CAJAMARCA

Facultad de ingeniería



Escuela Académico Profesional de Ingeniería de Sistemas

Curso: Programación Aplicada I

Docente: Vázquez Fernández ,Lisi Janet

Estudiante:

- ✓ Reyes Rodriguez, Sayra Silvana
- ✓ Villanueva Quispe, Brayan Alexander

Cajamarca, 30 de enero del 2024

EJERCICIOS PROPUESTOS SEMANA 7

1. Desarrollo de las ventanas WPF.

Encriptador y desencriptador de cadenas con AesManaged

Creamos los atributos que se muestran en la imagen ya que el método encriptar y desencriptaren los vamos a utilizar, en el constructor **Main()**, se inicializan las claves (publickey y secretkey) con valores predefinidos y valorRetorno con una cadena vacía. Además, se llama al método **InitializeComponent()**, que generalmente se utiliza para inicializar componentes de la interfaz de usuario en aplicaciones de Windows Forms o WPF.

```
24 private string publickey, secretkey, valorRetorno ;
25 private MemoryStream ms = null;
26 private CryptoStream cs = null;
27 0 referencias
28 public Main()
29 {
30     InitializeComponent();
31     this.publickey = "12345678";
32     this.secretkey = "87654321";
33     this.valorRetorno = "";
34 }
```

radioButton_encriptar_Checked: Este método se ejecuta cuando el usuario selecciona el radiobutton para encriptar un texto. Realiza las siguientes acciones:

- Verifica si el formulario está validado mediante el método **validarForm()**. Si no está validado, el método retorna tempranamente.
- Invoca el método **encriptar()** para obtener el texto encriptado.
- Si el texto encriptado es nulo, muestra un mensaje de error y desmarca el radiobutton de encriptación.
- Si el texto encriptado es válido, lo establece como el texto del cuadro de texto (**textBox_Copiar**) y desmarca el radiobutton de encriptación.

radioButton_desencriptar_Checked: Este método se ejecuta cuando el usuario selecciona el radiobutton para desencriptar un texto. Realiza las siguientes acciones:

- Verifica si el formulario está validado mediante el método **validarForm()**. Si no está validado, el método retorna tempranamente.
- Invoca el método **desencriptar()** para obtener el texto desencriptado.
- Si el texto desencriptado es nulo, muestra un mensaje de error y desmarca el radiobutton de desencriptación.
- Si el texto desencriptado es válido, lo establece como el texto del cuadro de texto (**textBox_Copiar**) y desmarca el radiobutton de desencriptación.

```

35 1 referencia
36 private void radioButton_encryptar_Checked(object sender, RoutedEventArgs e)
37 {
38     if(!this.validarForm())
39         return;
40
41     String cadenaEncryptar = this.encryptar();
42     if (cadenaEncryptar == null)
43     {
44         MessageBox.Show("Error al encryptar el texto", "Error", MessageBoxButton.OK, MessageBoxImage.Warning);
45         radioButton_encryptar.IsChecked = false;
46         return;
47     }
48     textBox_Copiar.Text = cadenaEncryptar;
49     radioButton_encryptar.IsChecked = false;
50 }
51
52 1 referencia
53 private void radioButton_descryptar_Checked(object sender, RoutedEventArgs e)
54 {
55     if (!this.validarForm())
56         return;
57
58     String cadenaDescryptada = this.descryptar();
59     if (this.descryptar() == null)
60     {
61         MessageBox.Show("Error al descryptar el texto", "Error", MessageBoxButton.OK, MessageBoxImage.Warning);
62         radioButton_descryptar.IsChecked = false;
63         return;
64     }
65
66     textBox_Copiar.Text = cadenaDescryptada;
67     radioButton_descryptar.IsChecked = false;
68 }

```

encryptar(): Este método toma el texto ingresado en un campo de entrada (input_entrada), lo convierte en un arreglo de bytes utilizando UTF-8, y luego lo encripta utilizando el algoritmo DES. El texto encriptado se convierte a una cadena Base64 y se devuelve. Si hay algún error durante el proceso de encriptación, se muestra un mensaje de error al usuario y el método devuelve null.

```

1 referencia
private String encryptar()
{
    try
    {
        string textToEncrypt = input_entrada.Text.Trim();
        byte[] secretkeyByte = { };
        secretkeyByte = System.Text.Encoding.UTF8.GetBytes(this.secretkey);
        byte[] publickeybyte = { };
        publickeybyte = System.Text.Encoding.UTF8.GetBytes(this.publickey);
        byte[] inputbyteArray = System.Text.Encoding.UTF8.GetBytes(textToEncrypt);
        using (DESCryptoServiceProvider des = new DESCryptoServiceProvider())
        {
            ms = new MemoryStream();
            cs = new CryptoStream(ms, des.CreateEncryptor(publickeybyte, secretkeyByte),
                                CryptoStreamMode.Write);
            cs.Write(inputbyteArray, 0, inputbyteArray.Length);
            cs.FlushFinalBlock();
            this.valorRetorno = Convert.ToBase64String(ms.ToArray());
        }
        return this.valorRetorno;
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error al encryptar el texto: " + ex.Message, "Error", MessageBoxButton.OK, MessageBoxImage.Warning);
        //throw new Exception(ex.Message, ex.InnerException);
        return null;
    }
}

```

descryptar(): Este método toma el texto encriptado ingresado en el mismo campo de entrada, lo convierte de Base64 a un arreglo de bytes, y luego lo descrypta utilizando el algoritmo DES. El texto descryptado se convierte de nuevo a una cadena y se devuelve. Si hay algún error durante el proceso de descryptación, el método devuelve null.

```

public String descryptar()
{
    try
    {
        string textToDecrypt = input_entrada.Text.Trim();
        byte[] privatekeyByte = { };
        privatekeyByte = System.Text.Encoding.UTF8.GetBytes(this.secretkey);
        byte[] publickeybyte = { };
        publickeybyte = System.Text.Encoding.UTF8.GetBytes(this.publickey);
        byte[] inputbyteArray = new byte[textToDecrypt.Replace(" ", "+").Length];
        inputbyteArray = Convert.FromBase64String(textToDecrypt.Replace(" ", "+"));
        using (DESCryptoServiceProvider des = new DESCryptoServiceProvider())
        {
            ms = new MemoryStream();
            cs = new CryptoStream(ms, des.CreateDecryptor(publickeybyte, privatekeyByte),
                                CryptoStreamMode.Write);
            cs.Write(inputbyteArray, 0, inputbyteArray.Length);
            cs.FlushFinalBlock();
            Encoding encoding = Encoding.UTF8;
            this.valorRetorno = encoding.GetString(ms.ToArray());
        }
        return this.valorRetorno;
    }
    catch (Exception ae)
    {
        //throw new Exception(ae.Message, ae.InnerException);
        return null;
    }
}

```

El método **validarForm()** verifica si hay algún texto ingresado en un campo de entrada (input_entrada). Si el campo está vacío, muestra un mensaje de advertencia al usuario y devuelve false. Si hay texto presente, devuelve true. Es utilizado para validar el formulario antes de proceder con operaciones de encriptación o desencriptación.

```

2 referencias
private bool validarForm()
{
    string textToEncrypt = input_entrada.Text.Trim();
    if (textToEncrypt == "")
    {
        MessageBox.Show("Ingresa algún valor", "Alert", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return false;
    }
    return true;
}

```

El método **bnt_copiar_Click** se activa cuando se hace clic en un botón (bnt_copiar). Su función es copiar el texto presente en un cuadro de texto (textBox_Copiar) al portapapeles del sistema utilizando Clipboard.SetText(). Luego muestra un mensaje de éxito al usuario utilizando MessageBox.Show().

```

1 referencia
private void bnt_copiar_Click(object sender, RoutedEventArgs e)
{
    Clipboard.SetText(textBox_Copiar.Text);
    MessageBox.Show("Se copio correctamente", "Success", MessageBoxButtons.OK, MessageBoxIcon.Information);
}
1 referencia

```

2. Ejecución de la aplicación.

Encriptar y Desencriptar

Clave:

Clave1234434holamundo

GIh4VbzW4fqHVMWNvQBctTiKj2R2yPNs

☐ Encriptar

☐ Desencriptar

Copiar

Encriptar y Desencriptar

Clave:

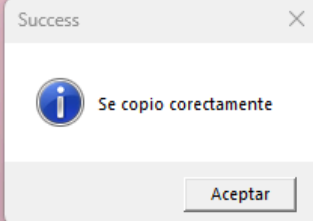
Clave1234434holamundo

GIh4VbzW4fqHVMWNvQBctTiKj2R2yPNs

☐ Encriptar

☐ Desencriptar

Copiar



Encriptar y Desencriptar

Clave:

☒ Encriptar ☐ Desencriptar

Copiar

3. Link del drive de la aplicación WPF.

<https://drive.google.com/file/d/1wwfIbNZhKRdiFAeCDODTojy2ufLezKJ4/view?usp=sharing>

Ejercicio 2.

Cualquier triángulo puede resolverse (resolución de triángulos) si se conocen tres de sus elementos, donde, como mínimo, uno de ellos debe de ser un lado. En particular, conociendo dos de sus lados y el ángulo que forman se puede calcular el área de un triángulo por razones trigonométricas. Se pide elaborar un programa que implemente el calculo del área de un triangulo tomando en cuenta lo comentado anteriormente y las formulas correspondientes.

1. Desarrollo de las ventanas WPF.

Creación de vista:

- ❖ Primeramente iniciamos con la pantalla inicial conectando a través de un button a nuestra ventana.
- ❖ Luego para esto añadí los checkbox que permiten al usuario seleccionar los lados que conoce al ingresar, al igual que su ángulo.
- ❖ Utilice textbox para recibir los valores que ingresan además agregue una imagen para que al usuario le sea mas entendible.
- ❖

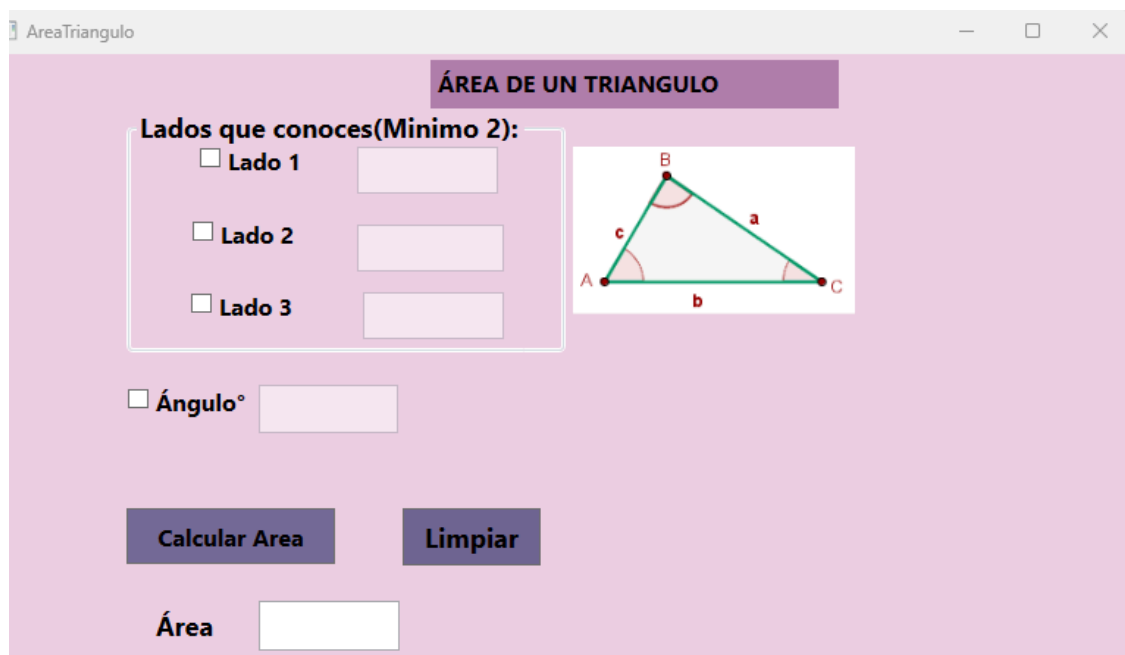
Pasos para crear la solución del ejercicio:

- ❖ Inicialmente configuramos a la ventana para que desactive las cajas de texto correspondientes a los lados y al ángulo porque inicialmente no se selecciona ninguno.
- ❖ Cuando el usuario hace clic en el botón "Calcular Área", se ejecuta el método `btn_calcularArea_Click`.
- ❖ Se recopilan los valores de los lados (`lado1`, `lado2`, `lado3`) y el ángulo (`Angulo`) dependiendo de cuáles casillas de verificación están marcadas.
- ❖ Luego, se utiliza la fórmula del seno para calcular el área del triángulo según los lados y el ángulo seleccionado.
- ❖ El resultado se muestra en la caja de texto `txt_area`.
- ❖ Utilizamos el botón "Limpiar", se ejecuta el método `txt_limpiar_Click`.
- ❖ Este método llama a `Limpiar()`, que restablece todos los campos y controles a sus valores iniciales.
- ❖ Manejamos los eventos de `CheckBoxes`: Cuando el usuario marca o desmarca las casillas de verificación, se activan o desactivan las cajas de texto correspondientes.
- ❖ Manejamos los eventos de `TextBoxes` validando: Aseguramos de que solo se puedan ingresar números en las cajas de texto, bloqueando otros caracteres.
- ❖ Manejamos los eventos de `CheckBoxes` Desmarcados: Cuando se desmarca una casilla de verificación, se limpia y desactiva la caja de texto correspondiente.

Ventana de inicio:



Ventana principal:



Código:


```

/// </summary>
2 referencias
public partial class MainWindow : Window
{
    0 referencias
    public MainWindow()
    {
        InitializeComponent();
    }

    1 referencia
    private void btn_areatrinagulo_Click(object sender, RoutedEventArgs e)
    {
        AreaTriangulo pantallaarea= new AreaTriangulo();
        pantallaarea.Show();
    }

    1 referencia
    private void txt_encryptar_Click(object sender, RoutedEventArgs e)
    {
        EncriptarClave encriptarClave = new EncriptarClave();
        encriptarClave.Show();
    }
}

```

```

/// </summary>
4 referencias
public partial class AreaTriangulo : Window
{
    1 referencia
    public AreaTriangulo()
    {
        InitializeComponent();
        txt_lado1.IsEnabled = false;
        txt_lado2.IsEnabled = false;
        txt_lado3.IsEnabled = false;
        txt_angulo.IsEnabled = false;
    }

    1 referencia
    private void btn_calcularArea_Click(object sender, RoutedEventArgs e)
    {
        double lado1 = 0, lado2 = 0, lado3 = 0;
        double Area = 0;
        double Angulo = 0;

        if (chbx_lado1.IsChecked == true)
            lado1 = Double.Parse(txt_lado1.Text);

        if (chbx_lado2.IsChecked == true)
            lado2 = Double.Parse(txt_lado2.Text);

        if (chbx_lado3.IsChecked == true)
            lado3 = Double.Parse(txt_lado3.Text);
        if (chbx_angulo.IsChecked == true)
            Angulo = Double.Parse(txt_angulo.Text);
        //diferentes maneras de calcular las areas dependiendo sus lados
        if (chbx_lado1.IsChecked == true && chbx_lado2.IsChecked == true)
        {

```

```

s_larea
    Sb.EjerciciosIarea.AreaInangulo
    AreaInangulo()

    Angulo = Double.Parse(txt_angulo.Text);
    //diferentes maneras de calcular las areas dependiendo sus lados
    if (chbx_lado1.IsChecked == true && chbx_lado2.IsChecked == true)
    {
        Area = lado1 * lado2 * Math.Sin(Angulo * Math.PI / 180) * 1 / 2;
    }

    if (chbx_lado1.IsChecked == true && chbx_lado3.IsChecked == true)
    {
        Area = lado1 * lado3 * Math.Sin(Angulo * Math.PI / 180) * 1 / 2;
    }

    if (chbx_lado2.IsChecked == true && chbx_lado3.IsChecked == true)
    {
        Area = lado2 * lado3 * Math.Sin(Angulo * Math.PI / 180) * 1 / 2;
    }

    if (chbx_lado1.IsChecked == true && chbx_lado2.IsChecked == true && chbx_lado3.IsChecked == true)
    {
        double sumaperimetro = (lado1 + lado2 + lado3) / 2;
        Area = Math.Sqrt(sumaperimetro * (sumaperimetro - lado1) * (sumaperimetro - lado2) * (sumaperimetro - lado3));
        txt_angulo.Clear();
        chbx_angulo.IsChecked = false;
    }
    txt_area.Text = Area.ToString("N2");
}

```

```

ros_larea
    Sb.EjerciciosIarea.AreaInangulo
    AreaInangulo()

    1 referencia
    private void txt_limpiar_Click(object sender, RoutedEventArgs e)
    {
        Limpiar();
    }

    1 referencia
    private void Limpiar()
    {
        txt_lado1.IsEnabled = false;
        txt_lado2.IsEnabled = false;
        txt_lado3.IsEnabled = false;
        txt_angulo.IsEnabled = false;
        chbx_lado1.IsChecked = false;
        chbx_lado2.IsChecked = false;
        chbx_lado3.IsChecked = false;
        chbx_angulo.IsChecked = false;
        txt_lado1.Clear();
        txt_lado2.Clear();
        txt_lado3.Clear();
        txt_angulo.Clear();
        txt_area.Clear();
    }

```

```

1 referencia
private void chbx_lado1_Checked(object sender, RoutedEventArgs e)
{
    txt_lado1.IsEnabled = true;
}

1 referencia
private void chbx_lado2_Checked(object sender, RoutedEventArgs e)
{
    txt_lado2.IsEnabled = true;
}

1 referencia
private void chbx_lado3_Checked(object sender, RoutedEventArgs e)
{
    txt_lado3.IsEnabled = true;
}

1 referencia
private void chbx_angulo_Checked(object sender, RoutedEventArgs e)
{
    txt_angulo.IsEnabled = true;
}

1 referencia
private void txt_lado1_PreviewTextInput(object sender, TextCompositionEventArgs e)
{
    // Solo permite números
    if (!char.IsDigit(e.Text, 0))
    {
        e.Handled = true; // Cancela la entrada
    }
}

1 referencia

```

```

1 referencia
private void txt_lado2_PreviewTextInput(object sender, TextCompositionEventArgs e)
{
    // Solo permite números
    if (!char.IsDigit(e.Text, 0))
    {
        e.Handled = true; // Cancela la entrada
    }
}

1 referencia
private void txt_lado3_PreviewTextInput(object sender, TextCompositionEventArgs e)
{
    // Solo permite números
    if (!char.IsDigit(e.Text, 0))
    {
        e.Handled = true; // Cancela la entrada
    }
}

1 referencia
private void txt_angulo_PreviewTextInput(object sender, TextCompositionEventArgs e)
{
    // Solo permite números
    if (!char.IsDigit(e.Text, 0))
    {
        e.Handled = true; // Cancela la entrada
    }
}

1 referencia

```

```

1 referencia
private void chbx_lado1_Unchecked(object sender, RoutedEventArgs e)
{
    txt_lado1.Clear();
    txt_lado1.IsEnabled = false;
}

1 referencia
private void chbx_lado2_Unchecked(object sender, RoutedEventArgs e)
{
    txt_lado2.Clear();
    txt_lado2.IsEnabled = false;
}

1 referencia
private void chbx_lado3_Unchecked(object sender, RoutedEventArgs e)
{
    txt_lado3.Clear();
    txt_lado3.IsEnabled = false;
}

1 referencia
private void chbx_angulo_Unchecked(object sender, RoutedEventArgs e)
{
    txt_angulo.Clear();
    txt_angulo.IsEnabled = false;
}

```

2. Ejecución de la aplicación.

AreaTriangulo

ÁREA DE UN TRIANGULO

Lados que conoces(Minimo 2):

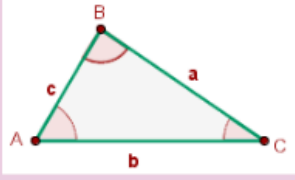
☒ Lado 1

☒ Lado 2

☐ Lado 3

☒ Ángulo°

Área



AreaTriangulo

ÁREA DE UN TRIANGULO

Lados que conoces(Minimo 2):

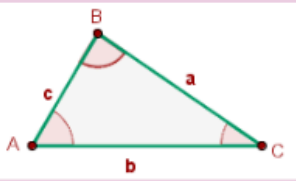
☐ Lado 1

☒ Lado 2

☒ Lado 3

☒ Ángulo°

Área



AreaTriangulo

ÁREA DE UN TRIANGULO

Lados que conoces(Minimo 2):

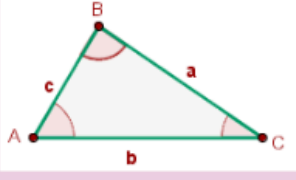
☒ Lado 1

☒ Lado 2

☒ Lado 3

☐ Ángulo°

Área



3. Link del drive de la aplicación WPF.

<https://drive.google.com/file/d/1wwflbNZhKRdiFAeCDODTojy2ufLezKJ4/view?usp=sharing>