Prueba de Caja Blanca

"Generación de proformas MarcallTex"

Integrantes:

Cañola Kevin

Marcalla

Cristhian

Lugamaña

Mateo

Tasiguano

Eduardo

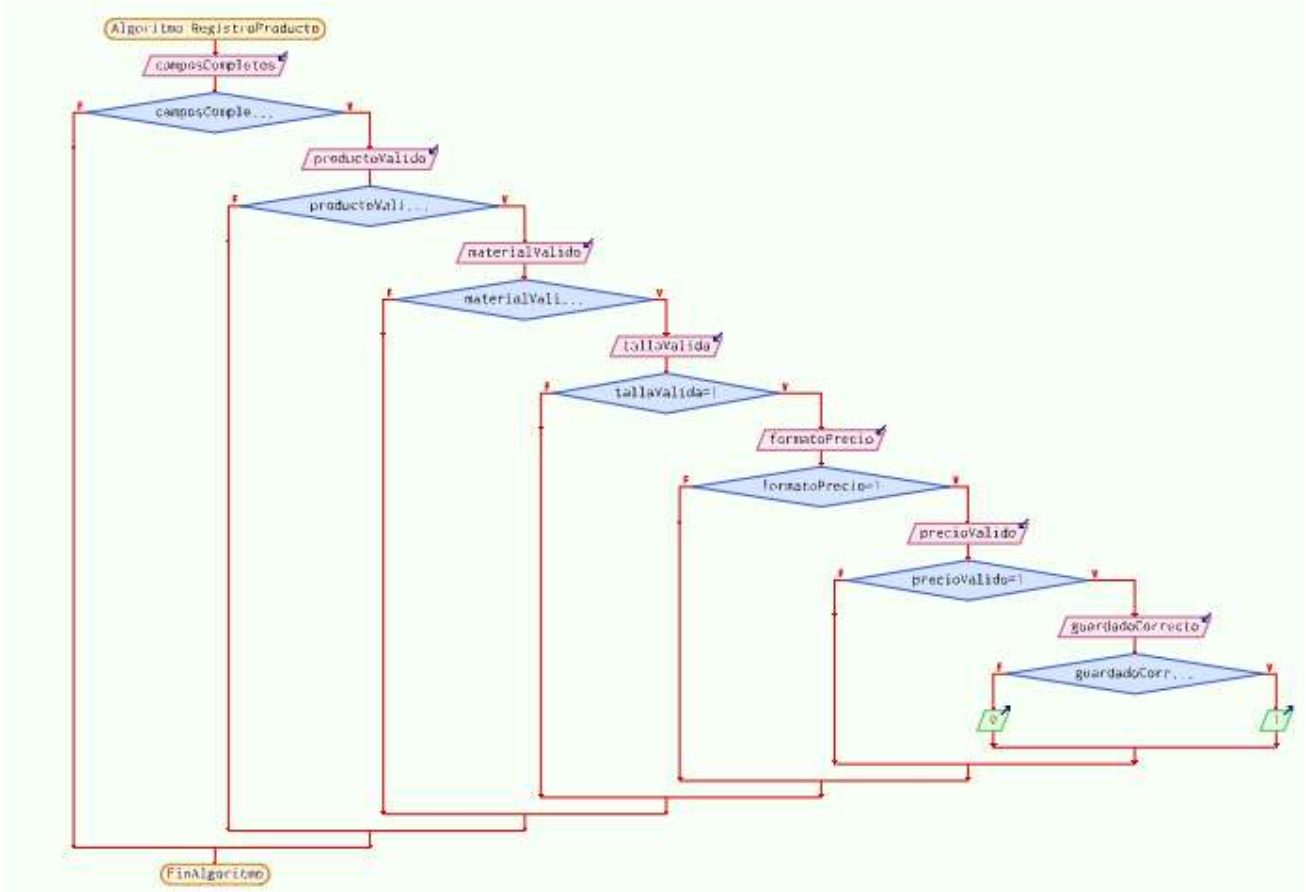
Fecha: 2025/11/25

RF N6ª REGISTROS DE PRODUCTOS

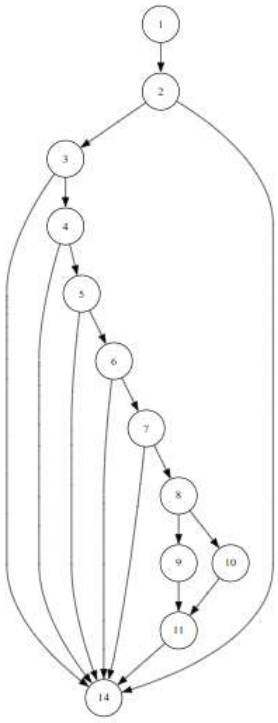
26. CÓDIGO FUENTE

```
' ===== BOTÓN GUARDAR =====
0 referencias
Private Sub btnGuardar_Click(sender As Object, e As EventArgs) Handles btnGuardar.Click
    ' Campos obligatorios
    If txtProducto.Text.Trim = "" OrElse
       txtTalla.Text.Trim = "" OrElse
       txtTipoMaterial.Text.Trim = "" OrElse
       txtPrecioUnitario.Text.Trim = "" Then
        MessageBox.Show("Complete todos los campos")
        Exit Sub
    End If

    ' Validación producto
    If Not Regex.IsMatch(txtProducto.Text.Trim, "^[a-zA-Z\s]+$") Then
        MessageBox.Show("El nombre del producto solo debe contener letras y espacios")
        txtProducto.Focus()
        Exit Sub
    End If

    ' Validación tipo de material
    If Not Regex.IsMatch(txtTipoMaterial.Text.Trim, "^[a-zA-Z\s]+$") Then
        MessageBox.Show("El tipo de material solo debe contener letras y espacios")
        txtTipoMaterial.Focus()
        Exit Sub
    End If

    ' Validación talla lógica
    Dim valorTalla As String = txtTalla.Text.Trim().ToUpper()
    Dim tallasValidas As String() = {"S", "M", "L", "XL", "XXL"}
    Dim esNumeroValido As Boolean = Integer.TryParse(valorTalla, Nothing) AndAlso CInt(valorTalla) >= 30 AndAlso CInt(valorTalla) <
    If Not tallasValidas.Contains(valorTalla) AndAlso Not esNumeroValido Then
        MessageBox.Show("Ingrese una talla válida (S, M, L, XL, XXL o número entre 30 y 50)")
        txtTalla.Focus()
        Exit Sub
    End If
        ' Validación precio
        If Not Regex.IsMatch(txtPrecioUnitario.Text.Trim, "^\d+(\.\d{1,2})?$") Then
            MessageBox.Show("El precio unitario debe ser un número válido, máximo 2 decimales")
            txtPrecioUnitario.Focus()
            Exit Sub
        End If

        Dim precio As Double
        If Not Double.TryParse(txtPrecioUnitario.Text.Trim, precio) Then
            MessageBox.Show("Ingrese un precio válido")
            txtPrecioUnitario.Focus()
            Exit Sub
        End If

        ' Guardar producto
        Dim nuevoProducto As New Producto With {
            .Producto = txtProducto.Text.Trim(),
            .Talla = valorTalla,
            .TipoMaterial = txtTipoMaterial.Text.Trim(),
            .PrecioUnitario = precio
        }

        Try
            productosCollection.InsertOne(nuevoProducto)
            MessageBox.Show("Producto registrado correctamente")
            LimpiarCampos()
        Catch ex As Exception
            MessageBox.Show("Error al guardar: " & ex.Message)
        End Try
    End Sub
```

27. DIAGRAMA DE FLUJO (DF)



28. GRAFO DE FLUJO (GF)



29. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

Ruta	Secuencia de nodos	Descripción
R1	1 → 2 → 14	Campos incompletos
R2	1 → 2 → 3 → 14	Producto inválido
R3	1 → 2 → 3 → 4 → 14	Material inválido
R4	1 → 2 → 3 → 4 → 5 → 14	Talla inválida
R5	1 → 2 → 3 → 4 → 5 → 6 → 14	Precio formato inválido
R6	1 → 2 → 3 → 4 → 5 → 6 → 7 → 14	Precio no numérico
R7	1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 → 9 → 11 → 14	Registro exitoso
R8	1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 → 10 → 11 → 14	Error al guardar

30. COMPLEJIDAD CICLOMÁTICA

La complejidad ciclomática permite determinar el número de caminos independientes del sistema.

Datos del grafo

- **N (Número de nodos):** 14
- **P (Número de nodos predicados):** 7
- **A (Número de aristas):** 4

Cálculo

Fórmula 1:
V(G) = P + 1
V(G) = 7 + 1 = 8

Fórmula 2:
V(G) = A − N + 2
V(G) = 20 − 14 + 2 = 8

Resultado

La complejidad ciclomática del requisito **Registro de Productos** es **8**,
por lo tanto existen **8 caminos básicos independientes** que deben probarse.

DONDE

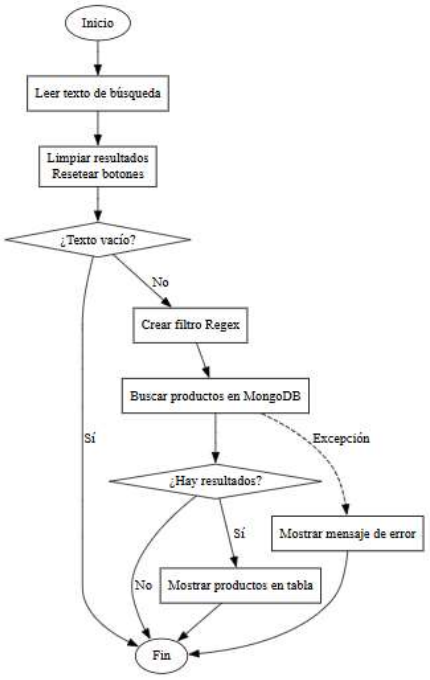
P: Número de nodos predicado
A: Número de aristas
N: Número de nodos

RF N7ª BUSQUEDA DE PRODUCTOS

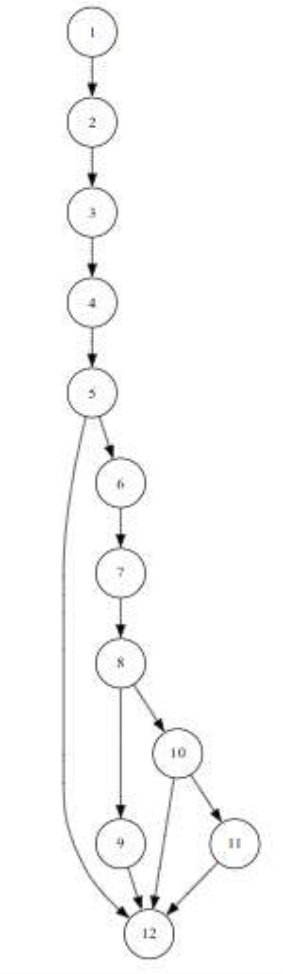
31. CÓDIGO FUENTE

```
3 referencias
Private Sub BuscarProducto()
    Dim texto As String = txtBuscarProducto.Text.Trim()

    dgvResultados.Rows.Clear()
    productoSeleccionado = Nothing
    btnEditar.Enabled = False
    btnEliminar.Enabled = False

    If texto = "" Then Exit Sub

    Try
        ' ⚠ SIMILITUD (EMPIEZA CON / CONTIENE)
        Dim filtro = Builders(Of Producto).Filter.Regex(
            "producto",
            New BsonRegularExpression(texto, "i")
        )

        Dim lista = productosCollection.Find(filtro).ToList()

        For Each p In lista
            dgvResultados.Rows.Add(
                p.Id,
                p.Producto,
                p.Talla,
                p.TipoMaterial,
                Math.Round(p.PrecioUnitario, 2)
            )
        Next

    Catch ex As Exception
        MessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
    End Try
End Sub
```

32. DIAGRAMA DE FLUJO (DF)



33. GRAFO DE FLUJO (GF)



34. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

Ruta	Secuencia de Nodos	Descripción
R1	1 – 2 – 3 – 4 – 5 – 12	El usuario no ingresa texto de búsqueda, el proceso termina sin consultar la base de datos.
R2	1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 12	El usuario ingresa texto válido, la consulta se ejecuta correctamente y se muestran los productos encontrados.
R3	1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 10 – 12	El usuario ingresa texto válido, la consulta se ejecuta correctamente pero no se encuentran productos.
R4	1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 10 – 11 – 12	El usuario ingresa texto válido pero ocurre un error durante la consulta a la base de datos y se muestra un mensaje de error.

35. COMPLEJIDAD CICLOMÁTICA

La complejidad ciclomática permite determinar el número de caminos independientes del sistema.

Datos del grafo

• **N (Número de nodos):** 12
• **P (Número de nodos predicados):** 4
• **A (Número de aristas):** 14

Cálculo

Fórmula 1:
V(G) = P + 1
V(G) = 4 + 1 = 5

Fórmula 2:
V(G) = A − N + 2
V(G) = 14 − 12 + 2 = 5

Resultado
La **complejidad ciclomática del requisito 7 (Búsqueda de productos por similitud)** es **5**, lo que indica que existen **cinco caminos básicos independientes** que deben ser probados mediante pruebas de caja blanca.

DONDE

P: Número de nodos predicado
A: Número de aristas
N: Número de nodos

RF N8ª MODIFICACION DE PRODUCTOS

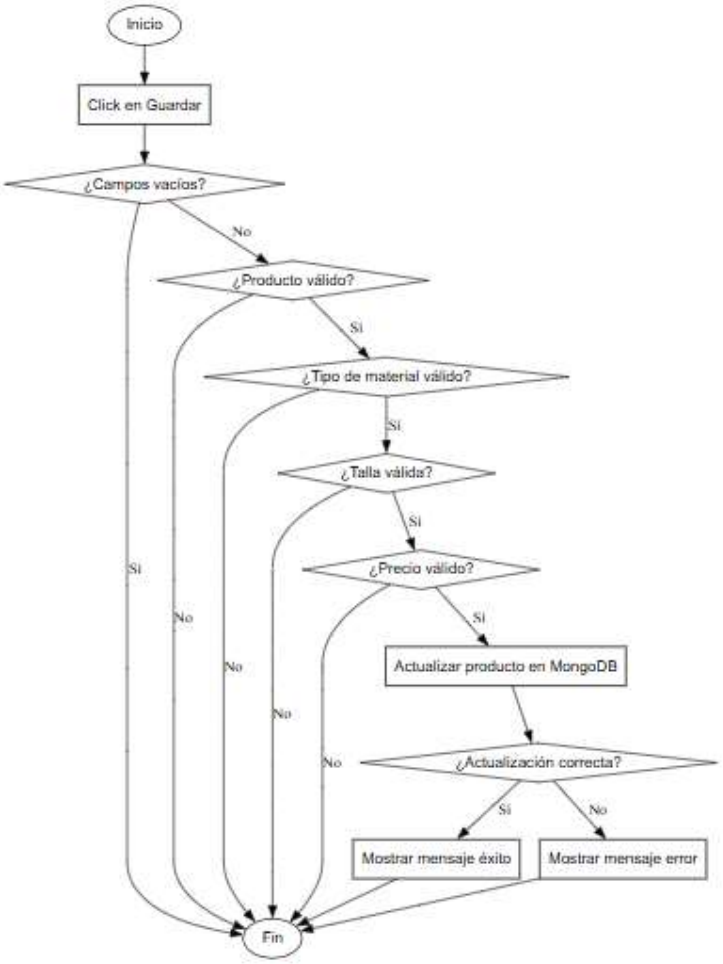
36. CÓDIGO FUENTE

```
28      Private Sub btnGuardar_Click(sender As Object, e As EventArgs) Handles btnGuardar.Click
29
30          ' ===== VALIDAR CAMPOS VACÍOS =====
31          If txtProducto.Text.Trim() = "" OrElse
32             txtTalla.Text.Trim() = "" OrElse
33             txtTipoMaterial.Text.Trim() = "" OrElse
34             txtPrecioUnitario.Text.Trim() = "" Then
35              MessageBox.Show("Complete todos los campos")
36              Exit Sub
37          End If
38
39          ' ===== VALIDAR PRODUCTO =====
40          If Not Regex.IsMatch(txtProducto.Text.Trim(), "^[A-Za-z\s]+$") Then
41              MessageBox.Show("El producto solo debe contener letras y espacios")
42              txtProducto.Focus()
43              Exit Sub
44          End If
45
46          ' ===== VALIDAR TIPO MATERIAL =====
47          If Not Regex.IsMatch(txtTipoMaterial.Text.Trim(), "^[A-Za-z\s]+$") Then
48              MessageBox.Show("El tipo de material solo debe contener letras y espacios")
49              txtTipoMaterial.Focus()
50              Exit Sub
51          End If
52
53          ' ===== VALIDAR TALLA =====
54          Dim talla As String = txtTalla.Text.Trim().ToUpper()
55          Dim tallasValidas As String() = {"S", "M", "L", "XL", "XXL"}
56
57          Dim esTallaNumero As Boolean = Regex.IsMatch(talla, "^\d+$")
58          Dim esTallaLetra As Boolean = tallasValidas.Contains(talla)
59
60          If esTallaNumero Then
61              Dim tallaNum As Integer = CInt(talla)
62              If tallaNum < 30 OrElse tallaNum > 50 Then
63                  MessageBox.Show("La talla numérica debe estar entre 30 y 50")
```

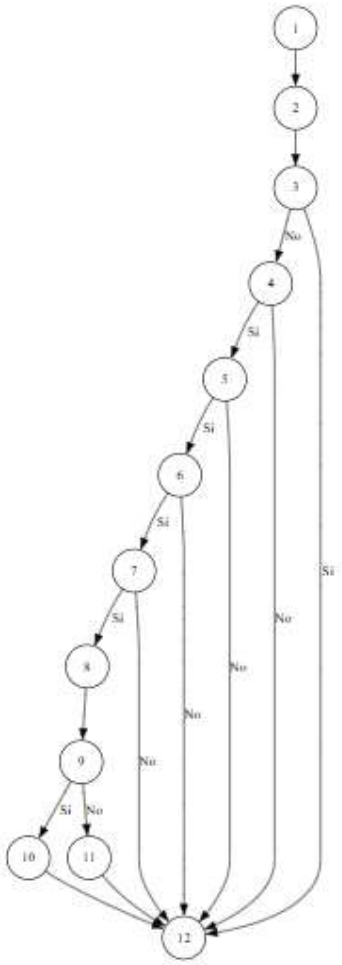
```
                txtTalla.Focus()
                Exit Sub
            End If
        ElseIf Not esTallaLetra Then
            MessageBox.Show("Ingrese una talla válida (S, M, L, XL, XXL o número entre 30 y 50)")
            txtTalla.Focus()
            Exit Sub
        End If

        ' ===== VALIDAR PRECIO =====
        Dim precio As Decimal
        If Not Decimal.TryParse(
            txtPrecioUnitario.Text.Trim(),
            NumberStyles.AllowDecimalPoint,
            CultureInfo.InvariantCulture,
            precio
        ) Then
            MessageBox.Show("Ingrese un precio válido")
            txtPrecioUnitario.Focus()
            Exit Sub
        End If

        precio = Math.Round(precio, 2)

        ' ===== ACTUALIZAR EN MONGODB =====
        Try
            Dim filter = Builders(Of Producto).Filter.Eq(Function(p) p.Id, idProducto)

            Dim update = Builders(Of Producto).Update _
                .Set(Function(p) p.Producto, txtProducto.Text.Trim()) _
                .Set(Function(p) p.Talla, talla) _
                .Set(Function(p) p.TipoMaterial, txtTipoMaterial.Text.Trim()) _
                .Set(Function(p) p.PrecioUnitario, Convert.ToDouble(precio))

            productosCollection.UpdateOne(filter, update)
            MessageBox.Show("Producto actualizado correctamente")
            Me.Close()

        Catch ex As Exception
            MessageBox.Show("Error al actualizar: " & ex.Message)
        End Try
    End Sub

    ' ---------------- CANCELAR ----------------
    0 referencias
    Private Sub btnCancelar_Click(sender As Object, e As EventArgs) Handles btnCancelar.Click
        Me.Close()
    End Sub

    ' ---------------- BLOQUEO DE ENTRADAS ----------------

    ' Producto: solo letras
    0 referencias
    Private Sub txtProducto_KeyPress(sender As Object, e As KeyPressEventArgs) Handles txtProducto.KeyPress
        If Not Char.IsLetter(e.KeyChar) AndAlso
           Not Char.IsWhiteSpace(e.KeyChar) AndAlso
           Not Char.IsControl(e.KeyChar) Then
            e.Handled = True
        End If
    End Sub

    ' Tipo material: solo letras
    0 referencias
    Private Sub txtTipoMaterial_KeyPress(sender As Object, e As KeyPressEventArgs) Handles txtTipoMaterial.KeyPress
        If Not Char.IsLetter(e.KeyChar) AndAlso
           Not Char.IsWhiteSpace(e.KeyChar) AndAlso
           Not Char.IsControl(e.KeyChar) Then
            e.Handled = True
        End If
    End Sub
```

```
132
133         ' Talla: letras o números (no símbolos)
            0 referencias
134         Private Sub txtTalla_KeyPress(sender As Object, e As KeyPressEventArgs) Handles txtTalla.KeyPress
135             If Not Char.IsLetterOrDigit(e.KeyChar) AndAlso
136                Not Char.IsControl(e.KeyChar) Then
137                 e.Handled = True
138             End If
139         End Sub
140
141         ' Precio: solo números y un punto
            0 referencias
142         Private Sub txtPrecioUnitario_KeyPress(sender As Object, e As KeyPressEventArgs) Handles txtPrecioUnitario.KeyPress
143             Dim txt As TextBox = CType(sender, TextBox)
144
145             If Not Char.IsDigit(e.KeyChar) AndAlso
146                Not Char.IsControl(e.KeyChar) AndAlso
147                e.KeyChar <> "."c Then
148                 e.Handled = True
149             End If
150
151             ' Solo un punto decimal
152             If e.KeyChar = "."c AndAlso txt.Text.Contains(".") Then
153                 e.Handled = True
154             End If
155         End Sub
156
157 End Class
```

37. DIAGRAMA DE FLUJO (DF)



38. GRAFO DE FLUJO (GF)



39. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

Ruta	Secuencia de nodos	Descripción
R1	1 → 2 → 3 → 12	Campos vacíos → el proceso termina sin actualizar
R2	1 → 2 → 3 → 4 → 12	Producto inválido → termina sin actualizar
R3	1 → 2 → 3 → 4 → 5 → 12	Tipo de material inválido → termina sin actualizar
R4	1 → 2 → 3 → 4 → 5 → 6 → 12	Talla inválida → termina sin actualizar
R5	1 → 2 → 3 → 4 → 5 → 6 → 7 → 12	Precio inválido → termina sin actualizar
R6	1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 → 9 → 10 → 12	Todo correcto → actualización exitosa
R7	1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 → 9 → 11 → 12	Todo correcto → error en actualización

40. COMPLEJIDAD CICLOMÁTICA

La complejidad ciclomática permite determinar el número de caminos independientes del sistema.

Datos del grafo

• **N (Número de nodos):** 12
• **P (Número de nodos predicados):** 6
• **A (Número de aristas):** 14

Cálculo

Fórmula 1:
V(G) = P + 1
V(G) = 6 + 1 = 7

Fórmula 2:
V(G) = A − N + 2
V(G) = 14 − 12 + 2 = 4

Resultado
La complejidad ciclomática del REQ 8 es 7, lo que significa que existen **7 caminos básicos independientes** que debes probar para cobertura de caja blanca.

DONDE

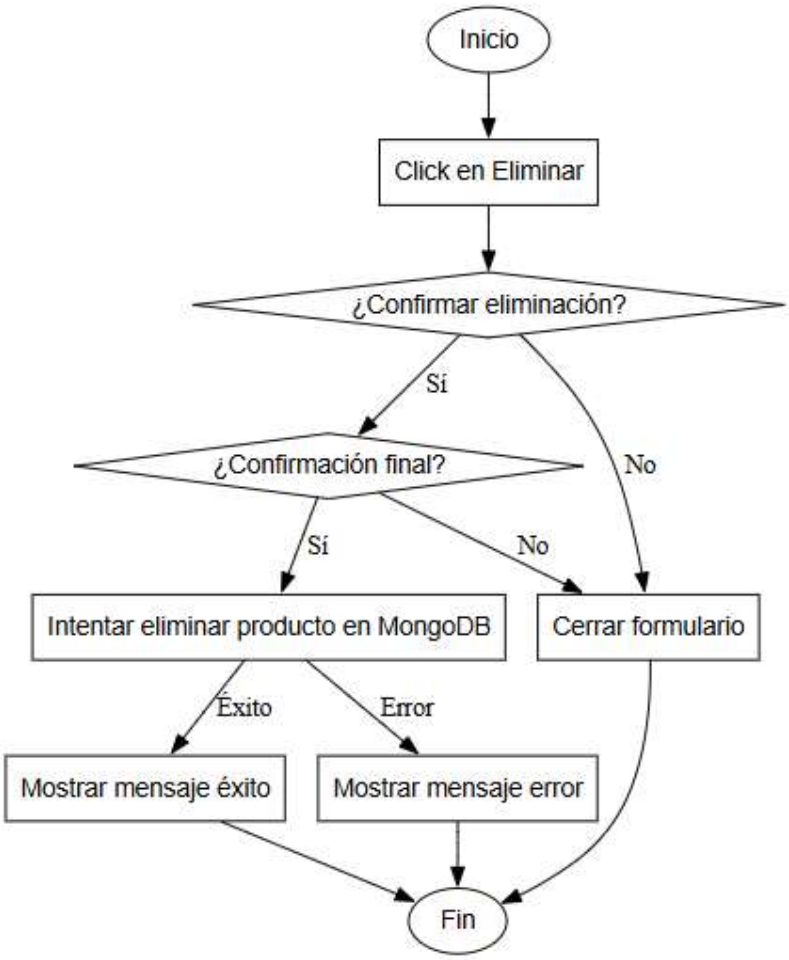
P: Número de nodos predicado
A: Número de aristas
N: Número de nodos

RF N9ª ELIMINACION DE PRODUCTO

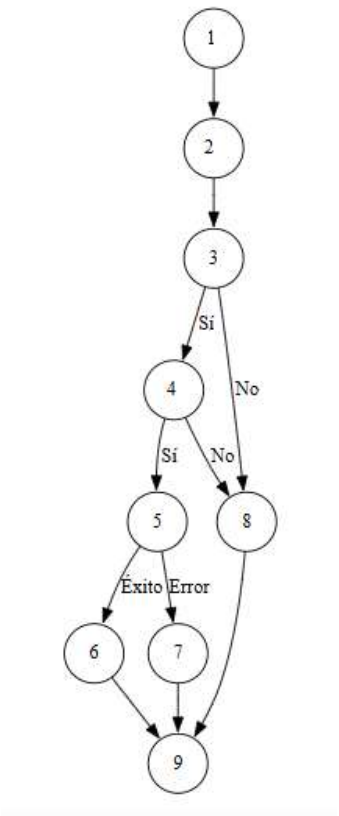
41. CÓDIGO FUENTE

```
28      Private Sub btnEliminar_Click(sender As Object, e As EventArgs) Handles btnEliminar.Click
29          ' Primera confirmación
30          Dim result = MessageBox.Show("¿Estás seguro de eliminar el producto '" & txtNombre.Text & "'?",
31                                       "Confirmar eliminación",
32                                       MessageBoxButtons.YesNo,
33                                       MessageBoxIcon.Warning)
34
35          If result = DialogResult.Yes Then
36              ' Segunda confirmación
37              Dim finalConfirm = MessageBox.Show("¡Esta acción no se puede deshacer! ¿Deseas eliminar este producto?",
38                                                 "Confirmación final",
39                                                 MessageBoxButtons.YesNo,
40                                                 MessageBoxIcon.Warning)
41              If finalConfirm = DialogResult.Yes Then
42                  Try
43                      Dim db = ConexionMongo.ObtenerConexion()
44                      Dim collection = db.GetCollection(Of BsonDocument)("Productos")
45
46                      Dim filtro = Builders(Of BsonDocument).Filter.Eq(Of ObjectId)("_id", New ObjectId(idProducto))
47                      collection.DeleteOne(filtro)
48
49                      MessageBox.Show("Producto eliminado correctamente.", "Éxito", MessageBoxButtons.OK, MessageBoxIcon.Information)
50                      Me.Close()
51
52                  Catch ex As Exception
53                      MessageBox.Show("Error al eliminar: " & ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
54                  End Try
55              End If
56          End If
57      End Sub
58
59      ' ===== BOTÓN CANCELAR =====
        0 referencias
60      Private Sub btnCancelar_Click(sender As Object, e As EventArgs) Handles btnCancelar.Click
61          Me.Close()
62      End Sub
63
        0 referencias
64      Private Sub EliminarProducto_Load(sender As Object, e As EventArgs) Handles MyBase.Load
65          ' Inicialización extra si necesitas
66      End Sub
```

42. DIAGRAMA DE FLUJO (DF)



43. GRAFO DE FLUJO (GF)



44. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

Ruta	Secuencia de nodos	Descripción
R1	1 → 2 → 3 → 8 → 9	Usuario cancela en la primera confirmación → fin del proceso
R2	1 → 2 → 3 → 4 → 8 → 9	Usuario confirma primero, pero cancela en la segunda confirmación → fin del proceso
R3	1 → 2 → 3 → 4 → 5 → 6 → 9	Usuario confirma ambas y eliminación es exitosa → fin del proceso
R4	1 → 2 → 3 → 4 → 5 → 7 → 9	Usuario confirma ambas y ocurre error al eliminar → fin del proceso

45. COMPLEJIDAD CICLOMÁTICA

La complejidad ciclomática permite determinar el número de caminos independientes del sistema.

Datos del grafo

- **N (Número de nodos):** 9
- **P (Número de nodos predicados):** 3
- **A (Número de aristas): 10**

Cálculo

Fórmula 1:
$V(G) = P + 1$
$V(G) = 3 + 1 = 4$

Fórmula 2:
$V(G) = A - N + 2$
$V(G) = 10 - 9 + 2 = 3$

Resultado
La complejidad ciclomática del REQ 9 es 7, lo que significa que existen **7 caminos básicos independientes** que debes probar para cobertura de caja blanca.

DONDE

P: Número de nodos predicado
A: Número de aristas
N: Número de nodos