

Prueba de Caja Blanca

“Urban Style Shop”

Integrantes:

Kevin Cañola
Alexandro Molina
Christian Marcalla

Fecha: 2025-06-16

REQUISITO 1:

Prueba caja blanca de describa el requisito funcional

REQUISITO 1. - Registro de Clientes

1. CÓDIGO FUENTE

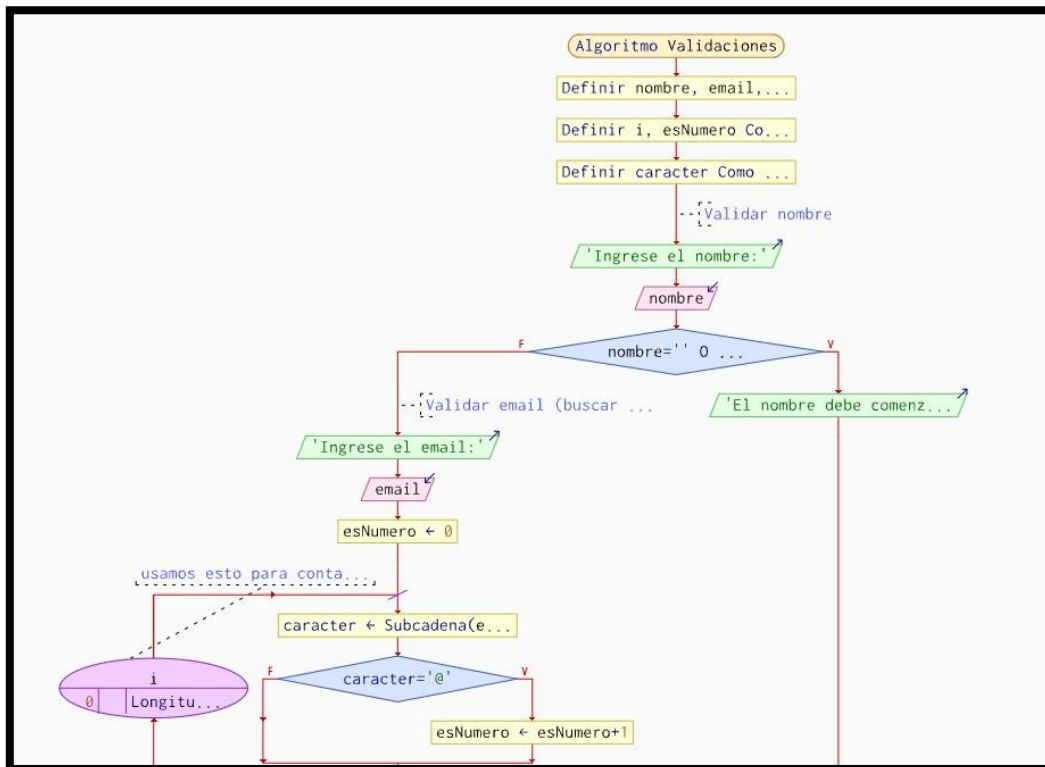
```
// Validar Nombre: primera letra mayúscula y no vacío
String nombre = txtNombre.getText().trim();
if (nombre.isEmpty() || !Character.isUpperCase(nombre.charAt(0))) {
    JOptionPane.showMessageDialog(this, "El nombre debe comenzar con mayúscula y no estar vacío.");
    txtNombre.requestFocus();
    return;
}

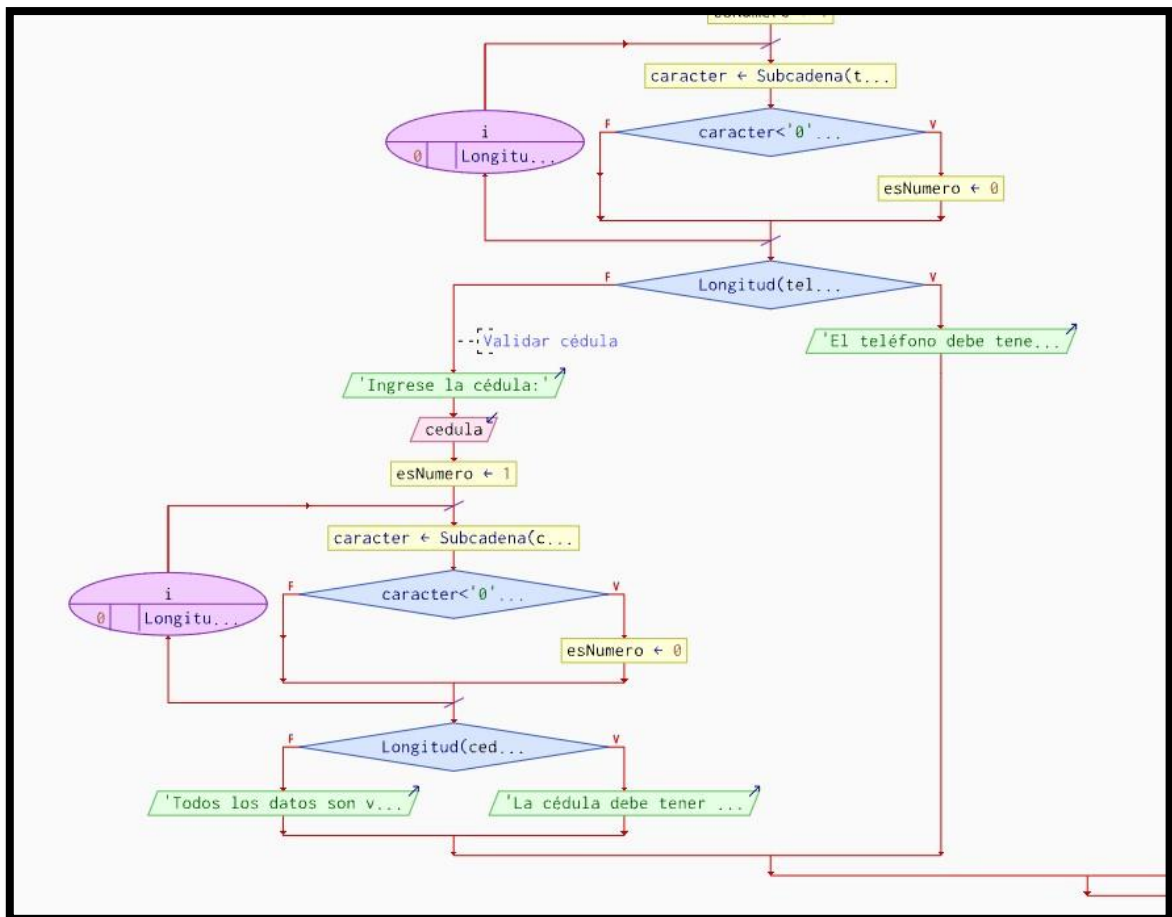
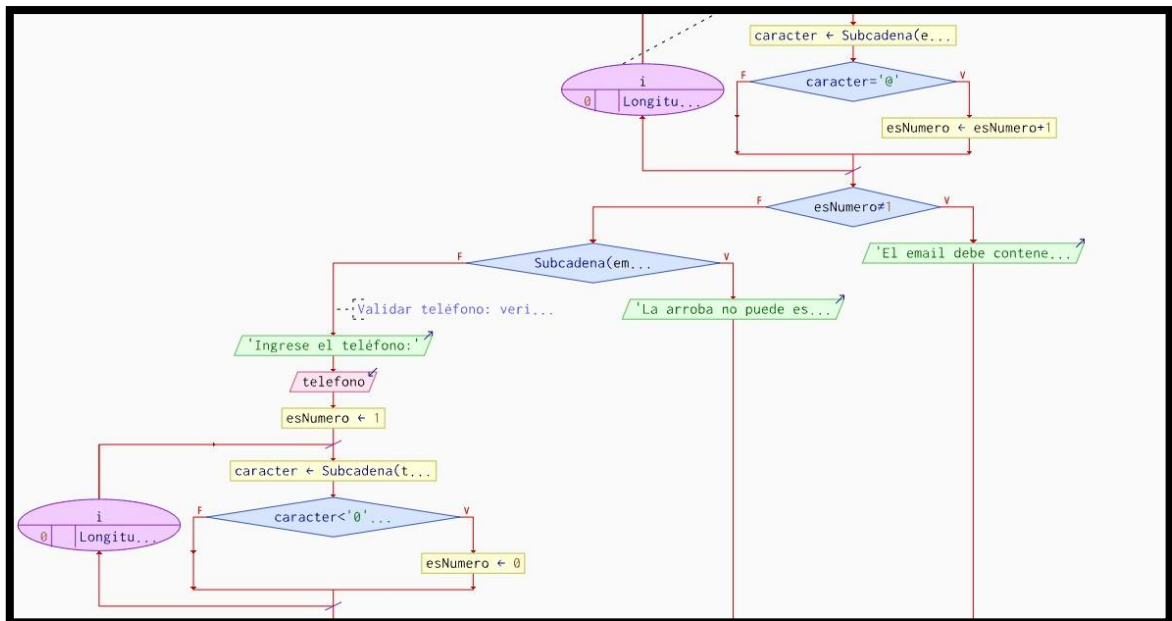
// Validar Email: debe contener una '@' válida
String email = txtEmail.getText().trim();
if (!email.contains("@") || email.startsWith("@") || email.endsWith("@")) {
    JOptionPane.showMessageDialog(this, "El email debe contener una '@' válida.");
    txtEmail.requestFocus();
    return;
}

// Validar Teléfono: máximo 10 dígitos y solo números
String telefono = txtTelefono.getText().trim();
if (telefono.length() > 10 || !telefono.matches("\\d+")) {
    JOptionPane.showMessageDialog(this, "El teléfono debe tener máximo 10 dígitos y solo números.");
    txtTelefono.requestFocus();
    return;
}

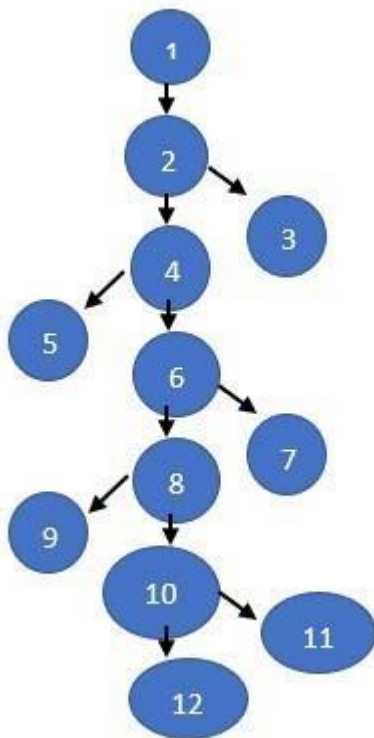
// Validar Cédula: debe tener exactamente 10 dígitos y solo números
String cedula = txtCedula.getText().trim();
if (cedula.length() != 10 || !cedula.matches("\\d+")) {
    JOptionPane.showMessageDialog(this, "La cédula debe tener exactamente 10 dígitos y solo números.");
    txtCedula.requestFocus();
    return;
}
```

2. DIAGRAMA DE FLUJO





3. GRAFO DE FLUJO (GF)



4. IDENTIFICACIÓN DE LAS RUTAS (Camino basico)

RUTAS

- R1: 1 → 2 → 3 → 4 → 5 → 6 → 9 → 11 → 12 → 14
R2: 1 → 2 → 3 → 7
R3: 1 → 2 → 3 → 4 → 5 → 8
R4: 1 → 2 → 3 → 4 → 5 → 6 → 9 → 10
R5: 1 → 2 → 3 → 4 → 5 → 6 → 9 → 11 → 12 → 13

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G)=P+1=5+1=6$
- $V(G)=A-N+2=17-12+2=7$

DONDE:

P: Número de nodos prediado

A: Número de aristas

N: Número de nodos

REQUISITO 2:

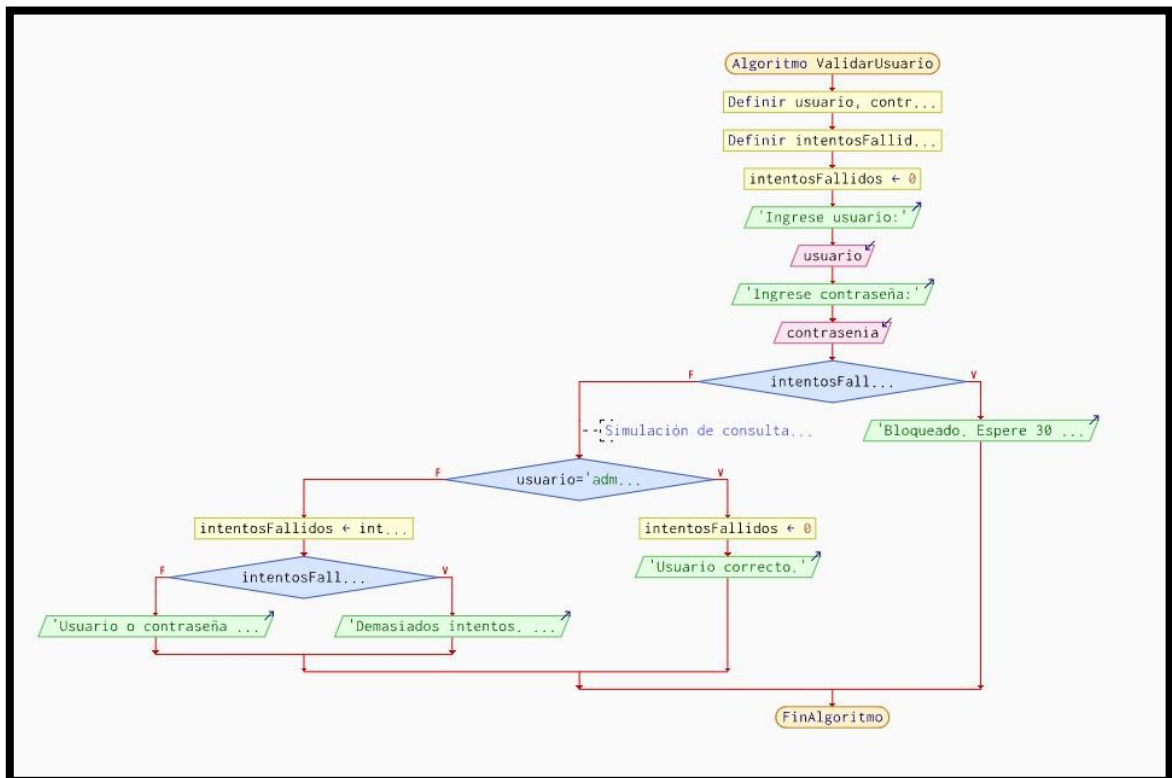
Prueba caja blanca de describa el requisito funcional

REQUISITO 1. - Registro del Administrador

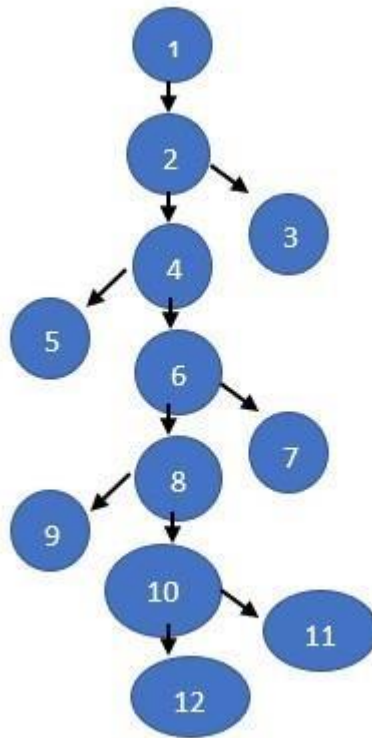
1. CÓDIGO FUENTE

```
lic ResultadoLogin validaUsuario(JTextField usuario, JPasswordField contrasenia) {  
    try {  
        if (intentosFallidos >= 3) {  
            return new ResultadoLogin(false, true, "Bloqueado. Espere 30 segundos.");  
        }  
  
        PreparedStatement ps;  
        ResultSet rs;  
  
        CConexion conexion = new CConexion();  
        String consulta = "SELECT * FROM usuarios WHERE BINARY ingresoUsuario = ? AND BINARY ingresoContrasena = ?";  
        ps = conexion.establishConexion().prepareStatement(consulta);  
  
        String contra = String.valueOf(contrasenia.getPassword());  
  
        ps.setString(1, usuario.getText().trim());  
        ps.setString(2, contra);  
  
        rs = ps.executeQuery();  
  
        if (rs.next()) {  
            intentosFallidos = 0;  
            return new ResultadoLogin(true, false, "Usuario correcto.");  
        } else {  
            intentosFallidos++;  
            if (intentosFallidos >= 3) {  
                return new ResultadoLogin(false, true, "Demasiados intentos. Bloqueado 30 segundos.");  
            } else {  
                return new ResultadoLogin(false, false, "Usuario o contraseña incorrectos. Intento " + intentosFallidos + " de 3.");  
            }  
        }  
    }  
}
```

2. DIAGRAMA DE FLUJO



3. GRAFO DE FLUJO (GF)



4. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

RUTAS

R1: 1 → 2 → 3 → 4 → 5 → 6 → 12

R2: 1 → 2 → 7

R3: 1 → 2 → 3 → 8

R4: 1 → 2 → 3 → 4 → 9

R5: 1 → 2 → 3 → 4 → 5 → 10

R6: 1 → 2 → 3 → 4 → 5 → 6 → 11

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

$$V(G) = P + 1$$

Donde P es el número de nodos de decisión (condiciones).

- En este grafo hay **5 nodos de decisión**:
 - ✦ ¿intentosFallidos >= 3?
 - ✦ ¿usuario y contraseña correctos?
 - ✦ ¿intentosFallidos >= 3?

$$V(G) = 5 + 1 = 6$$

$$V(G) = A - N + 2 \text{ Donde:}$$

- $A = 17$ (aristas),

- $N = 12$ (nodos).

$$V(G) = 17 - 12 + 2 = 7$$

P: Número de nodos prediado

A: Número de aristas

N: Número de nodos

REQUISITO 3:

Prueba caja blanca de describa el requisito funcional

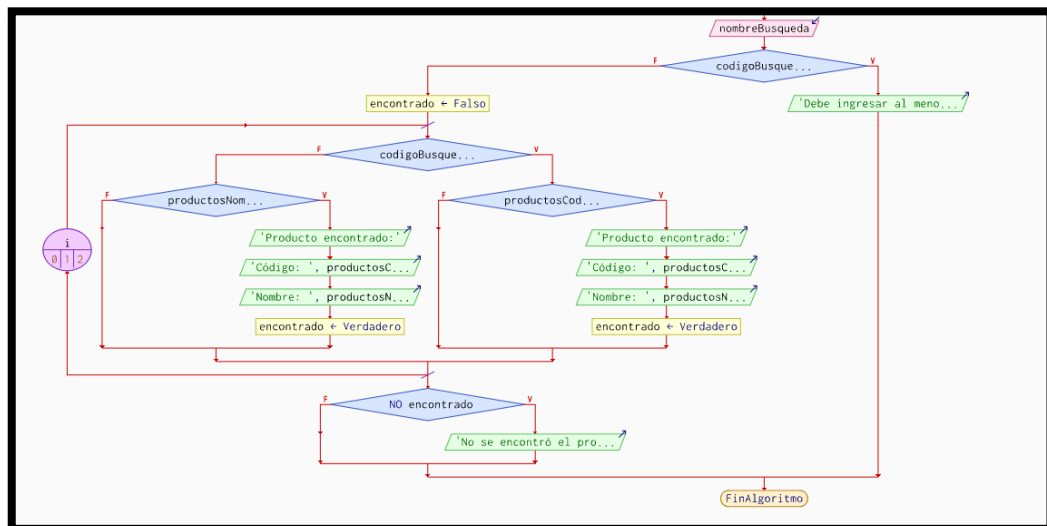
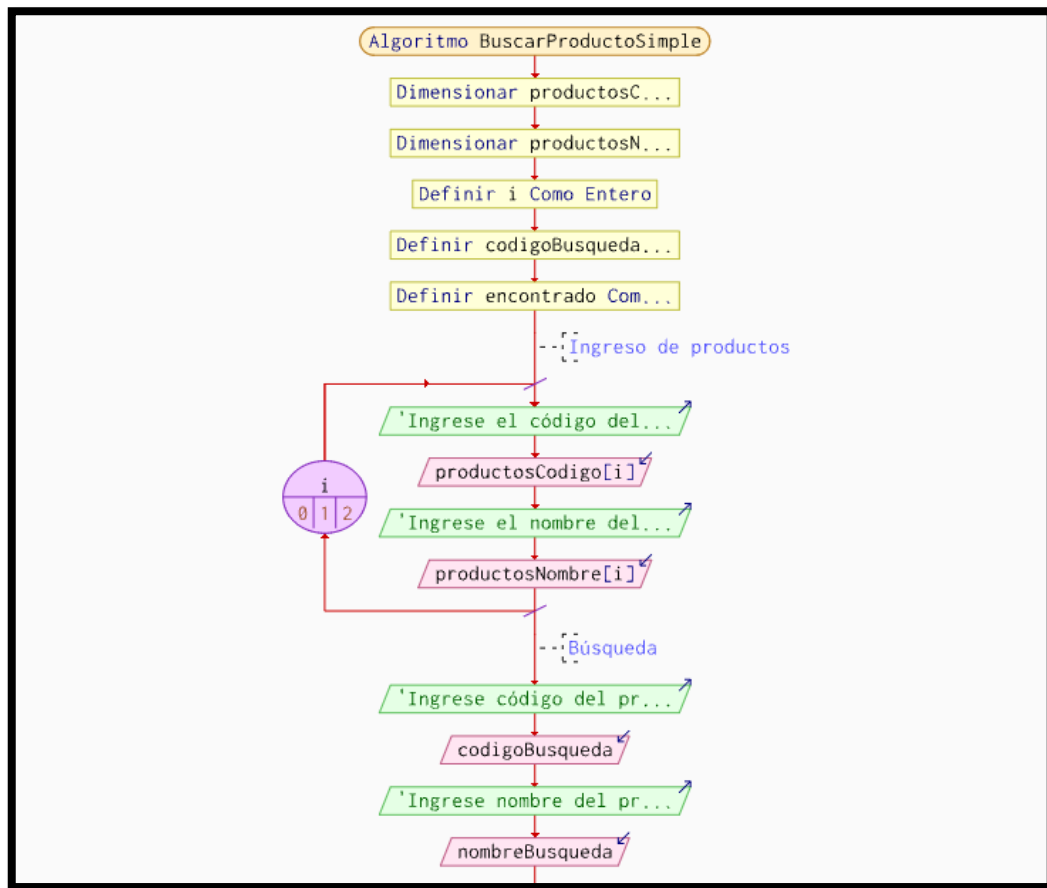
REQUISITO 3. - Sistema deficiente del control de stock

1. Código

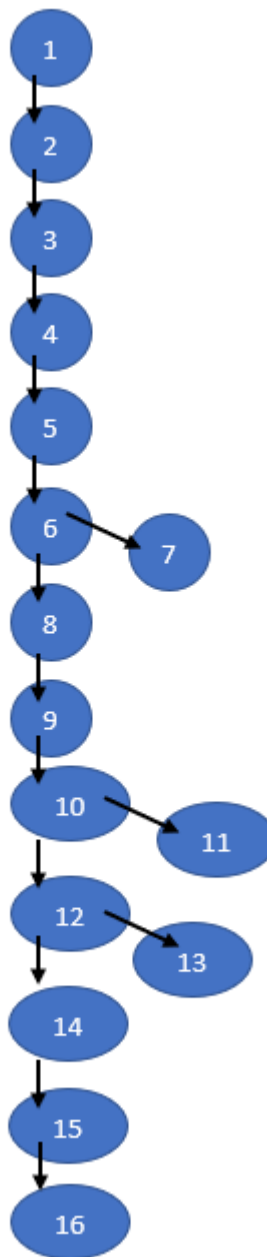
```
private void btnBusquedaActionPerformed(java.awt.event.ActionEvent evt) {  
    String codigoBusqueda = txtCodigo.getText().trim();  
    String nombreBusqueda = txtNombre.getText().trim();  
  
    if (codigoBusqueda.isEmpty() && nombreBusqueda.isEmpty()) {  
        JOptionPane.showMessageDialog(this, "Ingrese código o nombre para buscar.");  
        return;  
    }  
  
    try {  
        Connection con = new CCConexion().estableceConexion();  
  
        String sql = "";  
        PreparedStatement ps = null;  
  
        if (!codigoBusqueda.isEmpty()) {  
            // Buscar por código con distinción exacta mayúsculas/minúsculas  
            sql = "SELECT * FROM productos WHERE BINARY codigo = ?";  
            ps = con.prepareStatement(sql);  
            ps.setString(1, codigoBusqueda);  
        } else if (!nombreBusqueda.isEmpty()) {  
            // Buscar por nombre con distinción exacta mayúsculas/minúsculas  
            sql = "SELECT * FROM productos WHERE BINARY nombre = ?";  
            ps = con.prepareStatement(sql);  
            ps.setString(1, nombreBusqueda);  
        }  
    }  
}
```

```
ResultSet rs = ps.executeQuery();  
  
DefaultTableModel modelo = (DefaultTableModel) jTableaStock.getModel();  
modelo.setRowCount(0); // Limpiar tabla  
  
boolean encontrado = false;  
while (rs.next()) {  
    Object[] fila = new Object[]{  
        rs.getInt("id"),  
        rs.getString("codigo"),  
        rs.getString("nombre"),  
        rs.getBigDecimal("precio"),  
        rs.getInt("stock"),  
        rs.getDate("fecha")  
    };  
    modelo.addRow(fila);  
    encontrado = true;  
}  
  
if (!encontrado) {  
    JOptionPane.showMessageDialog(this, "No se encontró ningún producto con esos datos.");  
    // Opcional: mostrarProductos(); para volver a mostrar todo  
}  
  
con.close();  
  
} catch (Exception e) {  
    JOptionPane.showMessageDialog(this, "Error en la búsqueda: " + e.getMessage());  
}  
}
```

2. DIAGRAMA DE FLUJO



3. GRAFO DE FLUJO (GF)



4. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

RUTAS

R1: 1 → 2 → 3 → 4 → 5 → 6 → 7 → 17

R2: 1 → 2 → 3 → 4 → 5 → 6 → 8 → 9 → 10 → 11 → 12 → 15 → 16 → 17

R3: 1 → 2 → 3 → 4 → 5 → 6 → 8 → 9 → 10 → 13 → 14 → 15 → 16 → 17

R4: 1 → 2 → 3 → 4 → 5 → 6 → 8 → 9 → 15 → 16 → 17

R5: 1 → 2 → 3 → 4 → 5 → 6 → 8 → 9 → 10 → 11 → 15 → 16 → 17

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

$$V(G) = P + 1$$

Donde P es el número de nodos de decisión (condiciones).

- En este grafo hay **5 nodos de decisión**:
 - ✦ ¿intentosFallidos >= 3?
 - ✦ ¿usuario y contraseña correctos?
 - ✦ ¿intentosFallidos >= 3?

$$V(G) = 6 + 1 = 7$$

$$V(G) = A - N + 2 \text{ Donde:}$$

- **A = 18**
- **N = 13**

$$V(G) = 18 - 13 + 2 = 7$$

P: Número de nodos predicado

A: Número de aristas

N: Número de nodos