Prueba de Caja Blanca

"Sistema de Inventario"

Integrantes:

Robinson Estrella
Fenix Toapanta
Mathias Tapia

Fecha 2025 / 06 / 25

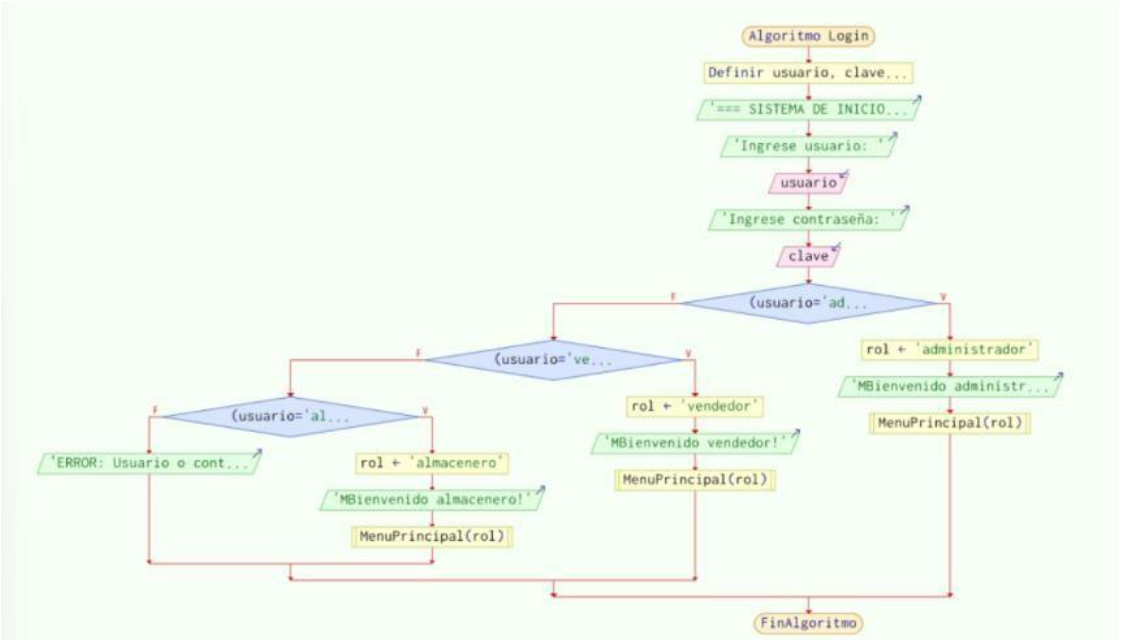
Prueba caja blanca de Inicio de sesión.

1. **CÓDIGO FUENTE**

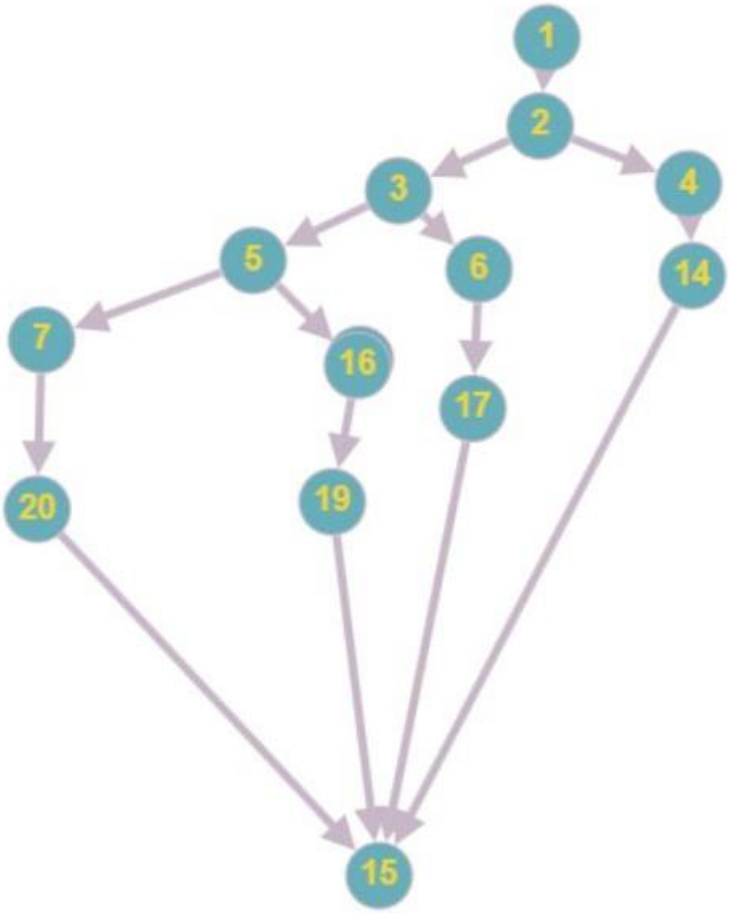
```
        if(us.getUsaurio()!=null && us.getPassword()!=null)
        {
            MenuPrincipal m=new MenuPrincipal();
//Administrador Vendedor Almacenero
          if(us.getTipoUsuario().equals("Vendedor")){...9 lines }else if(us.getTipoUsuario().equals("Alma
            MenuPrincipal.txtiduser.setText(us.getIdusuario()+"");
            MenuPrincipal.txtuser.setText(us.getUsaurio());
            Categorias.iduser=us.getIdusuario();
            Notification panel = new Notification(this, Notification.Type.SUCCESS, Notification.Location.T
            panel.showNotification();
            m.setVisible(true);
            dispose();
        }else{
            //JOptionPane.showMessageDialog(null, "Aceso denegado");
            Notification panel = new Notification(this, Notification.Type.ERROR, Notification.Location.TOP
            panel.showNotification();
        }
    }

    private void btnVerPasswordActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        if(btnVerPassword.isSelected()){
           txtpassword.setEchoChar((char)0);
        }else{
```

2. **DIAGRAMA DE FLUJO (DF)**



3. GRAFO DE FLUJO (GF)



4. IDENTIFIACCIÒN DE LAS RUTAS (Camino basico)

RUTAS

R1: 1 , 2 , 4 , 14 , 15
R2: 1 , 2 , 3 , 6 , 17 , 15
R3: 1 , 2 , 3 , 5 , 16 , 19 , 15
R4: 1 , 2 , 3 , 5 , 7 , 20 , 15

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)= 3 + 1 = 4

- V(G) = A – N + 2
  V(G)= 15 – 13 + 2 = 4

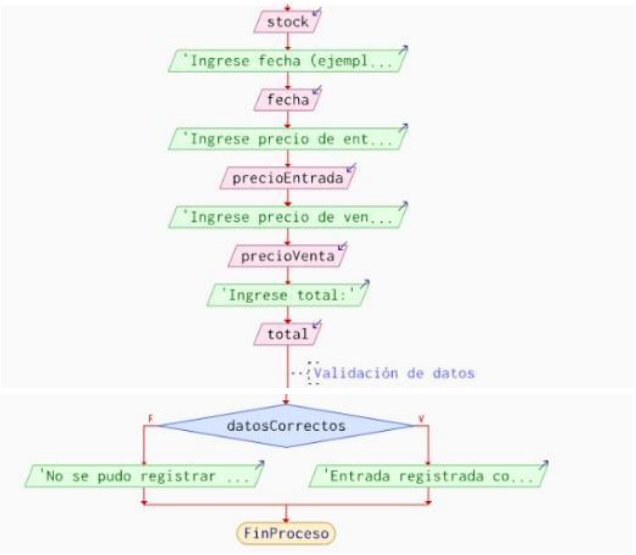
DONDE:
P: Número de nodos predicado
A: Número de aristas
N: Número de nodos

Prueba caja blanca de registro de productos

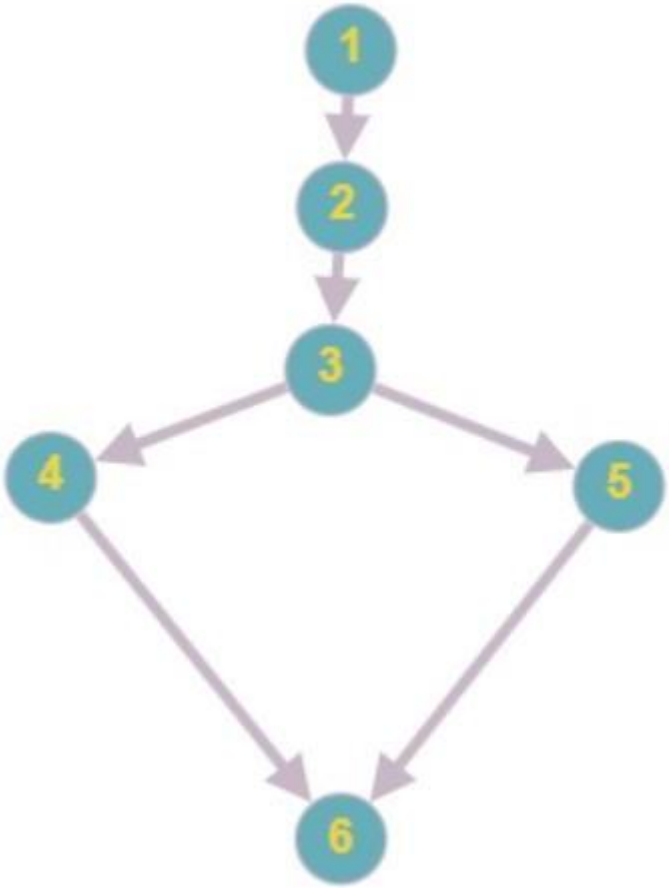
1. CÓDIGO FUENTE

```
void agregarEntrada(){
        Calendar cal;
        int d,m,a;
        cal=dateFecha.getCalendar();
        d=cal.get(Calendar.DAY_OF_MONTH);
        m=cal.get(Calendar.MONTH);
        a=cal.get(Calendar.YEAR)-1900;
        e.setIdproducto(Integer.parseInt(txtidProducto.getText()));
        e.setStock(Integer.parseInt(txtstock.getText()));
        e.setFecha(new Date(a,m,d));
        e.setIdproveedor(Integer.parseInt(txtidproveedor.getText()));
        e.setPrecioE(Double.parseDouble(txtprecioE.getText()));
        e.setPrecioV(Double.parseDouble(txtprecioV.getText()));
        e.setTotal(Double.parseDouble(txtTotal.getText()));
        p.setIdproducto(Integer.parseInt(txtidProducto.getText()));
        p.setPrecioV(Double.parseDouble(txtprecioV.getText()));
        if(dao.insertar(e)&&daoPr.sumarStock(Integer.parseInt(txtidProducto.getText()), Integer.parseInt
            MenuPrincipal menu=new MenuPrincipal();
            menu.exito("Entrada Registrada Con Exito");
        }else{
            MenuPrincipal menu=new MenuPrincipal();
            menu.error("No se pudo registrar la entrada");
        }
}
```

2. DIAGRAMA DE FLUJO (DF)



3. GRAFO DE FLUJO (GF)



4. IDENTIFIACCIÒN DE LAS RUTAS (Camino basico)

RUTAS

R1: 1 , 2 , 3 , 5 , 6
R2: 1 , 2 , 3 , 4 , 6

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)= 1 + 1 = 2

- V(G) = A – N + 2
  V(G)= 6 – 6 +2 = 2

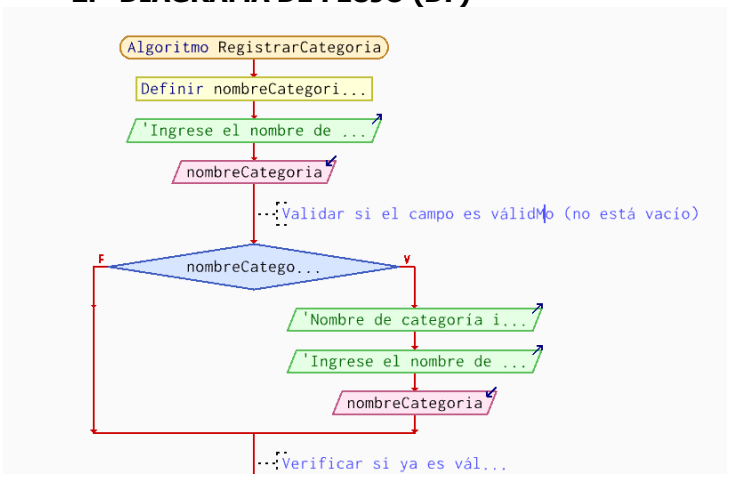
DONDE:
P: Número de nodos predicado
A: Número de aristas
N: Número de nodos

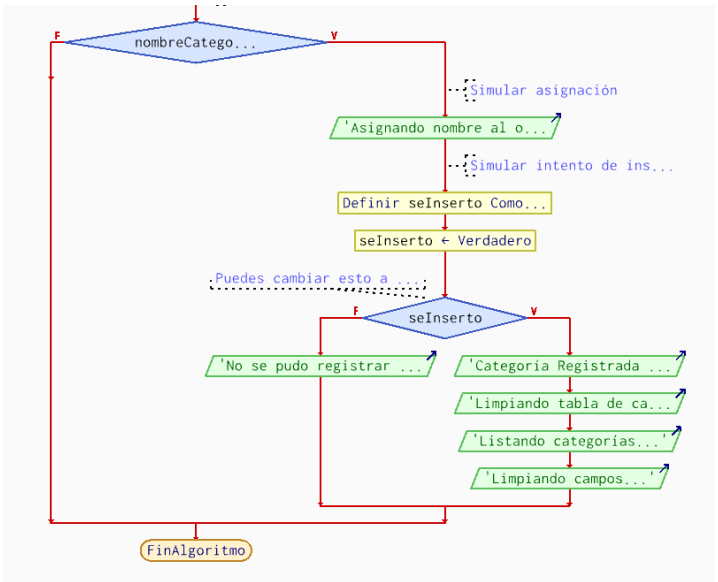
Prueba caja blanca de registro de categorías

1. CÓDIGO FUENTE

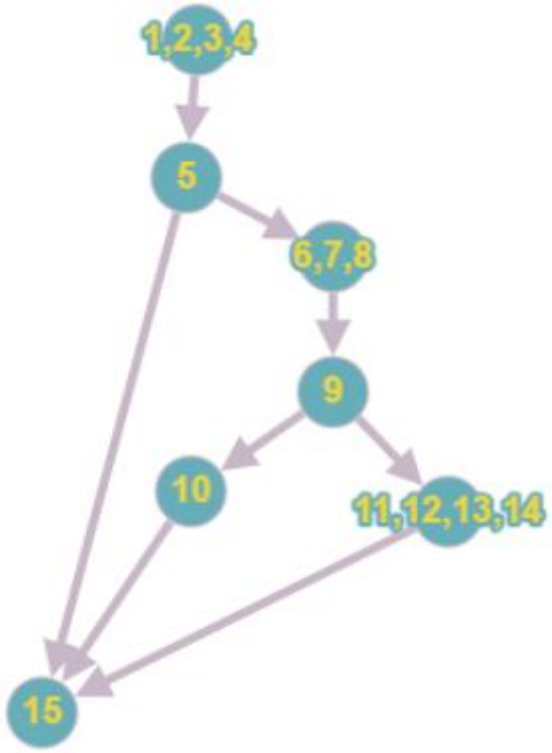
```
269     private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
270         String nombreCategoria = txtnomCategoria.getText().trim();
271 
272     // Validar que solo contenga letras y espacios (opcional)
273     if (!nombreCategoria.matches("^[a-zA-ZáéíóúÁÉÍÓÚñÑ\\s]+$")) {
274         m.error("El nombre de categoría solo puede contener letras");
275         return; // Salir del método si la validación falla
276     }
277 
278     ct.setNomCategoria(nombreCategoria);
279     if(daoCt.insertar(ct)){
280         m.exito("Categoria Registrada Con Exito");
281     }else{
282         m.error("No se pudo registrar la Categoria");
283     }
284     limpiarTablaCategoria();
285     listarCategorias();
286     limpiarCampos();
287     }
```

2. DIAGRAMA DE FLUJO (DF)





3. GRAFO DE FLUJO (GF)



4. IDENTIFIACCIÒN DE LAS RUTAS (Camino básico)

RUTAS

R1: 1 , 2 , 3, 4 , 5 , 6 , 7, 8 , 9 , 11 , 12 , 13, 14, 15
R2: 1 , 2 , 3 , 5 , 6 , 7, 8, 9, 10, 15
R3: 1 , 2 , 3 , 4, 5, 15

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)= 2 + 1 = 3

- V(G) = A – N + 2
  V(G)= 8 – 7 +2 = 3

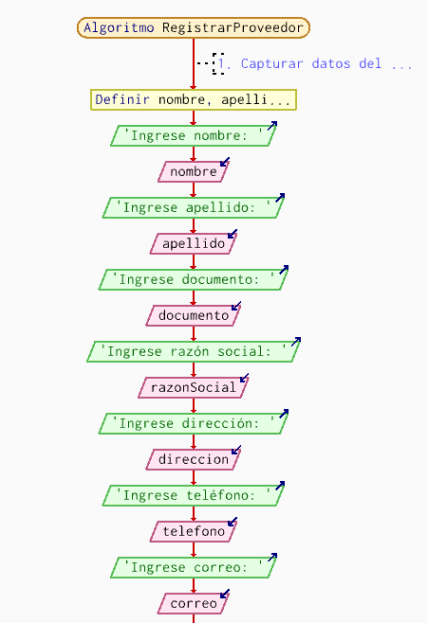
DONDE:
P: Número de nodos predicado
A: Número de aristas
N: Número de nodos

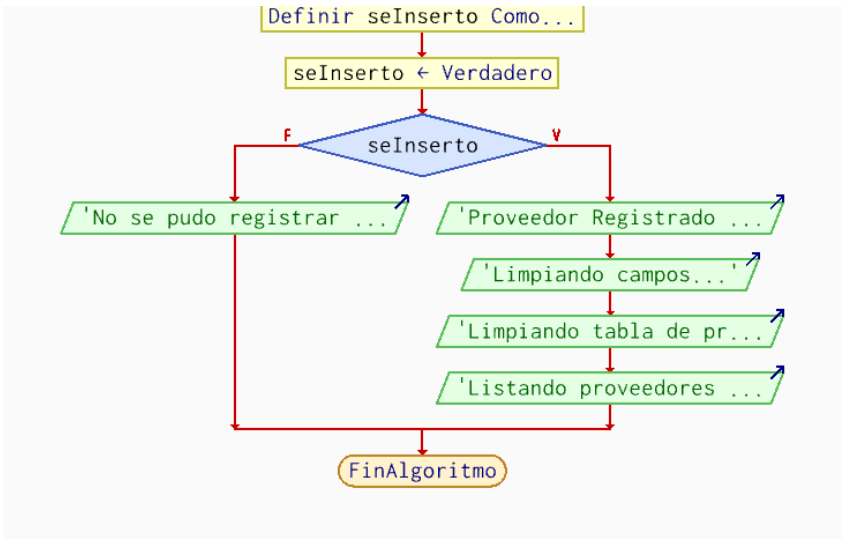
Prueba caja blanca de registro de proveedores

1. CÓDIGO FUENTE

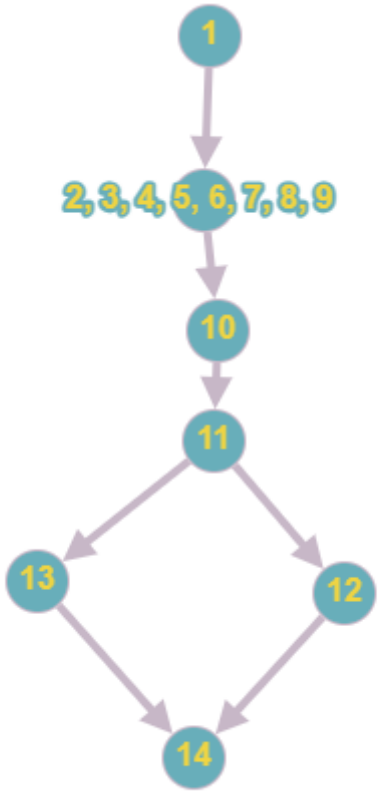
```
365     private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
366     // Validar campos primero
367     if (!validarCamposProveedor()) {
368         return;
369     }
370
371     // Si pasa las validaciones, proceder a guardar
372     p.setNombre(txtnombre.getText().trim());
373     p.setApellido(txtapellido.getText().trim());
374     p.setDocumento(txtdocumento.getText().trim());
375     p.setRsocial(txtRsocial.getText().trim());
376     p.setDireccion(txtdireccion.getText().trim());
377     p.setTelefono(txttelefono.getText().trim());
378     p.setCorreo(txtcorreo.getText().trim());
379
380     if(dao.insertar(p)) {
381         MenuPrincipal m = new MenuPrincipal();
382         m.exito("Proveedor Registrado Con Éxito");
383         limpiarCampos();
384         limpiarTablaProveedor();
385         listarProveedor();
386     } else {
387         MenuPrincipal m = new MenuPrincipal();
388         m.error("No se pudo registrar el Proveedor");
389     }
390 }
391
```

2. DIAGRAMA DE FLUJO (DF)





3. GRAFO DE FLUJO (GF)



4. IDENTIFIACCIÒN DE LAS RUTAS (Camino basico)

RUTAS

R1: 1 , 2 , 3 , 5 , 6 , 7 , 8 , 9 , 10 , 11, 12 , 14
R2: 1 , 2 , 3 , 5 , 6 , 7 , 8 , 9 , 10 , 11, 13 , 14

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)= 1 + 1 = 2

- V(G) = A – N + 2
  V(G)= 7 – 7 + 2 = 2

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

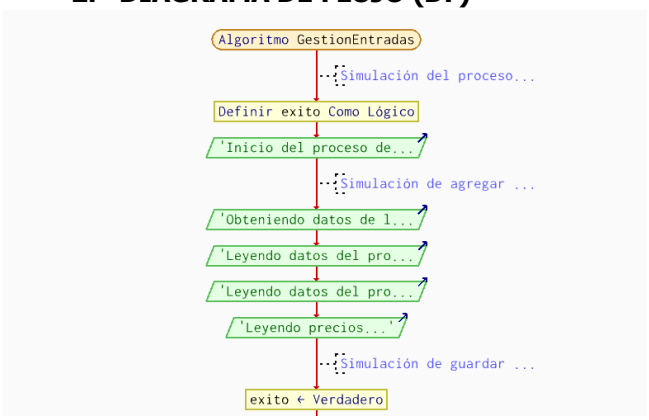
Prueba caja blanca de ingreso de entradas

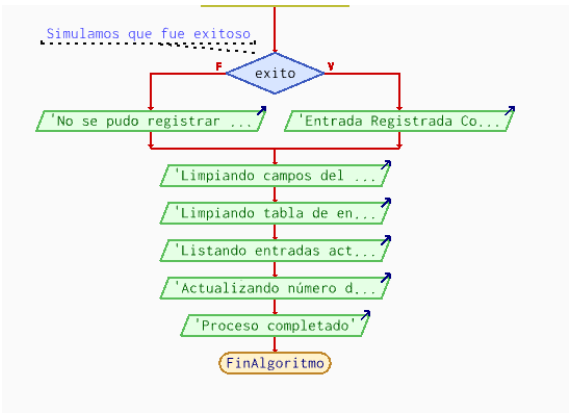
1. CÓDIGO FUENTE

```
    private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        agregarEntrada();
        limpiarCampos();
        limpiarTablaEntradas();
        listarEntradas();
        numProducto();
    }
```

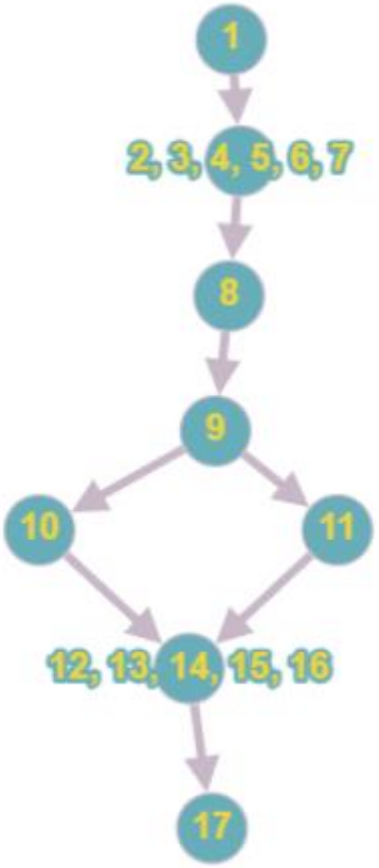
```
776  void agregarEntrada(){
777          Calendar cal;
778          int d,m,a;
779          cal=dateFecha.getCalendar();
780          d=cal.get(Calendar.DAY_OF_MONTH);
781          m=cal.get(Calendar.MONTH);
782          a=cal.get(Calendar.YEAR)-1900;
783          e.setIdproducto(Integer.parseInt(txtidProducto.getText()));
784          e.setStock(Integer.parseInt(txtstock.getText()));
785          e.setFecha(new Date(a,m,d));
786          e.setIdproveedor(Integer.parseInt(txtidproveedor.getText()));
787          e.setPrecioE(Double.parseDouble(txtprecioE.getText()));
788          e.setPrecioV(Double.parseDouble(txtprecioV.getText()));
789          e.setTotal(Double.parseDouble(txtTotal.getText()));
790          p.setIdproducto(Integer.parseInt(txtidProducto.getText()));
791          p.setPrecioV(Double.parseDouble(txtprecioV.getText()));
792          if(dao.insertar(e)&&daoPr.sumarStock(Integer.parseInt(txtidProducto.getText()), Integer.parseInt(txtstock.getText()))&&dac
793              MenuPrincipal menu=new MenuPrincipal();
794              menu.exito("Entrada Registrada Con Exito");
795          }else{
796              MenuPrincipal menu=new MenuPrincipal();
797              menu.error("No se pudo registrar la entrada");
798          }
799  }
```

2. DIAGRAMA DE FLUJO (DF)





3. GRAFO DE FLUJO (GF)



4. IDENTIFIACCIÒN DE LAS RUTAS (Camino basico)

RUTAS

R1: 1 , 2 , 3 , 4, 5 , 6 , 7 , 8 , 9 , 10 , 12, 13 , 14 , 15 , 16 , 17
R2: 1 , 2 , 3 , 4, 5 , 6 , 7 , 8 , 9 , 11 , 12, 13 , 14 , 15 , 16 , 17

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)= 1 + 1 = 2

- V(G) = A – N + 2
  V(G)= 8 – 8 + 2 = 2

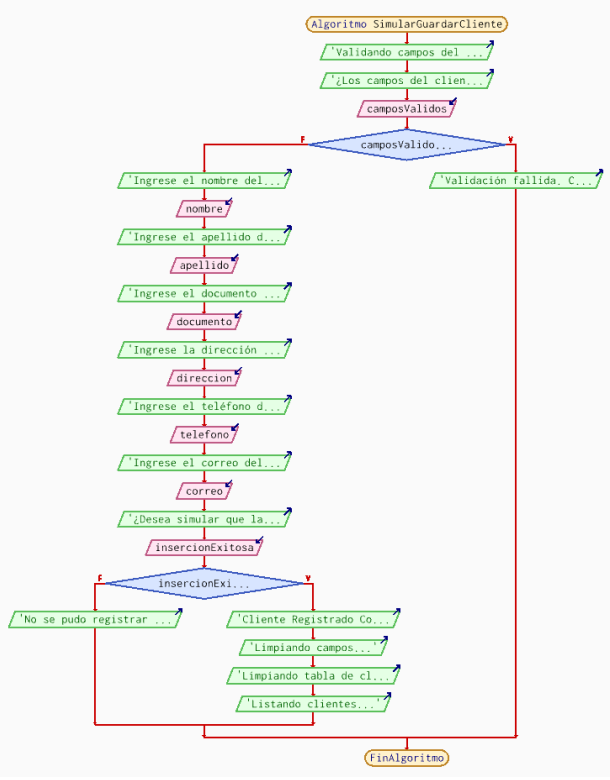
DONDE:
P: Número de nodos predicado
A: Número de aristas
N: Número de nodos

Prueba caja blanca de registro de clientes

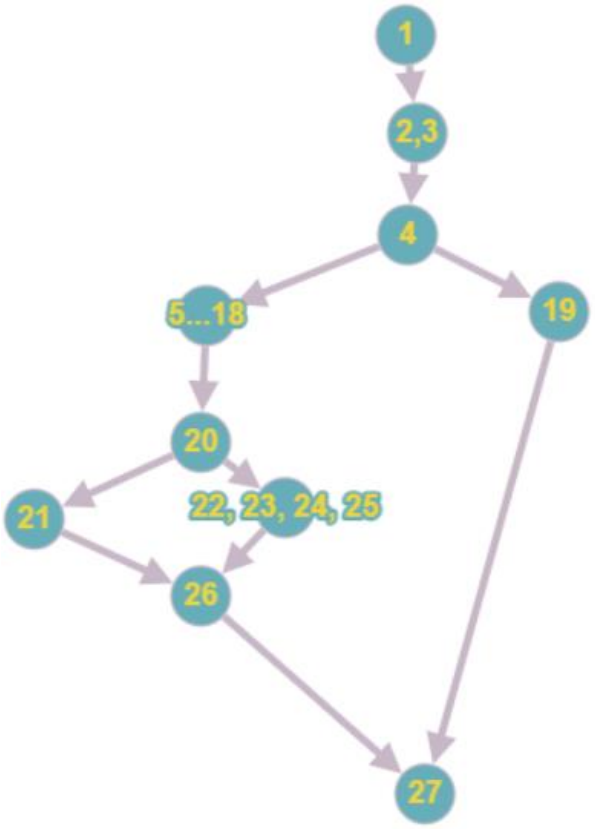
1. CÓDIGO FUENTE

```
330     private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
331     // Validar campos primero
332     if (!validarCamposCliente()) {
333         return;
334     }
335 
336     // Si pasan las validaciones, proceder a guardar
337     c.setNombre(txtnombre.getText().trim());
338     c.setApellido(txtapellido.getText().trim());
339     c.setDocumento(txtdocumento.getText().trim());
340     c.setDireccion(txtdireccion.getText().trim());
341     c.setTelefono(txttelefono.getText().trim());
342     c.setCorreo(txtcorreo.getText().trim());
343 
344     if(dao.insertar(c)) {
345         new MenuPrincipal().exito("Cliente Registrado Con Éxito");
346         limpiarCampos();
347         limpiarTablaCLientes();
348         listarClientes();
349     } else {
350         new MenuPrincipal().error("No se pudo registrar el Cliente");
351     }
352 }
```

2. DIAGRAMA DE FLUJO (DF)



3. GRAFO DE FLUJO (GF)



4. IDENTIFIACCIÒN DE LAS RUTAS (Camino basico)

RUTAS

R1: 1 , 2 , 3 , 4, 5 , 6 , 7 , 8 , 9 , 10 , 11, 12, 13, 14, 15, 16, 17, 18, 20, 21 , 26, 27
R2: 1 , 2 , 3 , 4, 19, 27
R3: 1, 2, 3, 4, 5, 6, 7, 8 , 9 , 10 , 11, 12, 13, 14, 15, 16, 17, 18, 20, 22, 23, 25, 26, 27

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)= 2 + 1 = 3

- V(G) = A – N + 2
  V(G)= 11 – 10 + 2 = 3

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

Prueba caja blanca de registro de salidas

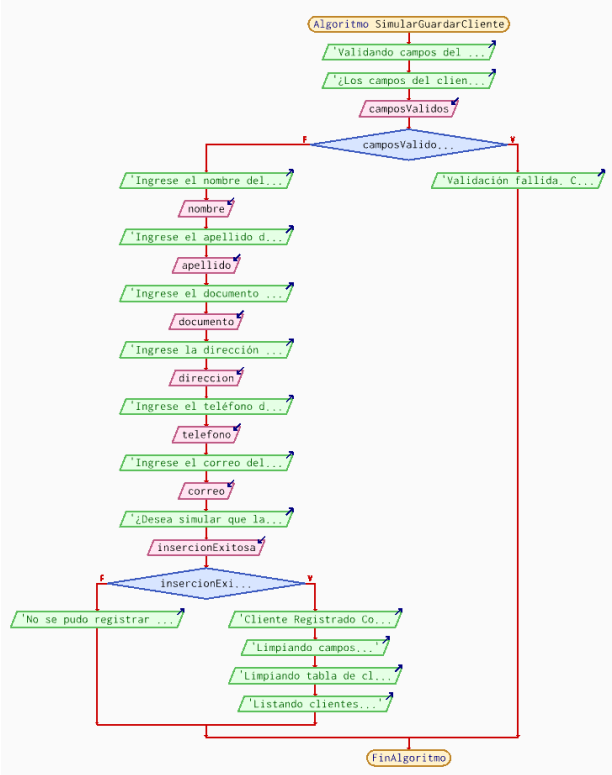
1. **CÓDIGO FUENTE**

```
627     private void btnAgregarActionPerformed(java.awt.event.ActionEvent evt) {
628     // Validar campos primero
629     if (!validarCamposSalida()) {
630         return;
631     }
632
633     // Verificar si el producto ya está agregado
634     for(int i = 0; i < tablaSalidas.getRowCount(); i++) {
635         if(tablaSalidas.getValueAt(i, 1).toString().equals(txtidProducto.getText())) {
636             new MenuPrincipal().advertencia("El Producto ya está agregado");
637             modelo.removeRow(i);
638         }
639     }
640
641     agregaEntrada();
642     limpiarDatosPod();
643     sumarTotal();
644 }
```

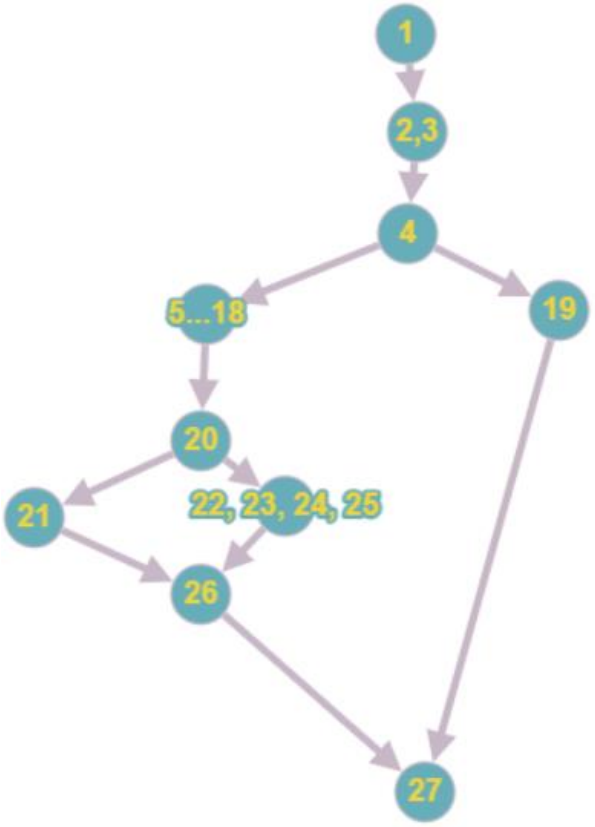
```
781     private void agregaEntrada(){
782         double precio,total,importe;
783         modelo=(DefaultTableModel) tablaSalidas.getModel();
784         int idEntrada=Integer.parseInt(txtidProducto.getText());
785         int idSalida=Integer.parseInt(txtnsalida.getText());
786         int cantidad=Integer.parseInt(txtcantidad.getText());
787         precio=Double.parseDouble(txtprecio.getText());
788         String prod=txtproducto.getText();
789         importe=cantidad*precio;
790         int stock=Integer.parseInt(txtstock.getText());
791         ArrayList lista=new ArrayList();
792         if(stock>0 && cantidad<=stock){
793             lista.add(idSalida);
794             lista.add(idEntrada);
795             lista.add(prod);
796             lista.add(precio);
797             lista.add(cantidad);
798             lista.add(importe);
799             Object[] ob=new Object[6];
800             ob[0]=lista.get(0);
801             ob[1]=lista.get(1);
802             ob[2]=lista.get(2);
803             ob[3]=lista.get(3);
804             ob[4]=lista.get(4);
805             ob[5]=lista.get(5);
806             modelo.addRow(ob);
807             tablaSalidas.setModel(modelo);
808         }else{
809             MenuPrincipal m=new MenuPrincipal();
810             m.error("Stock Insuficiente");
811         }
812     }
```

```java
    private void btnGenerarActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        s.setNumSalida(txtnsalida.getText());
        s.setIdCliente(Integer.parseInt(txtidcliente.getText()));
        s.setSubtotal(Double.parseDouble(txtsubtotal.getText()));
        s.setIgv(Double.parseDouble(txtigv.getText()));
        s.setTotal(Double.parseDouble(txtTotal.getText()));
        Calendar cal;
        int d,m,a;
        cal=jcFecha.getCalendar();
        d=cal.get(Calendar.DAY_OF_MONTH);
        m=cal.get(Calendar.MONTH);
        a=cal.get(Calendar.YEAR)-1900;
        s.setFecha(new Date(a,m,d));
        if(daoS.insertar(s)){
            guardarDetalle();
            MenuPrincipal m1=new MenuPrincipal();
            m1.exito("Salida Registrada Con Exito");
            restaStock();
            GenerarPDF(txtnsalida.getText());
            numSalida();
            limpiarDatosPod();
            limpaDatosCliente();
            txtTotal.setText("");
            txtsubtotal.setText("");
            txtigv.setText("");
            limpiarTablaSalida();
        }else{
            MenuPrincipal m2=new MenuPrincipal();
            m2.error("No se pudo registrar la Salida");
        }
    }
```

2. DIAGRAMA DE FLUJO (DF)



3. GRAFO DE FLUJO (GF)



4. IDENTIFIACCIÒN DE LAS RUTAS (Camino basico)

RUTAS

R1: 1 , 2 , 3 , 4, 5 , 6 , 7 , 8 , 9 , 10 , 11, 12, 13, 14, 15, 16, 17, 18, 20, 21 , 26, 27
R2: 1 , 2 , 3 , 4, 19, 27
R3: 1, 2, 3, 4, 5, 6, 7, 8 , 9 , 10 , 11, 12, 13, 14, 15, 16, 17, 18, 20, 22, 23, 25, 26, 27

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)= 2 + 1 = 3

- V(G) = A – N + 2
  V(G)= 11 – 10 + 2 = 3

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