

Prueba de Caja Blanca

“Panadería Asistencias”

Integrantes: Gerald Astudillo, Henry Chalcualan ,Isaac Erazo ,Henry Suin

V1.0

Fecha: 2025-01-09

INDICE

Prueba caja blanca de Inicio de Sesión	4
1. CÓDIGO FUENTE	4
2. DFD	5
3. GRAFO DE FLUJO (GF)	6
4. IDENTIFIACCIÓN DE LAS RUTAS (Camino básico)	6
Prueba caja blanca de Registro de Usuarios	7
1. CÓDIGO FUENTE	7
2. DFD	8
3. GRAFO DE FLUJO (GF)	8
4. IDENTIFIACCIÓN DE LAS RUTAS (Camino básico)	9
Prueba caja blanca de Registrar Asistencia	10
1. CÓDIGO FUENTE	10
2. DFD	12
3. GRAFO DE FLUJO (GF)	13
4. IDENTIFIACCIÓN DE LAS RUTAS (Camino básico)	13
Prueba caja blanca de Administrar usuarios	14
1. CÓDIGO FUENTE	14
2. DFD	15
3. GRAFO DE FLUJO	16
4. IDENTIFIACCIÓN DE LAS RUTAS (Camino básico)	16
Prueba caja blanca de Historial de asistencias	17
1. CÓDIGO FUENTE	17
2. DFD	19
3. GRAFO DE FLUJO	20
4. IDENTIFIACCIÓN DE LAS RUTAS (Camino básico)	20

Historial de Revisión

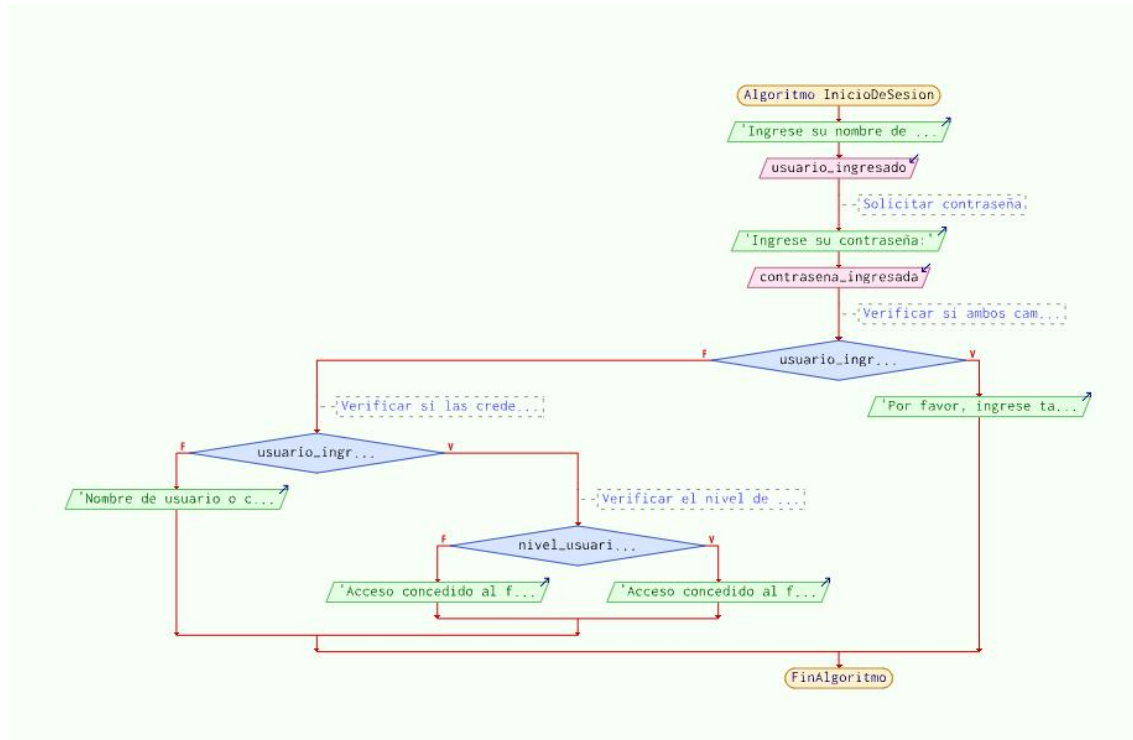
Fecha	Versión	Descripción	Autores
19/Noviembre/2024	1	Versión inicial	Gerald Astudillo Henry Chalcualan Lenin Erazo Henry Suin
11/Diciembre/2024	2	Correcciones de objetivos general y específico	Gerald Astudillo Henry Chalcualan Lenin Erazo Henry Suin
18/Diciembre/2024	3	Se añaden secciones 5, 6 y 7	Gerald Astudillo Henry Chalcualan Lenin Erazo Henry Suin
18/Enero/2025	4	Se añaden secciones 8, 9 y 10	Gerald Astudillo Henry Chalcualan Lenin Erazo Henry Suin
02/Febrero/2025	5	Correcciones finales según aclaraciones por el docente	Gerald Astudillo Henry Chalcualan Lenin Erazo Henry Suin

Prueba caja blanca de Inicio de Sesión

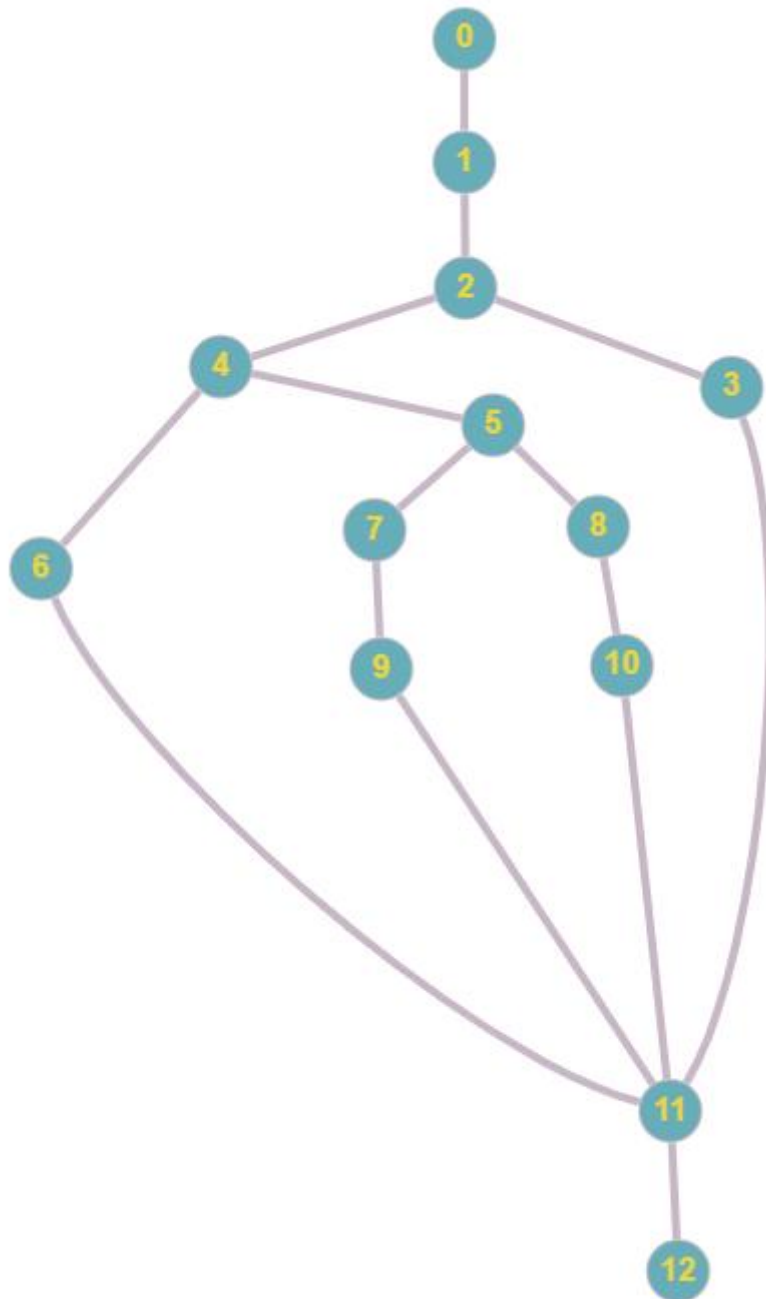
1. CÓDIGO FUENTE

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {  
    usuario = user.getText();  
    contraseña = new String(pass.getPassword());  
    if(usuario.equals("")||contraseña.equals("")){  
        JOptionPane.showMessageDialog(null,"Llenar todos los campos");  
    }  
    else{  
        try{  
            PreparedStatement ps=cn.prepareStatement("SELECT nivel FROM usuarios WHERE  
cedula='"+usuario+"' AND contraseña='"+contraseña+"'");  
            ResultSet rs=ps.executeQuery();  
            if(rs.next()){  
                String nivel=rs.getString("nivel");  
                if(nivel.equalsIgnoreCase("admin")){  
                    admin fa= new admin();  
                    fa.setVisible(true);  
                    this.setVisible(false);  
                }  
                else if(nivel.equalsIgnoreCase("usuario")){  
                    usuario fu= new usuario();  
                    fu.setVisible(true);  
                    this.setVisible(false);  
                }  
            }  
        }  
        else{  
            JOptionPane.showMessageDialog(null, "Usuario o contraseña incorrectos");  
        }  
    }catch(Exception e){  
        JOptionPane.showMessageDialog(null, e);  
    }  
}  
  
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
}
```

2. DFD



3. GRAFO DE FLUJO (GF)



4. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

RUTAS

R1: $1 \rightarrow 2 \rightarrow 3 \rightarrow 11 \rightarrow \text{Fin}$

R2: $1 \rightarrow 2 \rightarrow 4 \rightarrow 6 \rightarrow 11 \rightarrow \text{Fin}$

R3: $1 \rightarrow 2 \rightarrow 4 \rightarrow 5 \rightarrow 7 \rightarrow 9 \rightarrow 11 \rightarrow \text{Fin}$

R4: $1 \rightarrow 2 \rightarrow 4 \rightarrow 5 \rightarrow 8 \rightarrow 10 \rightarrow 11 \rightarrow \text{Fin}$

COMPLEJIDAD CICLOMÁTICA

Fórmula 1: $V(G)=P+1$

$P=3$ (nodos de decisión).

$V(G)=3+1=4$

Fórmula 2: $V(G)=A-N+2$

A=13 (aristas).
N=11(nodos).
 $V(G)=13-11+2=4$

Prueba caja blanca de Registro de Usuarios

1. CÓDIGO FUENTE

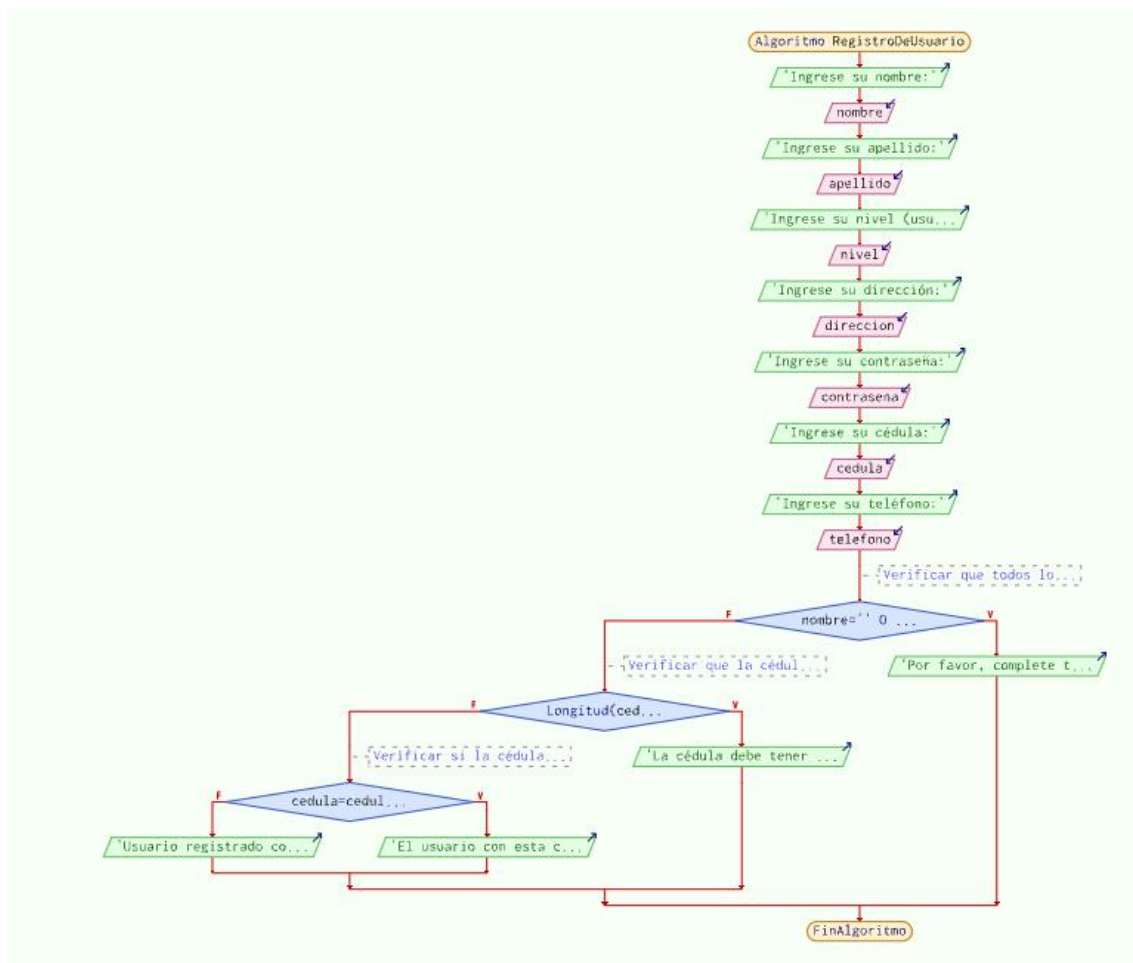
```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    String nombre,apellido,nivel,direc,contraseña,ceduV;
    int cedu,celular;
    nombre = nom.getText();
    apellido=apell.getText();
    nivel=level.getSelectedItem().toString();
    direc=jTextField6.getText();
    contraseña=jTextField1.getText();
    ceduV = jTextField4.getText();
    if (nombre.isEmpty()) {
        JOptionPane.showMessageDialog(null, "Falta ingresar el nombre.");
    } else if (apellido.isEmpty()) {
        JOptionPane.showMessageDialog(null, "Falta ingresar el apellido.");
    } else if (ceduV.isEmpty()) {
        JOptionPane.showMessageDialog(null, "Falta ingresar la cédula.");
    } else if (jTextField5.getText().isEmpty()) {
        JOptionPane.showMessageDialog(null, "Falta ingresar el teléfono.");
    } else if (direc.isEmpty()) {
        JOptionPane.showMessageDialog(null, "Falta ingresar la dirección.");
    } else if (contraseña.isEmpty()) {
        JOptionPane.showMessageDialog(null, "Falta ingresar la contraseña.");
    }
    }
    else if (ceduV.length() != 10) {
        JOptionPane.showMessageDialog(null, "La cédula debe tener exactamente 10 dígitos.");
    }
    else{
        try{
            cedu=Integer.parseInt(jTextField4.getText());
            celular=Integer.parseInt(jTextField5.getText());
            PreparedStatement ps = cn.prepareStatement("INSERT INTO `usuarios`(`nombre`,`apellido`,`cedula`,`celular`,`direccion`,`contraseña`,`nivel`) VALUES (?, ?, ?, ?, ?, ?, ?)");
            ps.setString(1, nombre);
            ps.setString(2, apellido);
            ps.setInt(3, cedu);
            ps.setInt(4, celular);
            ps.setString(5, direc);
            ps.setString(6, contraseña);
            ps.setString(7, nivel);
            int rowsInserted = ps.executeUpdate();
            if (rowsInserted > 0) {
                JOptionPane.showMessageDialog(null, "Usuario registrado exitosamente");
            } else {
                JOptionPane.showMessageDialog(null, "No se pudo registrar el usuario");
            }
        }catch(java.sql.SQLIntegrityConstraintViolationException e){
            JOptionPane.showMessageDialog(null, "La cédula ya está registrada.");
        }
    }
}
```

```

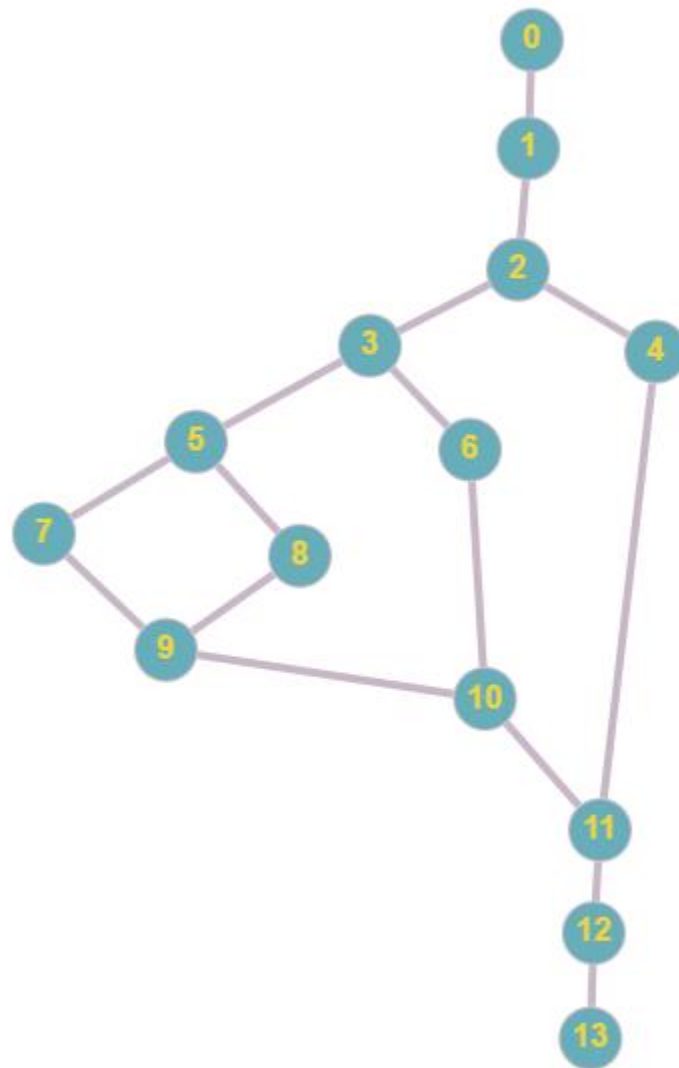
    catch(Exception e){}
  }
}

```

2. DFD



3. GRAFO DE FLUJO (GF)



4. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

RUTAS

R1: $1 \rightarrow 2 \rightarrow 4 \rightarrow 11 \rightarrow 12 \rightarrow \text{Fin}$

R2: $1 \rightarrow 2 \rightarrow 3 \rightarrow 6 \rightarrow 10 \rightarrow 11 \rightarrow 12 \rightarrow \text{Fin}$

R3: $1 \rightarrow 2 \rightarrow 3 \rightarrow 5 \rightarrow 8 \rightarrow 9 \rightarrow 10 \rightarrow 11 \rightarrow 12 \rightarrow \text{Fin}$

R4: $1 \rightarrow 2 \rightarrow 3 \rightarrow 5 \rightarrow 7 \rightarrow 9 \rightarrow 10 \rightarrow 11 \rightarrow 12 \rightarrow \text{Fin}$

COMPLEJIDAD CICLOMÁTICA

Fórmula 1: $V(G)=P+1$

$P=3$ (nodos de decisión).

$V(G)=3+1=4$

Fórmula 2: $V(G)=A-N+2$

$A=14$ (aristas).

$N=12$ (nodos).

$V(G)=14-12+2=4$

Prueba caja blanca de Registrar Asistencia

1. CÓDIGO FUENTE

Entrada:

```
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {  
    nombre.setText(usuario);  
  
    try {  
        // Primero se valida si el usuario tiene un registro de ingreso para hoy  
        PreparedStatement psCheck = cn.prepareStatement("SELECT * FROM asistencias  
WHERE usuario = ? AND fecha = ?");  
        LocalDateTime now = LocalDateTime.now();  
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");  
        String fecha = now.format(formatter); // Solo la fecha, sin la hora  
  
        psCheck.setString(1, usuario);  
        psCheck.setString(2, fecha);  
  
        ResultSet rsCheck = psCheck.executeQuery();  
  
        // Si ya existe un registro, no se inserta nuevo registro de asistencia  
        if (rsCheck.next()) {  
            JOptionPane.showMessageDialog(null, "Ya has registrado tu ingreso hoy.");  
        } else {  
            // Si no existe, se registra el nuevo ingreso  
            String sqlSelectNivel = "SELECT nivel FROM usuarios WHERE cedula=? AND  
contraseña=?";  
            PreparedStatement ps = cn.prepareStatement(sqlSelectNivel);  
            ps.setString(1, usuario);  
            ps.setString(2, contraseña);  
            ResultSet rs = ps.executeQuery();  
  
            if (rs.next()) {  
                // Registramos la asistencia  
                String sqlAsistencia = "INSERT INTO asistencias (usuario, fecha, hora,  
hora_salida,salida,nombre) VALUES (?, ?, ?, ?,?,?)";  
                PreparedStatement psAsistencia = cn.prepareStatement(sqlAsistencia);  
                String horaEntrada = now.format(DateTimeFormatter.ofPattern("HH:mm:ss"));  
  
                psAsistencia.setString(1, usuario);  
                psAsistencia.setString(2, fecha);  
                psAsistencia.setString(3, horaEntrada); // Hora de entrada  
                psAsistencia.setString(4, horaEntrada);  
                psAsistencia.setInt(5, 0);  
                psAsistencia.setString(6, nombre_user);  
                int rowsInserted = psAsistencia.executeUpdate();  
            }  
        }  
    }  
}
```



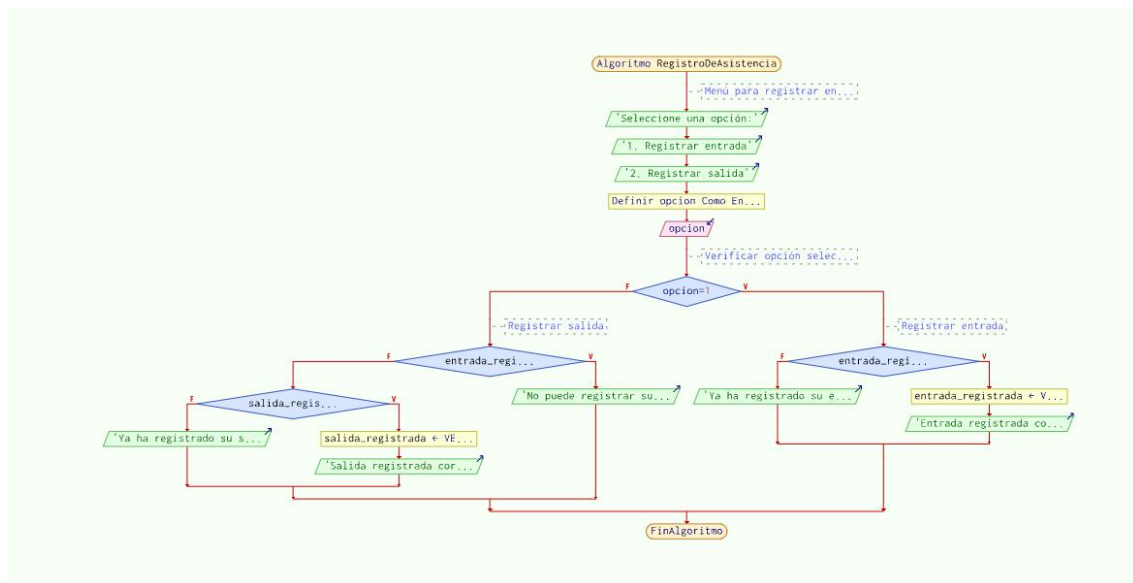
```

        int rowsUpdated = psAsistencia.executeUpdate();
        if (rowsUpdated > 0) {
            JOptionPane.showMessageDialog(null, "Hora de salida registrada exitosamente.");
        } else {
            JOptionPane.showMessageDialog(null, "No se encontró un registro de entrada pendiente para este usuario en la fecha actual.");
        }
    } else {
        JOptionPane.showMessageDialog(null, "Usuario o contraseña incorrectos.");
    }
}
} else {
    JOptionPane.showMessageDialog(null, "No se encontró un registro de asistencia para este usuario en la fecha actual.");
}

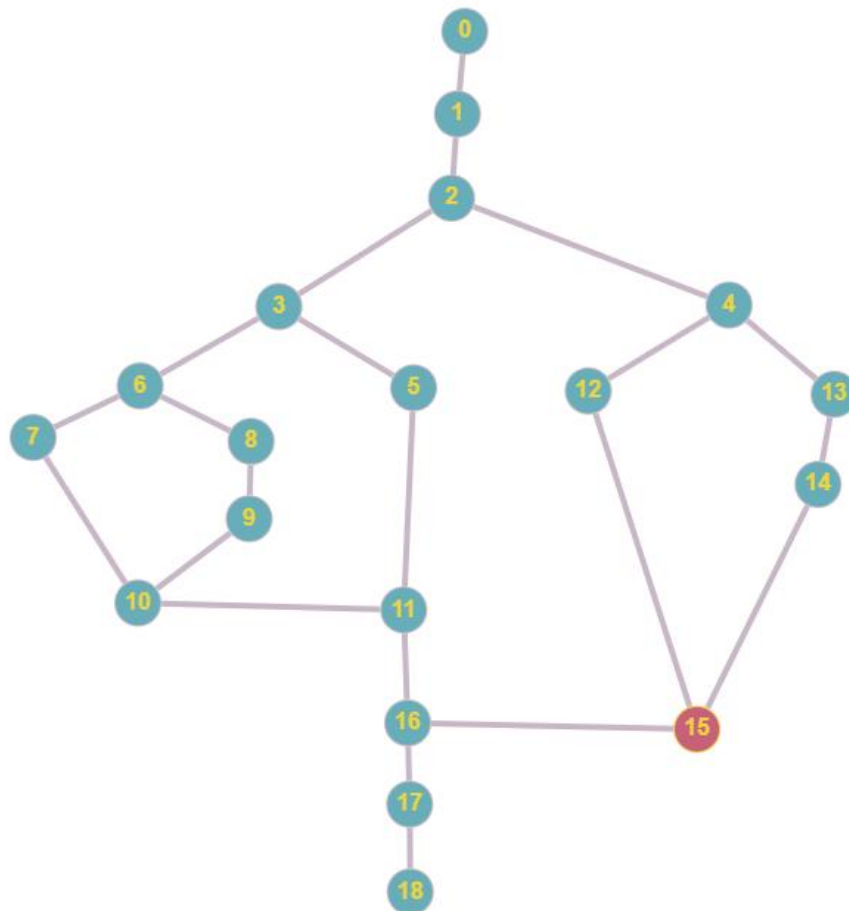
} catch (Exception e) {
    e.printStackTrace();
    JOptionPane.showMessageDialog(null, "Error: " + e.getMessage());
}
}

```

2. DFD



3. GRAFO DE FLUJO (GF)



4. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

RUTAS

R1: $1 \rightarrow 2 \rightarrow 4 \rightarrow 13 \rightarrow 14 \rightarrow 15 \rightarrow 16 \rightarrow 17 \rightarrow \text{Fin}$

R2: $1 \rightarrow 2 \rightarrow 4 \rightarrow 12 \rightarrow 15 \rightarrow 16 \rightarrow 17 \rightarrow \text{Fin}$

R3: $1 \rightarrow 2 \rightarrow 3 \rightarrow 5 \rightarrow 11 \rightarrow 16 \rightarrow 17 \rightarrow \text{Fin}$

R4: $1 \rightarrow 2 \rightarrow 3 \rightarrow 6 \rightarrow 8 \rightarrow 9 \rightarrow 10 \rightarrow 11 \rightarrow 16 \rightarrow 17 \rightarrow \text{Fin}$

R5: $1 \rightarrow 2 \rightarrow 3 \rightarrow 6 \rightarrow 7 \rightarrow 10 \rightarrow 11 \rightarrow 16 \rightarrow 17 \rightarrow \text{Fin}$

COMPLEJIDAD CICLOMÁTICA

Fórmula 1: $V(G)=P+1$

$P=4$ (nodos de decisión).

$V(G)=4+1=5$

Fórmula 2: $V(G)=A-N+2$

$A=20$ (aristas).

$N=17$ (nodos).

$V(G)=20-17+2=5$

Prueba caja blanca de Administrar usuarios

1. CÓDIGO FUENTE

Actualizar

```
private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {  
    String nivel, celular, direccion, contrasena;  
    nivel = B_nivel.getSelectedItem().toString();  
    String cedulaIngresada = B_cedula.getText();  
    if (cedulaIngresada.isEmpty()) {  
        JOptionPane.showMessageDialog(null, "Por favor, ingresa la cédula.");  
    }  
    try {  
        celular = B_celular.getText();  
        direccion = B_direccion.getText();  
        contrasena = B_contrasena.getText();  
        PreparedStatement ps = cn.prepareStatement("UPDATE usuarios SET nivel  
= ?, celular = ?, direccion = ?, contraseña = ? WHERE cedula = ?");  
        ps.setString(1, nivel);  
        ps.setString(2, celular);  
        ps.setString(3, direccion);  
        ps.setString(4, contrasena);  
        ps.setString(5, cedulaIngresada);  
        int rowsUpdated = ps.executeUpdate();  
        if (rowsUpdated > 0) {  
            JOptionPane.showMessageDialog(null, "Usuario actualizado exitosamente.");  
        }  
    } catch (Exception e) {}  
}
```

Eliminar

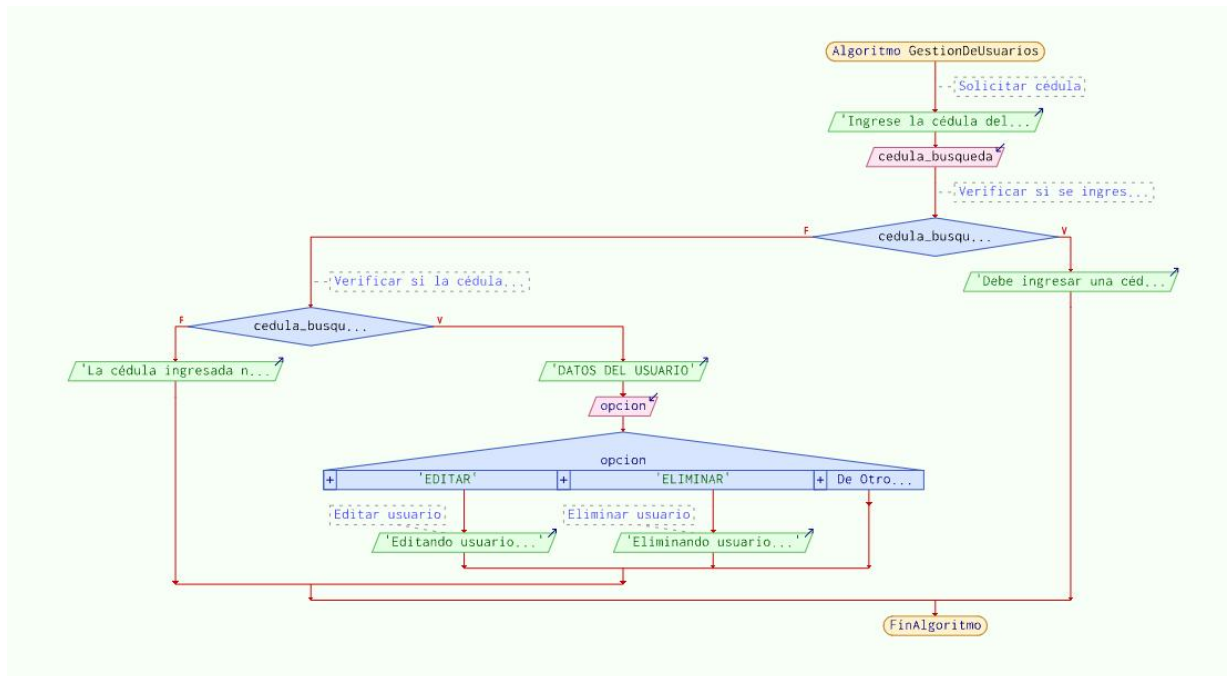
```
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {  
    String cedulaIngresada = B_cedula.getText();  
    String[] options = { "si", "no" };  
    if (cedulaIngresada.isEmpty()) {  
        JOptionPane.showMessageDialog(null, "Por favor, ingresa la cédula para eliminar el  
usuario.");  
    }  
    else {  
        var selection = JOptionPane.showOptionDialog(null, "¿Esta seguro que desea eliminar  
este usuario?", "Mensaje!", 0, 3, null, options, options[0]);  
        if (selection == 0) {  
            try {  
                String sql = "DELETE FROM usuarios WHERE cedula = ?";  
                PreparedStatement ps = cn.prepareStatement(sql);  
                ps.setString(1, cedulaIngresada);  
  
                int rowsAffected = ps.executeUpdate();  
  
                if (rowsAffected > 0) {  
                    JOptionPane.showMessageDialog(null, "Usuario eliminado exitosamente.");  
                }  
            }  
        }  
    }  
}
```

```

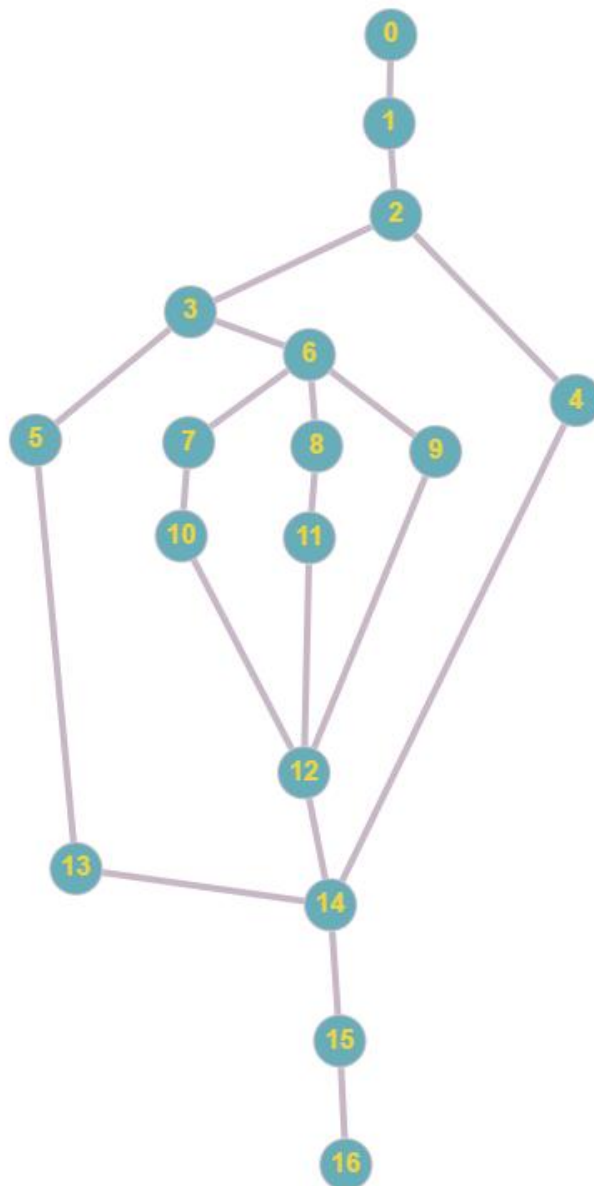
        B_nombre.setText("");
        B_apellido.setText("");
        B_cedula.setText("");
        B_celular.setText("");
        B_contraseña.setText("");
        B_direccion.setText("");
        jButton5.setEnabled(false);
        jButton6.setEnabled(false);
    }
} catch (Exception e) {
    JOptionPane.showMessageDialog(null, "Error al eliminar el usuario: " +
e.getMessage());
}
}
if (selection == 1) {
}}}

```

2. DFD



3. GRAFO DE FLUJO



4. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

RUTAS

R1: $1 \rightarrow 2 \rightarrow 4 \rightarrow 14 \rightarrow 15 \rightarrow \text{Fin}$

R2: $1 \rightarrow 2 \rightarrow 3 \rightarrow 5 \rightarrow 13 \rightarrow 14 \rightarrow 15 \rightarrow \text{Fin}$

R3: $1 \rightarrow 2 \rightarrow 3 \rightarrow 6 \rightarrow 9 \rightarrow 12 \rightarrow 14 \rightarrow 15 \rightarrow \text{Fin}$

R4: $1 \rightarrow 2 \rightarrow 3 \rightarrow 6 \rightarrow 8 \rightarrow 11 \rightarrow 12 \rightarrow 14 \rightarrow 15 \rightarrow \text{Fin}$

R5: $1 \rightarrow 2 \rightarrow 3 \rightarrow 5 \rightarrow 13 \rightarrow 14 \rightarrow 15 \rightarrow \text{Fin}$

COMPLEJIDAD CICLOMÁTICA

Fórmula 1: $V(G)=P+1$

$P=4$ (nodos de decisión).

$V(G)=4+1=5$

Fórmula 2: $V(G)=A-N+2$

$A=18$ (aristas).

$N=15$ (nodos).

$$V(G)=18-15+2=5$$

Prueba caja blanca de Historial de asistencias

1. CÓDIGO FUENTE

```
private void B_buscarActionPerformed(java.awt.event.ActionEvent evt) {
    String usuario_registro = H_buscar.getText().trim();
    String nombre_user = "";
    Date date = date_f.getDate();
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
    String fechaSeleccionada = (date != null) ? sdf.format(date) : "";

    try {
        // Obtener nombre del usuario si se ingresa una cédula específica
        if (!usuario_registro.isEmpty()) {
            String sql = "SELECT nombre FROM usuarios WHERE cedula = ?";
            PreparedStatement ps = cn.prepareStatement(sql);
            ps.setString(1, usuario_registro);
            ResultSet rs = ps.executeQuery();
            if (rs.next()) {
                nombre_user = rs.getString("nombre");
            } else {
                JOptionPane.showMessageDialog(null, "El usuario no existe");
            }
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, "Error al obtener usuario: " + e.getMessage());
    }

    try {
        String sql;
        PreparedStatement ps;

        // Construir la consulta según los filtros
        if (usuario_registro.isEmpty() && fechaSeleccionada.isEmpty()) {
            // Caso 1: Todos los registros
            sql = "SELECT nombre, fecha, hora, hora_salida FROM asistencias";
            ps = cn.prepareStatement(sql);
        } else if (usuario_registro.isEmpty()) {
            // Caso 2: Todos los usuarios en una fecha específica
            sql = "SELECT nombre, fecha, hora, hora_salida FROM asistencias WHERE fecha = ?";
            ps = cn.prepareStatement(sql);
            ps.setString(1, fechaSeleccionada);
        } else if (fechaSeleccionada.isEmpty()) {
            // Caso 3: Un usuario específico en todas las fechas
            sql = "SELECT nombre, fecha, hora, hora_salida FROM asistencias WHERE usuario = ?";
            ps = cn.prepareStatement(sql);
            ps.setString(1, usuario_registro);
        } else {
            // Caso 4: Un usuario específico en una fecha específica

```

```

        sql = "SELECT nombre, fecha, hora, hora_salida FROM asistencias WHERE usuario = ?
AND fecha = ?";
        ps = cn.prepareStatement(sql);
        ps.setString(1, usuario_registro);
        ps.setString(2, fechaSeleccionada);
    }

    ResultSet rs = ps.executeQuery();

    // Definir las columnas del JTable
    String[] columnNames = {"Usuario", "Fecha", "Hora Entrada", "Hora Salida"};
    DefaultTableModel model = new DefaultTableModel(columnNames, 0);

    // Agregar los registros al JTable
    while (rs.next()) {
        String usuario = usuario_registro.isEmpty() ? rs.getString("nombre") : nombre_user;
        String fecha = rs.getString("fecha");
        String hora = rs.getString("hora");
        String hora_salida = rs.getString("hora_salida");
        model.addRow(new Object[]{usuario, fecha, hora, hora_salida});
    }

    // Si no hay registros
    if (model.getRowCount() == 0) {
        model.addRow(new Object[]{"No hay asistencias registradas.", "", "", ""});
    }

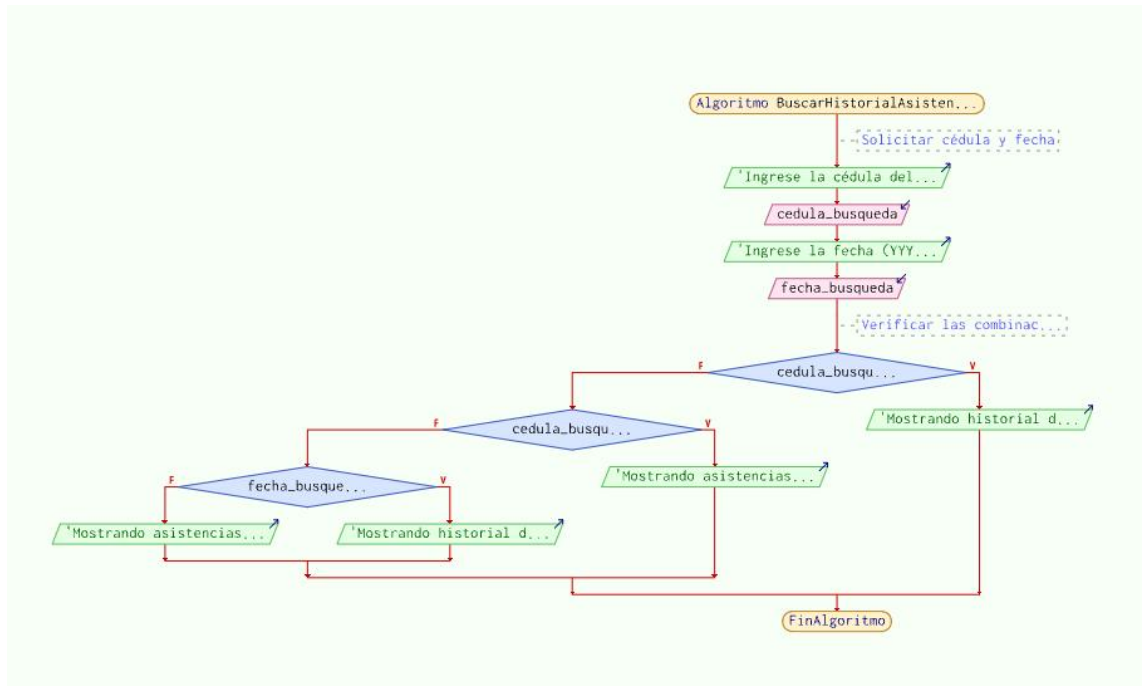
    // Establecer el modelo en el JTable
    Lista_asistencias.setModel(model);

} catch (Exception e) {
    JOptionPane.showMessageDialog(null, "Error al obtener el historial de asistencias: " +
e.getMessage());
}

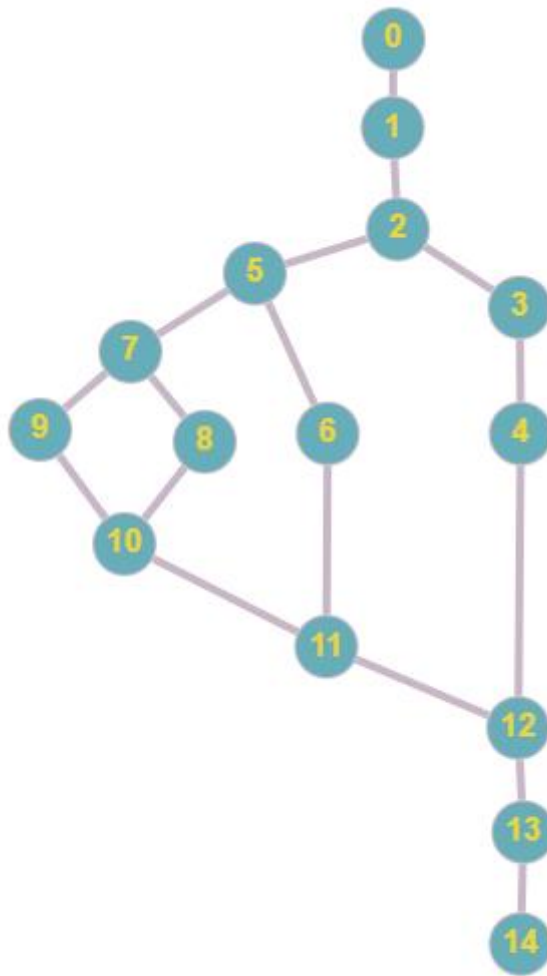
}

```

2. DFD



3. GRAFO DE FLUJO



4. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

RUTAS

R1: $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 12 \rightarrow 13 \rightarrow \text{Fin}$

R2: $1 \rightarrow 2 \rightarrow 5 \rightarrow 6 \rightarrow 11 \rightarrow 12 \rightarrow 13 \rightarrow \text{Fin}$

R3: $1 \rightarrow 2 \rightarrow 5 \rightarrow 7 \rightarrow 8 \rightarrow 10 \rightarrow 11 \rightarrow 12 \rightarrow 13 \rightarrow \text{Fin}$

R4: $1 \rightarrow 2 \rightarrow 5 \rightarrow 7 \rightarrow 9 \rightarrow 10 \rightarrow 11 \rightarrow 12 \rightarrow 13 \rightarrow \text{Fin}$

COMPLEJIDAD CICLOMÁTICA

Fórmula 1: $V(G)=P+1V$

$P=4$ (nodos de decisión).

$V(G)=3+1=4$

Fórmula 2: $V(G)=A-N+2$

$A=15$ (aristas).

$N=13$ (nodos).

$V(G)=15-13+2=4$