

# Prueba de Caja Negra

---

*“Diseño de sistema de gestión de contratos para una plaza comercial”*

## **Integrantes:**

Gallardo Vega Santiago Jose,  
Pérez Díaz David Ismael,  
Zambrano Cajas Isabela Valentina.

**Fecha: 2025-06-12**

## Partición de clases equivalentes

**TABLA 1 Requisito Funcional N°1: Validación de Credenciales**

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
Nombre de las variables	nombreUsuario=nombreUsuario	Válido	admin
	nombreUsuario!=nombreUsuario	No válido	-----
	password=password	Válido	123
	password!=password	No válido	-----

### Captura de la pantalla de la ejecución

#### Descripción de la ejecución de acuerdo a los casos de prueba de la tabla 1, casos pass (Ok) y fail (no OK)

El usuario ingresará al sistema mediante un usuario y contraseña previamente creados. Si el usuario ingresa erróneamente su nombre o contraseña, debe volver a intentarlo y a su vez se visualizará un mensaje que dirá "usuario o contraseña no válido". Después de 3 intentos de inicio de sesión consecutivos fallidos, se bloqueará el ingreso de credenciales por 5 minutos.

#### Caso pass

*Figura 1 Interfaz gráfica validación de usuario*

La interfaz gráfica de validación de usuario (LOGIN) presenta un encabezado de color verde oscuro. El título "LOGIN" está en negrita. Hay dos campos de entrada de texto con el prefijo "Usuario:" y "Contraseña:". Debajo de los campos hay tres botones: "Login" (verde), "Limpiar" (verde) y "Salir" (verde).

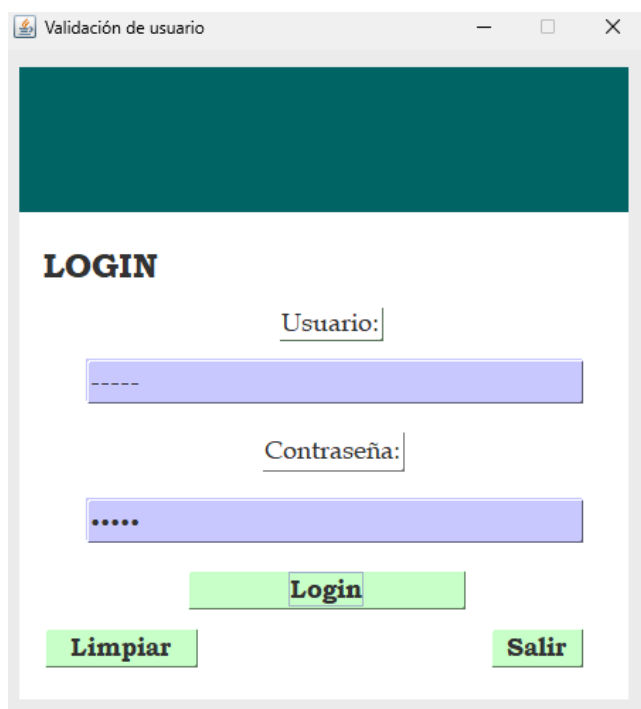
En la figura 1 se visualiza una interfaz de validación de usuario. Esta pantalla presenta un título destacado que dice "LOGIN", acompañado por dos campos de entrada: uno destinado al nombre de usuario y otro a la contraseña. Debajo, se encuentran tres botones con funciones claras: uno para limpiar los campos, otro para iniciar sesión y un tercero para salir de la aplicación.

*Figura 2\_ Usuario y contraseña correcta*

*Figura 3 Visualización de pantalla principal validado usuario y contraseña*

En la figura 2 y la figura 3 se visualiza cómo se insertan los mensajes correctos, en este caso siendo *admin* y *1234* como usuario y contraseña. Una vez se valide esta información, se abre la pantalla principal “Gestión de contratos”, la cual contiene recuadros que permiten acceder a diferentes opciones como nuevo contrato (civil o laboral), datos administrativos y el botón de salir.

*Figura 4 Usuario y contraseña incorrectas*



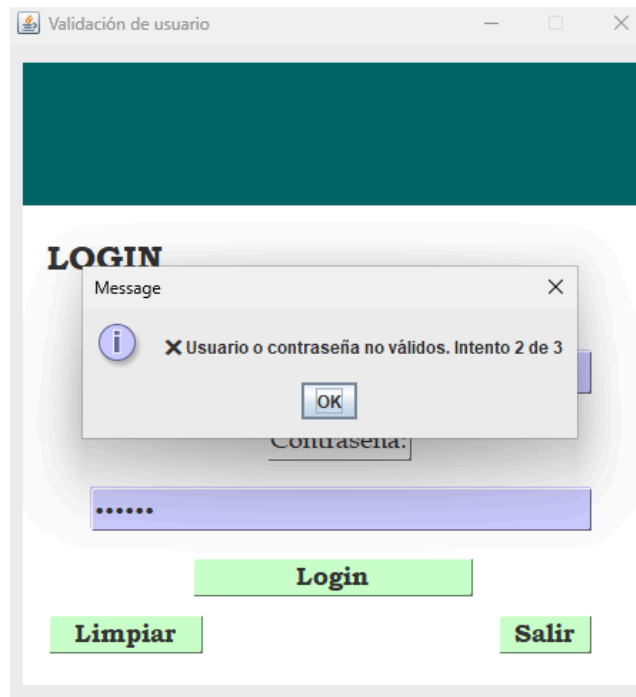
The screenshot shows a window titled "Validación de usuario". It has a dark green header bar. Below the header, the word "LOGIN" is displayed in bold. There are two input fields: "Usuario:" and "Contraseña:". The "Usuario:" field is empty, and the "Contraseña:" field contains five dots. Below the input fields, there are three buttons: "Login" (green), "Limpiar" (green), and "Salir" (green).

*Figura 5 Mensaje de usuario o contraseña invalido primer intento*

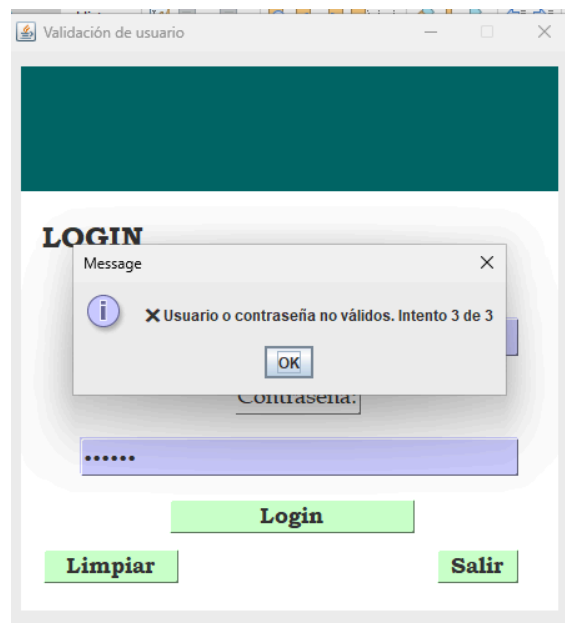


The screenshot shows the same "Validación de usuario" window as in Figure 4, but with an error message dialog box overlaid. The dialog box is titled "Mensaje" and contains the text "X Usuario o contraseña no válidos. Intento 1 de 3". There is an "OK" button in the dialog box. The background window is slightly dimmed.

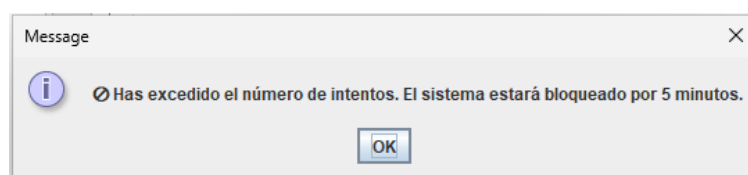
*Figura 6 Mensaje de usuario o contraseña invalido primer intento*



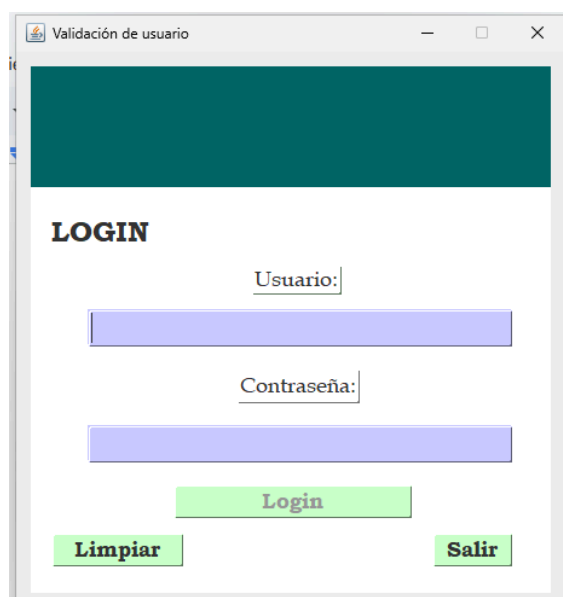
*Figura 7 Mensaje de usuario o contraseña invalido segundo intento*



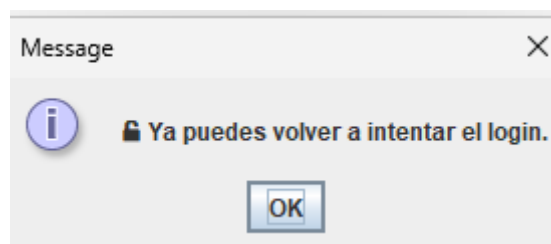
*Figura 8 Mensaje de usuario o contraseña invalido tercer intento*



*Figura 10 Bloqueo de ingreso tras tres intentos*



*Figura 11 Desbloqueo de acceso a la validación de credenciales*



En la figura 4 se visualiza el ingreso de credenciales incorrectas. En las figuras 5, 6 y 7 se muestra el mensaje que indica que el usuario o la contraseña son incorrectos. El sistema permite un total de tres intentos. Tras cada intento, como se observa en la figura 8, aparece un mensaje que informa que el inicio de sesión se ha bloqueado por 5 minutos. Una vez transcurrido ese tiempo, tal como se aprecia en la figura 11, se muestra un mensaje indicando que ya es posible volver a acceder al login.

# Prueba de Caja Blanca

---

*“Diseño de sistema de gestión de contratos para una plaza comercial”*

**Integrantes:**

Gallardo Vega Santiago Jose,  
Pérez Díaz David Ismael,  
Zambrano Cajas Isabela Valentina.

**Fecha: 2025-06-12**

## Prueba caja blanca

### Requisito funcional 1: Validar credenciales de administrador

El programa deberá tener una interfaz gráfica donde el administrador ingrese y valide sus credenciales. El usuario ingresará al sistema mediante un usuario y contraseña previamente creados. Si el usuario ingresa erróneamente su nombre o contraseña, debe volver a intentarlo y a su vez se visualizará un mensaje que dirá "usuario o contraseña no válido". Después de 3 intentos de inicio de sesión consecutivos fallidos, se bloqueará el ingreso de credenciales por 5 minutos.

#### 1. CÓDIGO FUENTE

Pegar el trozo de código fuente que se requiere para el caso de prueba

```
        if (!this.LoginEstado) {
            intentosFallidos++;

            JOptionPane.showMessageDialog(null,
                "X Usuario o contraseña no válidos. Intento "
                + intentosFallidos + " de " + MAX_INTENTOS);

            vista_validacion.txtContrasena.setText("");
            vista_validacion.txtUsuario.setText("");
            vista_validacion.txtUsuario.requestFocusInWindow();

            if (intentosFallidos >= MAX_INTENTOS) {
                bloquearLogin();
            }
        } else {
            vista_validacion.dispose();
            iniciarPrograma();
            System.out.println("Acceso Exitoso");
        }

    }).start();

    public boolean validarUsuarios() {
        return this.nombreUsuario.equals("Veronica") && this.password.equals("1234");
    }

    public boolean compararPass(String Pass1, String Pass2) {
        return Pass1.equals(Pass2);
    }
}
```

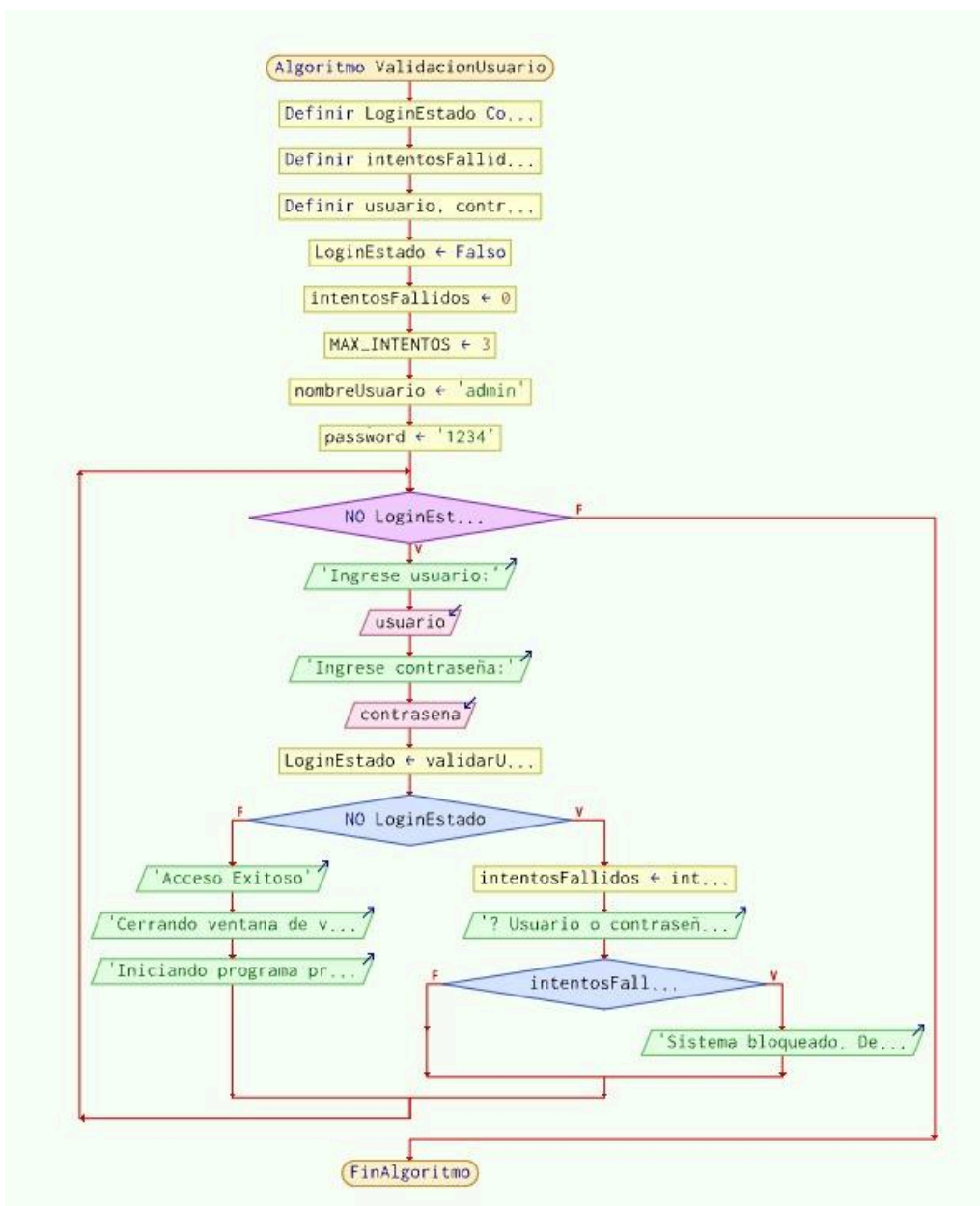


## CÓDIGO DE AUTENTICACIÓN CON MONGODB

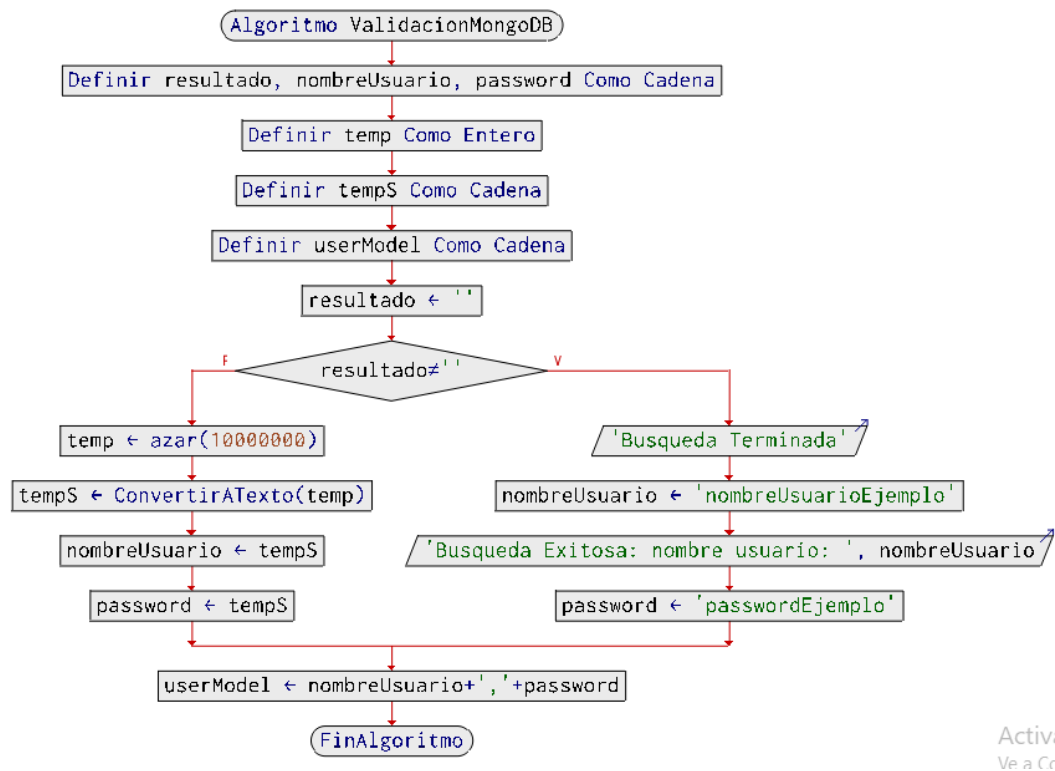
```
45 if (resultado != null) {
46     System.out.println("Busqueda Terminada ");
47     userModel.setNombreUsuario(resultado.getString("nombreUsuario"));
48     System.out.println("Busqueda Exitosa: nombre usuario: "+userModel.getNombreUsuario());
49     userModel.setPassword(resultado.getString("password"));
50 }
51 else{
52
53     int temp = random.nextInt(10000000);
54     String tempS = String.valueOf(temp);
55     userModel.setNombreUsuario(tempS);
56     userModel.setPassword(tempS);
57 }
58 }
```

## 2. DIAGRAMA DE FLUJO (DF)

Realizar un DF del código fuente del numeral 1

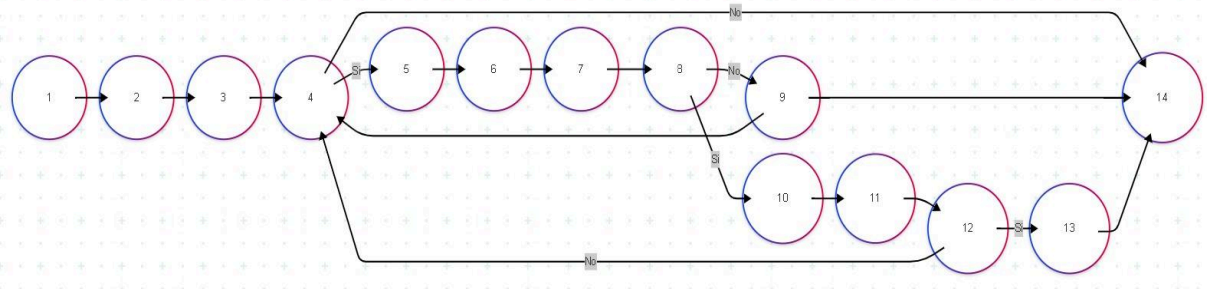


## DF DE CONEXIÓN CON MONGODB

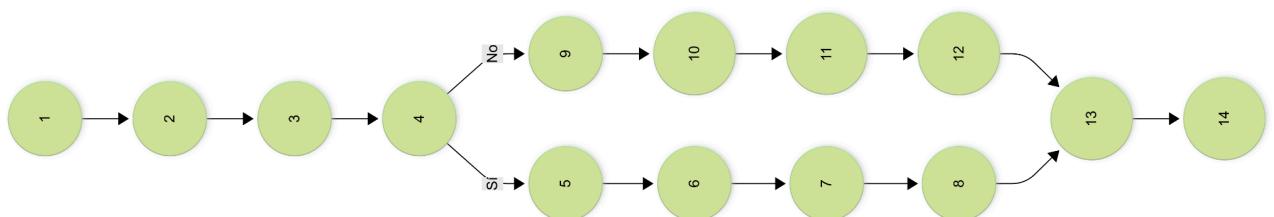


### 3. GRAFO DE FLUJO (GF)

Realizar un GF en base al DF del numeral



## GF DE CONEXIÓN CON MONGODB



### 4. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

Determinar en base al GF del numeral 4  
RUTAS

R1: 1-2-3-4-14  
R2: 1-2-3-4-5-6-7-8-9-14  
R3: 1-2-3-4-5-6-7-8-10-11-12-13-14  
R4: 1-2-3-4-9-14  
R5: 1-2-3-4-12-13-141

RUTAS

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

## **5. COMPLEJIDAD CICLOMÁTICA**

Se puede calcular de las siguientes formas:

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

$$V(G) = A - N + 2$$
$$V(G)= 17-14+2$$
$$V(G)=5$$

DONDE:

P: Número de nodos prediado

A: Número de aristas

N: Número de nodos

$$V(G)=1+1$$
$$V(G)=2$$

$$V(G)=A-N+2$$
$$V(G)=14-14+2$$
$$V(G)=2$$

# Prueba de Caja Negra

---

*“Diseño de sistema de gestión de contratos para una plaza comercial”*

## **Integrantes:**

Gallardo Vega Santiago Jose,  
Pérez Díaz David Ismael,  
Zambrano Cajas Isabela Valentina.

**Fecha: 2025-06-12**

## Partición de clases equivalentes

**TABLA 1 Requisito Funcional N°2.1 Cambio de usuario**

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
Nombre de las variables	nombreUsuario=admin	Válido	admin
	nombreUsuario=adminSistema	Válido	-----

Datos y Herramientas de Usuario

Usuario Login:  Cambiar Usuario

Contraseña actual

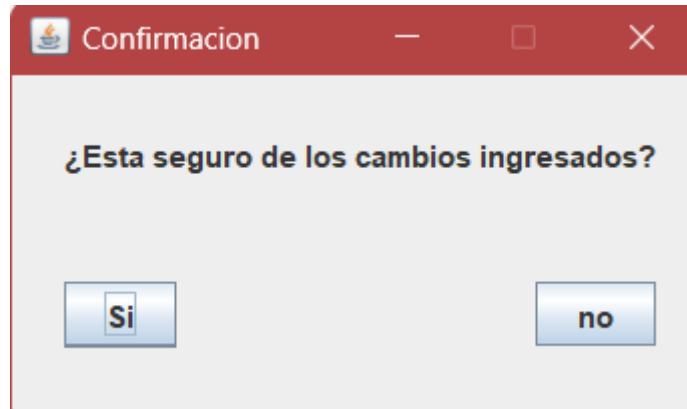
Contraseña Nueva  Cambiar Contraseña

Confirmacion Contraseña

Opciones Avanzadas:

Host MongoDB:

Regresar



### Descripción de la ejecución de acuerdo a los casos de prueba de la tabla 1, casos pass (Ok) y fail (no OK)

El usuario ingresará un nombre al cuadro de texto “usuario Login” y dará clic a “Cambiar usuario”. Al confirmar el cambio se cambiará el nombre.

#### Caso pass

*Figura 1 Confirmación en CLI de cambio de usuario*

```
Filtro: {"nombreUsuario": "admin"}
Update: {"$set": {"nombreUsuario": "admin1"}}
Documentos modificados: 1
Usuario cambiado correctamente a admin1
```

*Figura 2 Confirmación en MONGODB de cambio de usuario*

```
_id: ObjectId('684daf3f1722f305c2f96f6a')
nombreUsuario: "admin1"
password: "123"
```

#### Caso fail

*Figura 3 Confirmación en CLI de que no hubo cambios*

```

Busqueda Exitosa: nombre usuario: admin
Usuarioadminpssword:123
Acceso Exitoso
Boton Datos Usuario
Vista Datos
Filtro: {"nombreUsuario": "admin"}
Update: {"$set": {"nombreUsuario": "admin"}}
Documentos modificados: 0
Usuario cambiado correctamente a admin

```

Figura 4 Confirmación en MONGODB de que no hubo cambios

```

_id: ObjectId('684daf3f1722f305c2f96f6a')
nombreUsuario : "admin"
password : "123"

```

TABLA 1 Requisito Funcional N°2.2 Cambio de contraseña

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
Nombre de las variables	passVer=modelo.getPassword()	Válido	passVer
	passVer!=modelo.getPassword()	No válido	-----
	passConf= (passNueva eq passConf)	Válido	passConf
	passConf= !(passNueva eq passConf)	No válido	-----

```

else if (e.getSource() == vistadatos.BtnCambioPassword) {
    String passNueva = vistadatos.jPasswordnew.getText();
    String passConfir = vistadatos.jPasswordConfir.getText();
    String passActual = vistadatos.jPasswordactual.getText();

    boolean passConf = passNueva.equals(passConfir);
    boolean passVer = modelo.getPassword().equals(passActual);

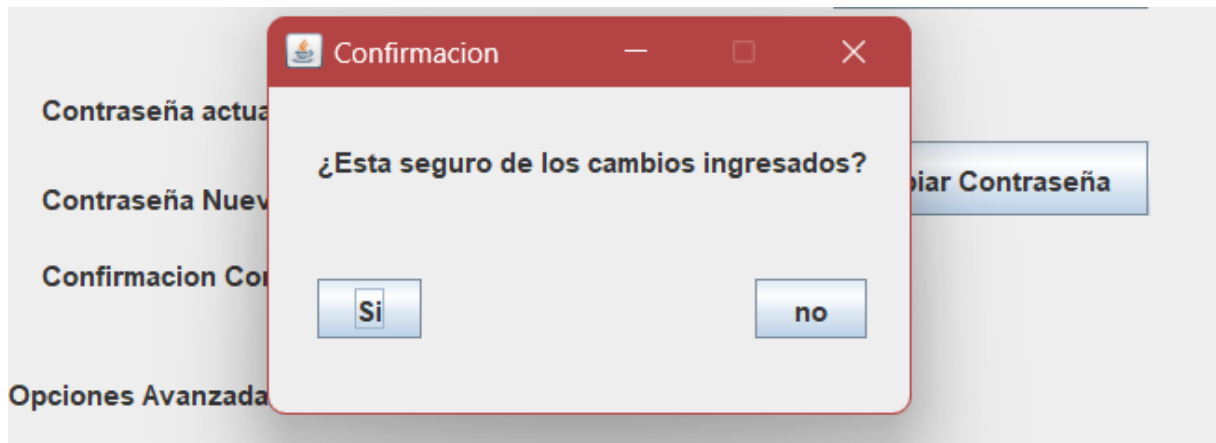
    if (!passVer) {
        JOptionPane.showMessageDialog(null, "La contraseña actual es incorrecta.");
    } else if (!passConf) {
        JOptionPane.showMessageDialog(null, "Las nuevas contraseñas no coinciden.");
    } else {
        // Ambas condiciones se cumplieron
        nuevaPasswordPendiente = passConfir; // almacenar temporalmente
        accion = AccionConfirmacion.CAMBIO_PASSWORD;
        iniciarConfirmacion(); // abrir ventana de confirmación
    }
}

```

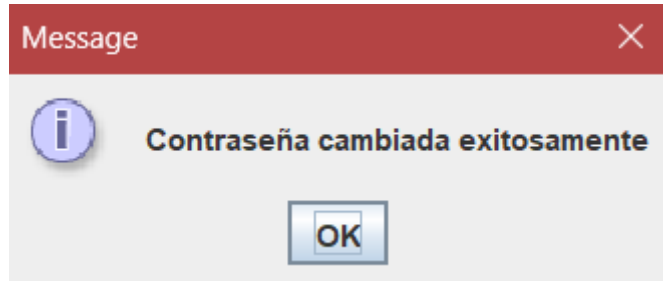
## Caso pass

Si la contraseña actual es digitada correctamente, y las contraseñas de cambio coinciden, el programa actualizará el registro en la base de datos y se podrán observar las confirmaciones respectivas en la interfaz gráfica.

*Figura 5 Confirmación gráfica de cambio*



*Figura 6 Confirmación gráfica de cambio exitoso*



*Figura 7 Confirmación en CLI de cambio exitoso*

```
Filtro: {"nombreUsuario": "admin"}
Update: {"$set": {"password": "1234"}}
Documentos modificados: 1
```

*Figura 8 Cambio en MONGODB*

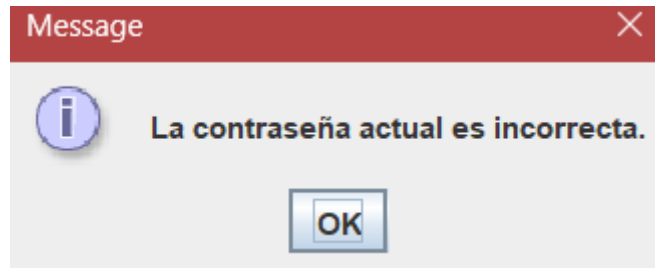
```
_id: ObjectId('684daf3f1722f305c2f96f6a')
nombreUsuario: "admin"
password: "1234"
```



## CASO FAIL

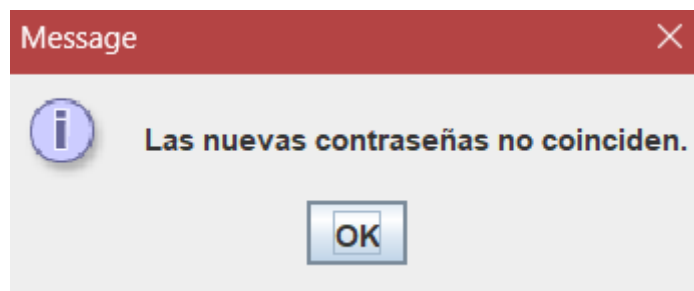
- 1) Si la contraseña actual está incorrecta, saltará un mensaje de error diciendo que no está correcta

*Figura 9 Mensaje de error, contraseña actual incorrecta*



- 2) Si la contraseña actual es correcta pero las contraseñas de confirmación no coinciden, saltará un error notificando que los campos no coinciden

*Figura 10 Mensaje de error, contraseñas de cambio diferentes*



# Prueba de Caja Blanca

---

*“Diseño de sistema de gestión de contratos para una plaza comercial”*

**Integrantes:**

Gallardo Vega Santiago Jose,  
Pérez Díaz David Ismael,  
Zambrano Cajas Isabela Valentina.

**Fecha: 2025-06-12**

Prueba caja blanca

## Requisito funcional 2: Validar credenciales de administrador

Permitir que el usuario registrado pueda visualizar su información de perfil (nombre de usuario) y modificar su nombre de usuario y contraseña.

### Verificación de credenciales:

#### 1. CÓDIGO FUENTE

Pegar el trozo de código fuente que se requiere para el caso de prueba

```
else if (e.getSource() == vistadatos.BtnCambioUser) {
    accion = AccionConfirmacion.CAMBIO_USER;
    iniciarConfirmacion();
}

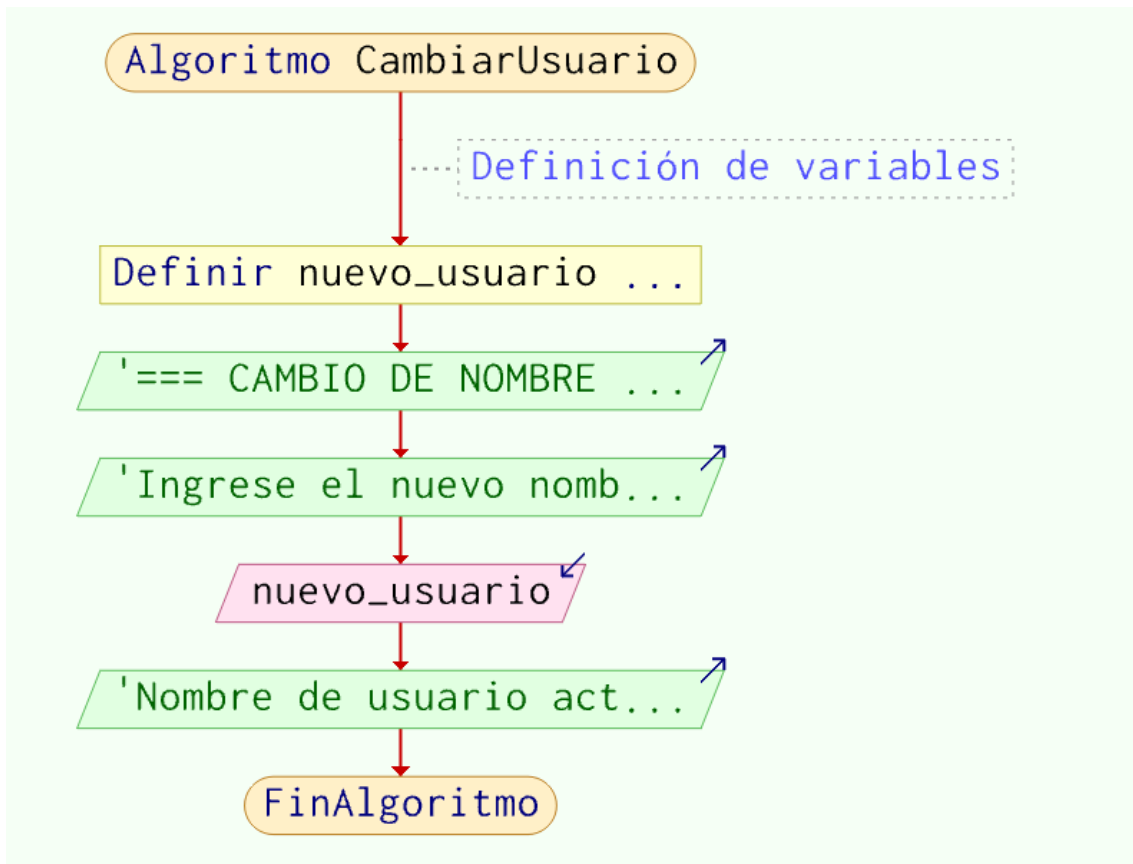
else if (e.getSource() == vistadatos.BtnCambioPassword) {
    String passNueva = vistadatos.jPasswordnew.getText();
    String passConfir = vistadatos.jPasswordConfir.getText();
    String passActual = vistadatos.jPasswordactual.getText();

    boolean passConf = passNueva.equals(passConfir);
    boolean passVer = modelo.getPassword().equals(passActual);

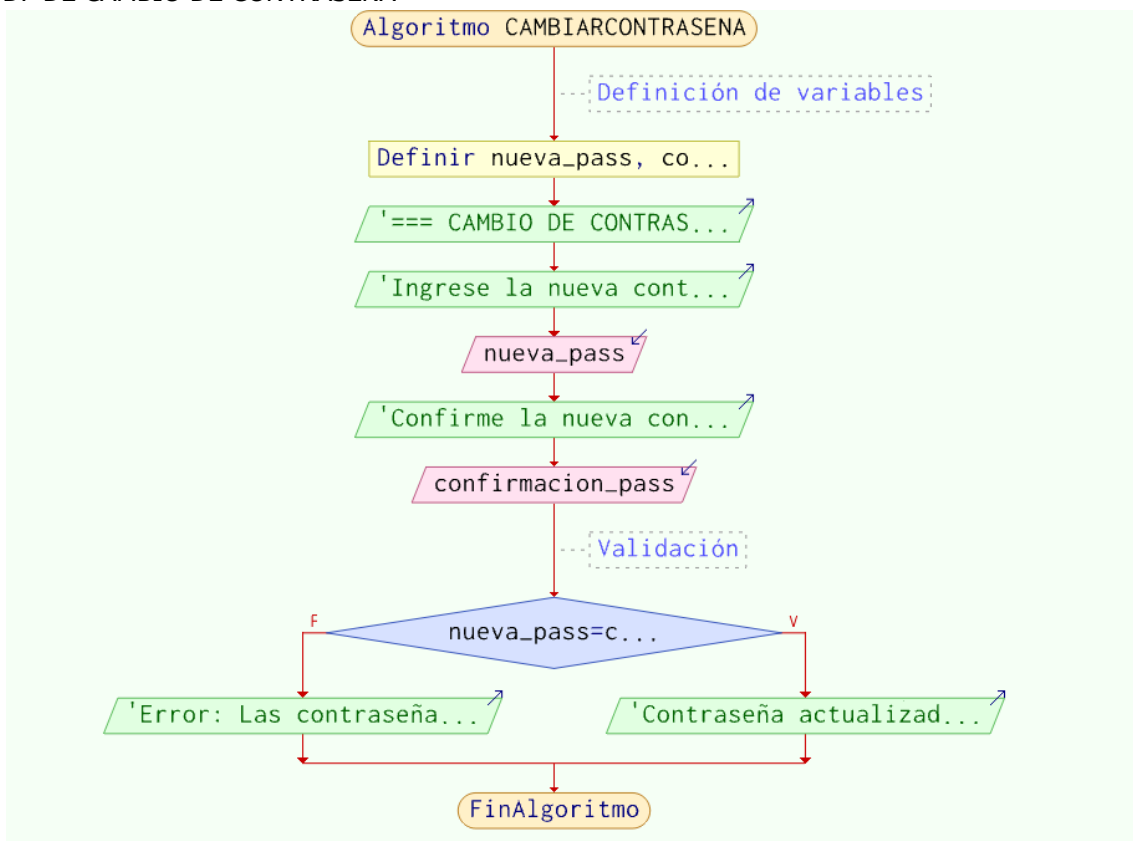
    if (!passVer) {
        JOptionPane.showMessageDialog(null, "La contraseña actual es incorrecta.");
    } else if (!passConf) {
        JOptionPane.showMessageDialog(null, "Las nuevas contraseñas no coinciden.");
    } else {
        // Ambas condiciones se cumplieron
        nuevaPasswordPendiente = passConfir; // almacenar temporalmente
        accion = AccionConfirmacion.CAMBIO_PASSWORD;
        iniciarConfirmacion(); // abrir ventana de confirmación
    }
}
```

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

DF DE CAMBIO DE USUARIO

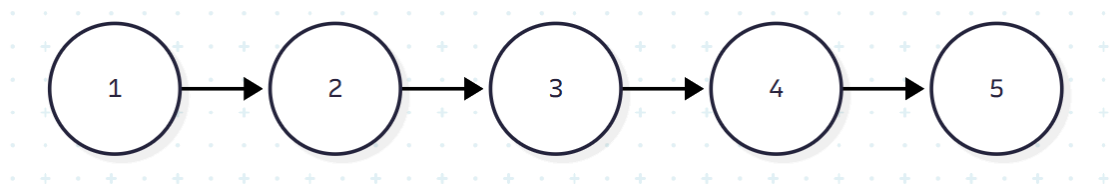


#### DF DE CAMBIO DE CONTRASEÑA

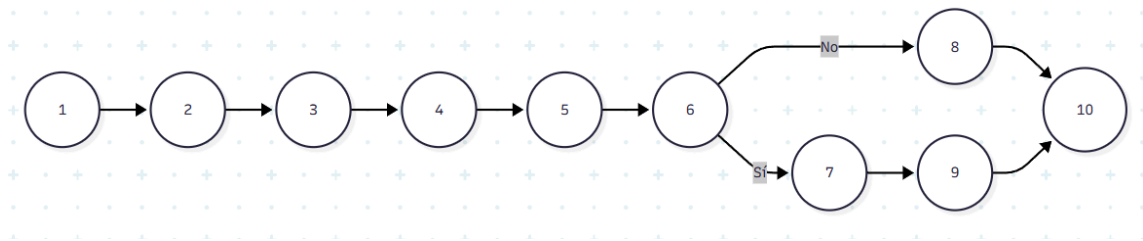


### 3. GRAFO DE FLUJO (GF)

Realizar un GF en base al DF del numeral



Grafo de proceso de cambio de usuario



Grafo de proceso de cambio de contraseña

### 4. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

RUTAS GRAFO CAMBIO DE USUARIO

R1: 1-2-3-4-5

RUTAS GRAFO CAMBIO DE CONTRASEÑA

R1: 1-2-3-4-5-6-8-10

R2: 1-2-3-4-5-6-7-9-10

### 5. COMPLEJIDAD CICLOMÁTICA

CAMBIO DE USUARIO

- $V(G) = 0 + 1$   
 $V(G) = 1$
- $V(G) = A - N + 2$   
 $V(G) = 4 - 5 + 2$   
 $V(G) = 1$

CAMBIO DE CONTRASEÑA

- $V(G) = 1 + 1$   
 $V(G) = 2$
- $V(G) = A - N + 2$   
 $V(G) = 10 - 10 + 2$   
 $V(G) = 2$

DONDE:

P: Número de nodos prediado

A: Número de aristas

N: Número de nodos

# Prueba de Caja Negra

---

*“Diseño de sistema de gestión de contratos para una plaza comercial”*

## **Integrantes:**

Gallardo Vega Santiago Jose,  
Pérez Díaz David Ismael,  
Zambrano Cajas Isabela Valentina.

**Fecha: 2025-06-12**

## Partición de clases equivalentes

**TABLA 1 Requisito Funcional N°3 Plantillas de contratos con texto editable: Civil y Laboral.**

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
Nombre de las variables	contratoCivil=contratoCivil	Válido	Contrato Civil
	contratoLaboral=contratoLaboral	Válido	Contrato laboral

### Captura de la pantalla de la ejecución

#### Descripción de la elución de acuerdo a los casos de prueba de la tabla 1, casos pass (Ok) y fail (no OK)

El programa deberá tener una opción de "agregar contrato" donde se pueda seleccionar una plantilla de contrato con cuadros de texto editable, Se creará 2 plantillas de contratos editables de índole: Civil y Laboral que se basen en el marco legal del ámbito civil y del ministerio de relaciones laborales:

**Civil:** Comparecientes: Arrendataria y Arrendador/a: Nombre, RUC, Representate Legal, Cargo, Nacionalidad. Antecedentes: Descripción general del inmueble y ubicación, Plazo: Fecha de inicio y Fin de contrato, Canon de Arrendamiento: Valor mensual en dólares y forma de pago, Deposito en Garantía: Monto de garantía entregada en dólares., Domicilio: Correo electrónico de Arrendataria y Arrendador/a

**Laboral:** Ciudad y Fecha del contrato, Comparecientes: Nombre completo del Empleador, Cédula de ciudadanía Empleador, Nombre completo trabajador, Cédula de ciudadanía del Trabajador, Ciudad de domicilio de ambas partes, Objeto del Contrato: Cargo del Trabajador, Remuneración: Monto de remuneración mensual en letras y números, forma de pago. Plazo del Contrato: Fecha de inicio y Fin de contrato

Figura 1 Gestión de contratos pantalla principal



A la hora de realizar de manera correcta la validación de credenciales del requisito funcional 1 se podrá visualizar una pantalla principal como se observa en la figura 1, esta nos dara la opción de escoger entre un contrato laboral o civil.

Figura 2 Contrato Civil





A la hora de escoger el botón civil este nos mandara a un formulario del Contrato civil como se visualiza en la figura 2

*Figura 3 Contrato Laboral*

**Contrato Laboral:**

Ciudad:  Fecha de contrato:

**Comparecientes:**

**Empleador**

Nombre completo:

Cedula de ciudadanía:  Ciudad de domicilio:

**Trabajador**

Nombre completo:

Cedula de ciudadanía:  Ciudad de domicilio:

**Objeto del contrato:**

Objeto/Cargo del trabajador:

**Jornada Laboral:**

HORAS DE JORNADA:  DIAS DE TRABAJO:  TIPO DE JORNADA:

**Remuneración**

Monto de remuneración mensual (números y letras):  Forma de pago:

**Plazo:**

Fecha de inicio de contrato:

**Lugar de trabajo:**

[Guardar](#) [Enviar](#) [Regresar](#)

Si se escoge el botón laboral este nos mandara a un formulario del Contrato laboral como se visualiza en la figura 3.

# Prueba de Caja Blanca

---

*“Diseño de sistema de gestión de contratos para una plaza comercial”*

**Integrantes:**

Gallardo Vega Santiago Jose,  
Pérez Díaz David Ismael,  
Zambrano Cajas Isabela Valentina.

**Fecha: 2025-06-12**

## Prueba caja blanca

### Requisito funcional 3: Plantillas de contratos con texto editable: Civil y Laboral.

El programa deberá tener una opción de "agregar contrato" donde se pueda seleccionar una plantilla de contrato con cuadros de texto editable. Al utilizar la opción de "nuevo contrato" se podrá escoger una plantilla e ingresar datos (que dependen del tipo de contrato) en cuadros de texto editables que podrán ser visualizados en una interfaz gráfica.

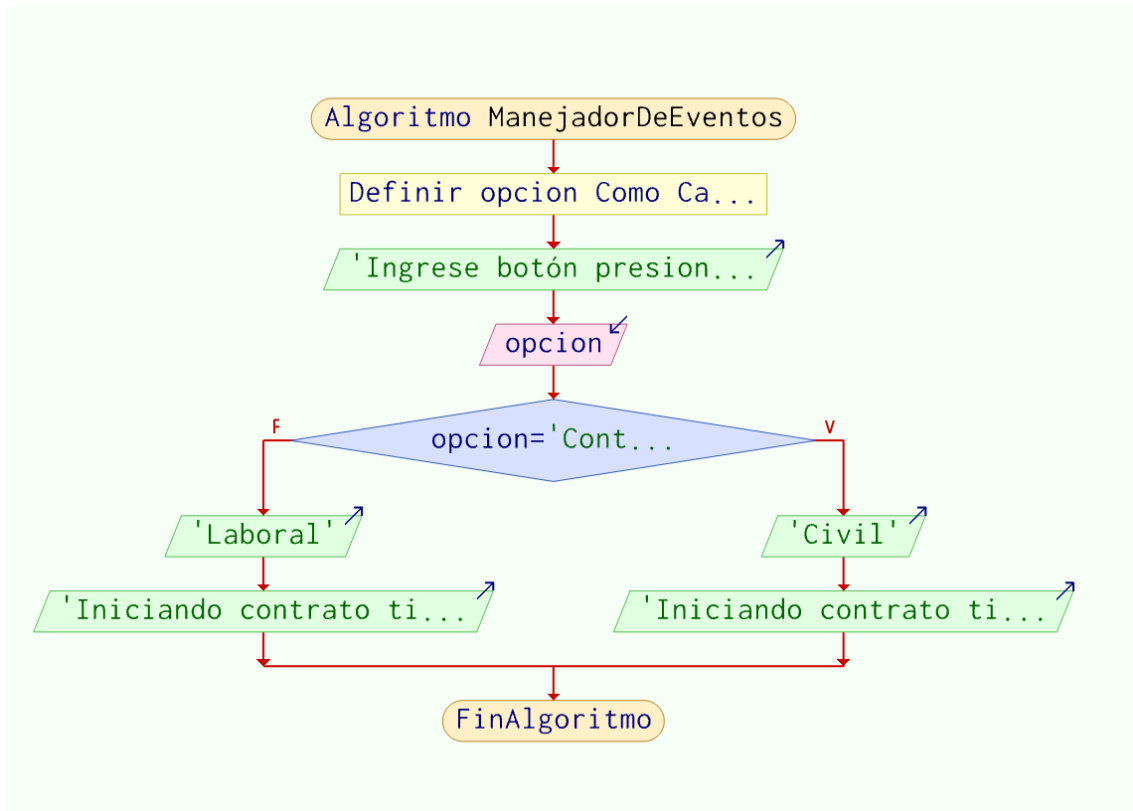
#### 1. CÓDIGO FUENTE

Pegar el trozo de código fuente que se requiere para el caso de prueba

```
}  
else if(e.getSource()==programa.BtnContratoCivil) {  
    System.out.println("Civil");  
    iniciarContrato(0);  
}  
else if(e.getSource()==programa.BtnContratoLaboral) {  
    System.out.println("Laboral");  
    iniciarContrato(1);  
}  
else if(e.getSource()==programa.BtnDatosUsuario) {  
    System.out.println("Boton Datos Usuario");  
    userctrl.iniciarVistaDatos();  
}
```

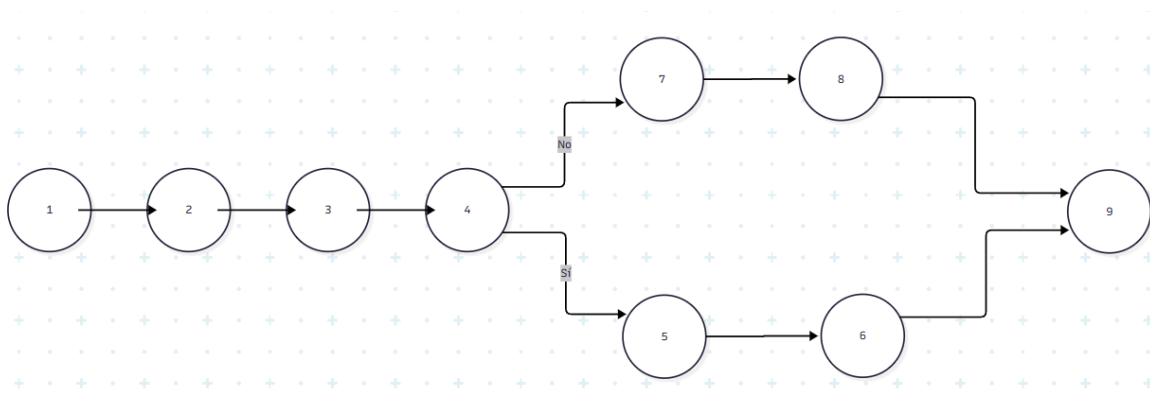
## 2. DIAGRAMA DE FLUJO (DF)

Realizar un DF del código fuente del numeral 1



## 3. GRAFO DE FLUJO (GF)

Realizar un GF en base al DF del numeral



#### **4. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)**

Determinar en base al GF del numeral 4  
RUTAS

R1: 1-2-3-4-7-8-9

R2: 1-2-3-4-5-6-9

#### **5. COMPLEJIDAD CICLOMÁTICA**

Se puede calcular de las siguientes formas:

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

$$V(G)= 2$$

$$V(G) = A - N + 2$$

$$V(G)= 9-9+2$$

$$V(G)=2$$

DONDE:

P: Número de nodos predado

A: Número de aristas

N: Número de nodos