

Prueba de Caja Blanca

“GESTIÓN DE FIDEICOMISOS:”

Integrantes:

Kevin Andino

Luis Calle

Denisse Quishpe

Madellyn

Tasipanta

Fecha: 2025/12/08

CONTROL DE VERSIONAMIENTO DE PRUEBAS CB

Versión	Fecha	Responsable	Aprobado por
PCB_V1.1.1.docx	08/12/2025	Denisse Quishpe Madellyn Tasipanta	

Prueba caja blanca SPRINT 2

1. CÓDIGO FUENTE

REQ003: Ingresar datos del contrato

El administrador debe ingresar los datos obligatorios del contrato en un formulario para que el sistema pueda visualizar las dos categorías legales principales del mismo.

REQ004: Búsqueda de contratos

El administrador debe poder buscar contratos registrados de Fideicomisos mediante filtros como cédula, RUC y tipo, para facilitar una búsqueda rápida y obtener resultados consistentes.

REQ005: Edición de contratos

El administrador debe acceder a un botón "Editar" y modificar el contrato, con un sistema de control de versiones que registre automáticamente fecha, usuario y permita acceder al historial de versiones.

REQ006: Eliminación de contratos

El administrador debe seleccionar un contrato, ejecutar la opción "Eliminar" y confirmar la acción para que el contrato sea eliminado definitivamente del sistema.

```
79 |         if (nombre.isEmpty() || banco.isEmpty() || cuenta.isEmpty() ||
80 |             cedula.isEmpty() || correo.isEmpty() || valorStr.isEmpty()) {
81 |             view.mostrarMensaje("Error: Todos los campos son obligatorios.");
82 |             return; // Detiene el proceso
83 |         }
```

De SolicitudRescateController.java

```
86 |         if (!Validador.esSoloNumeros(cuenta)) {
87 |             view.mostrarMensaje("Error: El número de cuenta solo debe contener dígitos numéricos.");
88 |             return;
89 |         }
```

De SolicitudRescateController.java

```
92 |         if (!Validador.esCedulaValida(cedula)) {
93 |             view.mostrarMensaje("Error: La cédula ingresada NO es válida (formato ecuatoriano incorrecto).");
94 |             return;
95 |         }
```

De SolicitudRescateController.java

```
98 |         if (!Validador.esCorreoValido(correo)) {
99 |             view.mostrarMensaje("Error: El correo electrónico no tiene un formato válido (ej: usuario@dominio.com).");
100 |             return;
101 |         }
```

De SolicitudRescateController.java

```
104 |         if (!Validador.esMontoValido(valorStr)) {
105 |             view.mostrarMensaje("Error: El valor a restituir debe ser un número positivo válido.");
106 |             return;
107 |         }
```

De ListaContratosController.java

```
93 |         if (textoId.isEmpty()) {
94 |             // Si la caja de texto está vacía, nos comportamos como el filtro normal
95 |             filtrarSoloPorCategoria();
96 |             return;
97 |         }
```

```

110         if (coincideTexto && coincideTipo) {
111             listaFiltrada.add(c);
112         }

```

De ListaContratosController.java

```

116         if (listaFiltrada.isEmpty()) {
117             JOptionPane.showMessageDialog(view, "No se encontró ese ID en la categoría seleccionada.");
118         }

```

De DashboardController.java

```

52     if (id != null && !id.isEmpty()) {
53         // A. Buscamos el contrato como GENÉRICO
54         ContratoBase contratoEncontrado = GestorContratos.getInstancia().buscarContrato(id.trim());
55         if (contratoEncontrado != null) {
56             // B. DETECTAMOS QUÉ TIPO ES PARA ABRIR LA VENTANA CORRECTA
57             if (contratoEncontrado instanceof SolicitudRescateModel) {
58                 // TIPO 1: RESCATE PARCIAL (AZUL)
59                 SolicitudRescateView viewEdicion = new SolicitudRescateView(view);
60                 new SolicitudRescateController(viewEdicion, (SolicitudRescateModel) contratoEncontrado);
61                 viewEdicion.setVisible(true);
62             } else if (contratoEncontrado instanceof SolicitudRescateGenericoModel) {
63                 // TIPO 2: RESCATE EXCEL (BLANCO)
64                 SolicitudRescateGenericoView viewEdicion = new SolicitudRescateGenericoView(view);
65                 new SolicitudRescateGenericoController(viewEdicion, (SolicitudRescateGenericoModel) contratoEncontrado);
66                 viewEdicion.setVisible(true);
67             } else if (contratoEncontrado instanceof SolicitudDesvinculacionModel) {
68                 // TIPO 3: DESVINCULACIÓN (ROJO)
69                 SolicitudDesvinculacionView viewEdicion = new SolicitudDesvinculacionView(view);
70                 new SolicitudDesvinculacionController(viewEdicion, (SolicitudDesvinculacionModel) contratoEncontrado);
71                 viewEdicion.setVisible(true);
72             } else if (contratoEncontrado instanceof AutorizacionDebitoModel) {
73                 // TIPO 4: DÉBITO MENSUAL (OSCURO) <--- NUEVO
74                 AutorizacionDebitoView viewEdicion = new AutorizacionDebitoView(view);
75                 new AutorizacionDebitoController(viewEdicion, (AutorizacionDebitoModel) contratoEncontrado);
76                 viewEdicion.setVisible(true);
77             }

```

De DashboardController.java

```

91     if (id != null && !id.isEmpty()) {
92         ContratoBase contrato = GestorContratos.getInstancia().buscarContrato(id.trim());
93
94         if (contrato != null) {
95             int confirmacion = JOptionPane.showConfirmDialog(view,
96                 "¿Está seguro de eliminar el contrato de " + contrato.getNombreCliente() + "?\nEsta acción no se puede deshacer",
97                 "Confirmar Eliminación", JOptionPane.YES_NO_OPTION);
98
99             if (confirmacion == JOptionPane.YES_OPTION) {
100                 GestorContratos.getInstancia().eliminarContrato(id.trim());
101                 JOptionPane.showMessageDialog(view, "Contrato eliminado correctamente.");
102             }

```

De DashboardController.java

2. DIAGRAMA DE FLUJO (DF)

Algoritmo SistemaCRUDContratos

=====

VARIABLES GLOBALES

=====

Definir nombre, banco,...

Definir textoId, selec...

Definir contratoEncont...

Definir confirmacion C...

Definir esNumero, esCe...

=====

PARTE 1: INGRESAR CONT...

=====

=== INGRESAR NUEVO CO...

'Ingrese nombre:'

nombre

'Ingrese banco:'

banco

'Ingrese número de cue...

cuenta

'Ingrese cédula:'

cedula

'Ingrese correo:'

correo

'Ingrese valor:'

valorStr

IF 1: ¿Campos vacíos?

nombre="" 0 ...

IF 2: ¿Cuenta es solo ...

¿? Todos los campos so...

esNumero = Verdadero

Simulación de validación

esNumero=False

IF 3: ¿Cédula válida?

esCedulaValida = Verda...

Simulación de validación

esCedulaVali...

IF 4: ¿Correo válido?

esCorreoValido = Verda...

Simulación de validación

esCorreoVali...

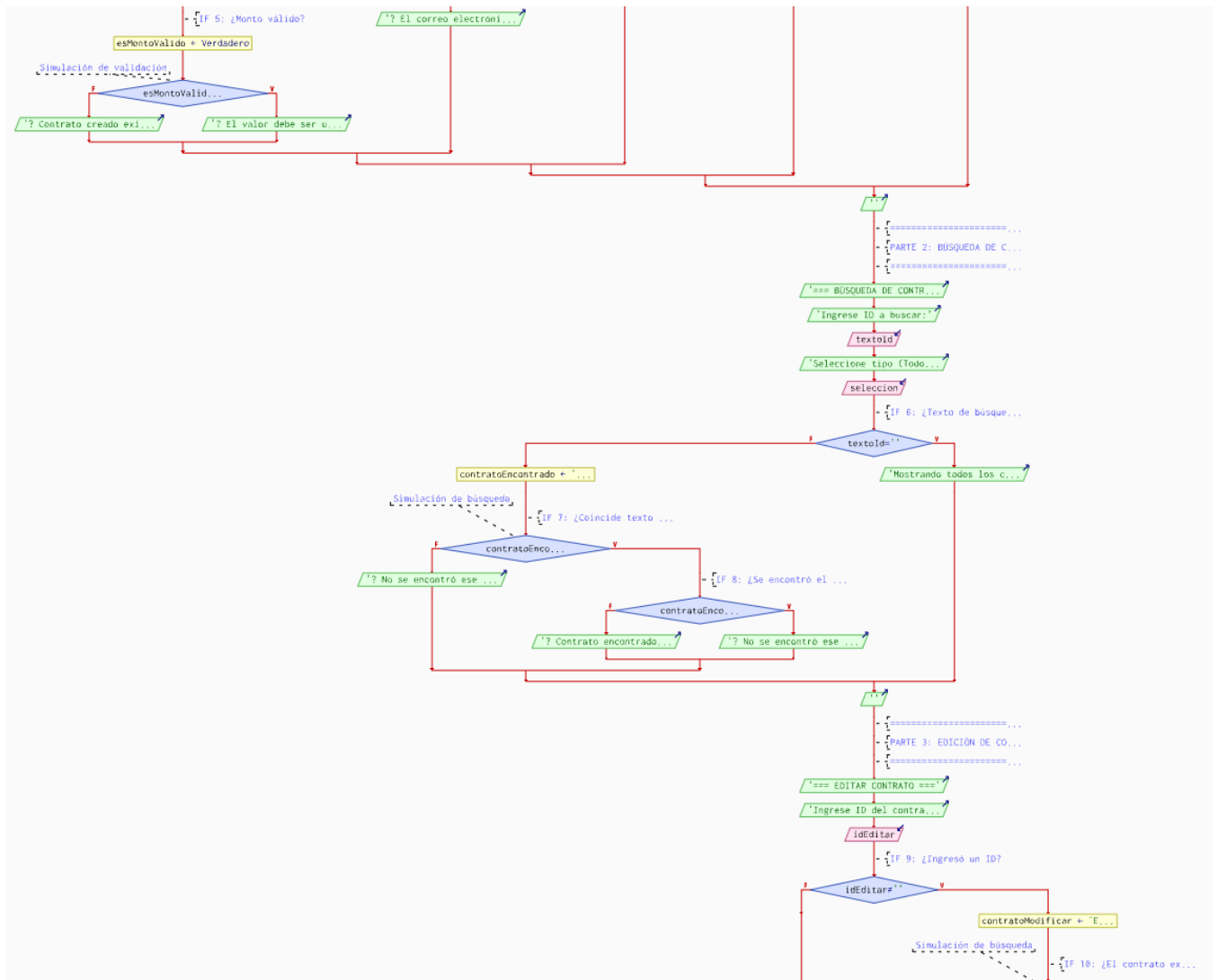
IF 5: ¿Monto válido?

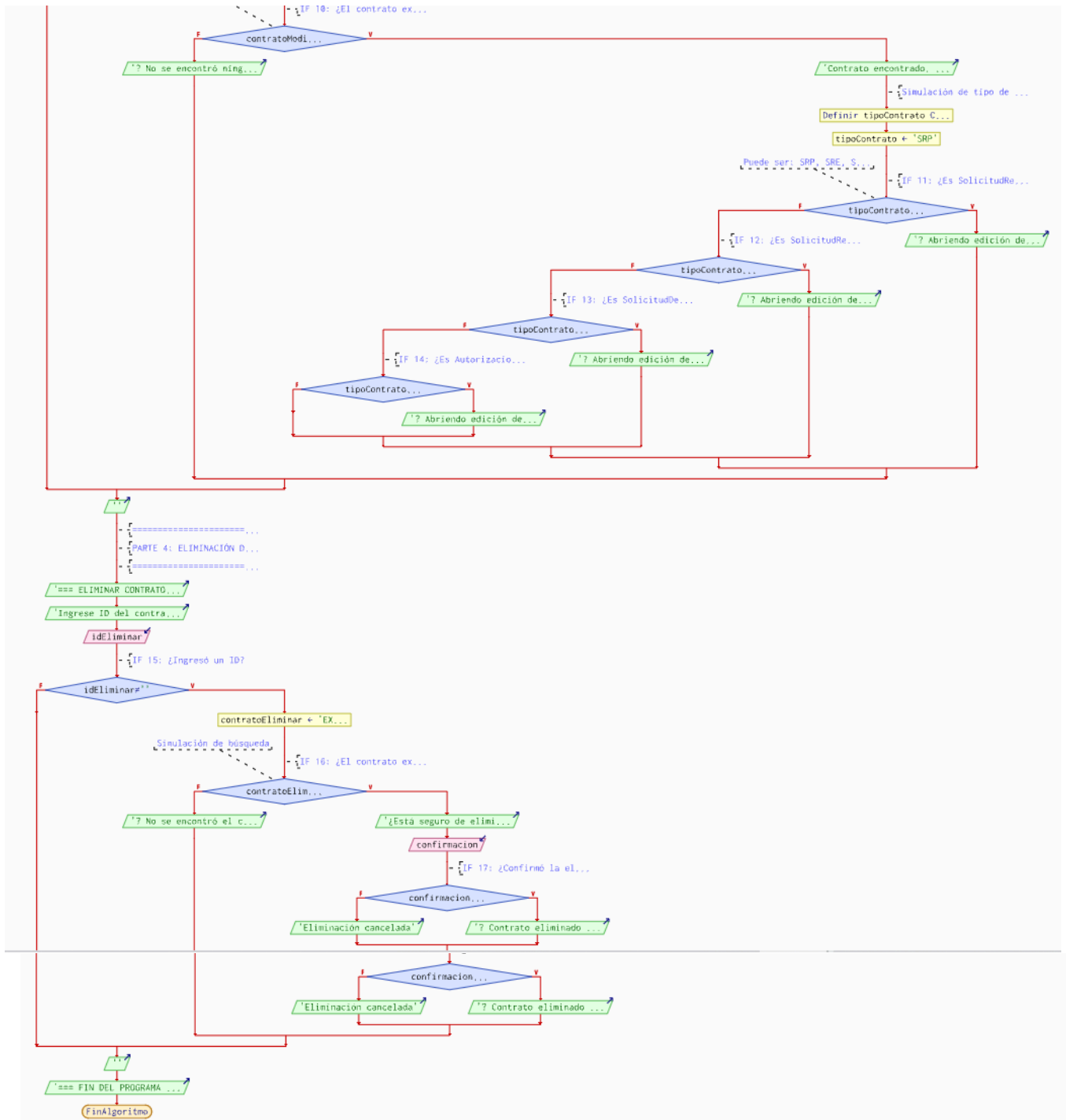
¿? El correo electróni...

¿? El número de cuenta...

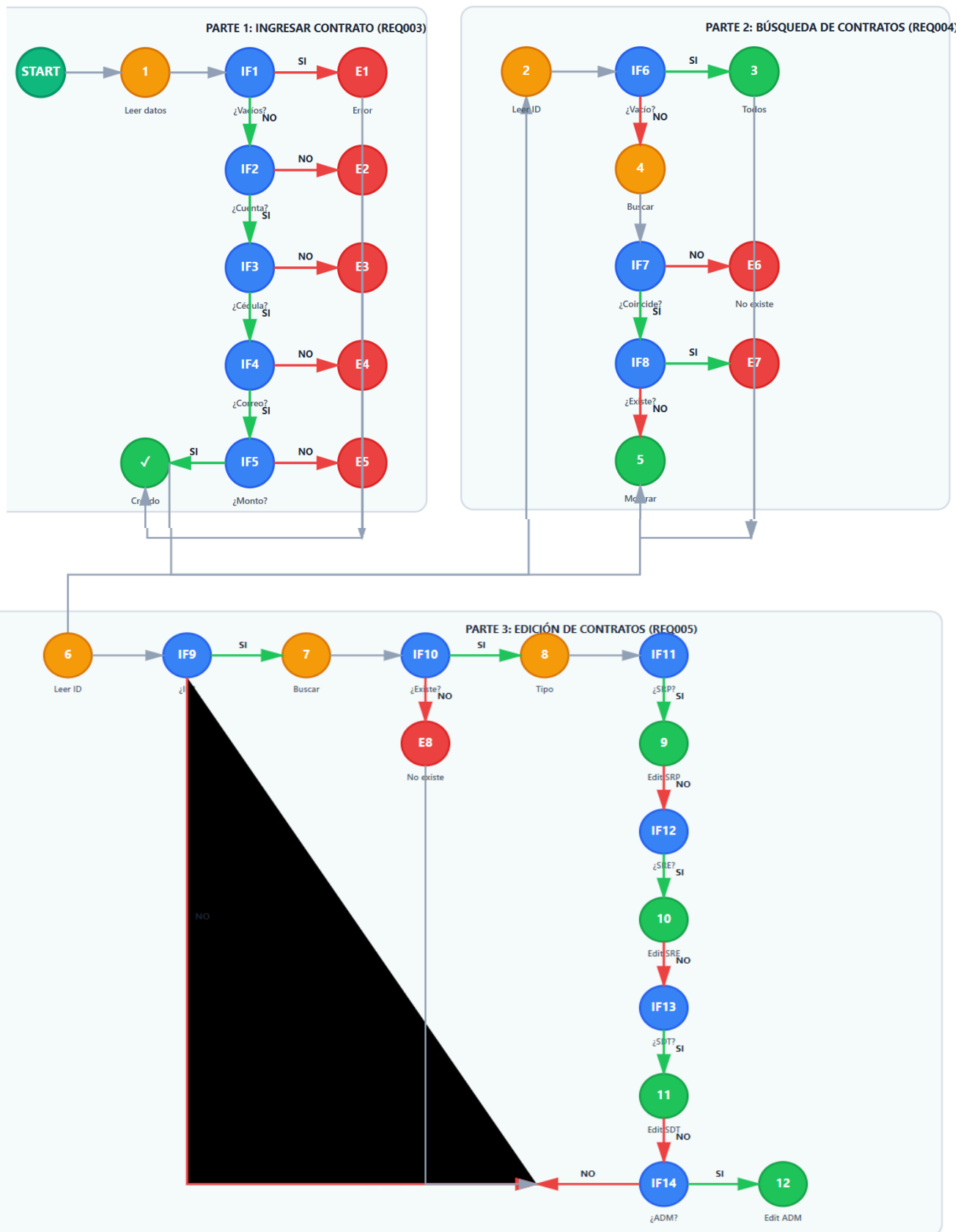
¿? La cédula ingresada...

¿? El correo electróni...





3. GRAFO DE FLUJO (GF)



4. IDENTIFICACIÓN DE LAS RUTAS (Camino básico)

Determinar en base al GF del numeral 4

RUTAS

R1: 1→2→3→4S→5→15→16→17→18S→19→26→27→28→29N→42→43→44→45N→53→54→55

R2: 1→2→3→4N→6N→7→15→16→17→18S→19→26→27→28→29N→42→43→44→45N→53→54→55

R3: 1→2→3→4N→6S→8N→9→15→16→17→18S→19→26→27→28→29N→42→43→44→45N→53→54→55

R4:

1→2→3→4N→6S→8S→10N→11→15→16→17→18S→19→26→27→28→29N→42→43→44→45N→53→54→55

R5:

1→2→3→4N→6S→8S→10S→12N→13→15→16→17→18S→19→26→27→28→29N→42→43→44→45N→53→54→55

R6:

1→2→3→4N→6S→8S→10S→12S→14→15→16→17→18S→19→26→27→28→29N→42→43→44→45N→53→54→55

R7:

1→2→3→4N→6S→8S→10S→12S→14→15→16→17→18S→19→26→27→28→29N→42→43→44→45N→53→54→55

R8:

1→2→3→4N→6S→8S→10S→12S→14→15→16→17→18N→20→21N→25→26→27→28→29N→42→43→44→45N→53→54→55

R9:

1→2→3→4N→6S→8S→10S→12S→14→15→16→17→18N→20→21S→22S→23→26→27→28→29N→42→43→44→45N→53→54→55

R10:

1→2→3→4N→6S→8S→10S→12S→14→15→16→17→18N→20→21S→22N→24→26→27→28→29N→42→43→44→45N→53→54→55

R11:

1→2→3→4N→6S→8S→10S→12S→14→15→16→17→18N→20→21S→22N→24→26→27→28→29N→42→43→44→45N→53→54→55

R12:

1→2→3→4N→6S→8S→10S→12S→14→15→16→17→18N→20→21S→22N→24→26→27→28→29S→30→31N→41→42→43→44→45N→53→54→55

R13:

1→2→3→4N→6S→8S→10S→12S→14→15→16→17→18N→20→21S→22N→24→26→27→28→29S→30→31S→32→33S→34→42→43→44→45N→53→54→55

R14:

1→2→3→4N→6S→8S→10S→12S→14→15→16→17→18N→20→21S→22N→24→26→27→28→29S→30→31S→32→33N→35S→36→42→43→44→45N→53→54→55

R15:

1→2→3→4N→6S→8S→10S→12S→14→15→16→17→18N→20→21S→22N→24→26→27→28→29S→30→31S→32→33N→35N→37S→38→42→43→44→45N→53→54→55

R16:

1→2→3→4N→6S→8S→10S→12S→14→15→16→17→18N→20→21S→22N→24→26→27→28→29S→30→31S→32→33N→35N→37N→39S→40→42→43→44→45N→53→54→55

R17:

1→2→3→4N→6S→8S→10S→12S→14→15→16→17→18N→20→21S→22N→24→26→27→28→29S→30→31S→32→33S→34→42→43→44→45S→46→47S→48→49S→50→53→54→55

R18:

1→2→3→4N→6S→8S→10S→12S→14→15→16→17→18N→20→21S→22N→24→26→27→28→29S→30→31S→32→33S→34→42→43→44→45S→46→47S→48→49N→51→53→54→55

5. COMPLEJIDAD CICLOMÁTICA

DONDE

- P: Número de nodos prediado
- A: Número de aristas
- N: Número de nodos

Se puede calcular de las siguientes formas:

Nodos (N): 55

Nodos Predicados (P): 17

Aristas (A): 71

- $V(G) = P + 1$
- $V(G) = 17 + 1 = 18$
- $V(G) = A - N + 2$
- $V(G) = 71 - 55 + 2 = 18$