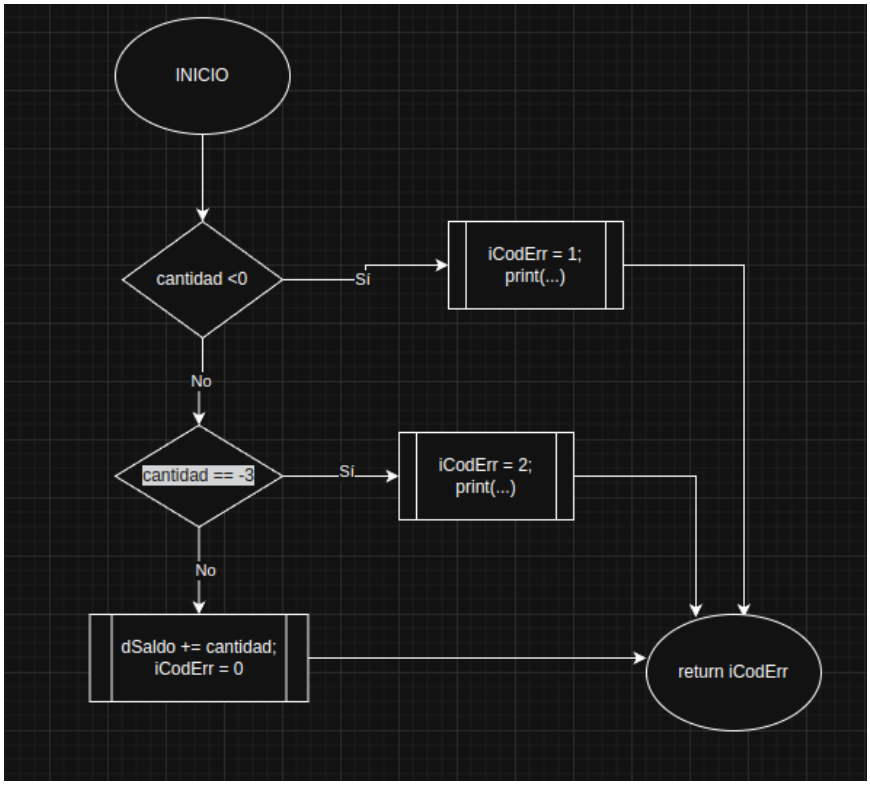
1. Realiza un análisis de caja blanca completo del método ingresar. Crea el grafo a mano y lo escaneas para subirlo o créalo con la herramienta que más fácil te resulte.

```
public int ingresar(double cantidad) {
   int iCodErr;
   if (cantidad < 0) {
      System.out.println("No se puede ingresar una cantidad negativa");
      iCodErr = 1;
   } else if (cantidad == -3) {
      System.out.println("Error detectable en pruebas de caja blanca");
      iCodErr = 2;
   } else {
      dSaldo = dSaldo + cantidad;
      iCodErr = 0;
   }
   return iCodErr;
}
```



2. Realiza un análisis de caja negra, incluyendo valores límite y conjetura de errores del método retirar. Debes considerar que este método recibe como parámetro la cantidad a retirar, que no podrá ser menor a 0. Además, en ningún caso esta cantidad podrá ser mayor al saldo actual. Al tratarse de pruebas funcionales no es necesario conocer los detalles del código, pero te lo pasamos para que lo tengas.

```
public void retirar(double cantidad) {
if (cantidad <= 0) {
System.out.println("No se puede retirar una cantidad negativa");
} else if (dSaldo < cantidad) {
System.out.println("No se hay suficiente saldo");
} else {
dSaldo = dSaldo - cantidad;
}
}
```

CONDICIONES DE ENTRADA

- Cantidad >0
- Cantidad <=dSaldo

PARTICIONES EQUIVALENTES

Identificamos grupos de valores que se comporten de manera diferente unos de otros.

1. Grupo 1: Cantidad<=0. En este caso mostramos mensaje de que no se puede retirar y el saldo no cambia. Lo llamaremos Clase de Equivalencia Inválida 1 (CEI1)
2. Grupo 2: Cantidad >0 y mayor que el saldo actual. Mostramos mensaje no hay suficiente saldo, el saldo no cambia. Clase CEI2
3. Grupo 3: Válido, todo lo demás. No mostramos mensaje y el saldo disminuye en el valor de cantidad. Clase de Equivalencia válida (CEV)

ANÁLISIS DE VALORES LÍMITE

Los límites están alrededor de 2 valores:

1. Borde alrededor de 0: 0(CE1), -0.01(CE1 y +0.01(CE2 o CV)
2. Borde alrededor de dSaldo: Por ejemplo si el saldo es 100… 100(CV),99.99(CV) y 100.01(CE2)

CONJETURA DE ERRORES

Podemos suponer el caso de que el saldo esté a 0 o que haya deudas(saldo negativo).

En ese caso, intentar retirar daría error de muchas maneras diferentes. Si intentas retirar 0 da error, y si intentas retirar de 0.01 en adelante da otro tipo de error posiblemente. Llamémos a esta situación CE3

3. Crea la clase CCuentaTest del tipo Caso de prueba JUnit en Eclipse que nos permita pasar las pruebas unitarias de caja blanca del método ingresar. Los casos de prueba ya los habrás obtenido en el primer apartado del ejercicio. Copia el código fuente de esta clase en el documento.

```
import static org.junit.Assert.*;
import org.junit.Before;
import org.junit.Test;

public class CCuentaTest {
private CCuenta cuenta;

@Before
```

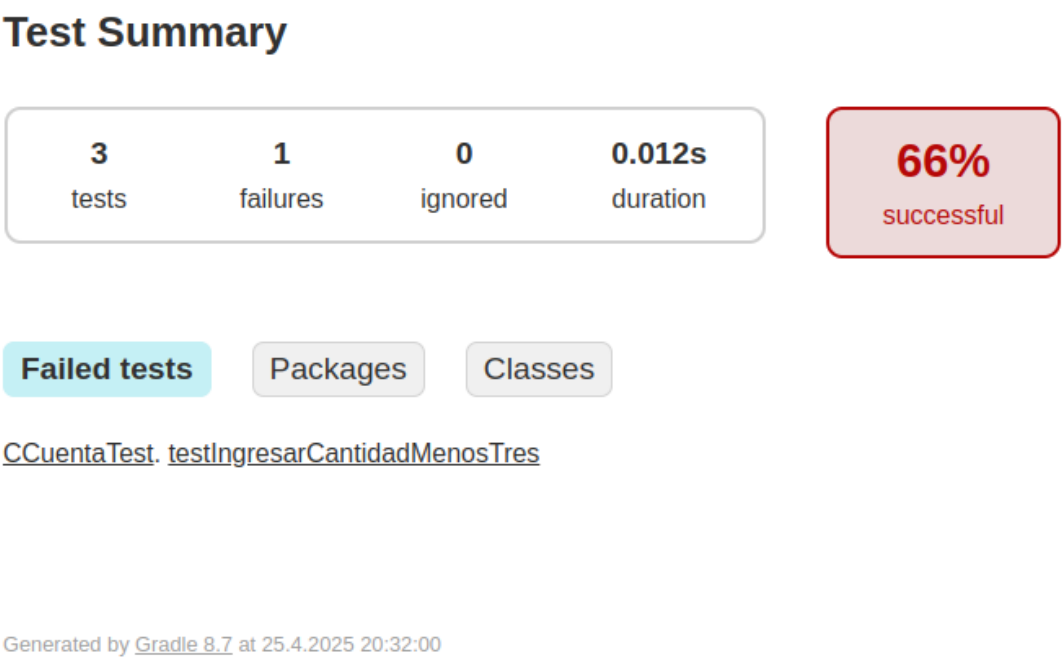
```
public void setUp() {
cuenta = new CCuenta();
}

@Test
public void testIngresarCantidadNegativa() {
int resultado = cuenta.ingresar(-100);
assertEquals(1, resultado);
}

@Test
public void testIngresarCantidadMenosTres() {
int resultado = cuenta.ingresar(-3);
assertEquals(2, resultado);
}

@Test
public void testIngresarCantidadPositiva() {
int resultado = cuenta.ingresar(100);
assertEquals(0, resultado);
assertEquals(100, cuenta.dSaldo, 0.001);
}
}
```

Si ejecutamos los test y abrimos el reporte en el navegador vemos:



4. Genera los siguientes puntos de ruptura para validar el comportamiento del método ingresar en modo depuración.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<breakpoints xmlns="http://eclipse.org/debug.core/breakpoints" version="1.0">
```

- Punto de parada sin condición al crear el objeto miCuenta en la función main. Línea 3 del código del método main.

```
<breakpoint enabled="true" hit_count="-1" id="org.eclipse.jdt.debug.javaLineBreakpoint:..."
markerType="org.eclipse.jdt.debug.javaLineBreakpointMarker" persist="true" register="true"
resource_path="/Tarea-sw Alumnos/src/main/java/CCuenta.java" typeName="CCuenta">
<marker>
<attrib name="org.eclipse.debug.core.enabled" value="true"/>
<attrib name="org.eclipse.jdt.debug.core.hitCount" value="-1"/>
<attrib name="org.eclipse.jdt.debug.core.condition" value=""/>
<attrib name="org.eclipse.jdt.debug.core.conditionEnabled" value="false"/>
<attrib name="org.eclipse.jdt.debug.core.suspendPolicy" value="0"/>
<attrib name="org.eclipse.jdt.debug.core.lineBreakpoint.lineNumber" value="6"/>
<attrib name="org.eclipse.jdt.debug.core.typeName" value="CCuenta"/>
</marker>
</breakpoint>
```

- Punto de parada en la instrucción return del método ingresar sólo si la cantidad a ingresar es menor de 0. Línea 20 del código del método ingresar.

```
<breakpoint enabled="true" hit_count="-1" id="org.eclipse.jdt.debug.javaLineBreakpoint:..."
markerType="org.eclipse.jdt.debug.javaLineBreakpointMarker" persist="true" register="true"
resource_path="/Tarea-sw Alumnos/src/main/java/CCuenta.java" typeName="CCuenta">
<marker>
<attrib name="org.eclipse.debug.core.enabled" value="true"/>
<attrib name="org.eclipse.jdt.debug.core.hitCount" value="-1"/>
<attrib name="org.eclipse.jdt.debug.core.condition" value="cantidad &lt; 0"/>
<attrib name="org.eclipse.jdt.debug.core.conditionEnabled" value="true"/>
<attrib name="org.eclipse.jdt.debug.core.suspendPolicy" value="0"/>
<attrib name="org.eclipse.jdt.debug.core.lineBreakpoint.lineNumber" value="41"/>
<attrib name="org.eclipse.jdt.debug.core.typeName" value="CCuenta"/>
</marker>
</breakpoint>
```

- Punto de parada en la instrucción donde se actualiza el saldo, sólo deberá parar la tercera vez que sea actualizado. Línea 16 del código del método ingresar.

```
<breakpoint enabled="true" hit_count="3" id="org.eclipse.jdt.debug.javaLineBreakpoint:..."
markerType="org.eclipse.jdt.debug.javaLineBreakpointMarker" persist="true" register="true"
resource_path="/Tarea-sw Alumnos/src/main/java/CCuenta.java" typeName="CCuenta">
<marker>
<attrib name="org.eclipse.debug.core.enabled" value="true"/>
<attrib name="org.eclipse.jdt.debug.core.hitCount" value="3"/>
<attrib name="org.eclipse.jdt.debug.core.condition" value=""/>
<attrib name="org.eclipse.jdt.debug.core.conditionEnabled" value="false"/>
<attrib name="org.eclipse.jdt.debug.core.suspendPolicy" value="0"/>
<attrib name="org.eclipse.jdt.debug.core.lineBreakpoint.lineNumber" value="38"/>
<attrib name="org.eclipse.jdt.debug.core.typeName" value="CCuenta"/>
</marker>
</breakpoint>
</breakpoints>
```