1. Realiza un **análisis de caja blanca** completo del método ingresar.

El método ingresar tiene un parámetro cantidad que es del tipo double. Tenemos 3 casos posibles:

* En el caso de que sea menor que 0, nos mostrará un mensaje (No se puede ingresar una cantidad negativa) y setea el valor de la variable **iCodErr** a -1
* En el caso de que sea -3, nos mostrará un mensaje (Error detectable en pruebas de caja blanca) y setea el valor de la variable **iCodErr** a 2. Este caso, es muy concreto, ya que sin ver el código (caja negra) no podríamos detectarlo.
* Si no se cumple ninguno de los anteriores casos, no se mostrará ningún mensaje) y setea el valor de la variable **iCodErr** a 0.

Aunque el 2º caso no se ejecutara nunca ya que entrara en el 1º caso.

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.

Los valores a probar serían los siguientes:

* Numero positivo, siendo mayor que el saldo actual: no debería dejarnos retirar la cantidad indicada.
* Numero positivo, siendo menor que el saldo actual: debería dejarnos retirar la cantidad indicada.
* Numero positivo, siendo igual al saldo actual: debería dejarnos retirar la cantidad indicada.
* Valor 0: deberíamos esperar que no nos retirara nada de nuestra cuenta ya que el valor no es menor que 0.
* Numero negativo: deberíamos esperar que nos indique que no podemos retirar por ser el valor negativo.

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.

@ParameterizedTest

@CsvSource({"-10,1","-3,2","10,0"})

@DisplayName("Caja Blanca - Ingresar")

**void** testIngreso(**double** cant,**int** resul) {

*assertEquals*(resul,miCuenta.ingresar(cant));

}

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

* Punto de parada sin condición al crear el objeto miCuenta en la función main. Línea 3 del código del método main que se presenta en la siguiente página de este libro.
* 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 que se presenta más adelante.
* Punto de parada en la instrucción donde se actualiza el saldo, sólo deberá parar la tercera vez que sea actualizado. Línea

|  |
| --- |
| Pulsando el botón derecho sobre la ventana de puntos de ruptura selecciona la opción "Exportar puntos de ruptura". Seleccionas los tres puntos de ruptura generados y guardas el fichero. El fichero tendrá la extensión bkpt, la cambias por txt. Ahora abres el fichero y copias el contenido íntegramente al documento. |

<?xml version="1.0" encoding="UTF-8"?>

<breakpoints>

<breakpoint enabled="true" persistant="true" registered="true">

<resource path="/CCuenta/src/CCuenta.java" type="1"/>

<marker charStart="113" lineNumber="5" type="org.eclipse.jdt.debug.javaLineBreakpointMarker">

<attrib name="charStart" value="113"/>

<attrib name="org.eclipse.jdt.debug.core.suspendPolicy" value="2"/>

<attrib name="org.eclipse.jdt.debug.ui.JAVA\_ELEMENT\_HANDLE\_ID" value="=CCuenta/src&lt;{CCuenta.java[CCuenta"/>

<attrib name="charEnd" value="151"/>

<attrib name="org.eclipse.debug.core.enabled" value="true"/>

<attrib name="message" value="Line breakpoint:CCuenta [line: 5] - main(String[])"/>

<attrib name="org.eclipse.debug.core.id" value="org.eclipse.jdt.debug"/>

<attrib name="org.eclipse.jdt.debug.core.typeName" value="CCuenta"/>

<attrib name="workingset\_name" value=""/>

<attrib name="workingset\_id" value="org.eclipse.debug.ui.breakpointWorkingSet"/>

</marker>

</breakpoint>

<breakpoint enabled="true" persistant="true" registered="true">

<resource path="/CCuenta/src/CCuenta.java" type="1"/>

<marker charStart="1600" lineNumber="45" type="org.eclipse.jdt.debug.javaLineBreakpointMarker">

<attrib name="charStart" value="1600"/>

<attrib name="org.eclipse.jdt.debug.core.suspendPolicy" value="2"/>

<attrib name="org.eclipse.jdt.debug.ui.JAVA\_ELEMENT\_HANDLE\_ID" value="=CCuenta/src&lt;{CCuenta.java[CCuenta"/>

<attrib name="charEnd" value="1636"/>

<attrib name="org.eclipse.debug.core.enabled" value="true"/>

<attrib name="message" value="Line breakpoint:CCuenta [line: 45] - ingresar(double)"/>

<attrib name="org.eclipse.debug.core.id" value="org.eclipse.jdt.debug"/>

<attrib name="org.eclipse.jdt.debug.core.typeName" value="CCuenta"/>

<attrib name="workingset\_name" value=""/>

<attrib name="workingset\_id" value="org.eclipse.debug.ui.breakpointWorkingSet"/>

</marker>

</breakpoint>

<breakpoint enabled="true" persistant="true" registered="true">

<resource path="/CCuenta/src/CCuenta.java" type="1"/>

<marker charStart="1761" lineNumber="50" type="org.eclipse.jdt.debug.javaLineBreakpointMarker">

<attrib name="charStart" value="1761"/>

<attrib name="org.eclipse.jdt.debug.core.suspendPolicy" value="2"/>

<attrib name="org.eclipse.jdt.debug.ui.JAVA\_ELEMENT\_HANDLE\_ID" value="=CCuenta/src&lt;{CCuenta.java[CCuenta"/>

<attrib name="charEnd" value="1784"/>

<attrib name="org.eclipse.debug.core.enabled" value="true"/>

<attrib name="message" value="Line breakpoint:CCuenta [line: 50] - ingresar(double)"/>

<attrib name="org.eclipse.debug.core.id" value="org.eclipse.jdt.debug"/>

<attrib name="org.eclipse.jdt.debug.core.typeName" value="CCuenta"/>

<attrib name="workingset\_name" value=""/>

<attrib name="workingset\_id" value="org.eclipse.debug.ui.breakpointWorkingSet"/>

</marker>

</breakpoint>

</breakpoints>