Nombre: Ayoze Pestano de la Rosa

Curso: 1 de Ciclo Superior de Desarrollo de Aplicaciones Web a distancia

#### ÍNDICE

- Introducción
- Objetivos
- Material empleado
- Desarrollo
- Conclusiones

#### <u>Introducción</u>

Se va a realizar la práctica 3 de la Asignatura de Entornos de desarrollo, en la que se nos pide realizar:

- 1. Análisis de caja blanca completo del método ingresar.
- 2. **Análisis de caja negra**, en donde se incluyan valores límite y la conjetura de errores del método retirar.
- 3. **Crear una clase CCuentaTest** del tipo JUnit en el IDE Eclipse que permita pasar pruebas unitarias de Caja blanca del método ingresar.
- 4. **Generar Puntos de ruptura** (Punto de parada sin condición, punto de parada en la instrucción return del método ingresar y punto de parada en la instrucción donde se actualiza el saldo)

#### Los **objetivos** de esta práctica son:

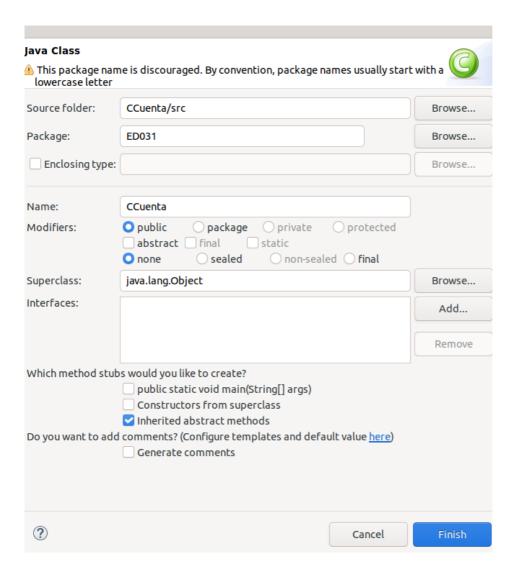
- Realizar varios análisis de un código de programación:
  - Análisis de caja blanca
  - Análisis de caja negra
  - Crear una clase para testear las pruebas de la caja blanca
  - Generar varios puntos de ruptura

Como material para elaborar dicha práctica se ha empleado:

- El IDE Eclipse, en su última versión
- El código de programación dado para dicha práctica (Método Main, Ingresar, Retirar)
- Un Editor de texto (Google Docs)

#### **Desarrollo**

Procedemos a la creación de la clase CCuenta y a pegar el código que se nos ha dado en el enunciado de la práctica:



#### 1. ANÁLISIS DE CAJA BLANCA:

El método ingresar posee un parámetro de 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 establece 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 blancfa) y establece el valor de la variable **iCodErr** a 2. Este caso es muy concreto, ya que sin ver el código (caja negra) no hubiéramos podido detectarlo.
- Si no se cumple ninguno de los anteriores casos, no se mostrará ningún mensaje y se establecerá el valor de la variable **iCodErr** a 0.

Aunque el segundo caso no se ejecutará nunca, ya que este entra en el primer caso.

#### 2. ANÁLISIS DE CAJA NEGRA

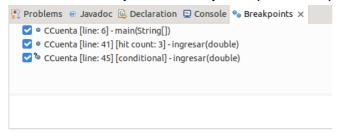
Teniendo en cuenta lo que es el análisis de caja negra, vamos a establecer los puntos de cómo el código debería de estar establecido de manera lógica teniendo en cuenta lo que es una Cuenta de ahorros, y lo que nos debería de devolver dicha cuenta:

- Número positivo, siendo mayor que el saldo actual: no debería dejarnos retirar la cantidad indicada.
- Número positivo, siendo menor que el saldo actual: debería dejarnos retirar la cantidad indicada.
- Número 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 es 0.
- Número negativo: deberíamos esperar que nos indique que no podemos retirar porque es un número negativo y se consideraría un INGRESO.

#### 3. CREACIÓN CLASE CCUENTATEST

	New JUnit Test Case ×		
ccu	JUnit Test Case  1 The use of the	default package is discouraged.	E
	○ New JUnit 3 test ○ New JUnit 4 test ○ New JUnit Jupiter test		
	Source folder:	CCuenta	Browse
	Package:	(default)	Browse
	Name:	CCuentaTest	
	Superclass:	java.lang.Object	Browse
	Do you want to a	@BeforeAll setUpBeforeClass()	rDown()
	Class under test:		Browse
[J	?	< Back Next > Cancel	Finish

Generamos los 3 puntos de ruptura que se nos pide en el enunciado de la práctica:



```
CCuenta.java
 ₽ Package Explorer
                                                                                                   CCuentaTest.java ×
                                                                                                                                                                                                                                                       □ □ B Outline ×
                                                                                     package ED03;
                           U ↑ nº □ □ ■ Q Q ■ ■ □ ✓ 1
 Finished after 0,11 seconds
                                                                                     import static org.junit.Assert.assertEquals;
                                                                                                                                                                                                                                                                    # ED03
                                                                                     import org.junit.jupiter.*;
import org.junit.jupiter.params.provider.*;
import org.junit.jupiter.params.*;
import org.junit.jupiter.api.DisplayName;
                                                                                                                                                                                                                                                                    Q CCuental
                         ☐ Errors: 0
                                                   □ Failures: 1
  Runs: 3/3
                                                                                                                                                                                                                                                                        △ miCue
    CCuentaTest [Runner: JUnit 5] (0,018 s)
                                                                                      class CCuentaTest {
    CCuenta miCuenta = new CCuenta();

√ Bi Caja Blanca - Ingresar (0,018 s)

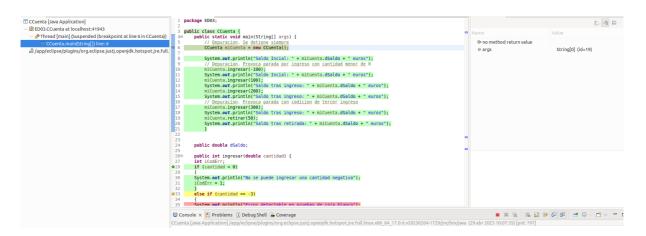
         ∰ [1] -10, 1 (0,018 s)
                                                                                           @ParameterizedTest
@CsvSource{{"-10,1","-3,2","10,0"}}
@DisplayName("Caja Blanca - Ingresar")
void testIngreso(double cant,int resul) {
    assertEquals(resul,micuenta.ingresar(cant));
         [3] 10, 0 (0,000 s)
                                                                  3 7 50
 J java.lang.AssertionError: expected:<2> but was:<1>
at ED03.CCuentaTest.testIngreso(CCuentaTest.java:16)
 at java.base/java.util.stream.ForEachOps$ForEachOp$OfRef.accept
at java.base/java.util.stream.ReferencePipeline$3$1.accept(Referen
■ × ¾ 🔒 🚮 🚱
a cjava.base/java.util.stream.kererencerpeimes5351.accept(kereren = derminated > CCuentaTest [Junit] /app/eclipse/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.6.v20230204-1729/jre/bin/java (29 abr 2023 9:56:20 – 9:56:20) [i at java.base/java.util.stream.ReferencePipeline5351.accept(Referen No se puede ingresar una cantidad negativa
■ at java.base/java.util.stream.ReferencePipeline$3$1.accept(Referen
■ at java.base/java.util.stream.ReferencePipeline$3$1.accept(Referen
at java base/java util ArrayListSArrayListSpliterator forEachRemain
 ■ at java.base/java.util.stream.ReferencePipeline$Head.forEach(Refer
at java.base/java.util.stream.ReferencePipeline$7$1.accept(Referen
```

Generamos el archivo .bkpt de dichos puntos de ruptura y con la ayuda de un editor de texto lo abrimos para observarlo y adjuntarlo:

```
<?xml version="1.0" encoding="UTF-8"?>
<bre>dreakpoints>
<bre>cbreakpoint enabled="true" persistant="true" registered="true">
<resource path="/CCuenta/src/ED03/CCuenta.java" type="1"/>
<marker charStart="116" lineNumber="6" type="org.eclipse.jdt.debug.javaLineBreakpointMarker">
<attrib name="charStart" value="116"/>
<attrib name="org.eclipse.jdt.debug.core.suspendPolicy" value="2"/>
                                                             name="org.eclipse.jdt.debug.ui.JAVA_ELEMENT_HANDLE_ID"
value="=CCuenta/src<ED03{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: 6] - main(String[])"/>
<attrib name="org.eclipse.debug.core.id" value="org.eclipse.jdt.debug"/>
<attrib name="org.eclipse.jdt.debug.core.typeName" value="ED03.CCuenta"/>
<attrib name="workingset name" value=""/>
<attrib name="workingset_id" value="org.eclipse.debug.ui.breakpointWorkingSet"/>
</marker>
</breakpoint>
<bre>cbreakpoint enabled="true" persistant="true" registered="true">
<resource path="/CCuenta/src/ED03/CCuenta.java" type="1"/>
<marker charStart="1223" lineNumber="41" type="org.eclipse.jdt.debug.javaLineBreakpointMarker">
<attrib name="charStart" value="1223"/>
<attrib name="org.eclipse.jdt.debug.core.suspendPolicy" value="2"/>
                                                             name="org.eclipse.jdt.debug.ui.JAVA ELEMENT HANDLE ID"
value="=CCuenta/src<ED03{CCuenta.java[CCuenta"/>
<attrib name="org.eclipse.jdt.debug.core.hitCount" value="3"/>
<attrib name="charEnd" value="1251"/>
<attrib name="org.eclipse.debug.core.enabled" value="true"/>
```

```
<attrib name="org.eclipse.jdt.debug.core.expired" value="false"/>
<a trib name="message" value="Line breakpoint:CCuenta [line: 41] [hit count: 3] - ingresar(double)"/>
<attrib name="org.eclipse.debug.core.id" value="org.eclipse.jdt.debug"/>
<attrib name="org.eclipse.jdt.debug.core.typeName" value="ED03.CCuenta"/>
<attrib name="workingset name" value=""/>
<attrib name="workingset id" value="org.eclipse.debug.ui.breakpointWorkingSet"/>
</marker>
</breakpoint>
<bre>cbreakpoint enabled="true" persistant="true" registered="true">
<re>ource path="/CCuenta/src/ED03/CCuenta.java" type="1"/>
<marker charStart="1334" lineNumber="45" type="org.eclipse.jdt.debug.javaLineBreakpointMarker">
<attrib name="org.eclipse.jdt.debug.core.conditionEnabled" value="true"/>
<attrib name="charStart" value="1334"/>
<attrib name="org.eclipse.jdt.debug.core.suspendPolicy" value="2"/>
<attrib name="org.eclipse.jdt.debug.core.condition" value="cantidad &lt; 0"/>
<attrib
                                                             name="org.eclipse.jdt.debug.ui.JAVA_ELEMENT_HANDLE_ID"
value="=CCuenta/src<ED03{CCuenta.java[CCuenta"/>
<attrib name="charEnd" value="1350"/>
<attrib name="org.eclipse.debug.core.enabled" value="true"/>
<a trib name="message" value="Line breakpoint:CCuenta [line: 45] [conditional] - ingresar(double)"/>
<attrib name="org.eclipse.debug.core.id" value="org.eclipse.jdt.debug"/>
<attrib name="org.eclipse.jdt.debug.core.typeName" value="ED03.CCuenta"/>
<attrib name="workingset name" value=""/>
<attrib name="workingset_id" value="org.eclipse.debug.ui.breakpointWorkingSet"/>
</marker>
</breakpoint>
</breakpoints>
```

Como último punto, procedemos a depurar el código con una depuración básica:



Como <u>conclusión</u> hemos sacado que el programa que se nos ha adjuntado en la práctica falla en 1 de los 3 puntos de ruptura establecidos, como indica la imagen adjuntada más arriba en este mismo documento.

Se han hecho varios análisis al código de programación y se ha concretado que **no está** listo para empezar a usar pues no ha pasado las pruebas pertinentes.