# INFORME PRÁCTICA 4: PRUEBAS UNITARIAS DE SOFTWARE

## Introducción

Este documento contiene la memoria de la práctica 4.

El proceso seguido para el desarrollo de las pruebas de caja negra es el siguiente:

1.-Desarrollo de las clases de equivalencia (válidas y no válidas).

2.-Creación de los casos de prueba válidos y no válidos.

3.-Implementación de los casos de prueba en código.

4.-Comprobación de resultados de casos de prueba.

5.-Corrección de errores.

En la práctica se pide usar las técnicas de clases de equivalencia y AVL para obtener valores interesantes, de esta forma, de cada valor que queramos probar deberemos obtener sus valores límites y un valor medio.

En el caso de las pruebas de caja blanca, la forma de crearlas es la siguiente:

1.-Análisis de las pruebas de caja negra ya probadas para comprobar la cobertura actual.

2.-Definir las pruebas de caja blanca necesarias para completar la cobertura deseada.

En la práctica se pide usar una cobertura de decisión/condición, por lo que en cada una de las sentencias condicionales del código se deberá probar cada una de las decisiones dentro de las sentencias a verdadero y falso y probar la sentencia completa a verdadero y falso también.

Para las pruebas de caja negra se ha usado JUnit para programar y ejecutar cada uno de los tests individuales.

<Breve descripción del proceso de pruebas general que se ha aplicado: fases, técnicas de prueba aplicadas, criterios de cobertura elegidos, herramientas utilizadas para cada tipo de prueba, etc.>

En las siguientes secciones (2, 3 y 4), explicar el detalle de la aplicación de las técnicas de prueba aplicadas para cada una de las clases involucradas y los resultados obtenidos.

* En el caso de pruebas de caja negra, incluir una tabla con las clases de equivalencia identificadas, los valores interesantes seleccionados y los casos de prueba definidos a partir de ellos (tanto válidos como no válidos).
* Para pruebas de caja blanca, criterios de cobertura aplicados y cobertura conseguida. Si ha sido necesario ampliar el conjunto de casos de prueba definidos en el paso anterior, indicar cuáles son los casos añadidos.
* Para pruebas de integración, especificación de los casos de prueba definidos (si se ha aplicado clases de equivalencia, exponerlas) y enumeración de los que se han implementado.
* En la medida de lo posible, listar los errores que se han detectado en el proceso de pruebas.

## Proceso de pruebas de la clase Empleado

**1.-Pruebas de caja negra**

**1.1.-Clases de equivalencia y valores interesantes identificados:**

|  |  |  |
| --- | --- | --- |
| **Parámetro** | **Clases Equivalencia Válidas** | **Clases Equivalencia No Válidas** |
| **Fecha** | **1.[Hoy-5 años, Hoy]**  **2.[Hoy-10 años, Hoy-5 años)**  **3.[Hoy-20 años, Hoy-10 años)**  **4.[Hoy-∞, Hoy-20 años)** | **7.{Hoy+1día, null}** |
| **Categoría** | **5.{DIRECTIVO/EJECUTIVO/OBRERO}** | **8.null, !{DIRECTIVO/EJECUTIVO/OBRERO}** |
| **Baja** | **6.{TRUE/FALSE}** | **9.null** |

**1.2.-Casos de prueba definidos:**

**CASOS DE PRUEBA VÁLIDOS**

1.[DIRECTIVO, Hoy, true]=1125€

2. [GESTOR, Hoy-3 años, false]=1200€

3. [OBRERO, Hoy-5 años+1día, true]=750€

4. [DIRECTIVO, Hoy-5 años, false]=1550€

5. [GESTOR, Hoy-7 años, true]=937.5€

6. [OBRERO, Hoy-10 años+1dia, false]=1050€

7. [GESTOR, Hoy-10años, true]=975€

8. [DIRECTIVO, Hoy-15 años, false]=1600€

9. [GESTOR, Hoy-20 años+1día, true]=975€

10. [GESTOR, Hoy-20 años, false]=1400€

11. [GESTOR, Hoy-50 años, true]=1050€

**CASOS DE PRUEBA NO VÁLIDOS**

12.[GESTOR, Hoy+1día, true]

13. [GESTOR, null, true]

14. [null, Hoy, true]

15. [PEPE, Hoy, true]

16. [GESTOR, Hoy, null]

(15 y 16 no probables en código, comentadas dentro de la clase EmpleadoTest)

**2.-Pruebas de caja blanca**

**2.1.-Criterios de cobertura**

El criterio de cobertura usado será decisión/condición

**2.2.-Casos de prueba añadidos**

En este caso no fue necesario añadir ningún caso nuevo para cumplir la cobertura, ya que cada una de las sentencias condicionales es evaluada en su totalidad:

1.-fechaContratacion==null🡪True: **Caso de prueba 13**

2.-fechaContratacion==null🡪False: **Casos de prueba 1-12 y 14-16**

3.-fechaContratación.isAfter(LocalDate.now())🡪True: **Caso de prueba 12**

4.-fechaContratación.isAfter(LocalDate.now())🡪False: **Casos de prueba 1-11 y 13-16**

5.-Decisión 1🡪True: **Casos de prueba 12,13,14,15,16**

6.-Decisión 1🡪False: **Casos de prueba 1-11**

7.-Directivo🡪True: **Casos de prueba 1,4,8**

8.-Directivo🡪False: **Casos de prueba 1,3,5,6,7,9,10,11,12,13,14,15,16**

9.-Gestor🡪True: **Casos de prueba 2,5,7,9,10,11**

10.-Gestor🡪False: **Casos de prueba 1,3,4,6,8**

11.-Obrero🡪True: **Casos de prueba 3,6**

12.-Obrero🡪False: **Casos de prueba 1,2,4,5,7,8,9,10,11,12,13,14,15,16**

13.-fechaContratación.isBefore(LocalDate.now().minusYears(5).plusDays(1))🡪True:

**Casos de prueba 1,2,3**

14.-fechaContratación.isBefore(LocalDate.now().minusYears(5).plusDays(1))🡪False:

**Casos de prueba 4,5,6,7,8,9,10,11,13,14,15,16**

15.-fechaContratación.isAfter(LocalDate.now().minusYears(10))🡪True:

**Casos de prueba 1,2,3**

16.-fechaContratación.isAfter(LocalDate.now().minusYears(10))🡪False:

**Casos de prueba 4,5,6,7,8,9,10,11**

17.- fechaContratación.isBefore(LocalDate.now().minusYears(10).plusDays(1))🡪True:

**Casos de prueba 4,5,6**

18.- fechaContratación.isBefore(LocalDate.now().minusYears(10).plusDays(1))🡪False:

**Casos de prueba 1,2,3,7,8,9,10,11**

19.-fechaContratación.isAfter(LocalDate.now().minusYears(20))🡪True:

**Casos de prueba 4,5,6**

20.-fechaContratación.isAfter(LocalDate.now().minusYears(20))🡪False

**Casos de prueba 1,2,3,7,8,9,10,11**

21.-fechaContratación.isBefore(LocalDate.now().minusYears(20).plusDays(1))🡪True:

**Casos de prueba 10,11**

22.-fechaContratación.isBefore(LocalDate.now().minusYears(20).plusDays(1))🡪False:

**Casos de prueba 1,2,3,4,5,6,7,8,9**

23.-baja🡪True: **Casos de prueba 1,3,5,7,9,11,12,13,14,15**

24.-baja🡪False: **Casos de prueba 2,4,6,8,10**

## Proceso de pruebas de la clase EmpleadosGUI

## Proceso de pruebas de la clase ListaOrdenadaAcotada