ED05 - Pruebas de Caja Blanca y Caja Negra

Descripción

En esta tarea trabajaremos con una clase Java `GestorJuegos` que implementa un sistema básico de gestión de inventario para una tienda de videojuegos. La clase dispone de los métodos `main`, `registrarLoteJuegos` y `venderJuego`, además de métodos auxiliares como `mostrarStockJuegos` y `obtenerStockActual`, o métodos de validación, además del método `main` que se encarga de ejecutar el programa.

Código Base

Este es el código que deberás tener en cuenta para resolver la tarea:

```
import java.util.HashMap;
import java.util.Map;

/**
 * Clase principal que se encarga de gestionar los juegos
 */
public class GestorJuegos {
    // Utilizamos un mapa para almacenar los juegos y su cantidad
    private Map<String, Integer> stockJuegos;
    private final int maxStock = 200; // Stock máximo del total de juegos

    public GestorJuegos() {
        stockJuegos = new HashMap<>();
    }

    public static void main(String[] args) {
        GestorJuegos tienda = new GestorJuegos();

        // Registramos un lote de juegos
        if (tienda.registrarLoteJuegos(new String[]{"ABC123", "ABC124", "ABC125"},
new int[]{2, 3, 1}) < 0){
            System.out.println("Error al registrar los juegos");
        } else {
            System.out.println("Juegos registrados correctamente");
        }

        if (tienda.registrarLoteJuegos(new String[]{"XYZ123", "ABC125"}, new int[]
{2, 3, 1}) < 0){
            System.out.println("Error al registrar los juegos"); // Debería
imprimir este mensaje por no ser iguales las longitudes de los arrays
        } else {
            System.out.println("Juegos registrados correctamente");
        }

        if (tienda.registrarLoteJuegos(new String[]{"ABC123", "ABC126", "ABC112"},
```

```
new int[]{199, 3, 1}) < 0){
            System.out.println("Error al registrar los juegos"); // Debería
imprimir este mensaje por exceder el stock máximo
        } else {
            System.out.println("Juegos registrados correctamente");
        }

        if (tienda.registrarLoteJuegos(new String[]{"XYZ123", "ABC123", "ABC125"},
new int[]{100, 4, 3}) < 0){
            System.out.println("Error al registrar los juegos"); // Debería
imprimir este mensaje por exceder el stock máximo
        } else {
            System.out.println("Juegos registrados correctamente");
        }

        // Mostramos el stock de juegos
        tienda.mostrarStockJuegos();

        // Vendemos un juego
        int cantidadVendida = tienda.venderJuego("ABC123", 2); // Debería vender 2
juegos
        if (cantidadVendida > 0) {
            System.out.println("Se han vendido " + cantidadVendida + " juegos del
código ABC123");
        }

        tienda.mostrarStockJuegos();

        // Vendemos un juego que no existe
        cantidadVendida = tienda.venderJuego("XYZ264", 2); // Debería mostrar un
mensaje de error
        if (cantidadVendida > 0) {
            System.out.println("Se han vendido " + cantidadVendida + " juegos del
código XYZ123\n");
        } else {
            System.out.println("No se ha podido vender el juego con código
XYZ123\n");
        }

        tienda.mostrarStockJuegos();

        // Vendemos un juego con cantidad insuficiente
        cantidadVendida = tienda.venderJuego("ABC123", 20); // Debería mostrar un
mensaje de error
        if (cantidadVendida > 0) {
            System.out.println("Se han vendido " + cantidadVendida + " juegos del
código ABC123\n");
        } else {
            System.out.println("No se ha podido vender el juego con código
ABC123\n");
        }
```

```
        tienda.mostrarStockJuegos();

        //Vendemos un juego con código ilegal
        cantidadVendida = tienda.venderJuego("ABC1234", 2); // Debería mostrar un
mensaje de error
        if (cantidadVendida > 0) {
            System.out.println("Se han vendido " + cantidadVendida + " juegos del
código ABC123\n");
        } else {
            System.out.println("No se ha podido vender el juego con código
ABC123\n");
        }

        tienda.mostrarStockJuegos();

        //vendemos un juego con cantidad negativa
        cantidadVendida = tienda.venderJuego("ABC123", -2); // Debería mostrar un
mensaje de error
        if (cantidadVendida > 0) {
            System.out.println("Se han vendido " + cantidadVendida + " juegos del
código ABC123\n");
        } else {
            System.out.println("No se ha podido vender el juego con código
ABC123\n");
        }
    }

    private void mostrarStockJuegos() {
        System.out.println("Stock de juegos:");
        for (Map.Entry<String, Integer> juego : stockJuegos.entrySet()) {
            System.out.println(juego.getKey() + ": " + juego.getValue());
        }
    }

    // Metodo que registra un lote de juegos en el mapa de juegos de la tienda
    public int registrarLoteJuegos(String[] codigos, int[] unidadesPorCodigo) {
        int juegosRegistrados = 0;

        //comprobamos que las longitudes de los arrays sean iguales
        if (codigos.length != unidadesPorCodigo.length) {
            return -1;
        }

        //Comprobamos que los valores de cantidad sean positivos y sino, la
sumamos a cantidadTotal
        int cantidadTotal = 0;
        for (int cantidad : unidadesPorCodigo) {
            if (cantidad < 0) {
                return -2;
            } else {
                cantidadTotal += cantidad;
            }
```

```
        }

        //comprobamos que no se exceda el stock máximo
        if (obtenerStockActual() + cantidadTotal > maxStock) {
            return -3;
        }

        for (int i = 0; i < codigos.length; i++) {
            if (stockJuegos.containsKey(codigos[i])) {
                stockJuegos.put(codigos[i], stockJuegos.get(codigos[i]) +
unidadesPorCodigo[i]);
            } else {
                stockJuegos.put(codigos[i], unidadesPorCodigo[i]);
            }
            juegosRegistrados += unidadesPorCodigo[i];
        }
        return juegosRegistrados;
    }

    private int obtenerStockActual() {
        int stockActual = 0;
        for (int cantidad : stockJuegos.values()) {
            stockActual += cantidad;
        }
        return stockActual;
    }

    public int venderJuego(String codigo, int cantidad) {
        try {
            // Comprobamos el formato del código: ABC123
            validarCodigo(codigo);
            validarCantidad(cantidad);

            if (stockJuegos.containsKey(codigo)) {
                if (stockJuegos.get(codigo) >= cantidad) {
                    stockJuegos.put(codigo, stockJuegos.get(codigo) - cantidad);
                    return cantidad;
                } else {
                    System.out.println("No hay suficiente stock para el juego con
código " + codigo);
                    return -2;
                }
            } else {
                System.out.println("No existe el juego con código " + codigo);
                return -1;
            }
        } catch (IllegalArgumentException e) {
            System.out.println("Error: " + e.getMessage());
        }
        return 0;
    }
```

```
    private void validarCantidad(int cantidad) throws IllegalArgumentException {
        if (cantidad <= 0) {
            throw new IllegalArgumentException("La cantidad debe ser mayor que
0");
        }
    }

    private void validarCodigo(String codigo) throws IllegalArgumentException {
        if (codigo.length() != 6) {
            throw new IllegalArgumentException("El código debe tener 6
caracteres");
        }
        if (!codigo.substring(0, 3).matches("[A-Z]+")) {
            throw new IllegalArgumentException("Los primeros 3 caracteres deben
ser letras mayúsculas");
        }
        if (!codigo.substring(3).matches("[0-9]+")) {
            throw new IllegalArgumentException("Los últimos 3 caracteres deben ser
dígitos");
        }
    }
}
```

Puedes encontrar el código de esta práctica en el proyecto de Intellij IDEA subido a este enlace de GitHub.

Tareas a Realizar

Deberás crear un documento donde des respuesta a los siguientes apartados:

1. Realiza un análisis de caja blanca completo del método `registrarLoteJuegos`. En este análisis deberás:
    - Crear el grafo de flujo del método. Puedes hacerlo con drawio, mermaid o cualquier otra herramienta que prefieras (incluido Paint o, incluso, a mano).
    - Calcular la complejidad ciclomática, utilizando las 3 fórmulas disponibles
    - Identificar los caminos independientes
    - Diseñar los casos de prueba para cada camino

    > **Nota** 💡 Debes tener en cuenta los diferentes bucles y condiciones para identificar todos los caminos posibles

2. Realiza un análisis de caja negra, incluyendo valores límite y conjetura de errores del método `venderJuego`. Para este análisis considera que:
    - El método recibe como parámetro la cantidad de juegos a vender
    - La cantidad no podrá ser menor o igual a 0
    - La cantidad no podrá ser mayor al stock actual
    - El código del juego deberá tener 6 caracteres, los 3 primeros letras mayúsculas y los 3 últimos dígitos

    - Al tratarse de pruebas funcionales no es necesario conocer los detalles del código, aunque dispones de él en el código fuente.

3. Completa la clase `GestionJuegosTest` del tipo Caso de prueba JUnit en Intellij IDEA, que nos permita pasar las pruebas unitarias de caja blanca del método `registrarLoteJuegos`. Los casos de prueba ya los habrás obtenido en el primer apartado del ejercicio.

Entrega

Deberás crear un documento en el que incluyas:

- El análisis completo de caja blanca del método `registrarLoteJuegos`
- El análisis de caja negra del método `venderJuego`
- Capturas de pantalla mostrando la ejecución exitosa de las pruebas en Intellij IDEA

Ten en cuenta la sección UD5_5 relacionada con la documentación de pruebas para crear dicho documento.

Deberás entregar también el proyecto de Intellij IDEA con la clase de test creada, y crear un archivo comprimido junto al documento con el nombre **apellidos_nombre_ed05.zip**.