Actividad: Realizar una librería de validación de valores

Índice

Reglamentos generales para la actividad

1. La siguiente actividad consta un ejercicio práctico de código java. Se espera que la entrega contenga el programa, junto a un main() para poder probar que cumple con el objetivo. De lo contrario, no se considerará “entregada”.
2. La entrega consiste en un paquete exportado desde Eclipse, según se explica en el tutorial. De usar otro IDE, la entrega deberá estar empaquetada según los procedimientos de este.
3. No acoplar la salida de los métodos a la consola. No usar la salida por consola dentro de un método sin una justificación. De haberla, incluir el comentario en el código.
4. El código debe documentarse por sí mismo: no usar nombre de métodos o variables como “x”, “nom”, “val” etc. Usar, en cambio, “incógnita”, “nombre”, “valorDeRetorno”, etc.
5. No se considerará como “entregada” la actividad si se ignoran las convenciones de código descriptas en los módulos teóricos.
6. La presente actividad tendrá un valoración de “entregada” o “no entregada”
7. Se admitirá una sola entrega.
8. Se dará una devolución global a los alumnos haciendo hincapié en los puntos flojos comunes mediante la plataforma de la materia y remarcando los puntos correctos.

Ejercicio

Implementar una librería de validación de valores. Para cada valor inválido, deberá una excepción al flujo indicando el problema. Por ejemplo, evaluar un valor de texto, y si es “vacio”, lanzar una excepción. En código sería:

```
public String validarTexto(String texto) throws TextoVacioException {
   if(texto.length == 0) {
      throw new TextoVacioException(“El texto es invalido”);
   }
}
```

Tener en cuenta que esta librería será de utilidad para cuando se desarrolle el proyecto final, donde habrá muchas pantallas con muchos campos cada una y necesitarán de validaciones. Se pueden validar rangos de números, números negativos, que un texto sea solo letras o solo números o que necesariamente tenga ambos, etc., etc.

Realizar todas las excepciones que se consideren necesarias (ejemplo: TextoVacioExcepcion, ValorNuméricoException, etc.). Evaluar el uso de Checked Exceptions y de Unchecked Exceptions de ser necesario.

Se pueden usar métodos estáticos u otro enfoque. Justificar, mediante un comentario en el código, el porqué del enfoque.