Informe Practica Java 2

## Unai Sainz de la Maza Gamboa

* El único autor de la practica he sido yo.
* He realizado todos los apartados de la práctica. No he tenido ningún problema destacable.
* UML adjunto como imagen (no quedaba bien pegándolo aquí).
* **Errores corregidos:**

1. Documentación  
   - Las indicaciones sobre los parámetros se repite al menos en la clase Producto. **Creo que lo he solucionado, pero no sé muy bien a qué te refieres.**  
   - No se deben modificar las especificaciones indicadas en el enunciado. Forman parte de los requisitos del usuario: incluirNueviProducto, actualizarCantidadProducto, ... **Solucionado.**  
     
   (2) Implementacion 1.1  
   - Todas las constructoras de Producto deben ser pública. ¿Qué pasa si quitas el public a las constructoras con parámetros? **¿Hay algunas constructoras que no tienen por qué ser públicas, todo depende desde donde se las vaya a llamar, cuanto más escondidas de las demás clases estén los métodos, mejor no? Pero bueno, he puesto todas publicas.**  
   - crearProductoConDatos, ¿qué se gana con ese subprograma si ya hay una constructora que hace exactamente lo mismo? **Lo hice de cara a esta parte de la práctica, era lógico que luego íbamos a tener más tipos de productos e iba a hacer falta algo parecido.**  
   - cargarProductosDelFichero, para obtener el último código si te vas a quedar siempre con el último, tendrías que sacar la asignación fuera del bucle accediendo al código del último elemento del inventario. **No entiendo a qué te refieres.**  
   En lugar de generar tantas variables en cada ciclo, podrías crear directamente el producto con esos valores.  
   - actualizarCantidadProducto, como el caso del laboratorio, no es necesario que te pasen todo el producto (en este caso), basta su código. **Si estamos en programación orientada a objetos, no sería mejor trabajar todo con objetos, aunque estos solo tengan un atributo seteado, así lo veo yo al menos. He solucionado pasando solo el atributo correspondiente.**  
   - existeProductoDevuelvePos, bien el refinamiento de la operación auxiliar aunque te pasaran el código del producto. Es  
   más adecuado diferencias los objetos por sus códigos, dni, etc., valores que por defecto los distingue unívocamente. **He cambiado esto, y ahora le pasas el código.**   
   Si sólo vas a utilizar ese valor, para que poner en la interfaz más cosas??  
   - incluirNuevoProducto, aquí si te pasan el producto a añadir pero no tiene código, hay que añadirlo aquí ya que te lo mandan sin código según se ha establecido.   
   - getCodigoParaNuevoProducto, debería ser un subprograma auxiliar. **¿Y no lo es? Es una función que suma 1 al último código y devuelve ese código… igual no estoy entiendo tu observación.**  
   - mostrarProductos, no hace falta distinguir el caso listaProductos.length == 0. Cuando lo es la condición del bucle nunca es cierta y no se ejecuta ninguna vez. Elimina la sentencia if. **Solucionado.**  
   - getProducto, los mismos problemas que los señalados anteriormente: no se pasa el producto y se comparan por los códigos. **Sigo creyendo que es más adecuado trabajar solo con objetos, pero solucionado.**  
   - sacarMensajeError, bien este subprograma.  
     
   (3) Implementación 1.2  
   - actualizarCantidadProducto, con el uso de excepcione ya no tiene sentido la devolución de un valor boolean. Cuando devuelva siempre será true. **Solucionado.**