Paquete datos

Clase Lote:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?	X	
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones		
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	

Flujo de control	Si	No
Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?		X
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?	X	
¿Las salidas están siendo corroboradas que sean las correctas?		X
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?		X
Interfaz de módulos		
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?	X	
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?	X	
¿Ayudan los comentarios a entender el código?	X	
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?	X	
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)		X
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?		X
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	

¿Puede un cálculo ser realizado fuera de un ciclo?		X
¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGUEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el método **public Pedido getPedido()** se utiliza el atributo **pedido** sin verificar que no sea null.

En el método **public String detalles()** se realiza el catch de ArgumentoIllegalException pero no se realiza nada con ella.

Faltan comentarios y JavaDoc en:

- Cabecera de la clase
- Atributos de la clase
- Constructor de la clase
- Método **public String toString()**
- Método **public String detalles()**
- Método **public Pedido getPedido()**

Clase material:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		X
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?	X	
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones	Si	No
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No

Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?	X	
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?	X	
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos		
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?		X
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)		X
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)	X	
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?	X	
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X

¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGUEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el método **private boolean verificaCodigo(String código)** no verifica que el parámetro **codigo** sea distinto de null antes de acceder al método length() correspondiente a la clase String.

En el método **private boolean verificaDescripcion(String descripcion)** no verifica que el parámetro **descripcion** sea distinto de null antes de acceder al método length() correspondiente a la clase String.

Ni la clase ni ningún método ni declaraciones de atributos ni variables poseen sus correspondientes comentarios.

Comentarios:

- Solamente los constructores están abiertos al acceso para que el objeto pueda ser creado, los métodos sensibles no.
- Los atributos nunca pueden ser retornados como null mediante un método getter debido a que si estos son establecidos de manera incorrecta anteriormente, el programa finalizara mediante una excepción.
- Los métodos necesarios poseen el acoplamiento correcto con los demás métodos, aquellos que no lo necesitan no.

Clase pedido:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		X
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones		
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No

Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?	X	
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos	Si	No
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?	X	
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?	X	
¿Son todos los comentarios consistentes con el código?	X	
¿Ayudan los comentarios a entender el código?	X	
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?		X
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)		X
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)	X	
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?	X	
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X

¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGUEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

El atributo **estadoActual** no posee inicialización en null.

En el método **private boolean verificaNumeroPedido(String numeroPedido)** el parámetro numeroPedido no se verifica que sea distino de null ni que sea de la longitud correcta. La longitud de **numeroPedido** debería ser verificada en este método y no en el método **private boolean verifica(String str)**.

El atributo **estadoActual** no posee ningún valor antes del constructor.

Métodos que no poseen comentarios:

- evaluarPedido()
- detalles()

Faltan comentarios a las declaraciones de variables, atributos y/o constantes.

Comentarios:

- Solamente los constructores están abiertos al acceso para que el objeto pueda ser creado, los métodos sensibles no.
- Los atributos nunca pueden ser retornados como null mediante un método getter debido a que si estos son establecidos de manera incorrecta anteriormente, el programa finalizara mediante una excepción.
- Los métodos necesarios poseen el acoplamiento correcto con los demás métodos, aquellos que no lo necesitan no.

Clase Observación:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones		
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No
Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	

¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?	X	
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos		
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?	X	
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?	X	
¿Son todos los comentarios consistentes con el código?	X	
¿Ayudan los comentarios a entender el código?	X	
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?	X	
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)		X
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?		X
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X
¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X

¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGUEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

Falta aclarar que los parámetros de la clase se suponen distintos de null en todo momento luego de su creación, ya que son verificados e inicializados en el constructor.

En el método **private boolean verificaTexto(String texto)** se utiliza el parametro **texto** sin verificar si es o no null.

En el método **public int CompareTo(Object object)** se utiliza el parámetro **object** sin verificar si es o no null.

Faltan comentarios y JavaDoc en:

- Atributos de la clase
- Método **public boolean verificacion()**
- Método **String toString()**
- Método **compareTo(Object object)**

Clase tipoProducto:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		X
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones	Si	No
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No

Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?		X
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos		
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?	X	
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?		X
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)		X
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)	X	
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?	X	
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X

¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGUEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

El método **public void setListaMat(ListaMateriales listaMat)** no verifica que el parámetro **listaMat** sea distinto de null antes de asignarlo al atributo **this.listaMat.**

Los parámetros pasados en el constructor no son verificados que sean distintos de null.

Los parámetros no son verificados de ser distintos de null, por lo tanto mediante los getters se puede obtener un objeto null indeseadamente.

No hay comentarios ni en la clase, ni en ningún método, ni en ningún parámetro o constante.

En el método **public void generarTipoProd()** la variable **aux** no es setada en null luego de ser utilizada.

Comentarios:

- El constructor posee un modificador de acceso público para que sea posible crear el objeto.

Paquete datos.estadosPedido

Clase Estado: falta agregar comentarios

Clase Aceptado:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?	X	
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones		
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X

¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No
Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?		X
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos		
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?	X	
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)		X
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?		X
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X

¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X
¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGUEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el constructor de la clase no se verifica que el parámetro **pedido** sea distinto de null.

En el método **public void agregarObservacion(Observacion obs)** no se verifica que el parámetro **obs** sea distinto de null, aunque no es utilizado.

Faltan comentarios en:

- Cabecera de la clase
- Constructor de la clase
- Todos los métodos

Clase evaluación:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		X
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones	Si	No
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No

Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?		X
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos	Si	No
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?		X
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)		X
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?	X	
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X

¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGÜEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el constructor de la clase no se verifica que el parámetro **pedido** sea distinto de null.

Clase EstadoBase:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?	X	
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones		
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No

Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?		X
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos		
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?	X	
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)		X
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?		X
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X

¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGUEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el constructor de la clase no se verifica que el parámetro **pedido** sea distinto de null.

Faltan comentarios en:

- Cabecera de la clase
- Constructor de la clase
- Atributos de la clase
- Todos los métodos

Clase Iniciado:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?	X	
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones		
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No

Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?		X
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos		
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo, variable y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?	X	
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)		X
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?		X
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X

¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGUEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el constructor de la clase no se verifica que el parámetro **pedido** sea distinto de null.

En el método **public void agregarObservacion(Observacion obs)** no se verifica que el parámetro **obs** sea distinto de null, aunque no es utilizado.

En el método **public void evaluarPedido()** no se verifica que el atributo de clase **pedido** sea distinto de null antes de utilizarlo.

Faltan comentarios en:

- Cabecera de la clase
- Constructor de la clase
- Todos los métodos

Paquete exceptions

Clase ArgumentoIlegalException:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones		
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	

Flujo de control	Si	No
Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?	X	
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos		
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?	X	
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)		X
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?		X
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	

¿Puede un cálculo ser realizado fuera de un ciclo?		X
¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGUEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el constructor no se verifica que el parámetro **argumentoInvalido** sea distinto de null. Tampoco se verifica que el parámetro String **texto** sea distinto de null.

Faltan comentarios en:

- Cabecera de la clase
- Atributos de la clase
- Todos los métodos

Clase FaltantesException:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		X
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones	Si	No
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No

Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?	X	
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos	Si	No
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?		X
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)		X
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?	X	
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X

¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGÜEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

No se verifica que en el constructor los parámetros **mensaje** y **faltantes** sean distintos de null.

Clase StateException

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		X
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones	Si	No
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No

Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?	X	
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos	Si	No
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?		X
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)		X
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?	X	
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X

¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGÜEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

No se verifica que en el constructor el parámetro **mensaje** sea distinto de null.

Clase LengthException:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones		
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No
Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	

¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?	X	
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos		
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?	X	
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)		X
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?		X
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X
¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X

¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGUEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el constructor no se verifica que ambos parámetros sean distintos de null.

Faltan comentarios en:

- Cabecera de la clase
- Atributos de la clase
- Todos los métodos

Paquete listas

Clase ListaLotes:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		X
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?	X	
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones	Si	No
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	

Flujo de control	Si	No
Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	**Si**	**No**
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?		X
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos	**Si**	**No**
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?		X
¿Se retornan objetos nulos?		X
Comentarios	**Si**	**No**
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	**Si**	**No**
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?		X
Modularidad	**Si**	**No**
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)		X
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	**Si**	**No**
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?	X	
Performance	**Si**	**No**
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	

¿Puede un cálculo ser realizado fuera de un ciclo?		X
¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGÜEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el método **public void borrarLote(Lote lot)** no se verifica que lot sea distinto de null.

Faltan comentarios de la clase y del método **borrarLote(Lote lot).**

Clase ListaPedidos:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?	X	
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones		
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No
Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	

¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?		X
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos		
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?	X	
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?	X	
¿Ayudan los comentarios a entender el código?	X	
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?	X	
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)		X
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?		X
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X
¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X

¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGUEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el método public **void agregarNuevo(Pedido nuevo)** no se verifica que el parámetro sea distinto de null antes de utilizarlo.

Faltan comentarios en:

- Atributos de clase
- Constructor de clase
- Método **borrarPedido**
- Método **uptade**

Clase Empleado:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		X
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?	X	
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones	Si	No
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No

Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?		X
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos	Si	No
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?	X	
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?	X	
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?	X	
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?	X	
¿Son todos los comentarios consistentes con el código?	X	
¿Ayudan los comentarios a entender el código?	X	
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?		X
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)		X
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)	X	
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?	X	
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X

¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGÜEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

El método **private boolean verificaLegajo(String legajo)** no verifica que el parámetro **legajo** sea distinto de null.

El método **private boolean verificaNombreyApellido(String nya)** no verifica que el parámetro **nya** sea distinto de null.

Paquete visual

Ventana Producción

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		X
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones	Si	No
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	

Flujo de control	Si	No
Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	**Si**	**No**
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?		X
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos	**Si**	**No**
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?		X
¿Se retornan objetos nulos?		X
Comentarios	**Si**	**No**
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	**Si**	**No**
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?	X	
Modularidad	**Si**	**No**
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)		X
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)	X	
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	**Si**	**No**
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?	X	
Performance	**Si**	**No**
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	

¿Puede un cálculo ser realizado fuera de un ciclo?		X
¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGÜEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el constructor el parámetro **control** no se verifica que sea distinto de null.

Comentarios:

- El constructor es accesible pero no es un método sensible.
- El método **protected void IniciarComponentes()** supera las 60 líneas, pero es la única manera de ser codificado debido a que debe iniciar todos los componentes pertenecientes a la ventana.

Clase VentanaVentas:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		X
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones	Si	No
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No

Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?		X
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos	Si	No
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?		X
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?	X	
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)		X
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)	X	
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?	X	
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X

¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGÜEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el constructor el parámetro **control** no se verifica que sea distinto de null.

El método **protected void IniciarComponentes()** supera las 60 líneas, pero es la única manera de ser codificado debido a que debe iniciar todos los componentes pertenecientes a la ventana.

Faltan comentarios:

- Cabecera de la clase
- Atributos de la clase
- Métodos de la clase

Clase PanelFechas:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		X
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?		X
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?	X	
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?	X	
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones	Si	No
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No
Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	

¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?	X	
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos	Si	No
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?		X
¿Se retornan objetos nulos?		X
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?		X
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)		X
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)	X	
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?	X	
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X
¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X

¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGÜEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el método private void verificaDias(boolean bisiesto) no declara las variables dentro del for.

Clase Controlador:

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?	X	
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?		X
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?	X	
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?	X	
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones		
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?	X	
Flujo de control	Si	No
Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	

¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?		X
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos		
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?		
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?		
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?	X	
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)		X
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?		X
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X
¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X

¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGUEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X

En el método getPedidosEvaluacion hay casteos innecesarios.

Faltan comentarios en:

- Atributos de la clase
- Constructor
- Mayoría de los métodos

En ningún método se verifica que los parámetros sean distintos de null.

Tabla resultante de la aplicación

DESVIACIÓN		
Desviación	Si	No
¿Hay algún requerimiento que no se haya implementado?		X
DEFECTOS		
Variables y declaraciones de constantes	Si	No
¿Están nombradas las variables y las constantes con respecto a las convenciones de nomenclatura?	X	
¿Están todas las variables correctamente tipeadas?	X	
¿Están todas las variables correctamente inicializadas?	X	
¿Están todas las variables del ciclo for declaradas en la cabecera del ciclo?		X
¿Hay variables que debieran ser constantes?		X
¿Hay atributos que debieran ser variables locales?		X
¿Todos los atributos poseen correctos modificadores de acceso (privado, protegido, público)?	X	
¿Hay atributos estáticos que no debieran serlo, o viceversa?	X	
¿Hay variables o atributos con nombres similares confusos?		X
¿Puede cualquier parámetro ser convertido a variable local?		X
Definiciones de métodos	Si	No
¿Los métodos están nombrados de acuerdo a las convenciones de nomenclatura?	X	
¿Poseen todos los métodos correctos modificadores de acceso (privado, protegido, público)?	X	
¿Son todos los valores de los parámetros chequeados antes de ser usados?		X
¿Hay métodos estáticos que no debieran serlo, o viceversa?		X
¿Devuelven todos los métodos el valor correcto en cada retorno del mismo?	X	
Siempre que se utiliza toString: ¿Es sobrescrito?		X
Definiciones de clases	Si	No
¿Están nombradas las clases de acuerdo a las convenciones de nomenclatura?	X	
¿Posee cada clase un constructor apropiado?	X	
¿Poseen algunas sub-clases miembros en común que debieran estar en la super-clase?		X
¿Puede ser la jerarquía de herencia simplificada?		X
Referencia a datos	Si	No
Para cada referencia a un arreglo: ¿Se encuentran todos los valores de subíndices dentro de los límites definidos?	X	
Para cada referencia a objeto o arreglo: ¿Es el valor distinto de null?		X
Cálculos	Si	No
¿Existe algún cálculo con tipos de datos mezclados?		X
¿Existe posibilidad de overflow o underflow durante un cálculo?		X
Para cada operación con más de un operador: ¿Son las asunciones sobre el orden de evaluación y precedencia correctas?		X
¿Se usan paréntesis para evitar ambigüedad?	X	
Comparaciones y relaciones		
Por cada condición booleana: ¿Es correcta la condición chequeada?	X	
¿Son los operadores de las comparaciones correctos?	X	
¿Hay comparaciones entre variables de tipos inconsistentes?		X
¿Existen efectos secundarios inesperados resultantes de las comparaciones?		X
¿Han sido confundidos los operadores “&&” o “||” con los operadores binarios “&” y “|”?		X
¿Están siendo cubiertas todas las ramas (igual, menor, mayor)?		X
Flujo de control	Si	No

Por cada bucle: ¿Es la mejor opción de construcciones de bucles elegida?	X	
¿Poseen fin todos los bucles?	X	
¿Poseen todas las estructuras switch un default case?	X	
¿Son correctos todos los breaks dentro de una estructura switch?	X	
¿Resultan las anidaciones de bucles correctas y necesarias?	X	
¿Puede modificarse alguna anidación de if por una estructura switch?		X
¿Son todas las excepciones manejadas correctamente?	X	
¿Todos los métodos terminan?	X	
Entrada/Salida	Si	No
¿Hay errores gramaticales en el texto mostrado por pantalla?		X
¿Están siendo corroboradas las entradas con respecto a los valores que pueden tomar?		X
¿Las salidas están siendo corroboradas que sean las correctas?	X	
¿Son todas las excepciones de entrada/salida manejadas de la manera correcta?	X	
Interfaz de módulos		
¿Son correctos el número, tipo, orden y valor de los parámetros en cada llamada al método, con respecto a la declaración del mismo?	X	
¿Coinciden las unidades de los valores?	X	
Si un objeto o un arreglo es pasado por parámetro: ¿Este es modificado correctamente?	X	
¿Los constructores y los métodos sensibles de una clase están expuestos al acceso?	X	
¿Se retornan objetos nulos?	X	
Comentarios	Si	No
Cada clase y método, ¿poseen su correspondiente y apropiado comentario en la cabecera?		X
¿Posee cada declaración de atributo y constante un comentario?		X
¿Son expresados correctamente los comportamientos de cada método?		X
¿Es el comentario de la cabecera de cada clase o método correspondiente con el comportamiento de la misma?		X
¿Son todos los comentarios consistentes con el código?		X
¿Ayudan los comentarios a entender el código?		X
¿Hay suficiente cantidad de comentarios en el código?		X
¿Hay demasiados comentarios en el código?		X
Diseño	Si	No
¿Se utiliza una indentación estándar?	X	
Para cada método: ¿No supera las 60 líneas de largo?	X	
Modularidad	Si	No
¿Hay un bajo nivel de acoplamiento entre módulos? (Clases o métodos)	X	
¿Hay un alto nivel de cohesión entre módulos? (Clases o métodos)	X	
¿Existe código repetitivo que pueda ser reemplazado por la llamada a un método que posea el comportamiento del código repetitivo?		X
¿Están siendo utilizadas correctamente las clases de librerías de Java? (En caso de utilizar)	X	
Uso de almacenamiento	Si	No
¿Son los arreglos suficientemente largos?	X	
¿Son las referencias a objetos o arreglos establecidas en nulo una vez que estos no se necesitan más?		X
Performance	Si	No
¿Pueden ser usadas mejores estructuras de datos o algoritmos más eficientes?		X
¿Puede ser disminuido el coste de recalcular un valor calculándolo a este una sola vez y guardando su valor?		X
¿Son todos los valores calculados y almacenados verdaderamente utilizados?	X	
¿Puede un cálculo ser realizado fuera de un ciclo?		X

¿Puede desdoblarse un bucle corto de tal forma de evitar la utilización de un bucle?		X
¿Existen 2 bucles que operan sobre los mismos datos que pueden ser fusionados en uno?		X
INCONSISTENCIA		
¿Existe alguna parte del código implementada de manera inconsistente?		X
¿Las acciones de concatenación de strings se realizan correctamente?	X	
AMBIGUEDAD		
¿Existe algún método excesivamente complejo que deba ser reestructurado o dividido en varios métodos?		X
REDUNDANCIA		
¿Existe alguna variable o atributo redundante o sin uso?		X
¿Existe algún método sin llamar o innecesario?		X
¿Existe algún objeto que sea innecesario o redundante?		X
¿Hay código que pueda ser reemplazado por alguna llamada a un objeto externo reutilizable?		X