

Lista de chequeo para la revisión de código en lenguaje JAVA
Taller de Programación I - F. I. - UNMdP - Ingeniería Informática

| | |
|---------------|----------|
| ID. Proyecto: | Autores: |
| Revisores: | Fecha: |
| Notas: | |



N/A: No aplica

| I – Desviación de los Objetivos | | | | |
|---------------------------------|--|----|----|-----|
| # | I.1 Desviación | Si | No | N/A |
| 1 | El código implementa correctamente el diseño ? | | | |
| 2 | El código implementa más de lo que establece el diseño ? | | | |
| 3 | El mecanismo de envío (valor o referencia) de todos los parámetros de cada método es apropiado ? | | | |
| 4 | Cada método retorna el valor correcto en cada punto de retorno ? | | | |
| II – Omisión de Objetivos | | | | |
| # | II.1 Omisión | Si | No | N/A |
| 5 | El código no implementa completamente el diseño ? | | | |
| | Hay restos de código innecesario o test de prueba en el código ? | | | |
| III – Defectos en los Objetivos | | | | |
| # | III.1 Declaración de Variables y Constantes | Si | No | N/A |
| 7 | Los nombres de las variables y constantes son descriptivos y cumplen con las convenciones de nombres ? | | | |
| 8 | Los tipos de las variables son correctos ? | | | |
| 9 | Cada variables esta inicializada apropiadamente ? | | | |
| 10 | Todas las variables que controlan ciclos (ciclos for) están declaradas en la cabecera del ciclo ? | | | |
| 11 | Hay variables que deberían se constantes ? | | | |
| 12 | Ha atributos que deberían ser variables locales ? | | | |
| 13 | Todos los atributos tienen un indicador de acceso apropiado (private, protected, public)? | | | |
| 14 | Hay atributos estáticos (static) que no deberían serlo o viceversa ? | | | |
| # | III.2 Definición de Métodos | Si | No | N/A |
| 15 | Los nombres de los método son descriptivo y cumplen con las convenciones de nombres ? | | | |
| 16 | Todos los métodos tienen un indicador de acceso apropiado (private, protected, public) ? | | | |
| 17 | El valor de los parámetros de cada método es chequeado antes de usarlo ? | | | |
| 18 | Hay métodos estáticos (static) que no debieran serlo p viceversa ? | | | |
| # | III.3 Definición de Clases | Si | No | N/A |
| 19 | Cada clase tiene un constructor adecuado ? | | | |
| 20 | Existe algunas subclases con miembros comunes que deberían estar en una superclase ? | | | |
| 21 | Puede simplificarse la jerarquía de herencia de la clase ? | | | |
| # | III.4 Referencia a los Datos | Si | No | N/A |
| 22 | Para referencia a un arreglo los valores de los subindices está dentro del rango permitido ? | | | |
| 23 | Se verifica que toda referencia a un objeto o arreglo no sea nula ? | | | |
| # | III.5 Expresiones y Tipos de Datos | Si | No | N/A |

Lista de chequeo para la revisión de código en lenguaje JAVA
Taller de Programación I - F. I. - UNMdP - Ingeniería Informática

| | |
|---------------|----------|
| ID. Proyecto: | Autores: |
| Revisores: | Fecha: |
| Notas: | |

N/A: No aplica

| | | | | |
|----|---|----|----|---|
| 24 | Hay algún cálculo con tipos de datos mezclados ? | | | |
| 25 | Es posible el overflow or el underflow, durante un cálculo ? | | | |
| 26 | Por cada expresión se respet el orden de evaluación y precedencia correcta ? | | | |
| 27 | Se usan paréntesis para evitar ambigüedades ? | | | |
| 28 | El código previene los errores por redondeo en forma sistemática | | | |
| 29 | El código evita sumas y restas sobre números con magnitudes muy diferentes ? | | | |
| 30 | Se chequea la división por cero o el ruido ? | | | |
| # | III.6 Comparacion y Relaciones | Si | No | N/A |
| 31 | Las expresiones booleanas han sido simplificadas, usando "driving negations inward" ? | | | |
| 32 | Cada prueba booleana chequea la condición correcta ? | | | |
| 33 | Hay comparaciones entre variables de tipos inconsistentes ? | | | |
| 34 | Son correctos los operadores de comparación ? | | | |
| 35 | Todas laas expresiones booleanas son correctas ? | | | |
| 36 | Existen efectos colaterales inapropiados de una comparación ? | | | |
| 37 | Se intercambiado un "&" por un "&&" ó un " " por un " " ? | | | |
| 38 | El código evita la comparación de igualdad en números de punto flotante ? | | | |
| 39 | Estan cubiertas las tres ramas de los if (menor,igual,mayor) | | | |
| # | III.7 Control de Flujo | Si | No | N/A |
| 40 | Por cada ciclo se usa la mejor elección de construcción de ciclos ? | | | |
| 41 | Todos los ciclos terminan ? | | | |
| 42 | Cuando un ciclo tiene multiples condiciones de salida todas estan manejadas apropiadamente ? | | | |
| 43 | Todas las sentencias SWITCH tienen un caso por defecto ? | | |  |
| 44 | Las salidas de un Switch no manejadas esta debidamente comentadas y con una sentencia break ? | | |  |
| 45 | Es correcta la profundidad en el anidamiento de ciclos ? | | | |
| 46 | Se pueden convertir algún if anidado en sentencias SWITCH ? | | | |
| 47 | Los cuerpos nulos en las estructuras de control estan marcados con llaves, marcados y comentados correctamente? | | | |
| 48 | Todos los métodos terminan ? | | | |
| 49 | Todas las excepciones son manipuladas apropiadamente ? | | | |
| 50 | Las sentencias break con con etiqueta derivan el control al lugar correcto ? | | | |
| # | III.8 Entrada/Salida | Si | No | N/A |
| 51 | Todos los archivos se abren antes de usarlos ? | | | |
| 52 | Los atributos de las sentencias de apertura de los archivos son consistente con el uso de los mismos ? | | | |
| 53 | Todos los archivos se cierran cuando dejan de usarse ? | | | |
| 54 | Los datos en el buffer so envían al disco ? | | | |

Lista de chequeo para la revisión de código en lenguaje JAVA
Taller de Programación I - F. I. - UNMDP - Ingeniería Informática

| | |
|---------------|----------|
| ID. Proyecto: | Autores: |
| Revisores: | Fecha: |
| Notas: | |

N/A: No aplica

| | | | | |
|----|---|----|----|-----|
| 55 | Hay errores de ortografía o gramática en el texto impreso o en la pantalla ? | | | |
| 56 | Están chequeadas las condiciones de error ? | | | |
| 57 | Se verifica la existencia de los archivos antes de intentar abrirlos ? | | | |
| 58 | Todas las excepciones de entrada/salida están razonablemente manejadas ? | | | |
| # | III.9 Interface del Módulo | Si | No | N/A |
| 59 | El número, orden, tipo y valores de parámetros en cada llamada de un método esta de acuerdo con la declaración del método ? | | | |
| 60 | Los valores respetan los acuerdos de unidades (por.ej., pulgadas versus yardas) ? | | | |
| 61 | Si un objeto o arreglo es pasado a un método que lo altera, esta alteración es realizada correctamente por dicho método ? | | | |
| # | III.10 Comentarios | Si | No | N/A |
| 62 | Todos los métodos, clases y archivos tienen los comentarios de cabecera apropiados ? | | | |
| 63 | Cada atributo, variable ó declaración de constante ha sido comentada ? | | | |
| 64 | El comportamiento de cada método y clase es expresado en lenguaje plano ? | | | |
| 65 | Los comentarios en la cabecera de cada método y clase son consistentes con el comportamiento del método o clase ? | | | |
| 66 | Todos los comentarios son consistentes con el código ? | | | |
| 67 | Los comentarios ayudan a entender el código ? | | | |
| 68 | Hay suficientes comentarios en el código ? | | | |
| 69 | Hay demasiados comentarios en el código ? | | | |
| # | III.11 Diseño y Empaquetado | Si | No | N/A |
| 70 | El formato standard en el diseño e indentación del código es usado consistentemente ? | | | |
| 71 | Algún método excede las 60 líneas ? | | | |
| 72 | Algún módulo excede las 600 líneas ? | | | |
| # | III.12 Modularidad | Si | No | N/A |
| 73 | Hay un bajo nivel de acoplamiento entre módulos (métodos y clases) ? | | | |
| 74 | Hay un alto nivel de cohesión encada módulo (métodos y clases) ? | | | |
| 75 | Hay código repetido que se puede reemplazar por un método que implemente el comportamiento de dicho código ? | | | |
| 76 | Se usan las librerías de clase java cuando y donde deben usarse ? | | | |
| # | III.13 Almacenamiento | Si | No | N/A |
| 77 | Los arreglos tienen previsto el tamaño suficiente ? | | | |
| 78 | Las referencias a los objetos y arreglos son seteados a nulo una vez que dejan de usarse? | | | |
| # | III.14 Perfomance | Si | No | N/A |
| 79 | Pueden mejorarse las estructuras de datos o usar algoritmos más eficientes ? | | | |
| 80 | Los test lógicos están organizados, de manera que los más frecuentes y caros estén primero ? | | | |
| 81 | Puede reducirse el costo de recálculo mediante el almacenamiento de los resultados ? | | | |

Lista de chequeo para la revisión de código en lenguaje JAVA
Taller de Programación I - F. I. - UNMDP - Ingeniería Informática

| | |
|---------------|----------|
| ID. Proyecto: | Autores: |
| Revisores: | Fecha: |
| Notas: | |

N/A: No aplica

| | | | | |
|---|---|----|----|-----|
| 82 | Actualmente, se usa cada resultado calculado y almacenado ? | | | |
| 83 | Puede un cálculo sacarse fuera de un ciclo ? | | | |
| 84 | Hay test dentro de un ciclo que no necesitan ser realizados ? | | | |
| 85 | Puede un ciclo corto ser convertido en una estructura más simple ? | | | |
| 86 | Dos ciclos sobre los mismos datos se pueden combinar en uno? | | | |
| IV – Inconsistencia en los Objetivos | | | | |
| # | IV.1 Performance | Si | No | N/A |
| 87 | Hay algún código implementado en modo inconsistente ? | | | |
| V – Ambigüedad en los Objetivos | | | | |
| # | V.1 Declaración de Variables y Constantes | Si | No | N/A |
| 88 | Hay variables con nombres similares y confusos ? | | | |
| 89 | Todas las variables están definidas con nombres claros, consistentes y significativos ? | | | |
| # | V.2 Performance | Si | No | N/A |
| 90 | Hay módulos excesivamente confusos que se pueden reestructurar o dividir en varias rutinas ? | | | |
| VI – Redundancia en los Objetivos | | | | |
| # | VI.1 Variables | Si | No | N/A |
| 91 | Existen variables o atributos redundante o no usados ? | | | |
| 92 | Podría alguna variable no local convertirse en local ? | | | |
| # | VI.2 Definición de Métodos | Si | No | N/A |
| 93 | Hay algunos metodos que no son llamados o son innecesarios ? | | | |
| # | VI.3 Performance | Si | No | N/A |
| 94 | Puede algún código reemplazarse con llamadas a objetos externos reusables ? | | | |
| 95 | Existen bloques de código repetidos que pueden condensarse en un método simple ? | | | |
| 96 | Existen restos de código no usado o restos de rutinas de test ? | | | |
| VII – Efectos Colaterales en los Objetivos | | | | |
| # | VII.1 Definición de Métodos | Si | No | N/A |
| 97 | Después de cambiar un método se analizan los metodos que lo llaman | | | |
| # | VII.2 Base de Datos | Si | No | N/A |
| 98 | El proceso de actualización y migración sigue el cambio de estructuras o contenidos en la base del proyecto ? | | | |