|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I – Desviación de los Objetivos | | | | |
| # | I.1 Desviación | Si | No | N/A |
| 1 | El código implementa correctamente el diseño ? |  | X |  |
| 2 | El código implementa más de lo que establece el diseño ? | X |  |  |
| 3 | El mecanismo de envío (valor o referencia) de todos los parámetros de cada método es apropiado ? | X |  |  |
| 4 | Cada método retorna el valor correcto en cada punto de retorno ? | X |  |  |
| II – Omisión de Objetivos | | | | |
| # | II.1 Omisión | Si | No | N/A |
| 5 | El código implementa completamente el diseño ? |  | X |  |
|  | No hay restos de código innecesario o test de prueba en el código ? |  | X |  |
| III – Defectos en los Objetivos | | | | |
| # | III.1 Declaración de Variables y Constantes | Si | No | N/A |
| 7 | Los nombres de las variablees y constantes son descriptivos y cumplen con las convenciones de nombres ? | X |  |  |
| 8 | Los tipos de las variables son correctos ? | X |  |  |
| 9 | Cada variables esta inicializada apropiadamente ? | X |  |  |
| 10 | Todas las variables que controlan ciclos (ciclos for) están declaradas en la cabecera del ciclo ? |  | X |  |
| 11 | Hay variables que deberían se constantes ? | X |  |  |
| 12 | No hay atributos que deberían ser variables locales ? |  | X |  |
| 13 | Todos los atributos tienen un indicador de acceso apropiado (private, protected, public)? | X |  |  |
| 14 | No hay atributos estáticos (static) que no deberían serlo o viceversa ? | X |  |  |
| # | III.2 Definición de Métodos | Si | No | N/A |
| 15 | Los nombres de los método son descriptivos y cumplen con las convenciones de nombres ? | X |  |  |
| 16 | Todos los métodos tienen un indicador de acceso apropiado (private, protected, public) ? | X |  |  |
| 17 | El valor de los parámetros de cada método es chequeado antes de usarlo ? |  | X |  |
| 18 | No hay métodos estáticos (static) que no debieran serlo ó viceversa ? | X |  |  |
| # | III.3 Definición de Clases | Si | No | N/A |
| 19 | Cada clase tiene un constructor adecuado ? | X |  |  |
| 20 | No existe subclases con miembros comunes que deberían estar en una superclase ? | X |  |  |
| 21 | No puede simplificarse la jerarquía de herencia de la clase ? | X |  |  |
| # | 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 ? | X |  |  |
| 23 | Se verifica que toda referencia a un objeto o arreglo no sea nula ? |  | X |  |
| # | III.5 Expresiones y Tipos de Datos | Si | No | N/A |
| 24 | No hay cálculos con tipos de datos mezclados ? |  |  | X |
| 25 | No es posible el overflow or el underflow, durante un cálculo ? |  |  | X |
| 26 | Por cada expresión se respeta el orden de evaluación y precedencia correcta ? |  | NOSE |  |
| 27 | Se usan paréntesis para evitar ambigüedades ? |  |  | X |
| 28 | El código previene los errores por redondeo en forma sistemática ? |  |  | X |
| 29 | El código evita sumas y restas sobre números con magnitudes muy diferentes ? |  |  | X |
| 30 | Se chequea la división por cero o el ruido ? |  |  | X |
| # | III.6 Comparacion y Relaciones | Si | No | N/A |
| 31 | Las expresiones booleanas han sido simplificadas, usando “driving negations inward” ? |  |  | X |
| 32 | Cada prueba booleana chequea la condición correcta ? | X |  |  |
| 33 | No hay comparaciones entre variables de tipos inconsistentes ? | X |  |  |
| 34 | Son correctos los operadores de comparación ? | X |  |  |
| 35 | Todas las expresiones booleanas son correctas ? | X |  |  |
| 36 | No existen efectos colaterales inapropiados de una comparación ? | X |  |  |
| 37 | No se intercambia "&" por un "&&" ó un "|" por un "||" ? | X |  |  |
| 38 | El código evita la comparación de igualdad en números de punto flotante ? |  | X |  |
| 39 | Estan cubiertas las tres ramas de los if (menor,igual,mayor) ? | X |  |  |
| # | III.7 Control de Flujo | Si | No | N/A |
| 40 | Por cada ciclo se usa la mejor elección de construcción de ciclos ? | X |  |  |
| 41 | Todos los ciclos terminan ? | X |  |  |
| 42 | Cuando un ciclo tiene multiples condiciones de salida todas estan manejadas apropiadamente ? | X |  |  |
| 43 | Todas las sentencias SWITCH tienen un caso por defecto ? | X |  |  |
| 44 | Las salidas de un Switch no manejadas esta debidamente  comentadas y con una sentencia break ? | X |  |  |
| 45 | Es correcta la profundidad en el anidamiento de ciclos ? | X |  |  |
| 46 | Ningún if anidado se puede convertir en sentencias SWITCH ? |  | X |  |
| 47 | Los cuerpos nulos en las estructuras de control estan marcados con llaves, marcados y comentados correctamente? |  | NOSE |  |
| 48 | Todos los métodos terminan ? | X |  |  |
| 49 | Todas las excepciones son manipuladas apropiadamente ? | X |  |  |
| 50 | Las sentencias break con con etiqueta derivan el control al lugar correcto ? |  |  | X |
| # | III.8 Entrada/Salida | Si | No | N/A |
| 51 | Todos los archivos se abren antes de usarlos ? | X |  |  |
| 52 | Los atributos de las sentencias de apertura de los archivos son consistentes con el uso de los mismos ? | X |  |  |
| 53 | Todos los archivos se cierran cuando dejan de usarse ? |  | X |  |
| 54 | Los datos en el buffer se envían al disco ? | X |  |  |
| 55 | No hay errores de ortografía o gramática en el texto impreso o en la pantalla ? | X |  |  |
| 56 | Están chequeadas las condiciones de error ? | X |  |  |
| 57 | Se verifica la existencia de los archivos antes de intentar abrirlos ? | X |  |  |
| 58 | Todas las excepciones de entrada/salida están razonablemente manejadas ? | X |  |  |
| # | 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 ? | X |  |  |
| 60 | Los valores respetan los acuerdos de unidades (por.ej., pulgadas versus yardas) ? | X |  |  |
| 61 | Si un objeto o arreglo es pasado a un método que lo altera, esta alteración es realizada correctamente por dicho método ? | X |  |  |
| # | III.10 Comentarios | Si | No | N/A |
| 62 | Todos los métodos, clases y archivos tienen los comentarios de cabecera apropiados ? |  | X |  |
| 63 | Cada atributo, variable ó declaración de constante ha sido comentada ? |  | X |  |
| 64 | El comportamiento de cada método y clase es expresado en legüaje plano ? |  | NOSE |  |
| 65 | Los comentarios en la cabecera de cada método y clase son consistentes con el comportamiento del método o clase ? |  |  | X |
| 66 | Todos los comentarios son consistentes con el código ? |  |  | X |
| 67 | Los comentarios ayudan a entender el código ? |  |  | X |
| 68 | Hay suficientes comentarios en el código ? |  | X |  |
| 69 | No hay demasiados comentarios en el código ? | X |  |  |
| # | III.11 Diseño y Empaquetado | Si | No | N/A |
| 70 | El formato standard en el diseño e indentación del código es usado consistentemente ? |  | NOSE |  |
| 71 | Ningún método excede las 60 lineas ? |  | X |  |
| 72 | Ningún módulo excede las 600 lineas ? | X |  |  |
| # | III.12 Modularidad | Si | No | N/A |
| 73 | Hay un bajo nivel de acoplamiento entre módulos (métodos y clases) ? | X |  |  |
| 74 | Hay un alto nivel de cohesión encada módulo (métodos y clases) ? | X |  |  |
| 75 | No hay código repetido que se puede reemplazar por un método que implemente el comportamiento de dicho código ? |  | X |  |
| 76 | Se usan las librerías de clase java cuando y donde deben usarse ? | X |  |  |
| # | III.13 Almacenamiento | Si | No | N/A |
| 77 | Los arreglos tienen previsto el tamaño suficiente ? | X |  |  |
| 78 | Las referencias a los objetos y arreglos son seteados a nulo una vez que dejan de usarse? |  | X |  |
| # | III.14 Perfomance | Si | No | N/A |
| 79 | Las estructuras de datos y algoritmos no tienen mejoras aparentes en eficiencia ? |  |  | X |
| 80 | Los test lógicos están organizados, de manera que los más frecuentes y caros estén primero ? |  | NOSE |  |
| 81 | Puede reducirse el costo de recálculo mediante el almacenamiento de los resultados ? | X |  |  |
| 82 | Actualmente, se usa cada resultado calculado y almacenado ? | X |  |  |
| 83 | Ningún cálculo puede sacarse fuera de un ciclo ? | X |  |  |
| 84 | Ninguún test dentro de un ciclo es innecesario ? |  | NOSE |  |
| 85 | Ningún ciclo corto se puede convertir en una estructura más simple ? | X |  |  |
| 86 | No hat dos ciclos sobre los mismos datos que se puedan combinar en uno? |  | X |  |
| IV – Inconsistencia en los Objetivos | | | | |
| # | IV.1 Perfomance | Si | No | N/A |
| 87 | Hay algún código implementado en modo inconsistente ? |  | X |  |
| V – Ambigüedad en los Objetivos | | | | |
| # | V.1 Declaración de Variables y Constantes | Si | No | N/A |
| 88 | No hay variables con nombres similares y confusos ? |  | X |  |
| 89 | Todas las variables están definidas con nombres claros, consistentes y significativos ? | X |  |  |
| # | V.2 Perfomance | Si | No | N/A |
| 90 | No existen módulos excesivamente confusos que se pueden reestructurar o dividir en varias rutinas ? | X |  |  |
| VI – Redundancia en los Objetivos | | | | |
| # | VI.1 Variables | Si | No | N/A |
| 91 | No existen variables o atributos redundante o no usados ? |  | X |  |
| 92 | Nínguna variable no local puede convertirse en local ? |  | X |  |
| # | VI.2 Definición de Métodos | Si | No | N/A |
| 93 | No hay metodos no invocados o innecesarios ? | X |  |  |
| # | VI.3 Perfomance | Si | No | N/A |
| 94 | No existe código que se pueda reemplazar con llamadas a objetos externos reusables ? | X |  |  |
| 95 | No existen bloques de código repetidos que pueden condensarse en un método simple ? |  | X |  |
| 96 | No existen restos de código no usado o restos de rutinas de test ? |  | X |  |
| 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 |  |  | X |
| # | 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 ? |  | X |  |