|  |  |
| --- | --- |
| ID. Proyecto: | Autores: Mouse |
| Revisores: Carlos Andrés Efstratiadis | Fecha: 27/08/2025 |
| Notas: | |

N/A: No aplica

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 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 |
| 6 | El código no tiene partes innecesarias y los test de prueba se han eliminado totalmente ? |  | X |  |
| 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 ? |  | 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 | Todas las variables son necesarias (no se pueden reemplazar por una constante) ? |  | X |  |
| 12 | Todos los atributos son necesarios (no se pueden reemplazar por variables locales) ? |  | X |  |
| 13 | Todos los atributos tienen un indicador de acceso apropiado (private, protected, public) ? | X |  |  |
| 14 | Los atributos estáticos (static) son necesarios (no se los puede reemplazar por uno no estático) y lo mismo ocurre en el caso inverso ? |  | 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 ? |  |  |  |
| 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 | Los métodos estáticos (static) del código son necesarios (no se pueden reemplazar por uno no estático) y lo mismo ocurre en el caso inverso ? |  |  |  |
| # | III.3 Definición de Clases | Si | No | N/A |
| 19 | Cada clase tiene un constructor adecuado ? |  |  |  |
| 20 | Las subclases con miembros comunes son necesarias (no pueden estar en una superclase) ? |  |  |  |
| 21 | La jerarquía de herencia de la clase es la necesaria (no puede simplificarse) ? |  |  |  |
| # | III.4 Referencia a los Datos | Si | No | N/A |
| 22 | Los valores de los subíndices de los arreglos estan 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 |
| 24 | Se evita la mezcla en tipos de datos en los cálculos ? |  |  |  |
| 25 | Se evita el posible overflow o underflow, durante un cálculo ? |  |  |  |
| 26 | Por cada expresión se respeta 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 ? |  |  |  |

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

N/A: No aplica

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 33 | No hay comparaciones entre variables de tipos inconsistentes ? |  |  |  |
| 34 | Son correctos los operadores de comparación ? |  |  |  |
| 35 | Todas las expresiones booleanas son correctas ? |  |  |  |
| 36 | Se evitan los efectos colaterales inapropiados de una comparación ? |  |  |  |
| 37 | Se han reemplazado todos los "&" por "&&" ó todos los "|" por "||" ? |  |  |  |
| 38 | El código evita la comparación de igualdad en números de punto flotante ? |  |  |  |
| 39 | Están 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 múltiples condiciones de salida, todas están 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 | Todos los if anidados son necesarios (no se pueden reemplazar por una sentencia SWITCH) ? |  |  |  |
| 47 | Los cuerpos nulos en las estructuras de control están marcados con llaves, marcados y comentados correctamente? |  |  |  |
| 48 | Todos los métodos terminan ? |  |  |  |
| 49 | Todas las excepciones son manipuladas apropiadamente ? |  |  |  |
| 50 | Las sentencias break 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 consistentes con el uso de los mismos ? |  |  |  |
| 53 | Todos los archivos se cierran cuando dejan de usarse ? |  |  |  |
| 54 | Los datos en el buffer se envían al disco ? |  |  |  |
| 55 | No 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 ? |  |  |  |
| # | III.11 Diseño y Empaquetado | Si | No | N/A |
| 69 | El formato standard en el diseño e indentación del código es usado consistentemente ? |  |  |  |

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

N/A: No aplica

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 70 | Algún método excede las 60 lineas ? |  |  |  |
| 71 | Algún módulo excede las 600 lineas ? |  |  |  |
| # | III.12 Modularidad | Si | No | N/A |
| 72 | Hay un bajo nivel de acoplamiento entre módulos (métodos y clases) ? |  |  |  |
| 73 | Hay un alto nivel de cohesión en cada módulo (métodos y clases) ? |  |  |  |
| 74 | Todo el código es diferente (no hay código repetido que pueda reemplazarse por un método que implemente el comportamiento de dicho código) ? |  |  |  |
| 75 | Se usan las librerías de clase java cuando y donde deben usarse ? |  |  |  |
| # | III.13 Almacenamiento | Si | No | N/A |
| 76 | Los arreglos tienen previsto el tamaño suficiente ? |  |  |  |
| 77 | Las referencias a los objetos y arreglos son seteados a nulo una vez que dejan de usarse? |  |  |  |
| # | III.14 Perfomance | Si | No | N/A |
| 78 | Las estructuras de datos no se pueden mejorar ni usar algoritmos más eficientes ? |  |  |  |
| 79 | Los test lógicos están organizados, de manera que los más frecuentes y caros estén primero ? |  |  |  |
| 80 | Almacenar los resultados permite reducir el costo de recálculo ? |  |  |  |
| 81 | Actualmente, se usa cada resultado calculado y almacenado ? |  |  |  |
| 82 | Todos los cálculos dentro del ciclo son necesarios (no puede sacarse fuera de este)? |  |  |  |
| 83 | Todos los test dentro de un ciclo necesitan ser realizados ? |  |  |  |
| 84 | Si existe un ciclo corto esta es la única opción (el mismo no se puede convertir en una estructura más simple) ? |  |  |  |
| 85 | SI existen dos ciclos que operan sobre los mismos datos, esta es la única opción (no se pueden combinar) ? |  |  |  |
| IV – Inconsistencia en los Objetivos | | | | |
| # | IV.1 Perfomance | Si | No | N/A |
| 86 | Todo el código implementado es consistente ? |  |  |  |
| V – Ambigüedad en los Objetivos | | | | |
| # | V.1 Declaración de Variables y Constantes | Si | No | N/A |
| 87 | Los nombres de las variables son todos bien distintos y no se prestan a confusión ? |  |  |  |
| 88 | Todas las variables están definidas con nombres claros, consistentes y significativos ? |  |  |  |
| # | V.2 Perfomance | Si | No | N/A |
| 89 | Las estructuras de los módulos son las mínimas y no admiten su división en varias rutinas ? |  |  |  |
| VI – Redundancia en los Objetivos | | | | |
| # | VI.1 Variables | Si | No | N/A |
| 90 | Todas las variables y atributos se usan y no son redundantes ? |  |  |  |
| 91 | Ninguna variables externas a una clase o método, puede convertirse en local ? |  |  |  |
| # | VI.2 Definición de Métodos | Si | No | N/A |
| 92 | Todos los métodos se llaman y son útiles ? |  |  |  |
| # | VI.3 Perfomance | Si | No | N/A |
| 93 | Todo el código es necesario y no puede reemplazarse con llamadas a objetos externos reusables ? |  |  |  |
| 94 | Todos los bloques de código son diferentes y no pueden condensarse en un método simple ? |  |  |  |
| 95 | Todo el código se usa y las rutinas de test fueron retiradas totalmente ? |  |  |  |
| VII – Efectos Colaterales en los Objetivos | | | | |
| # | VII.1 Definición de Métodos | Si | No | N/A |
| 96 | Después de cambiar un método se analizan los métodos que lo llaman ? |  |  |  |
| # | VII.2 Base de Datos | Si | No | N/A |

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

N/A: No aplica

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 97 | El proceso de actualización y migración sigue el cambio de estructuras o contenidos en la base del proyecto ? |  |  |  |