**Checklist de Revisión del Diseño**

|  |  |  |  |
| --- | --- | --- | --- |
| Estudiante | David Alejandro Martínez Tristán | Fecha | 19/03/2021 |
| Programa | Calidad y pruebas de software | # Programa | 3 |
| Instructor | Adriana González Ugalde | Lenguaje | C++ |

|  |  |
| --- | --- |
| Propósito | Guiarte en la conducción de una revisión efectiva de tu diseño |
| General | * Escribe en el primer renglón el nombre de las partes que vas a revisar (aquellas que son nuevas o que van a ser modificadas) * Revisa tu diseño parte por parte. No continúes con la siguiente parte hasta que no hayas revisado por completo la parte anterior. * Cuando estés revisando una parte, revisa una categoría a la vez * Cuando termines de revisar una categoría para una parte, márcala como revisada |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nombre de las partes: | | Main | FileReader | Printer | Errors |
| Completo | Verificar que el diseño cumple con todos los requisitos:   * Se producen todas las salidas requeridas * Se reciben todas las entradas especificadas | X | X | X | N/A |
| Estructuras | * Examinar la lógica del programa para identificar el tipo de estructuras necesarias de acuerdo con la manera en la que se manipulará y accederá a los datos | X | X | X | X |
| Casos especiales | Verificar todos los casos especiales   * Variables con valores vacíos, completos, mínimos, máximos, negativos y cero. * Lectura y asignación de datos del tipo incorrecto. * Segmentation fault.   Asegurar que resulte imposible para el usuario entrar a condiciones de error  Manejar adecuadamente todas las condiciones incorrectas o de error | X | X | X | X |
| Uso de funciones | * Todas las funciones, procedimientos y métodos son definidos, completamente entendidos y propiamente utilizados. | X | X | X | X |
| Integridad | Verificar que el programa mantenga la integridad de los archivos de entrada | X | X | X | N/A |
| Nombres | * Todos los nombres son claros, descriptivos y distintos entre sí | X | X | X | X |
| Diagramas | Verificar que el diagrama de clases sigue la notación de su estándar   * Atributos y métodos * Relaciones y cardinalidad | X | X | X | X |

**Checklist de Revisión del Código**

|  |  |  |  |
| --- | --- | --- | --- |
| Estudiante | David Alejandro Martínez Tristán | Fecha | 19/03/2021 |
| Programa | Calidad y pruebas de software | # Programa | 3 |
| Instructor | Adriana González Ugalde | Lenguaje | C++ |

|  |  |
| --- | --- |
| Propósito | Guiarte en la conducción de una revisión efectiva de tu código |
| General | * Escribe en el primer renglón el nombre de las partes que vas a revisar (aquellas que son nuevas o que van a ser modificadas) * Revisa tu código parte por parte. No continúes con la siguiente parte hasta que no hayas revisado por completo la parte anterior. * Cuando estés revisando una parte, revisa una categoría a la vez * Cuando termines de revisar una categoría para una parte, márcala como revisada |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nombre de las partes: | | Main | FileReader | Printer | Errors |
| Inicialización | Verificar que el método de inicialización de una clase incluya a todas las variables que requieran reiniciar sus valores al inicio de un nuevo procedimiento.  Verificar la inicialización de variables   * Al inicio del programa * Al inicio de un nuevo procedimiento | X | X | X | N/A |
| Llamadas | Verificar los formatos de llamada en cada función   * Punteros * Parámetros * Uso de “&” | X | X | X | N/A |
| Punteros | Verificar que   * Cada puntero sea inicializado con NULL * Las secciones que utilicen punteros se ejecuten si su valor es diferente de NULL y es el valor deseado * Toda la memoria dinámicamente asignada sea liberada cuando deje de ser usada o al final del programa. | N/A | N/A | N/A | N/A |
| Variables | Verificar que no existan variables sin usar. | X | X | X | N/A |
| Operadores lógicos | * Verificar el uso correcto de ==, =, ¡=, ||, && * Verificar la asociación entre operadores y sus resultados posibles | X | X | X | N/A |
| Formato de salida | Verificar el formato de salida   * Sangría y espaciado * Mayúsculas y minúsculas | X | X | X | X |
| Archivos | Verificar que todos los archivos   * Existan * Sean abiertos correctamente * Sean cerrados cuando su uso termine | X | X | N/A | N/A |
| Errores | Definir todos los errores posibles  Agregar cada error encontrado al log de errores  Manejar adecuadamente cada error encontrado  Alertar al usuario de los errores encontrados | X | X | X | X |
| Estándar | Verificar que cumpla con el estándar de codificación.   * Bloque inicial de comentarios * Comentarios multilínea en cada función y método * Comentarios de una sola línea relevantes a lo largo del código.   Verificar que cumpla con el estándar de contabilización.   * Etiquetas de líneas base con número de líneas originales * Etiquetas de ítems en cada función y método, menos getters y setters * Etiquetas de líneas borradas con número de líneas   Etiquetas de líneas modificadas |  |  |  |  |