**Checklist de Revisión del Diseño**

|  |  |  |  |
| --- | --- | --- | --- |
| Estudiante | Iker Arbulu Lozano | Fecha | 2/Abril/2016 |
| Programa | Programas PSP | # Programa | 5 |
| Instructor | Rafael Salazar | Lenguaje | Java |

|  |  |
| --- | --- |
| 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: | | Programa1 | CalculadorX |
| Completez | Verificar que el diseño cubra a todos los requerimientos aplicables:   * Se producen todos los outputs necesarios * Todos los inputs se procesan * La extensión del sistema está definida (interfaces) |  |  |
| Casos Especiales | * Se revisaron todos los casos especiales * Se dan las respuestas definidas en caso de casos especiales * Se hace un handle donde es necesario porque dan valores ilegales * Hacer que las condiciones imposibles sean imposibles |  |  |
| Clases | * Existe una main class que se puede ejecutar * Existe una clase para cada objeto que se quiere instanciar * Existe una manera de acceder métodos de procesamiento dentro del objeto * Existe un diagrama que explique la relación entre las clases |  |  |
| Funciones | * Asegurarse que existan funciones para cumplir con el flujo de cada caso de uso * Asegurarse que existan los parámetros necesarios del lado del emisor * Asegurarse que se manejen de mantengan los parámetros del lado receptor |  |  |
| Logica | * Estructuras de datos a utilizar propias para la solución * Estructuras de control a utilizar propias para la solución * Verificar condicionales para saber que son exclusivas * Verificar inicio y fin de ciclos para que cumplan con las iteraciones necesarias * Verificar que las comparaciones de tipos de dato sean válidas |  |  |

**Checklist de Revisión del Código**

|  |  |  |  |
| --- | --- | --- | --- |
| Estudiante | Iker Arbulu Lozano | Fecha | 4/Abril/2016 |
| Programa | Entrenamiento PSP | # Programa | 5 |
| Instructor | Rafael Salazar | Lenguaje | Java |

|  |  |
| --- | --- |
| 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: | | Programa1 | CalculadorX |
| Completez | Verificar que el código cumple con todo el diseño |  |  |
| Includes | Verificar que se están incluyendo todas las librerías a utilizar |  |  |
| Strings | * Revisar que sean escritos con comillas dobles * Revisar que se asigne a una variable de tipo string y no char * Revisar posibles escapes que se les pueda hacer |  |  |
| Formato de Salida | * Revisar que se imprimen los espacios necesarios para el output * Revisar que se están realizando los saltos de línea o que se usó un println |  |  |
| {} Pairs | Revisar que abren y cierran de manera correcta |  |  |
| Operadores Lógicos | * Revisar que estén dentro de una pareja de () * Revisar que estén comparando datos del mismo tipo * Revisar que estén bien construidos con doble == para no realizar asignaciones |  |  |
| Catch exception | * Revisar que métodos utilizan catch exception e implementarlos * Revisar que tipo de excepción arrojan y atraparla |  |  |
| Streams | * Revisar que se abre de manera correcta * Revisar que se cierra de manera correcta * Revisar que se atrapa cualquier tipo de error * Revisar que tenga un fin en el stream |  |  |
| Funciones | * Revisar que mande parámetros del mismo tipo de dato * Revisar que exista la función que se intenta llamar * Revisar que cuente con los parámetros necesarios * Revisar que regrese el tipo de dato para el que está prototipado |  |  |