Nombre de la Institución:

Instituto Tecnológico de Costa Rica.

Nombre del curso:

Programación orientada a objetos.

Número de grupo:

1

Título del trabajo:

Programa 3 Mantenimiento de software, mejoras al programa Ken Ken.

Nombre del estudiante:

Luis Andrés Arrieta Víquez.

Semestre y año:

ll semestre 2023

Nombre del profesor:

William Mata Rodríguez.

Contenido de la documentación

Enunciado del proyecto:

En la ingeniería de software el ciclo de vida de desarrollo de software (SDLC: System Development Life Cycle) se refiere al conjunto de etapas necesarias para la creación y utilización de software a través del tiempo.

Hay diferentes metodologías que soportan este ciclo de vida, algunas de ellas son: modelo en cascada (el más antiguo), modelo en espiral, prototipos, modelo incremental e iterativo, desarrollo ágil (usando herramientas como SCRUM, programación extrema-XP, Kanban, etc.).

El mantenimiento de software es una actividad natural del ciclo de vida del software, es muy común que nos encontremos trabajando en ello. Consiste en modificar el software por diversas razones:

- Corrección de errores

- Adaptar el software a nuevos requerimientos

- Mejoras (funcionales y técnicas)

En este proyecto vamos a hacer mantenimiento al software realizado en el proyecto anterior: específicamente vamos a hacer mejoras de dos tipos:

- A nivel funcional: se agregarán nuevas funcionalidades.

- A nivel técnico: se implementará algún patrón básico de diseño seleccionado por el programador.

REQUERIMIENTOS DEL PROGRAMA NUEVAS FUNCIONALIDADES

A) OPCIÓN DE JUGAR CON DIFERENTES TAMAÑOS DE CUADRÍCULA

1) La versión actual del software es para jugar con el tamaño de cuadrícula 6x6. Esta nueva funcionalidad habilitará juegos con cuadrículas de tamaño 3x3, 4x4, 5x5, 6x6, 7x7, 8x8 y 9x9. Considere que debe modificar todos los componentes del software necesarios para proveer esta nueva funcionalidad, entre ellos: tamaño, despliegue y manejo de la cuadrícula según seleccione el jugador, panel de números permitidos y reglas de juego.

2) A la opción configurar se le agrega un nuevo parámetro:

5. Tamaño de la cuadrícula **o** 3x3
o 4x4
o 5x5
o 6x6
o 7x7
o 8x8
o 9x9

3) Modificación de la estructura del archivo de partidas que para esta versión se llamará kenken_partidas2023.xml: se agrega el dato de cuadrícula (antes del dato de nivel de dificultad):

```
<KenKen>
    <partida>
        <cuadrícula>tamañoCuadrícula</cuadrícula> 3x3/4x4/5x5/…
        <nivel de dificultad>nivelDificultad</nivel de dificultad>
        fácil/intermedio/difícil
        <jaula>valor, operación aritmética, (fila, columna), (fila,
        columna)), … </jaula>
        <jaula>valor, operación aritmética, (fila, columna), (fila,
        columna)), … </jaula>
        …
        <constantes>(constante, fila, columna), (constante, fila,
        columna), …</constantes>
    </partida>
</KenKen>
```

Ejemplo para una partida con tamaño de cuadrícula 6:

```
<KenKen>
    <partida>
        <cuadrícula>6x6</cuadrícula>
        <nivel de dificultad>fácil</nivel de dificultad>
        <jaula>11, +, (1,1), (1,2), (2,1)</jaula>
        <jaula>120, x, (1,3), (2,2), (2,3)</jaula>
        <jaula>3, +, (1,4), (2,4)</jaula>
        <jaula>2, -, (1, 6), (2, 6)</jaula>
        <jaula>11, +, (2, 5), (3, 5)</jaula>
        <jaula>3, +, (3, 1), (3, 2)</jaula>
        <jaula>15, x, (3, 3), (3, 4)</jaula>
        <jaula>1, -, (3,4), (4,4)</jaula>
        <jaula>72, x, (4,1), (4,2), (5,1)</jaula>
        <jaula>8, x, (4,5), (4,6)</jaula>
        <jaula>3, +, (5,2), (5,3)</jaula>
        <jaula>13, +, (5,4), (6,4), (6,5)</jaula>
        <jaula>9, +, (5,6), (6,6)</jaula>
```

```
            <jaula>2, /, (6,1), (6,2)</jaula>
            <constantes>(3, 1, 5), (6, 3, 6), (1, 6, 3)</constantes>
        </partida>
</KenKen>
```

B) OTROS CAMBIOS EN LA OPCIÓN JUGAR

1) Agregar el dato Nombre del jugador (string de 1 a 40 caracteres): antes de iniciar el juego el jugador debe obligatoriamente dar un nombre.

2) Podio

Se refiere a los mejores 3 jugadores de cada nivel de dificultad para cada tamaño de la cuadrícula. Los mejores jugadores son los que completan el juego en el menor tiempo. El programa registra esta información en el archivo “kenken2023podio.dat” cuando el jugador usa el botón VALIDAR JUEGO. En ese momento el programa debe determinar si el jugador debe registrarse en el Podio. Las partidas que no usan reloj no entran al Podio. Si se tienen los 3 mejores jugadores y el jugador actual hace un mejor tiempo que esas marcas, hay que eliminar la marca con mayor tiempo para seguir teniendo un máximo de 3 por nivel y cuadrícula. La marca contiene el nombre del jugador y el tiempo (horas, minutos, segundos) que el jugador usó para completar un juego.

Registre también la fecha y hora del sistema. Note que si usa el timer hay que calcular la duración del juego (al tiempo establecido inicialmente se le resta el tiempo en que se detuvo el timer).

3) Botón nuevo

Esta opción se puede usar en cualquier momento del juego. Detiene el reloj si lo está usando.

Despliega los registros de los mejores 3 jugadores por cada nivel de dificultad para cada tamaño de cuadrícula. Esta información se toma del archivo “kenken2023podio.dat”.

La primera información que debe desplegarse es la del podio correspondiente al nivel y tamaño de la cuadrícula indicada en la opción

jugar. Por ejemplo, si está en el nivel FÁCIL con una cuadrícula 6 x 6 se despliega primero ese podio y luego los demás:

PODIO DEL NIVEL ACTUAL

NIVEL FÁCIL		*JUGADOR*	*TIEMPO*
6 x 6	ORO	Nombre del jugador	0:10:11
	PLATA	Nombre del jugador	0:10:15
	BRONCE	Nombre del jugador	0:11:20

PODIO GENERAL POR NIVEL Y CUADRÍCULA

NIVEL DIFÍCIL:		*JUGADOR*	*TIEMPO*
3 x 3	ORO	Nombre del jugador	0:05:15
	PLATA	Nombre del jugador	0:05:55
	BRONCE	Nombre del jugador	0:06:59
6 x 6	ORO	Nombre del jugador	0:35:19
	PLATA	Nombre del jugador	0:36:38
NIVEL INTERMEDIO:		*JUGADOR*	*TIEMPO*
6 x 6	ORO	Nombre del jugador	0:30:19
	PLATA	Nombre del jugador	0:31:38
	BRONCE	Nombre del jugador	0:32:00
NIVEL FÁCIL:		*JUGADOR*	*TIEMPO*
4 x 4	ORO	Nombre del jugador	0:04:00
	PLATA	Nombre del jugador	0:04:49

C) PATRÓN BÁSICO DE DISEÑO

El programador selecciona algún patrón básico de diseño. Ejemplos: patrón Singleton para aplicarlo de tal forma que garantice la creación de solo una instancia con la lista de partidas de juego almacenadas en el archivo kenken_partidas2023.xml, patrón Factory para crear objetos según la cuadrícula que se requiera en el momento para jugar (3x3, 4x4, etc.), patrón Observer para tener la lista de jugadores de tal forma que cuando ocurra un cambio en algún podio se les informe mediante un mensaje o correo, etc.

DOCUMENTACIÓN DEL PROYECTO

Trabajo en grupos de 2 personas máximo: se mantendrán los grupos del programa anterior.

Se coordinará un día y hora para revisar el proyecto junto con los estudiantes del grupo, quienes siendo sus autores deben demostrar un completo dominio de la solución implementada tanto desde el punto de vista técnico como de la funcionalidad (lo que hace la solución). En la revisión se pueden realizar estas actividades:

- Revisar esta solución particular
- Revisar conceptos incluidos en la evaluación
- Aplicar otras actividades con una complejidad igual o menor a la evaluación.

Se revisan los proyectos que cumplan con todos estos requisitos:

a- El programa debe tener documentación interna y usar JavaDoc como parte de esa documentación (no hace falta documentar los setters/getters)

b- Desarrollar en Java usando GUI.

c- Usar algún software de control de versiones y trabajo colaborativo (por ejemplo, github).

d- La nota de la documentación del proyecto indicada abajo sirve para aceptar o rechazar el proyecto: se revisan los proyectos que cumplan con esa documentación en un 90% o más.

e- Para revisar el proyecto deben estar todos los miembros del grupo.

Enviar vía tecDigital, sección EVALUACIONES / PROGRAMAS, una carpeta comprimida (nombre **programa3_sus_nombres.zip**) que contenga las siguientes partes:

PARTE 1: Documentación del proyecto en un archivo formato PDF.

Nombre de esta documentación:
programa3_documentación_del_proyecto.PDF)

- Portada. (1P)
- Contenido de la documentación. (2P) -enumere las páginas, use letra Arial 12, espaciado sencillo, márgenes convencionales-
- Enunciado del proyecto -esta especificación-. (2P)
- Temas investigados: básicamente sería el patrón básico de diseño seleccionado por el programador. Organice esta sección al menos con los siguientes puntos porcada tema investigado: (0P o 25P)

- Título del tema investigado

- Detalle de cada aspecto del tema que fue necesitado en el programa

- Cómo se aplicó al programa

- Bibliografía (libros, manuales, sitios de internet, etc.)

Omita en temas investigados la parte relacionada con el uso de interfaces gráficas.

▪ Solución (0P o 25P) o Modelo del sistema con un diagrama de clases UML que incluya (actualizar modelo del programa 2 para que contenga mejoras de este nuevo programa):

▪ Atributos

▪ Métodos (no incluya los setters/getters)

▪ Relaciones entre los objetos de las clases: dependencia, asociación, agregación, etc.

▪ Navegabilidad, multiplicidad

▪ Opcionalmente nombre de asociaciones y roles

En el diagrama no presente atributos ni estructuras de datos que soportan la implementación de las relaciones, el diagrama al mostrar las relaciones muestra esos aspectos que luego se tratan en la implementarán. Los constructores de cada clase deben ser con parámetros. No se permiten componentes duplicados.

▪ Conclusiones del trabajo: (15P)

- Problemas encontrados y soluciones a los mismos.
- Aprendizajes obtenidos.

▪ Lista de revisión del proyecto (PONGA ESTA LISTA EN PÁGINA NUEVA). (0P o 15P)

- Por cada concepto de la lista de revisión usted debe indicar el % de avance que logró, los puntos obtenidos según ese avance y el análisis de resultados de su proyecto:

  ▪ 100: Totalmente desarrollado. No hace falta análisis excepto que requiera hacer alguna observación.

  ▪ Un % específico, por ejemplo 80 significaría un desarrollo parcial del 80%. En el análisis indicar tres partes: ¿qué hace?, ¿qué falta?, ¿por qué no se completó?

  ▪ 0: No desarrollado. En el análisis indicar ¿por qué no se desarrolló?

- Partes que desarrolló adicionales a lo solicitado en el proyecto.

▪ Manual de usuario (0P o 15P): actualizar el manual del programa 2 para que incluya la funcionalidad de este nuevo programa.

PARTE 2: carpeta del proyecto con el nombre programa3_ken_ken

Temas investigados:

1.1 Patrón básico de diseño: Factory

1.1.1 Detalles de cada aspecto del tema que fue necesitado en el programa.

Hay que comenzar diciendo que el patrón básico de diseño Factory, “es un patrón de diseño creacional que proporciona una interfaz para crear objetos en una superclase, mientras permite a las subclases alterar el tipo de objetos que se crearán” (Refactoring Guru, 2023, p.1).

El patrón básico de diseño Factory es de bastante utilidad ya que en vez de llamar al new, cuando se requiera construir un objeto directamente, se utiliza un método de Factory especial, que de igual forma, si utiliza el new, pero cada objeto es invocado de forma organizada desde este método Factory.

Este patrón básico de diseño fue necesitado en el programa ya que evita que se haga un acoplamiento grande entre el creador y los productos concretos, además proporciona un principio de responsabilidad al programa, ya que permite mover el código de creación de producto a un lugar cualquiera del programa, de tal forma, que el código sea más sencillo de mantener.

1.1.2 Cómo se aplicó al programa.

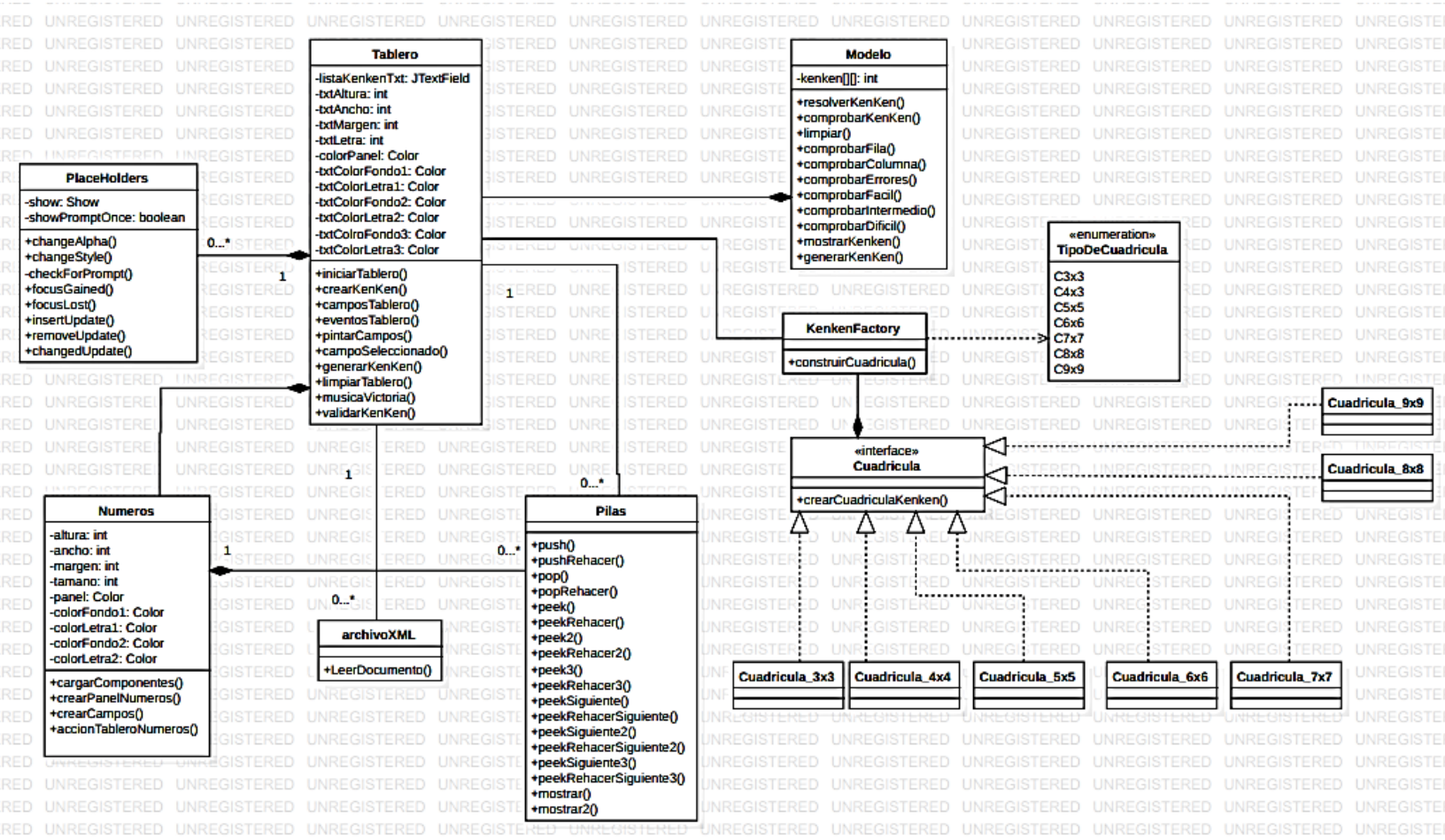
En el programa se aplicó el patrón básico de diseño Factory para poder generar y construir objetos de tableros del juego Kenken según por la cuadrícula que se haya seleccionado en la configuración del juego: 3x3, 4x4, 5x5, 6x6, 7x7, 8x8 o 9x9, todo esto gracias a una interfaz y algunas clases, por ejemplo, la interfaz llamada Cuadricula, se utilizó para crear el método abstracto llamado crearCuadriculaKenken(), donde diversas clases de cada tipo de cuadricula, implementan esta interfaz y utilizan ese método abstracto, pero ya definiendo que hace cada una dependiendo del tipo de cuadrícula, además se utilizó una clase llamada KenkenFactory, la cual se usó para indicar cual cuadricula se va a utilizar para jugar el Kenken, creando un nuevo objeto, de acuerdo al "TipoDeCuadricula", que proviene de una clase con ese nombre y crea datos tipo enum.

1.1.3 Bibliografía.

Factory Method.

https://refactoring.guru/es/design-patterns/factory-method

Solución:


Tablero
-listaKenkenTxt: JTextField
-txtAltura: int
-txtAncho: int
-txtMargen: int
-txtLetra: int
-colorPanel: Color
-txtColorFondo1: Color
-txtColorLetra1: Color
-txtColorFondo2: Color
-txtColorLetra2: Color
-txtColroFondo3: Color
-txtColorLetra3: Color
+iniciarTablero()
+crearKenKen()
+camposTablero()
+eventosTablero()
+pintarCampos()
+campoSeleccionado()
+generarKenKen()
+limpiarTablero()
+musicaVictoria()
+validarKenKen()
Modelo
-kenken[][]: int
+resolverKenKen()
+comprobarKenKen()
+limpiar()
+comprobarFila()
+comprobarColumna()
+comprobarErrores()
+comprobarFacil()
+comprobarIntermedio()
+comprobarDificil()
+mostrarKenken()
+generarKenKen()
PlaceHolders
-show: Show
-showPromptOnce: boolean
+changeAlpha()
+changeStyle()
-checkForPrompt()
+focusGained()
+focusLost()
+insertUpdate()
+removeUpdate()
+changedUpdate()
«enumeration»
TipoDeCuadricula
C3x3
C4x3
C5x5
C6x6
C7x7
C8x8
C9x9
KenkenFactory
+construirCuadricula()
«interface»
Cuadricula
+crearCuadriculaKenken()
Cuadricula_9x9
Cuadricula_8x8
Numeros
-altura: int
-ancho: int
-margen: int
-tamano: int
-panel: Color
-colorFondo1: Color
-colorLetra1: Color
-colorFondo2: Color
-colorLetra2: Color
+cargarComponentes()
+crearPanelNumeros()
+crearCampos()
+accionTableroNumeros()
Pilas
+push()
+pushRehacer()
+pop()
+popRehacer()
+peek()
+peekRehacer()
+peek2()
+peekRehacer2()
+peek3()
+peekRehacer3()
+peekSiguiente()
+peekRehacerSiguiente()
+peekSiguiente2()
+peekRehacerSiguiente2()
+peekSiguiente3()
+peekRehacerSiguiente3()
+mostrar()
+mostrar2()
archivoXML
+LeerDocumento()
Cuadricula_3x3
Cuadricula_4x4
Cuadricula_5x5
Cuadricula_6x6
Cuadricula_7x7
0...*
1
1
1
0...*
1
0...*
0...*

Conclusiones del trabajo:

Problemas encontrados y soluciones a los mismos:

En el desarrollo de todo el programa de “mantenimiento de software, mejoras al programa KenKen”, los principales problemas que encontré en un inicio fue sobre cómo desarrollar aparte de la cuadrícula 6x6, cada una de las otras cuadrículas (3x3, 4x4, 5x5, 7x7, 8x8, 9x9) ya que no sabía cómo modificar diversas cosas del programa de tal manera que dependiendo de la cuadrícula seleccionada, realmente se desplegara la cuadrícula correctamente, con un panel de números cambiado para cada número permitido según la cuadrícula, y las reglas de juego, entonces, después de pensarlo un rato pensé que la mejor idea para hacer funcionar cada uno de esos cambios, era con la utilización de variables globales en la configuración del juego, haciendo que se active en true la variable global de la cuadricula que el jugador haya seleccionado, y con base a eso, ir implementando varios if dependiendo del tipo de cuadricula, por ejemplo, en la clase “Tablero”, dependiendo del tipo de cuadricula seleccionada y con el uso de condicionales if, si la cuadricula era 3x3, entonces el atributo llamado “listaKenkenTxt”, iba a ser un JTextField[3][3], pero si la cuadricula era 7x7, entonces su valor sería de un JTextField[7][7], y así con los demás, en la clase llamada “Modelo”, con ayuda de condicionales if y dependiendo de la variable global de tipo de cuadrícula que estaba en true, cambia el valor del atributo llamado “kenken”, (utilizado para hacer validaciones en el juego en fila, columna, y dificultad), en kenken = new int [4][4], por ejemplo, o kenken = new int [6][6], además cambiando el valor de algunas variables llamadas matriz, para que tengan el valor de la nueva respuesta del tablero kenken, según por el tipo de cuadrícula, y dificultad elegida. Y con respecto al panel de números actualizado, dependiendo del valor de la variable global de cada uno de los tipos de cuadricula, junto a la ayuda de condicionales if, se cambia el tamaño, ancho, y cantidad de número máximo en el ciclo for que crea los JTextField del panel de números.

Y para concluir, con respecto a otro de los problemas que encontré durante el desarrollo del programa pero que pude solucionar, fue la implementación de algún patrón básico de diseño, ya que para implementar cualquiera de esos patrones debía investigar bastante, y comprender como aplicarlo en el programa, entonces, el que decidí utilizar fue el Factory, después de haber investigado, y haberme dado cuenta de que su uso también me iba a funcionar para mi problema mencionado anteriormente, fui solucionado el problema, comenzando con la creación de una interfaz llamada Cuadricula para crear un método abstracto, después, lo implementé a cada una de las clases de tipos de cuadricula, junto a ese método abstracto, pero ya con una funcionalidad dependiendo del tipo de cuadricula, y la dificultad, ya que, esa función es necesaria para generar el juego del kenken, del tipo de cuadricula y con sus respectivas reglas de juego, implementando los bordes en los JTextField según corresponda, y los números con signos de suma, resta, multiplicación, o división, a modo de place holders, además, utilicé una clase llamada KenkenFactory, que es la fundamental para el

patrón Factory, ya que indica cual cuadricula se va a utilizar para jugar el Kenken, creando un nuevo objeto, de acuerdo al "TipoDeCuadricula", que es otra clase con datos de tipo enum, y todos esas clases e interfaz, las llamé en la función generarKenKen(), de la clase Tablero, creando un objeto de la clase KenkenFactory, aplicando ya el patrón básico de diseño Factory, y reduciendo bastantes líneas de código en la clase “Tablero”.

Aprendizajes obtenidos:

En este tercer proyecto de programación orientada a objetos, tuve un gran aprendizaje, ya que es la primera vez en la que aplico en un proyecto de programación en Java, mantenimiento al software, tanto a nivel funcional, como a nivel técnico en dónde se debía de implementar un patrón de diseño seleccionado por uno mismo, y agregarle distintas y nuevas funcionalidades al programa, y en dónde a partir de diversas investigaciones, sobre todo en el patrón de diseño Factory, para generar un juego de Kenken dependiendo del tipo de cuadricula, aprendí bastante sobre ese patrón de diseño.

Para concluir, el saber que realizar cada vez más programas como estos, en donde esta vez principalmente era sobre el mantenimiento de software, y el saber que me ayudan bastante a obtener más conocimientos, practica, y lógica en la programación, es algo que me alegra y me motiva, ya que la mejor forma de aprender a programar, es a partir de la misma práctica constante, y a partir de cada error durante la programación, por ejemplo, pude aprender a ir disminuyéndolos y saber qué cosas no deben realizarse, además la investigación para adquirir conocimientos nuevos es de utilidad en programaciones futuras, y en este tercer proyecto, se reflejó eso, ya que dedique bastante tiempo al proyecto, y noté cada vez más, avances en mi lógica y en la misma programación.

Lista de revisión del proyecto:

Concepto	Puntos	% de avance 100%/0	Puntos obtenidos	Análisis de resultados
Opción Jugar (7 cuadrículas): Desplegar cuadriculado Desplegar información de cada casilla según partida Enmarcado de las jaulas según partida Desplegar el resto de la opción	 7 7 7 7	100%	 7 7 7 7	
Iniciar Juego: Actualizar casillas (colocar y borrar números) Controles del proceso	 7 7	100%	 7 7	
Validar Juego	7	100%	7	
Opción 5 de configurar	5	100%	5	
Modificación archivo de partidas kenken_2023.xml	3	100%	3	
Nombre jugador	2	100%	2	
Crear podio en opción VALIDAR JUEGO	6	0%	0	Por qué no se desarrolló: No se desarrolló la creación del podio en la opción de validar juego, ya que no tuve alguna idea en mente de como programar el podio de manera efectiva
Botón Podio Desplegar primero podio nivel actual Desplegar otros podios	 5 5	0%	 0 0	Por qué no se desarrolló: No se desarrolló el botón del podio, debido a que no tenía alguna idea en mente de cómo poder programar el podio de manera correcta
Patrón básico de diseño seleccionado por el programador	15	100%	15	
Ayuda (despliegue del PDF del manual de usuario)	2	100%	2	
Documentación del proyecto: Manual de usuario Resto de la documentación	 4 4	100%	 4 4	
TOTAL	**100**	84%	84	Es un 84% con base a la suma de los puntos obtenidos

Funcionalidades desarrolladas adicionalmente				En este caso no se desarrolló alguna parte adicional.