Nombre de la Institución:

Instituto Tecnológico de Costa Rica.

Nombre del curso:

Programación orientada a objetos.

Número de grupo:

1

Título del trabajo:

Programa 2 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:

KenKen es un pasatiempo aritmético inventado alrededor del 2004 por el profesor japonés de matemáticas Tetsuya Miyamoto para ayudar a sus estudiantes a aprender aritmética básica.

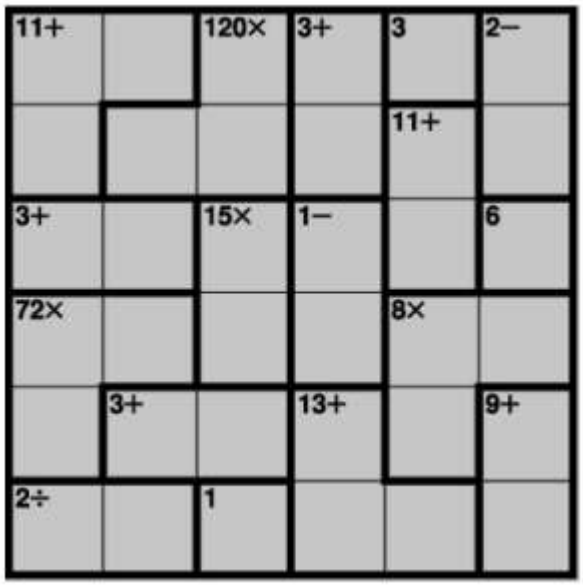
El nombre KenKen significa en japonés cuadrado inteligente. Inicialmente se crearon libros con el juego y desde hace algunos años se popularizó al aparecer en diversos periódicos,revistas y en programas de computadora, entre ellos https://www.kenkenpuzzle.com/game.

Se considera el sucesor del pasatiempo Sudoku el cual hace un manejo de números, pero sin las operaciones aritméticas.

KenKen combina números con las cuatro operaciones básicas de aritmética: suma, resta, multiplicación y división.

Se juega en una cuadrícula (matriz) que puede tener desde 9 casillas (cuadrícula de menor tamaño) para juegos de 3 x 3 (3 filas, 3 columnas) usando los números del 1 al 3, hasta 81 casillas (cuadrícula de mayor tamaño) para juegos de 9 x 9 (9 filas, 9 columnas) usando los números del 1 al 9.

En este proyecto vamos a implementar el juego con un tamaño de cuadrícula 6 x 6 (6 filas, 6 columnas) usando los números del 1 al 6. Ejemplo:



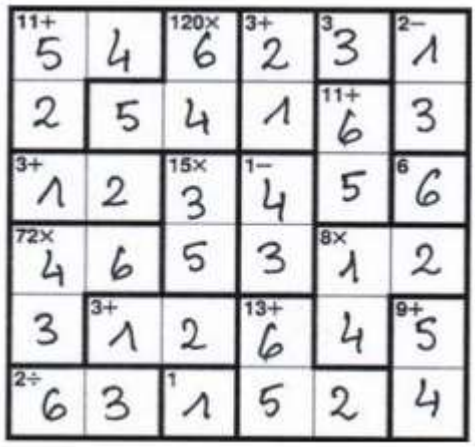
Reglas del juego:

1) Las cuadrículas son organizadas en jaulas, donde una jaula es un grupo de casillas enmarcadas. Los números que se coloquen en las jaulas, en este caso del 1 al 6 por ser un KenKen 6 x 6, deben dar como resultado el número ubicado en la casilla superior izquierda de la jaula utilizando la operación indicada en esa misma casilla. Ejemplos: la jaula que agrupa las casillas que están en fila 1 columna 1, fila 1 columna 2 y fila 2 columna 1 tiene la operación “11+” lo cual significa que la suma de esas tres casillas debe dar 11. La jaula que agrupa las casillas que están en la fila 4 columna 5, fila 4 columna 6 y fila 5 columna 5 tiene la operación “8x” lo cual significa que la multiplicación de esas casillas debe dar 8.

2) Cada fila y columna deben contener sin repetición los números del 1 al 6.

3) Las casillas que tienen un número sin una operación, por ejemplo, la casilla en fila 1 columna 5, indican que en esa casilla va específicamente ese número.

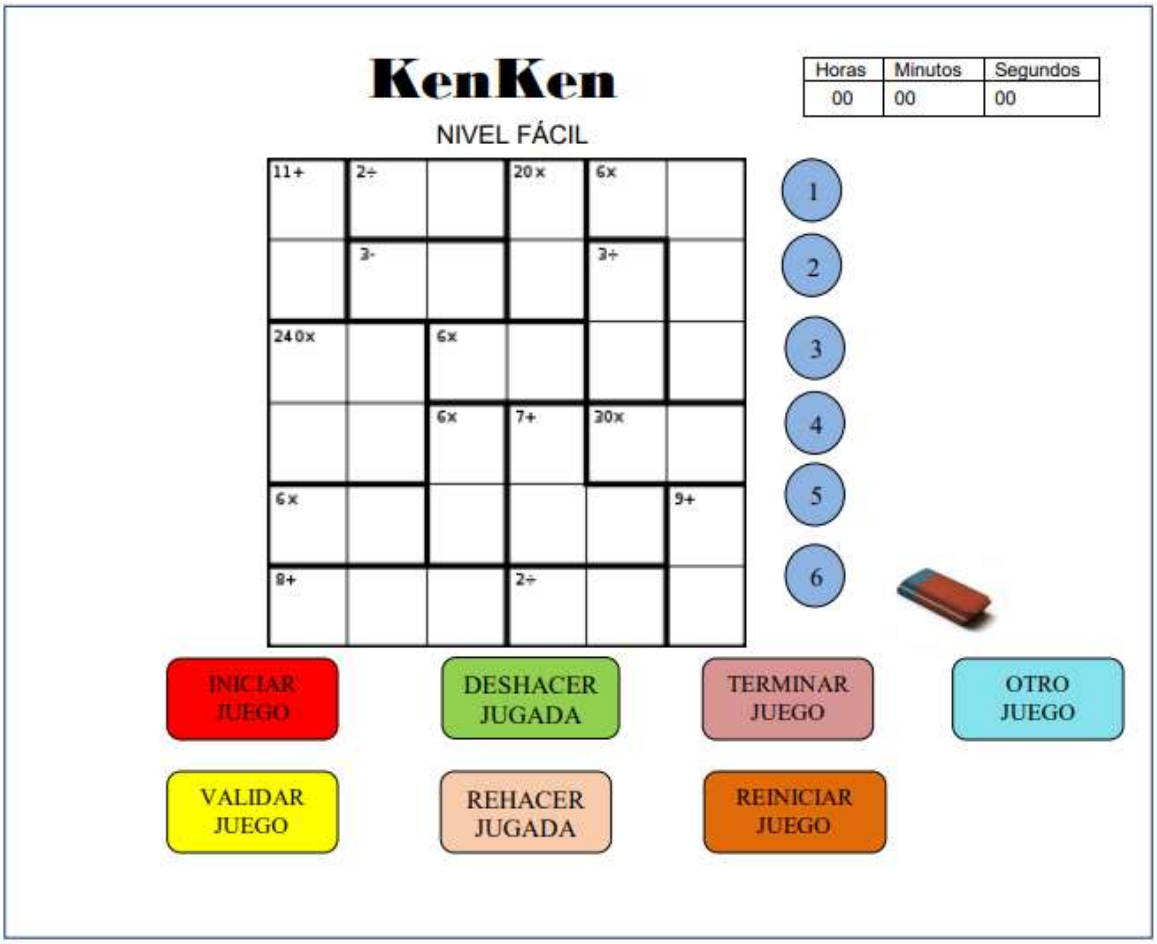
Solución al juego del ejemplo:



REQUERIMIENTOS DEL PROGRAMA

A) Jugar

Esta opción permite jugar el KenKen. Cuando se da esta opción se muestra una pantalla como la siguiente según la configuración:



El programa toma los juegos previamente registrados en el archivo kenken_partidas.xml y debe seleccionar aleatoriamente uno según el nivel de dificultad configurado. Java tiene funciones para generar números aleatorios que pueden servir para seleccionar alguno de los juegos de tal forma que siempre se elija uno al azar. Puede usar otros algoritmos para esta selección aleatoria de juegos. Explique en la documentación del proyecto el algoritmo que va a usar considerando que en una misma corrida del programa si hay n juegos para un nivel, primero se deben escoger los n juegos en forma aleatoria antes de volver a repetirlos.

Uso de los botones:

El jugador clica (hace clic) en este botón para iniciar el juego.

¿Cómo poner un número en una casilla de la cuadrícula?

El jugador clica una casilla válida de la cuadrícula en la cual quiere poner un número. Esa casilla cambia al color celeste para identificarla de las demás que tienen el color blanco. Luego clica en el panel de números aquel que va a poner en la casilla seleccionada. La casilla sigue manteniendo el color como casilla seleccionada hasta que otra casilla sea seleccionada. Si clica una casilla a la que anteriormente le asignó un número quiere decir que va a sustituir ese contenido. Para borrar el contenido de una casilla clica sobre ella y luego sobre el ícono del “borrador”.

Otras consideraciones:

- Luego de dar este botón hay que deshabilitarlo.

- En caso de seleccionar un número o el borrador antes de seleccionar una casilla se envía el mensaje PRIMERO DEBE SELECCIONAR UNA CASILLA.

- En caso de haber configurado la opción de Timer, el jugador puede dejar el tiempo registrado en la configuración o modificarlo antes de “INICIAR JUEGO”. El tiempo empieza a correr cuando le den “INICIAR JUEGO”. Hace las mismas validaciones que en la configuración.

- En caso de no usar el reloj no debe aparecer esa parte en la ventana del juego.

- En caso de haber configurado la opción de Timer y éste llegue a 0 y el juego no haya terminado se envía el mensaje TIEMPO EXPIRADO. ¿DESEA CONTINUAR EL MISMO JUEGO (SI/NO)?. Si responde SI entonces el timer pasa a ser reloj inicializado con el tiempo que se había establecido en el timer.

Por ejemplo, si el timer estaba para 1 hora y 30 minutos, ahora el reloj debe marcar que ya ha pasado 1 hora y 30 minutos y sigue contando el tiempo. Si responde NO el juego finaliza regresando a la opción de Jugar.

- En caso de no existir algún juego en el archivo para el nivel seleccionado se da el mensaje NO HAY JUEGOS PARA ESTE NIVEL y el programa regresa al menú principal.

Para cualquier situación que cause un mensaje, el programa se detiene, despliega el mensaje y espera a que el usuario dé la tecla <Enter> para continuar.

Luego de iniciar el juego este botón se puede usar en cualquier momento para verificar que las jugadas estén según las reglas del juego.

Las jaulas que tengan al menos un valor y que no cumplan con las reglas del juego se pondrán en color rojo y se envía el mensaje “HAY ERRORES EN EL JUEGO”. Así el jugador debe proceder a hacer los arreglos.

El juego termina cuando el jugador selecciona este botón y todas las casillas cumplen con las reglas del juego, ahí para el reloj o el timer (en caso de usarlos) y despliega el mensaje ¡FELICITACIONES, JUEGO COMPLETADO!

Con las felicitaciones revisar la configuración para determinar si debe poner un sonido, ejemplos: aplausos, silbato, música, etc. Usted decide que sonido poner.

Cuando un juego es completado el programa regresa a la opción de Jugar.

Se puede seleccionar esta opción solamente si el juego ha iniciado de lo contrario hay que enviar el mensaje NO SE HA INICIADO EL JUEGO. Otra forma para controlar esto es deshabilitar el botón cuando éste no pueda usarse.

Use la estructura de datos pila para ir registrando cada jugada en la que se asignó o borró un número. Operación push: cuando el jugador realiza una de esas jugadas. Operación pop: cuando el jugador usa este botón, se deshace o quita la última jugada que esté en la pila. ¿Cuántas jugadas se pueden deshacer? Las que al momento estén registradas en la pila. Un pop en esta pila también causa un push en la pila de jugadas deshechas.

También usa la estructura de datos pila para ir registrando cada jugada que se deshizo (con el botón DESHACER JUGADA).

Esta función rehace o reconstruye la última jugada que se deshizo. Operación push: cuando se deshace una jugada (con el botón anterior). Operación pop: cuando el jugador usa este botón, se rehace la última jugada que esté en la pila.

¿Cuántas jugadas se pueden rehacer? Las que al momento estén registradas en la pila. Un pop en esta pila también causa un push en la pila de jugadas realizadas.

Cuando el jugador selecciona esta opción se le pregunta

¿ESTÁ SEGURO DE TERMINAR EL JUEGO (SI/NO)?

Si responde SI termina de inmediato el juego y vuelve al menú del programa.

Si responde NO sigue jugando con el mismo juego.

Cuando el jugador selecciona esta opción se le pregunta

¿ESTÁ SEGURO DE EMPEZAR NUEVAMENTE ESTE MISMO JUEGO (SI/NO)?

Si responde SI termina de inmediato el juego y vuelve a la opción de Jugar mostrando el mismo juego sin jugadas.

Si responde NO sigue jugando con el mismo juego.

Se puede seleccionar esta opción solamente si el juego ha iniciado de lo contrario hay que enviar el mensaje NO SE HA INICIADO EL JUEGO. Otra forma para controlar esto es deshabilitar el botón cuando éste no pueda usarse.

Cuando el jugador selecciona esta opción se le pregunta

¿ESTÁ SEGURO DE TERMINAR ESTE JUEGO Y EMPEZAR CON OTRO (SI/NO)?

Si responde SI termina de inmediato el juego y vuelve a la opción de Jugar mostrando otro juego.

Si responde NO sigue jugando con el mismo juego.

Se puede seleccionar esta opción solamente si el juego ha iniciado de lo contrario hay que enviar el mensaje NO SE HA INICIADO EL JUEGO. Otra forma para controlar esto es deshabilitar el botón cuando éste no pueda usarse.

B) Configurar

Con esta opción establecemos algunas condiciones para jugar. Contiene los siguientes datos que se van a guardar en el archivo “kenken_configuración.dat. Inicialmente cada dato tiene valores por omisión –o default- señalados con el círculo en rojo.

1. Nivel de dificultad
   - o Fácil
   - o Intermedio
   - o Difícil

2. Reloj:
   - o Cronómetro
   - o Timer
   - o No

En caso de seleccionar Timer hay que poner aquí un tiempo sugerido que puede ser cambiado aquí mismo o en la opción de Jugar antes de iniciar el juego:

Horas	Minutos	Segundos
0	0	0

Para el timer las horas pueden estar entre 0 y 5, los minutos entre 0 y 59 y los segundos entre 0 y 59. El timer debe tener al menos uno de estos valores. Hay que realizar estas validaciones y enviar los mensajes respectivos en caso de errores.

Para el caso de que la opción sea reloj todos esos datos deben ponerse en 0.

3. Posición del panel de números y el borrador:
   - o Derecha
   - o Izquierda

4. Sonido cuando termina el juego exitosamente:
   - o Si
   - o No

C) Ayuda

Esta opción despliega en la computadora el PDF que contiene el manual de Usuario.

D) Acerca de

Esta opción la usaremos para desplegar información “Acerca del programa” donde pondremos al menos los datos del nombre del programa, la versión, la fecha de creación y el autor (o autores).

E) Salir

Esta opción se usa para salir del programa (también se puede salir con el botón de cerrar “X” en la interfaz gráfica).

ARCHIVO kenken_partidas.xml

Este archivo contiene las partidas disponibles para jugar. Puede buscar algunas partidas, al menos 3 por nivel y grábelas directamente en el archivo, fuera de este programa ya que no es una funcionalidad del mismo pero se ocupa para hacer pruebas. El formato del archivo es xml (eXtensible Markup Language) con la siguiente estructura:

<KenKen>

<partida>

<nivel de dificultad>nivelDificultad</nivel de dificultad>

<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>

Dentro de cada jaula el valor y la operación aritmética se ponen en la casilla que está a continuación de esos datos.

nivelDificultad: fácil, intermedio, difícil

Ejemplo con la partida mostrada en la especificación:

<KenKen>

<partida>

<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>

DOCUMENTACIÓN DEL PROYECTO

Trabajo en grupos de 2 personas máximo: se mantendrán los grupos del programa anterior, en caso de algún cambio en el grupo envían un correo al profesor a más tardar el 15 de octubre.

Se coordinará un día y hora para revisar el proyecto junto con el estudiante, quien siendo su autor debe 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- Aplicar el patrón MVC (Modelo Vista Controlador).

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

e- 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.

Enviar vía tecDigital, sección EVALUACIONES / PROGRAMAS, una carpeta comprimida (nombre **programa2_sus_nombres.zip**) que contenga las siguientes partes:

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

Nombre de esta documentación:
programa2_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: cualquier material no estudiado en el curso, por ejemplo, software de control de versiones y trabajo colaborativo, manejo de archivos en

formato xml, etc. Organice esta sección al menos con los siguientes puntos por cada 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)

- Modelo del sistema con un diagrama de clases UML que incluya:
  - ▪ 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)

PARTE 2: carpeta del proyecto con el nombre programa2_ken_ken

NOTA IMPORTANTE: este proyecto será la base para desarrollar el siguiente proyecto.

Temas investigados:

1.1 Software de control de versiones y trabajo colaborativo

1.1.1 Detalles de los aspectos del tema que fueron necesitados en el programa.

Hay iniciar diciendo que el software de control de versiones y trabajo colaborativo, “es la práctica de rastrear y gestionar los cambios en el código de software.” (Atlassian, 2023, p.1).

Estos son de suma utilidad ya que permiten la facilitación en la gestión de cambios que se vayan realizando en el código fuente, a través de los días que van transcurriendo, y de esta manera varias personas que conforman un equipo, pueden ir haciendo cambios en el programa, en tiempo real.

Para este proyecto en específico, se utilizó github para el control de versiones, y trabajo colaborativo, el cual “es una plataforma de gestión y organización de proyectos basada en la nube que incorpora las funciones de control de versiones de Git.” (Gustavo, 2023, p.1).

Github permite a los distintos programadores y desarrolladores, un espacio en dónde se pueden alojar proyectos al permitir la creación de repositorios de manera gratuita.

1.1.2 Cómo se aplicó al programa.

En mi proyecto utilicé github para poder alojar las carpetas, y archivos .wav y .xml del proyecto, en un lugar donde es de bastante utilidad para el control de versiones y trabajo colaborativo.

1.1.3 Bibliografía.

Qué es el control de versiones.

https://www.atlassian.com/es/git/tutorials/what-is-version-control

Qué es GitHub y cómo empezar a usarlo

https://www.hostinger.es/tutoriales/que-es-github#%C2%BFQue_es_GitHub

2.1 **Manejo de archivos en formato xml**

2.1.1 Detalles de los aspectos del tema que fueron necesitados en el programa.

Para comenzar, se puede decir que XML “es un metalenguaje extensible de etiquetas desarrollado por el World Wide Weeb Consortium (W3C) que permite definir la gramática de lenguajes específicos.” (Junta de Andalucía, año desconocido, p.1).

El manejo de archivos en formato xml es de mucha utilidad ya que permite, su utilización en editores de texto, bases de datos, y otras cosas más, permitiendo el almacenamiento de datos importantes y el tener compatibilidad entre sistemas de una manera muy confiable.

El manejo de archivos en formato xml, en este caso concreto, para Java, necesito la creación del objeto DocumentBuilder, el cual, “es el objeto que permitirá tomar un archivo XML y convertirlo a un objeto Document del cual podremos consultar sus componentes.” (Raygoza, 2020, p.1). Y la obtención de elementos que se desean del XML, así como obtener sus atributos.

2.1.2 Cómo se aplicó al programa.

En mi proyecto se utilizó el manejo de archivos en formato xml, para tomar juegos del Kenken previamente registrados, y de esa manera poder leer su información, en dónde se especifica para cada uno, el nivel de dificultad, las jaulas, con su valor, operación aritmética en fila y columna, y el valor de los números constantes.

2.1.3 Bibliografía.

Librerías para el tratamiento de XML en Java.

https://www.juntadeandalucia.es/servicios/madeja/contenido/libro-pautas/348#libro-pautas-toc

Leer un archivo XML con Java.

https://medium.com/el-acordeon-del-programador/leer-un-archivo-xml-con-java-a598a9cdcc44

3.1 Utilización de timers y cronómetros en Java

3.1.1 Detalles de los aspectos del tema que fueron necesitados en el programa.

La utilización de timers y crónometros es utilizada en algunos proyectos de Java, en donde se requiera tomar tiempo de cuánto dura una acción en realizarse, o cuánto tiempo queda para que suceda algo en el programa.

Java permite la utilización de algunas clases para ayudar con funciones del programa que requieran timers o cronómetros, en este caso, se utilizó la clase llamada: “java.swing.Timer”, la cual es utilizada cuando queremos el aviso (como un aviso cada segundo, minuto o hora), y esta se encarga de llamar a un método que se haya creado anteriormente. (Autor desconocido, 2007).

En este programa en específico fue necesario el uso del ActionListener(), del start(), y del stop(), para poder implementar tanto el cronómetro como el timer, aprovechando las funciones de la clase javax.swing.Timer.

3.1.2 Cómo se aplicó al programa.

En el proyecto se utilizaron los timers y cronómetros, con ayuda de la clase javax.swing.Timer , para tomar los tiempos de cuánto duró una persona en completar una partida del Kenken de manera correcta, en el caso del cronómetro, y en el caso del timer, se utilizó para disponer de un tiempo límite que se iba reduciendo, para que el jugador tuviera un límite en el cual se debe completar el juego, y si el timer llegaba a 0, se le informaba al jugador que el tiempo ha expirado, y si deseaba o no continuar con el mismo juego.

3.1.3 Bibliografía.

Ejemplos con Timer en java.
https://old.chuidiang.org/java/timer/timer.php

4.1 Implementación de archivos de sonidos en Java.

4.1.1 Detalles de los aspectos del tema que fueron necesitados en el programa.

La utilización de sonidos en Java, es algo utilizado, para aquellos programas en donde se quiera crear una mejor ambientación, por ejemplo, a la hora de programar videojuegos, o simuladores.

Para implementar sonidos en el programa, uno de los aspectos importantes que se tuvieron que investigar fue el uso de la clase “javax.sound.sampled.Clip”, la cual representa un tipo especial de línea de datos en donde sus datos de audio se pueden cargar antes de su reproducción, o en tiempo real. (Oracle Help Center, 2020).

Además, se necesitó la clase llamada: “javax.sound.sampled.AudioSystem”, que es una clase que funciona como un punto de entrada a los recursos del sistema de audio que se ha cargado y muestreado (Oracle, 2020), permitiendo el acceso y la búsqueda a los archivos de audio, que en este caso fue uno de tipo .wav, instalado en el sistema.

4.1.2 Cómo se aplicó al programa.

En el proyecto se aplicó el uso de sonidos en Java, mediante clases como javax.sound.sampled.Clip, javax.sound.sampled.AudioSystem, y javax.sound.sampled.AudioInputStream;, y se usó en el proyecto para que, a la hora de que el jugador presionara el botón llamado “Validar Juego”, sonara un sonido, si este se había habilitado en las configuraciones del mismo programa del Kenken.

4.1.3 Bibliografía.

Interface Clip.
https://docs.oracle.com/javase%2F7%2Fdocs%2Fapi%2F%2F/javax/sound/sampled/Clip.html

Class AudioSystem
https://docs.oracle.com/javase%2F7%2Fdocs%2Fapi%2F%2F/javax/sound/sampled/AudioSystem.html

5.1 Utilización de librerías para implementación de texto similar a un placeholder en Java.

5.1.1 Detalles de los aspectos del tema que fueron necesitados en el programa.

La utilización de algunas librerías permite implementar texto sobre un JTextField, como si de un placeholder se tratara, para que de esta manera haya texto sin que se afecte el contenido que realmente sea digitado en el JTextField.

Una de las librerías/paquetes que fue necesaria para implementar lo dicho anteriormente, es la llamada: “javax.swing.event.*”, la cual “proporciona eventos activados por componentes Swing.” (Oracle, 2020). Y además tiene diversas clases de eventos e interfaces de escucha de eventos cuando un evento se activa por el Swing.

Otra de las librerías/paquetes es la llamada: “javax.swing.border.*”, que pemite dibujar bordes especializados alrededor de un componente Swing (Oracle, 2020), y la librería/paquete llamada: “java.swing.text.*” que proporciona clases e interfaces que tratan componentes de texto editables y también no editables. (Oracle, 2020).

5.1.2 Cómo se aplicó al programa.

En el proyecto se aplicó el uso de esas librerías mencionados previamente para implementar texto similar a un placeholder en los JTextField de Java, de tal manera que se pudiera ver el texto que es necesario en un juego de KenKen, ya que se necesitan ver los números y resultados junto a una operación, para que el jugador sepa que números podrían dar ese valor resultante, en una jaula dada.

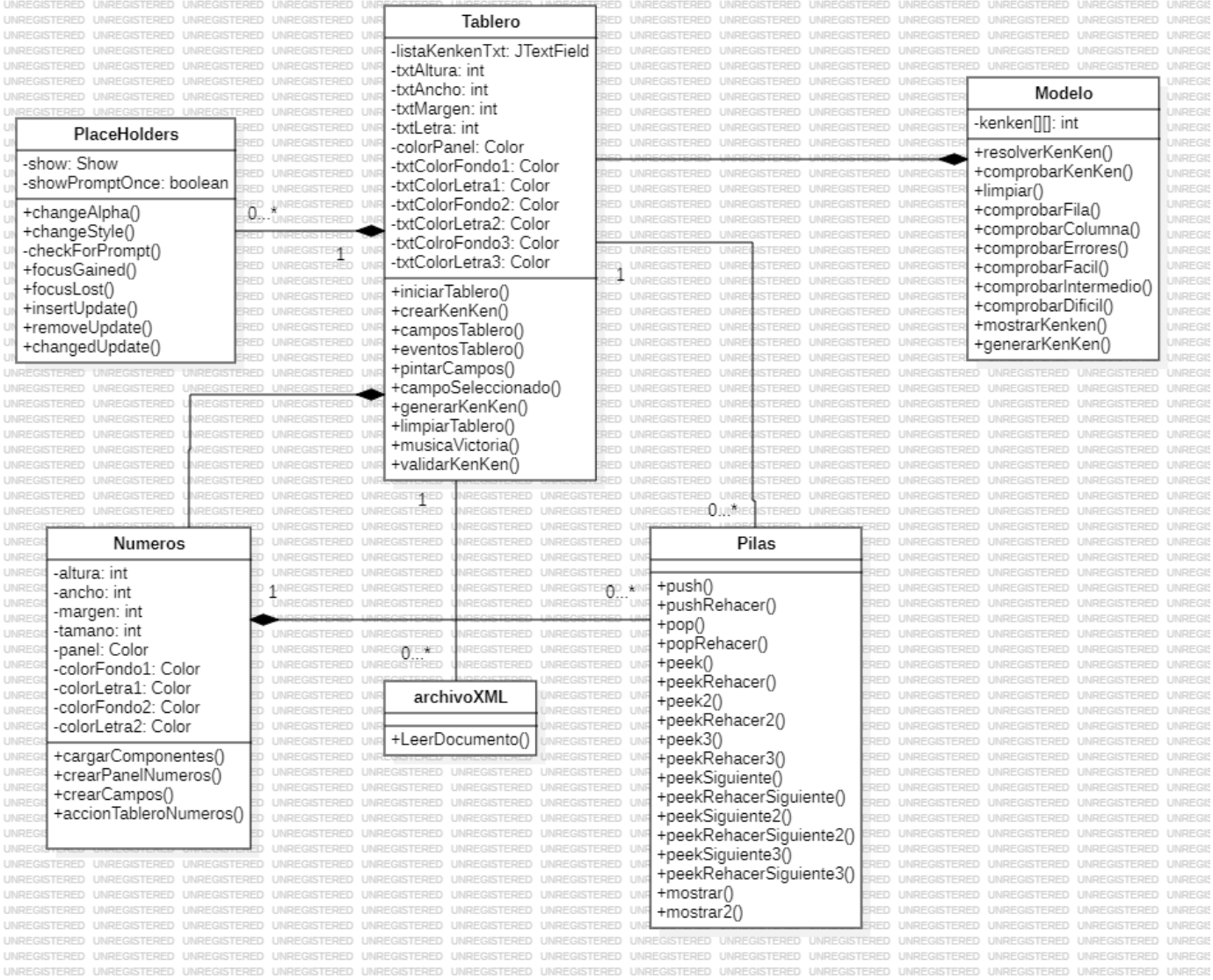
5.1.3 Bibliografía.

Package javax.swing.event.
https://docs.oracle.com/javase%2F7%2Fdocs%2Fapi%2F%2F/javax/swing/event/package-summary.html

Package javax.swing.border.
https://docs.oracle.com/javase/8/docs/api/javax/swing/border/package-summary.html

Package javax.swing.text.
https://docs.oracle.com/javase%2F7%2Fdocs%2Fapi%2F%2F/javax/swing/text/package-summary.html

Soluciones:


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()
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()
0...*
1
1
1
1
0...*
0...*
1
0...*

Conclusiones del trabajo:

Problemas encontrados y soluciones a los mismos:

En el desarrollo de todo el programa del KenKen, los principales problemas que encontré en un inicio fue sobre cómo desarrollar la cuadrícula en su totalidad, ya que no sabía cómo establecer una cuadrícula de 6x6 y que dependiendo del juego, se dividiera por jaulas, en los bordes y con los números correspondientes a las operaciones de suma, resta, multiplicación o división, entonces, después de pensarlo un rato tuve la idea de realizar la cuadrícula del kenken, a partir de un JTextField[ ][ ], llamado listaKenkenTxt, y al irlo recorriendo con dos ciclos for, pude ir agregando cada uno de los campos, indicando su ancho, alto, posición, color de fondo, color de letra y el borde inicial, así como los eventos del tablero, mediante una función, que comprobaba que acciones se realizaban con el mouse o con el teclado, cuando ya se hizo click sobre una de las cuadrículas del KenKen, y con respecto a los bordes las jaulas y los números de las operaciones, dependiendo del nivel de dificultad, lo que hice fue a partir de una clase llamada PlaceHolders, como su nombre lo indica, se iba a utilizar para colocar números a modo de placeholder, sobre los JTextField, utilizando librerías como javax.swing.border.*, javax.swing.event.*, y javax.swing.text.* para lograr este objetivo, y con respecto a los bordes de las jaulas, lo que hice fue que dependiendo de la dificultad, a cada uno de los campos necesarios, con ayuda del ".setBorder(BorderFactory.createMatteBorder()", asigne diversos tamaños de borde en los JTextField.

Después de realizar esa parte, otro problema que encontré durante la realización del programa, fue el de desarrollar el botón de "Validar juego", ya que no tenía alguna idea de cómo comprobar que el KenKen fuera correcto, según por los números ingresados por el usuario, luego de pensarlo un rato, tuve la idea de implementar una función llamada: "comprobarKenKen()", la cual, se encarga de recorrer la cuadricula e ir revisando si dependiendo de la dificultad, el número ingresado por el usuario en fila, columna del kenken tiene la respuesta correcta, al implementar otras funciones como comprobarFacil(), comprobarIntermedio(), y comprobarDifícil()", las tres funciones reciben como parámetros el número de la fila, la columna, y el número ingresado por el usuario en el Kenken, donde si esa función indicaba que había algún error al no ser igual a la variable matriz[ ][ ] que tenía la respuesta correcta, se iba a ejecutar otra función llamada: "comprobarErrores()", para pintar de color rojo cada uno de los JTextField que estaban incorrectos, y también se utilizaron dos funciones llamadas: "comprobarFila(), y comprobarColumna", las cuáles con ayuda de ciclos for evalúan que no haya números repetidos ni en las filas, ni en las columnas.

Y para finalizar, otro de los problemas que encontré durante el desarrollo del programa pero que pude solucionar, fue la implementación del cronómetro, el timer, y el sonido al ganar una partida, con respecto al cronómetro, y el timer, lo pude solucionar al implementar la librería/paquete, llamada: "javax.swing.Timer",

que es utilizada mediante ActionListener, para ir llevando el tiempo que dura una acción en completarse, o cuánto tiempo límite se tiene para realizar algo, con ayuda de un JLabel, implemente en pantalla el cronómetro, o timer, dependiendo de cuál haya elegido el usuario, y a través de variables como segundos, minutos, horas, y milisegundos, pude ir realizando funciones que asemejaran el funcionamiento de un cronómetro o timer sumando o restando el tiempo, de acuerdo al que se haya elegido, ya que cuando los milisegundos llegaban a 100, la variable de segundos aumentaba o disminuía dependiendo del caso, luego cuando los segundos llegaban a 60 o a 0 dependiendo del que se estuviera utilizando, la variable de minutos aumentaba o disminuía dependiendo del caso, y cuando los minutos llegaran a 60 o a 0 según el caso, la variable de horas aumentaba o disminuía. Y con respecto a la implementación del sonido al ganar la partida, si esta opción estaba activada, lo que hice fue que con ayuda de librerías/paquetes como javax.sound.sampled.AudioInputStream, javax.sound.sampled.AudioSystem , y javax.sound.sampled.Clip, utilice sus características en una función llamada: “musicaVictoria”, que se utilizó para ver si podía abrir el archivo .wav de música, y si erá así sonar cuando se ganara la partida, al implementarlo en la función validarKenKen().

Aprendizajes obtenidos:

En este segundo proyecto de programación orientada a objetos, tuve un gran aprendizaje, debido a que es la primera vez en la que desarrollo un juego en Java, en dónde se debían de implementar diversos, JTextField, JTextFrame Form, clases, métodos, y variables globales, y en dónde a partir de diversas investigaciones, tuve que aprender, sobre cómo usar github para el software de control de versiones y trabajo colaborativo, el manejo de archivos en formato xml, para tener un lugar dónde se almacenaran los datos de las partidas del kenken, la implementación de cronómetros y timers para llevar el tiempo que está durando una partida del kenken, o ver cuánto tiempo restante queda, el uso de archivos de audio, en este caso .wav, para su implementación y utilización en Java, al realizarse una condición en específico, en este caso, al ganar una partida, y el uso de librerías para implementar números a modo de placeholders en los JTextField.

Como conclusión, el hecho de saber que realizar cada vez más programas como estos, y saber que me ayudan bastante a incrementar mis conocimientos y lógica en la programación cada vez más, es algo que realmente me alegra, ya que la mejor forma de aprender a programar, es a partir de la misma práctica constante, y la investigación para adquirir nuevos conocimientos que van a ser de mucha utilidad en programaciones futuras, y este proyecto, se reflejó eso, ya que en diversos días le fui dedicando bastante tiempo al proyecto, y noté cada vez más, avances, dándome cuenta en el proceso, de que algunas cosas ya programadas no funcionaban del todo bien, y el ir arreglando esos errores que yo mismo iba notando durante la programación, también me fue de bastante utilidad, ya que corregir

errores es algo normal durante el desarrollo de un programa, y sirve para tratar de no seguir cometiendo esos errores e ir mejorando.

Lista de revisión del proyecto:

Concepto	Puntos	% de avance 100%/0	Puntos obtenidos	Análisis de resultados
Opción Jugar: Desplegar cuadriculado Desplegar información de cada casilla según partida Enmarcado de las jaulas según partida Desplegar el resto de la opción Algoritmo de selección de partida	 5 5 8 5 1	 100%	 5 5 8 5 1	
Iniciar Juego: Actualizar casillas (colocar y borra números) Controles del proceso	 5 5	 100%	 5 5	
Reloj en tiempo real usado en el control del juego	10	100%	10	
Validar Juego	10	100%	10	
Sonido cuando el jugador gana	5	100%	5	
Deshacer jugada	5	100%	5	
Rehacer jugada	5	80%	4	Considero que es un 80%, por lo siguiente: Lo que hace es: cuando el usuario ya ha deshecho varios números en el juego del kenken, permite rehacer o reconstruir la última jugada que se deshizo, todas las que al momento estén registradas. Lo que faltó: Que, si va a rehacer nuevamente el primer número ingresado en toda la partida, luego de haberlo deshecho, este se va a posicionar en la misma fila, pero en otra columna. No se desarrolló: debido a que no tenía en mente como

				poder solucionar eso tan específico
Otro Juego	3	100%	3	
Reiniciar Juego	2	100%	2	
Terminar Juego	1	100%	1	
Opción configurar	6	100%	6	
Aplicación patrón MVC	10	10%	1	Considero que es un 10%, por lo siguiente: Lo que hace es: se implementa un poco del patrón, pero es bastante mínimo, casi nulo. Lo que faltó: Falto la mayoría de cosas del patrón MVC, ya que realmente no pude implementar y dividir el programa exactamente en este patrón, al no incluir diversos paquetes para el modelo, la vista, y el controlador No se desarrolló: debido a que no tenía en mente como implementar exactamente el patrón MVC.
Ayuda (despliegue del PDF del manual de usuario)	1	100%	1	
Documentación del proyecto: Manual de usuario Resto de la documentación	 4 4	100%	 4 4	
TOTAL	**100**	90%	90	Es un 90% en base a la suma de los puntos obtenidos
Funcionalmente desarrolladas adicionalmente				En este caso no se desarrolló alguna parte adicional.