Estudio
Trabajo
Esfuerzo
Honestidad

PROYECTO: MANTENIMIENTO DE SOFTWARE
MEJORAS AL PROGRAMA FUTOSHIKI

DEFINICIÓN 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.).

Dentro de las metodologías de desarrollo está la etapa de mantenimiento de software.

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 diversos motivos:

- Corrección de errores
- Adaptar el software a nuevos requerimientos
- Mejoras

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

- A nivel técnico: se implementará el patrón de arquitectura MVC (Modelo/Vista/Controlador)
- A nivel funcional: se agregarán nuevas funcionalidades.

REQUERIMIENTOS DEL PROGRAMA: NUEVAS FUNCIONALIDADES

1- Tamaño de la cuadrícula

La versión actual maneja una cuadrícula de tamaño 5, esta nueva funcionalidad agrega cuadrículas de tamaño 6, 7, 8 y 9. Este cambio modifica la estructura del archivo de partidas: se le agrega el dato de cuadrícula según puede observar en la definición del archivo PARTIDAS DE JUEGO. También cambian las partes donde se trata el tamaño de la cuadrícula ya que pasa de tamaño 5 a tamaños 5, 6, 7, 8 y 9.

Para desarrollar esta funcionalidad hay que cambiar la opción de Configurar: Agregar el dato 4. Tamaño de la cuadrícula.

2- Juegos multinivel

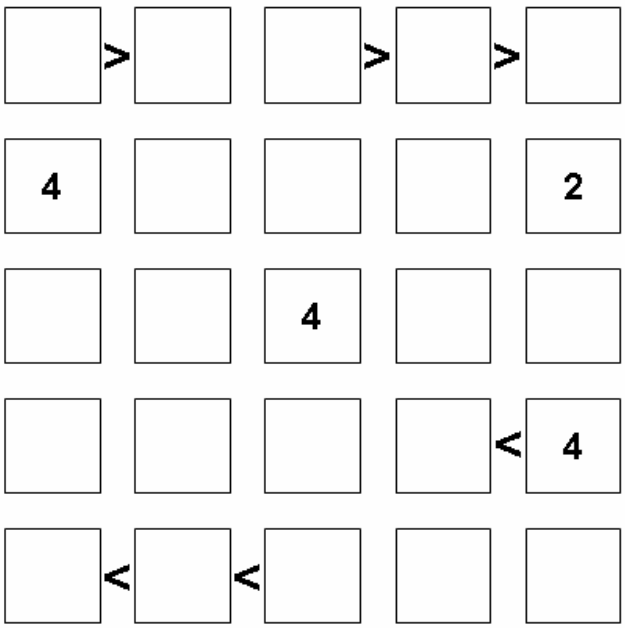
El objetivo de esta funcionalidad es que el programa pueda ir avanzado automáticamente al jugador en los niveles de juego: el jugador empieza a jugar en el nivel fácil. Cuando logre terminar exitosamente un juego en este nivel, el programa automáticamente lo envía a jugar al nivel intermedio. Cuando logre terminar exitosamente un juego en este nivel, el programa automáticamente lo envía a jugar al nivel difícil. En cada nivel el programa determinará si va al Top-10 correspondiente. Cuando termina el último nivel se queda jugando en dicho nivel.
Cuando se juega con reloj o timer, este abarca el tiempo utilizado en todos los niveles. Por ejemplo: si hay un timer de 1 hora quiere decir que esa hora es para completar todos los niveles.

Para desarrollar esta funcionalidad hay que cambiar la opción de Configurar:

- En Nivel: agregar la opción de Multinivel

3- Posibles jugadas para una casilla.

Esta funcionalidad sirve para desplegar al usuario todas las posibles jugadas correctas (dígitos) que puede hacer en ese momento para una casilla específica. Por ejemplo, si tenemos el siguiente juego y queremos saber las posibles jugadas para la casilla en la fila 3 columna 5 hay que desplegar esta información: 1, 3, 5



Los cálculos de los posibles dígitos deben ser dinámicos, es decir, se calculan en el momento considerando el juego que se tiene.
Decida usted la interfaz de usuario para implementar esta funcionalidad.

Opción Configurar

Esta opción es para indicar las condiciones con que se va a jugar. Contiene los siguientes datos que se van a guardar en el archivo "futoshiki2022configuración.dat" : (los valores por omisión –o default- están señalados con el círculo en rojo)

1. Nivel:
   - o Fácil
   - o Intermedio
   - o Difícil
   - o Multinivel

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

Horas	Minutos	Segundos
0	30	0

Restricciones de los datos para el timer: las horas pueden estar entre 0 y 2, 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.

La medición del tiempo es tiempo real.

3. Posición en la ventana del panel de dígitos:
   - o Derecha
   - o Izquierda

4. Tamaño de la cuadrícula:
   - o 5
   - o 6
   - o 7
   - o 8
   - o 9

PARTIDAS DEL JUEGO

Registre partidas de este juego en el archivo "futoshiki2022partidas.xml". Busque algunas partidas, al menos 3 por cada nivel, y grábelas directamente en el archivo, fuera del programa del juego.

Estructura del archivo con formato XML:

```
<partidasFutoshiki>
        <partida>
                <nivel>nivelDificultad</nivel>
                <cuadricula>numero</cuadricula>
                <desigualdades>tipoDesigualdad, indiceFila, indiceColumna, …
                </desigualdades>
                <constantes>constante, indiceFila, indiceColumna, …
                </constantes>
        </partida>
</partidasFutoshiki>
```

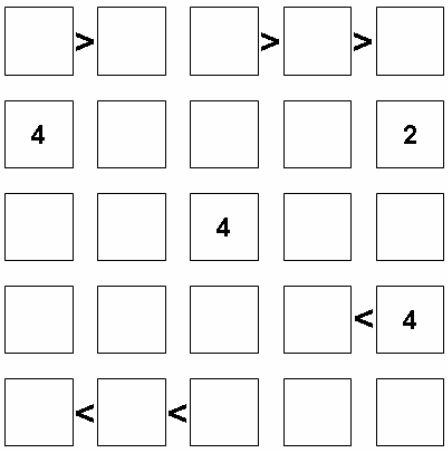
nivelDificultad: Facil, Intermedio, Dificil

numero: 5, 6, 7, 8, 9

tipoDesigualdad: a (>), b (<), y (ᵛ), z (^),
índice de fila, índice de columna: de casilla que está a la izquierda para desigualdad de fila o de casilla superior para desigualdad de columna

constante: entero entre 1 y el tamaño de la cuadrícula
índice de fila, índice de columna de la constante

Ejemplo con esta partida de nivel fácil:




```
<partidasFutoshiki>
	<partida>
		<nivel>Facil</nivel>
		<cuadricula>5</cuadricula>
		<des>a,0,0</des>
		<des>a,0,2</des>
		<des>a,0,3</des>
		<des>b,3,3</des>
		<des>b,4,0</des>
		<des>b,4,1</des>
		<cons>4,1,0</cons>
		<cons>2,1,4</cons>
		<cons>4,2,2</cons>
		<cons>4,3,4</cons>
	</partida>
</partidasFutoshiki>
```

DOCUMENTACIÓN DEL PROYECTO

Trabajo en grupos de 2 personas máximo. Se mantienen los mismos grupos que trabajaron en el proyecto 2 ya que son cambios a dicho trabajo.

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.

REQUISITOS PARA REVISAR EL PROYECTO

a- El programa debe tener documentación interna y usar JavaDoc como parte de esa documentación.
b- 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.
c- Desarrollo en Java usando GUI.
d- Opcionalmente pueden usar software de control de versiones y trabajo colaborativo (por ejemplo, github)

Enviar vía tecDigital, sección EVALUACIONES / PROGRAMAS, una carpeta comprimida (nombre **programa3_sus_nombres.zip**) que contenga la siguiente documentación (nombre **programa3_documentación_del_proyecto.PDF**) y la carpeta de desarrollo del proyecto (nombre **programa3_futoshikiV2**):

- Portada. (1P)
- Contenido de la documentación. (2P)
- Enunciado del proyecto -esta especificación-. (2P)
- Temas investigados (material no estudiado en el curso) (0P o 35P). Por cada uno de estos temas debe poner el marco teórico: de qué trata, cómo se usa. Omita la parte relacionada con el uso de interfaces gráficas. Al menos deben investigar:
  - El patrón de arquitectura MVC (Modelo/Vista/Controlador)

- Solución (0P o 20P)
  - Modelo del sistema con un diagrama de clases UML que incluya (actualizar modelo con las mejoras):
    - Atributos sus tipos de datos
    - Métodos, incluyendo setters/getters: usar nombres como setIdentificacion(String pIdentificacion, getIdentificacion(). Incluya tipos de datos, parámetros, valores de retorno.
    - Relaciones entre los objetos de las clases: dependencia, asociación, agregación, composición, herencia

        - 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 implementarán. Los constructores de cada clase deben ser con parámetros. No se permiten componentes duplicados.

  - Conclusiones del trabajo: (0P o 20P)
    - Problemas encontrados y soluciones a los mismos.
    - Aprendizajes obtenidos.
  - Lista de revisión del proyecto (PONGA ESTA LISTA EN PÁGINA NUEVA). (0P o 20P)
    - Por cada concepto de la lista de revisión usted debe indicar el % de avance que logró, 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.

LISTA DE REVISIÓN DEL PROYECTO

Concepto	Puntos	Avance 100/%/0	Puntos obtenidos	Análisis de resultados
Juegos multinivel (7 puntos por cada nivel)	18			
Posibles jugadas para una casilla	12			
Cambios: Opción Configurar Ventana del juego Grabar juego Cargar juego Archivo Partidas del juego	 10 10 5 5 5			
Aplicación correcta del patrón de arquitectura MVC	25			
Ayuda (Desplegar achivo con la documentación actualizada)	10			
TOTAL	**100**			
Partes desarrolladas adicionalmente				

Última línea